Network Working Group
Request for Comments: 4019
Category: Standards Track

G. Pelletier
Ericsson AB

April 2005

RObust Header Compression (ROHC): Profiles for User Datagram Protocol (UDP) Lite

Status of This Memo

Copyright Notice

Abstract

This document defines Robust Header Compression (ROHC) profiles for compression of Real-Time Transport Protocol, User Datagram Protocol-Lite, and Internet Protocol (RTP/UDP-Lite/IP) packets and UDP-Lite/IP. These profiles are defined based on their differences with the profiles for UDP as specified in RFC 3095.

Table of Contents

1. Introduction

The ROHC WG has developed a header compression framework on top of which various profiles can be defined for different protocol sets or compression strategies. Due to the demands of the cellular industry for an efficient way to transport voice over IP over wireless, ROHC [2] has mainly focused on compression of IP/UDP/RTP headers, which are generous in size, especially compared to the payloads often carried by packets with these headers.

ROHC RTP has become a very efficient, robust, and capable compression scheme, able to compress the headers down to a total size of one octet only. Also, transparency is guaranteed to an extremely high extent, even when residual bit errors are present in compressed headers delivered to the decompressor.

UDP-Lite [4] is a transport protocol similar to the UDP protocol [7]. UDP-Lite is useful for applications designed with the capability to tolerate errors in the payload, for which receiving damaged data is better than dealing with the loss of entire packets. This may be particularly suitable when packets are transported over link technologies in which data can be partially damaged, such as wireless links.

Although these transport protocols are very similar, ROHC profiles must be defined separately for robust compression of UDP-Lite headers because UDP-Lite does not share the same protocol identifier with UDP. Also, the UDP-Lite Checksum Coverage field does not share the semantics of the corresponding UDP Length field, and as a consequence it cannot always be inferred anymore.

This document defines two ROHC profiles for efficient compression of UDP-Lite headers. The objective of this document is to provide simple modifications to the corresponding ROHC profiles for UDP, specified in RFC 3095 [2]. In addition, the ROHC profiles for UDP-Lite support some of the mechanisms defined in the profile for compression of IP headers [3] (ROHC IP-Only). This specification includes support for compression of multiple IP headers and for compressing IP-ID fields with constant behavior, as well as improved mode transition logic and a feedback option for decompressors with limited memory resources.

2. Terminology

In this document, the key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD, "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" are to be interpreted as described in RFC 2119 [1].

```
ROHC RTP          : RTP/UDP/IP profile 0x0001 defined in RFC 3095 [2].
ROHC UDP          : UDP/IP profile 0x0002 defined in RFC 3095 [2].
ROHC UDP-Lite     : UDP-Lite/IP profile defined in this document.
ROHC RTP/UDP-Lite: RTP/UDP-Lite/IP profile defined in this document.
```

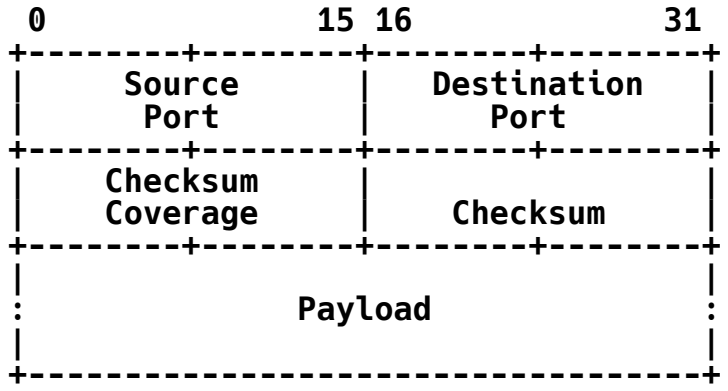
3. Background

3.1. Overview of the UDP-Lite Protocol

UDP-Lite is a transport protocol defined as an independent variant of the UDP transport protocol. UDP-Lite is very similar to UDP, and it allows applications that can tolerate errors in the payload to use a checksum with an optional partial coverage. This is particularly useful with IPv6 [6], in which the use of the transport-layer checksum is mandatory.

UDP-Lite replaces the Length field of the UDP header with a Checksum Coverage field. This field indicates the number of octets covered by the 16-bit checksum, which is applied on a per-packet basis. The coverage area always includes the UDP-Lite header and may cover the entire packet, in which case UDP-Lite becomes semantically identical to UDP. UDP-Lite and UDP do not share the same protocol identifier.

The UDP-Lite header format:



Like the UDP checksum, the UDP-Lite checksum is an end-to-end mechanism against erroneous delivery of error sensitive data. This checksum is mandatory with IPv6 [5] for both protocols. However, unlike its UDP counterpart, the UDP-Lite checksum may not be transmitted as all zeroes and cannot be disabled for IPv4 [5]. For UDP, if the checksum is disabled (IPv4 only), the Checksum field maintains a constant value and is normally not sent by the header compression scheme. If the UDP checksum is enabled (mandatory for IPv6), such an unpredictable field cannot be compressed and is sent uncompressed. The UDP Length field, however, is always redundant and can be provided by the IP module. Header compression schemes do not normally transmit any bits of information for this field, as its value can be inferred from the link layer.

For UDP-Lite, the checksum also has unpredictable values, and this field must always be included as-is in the compressed header for both IPv4 and IPv6. Furthermore, as the UDP Length field is redefined as the Checksum Coverage field by UDP-Lite, this leads to different properties for this field from a header-compression perspective.

The following summarizes the relationship between UDP and UDP-Lite:

- UDP-Lite and UDP have different protocol identifiers.
- The UDP-Lite checksum cannot be disabled for IPv4.
- UDP-Lite redefines the UDP Length field as the Checksum Coverage field, with different semantics.
- UDP-Lite is semantically equivalent to UDP when the Checksum Coverage field indicates the total length of the packet.

The next section provides a more detailed discussion of the behavior of the Checksum Coverage field of UDP-Lite in relation to header compression.

3.2. Expected Behaviours of UDP-Lite Flows

3.2.1. Per-Packet Behavior

As mentioned in the previous section, the checksum coverage value is applied independently of other packets that may belong to the same flow. Specifically, the value of the checksum coverage may indicate that the UDP-Lite packet is either entirely covered by the checksum or covered up to some boundary less than the packet size but including the UDP-Lite header.

3.2.2. Inter-Packet Behavior

In relation to each other, UDP-Lite packets may exhibit one of three possible change patterns, where within a sequence of packets the value of the Checksum Coverage field is

1. changing, while covering the entire packet;
2. unchanging, covering up to a fixed boundary within the packet; or
3. changing, but it does not follow any specific pattern.

The first pattern above corresponds to the semantics of UDP, when the UDP checksum is enabled. For this case, the checksum coverage field varies according to the packet length and may be inferred from the IP header, as is the UDP Length field value.

The second pattern corresponds to the case where the coverage is the same from one packet to another within a particular sequence. For this case, the Checksum Coverage field may be a static value defined in the context, and it does not have to be sent in the compressed header. For the third case, no useful change pattern can be identified from packet to packet for the value of the checksum coverage field, and it must be included in the compressed header.

3.2.3. Per-Flow behavior

It can be expected that any one of the above change patterns for sequences of packets may be predominant at any time during the lifetime of the UDP-Lite flow. A flow that predominantly follows the first two change patterns described above may provide opportunities for compressing the Checksum Coverage field for most of the packets.

3.3. Header Field Classification

In relation to the header field classification of RFC 3095 [2], the first two patterns represent the case where the value of the Checksum Coverage field behavior is fixed and may be either INFERRED (pattern

1) or STATIC (pattern 2). Pattern 3 is for the case where the value varies unpredictably, the field is CHANGING, and the value must be sent along with every packet.

Additional information regarding the analysis of the behavior of the UDP-Lite fields may be found in Appendix A.

4. Rationale behind the Design of ROHC Profiles for UDP-Lite

4.1. Design Motivations

Simplicity is a strong motivation for the design of the UDP-Lite header compression profiles. The profiles defined for UDP-Lite should entail only a few simple modifications to the corresponding profiles defined for UDP in RFC 3095 [2]. In addition, it is desirable to include some of the improvements found in the ROHC IP-Only profile [3]. Finally, whenever UDP-Lite is used in a manner that is semantically identical to UDP, the compression efficiency should be similar.

4.2. ROHC Considerations

The simplest approach to the definition of ROHC profiles for UDP-Lite is to treat the Checksum Coverage field as an irregular value, and to send it uncompressed for every packet. This may be achieved simply by adding the field to the definition of the general packet format [2]. However, then the compression efficiency would always be less than for UDP.

Some care should be given to achieve compression efficiency for UDP-Lite similar to that for UDP when the Checksum Coverage field behaves like the UDP Length field. This requires the possibility to infer the Checksum Coverage field when it is equal to the length of the packet. Otherwise, this would put the UDP-Lite protocol at a disadvantage over links where header compression is used, when its behavior is made similar to the semantics of UDP.

A mechanism to detect the presence of the Checksum Coverage field in compressed headers is thus needed. This is achieved by defining a new packet type with the identifiers left unused in RFC 3095 [2].

5. ROHC Profiles for UDP-Lite

This section defines two ROHC profiles:

- RTP/UDP-Lite/IP compression (profile 0x0007)
- UDP-Lite/IP compression (profile 0x0008)

These profiles build on the specifications found in RFC 3095 [2], with as little modification as possible. Unless it is explicitly stated otherwise, the profiles defined herein follow the specifications of ROHC UDP and ROHC RTP, respectively.

Note also that this document reuses the notation found in [2].

5.1. Context Parameters

As described in [2], information about previous packets is maintained in a context. This includes information describing the packet stream and compression parameters. Although the UDP and UDP-Lite protocols share many commonalities, the differences in semantics as described earlier render the following parameter inapplicable:

The parameter context(UDP Checksum)

The UDP-Lite checksum cannot be disabled, as opposed to UDP. The parameter context(UDP Checksum) defined in [2] (section 5.7) is therefore not used for compression of UDP-Lite.

In addition, the UDP-Lite checksum is always sent as-is in every compressed packet. However, the Checksum Coverage field may not always be sent in each compressed packet, and the following context parameter is used to indicate whether the field is sent:

The parameter context(UDP-Lite Coverage Field Present)

Whether the UDP-Lite Checksum Coverage field is present or not in the general packet format (see section 5.3.1) is controlled by the value of the Coverage Field Present (CFP) flag in the context.

If context(CFP) is nonzero, the Checksum Coverage field is not compressed, and it is present within compressed packets. If context(CFP) is zero, the Checksum Coverage field is compressed, and it is not sent. This is the case when the value of the Checksum Coverage field follows a stable inter-packet change pattern; the field has either a constant value or it has a value equal to the packet length for most packets in a sequence (see section 3.2).

Finally, the following context parameter is needed to indicate whether the field should be inferred or taken from a value previously saved in the context:

The parameter context(UDP-Lite Coverage Field Inferred)

When the UDP-Lite Checksum Coverage field is not present in the compressed header (CFP=0), whether it is inferred is controlled by the value of the Coverage Field Inferred (CFI) flag in the context.

If context(CFI) is nonzero, the Checksum Coverage field is inferred from the packet length, similarly as for the UDP Length field in ROHC RTP. If context(CFI) is zero, the Checksum Coverage field is decompressed by using context(UDP-Lite Checksum Coverage). Therefore, when context(CFI) is updated to a nonzero value, the value of the Checksum Coverage field stored in the context must also be updated.

5.2. Initialization

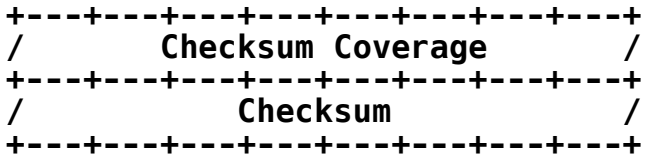
Unless it is stated otherwise, the mechanisms of ROHC RTP and ROHC UDP found in [2] are used also for the ROHC RTP/UDP-Lite and the ROHC UDP-Lite profiles, respectively.

In particular, the considerations of ROHC UDP regarding the UDP SN taking the role of the RTP Sequence Number apply to ROHC UDP-Lite. Also, the static context for ROHC UDP-Lite may be initialized by reusing an existing context belonging to a stream compressed by using ROHC RTP/UDP-Lite (profile 0x0007), similarly as for ROHC UDP.

5.2.1. Initialization of the UDP-Lite Header [1]

The structure of the IR and IR-DYN packets and the initialization procedures are the same as for the ROHC profiles for UDP [2], with the exception of the dynamic part as specified for UDP. A 2-octet field containing the checksum coverage is added before the Checksum field. This affects the format of dynamic chains in both IR and IR-DYN packets.

Dynamic part:



CRC-DYNAMIC: Checksum Coverage field, Checksum field (octets 5 - 8).

CRC-STATIC: All other fields (octets 1 - 4).

5.2.2. Compressor and Decompressor Logic

The following logic must be used by both the compressor and the decompressor for assigning values to the parameters context(CFP) and context(CFI) during initialization:

Context(CFP)

During context initialization, the value of context(CFP) MUST be set to a nonzero value if the Checksum Coverage field differs from the length of the UDP-Lite packet, for any one IR or IR-DYN packet sent (compressor) or received (decompressor); otherwise, the value MUST be set to zero.

Context(CFI)

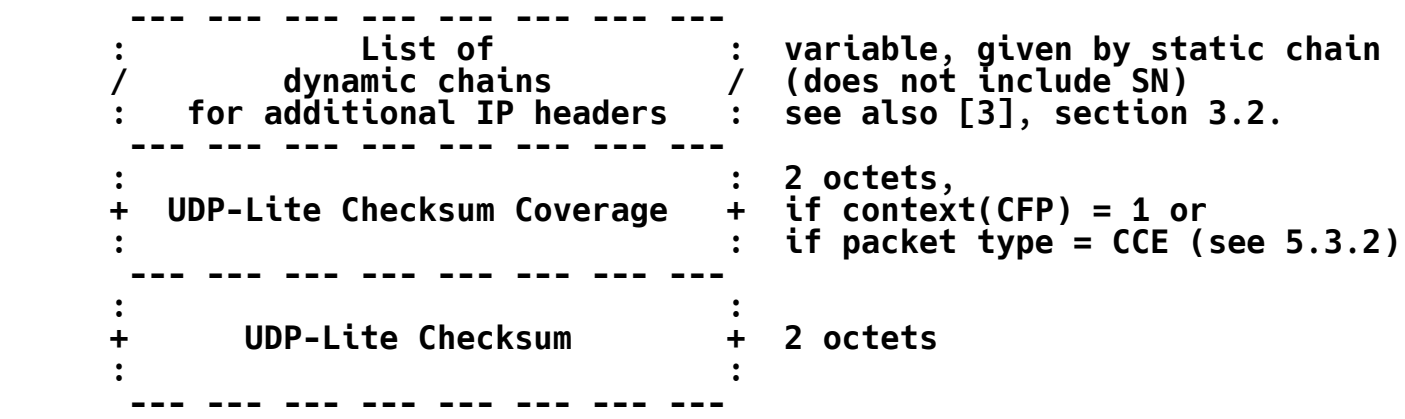
During context initialization, the value of context(CFI) MUST be set to a nonzero value if the Checksum Coverage field is equal to the length of the UDP-Lite packet within an IR or an IR-DYN packet sent (compressor) or received (decompressor); otherwise, the value MUST be set to zero.

5.3. Packet Formats

The general packet format, as defined in RFC 3095 [2], is modified to include an additional field for the UDP-Lite checksum coverage. A packet type is also defined to handle the specific semantics and characteristics of this field.

5.3.1. General Packet Format

The general packet format of a compressed ROHC UDP-Lite header is similar to the compressed ROHC RTP header ([2], section 5.7), with modifications to the Checksum field, as well as additional fields for handling multiple IP headers and for the UDP-Lite checksum coverage:



The list of dynamic header chains carries the dynamic header part for each IP header in excess of the initial two, if there is any (as indicated by the presence of corresponding header parts in the static chain). Note that there is no sequence number at the end of the chain, as SN is present within compressed base headers.

The order of the fields following the optional extension of the general ROHC packet format is the same as the order between the fields in the uncompressed header.

When the CRC is calculated, the Checksum Coverage field is CRC-DYNAMIC.

5.3.2. Packet Type CCE: CCE(), CCE(ON), and CCE(OFF)

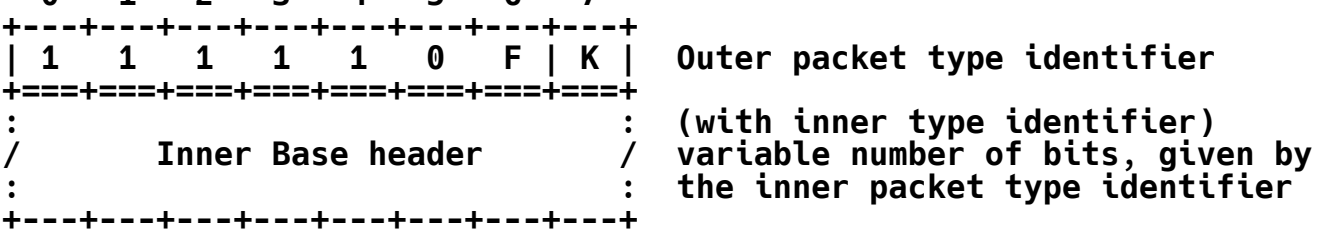
The ROHC profiles for UDP-Lite define a packet type to handle the various possible change patterns of the checksum coverage. This packet type may be used to manipulate the context values that control the presence of the Checksum Coverage field within the general packet format (i.e., context(CFP)) and how the field is decompressed (i.e., context(CFI)). The 2-octet Checksum Coverage field is always present within the format of this packet (see section 5.3.1).

This type of packet is named Checksum Coverage Extension, or CCE, and its updating properties depend on the final two bits of the packet type octet (see format below). A naming scheme of the form CCE(<some_property>) is used to uniquely identify the properties of a particular CCE packet.

Although this packet type defines its own format, it may be considered as an extension mechanism for packets of type 2, 1, or 0 [2]. This is achieved by substitution of the packet type identifier of the first octet of the base header (the "outer" identifier) with

one of the unused packet types from RFC 3095 [2]. The substituted identifier is then moved to the first octet of the remainder of the base header (the "inner" identifier).

The format of the ROHC UDP-Lite CCE packet type is as follows:



F,K: F,K = 00 is reserved at framework level (IR-DYN);
F,K = 01 indicates CCE();
F,K = 10 indicates CCE(ON);
F,K = 11 indicates CCE(OFF).

Updating properties: The updating properties of the inner packet type carried within any of the CCE packets are always maintained. CCE(ON) and CCE(OFF) MUST NOT be used to extend R-0 and R-1* headers. In addition, CCE(ON) always updates context(CFP); CCE(OFF) always updates context(CFP), context(CFI), and context(UDP-Lite Checksum Coverage).

Appendix B provides an expanded view of the resulting format of the CCE packet type.

5.3.2.1. Properties of CCE()

Aside from the updating properties of the inner packet type carried within CCE(), this packet does not update any other context values. CCE() thus is mode-agnostic; e.g., it can extend any of packet types 2, 1, and 0, regardless of the current mode of operation [2].

CCE() may be used when the checksum coverage deviates from the change pattern assumed by the compressor, where the field could previously be compressed. This packet is useful if the occurrence of such deviations is rare.

5.3.2.2. Properties of CCE(ON)

In addition to the updating properties of the inner packet type, CCE(ON) updates context(CFP) to a nonzero value; i.e., it effectively turns on the presence of the Checksum Coverage field within the

general packet format. This is useful when the predominant change pattern of the checksum coverage precludes its compression.

CCE(ON) can extend any of the context-updating packets of type 2, 1, and 0; that is, packets with a compressed header containing a CRC [2]. Specifically, R-0 and R-1* headers MUST NOT be extended by using CCE(ON).

5.3.2.3. Properties of CCE(OFF)

In addition to the updating properties of the inner packet type, CCE(OFF) updates context(CFP) to a value of zero; i.e., it effectively turns off the presence of the Checksum Coverage field within the general packet format. This is useful when the change pattern of the checksum coverage seldom deviates from the pattern assumed by the compressor.

CCE(OFF) also updates context(CFI) to a nonzero value, if field(UDP-Lite Checksum Coverage) is equal to the packet length; otherwise, it must be set to zero. Note that when context(CFI) is updated by using packet type CCE(OFF), a match of field(Checksum Coverage) with the packet length always has precedence over a match with context(Checksum Coverage). Finally, context(UDP-Lite Checksum Coverage) is also updated by CCE(OFF).

Similarly to CCE(ON), CCE(OFF) can extend any of the context updating packets of type 2, 1, and 0 [2].

5.4. Compressor Logic

If hdr(UDP-Lite Checksum Coverage) is different from context(UDP-Lite Checksum Coverage) and different from the packet length when context(CFP) is zero, the Checksum Coverage field cannot be compressed. In addition, if hdr(UDP-Lite Checksum Coverage) is different from the packet length when context(CFP) is zero and context(CFI) is nonzero, the Checksum Coverage field cannot be compressed by either. For both cases, the field must be sent uncompressed using a CCE packet, or the context must be reinitialized by using an IR packet.

5.5. Decompressor Logic

For packet types other than IR, IR-DYN, and CCE that are received when the value of context(CFP) is zero, the Checksum Coverage field must be decompressed by using the value stored in the context if the value of context(CFI) is zero; otherwise, the field is inferred from the length of the UDP-Lite packet derived from the IP module.

5.6. Additional Mode Transition Logic

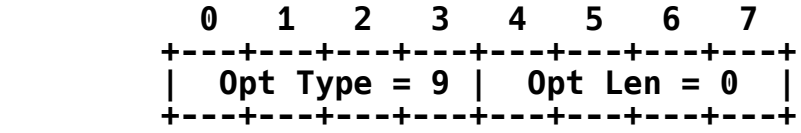
The profiles defined in this document allow the compressor to decline a mode transition requested by the decompressor. This is achieved by redefining the Mode parameter for the value mode = 0 (in packet types UOR-2, IR, and IR-DYN) as follows (see also [3], section 3.4):

Mode: Compression mode. 0 = (C)ancel Mode Transition

Upon receiving the Mode parameter set to 0, the decompressor MUST stay in its current mode of operation and SHOULD refrain from sending further mode transition requests for the declined mode.

5.7. The CONTEXT_MEMORY Feedback Option

This feedback option informs the compressor that the decompressor does not have sufficient memory resources to handle the context of the packet stream required by the current compressed structure.

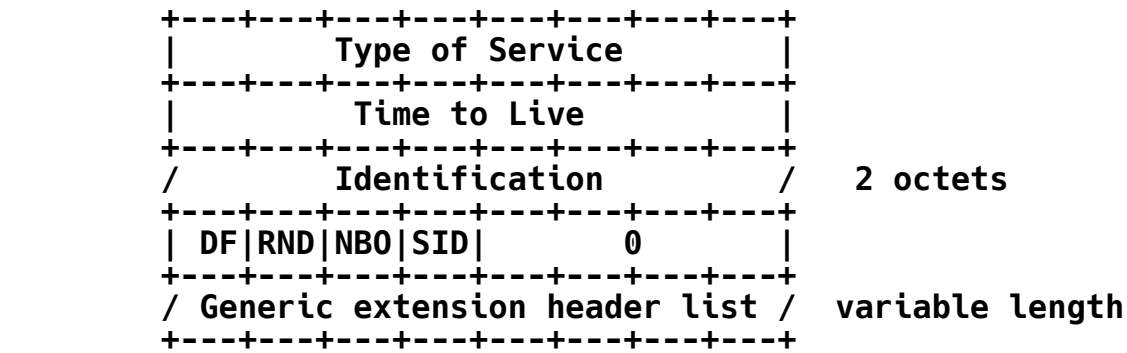


When receiving a CONTEXT_MEMORY option, the compressor SHOULD take actions to compress the packet stream in a way that requiring less decompressor memory resources or stop compressing the packet stream.

5.8. Constant IP-ID

The profiles for UDP-Lite support compression of the IP-ID field with constant behavior, with the addition of the Static IP Identifier (SID) flag within the dynamic part of the chain used to initialize the IPv4 header, as follows (see also [3], section 3.3):

Dynamic part:



SID: Static IP Identifier.

For IR and IR-DYN packets:

The logic is the same as that for the respective ROHC profiles for UDP, with the addition that field (SID) must be kept in the context.

For compressed headers other than IR and IR-DYN:

If value(RND) = 0 and context(SID) = 0, hdr(IP-ID) is compressed by using Offset IP-ID encoding (see [2], section 4.5.5) using p = 0 and default-slope(IP-ID offset) = 0.

If value(RND) = 0 and context(SID) = 1, hdr(IP-ID) is constant and compressed away; hdr(IP-ID) is the value of context(IP-ID).

If value(RND) = 1, IP-ID is the uncompressed hdr(IP-ID). IP-ID is then passed as additional octets at the end of the compressed header, after any extensions.

Note: Only IR and IR-DYN packets can update context(SID).

Note: All other fields are the same as for the respective ROHC profiles for UDP [2].

6. Security Considerations

The security considerations of RFC 3095 [2] apply integrally to this document, without modification.

7. IANA Considerations

ROHC profile identifiers 0x0007 (ROHC RTP/UDP-Lite) and 0x0008 (ROHC UDP-Lite) have been reserved by the IANA for the profiles defined in this document (RFC 4019).

Two ROHC profile identifiers must be reserved by the IANA for the profiles defined in this document. Since profile number 0x0006 is being saved for the TCP/IP (ROHC-TCP) profile, profile numbers 0x0007 and 0x0008 are the most suitable unused identifiers available, and should thus be used. As for previous ROHC profiles, profile numbers 0xnn07 and 0xnn08 must also be reserved for future variants of these profiles. The registration suggested for the "RObust Header Compression (ROHC) Profile Identifiers" name space:

OLD:	0x0006-0xnn7F	To be Assigned by IANA	
NEW:	0xnn06	To be Assigned by IANA	
	0x0007	ROHC RTP/UDP-Lite	[RFC4019]
	0xnn07	Reserved	
	0x0008	ROHC UDP-Lite	[RFC4019]
	0xnn08	Reserved	
	0x0009-0xnn7F	To be Assigned by IANA	

8. Acknowledgments

The author would like to thank Lars-Erik Jonsson, Kristofer Sandlund, Mark West, Richard Price, Gorry Fairhurst, Fredrik Linstroem and Mats Nordberg for useful reviews and discussions around this document.

9. References

9.1. Normative References

[1] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[2] Bormann, C., Burmeister, C., Degermark, M., Fukushima, H., Hannu, H., Jonsson, L-E., Hakenberg, R., Koren, T., Le, K., Liu, Z., Martensson, A., Miyazaki, A., Svanbro, K., Wiebke, T., Yoshimura, T., and H. Zheng, "RObust Header Compression (ROHC): Framework and four profiles: RTP, UDP, ESP, and uncompressed", RFC 3095, July 2001.

[3] Jonsson, L-E. and G. Pelletier, "RObust Header Compression (ROHC): A Compression Profile for IP", RFC 3843, June 2004.

[4] Larzon, L-A., Degermark, M., Pink, S., Jonsson, L-E., and G. Fairhurst, "The Lightweight User Datagram Protocol (UDP-Lite)", RFC 3828, July 2004.

9.2. Informative References

[5] Postel, J., "Internet Protocol", STD 5, RFC 791, September 1981.

[6] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", RFC 2460, December 1998.

[7] Postel, J., "User Datagram Protocol", STD 6, RFC 768, August 1980.

[8] Schulzrinne, H., Casner, S., Frederick, R., and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications", STD 64, RFC 3550, July 2003.

Appendix A. Detailed Classification of Header Fields

This section summarizes the difference from the classification found in the corresponding appendix in RFC 3095 [2] and similarly provides conclusions about how the various header fields should be handled by the header compression scheme to optimize compression and functionality. These conclusions are separated based on the behavior of the UDP-Lite Checksum Coverage field and use the expected change patterns described in section 3.2 of this document.

A.1. UDP-Lite Header Fields

The following table summarizes a possible classification for the UDP-Lite header fields in comparison with the classification for UDP, using the same classes as in RFC 3095 [2].

Header fields of UDP-Lite and UDP:

		UDP-Lite	UDP
Header Field	Size (bits)	Class	Class
Source Port	16	STATIC-DEF	STATIC-DEF
Destination Port	16	STATIC-DEF	STATIC-DEF
Checksum Coverage	16	INFERRED STATIC CHANGING	
Length	16		INFERRED
Checksum	16	CHANGING	CHANGING

Source and Destination Port

Same as for UDP. Specifically, these fields are part of the definition of a stream and must thus be constant for all packets in the stream. The fields are therefore classified as STATIC-DEF.

Checksum Coverage

This field specifies which part of the UDP-Lite datagram is covered by the checksum. It may have a value of zero or be equal to the datagram length if the checksum covers the entire datagram, or it may have any value between eight octets and the length of the datagram to specify the number of octets protected by the checksum,

calculated from the first octet of the UDP-Lite header. The value of this field may vary for each packet, and this makes the value unpredictable from a header-compression perspective.

Checksum

The information used for the calculation of the UDP-Lite checksum is governed by the value of the checksum coverage and minimally includes the UDP-Lite header. The checksum is a changing field that must always be sent as-is.

The total size of the fields in each class, for each expected change pattern (see section 3.2), is summarized in the tables below:

Pattern 1:

Class	Size (octets)	
INFERRED	2	Checksum Coverage
STATIC-DEF	4	Source Port / Destination Port
CHANGING	2	Checksum

Pattern 2:

Class	Size (octets)	
STATIC-DEF	4	Source Port / Destination Port
STATIC	2	Checksum Coverage
CHANGING	2	Checksum

Pattern 3:

Class	Size (octets)	
STATIC-DEF	4	Source Port / Destination Port
CHANGING	4	Checksum Coverage / Checksum

A.2. Header Compression Strategies for UDP-Lite

The following table revisits the corresponding table (table A.1) for UDP from [2] (section A.2) and classifies the changing fields based on the change patterns previously identified in section 3.2.

Header compression strategies for UDP-Lite:

Field	Pattern	Value/Delta	Class	Knowledge
Checksum Coverage	#1	Value	CHANGING	INFERRED
	#2	Value	RC	UNKNOWN
	#3	Value	IRREGULAR	UNKNOWN
Checksum	All	Value	IRREGULAR	UNKNOWN

A.2.1. Transmit initially but be prepared to update

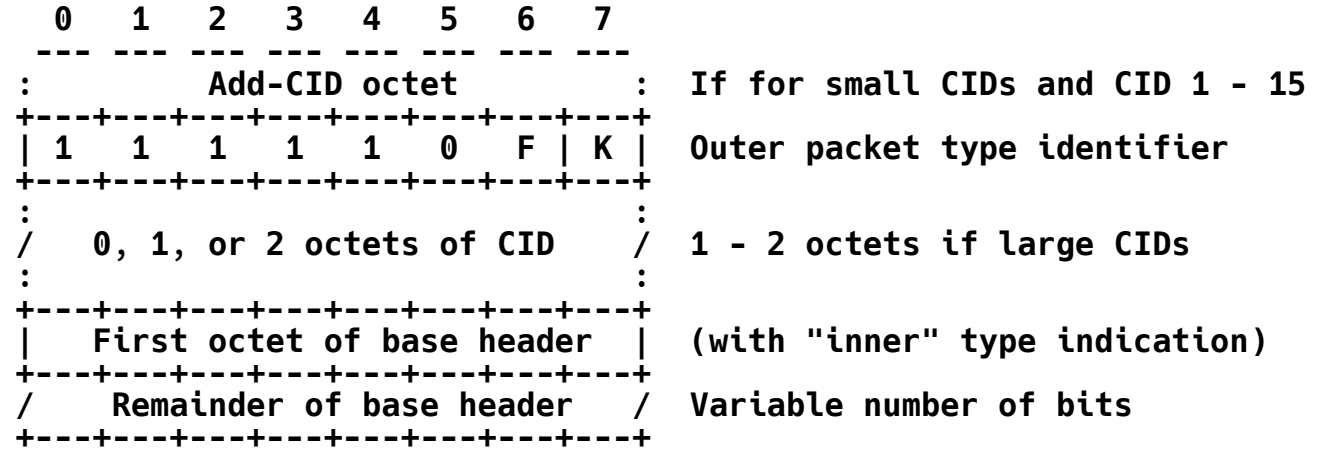
UDP-Lite Checksum Coverage (Patterns #1 and #2)

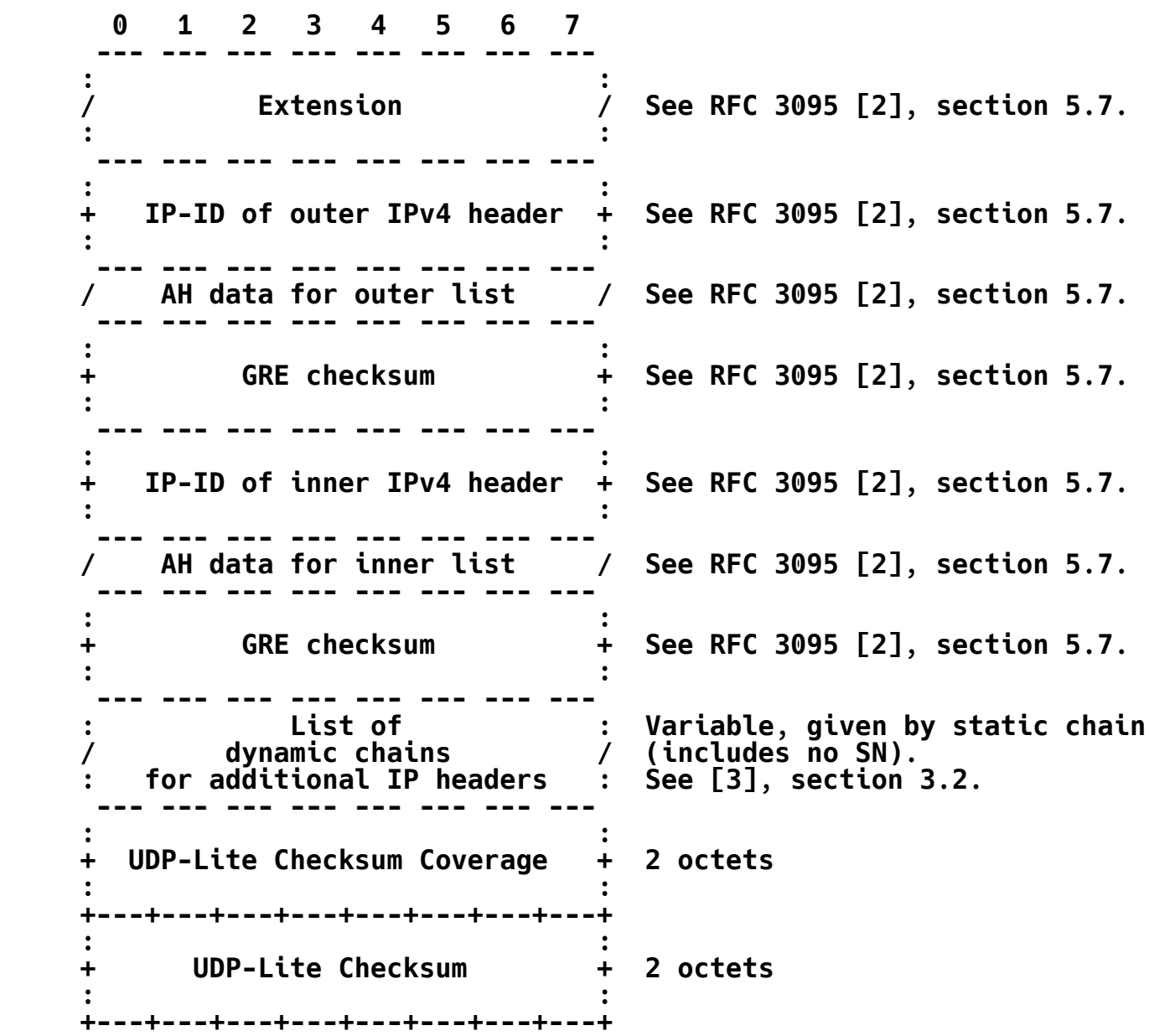
A.2.2. Transmit as-is in all packets

UDP-Lite Checksum
UDP-Lite Checksum Coverage (Pattern #3)

Appendix B. Detailed Format of the CCE Packet Type

This section provides an expanded view of the format of the CCE packet, based on the general ROHC RTP compressed header [2] and the general format of a compressed header of the ROHC IP-Only profile [3]. The modifications necessary to carry the base header of a packet of type 2, 1 or 0 [2] within the CCE packet format, along with the additional fields to properly handle compression of multiple IP headers, result in the following structure for the CCE packet type:





F,K: F,K = 00 is reserved at framework level (IR-DYN);
F,K = 01 indicates CCE();
F,K = 10 indicates CCE(ON);
F,K = 11 indicates CCE(OFF).

Note that this document does not define (F,K) = 00, as this would collide with the IR-DYN packet type already reserved at the ROHC framework level.

Author's Address

Ghyslain Pelletier
Ericsson AB
Box 920
SE-971 28 Lulea, Sweden

Phone: +46 840 429 43
Fax : +46 920 996 21
EMail: ghyslain.pelletier@ericsson.com

Full Copyright Statement

Intellectual Property

Acknowledgement

Funding for the RFC Editor function is currently provided by the Internet Society.