

ICN Research Group
Internet-Draft
Intended status: Experimental
Expires: May 31, 2019

C. Gundogan
TC. Schmidt
HAW Hamburg
M. Waehlich
link-lab & FU Berlin
C. Scherb
C. Marxer
C. Tschudin
University of Basel
November 27, 2018

ICN Adaptation to LowPAN Networks (ICN LowPAN)
draft-irtf-icnrg-icnlowpan-01

Abstract

In this document, a convergence layer for CCNx and NDN over IEEE 802.15.4 LowPAN networks is defined. A new frame format is specified to adapt CCNx and NDN packets to the small MTU size of IEEE 802.15.4. For that, syntactic and semantic changes to the TLV-based header formats are described. To support compatibility with other LowPAN technologies that may coexist on a wireless medium, the dispatching scheme provided by 6LoWPAN is extended to include new dispatch types for CCNx and NDN. Additionally, the link fragmentation component of the 6LoWPAN dispatching framework is applied to ICN chunks. In its second part, the document defines stateless and stateful compression schemes to improve efficiency on constrained links. Stateless compression reduces TLV expressions to static header fields for common use cases. Stateful compression schemes elide state local to the LowPAN and replace names in data packets by short local identifiers.

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 May 31, 2019.

Copyright Notice

Copyright (c) 2018 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to [BCP 78](https://trustee.ietf.org/license-info) and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document.

Table of Contents

1.	Introduction	3
2.	Terminology	4
3.	Overview of ICN LowPAN	5
3.1.	Link-Layer Convergence	5
3.2.	Stateless Header Compression	5
3.3.	Stateful Header Compression	6
4.	IEEE 802.15.4 Adaptation	7
4.1.	LowPAN Encapsulation	7
4.2.	Link Fragmentation	8
5.	Space-efficient Message Encoding for NDN	9
5.1.	TLV Encoding	9
5.2.	Name TLV Compression	10
5.3.	Interest Messages	11
5.4.	Data Messages	14
6.	Space-efficient Message Encoding for CCNx	16
6.1.	TLV Encoding	16
6.2.	Name TLV Compression	17
6.3.	Interest Messages	17
6.4.	Content Objects	23
7.	Stateful Header Compression	26
7.1.	LowPAN-local State	26
7.2.	En-route State	27
7.3.	Integrating Stateful Header Compression	29
8.	Implementations	29
9.	Security Considerations	29
10.	IANA Considerations	30
10.1.	Page Switch Dispatch Type	30
11.	References	30
11.1.	Normative References	30
11.2.	Informative References	30
Appendix A.	Estimated Size Reduction	33
A.1.	NDN	33

A.1.1.	Interest	33
A.1.2.	Data	34
A.2.	CCNx	36
A.2.1.	Interest	36
A.2.2.	Content Object	37
	Acknowledgments	38
	Authors' Addresses	38

[1.](#) Introduction

The Internet of Things (IoT) has been identified as a promising deployment area for Information Centric Networks (ICN), as infrastructureless access to content, resilient forwarding, and in-network data replication demonstrated notable advantages over the traditional host-to-host approach on the Internet [[NDN-EXP1](#)], [[NDN-EXP2](#)]. Recent studies [[NDN-MAC](#)] have shown that an appropriate mapping to link layer technologies has a large impact on the practical performance of an ICN. This will be even more relevant in the context of IoT communication where nodes often exchange messages via low-power wireless links under lossy conditions. In this memo, we address the base adaptation of data chunks to such link layers for the ICN flavors NDN [[NDN](#)] and CCNx.

The IEEE 802.15.4 [[ieee802.15.4](#)] link layer is used in low-power and lossy networks (see "LLN" in [[RFC7228](#)]), in which devices are typically battery-operated and constrained in resources. Characteristics of LLNs include an unreliable environment, low bandwidth transmissions, and increased latencies. IEEE 802.15.4 admits a maximum physical layer packet size of 127 octets. The maximum frame header size is 25 octets, which leaves 102 octets for the payload. IEEE 802.15.4 security features further reduce this payload length by up to 21 octets, yielding a net of 81 octets for CCNx or NDN packet headers, signatures and content.

6LoWPAN [[RFC4944](#)], [[RFC6282](#)] is a convergence layer that provides frame formats, header compression and link fragmentation for IPv6 packets in IEEE 802.15.4 networks. The 6LoWPAN adaptation introduces a dispatching framework that prepends further information to 6LoWPAN packets, including a protocol identifier for IEEE 802.15.4 payload and meta information about link fragmentation.

Prevalent Type-Length-Value (TLV) based packet formats such as in CCNx and NDN are designed to be generic and extensible. This leads to header verbosity which is inappropriate in constrained environments of IEEE 802.15.4 links. This document presents ICN LowPAN, a convergence layer for IEEE 802.15.4 motivated by 6LoWPAN that compresses packet headers of CCNx as well as NDN and allows for an increased payload size per packet. Additionally, reusing the

This document uses the terminology of [RFC7476], [RFC7927], and [RFC7945] for ICN entities.

The following terms are used in the document and defined as follows:

ICN LowPAN: Information-Centric Networking over Low-power Wireless Personal Area Network

LLN Low-Power and Lossy Network

CCNx: Content-Centric Networking Architecture

NDN: Named Data Networking Architecture

3. Overview of ICN LowPAN

3.1. Link-Layer Convergence

ICN LowPAN provides a convergence layer that maps ICN packets onto constrained link-layer technologies. This includes features such as link-layer fragmentation, protocol separation on the link-layer level, and link-layer address mappings. The stack traversal is visualized in Figure 2.

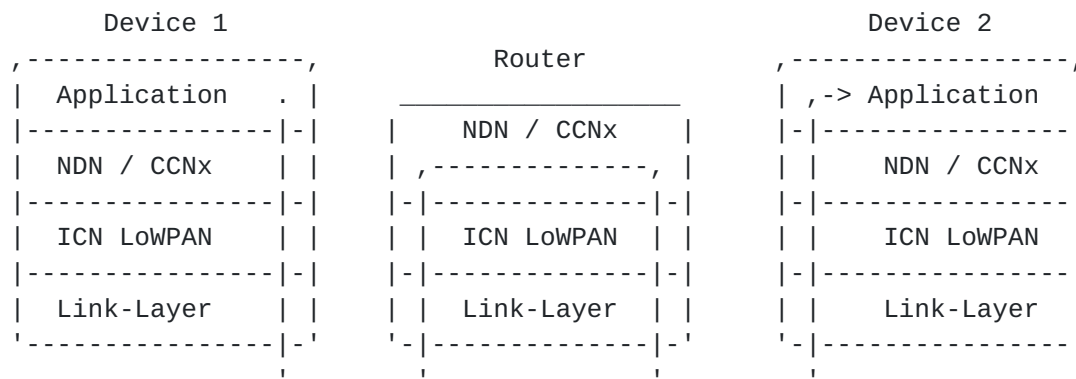


Figure 2: ICN LowPAN convergence layer for IEEE 802.15.4

[Section 4](#) of this document defines the convergence layer for IEEE 802.15.4.

3.2. Stateless Header Compression

ICN LowPAN also defines a stateless header compression scheme with the main purpose of reducing header overhead of ICN packets. This is of particular importance for link-layers with small MTUs. The stateless compression does not require pre-configuration of global state.

The CCNx and NDN header formats are composed of Type-Length-Value (TLV) fields to encode header data. The advantage of TLVs is its

native support of variable-sized data. The main disadvantage of TLVs is the verbosity that results from storing the type and length of the encoded data.

The stateless header compression scheme makes use of compact bit fields to indicate the presence of mandatory and optional TLVs in the uncompressed packet. The order of set bits in the bit fields corresponds to the order of each TLV in the packet. Further compression is achieved by specifying default values and reducing the codomain of certain header fields.

Figure 3 demonstrates the stateless header compression idea. In this example, the first type of the first TLV is removed and the corresponding bit in the bit field is set. The second TLV represents a fixed-length TLV (e.g., the Nonce TLV in NDN), so that the type and the length fields are removed. The third TLV represents a boolean TLV (e.g., the MustBeFresh selector in NDN) and is missing the type, length and the value field.

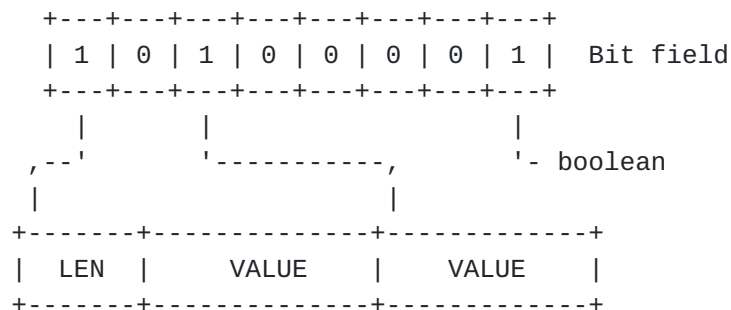


Figure 3: Compression using a compact bit field to encode context information.

Stateless TLV compression for NDN is defined in [Section 5](#). [Section 6](#) defines the stateless TLV compression for CCNx.

3.3. Stateful Header Compression

ICN LowPAN further employs two orthogonal stateful compression schemes for packet size reductions which are defined in [Section 7](#). These mechanisms rely on shared contexts that are either distributed and maintained in the entire LowPAN, or are generated on-demand hop-wise on a particular Interest-data path.

The shared context identification is defined in [Section 7.1](#). The hop-wise name compression "en-route" is specified in [Section 7.2](#).

4. IEEE 802.15.4 Adaptation

4.1. LowPAN Encapsulation

The IEEE 802.15.4 frame header does not provide a protocol identifier for its payload. This causes problems of misinterpreting frames when several network layers coexist on the same link. To mitigate errors, 6LoWPAN defines dispatches as encapsulation headers for IEEE 802.15.4 frames (see [Section 5 of \[RFC4944\]](#)). Multiple LowPAN encapsulation headers can prepend the actual payload and each encapsulation header is identified by a dispatch type.

[RFC8025] further specifies dispatch pages to switch between different contexts. When a LowPAN parser encounters a "Page switch" LowPAN encapsulation header, then all following encapsulation headers are interpreted by using a dispatch table as specified by the "Page switch" header. Page 0 and page 1 are reserved for 6LoWPAN. This document uses page 2 ("1111 0010 (0xF2)") for NDN and page 3 ("1111 0011 (0xF3)") for CCNx.

The base dispatch format (Figure 4) is used and extended by CCNx and NDN in [Section 5](#) and [Section 6](#).

```

      0   1   2   ...
      +---+---+-----
      | C | M |
      +---+---+-----

```

Figure 4: Base dispatch format for ICN LowPAN

C: Compression

0: The message is uncompressed.

1: The message is compressed.

M: Message Type

0: The payload contains an Interest message.

1: The payload contains a Data message.

ICN LowPAN frames with compressed CCNx and NDN messages (C=1) use the extended dispatch format in Figure 5.


```

      0   1   2   ...
    +---+---+-----
    | 1 | M |CID|
    +---+---+-----

```

Figure 5: Extended dispatch format for compressed ICN LowPAN

CID: Context Identifier

0: No context identifiers are present.

1: 1..n context identifiers are present.

The encapsulation format for ICN LowPAN is displayed in Figure 6.

```

+-----+-----+-----+-----+-----+-----+-----+
| IEEE 802.15.4 | RFC4944 Disp. | Page | ICN LowPAN Disp. | Payl. /
+-----+-----+-----+-----+-----+-----+-----+

```

Figure 6: LowPAN Encapsulation with ICN-LowPAN

IEEE 802.15.4: The IEEE 802.15.4 header.

[RFC4944](#) Disp.: Optional additional dispatches defined in [Section 5.1 of \[RFC4944\]](#)

Page: Page Switch. 2 for NDN and 3 for CCNx.

ICN LowPAN: Dispatches defined in [Section 5](#) and [Section 6](#).

Payload: The actual (un-)compressed CCNx or NDN message.

4.2. Link Fragmentation

Small payload sizes in the LowPAN require fragmentation for various network layers. Therefore, [Section 5.3 of \[RFC4944\]](#) defines a protocol-independent fragmentation dispatch type, a fragmentation header for the first fragment, and a separate fragmentation header for subsequent fragments. ICN LowPAN adopts this fragmentation handling of [\[RFC4944\]](#).

The Fragmentation LowPAN header can encapsulate other dispatch headers. The order of dispatch types is defined in [Section 5 of \[RFC4944\]](#). Figure 7 shows the fragmentation scheme. The reassembled ICN LowPAN frame does not contain any fragmentation headers and is depicted in Figure 8.

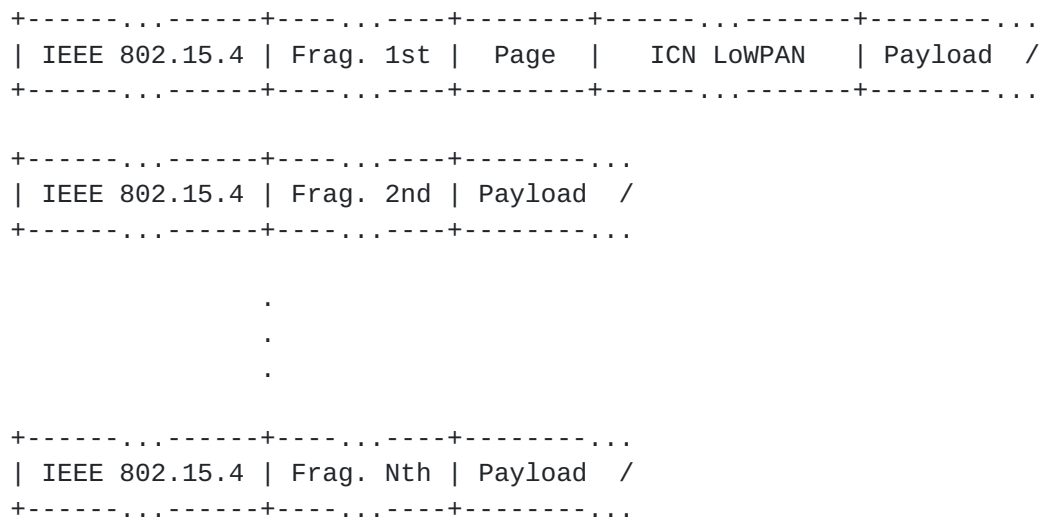


Figure 7: Fragmentation scheme

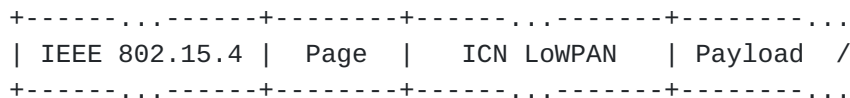


Figure 8: Reassembled ICN LowPAN frame

5. Space-efficient Message Encoding for NDN

5.1. TLV Encoding

The NDN packet format consists of TLV fields using the TLV encoding that is described in [\[NDN-PACKET-SPEC\]](#). Type and length fields are of variable size, where numbers greater than 252 are encoded using multiple octets.

If the type or length number is less than "253", then that number is encoded into the actual type or length field. If the number is greater or equals "253" and fits into 2 octets, then the type or length field is set to "253" and the number is encoded in the next following 2 octets in network byte order, i.e., from the most significant byte (MSB) to the least significant byte (LSB). If the number is greater than 2 octets and fits into 4 octets, then the type or length field is set to "254" and the number is encoded in the subsequent 4 octets in network byte order. For larger numbers, the type or length field is set to "255" and the number is encoded in the subsequent 8 octets in network byte order.

In this specification, compressed NDN TLVs make use of a different TLV encoding scheme that reduces size. Instead of using the first octet as a marker for the number of following octets, the compressed

NDN TLV scheme uses a method to chain a variable number of octets together. If an octet equals "255 (0xFF)", then the following octet will also be interpreted. The actual value of a chain equals the sum of all links.

If the type or length number is less than "255", then that number is encoded into the actual type or length field (Figure 9 a). If the type or length number (X) fits into 2 octets, then the first octet is set to "255" and the subsequent octet equals "X mod 255" (Figure 9 b). Following this scheme, a variable-sized number (X) is encoded using multiple octets of "255" with a trailing octet containing "X mod 255" (Figure 9 c).

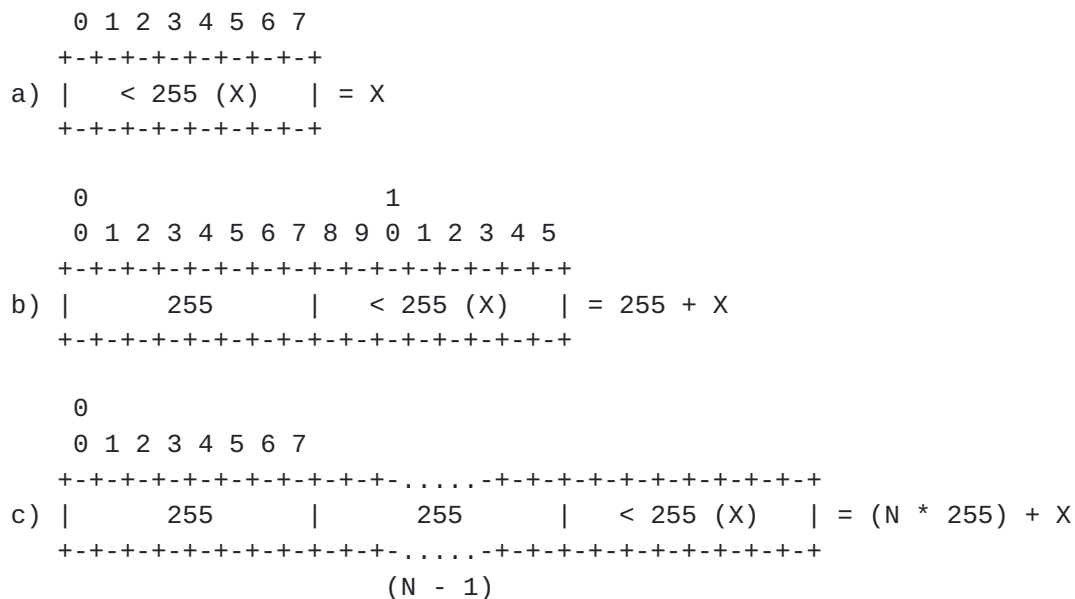


Figure 9: Compressed NDN TLV encoding scheme

5.2. Name TLV Compression

This Name TLV compression encodes length fields of two consecutive NameComponent TLVs into one octet, using 4 bits each. This process limits the length of a NameComponent TLV to 15 octets. A length of 0 marks the end of the compressed Name TLV.

Name: /HAW/Room/481/Humid/99

```

      0          1          2          3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
|0 0 1 1|0 1 0 0|          H          |          A          |          W          |
+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
|          R          |          o          |          o          |          m          |
+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
|0 0 1 1|0 1 0 1|          4          |          8          |          1          |
+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
|          H          |          u          |          m          |          i          |
+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
|          d          |0 0 1 0|0 0 0 0|          9          |          9          |
+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+

```

Figure 10: Name TLV compression for /HAW/Room/481/Humid/99

5.3. Interest Messages

5.3.1. Uncompressed Interest Messages

An uncompressed Interest message uses the base dispatch format (see Figure 4) and sets the C as well as the M flag to "0" (Figure 11). "resv" MUST be set to 0. The Interest message is handed to the NDN network stack without modifications.

```

      0      1      ...      7
+--+--+--+-----+
| 0 | 0 |          resv          |
+--+--+--+-----+

```

Figure 11: Dispatch format for uncompressed NDN Interest messages

5.3.2. Compressed Interest Messages

The compressed Interest message uses the extended dispatch format (Figure 5) and sets the C flag to "1" and the M flag to "0". In the default use case, the Interest message is compressed with the following minimal rule set:

1. The "Type" field of the outermost MessageType TLV is removed.
2. The Name TLV is compressed according to [Section 5.2](#). For this, all NameComponents are expected to be of type GenericNameComponent. Otherwise, the message MUST be sent uncompressed.

3. The InterestLifetime TLV length is set to 2. Messages with lifetimes that require more than 2 octets MUST be sent uncompressed.
4. The Nonce TLV, InterestLifetime TLV and HopLimit TLV MUST be moved to the end of the compressed Interest, keeping the order 1) Nonce TLV, 2) InterestLifetime TLV and 3) HopLimit TLV.
5. The Type and Length fields of Nonce TLV, InterestLifetime TLV and HopLimit TLV are elided. The presence of each TLV is deduced from the remaining length to parse. The Nonce TLV has a fixed length of 4, the InterestLifetime TLV has a fixed length of 2 and the HopLimit TLV has a fixed length of 1. Any combination yields a distinct value that matches the remaining length to parse.

The compressed NDN LowPAN Interest message is visualized in Figure 12.

T = Type, L = Length, V = Value

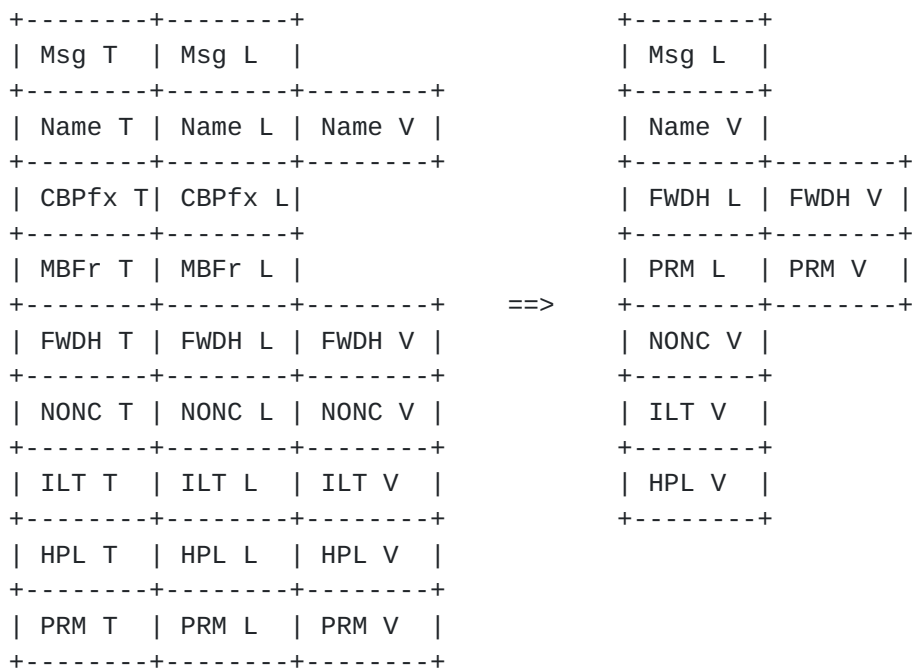


Figure 12: Compressed NDN LowPAN Interest Message

Further TLV compression is indicated by the ICN LowPAN dispatch in Figure 13.


```

      0   1   2   3   4   5   6   7
+---+---+---+---+---+---+---+---+
| 1 | 0 |CID|DIG|PFX|FRE|FWD|PRM|
+---+---+---+---+---+---+---+---+

```

Figure 13: Dispatch format for compressed NDN Interest messages

CID: Context Identifier See Figure 5.

DIG: ImplicitSha256DigestComponent TLV

- 0: The name does not include an ImplicitSha256DigestComponent as the last TLV.
- 1: The name does include an ImplicitSha256DigestComponent as the last TLV. The Type and Length fields are omitted.

PFX: CanBePrefix TLV

- 0: The uncompressed message does not include a CanBePrefix TLV.
- 1: The uncompressed message does include a CanBePrefix TLV and is removed from the compressed message.

FRE: MustBeFresh TLV

- 0: The uncompressed message does not include a MustBeFresh TLV.
- 1: The uncompressed message does include a MustBeFresh TLV and is removed from the compressed message.

FWD: ForwardingHint TLV

- 0: The uncompressed message does not include a ForwardingHint TLV.
- 1: The uncompressed message does include a ForwardingHint TLV. The Type field is removed from the compressed message.

PRM: Parameters TLV

- 0: The uncompressed message does not include a Parameters TLV.

- 1: The uncompressed message does include a Parameters TLV. The Type field is removed from the compressed message.

5.4. Data Messages

5.4.1. Uncompressed Data Messages

An uncompressed Data message uses the base dispatch format and sets the C flag to "0" and the M flag to "1" (Figure 14). "resv" MUST be set to 0. The Data message is handed to the NDN network stack without modifications.

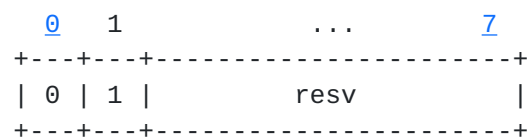


Figure 14: Dispatch format for uncompressed NDN Data messages

5.4.2. Compressed Data Messages

The compressed Data message uses the extended dispatch format (Figure 5) and sets the C flag as well as the M flag to "1". By default, the Data message is compressed with the following base rule set:

1. The "Type" field of the outermost MessageType TLV is removed.
2. The Name TLV is compressed according to [Section 5.2](#). For this, all NameComponents are expected to be of type GenericNameComponent. Otherwise, the message MUST be sent uncompressed.
3. The MetaInfo Type and Length fields are elided from the compressed Data message.
4. If present, the FinalBlockId TLV is encoded according to [Section 5.2](#).
5. The ContentType TLV length is set to 1. Messages with ContentTypes that require more than 1 octet MUST be sent uncompressed.
6. The FreshnessPeriod TLV length is set to 2. Messages with FreshnessPeriods that require more than 2 octets MUST be sent uncompressed.

7. The FreshnessPeriod TLV and ContntType TLV MUST be moved to the end of the compressed Data, keeping the order 1) FreshnessPeriod TLV and 2) ContentType TLV.
8. The Type and Length fields of ContentType TLV and FreshnessPeriod TLV are elided. The presence of each TLV is deduced from the remaining length to parse. The FreshnessPeriod TLV has a fixed length of 2 and the ContentType TLV has a fixed length of 1. Any combination yields a distinct value that matches the remaining length to parse.

The compressed NDN LowPAN Data message is visualized in Figure 15.

T = Type, L = Length, V = Value

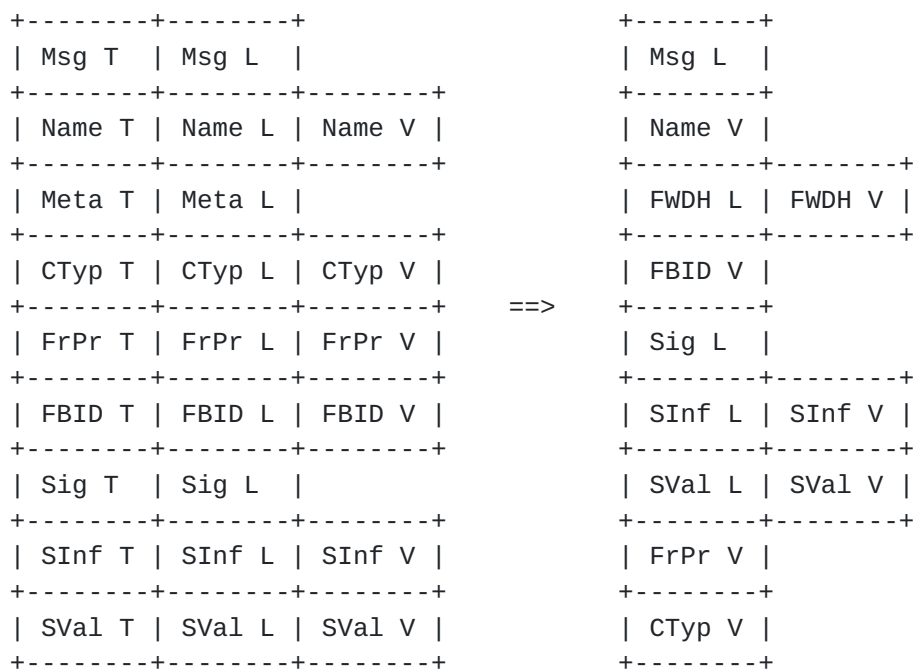


Figure 15: Compressed NDN LowPAN Data Message

Further TLV compression is indicated by the ICN LowPAN dispatch in Figure 16.

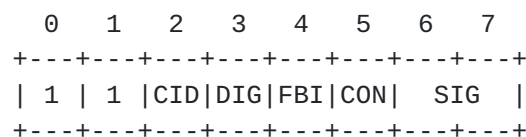


Figure 16: Dispatch format for compressed NDN Data messages

CID: Context Identifier See Figure 5.

DIG: ImplicitSha256DigestComponent TLV

- 0: The name does not include an ImplicitSha256DigestComponent as the last TLV.
- 1: The name does include an ImplicitSha256DigestComponent as the last TLV. The Type and Length fields are omitted.

FBI: FinalBlockId TLV

- 0: The uncompressed message does not include a FinalBlockId TLV.
- 1: The uncompressed message does include a FinalBlockId.

CON: Content TLV

- 0: The uncompressed message does not include a Content TLV.
- 1: The uncompressed message does include a Content TLV. The Type field is removed from the compressed message.

SIG: Signature TLV

- 00: The Type fields of the SignatureInfo TLV, SignatureType TLV and SignatureValue TLV are removed.
- 01: Reserved.
- 10: Reserved.
- 11: Reserved.

6. Space-efficient Message Encoding for CCNx**6.1. TLV Encoding**

The generic CCNx TLV encoding is described in [\[I-D.irtf-icnrg-ccnxmessages\]](#). Type and Length fields attain the common fixed length of 2 octets.

The TLV encoding for CCNx LowPAN is changed to the more space efficient encoding described in [Section 5.1](#). Hence NDN and CCNx use the same compressed format for writing TLVs.

6.2. Name TLV Compression

Name TLVs are compressed using the scheme already defined in [Section 5.2](#) for NDN. If a Name TLV contains T_IPID, T_APP, or organizational TLVs, then the name remains uncompressed.

6.3. Interest Messages

6.3.1. Uncompressed Interest Messages

An uncompressed Interest message uses the base dispatch format (see Figure 4) and sets the C as well as the M flag to "0" (Figure 17). "resv" MUST be set to 0. The Interest message is handed to the CCNx network stack without modifications.

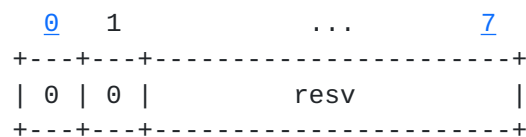


Figure 17: Dispatch format for uncompressed CCNx Interest messages

6.3.2. Compressed Interest Messages

The compressed Interest message uses the extended dispatch format (Figure 5) and sets the C flag to "1" and the M flag to "0". In the default use case, the Interest message is compressed with the following minimal rule set:

1. The Type and Length fields of the CCNx Message TLV are elided and are obtained from the Fixed Header on decompression.

The compressed CCNx LowPAN Interest message is visualized in Figure 18.

FLG: Flags field in the fixed header

- 0: The Flags field equals 0 and is removed from the Interest message.
- 1: The Flags field is carried in-line.

PTY: PacketType field in the fixed header

- 0: The PacketType field is elided and assumed to be "PT_INTEREST"
- 1: The PacketType field is elided and assumed to be "PT_RETURN"

HPL: HopLimit field in the fixed header

- 0: The HopLimit field is carried in-line
- 1: The HopLimit field is elided and assumed to be "1"

FRS: Reserved field in the fixed header

- 0: The Reserved field is carried in-line
- 1: The Reserved field is elided and assumed to be "0"

PAY: Optional Payload TLV

- 0: The Payload TLV is absent.
- 1: The Payload TLV is present and the type field is elided.

ILT: Optional Hop-By-Hop InterestLifetime TLV

See [Section 6.3.2.1](#) for further details on the ordering of hop-by-hop TLVs.

- 0: No InterestLifetime TLV is present in the Interest message.
- 1: An InterestLifetime TLV is present with a fixed length of 2 octets. The type and length fields are elided.

MGH: Optional Hop-By-Hop MessageHash TLV

See [Section 6.3.2.1](#) for further details on the ordering of hop-by-hop TLVs.

This TLV is expected to contain a T_SHA-256 TLV. If another hash is contained, then the Interest MUST be sent uncompressed.

- 0: The MessageHash TLV is absent.
- 1: A T_SHA-256 TLV is present and the type as well as the length fields are removed. The length field is assumed to represent 32 octets. The outer Message Hash TLV is omitted.

KIR: Optional KeyIdRestriction TLV

This TLV is expected to contain a T_SHA-256 TLV. If another hash is contained, then the Interest MUST be sent uncompressed.

- 0: The KeyIDRestriction TLV is absent.
- 1: A T_SHA-256 TLV is present and the type as well as the length fields are removed. The length field is assumed to represent 32 octets. The outer KeyIdRestriction TLV is omitted.

CHR: Optional ContentObjectHashRestriction TLV

This TLV is expected to contain a T_SHA-256 TLV. If another hash is contained, then the Interest MUST be sent uncompressed.

- 0: The ContentObjectHashRestriction TLV is absent.
- 1: A T_SHA-256 TLV is present and the type as well as the length fields are removed. The length field is assumed to represent 32 octets. The outer ContentObjectHashRestriction TLV is omitted.

VAL: Optional ValidationAlgorithm and ValidationPayload TLVs

- 0: No validation related TLVs are present in the Interest message.
- 1: Validation related TLVs are present in the Interest message. An additional octet follows immediately that handles validation related TLV compressions and is described in [Section 6.3.2.2](#).

EXT: Extension

- 0: No extension octet follows.
- 1: An extension octet follows immediately. Extension octets are used to extend the compression scheme, but are out of scope of this document.

RSV: Reserved Must be set to 0.

6.3.2.1. Hop-By-Hop Header TLVs Compression

Hop-By-Hop Header TLVs are unordered. For an Interest message, two optional Hop-By-Hop Header TLVs are defined in [\[I-D.irtf-icnrg-ccnxmessages\]](#), but several more can be defined in higher level specifications. For a compressed representation, this document defines the following ordering of Hop-By-Hop TLVs:

1. Interest Lifetime TLV
2. Message Hash TLV

Note: If the original Interest message includes Hop-By-Hop Header TLVs that follow a different ordering, then the message **MUST** be sent uncompressed.

6.3.2.2. Validation

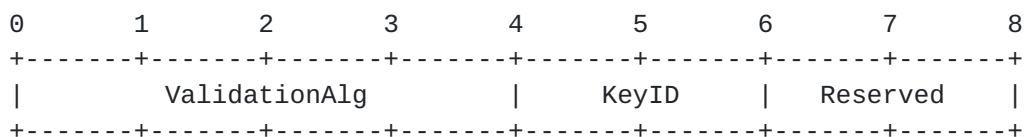


Figure 20: Dispatch for Interest Validations

ValidationAlg: Optional ValidationAlgorithm TLV

- 0000: An uncompressed ValidationAlgorithm TLV is included.
- 0001: A T_CRC32C ValidationAlgorithm TLV is assumed, but no ValidationAlgorithm TLV is included.
- 0010: A T_CRC32C ValidationAlgorithm TLV is assumed, but no ValidationAlgorithm TLV is included. Additionally, a Sigtime TLV is inlined without a type and a length field.
- 0011: A T_HMAC-SHA256 ValidationAlgorithm TLV is assumed, but no ValidationAlgorithm TLV is included.

- 0100: A T_HMAC-SHA256 ValidationAlgorithm TLV is assumed, but no ValidationAlgorithm TLV is included. Additionally, a Sigtime TLV is inlined without a type and a length field.
- 0101: Reserved.
- 0110: Reserved.
- 0111: Reserved.
- 1000: Reserved.
- 1001: Reserved.
- 1010: Reserved.
- 1011: Reserved.
- 1100: Reserved.
- 1101: Reserved.
- 1110: Reserved.
- 1111: Reserved.

KeyID: Optional KeyID TLV within the ValidationAlgorithm TLV

- 00: The KeyId TLV is absent.
- 01: The KeyId TLV is present and uncompressed.
- 10: A T_SHA-256 TLV is present and the type field as well as the length fields are removed. The length field is assumed to represent 32 octets. The outer KeyId TLV is omitted.
- 11: A T_SHA-512 TLV is present and the type field as well as the length fields are removed. The length field is assumed to represent 64 octets. The outer KeyId TLV is omitted.

The ValidationPayload TLV is present if the ValidationAlgorithm TLV is present. The type field is omitted.

6.4. Content Objects

6.4.1. Uncompressed Content Objects

An uncompressed Content object uses the base dispatch format (see Figure 4) and sets the C flag to "0" and the M flag to "1" (Figure 21). "resv" MUST be set to 0. The Content object is handed to the CCNx network stack without modifications.

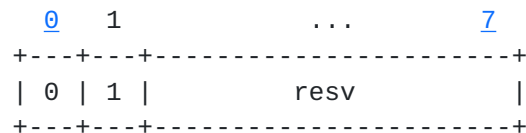


Figure 21: Dispatch format for uncompressed CCNx Content objects

6.4.2. Compressed Content Objects

The compressed Content object uses the extended dispatch format (Figure 5) and sets the C flag as well as the M flag to "1". By default, the Content object is compressed with the following base rule set:

1. The PacketType field is elided from the Fixed Header.
2. The Type and Length fields of the CCNx Message TLV are elided and are obtained from the Fixed Header on decompression.

The compressed CCNx LowPAN Data message is visualized in Figure 22.

T = Type, L = Length, V = Value

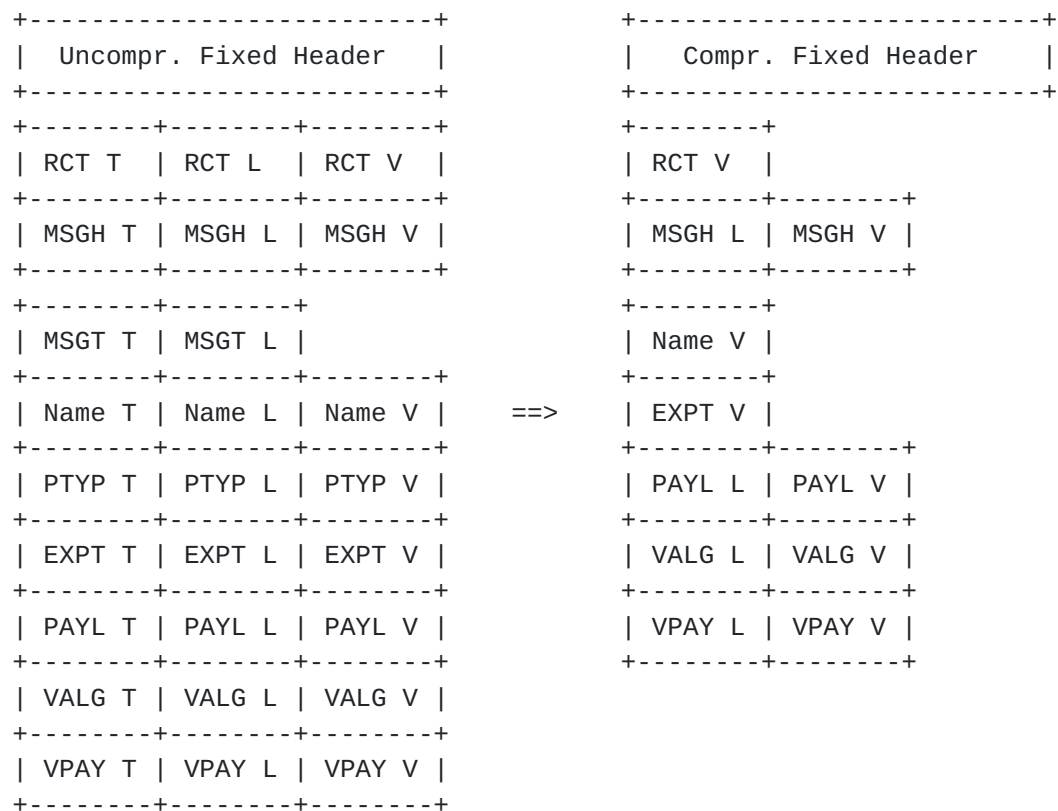


Figure 22: Compressed CCNx LowPAN Data Message

Further TLV compression is indicated by the ICN LowPAN dispatch in Figure 23.

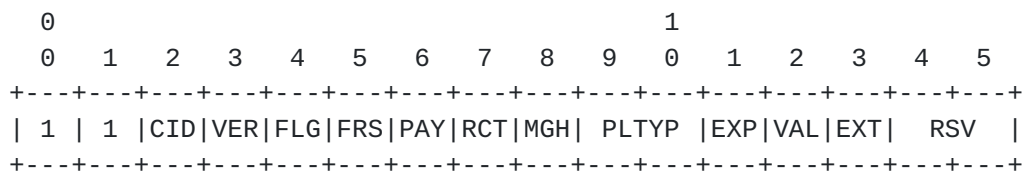


Figure 23: Dispatch format for compressed CCNx Content objects

CID: Context Identifier See Figure 5.

VER: CCNx protocol version in the fixed header

- 0: The Version field equals 1 and is removed from the fixed header.
- 1: The Version field is carried in-line.

FLG: Flags field in the fixed header See [Section 6.3.2](#).

FRS: Reserved field in the fixed header See [Section 6.3.2](#).

PAY: Optional Payload TLV See [Section 6.3.2](#).

RCT: Optional Hop-By-Hop RecommendedCacheTime TLV

- 0: The Recommended Cache Time TLV is absent.
- 1: The Recommended Cache Time TLV is present and the type as well as the length fields are elided.

MGH: Optional Hop-By-Hop MessageHash TLV

See [Section 6.4.2.1](#) for further details on the ordering of hop-by-hop TLVs.

This TLV is expected to contain a T_SHA-256 TLV. If another hash is contained, then the Content Object MUST be sent uncompressed.

- 0: The MessageHash TLV is absent.
- 1: A T_SHA-256 TLV is present and the type as well as the length fields are removed. The length field is assumed to represent 32 octets. The outer Message Hash TLV is omitted.

PLTYP: Optional PayloadType TLV

- 00: The PayloadType TLV is absent.
- 01: The PayloadType TLV is absent and T_PAYLOADTYPE_DATA is assumed.
- 10: The PayloadType TLV is absent and T_PAYLOADTYPE_KEY is assumed.
- 11: The PayloadType TLV is present and uncompressed.

EXP: Optional ExpiryTime TLV

- 0: The ExpiryTime TLV is absent.

- 1: The ExpiryTime TLV is present and the type as well as the length fields are elided.

RSV: Reserved Must be set to 0.

VAL: Optional ValidationAlgorithm and ValidationPayload TLVs See Section 6.3.2.

EXT: Extension See [Section 6.3.2](#).

[6.4.2.1](#). Hop-By-Hop Header TLVs Compression

Hop-By-Hop Header TLVs are unordered. For a Content Object message, two optional Hop-By-Hop Header TLVs are defined in [\[I-D.irtf-icnrg-ccnxmessages\]](#), but several more can be defined in higher level specifications. For better compression, an ordering of Hop-By-Hop TLVs is required as follows:

1. Recommended Cache Time TLV
2. Message Hash TLV

With this ordering in place, Type fields are elided from the Recommended Cache Time TLV and Message Hash TLV.

Note: If the original Content Object message includes Hop-By-Hop Header TLVs with a different ordering, then they remain uncompressed.

[7](#). Stateful Header Compression

Stateful header compression in ICN LowPAN enables packet size reductions in two ways. First, common information that is shared throughout the local LowPAN may be memorized in context state at all nodes and omitted from communication. Second, redundancy in a single Interest-data exchange may be removed from ICN stateful forwarding on a hop-by-hop bases and memorized in en-route state tables.

[7.1](#). LowPAN-local State

A context identifier (CID) is an octet that refers to a particular conceptual context between network devices and MAY be used to replace frequently appearing information, like name prefixes, suffixes, or meta information, such as Interest lifetime.

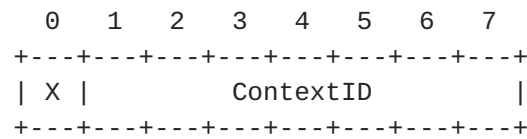


Figure 24: Context Identifier.

The ContextID refers to a locally-scoped unique identifier that represents contextual state shared between sender and receiver of the corresponding frame (see Figure 24).

TODO: Examples? Correspondance to 6Lo?

The initial distribution and maintenance of shared context is out of scope of this document. Frames containing unknown or invalid CIDs MUST be silently discarded.

7.2. En-route State

In CCNx and NDN, Name TLVs are included in Interest messages, and they return in data messages. Returning Name TLVs either equal the original Name TLV, or they contain the original Name TLV as a prefix. ICN LowPAN reduces this redundancy in responses by replacing Name TLVs with single octets that represent link-local HopIDs. HopIDs are carried as Context Identifiers of link-local scope as shown in Figure 25.

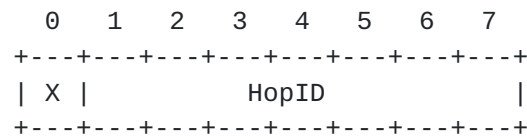


Figure 25: Context Identifier as HopID.

A HopID is valid, if not all ID bits are set to zero and invalid otherwise. This yields 127 distinct HopIDs. If this range (1...128) is exhausted, the messages MUST be sent without en-route state compression until new HopIDs are available. An ICN LowPAN node that forwards without replacing the name by a HopID (without en-route compression) MUST invalidate the HopID by setting all ID-bits to zero.

While an Interest is traversing, a forwarder generates an ephemeral HopID that is tied to a PIT entry. Each HopID MUST be unique within the local PIT and only exists during the lifetime of a PIT entry. To maintain HopIDs, the local PIT is extended by two new columns: HIDi (inbound HopIDs) and HIdo (outbound HopIDs).

HopIDs are included in Interests and stored on the next hop with the resulting PIT entry in the HIDi column. The HopID is replaced with a newly generated local HopID before the Interest is forwarded. This new HopID is stored in the HIDo column of the local PIT (see Figure 26).

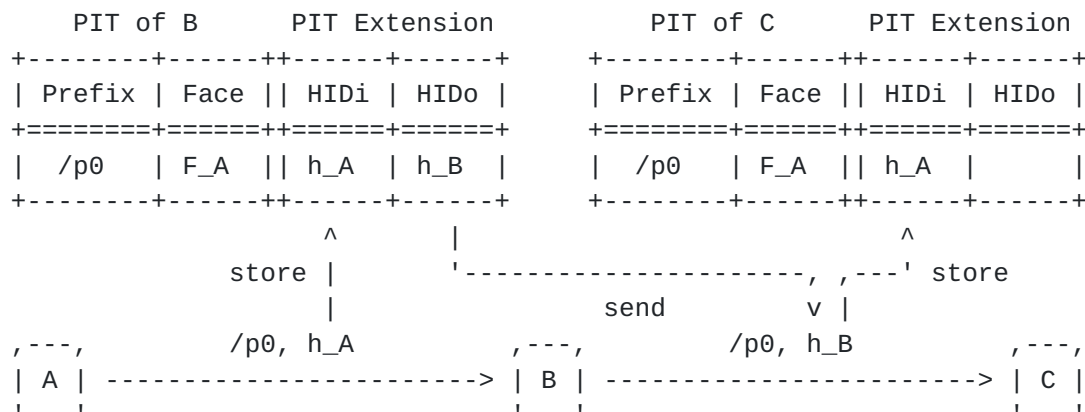


Figure 26: Setting compression state en-route (Interest).

Responses include HopIDs that were obtained from Interests. If the returning Name TLV equals the original Name TLV, then the name is entirely elided. Otherwise, the distinct suffix is included along with the HopID. When a response is forwarded, the contained HopID is extracted and used to match against the correct PIT entry by performing a lookup on the HIDo column. The HopID is then replaced with the corresponding HopID from the HIDi column prior to forwarding the response (Figure 27).

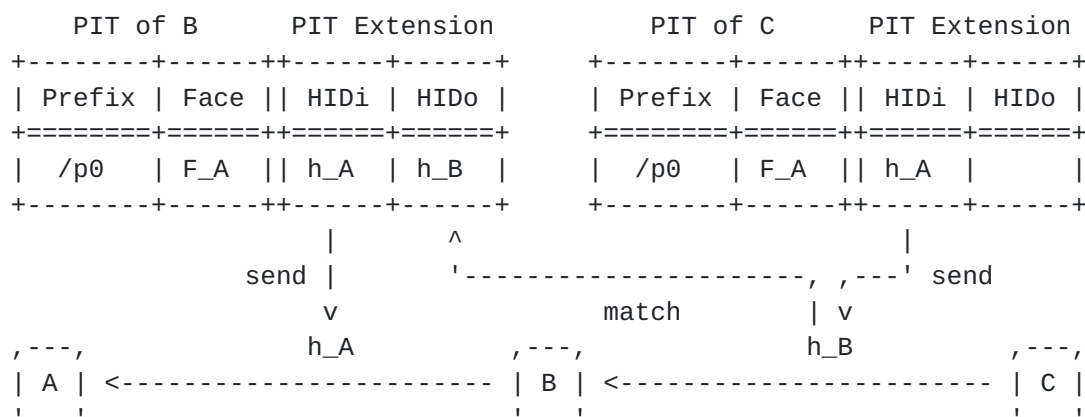


Figure 27: Eliding Name TLVs using en-route state (data).

It should be noted that each forwarder of an Interest in an ICN LowPAN network can individually decide whether to participate in en-route compression or not. However, an ICN LowPAN node SHOULD use en-

Main memory is typically a scarce resource of constrained networked devices. Fragmentation as described in this memo preserves fragments and purges them only after a packet is reassembled, which requires a buffering of all fragments. This scheme is able to handle fragments for distinctive packets simultaneously, which can lead to overflowing packet buffers which cannot hold all necessary fragments for packet reassembly. Implementers are thus urged to make use of appropriate buffer replacement strategies for fragments.

The stateful header compression generates ephemeral HopIDs for incoming and outgoing Interests and consumes them on returning Data packets. Forged Interests can deplete the number of available HopIDs, thus leading to a denial of compression service for subsequent content requests.

To further alleviate the problems caused by forged fragments or Interest initiations, proper protective mechanisms for accessing the link-layer should be deployed.

10. IANA Considerations

10.1. Page Switch Dispatch Type

This document makes use of "Page 2" from the existing paging dispatches in [[RFC8025](#)].

11. References

11.1. Normative References

- [ieee802.15.4]
IEEE Computer Society, "IEEE Std. 802.15.4-2015", April 2016, <<https://standards.ieee.org/findstds/standard/802.15.4-2015.html>>.
- [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>>.
- [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>>.

11.2. Informative References

- [CCN-LITE]
"CCN-lite: A lightweight CCNx and NDN implementation", <<http://ccn-lite.net/>>.

[I-D.irtf-icnrg-ccnxmessages]

Mosko, M., Solis, I., and C. Wood, "CCNx Messages in TLV Format", [draft-irtf-icnrg-ccnxmessages-08](#) (work in progress), July 2018.

[I-D.irtf-icnrg-ccnxsemantics]

Mosko, M., Solis, I., and C. Wood, "CCNx Semantics", [draft-irtf-icnrg-ccnxsemantics-09](#) (work in progress), June 2018.

[NDN]

Jacobson, V., Smetters, D., Thornton, J., and M. Plass, "Networking Named Content", 5th Int. Conf. on emerging Networking Experiments and Technologies (ACM CoNEXT), 2009, <<https://doi.org/10.1145/1658939.1658941>>.

[NDN-EXP1]

Baccelli, E., Mehlis, C., Hahm, O., Schmidt, TC., and M. Waehlich, "Information Centric Networking in the IoT: Experiments with NDN in the Wild", Proc. of 1st ACM Conf. on Information-Centric Networking (ICN-2014) ACM DL, pp. 77-86, September 2014, <<http://dx.doi.org/10.1145/2660129.2660144>>.

[NDN-EXP2]

Gundogan, C., Kietzmann, P., Lenders, M., Petersen, H., Schmidt, TC., and M. Waehlich, "NDN, CoAP, and MQTT: A Comparative Measurement Study in the IoT", Proc. of 5th ACM Conf. on Information-Centric Networking (ICN-2018) ACM DL, pp. , September 2018, <<http://dx.doi.org/>>.

[NDN-MAC]

Kietzmann, P., Gundogan, C., Schmidt, TC., Hahm, O., and M. Waehlich, "The Need for a Name to MAC Address Mapping in NDN: Towards Quantifying the Resource Gain", Proc. of 4th ACM Conf. on Information-Centric Networking (ICN-2017) ACM DL, pp. 36-42, September 2017, <<https://doi.org/10.1145/3125719.3125737>>.

[NDN-PACKET-SPEC]

"NDN Packet Format Specification", <<http://named-data.net/doc/NDN-packet-spec/0.3/>>.

[RFC7228]

Bormann, C., Ersue, M., and A. Keranen, "Terminology for Constrained-Node Networks", [RFC 7228](#), DOI 10.17487/RFC7228, May 2014, <<https://www.rfc-editor.org/info/rfc7228>>.

- [RFC7476] Pentikousis, K., Ed., Ohlman, B., Corujo, D., Boggia, G., Tyson, G., Davies, E., Molinaro, A., and S. Eum, "Information-Centric Networking: Baseline Scenarios", [RFC 7476](#), DOI 10.17487/RFC7476, March 2015, <<https://www.rfc-editor.org/info/rfc7476>>.
- [RFC7927] Kutscher, D., Ed., Eum, S., Pentikousis, K., Psaras, I., Corujo, D., Saucez, D., Schmidt, T., and M. Waehlich, "Information-Centric Networking (ICN) Research Challenges", [RFC 7927](#), DOI 10.17487/RFC7927, July 2016, <<https://www.rfc-editor.org/info/rfc7927>>.
- [RFC7945] Pentikousis, K., Ed., Ohlman, B., Davies, E., Spirou, S., and G. Boggia, "Information-Centric Networking: Evaluation and Security Considerations", [RFC 7945](#), DOI 10.17487/RFC7945, September 2016, <<https://www.rfc-editor.org/info/rfc7945>>.
- [RFC8025] Thubert, P., Ed. and R. Cragie, "IPv6 over Low-Power Wireless Personal Area Network (6LoWPAN) Paging Dispatch", [RFC 8025](#), DOI 10.17487/RFC8025, November 2016, <<https://www.rfc-editor.org/info/rfc8025>>.
- [RIOT] Baccelli, E., Guenes, M., Hahm, O., Schmidt, T.C., and M. Waehlich, "RIOT OS: Towards an OS for the Internet of Things", Proc. of the 32nd IEEE INFOCOM IEEE Press, pp. 79-80, April 2013, <<http://riot-os.org/>>.
- [TLV-ENC-802.15.4] "CCN and NDN TLV encodings in 802.15.4 packets", <<https://datatracker.ietf.org/meeting/interim-2015-icnrg-01/materials/slides-interim-2015-icnrg-1-2>>.
- [WIRE-FORMAT-CONSID] "CCN/NDN Protocol Wire Format and Functionality Considerations", <<https://datatracker.ietf.org/meeting/interim-2015-icnrg-01/materials/slides-interim-2015-icnrg-1-8>>.

Appendix A. Estimated Size Reduction

In the following a theoretical evaluation is given to estimate the gains of ICN LowPAN compared to uncompressed CCNx and NDN messages.

We assume that "n" is the number of name components, "comps_n" denotes the sum of n name component lengths. We also assume that the length of each name component is lower than 16 bytes. The length of the content is given by "clen". The lengths of TLV components is specific to the CCNx or NDN encoding and outlined below.

A.1. NDN

The NDN TLV encoding has variable-sized TLV fields. For simplicity, the 1 octet form of each TLV component is assumed. A typical TLV component therefore is of size 2 (type field + length field) + the actual value.

A.1.1. Interest

Figure 30 depicts the size requirements for a basic, uncompressed NDN Interest containing a CanBePrefix TLV, a MustBeFresh TLV, a InterestLifetime TLV set to 4 seconds and a HopLimit TLV set to 6. Numbers below represent the amount of octets.

-----	,	
Interest TLV	= 2	
-----	,	
Name	2 +	
NameComponents	= 2n +	
	comps_n	
-----	'	= 21 + 2n + comps_n
CanBePrefix	= 2	
MustBeFresh	= 2	
Nonce	= 6	
InterestLifetime	= 4	
HopLimit	= 3	
-----	'	

Figure 30: Estimated size of an uncompressed NDN Interest

Figure 31 depicts the size requirements after compression.

-----	,	
Dispatch Page Switch	= 1	
NDN Interest Dispatch	= 1	
Interest TLV	= 1	
-----	,	
Name		= 9 + n/2 + comps_n
NameComponents	= n/2 +	
	comps_n	
-----	,	
Nonce	= 4	
InterestLifetime	= 2	
-----	,	

Figure 31: Estimated size of a compressed NDN Interest

The size difference is:

$12 + 1.5n$ octets.

For the name "/DE/HH/HAW/BT7", the total size gain is 18 octets, which is 46% of the uncompressed packet.

[A.1.2.](#) Data

Figure 32 depicts the size requirements for a basic, uncompressed NDN Data containing a FreshnessPeriod as MetaInfo. A FreshnessPeriod of 1 minute is assumed. The value is thereby encoded using 2 octets. An HMACWithSha256 is assumed as signature. The key locator is assumed to contain a Name TLV of length klen.

Data TLV	= 2	
Name	2 +	
NameComponents	= 2n +	
	comps_n	
MetaInfo		
FreshnessPeriod	= 6	= 53 + 2n + comps_n +
		clen + klen
Content	= 2 + clen	
SignatureInfo		
SignatureType		
KeyLocator	= 41 + klen	
SignatureValue		
DigestSha256		

Figure 32: Estimated size of an uncompressed NDN Data

Figure 33 depicts the size requirements for the compressed version of the above Data packet.

Dispatch Page Switch	= 1	
NDN Data Dispatch	= 1	
Name		= 38 + n/2 + comps_n +
NameComponents	= n/2 +	clen + klen
	comps_n	
Content	= 1 + clen	
KeyLocator	= 1 + klen	
DigestSha256	= 32	
FreshnessPeriod	= 2	

Figure 33: Estimated size of a compressed NDN Data

The size difference is:
15 + 1.5n octets.

For the name "/DE/HH/HAW/BT7", the total size gain is 21 octets.

A.2. CCNx

The CCNx TLV encoding defines a 2-octet encoding for type and length fields, summing up to 4 octets in total without a value.

A.2.1. Interest

Figure 34 depicts the size requirements for a basic, uncompressed CCNx Interest. No Hop-By-Hop TLVs are included, the protocol version is assumed to be 1 and the reserved field is assumed to be 0. A KeyIdRestriction TLV with T_SHA-256 is included to limit the responses to Content Objects containing the specific key.

-----	,	
Fixed Header	= 8	
Message	= 4	
-----	,	
Name	4 +	= 56 + 4n + comps_n
NameSegments	= 4n +	
	comps_n	
-----	'	
KeyIdRestriction	= 40	
-----	'	

Figure 34: Estimated size of an uncompressed CCNx Interest

Figure 35 depicts the size requirements after compression.

-----	,	
Dispatch Page Switch	= 1	
CCNx Interest Dispatch	= 2	
Fixed Header	= 3	
-----	,	
Name		= 38 + n/2 + comps_n
NameSegments	= n/2 +	
	comps_n	
-----	'	
T_SHA-256	= 32	
-----	'	

Figure 35: Estimated size of a compressed CCNx Interest

The size difference is:
18 + 3.5n octets.

For the name "/DE/HH/HAW/BT7", the size is reduced by 53 octets, which is 53% of the uncompressed packet.

A.2.2. Content Object

Figure 36 depicts the size requirements for a basic, uncompressed CCNx Content Object containing an ExpiryTime Message TLV, an HMAC_SHA-256 signature, the signature time and a hash of the shared secret key. In the fixed header, the protocol version is assumed to be 1 and the reserved field is assumed to be 0

Fixed Header	= 8	
Message	= 4	
-----,		
Name	4 +	
NameSegments	= 4n +	
	comps_n	
-----,		
ExpiryTime	= 12	= 124 + 4n + comps_n + clen
Payload	= 4 + clen	
-----,		
ValidationAlgorithm		
T_HMAC-256	= 56	
KeyId		
SignatureTime		
-----,		
ValidationPayload	= 36	
-----,		

Figure 36: Estimated size of an uncompressed CCNx Content Object

Figure 37 depicts the size requirements for a basic, compressed CCNx Data.

-----	,	
Dispatch Page Switch	= 1	
CCNx Content Dispatch	= 3	
Fixed Header	= 2	
-----	,	
Name		
NameSegments	= n/2 +	
	comps_n	= 89 + n/2 + comps_n + clen
-----	'	
ExpiryTime	= 8	
Payload	= 1 + clen	
T_HMAC-SHA256	= 32	
SignatureTime	= 8	
ValidationPayload	= 34	
-----	'	

Figure 37: Estimated size of a compressed CCNx Data Object

The size difference is:
 35 + 3.5n octets.

For the name "/DE/HH/HAW/BT7", the size is reduced by 70 octets,
 which is 40% of the uncompressed packet containing a 4-octet payload.

Acknowledgments

This work was stimulated by fruitful discussions in the ICNRG research group and the communities of RIOT and CCNlite. We would like to thank all active members for constructive thoughts and feedback. In particular, the authors would like to thank (in alphabetical order) Peter Kietzmann, Dirk Kutscher, Martine Lenders, +++. The hop-wise stateful name compression was brought up in a discussion by Dave Oran, which is gratefully acknowledged. Larger parts of this work are inspired by [\[RFC4944\]](#) and [\[RFC6282\]](#). Special mentioning goes to Mark Mosko as well as G.Q. Wang and Ravi Ravindran as their previous work in [\[TLV-ENC-802.15.4\]](#) and [\[WIRE-FORMAT-CONSID\]](#) provided a good base for our discussions on stateless header compression mechanisms. This work was supported in part by the German Federal Ministry of Research and Education within the projects I3 and RAPstore.

Authors' Addresses

Cenk Gundogan
HAW Hamburg
Berliner Tor 7
Hamburg D-20099
Germany

Phone: +4940428758067

EMail: cenk.guendogan@haw-hamburg.de

URI: <http://inet.haw-hamburg.de/members/cenk-gundogan>

Thomas C. Schmidt
HAW Hamburg
Berliner Tor 7
Hamburg D-20099
Germany

EMail: t.schmidt@haw-hamburg.de

URI: <http://inet.haw-hamburg.de/members/schmidt>

Matthias Waehlich
link-lab & FU Berlin
Hoenower Str. 35
Berlin D-10318
Germany

EMail: mw@link-lab.net

URI: <http://www.inf.fu-berlin.de/~waehl>

Christopher Scherb
University of Basel
Spiegelgasse 1
Basel CH-4051
Switzerland

EMail: christopher.scherb@unibas.ch

Claudio Marxer
University of Basel
Spiegelgasse 1
Basel CH-4051
Switzerland

EMail: claudio.marxer@unibas.ch

Christian Tschudin
University of Basel
Spiegelgasse 1
Basel CH-4051
Switzerland

EMail: christian.tschudin@unibas.ch