High-bandwidth Digital Content Protection System

Direct Adaptation Amendment

Revision 2.3

19 May, 2021

Notice

Acknowledgement

Intellectual Property

Implementation of this specification requires a license from the Digital Content Protection LLC.

Contact Information

Digital Content Protection LLC
C/O Vital Technical Marketing, Inc.
3855 SW 153rd Drive
Beaverton, OR 97006

Email: info@digital-cp.com

Web: www.digital-cp.com

Revision History

1. Introduction

1.0 Introduction to this Amendment

This section explains the method of deriving the "Direct Adaptation Spec" from the "Independent adaptation Spec".

This amendment to the HDCP Interface Independent Adaptation Specification, Revision 2.3 intends to expand the scope of HDCP to a family of IP protocols that encapsulate the media directly over RTP protocol. This family of protocols deliver the Audiovisual content over the network as independent components (i.e. Audio, Video, Metadata), each encapsulated independently and directly over the RTP protocol and sent as an independent RTP stream.

The HDCP Interface Independent Adaptation Specification, Revision 2.3 is based on the MPEG2-TS (and PS) protocol delivering a bundle of Audiovisual programs. A program is composed of a set of elementary streams, each carrying one component of the Audiovisual stream. This amendment uses an analogy, illustrated in Table 1, where "Elementary Stream" in MPEG2-TS protocol is analogous to an "RTP stream" in Direct encapsulation over RTP. A "PES" in MPEG2-TS protocol is analogous to an "RTP Stream Payload" in Direct encapsulation over RTP. A "PES Packet" in MPEG2-TS protocol is analogous to an "HDCP Data Unit (HDU)", as defined in this amendment (refer to 1.2). A "Program" in MPEG2-TS protocol is analogous to a "Video Stream", i.e. a bundle of related RTP streams, each carrying one component of a Video Stream. An MPEG2-TS carrying a bundle of multiple programs is analogous to a "Video Stream Group", i.e. the collection of all RTP streams that carry all components of a group of video streams. The MPEG2-TS is uses dedicated elementary stream(s) to carry the PSI tables which enable the receiver to identify programs and control/descriptor data within the MPEG2-TS stream. The Direct over RTP protocol family uses the Session Description Protocol (SDP), defined in RFC 4566, to put an order in RTP streams, e.g. associate each RTP stream with a specific content or grouping of related RTP streams.

HDCP Interface Independent Adaptation Specification	HDCP Direct Adaptation Specification
Elementary stream	RTP Stream
Program	Video Stream
MPEG2-TS	Video Stream Group
PES	RTP Stream Payload
PES Packet	HDCP Data Unit (HDU)
MPEG2-TS PSI Tables	SDP Protocol

Table 1: Analogy between MPEG2-TS and "Direct over RTP"

Implementations must include all elements of the content protection system described herein and in the High-bandwidth Digital Content Protection System, Interface Independent Adaptation, Revision 2.3 ("HDCP Interface Independent Adaptation, Revision 2.3"), unless the element is specifically identified as informative or optional. Where the mandatory or optional requirements specified in the HDCP Interface Independent Adaptation, Revision 2.3 and this specification are different, the mandatory or optional requirements specified in this specification take precedence. Adopters must also ensure that implementations satisfy the robustness and compliance rules described in the technology license.

1.2 Definitions

RTP Stream. A real time transport stream, composed of RTP packets (RFC 3550), carrying a single type of content (i.e. Audio, Video, Metadata) as its payload. The Content Type of each RTP stream is identified by the PT (Payload Type) 7-bits field in the RTP header, dynamically assigned via SDP. The encapsulation rules of each content type over RTP packets (e.g. uncompressed video pixels, JPEG2000 compressed Codestream, H.264

Compressed video, Audio PCM, Compressed Audio) is associated with a specific “RTP Profile” definition, usually defined by a dedicated IETF RFC document, indicating a structured format for encapsulating the specific content as a payload in RTP packets. RTP Packets of RTP stream are usually encapsulated in UDP protocol. RTP packets are associated with a specific RTP stream by the source and destination IP addresses, by the source and destination UDP ports, and by the PT field in the RTP header.

Audio Packet-time. Data unit composed of concatenation of all audio-words that are related to a specified unit of time, e.g. 1ms, 125us, 250 us, etc. For a multi-channel Audio stream, the Audio Packet-time includes all Audio-words of all channels that are related to the time unit.

HDCP Data Unit (HDU). HDU is a Data Unit composed of one or more “Application Data Units” (ADU), as defined by [18] and referenced by RFC 3550 (RTP Framework). HDU is a unit of encoded video or encoded audio content, meaningful to video or audio application on one hand (for processing, manipulation, and presentation), and used as a synchronization point for HDCP receiver on the other hand, to acquire and/or validate the IV Counter values. Video HDU is usually the encoded data of a complete video frame. Audio HDU is usually the encoded data of Audio Packet-time. “PES Packet” in MPEG2-TS is analogous to HDU for Direct over RTP protocol family.

1.5 References

[15] RFC 8285: A General Mechanism for RTP Header Extensions, October 2017
[16] RFC 4175: RTP Payload Format for Uncompressed Video, September 2005
[17] RFC 3190: RTP Payload Format for 12-bit DAT Audio and 20- and 24-bit Linear Sampled Audio, January 2002
[18] Clark, D. and D. Tennenhouse, “Architectural Considerations for a New Generation of Proto-cols," in SIGCOMM Symposium on Communications Architectures and Protocols, (Philadel-phia, Pennsylvania), pp. 200{208, IEEE Computer Communications Review, Vol. 20(4), September 1990.

2 Authentication Protocol

2.6 Link Synchronization

After successful completion of SKE, HDCP Encryption is enabled and encrypted content starts to flow between the HDCP Transmitter and the HDCP Receiver. As explained in Section 3.4, the presence of the RTP Header Extension carrying the HDCP Full/Short IV Counters Refreshing, indicates that HDCP Encryption is enabled and the HDU payload is encrypted. Once encrypted content starts to flow, a periodic Link Synchronization is performed to maintain cipher synchronization between the HDCP Transmitter and the HDCP Receiver.

Link Synchronization is achieved every time a new HDU is transmitted, by the inclusion of the 64-bit *inputCtr* and 32-bit *streamCtr* in the RTP Header extension, carrying the HDCP synchronization header. Partial Link Synchronization, enabled to maintain synchronization after loss of an RTP Packet(s), is provided within each RTP Packet by inclusion of the 24 least significant bits of the *inputCtr* in the RTP Header extension (See Section 3.4 for details about *inputCtr* and *streamCtr*). The HDCP Receiver updates its *inputCtr* corresponding to the stream (as indicated by the *streamCtr* value) with the *inputCtr* value received from the transmitter.

3 HDCP Encryption

3.1 Description

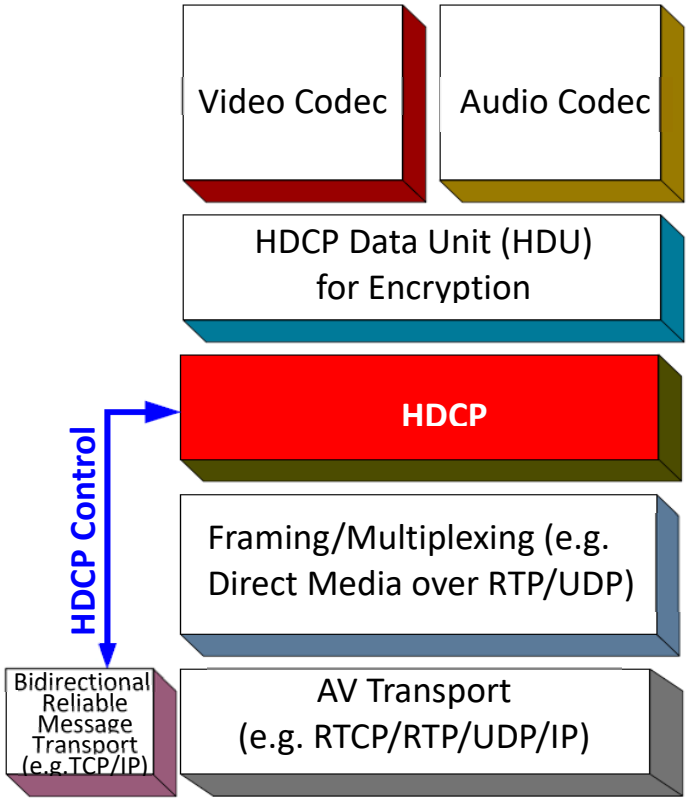

Figure 3.1. Transport Protocol w. HDCP Block Diagram (Informative)

Video in the HDCP Transmitter, together with any associated audio or data streams, are carried as independent RTP streams, as specified in [8] and by the appropriate RTP Profile (e.g. RFC 4175 for uncompressed video or RFC 3190 for PCM audio). Each RTP stream is encrypted as specified in Section 3.4 of this amendment. One or more RTP streams, together with timing and any other ancillary information, may be multiplexed using UDP [4] Protocol mechanisms to deliver a complete video stream with all its related components. Control and Status messages are transported over a reliable bidirectional mechanism, e.g., as specified in [5] and [9].

3.2 AV Stream

Video Stream consists of RTP streams, each carrying one component of a Video Stream. Associated RTP streams are grouped into a Video Stream. Aside from the AV streams, various control, status and timing and formatting information are also transported. Only the AV streams are subject to HDCP Encryption.

Note: An RTP Stream may contain compressed or uncompressed audio or video content. The compressed content may be encoded by one of the standard codecs or a derivative of a standard codec, or it may contain non-standard codec data if appropriate RTP Profile is defined for it.

3.4 HDCP Cipher

The HDCP cipher consists of a 128-bit AES module that is operated in a Counter (CTR) mode as illustrated in Figure 3.2.

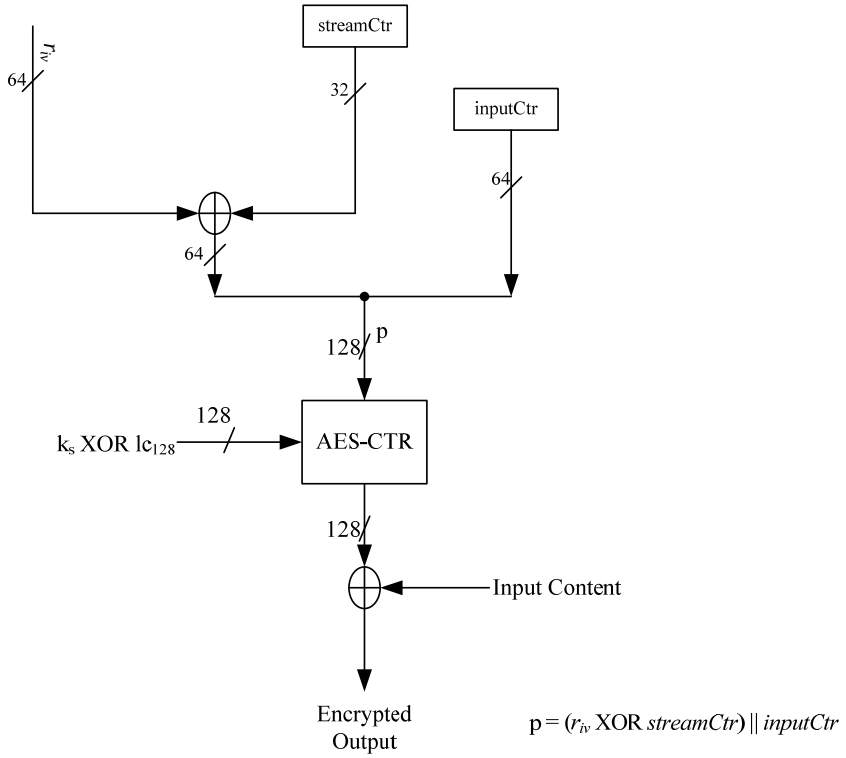

Figure 3.2. HDCP Cipher Structure

k_s is the 128-bit Session Key which is XORed with lc_{128}. RTP streams within a given Video Stream or across Video Stream Group may use the same k_s and r_{iv}.

p = (r_{iv} XOR *streamCtr*) || *inputCtr*. All values are in big-endian order.

streamCtr is a 32-bit counter. The HDCP Transmitter assigns a distinct *streamCtr* value for each RTP stream . The *streamCtr* value is distinct for RTP streams within a given Video Stream and across Video Stream Group i.e. no two RTP streams composing a given Video Stream or different Video Streams can have the same *streamCtr* if those RTP streams share the same k_s and r_{iv}. The HDCP Transmitter assigns *streamCtr* values for video and audio portions of Audiovisual Content as per the following guidelines. The HDCP Transmitter assigns *streamCtr* values where the least significant bit is zero to the video RTP streams. It assigns *streamCtr* values where the least significant bit is one to the audio RTP streams. *streamCtr* is initialized to zero after SKE and it must not be reset at any other time. It is XORed with the least significant 32-bits of r_{iv}.

If AKE_Transmitter_Info.TRANSMITTER_CONTENT_CATEGORY_SUPPORT bit is set, the HDCP receiver must verify that the assignment of *streamCtr* complies with the guidelines described above. The HDCP Receiver must not decrypt the received HDCP Content if the streamCtr assignment does not comply with the guidelines.

inputCtr is a 64-bit counter. It is initialized to zero after SKE and must not be reset at any other time. Each RTP stream within a given Video Stream is associated with its own *inputCtr*.

HDCP Encryption must be applied to RTP Stream payload data; RTP Headers and RTP Payload Header must not be encrypted.

During HDCP Encryption, the key stream produced by the AES-CTR module is XORed with 128-bit (16 Byte) block of payload data to produce the 128-bit encrypted output. *inputCtr* associated with an RTP stream is incremented by one following encryption of 128-bit block of payload data for that stream. The value of *inputCtr* must never be reused for a given set of encryption parameters i.e. k_s, r_{iv} and *streamCtr*.

The 16 Byte encryption block boundary must be aligned with the start of each HDU, as well as the start of each RTP packet. If size of the encrypted contents of the RTP packet is not an integer multiple of 16 Bytes, the unused key stream bits produced by the AES-CTR module must be discarded, and not carried over to a subsequent RTP packet.

Bit ordering is such that the most-significant bit of the 128-bit key stream produced by AES-CTR module is XORed with the first bit in time (as defined in [8]) in the 16 Byte payload data block.

Any HDU containing an HDCP encrypted payload must be sent over the network together with its HDCP header. The HDCP Header must contain the HDU IV Counters, i.e. the value of *streamCtr* for that stream, and the value of *inputCtr* used to encrypt the first 16 Byte block of the HDU payload, as shown in Table 3.1.

Syntax	**No. of Bits**	**Identifier**

IV Counters () {
streamCtr[31..0]
inputCtr[63..0]
}
	
	32
	64
		
		bslbf
		bslbf

Table 3.1 IV Counters of an HDCP Header

3.4.1 HDCP Data Unit (HDU) for Encryption

HDCP Data Unit (HDU) is an HDCP synchronization unit, associated with the IV Counter values used to encrypt the first 16-byte block of the HDU. An HDU for a Video stream is composed of the encoded video data of a complete Video Frame. An HDU for an Audio stream is composed of all Audio-Words related to a single Packet-Type unit (refer to 1.2 for packet-time definition). An HDU is transmitted over the network together with its IV Counters enabling an HDCP Receiver, that just joined a video session, to start decryption of a the received HDU. The HDU IV Counters also enable an HDCP Receiver that is already part of a session to refresh its IV Counters, making sure that the new HDU is decrypted correctly.

This amendment distinguishes between two types of refreshing of the IV Counter; full refreshing and short refreshing. The full IV Counter refreshing is provided together with the first RTP packet that carry a new HDU. The short IV Counter refreshing is provided together with each other RTP packet that does not carry the full refreshing.

The full IV Counter refreshing is composed of 12 bytes, 4 *streamCtr* and 8 *inputCtr*, depicted by Figure 3 and Table 3.1.

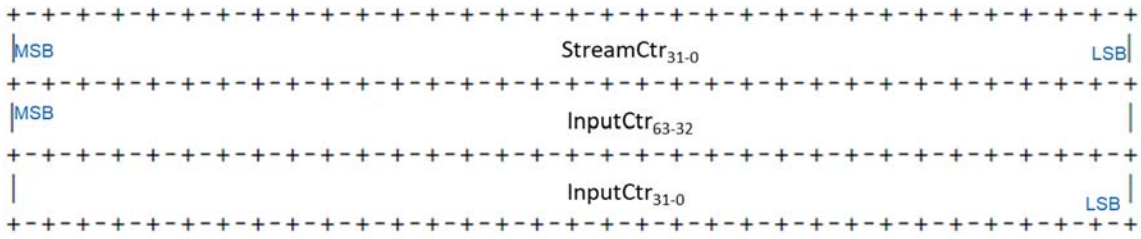

Figure 3: Full IV Counters, 12 bytes, composed of streamCtr and inputCtr

The short IV Counter refreshing is composed of 3 bytes, i.e. the least significant 24 bits of the *inputCtr*, depicted by Figure 4.

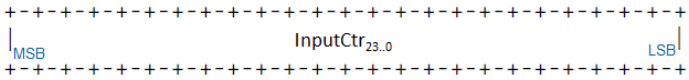

Figure 4: Short IV Counters, 3 bytes, composed of inputCtr$_{23..0}$

The short IV Counter refreshing enables an HDCP Receiver to recover its decryption process after RTP packet failure. Each encoded media, carried over RTP packets, is associated with RTP Profile. Each RTP Profile provide the rules for encapsulating the encoded media in RTP packets. For some encoded media cases, the number of payload bytes encapsulated in RTP Packet may vary in-between RTP packets. Thus, on RTP packet failure, the HDCP Receiver may be unable to guess the IV Counters value for the next RTP packet. The short refresh IV Counter value, provided at the RTP Header Extension of each RTP Packet, enables the Receiver to overcome a synchronization gap applied by the missing RTP Packets.

An HDCP Receiver that is already receiving the media stream can recover the streamCtr value and the 40-MSbits of the InputCtr from the last received full IV Counter refreshing, as follows:

*[**streamCtr$_{31..0}$**] = [Last-Full-Refresh **StreamCtr$_{31..0}$**]*
*If [Last-Full-Refresh **InputCtr$_{23..0}$**] < [short-Refresh **InputCtr$_{23..0}$**] ;compare of two unsigned numbers*
*then [**InputCtr$_{63..24}$**] = [Last-Full-Refresh **InputCtr$_{63..24}$**]*
*else [**InputCtr$_{63..24}$**] = [Last-Full-Refresh **InputCtr$_{63..24}$**]**+1** ;increment of unsigned number, where overflow is not expected in a life of a session*

3.4.2 Encrypted Block Boundary for RTP Encapsulation

The 16 Byte encryption block boundary must be aligned with the start of each RTP Packet. Since a new HDU always start in a new RTP packet, the 16 Byte encryption block boundary is also aligned with the start of HDU. If the size of the encrypted content inserted to RTP Payload data is not an integer multiple of 16 Bytes, the unused key stream bits produced by the AES-CTR module must be discarded, and not carried over to a subsequent RTP packet. The case of non-integral number of 16-byte block in an RTP packet may occur at the end of an HDU (single and/or multiple bulk HDU), and in any occurrence within an HDU where the next HDU byte should be aligned to an RTP packet start (multiple bulk HDU only), as explained in Section 3.4.3 below.

3.4.3 RTP Encapsulation Rules (Informative)

The rules for encapsulating an HDU in RTP packets depends on the encoded media related to the HDU. There are two types of HDU encapsulations:

- Single bulk HDU
- Multiple bulk HDU

Single bulk HDU:
For some encoded media cases, the whole HDU is encapsulated in one or more RTP Packets, where each RTP packet carries the maximum payload allowed for an RTP packet, except for the last RTP packet which may be partially populated. The maximum payload allowed for an RTP packet is reduced to the closest number divided by 16. That is, each of the RTP packets carry integral number of 16-byte blocks, where the last RTP packet may be partially populated with a 16-byte encrypted block. Examples for single bulk HDU cases includes the uncompressed video pixels, the traditional JPEG2000 Codestream, and a PCM Audio packet time.

Multiple bulk HDU:
In other encoded media HDU cases, a frame is divided into stripes or slices to achieve sub-frame latency

between the encoder and decoder over the network. For that purpose, an encoded stripe/slice may be encapsulated in independent set of RTP packets to enable immediate transmission of all RTP packets related to a slice/stripe, without waiting for the next encoded stripe/slice to be ready. Immediate transmission of encoded stripe/slice is the basis to guarantee sub-frame latency with continuous pipeline of video stripes/slices at the HDCP Receiver side, with no underrun or overrun. Thus, the encapsulation rules for encoded stripe/slice is that the first byte of the encoded stripe/slice is aligned to RTP packet start thereby decoupling RTP packets of new stripe/slice from RTP packets of previous stripe/slice. An HDU, composed of multiple encoded stripe/slice units, should be aligned to RTP packet start for each first byte of each encoded stripe/slice unit. Each encoded stripe/slice of an image may be of a different size depending on the CODEC type and the complexity of the image. That is, the number of RTP packets used to carry a single encoded stripe/slice may vary in accordance with the encoded stripe/slice size, where the last RTP packet for that stripe/slice may carry different size of payload with non-integral 16-byte blocks. Examples for such encoded media HDU cases includes the stripe-based JPEG2000 ULL, and the slice-based H.264/H.265 I frame only, providing sub-frame end-to-end latency.

For the multiple bulk HDU case, an HDU is split into sub-HDUs, where each sub-HDU is a single bulk (e.g. an encoded stripe/slice) encapsulated in an independent set of RTP packets. Each sub-HDU is inserted in one or more RTP Packets, depending on its length, where each RTP packet carries an integral number of 16-byte blocks not exceeding the maximum payload allowed for an RTP packet except for the last RTP packet which may be partially populated with non-integral number of 16-byte blocks. The short IV Counters refreshing is mainly useful when the last RTP packet of a sub-HDU failed due to network conditions, as the HDCP Receiver can recover the IV Counters and continue the decryption process correctly.

3.4.4 HDCP Header for RTP Packets

RTP Header Extension technique is used by this amendment to attach a short/full IV counter refresh value to an RTP packet. The “one-byte” Header Extension technique, defined by RFC 8285, shall be used.
Note: RFC 8285 expands the RTP Framework, defined by RFC 3550, to enable appending of multiple RTP extension types to a single RTP packet.

The RTP Header Extension for full refresh IV counters, as depicted in Figure 5, adds a total of 20 bytes to the RTP packet.

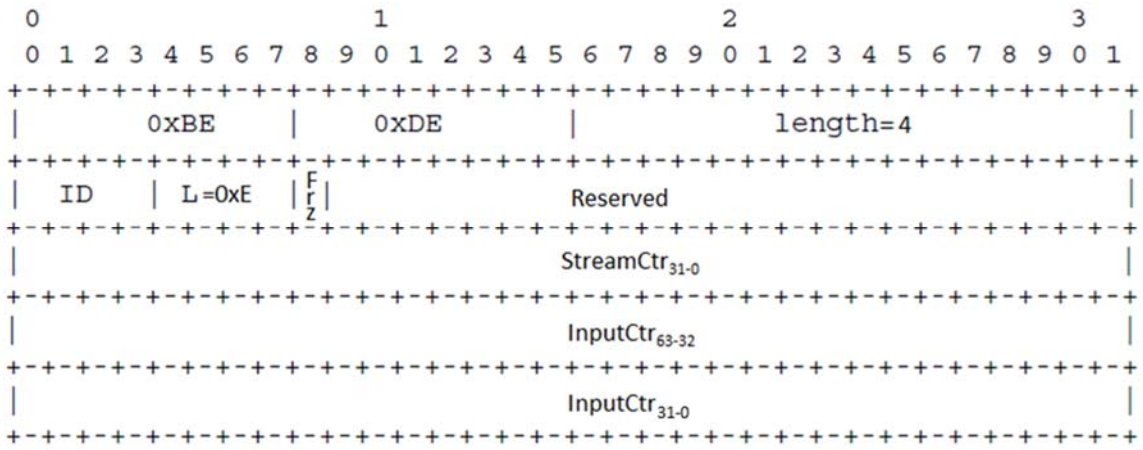

Figure 5: HDCP Header encapsulated in RTP Extension per RFC 8285

The ID field is dynamically assigned using the SDP protocol. A well know URN should be assigned for each type of one-byte RTP extension, as defined by RFC 8285. The following URN shall be used to associate an ID

value for the Full IV Counters refresh:

urn:ietf:params:rtp-hdrext:HDCP-Full-IV-Counter-metadata

Note: this URN should be formally approved by IETF/IANA

Table 3 elaborates each field of the RTP Packet extension for Full IV Counters refreshing.

Field	Description
"Defined by Profile"	= 0xBEDE . 16-bits field Defined by RFC 3550 for general Extension ID. RFC 8285 have reserved a constant value of 0xBEDE for "one-byte" extension.
Length	16-bits. Indicate the number of 32-bit words composing all extensions integrated to the RTP packet. In the example above, only one extension is included, and the three last bytes are padded to 32-bit alignment.
ID	4-bits. Local ID of the one-byte extension. The value of the ID is coordinated via SDP statement: "a=extmap:<value>["/"<direction>] <URI> <extensionattributes>" Where: <value>: Local ID, 1-14 are allowed, 0: reserved for padding, 15: reserved for future use. <direction>: "sendonly", "recvonly", "sendrecv", or "inactive" (without the quotes) <URI>: SHOULD be a URN starting with "urn:ietf:params:rtp-hdrext:" followed by a registered, descriptive name. Proposed URN: **urn:ietf:params:rtp-hdrext:HDCP-Full-IV-Counter-metadata**
L	4-bits. The length of the one-byte extension minus 1. The HDCP Header is of 12 bytes length, thus, a value of 0xE is interpreted as 12 bytes
Frz	1-bit. When set, indicates that the current HDU is not encrypted and a Freeze of the IV Counters. Useful for HDMI AVMUTE implementation
StreamCtr	32 bits of streamCtr
InputCtr	64 bits of inputCtr
Pad	3 x zero bytes for padding to align the extension to a 32-bit word.

Table 3: HDCP Header for full IV refresh counters encapsulated in RTP Extension field

Figure 6 show an example of a single one-byte RTP extension, as defined by RTC 8285, carrying the HDCP short refresh IV Counters, i.e. 24 least significant bits of the *InputCtr*.

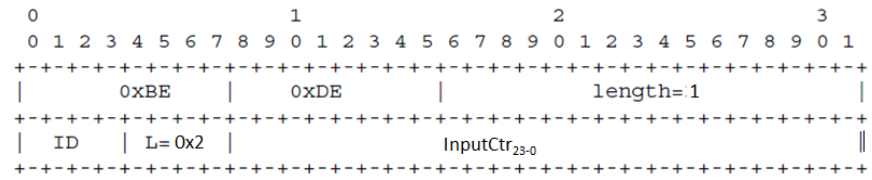

Figure 6: Example for RTP One-Byte Extension, per RFC 8285, for Short IV Counters refresh

The following URN shall be used to associate an ID value for the short IV Counters refreshing:

urn:ietf:params:rtp-hdrext:HDCP-Short-IV-Counter-metadata

Note: this URN should be formally approved by IETF/IANA

The Short Refresh RTP Extension is composed of 8 Bytes. The fields are described by Table 4.

Field	Description
"Defined by Profile"	= 0xBEDE . 16-bits field Defined by RFC 3550 for general Extension ID. RFC 8285 have reserved a constant value of 0xBEDE for "one-byte" extension.
Length	16-bits. Indicate the number of 32-bit words composing all extensions integrated to the RTP packet. In the example above, only one extension is included, and the three last bytes are padded to 32-bit alignment.
ID	4-bits. Local ID of the one-byte extension. The value of the ID is coordinated via SDP statement: "a=extmap:<value>["/"<direction>] <URI> <extensionattributes>" Where: <value>: Local ID, 1-14 are allowed, 0: reserved for padding, 15: reserved for future use. <direction>: "sendonly", "recvonly", "sendrecv", or "inactive" (without the quotes) <URI>: SHOULD be a URN starting with "urn:ietf:params:rtp-hdrext:" followed by a registered, descriptive name. Proposed URN: **urn:ietf:params:rtp-hdrext:HDCP-Short-IV-Counter-metadata**
L	4-bits. The length of the one-byte extension minus 1. The IV-Counters are of 3 bytes length, thus, a value of 0x2 is interpreted as 3 bytes
InputCtr	24 LSbits of inputCtr

Table 4: HDCP Header for Short IV refresh counters encapsulated in RTP Extension field

Figure 7 to Figure 10 below show various use-cases for HDCP encryption and encapsulation in RTP packets.

a. Example for uncompressed video, in accordance with RFC 4175 (refer to Figure 7):

HDU in this case is a concatenation of byte-aligned pixel-groups of a complete video frame. Prior to encapsulating the HDU in RTP packets, the HDU is encrypted (XORed) by a sequence of 16-byte encryption blocks generated by the AES-CTR Cipher module, where the IV Counters used to generate the first 16-byte encryption block of that sequence are considered as the HDU IV Counters. Since the HDU length may not be an integral multiplication of a 16-byte block, the last 16-byte encryption block generated by the AES-CTR Cipher module is partially utilized and the rest of the bytes, i.e. the un-used bytes of that encryption block, are discarded.

The encrypted HDU, having the exact same length as the unencrypted HDU, is encapsulated in RTP packets. The first RTP packet, carrying the first byte of the encrypted HDU, also carries the HDCP Header with the full IV Counters Refreshing as an RTP Extension. The next RTP packets, carrying the next bytes of the encrypted HDU, also carry the HDCP Header with the short IV Counters Refreshing as an RTP Extension. Each RTP Packet carries an integral number of 16-byte blocks, except for the last RTP packet of that encrypted HDU, which may be partially populated and include non-integral number of 16-byte clocks.

Only the video content is encrypted, and the RTP header, RTP Payload Header and HDCP headers are not encrypted.

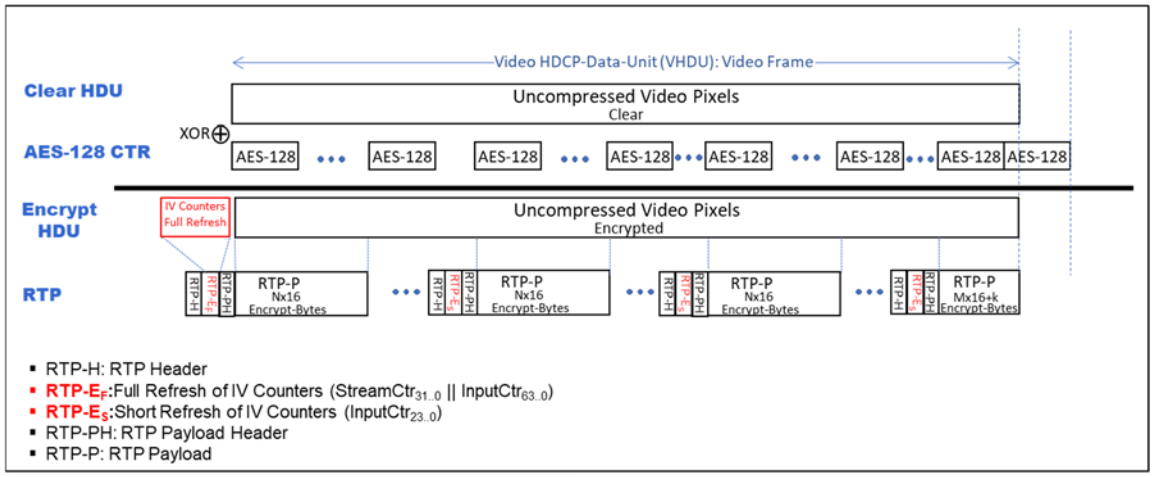

Figure 7: Example for encrypted uncompressed video frame encapsulated in RTP Packets

b. Example for JPEG2000 ULL compressed video with sub-frame end-to-end latency, in accordance with RFC 5371 (refer to Figure 8):

The JPEG2000 ULL encoder divides an image into equal size stripes and generates independent Codestream for each stripe. The RTP Profile, defined by RFC 5371, enables encapsulation of each Codestream sequence of a stripe in to independent RTP Packets and to send them over the network immediately as generated. The HDCP Receiver can start decoding a stripe once it receives the RTP Packets of that stripe and keep the sub-frame delay.

HDU, in this case, is a concatenation of Codestreams of all stripes composing a complete video frame, where each Codestream is considered as a sub-HDU. Prior to encapsulating the HDU in RTP packets, the HDU is encrypted (XORed) by a sequence of 16-byte encryption blocks generated by the AES-CTR Cipher module, where the IV Counters used to generate the first 16-byte encryption block of that sequence are considered as the HDU IV Counters. Since the sub-HDU length may not be an integral multiple of a 16-byte block, the last 16-byte encryption block generated by the AES-CTR Cipher moduleis partially utilized, and the rest bytes, i.e. the un-used bytes of that encryption block, are discarded.

Each encrypted sub-HDU, having the exact same length as the unencrypted sub-HDU, is encapsulated in an independent set of RTP packets. The first RTP packet, carrying the first byte of the HDU, i.e. the first encrypted bytes of the sub-HDU, also carries the HDCP Header with the full IV Counters Refreshing as an RTP Extension. The next RTP packets, carrying the next bytes of the encrypted HDU, also carry the HDCP Header with the short IV Counters Refreshing as an RTP Extension. Each RTP Packet carry an integral number of 16-byte blocks, except for the last RTP packet of each encrypted sub-HDU, which may be partially populated and include non-integral number of 16-byte clocks.

Only the video content is encrypted, and the RTP header, RTP Payload Header and HDCP headers are not encrypted.

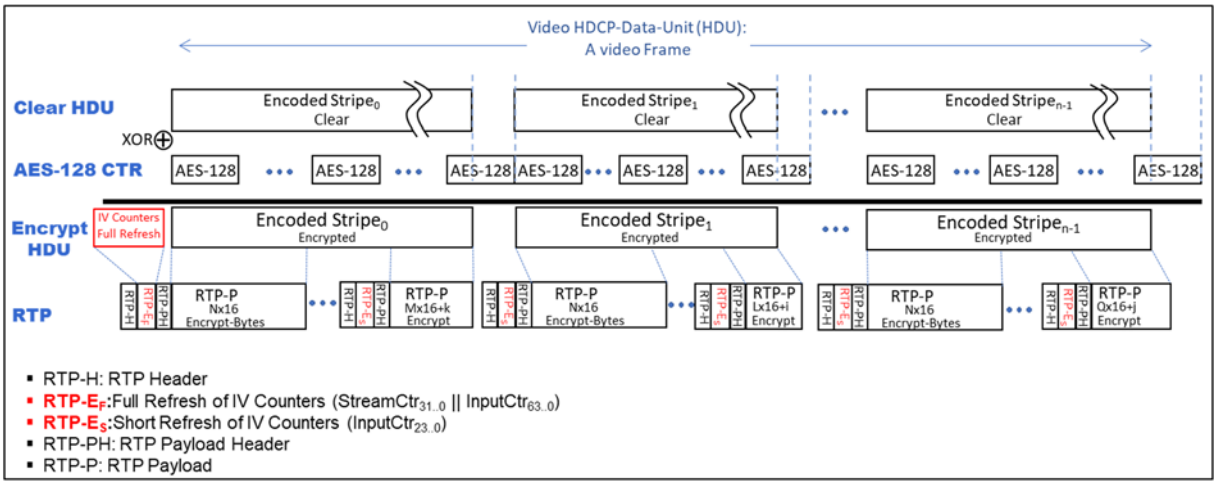

Figure 8: Example for compressed video frame, divided into stripes/slices, encrypted and encapsulated in RTP Packets

c. Example for uncompressed audio, in accordance with RFC 3190 (refer to Figure 9):

HDU in this case is an audio packet-time unit, short enough to be encapsulated in a single RTP packet, i.e. a concatenation of all audio-words related to a unit of time can fit into a single RTP packet (refer to 1.2 for packet-time unit definition). Prior to encapsulating the HDU in a single RTP packet, the HDU is encrypted (XORed) by a sequence of 16-byte encryption blocks generated by the AES-CTR Cipher module, where the IV Counters used to generate the first 16-byte encryption block of that sequence are considered as the HDU IV Counters. Since the HDU length may not be an integral multiplication of 16-byte block, the last 16-byte encryption block generated by the AES-CTR Cipher module is partially utilized, and the rest bytes, i.e. the un-used bytes of that encryption block, are discarded.

The encrypted HDU, having the exact same length as the unencrypted HDU, is encapsulated in a single RTP packet. Each RTP packet carrying a complete encrypted HDU also carries the HDCP Header with the full IV Counters Refreshing as an RTP Extension. The short IV Counters Refreshing is not used in this case. Each RTP Packet may carry non-integral number of 16-byte blocks depending on the audio packet-time unit length.

Only the audio content is encrypted, and the RTP header and HDCP headers are not encrypted. RFC 3190 does not define an RTP Payload Header.

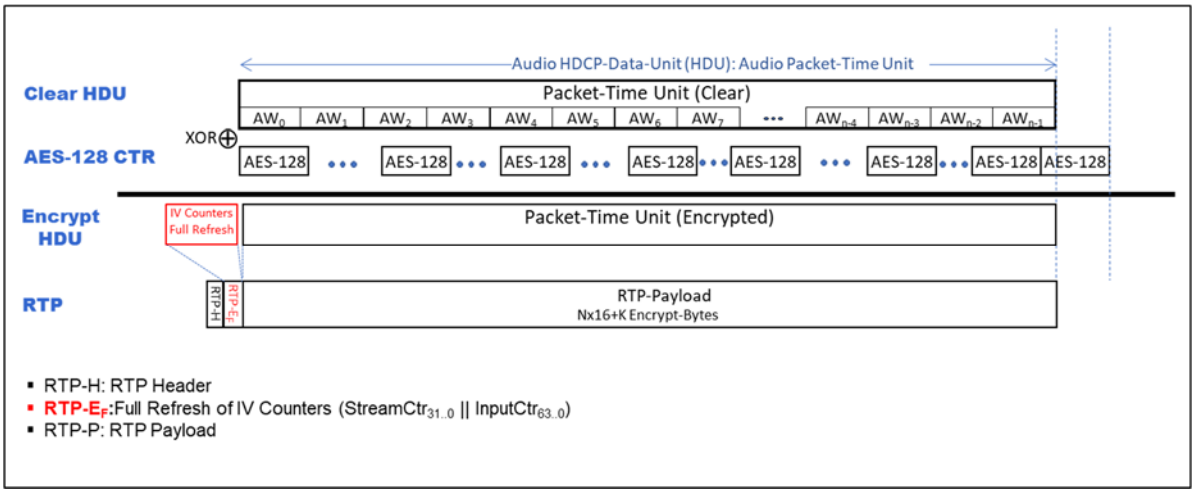

Figure 9: Example for Audio Packet time unit, encrypted and encapsulated in a single RTP Packet

d. Example for uncompressed audio, in accordance with RFC 3190 (refer to Figure 10):

HDU in that case is an audio packet-time unit long enough to be encapsulated in a multiple RTP packet, i.e. a concatenation of all audio-words of all audio channels related to a unit of time should be split between multiple RTP packets (refer to 1.2 for packet-time unit definition). Prior to encapsulating the HDU in RTP packets, the HDU is encrypted (XORed) by a sequence of 16-byte encryption blocks generated by the AES-CTR Cipher module, where the IV Counters used to generate the first 16-byte encryption block of that sequence are considered as the HDU IV Counters. Since the HDU length may not be an integral multiplication of 16-byte block, the last 16-byte encryption block generated by the AES-CTR Cipher module is partially utilized, and the rest bytes, i.e. the un-used bytes of that encryption block, are discarded.

The encrypted HDU, having the exact same length as the unencrypted HDU, is encapsulated in multiple RTP packets. The first RTP packet, carrying the first bytes of the encrypted HDU, also carries the HDCP Header with the full IV Counters Refreshing as an RTP Extension. The next RTP packets, carrying the next bytes of the encrypted HDU, also carry the HDCP Header with the short IV Counters Refreshing as an RTP Extension. Each RTP Packet carry an integral number of 16-byte blocks, except for the last RTP packet of that encrypted HDU, which may be partially populated and include non-integral number of 16-byte clocks.

Only the audio content is encrypted, and the RTP header and HDCP headers are not encrypted. RFC 3190 does not define an RTP Payload Header.

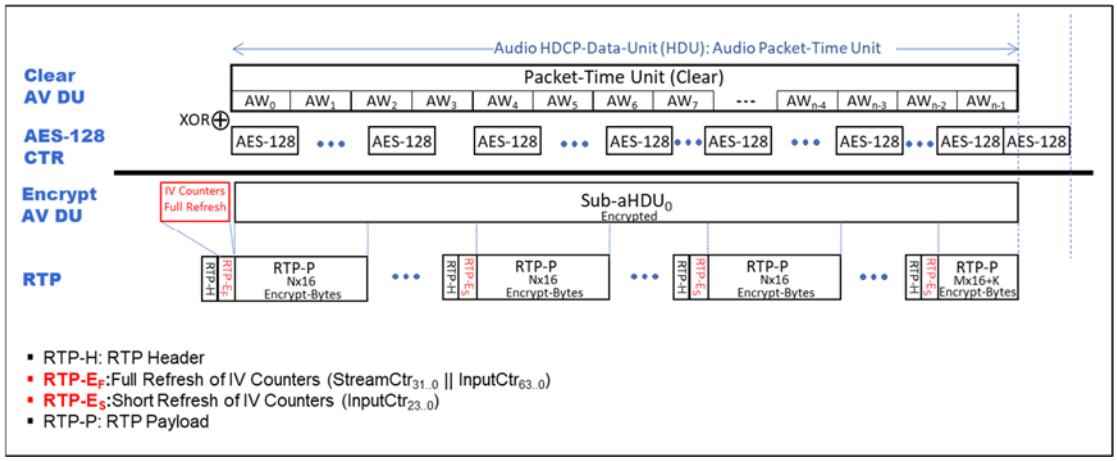

Figure 10: Example for Audio Packet time unit, encrypted and encapsulated in a multiple RTP Packets

3.5 HDCP Cipher Block

The HDCP cipher block consists of multiple HDCP cipher (AES-CTR) modules. The input encryption parameters to each HDCP cipher module satisfy the requirements in Section 3.4 i.e. the *streamCtr* value is distinct for each RTP stream within an HDCP Cipher Block, an *inputCtr* is associated with each RTP stream, the same k_s and r_{iv} is used for encryption of all RTP streams within an HDCP Cipher Block.

Figure 3.3 illustrates an HDCP cipher block used for encryption of multiple video streams related to the same HDCP Cipher Block.

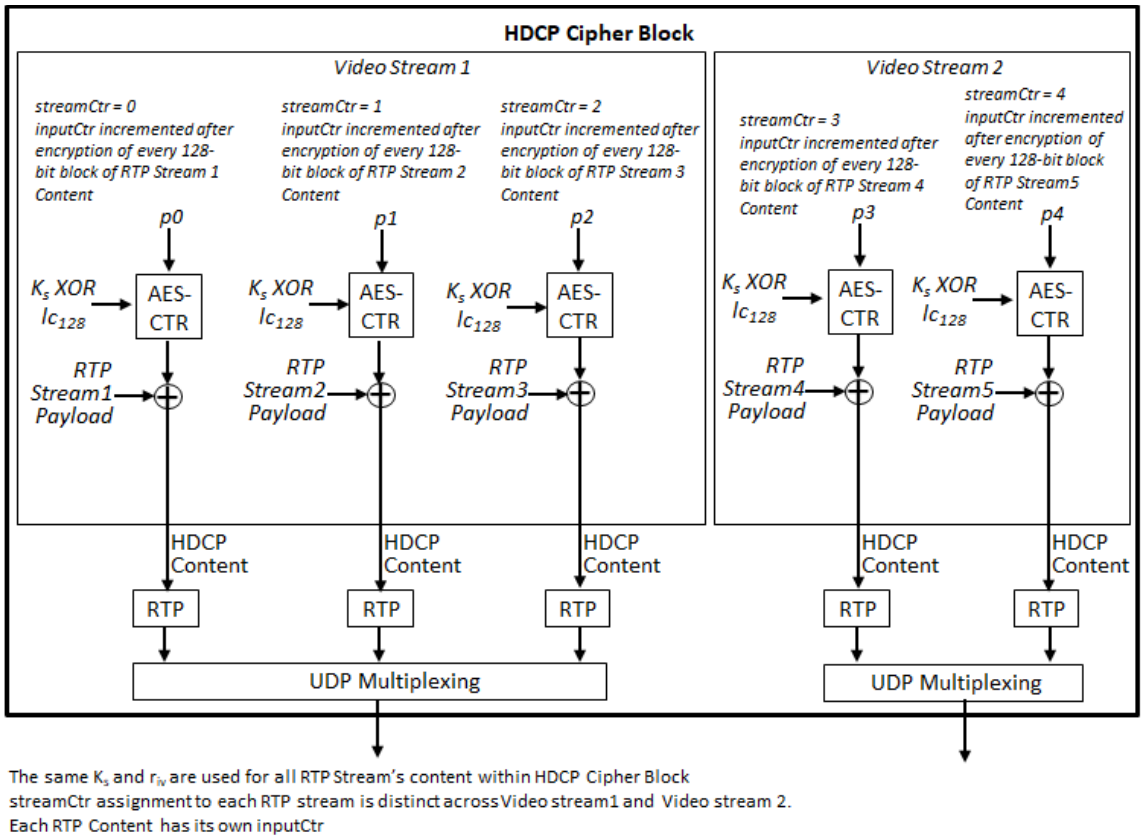

Figure 3.3. HDCP Encryption of Multiple Video Streams

3.6 Video Stream Multiplexing

This section defines procedures used when Video Stream Group is protected by a single HDCP Cipher Block.

3.6.1 HDCP Announcement Descriptor

For Video Streams that multiplex (RTP Streams, the RTP streams subject to HDCP Encryption must include an SDP Announcement descriptor of the form shown below. It serves to indicate that the HDCP Header Extension carrying the Full/Short IV Counters Refreshing is defined by this amendment document.

The inclusion of the SDP Announcement Descriptor is required only when Direct media over RTP is used. The HDCP Transmitter must not include the SDP Announcement descriptor unless it determines that the receiver is HDCP-capable.

Session Description Protocol (SDP) is used by an RTP Sender to inform the session attributes to an RTP Receiver.

The following URNs, recommended by RFC 8285, shall be used to associate the ID field of the "one-byte" RTP extension with the "Full" or "Short" IV Counters Refreshing.

- **urn:ietf:params:rtp-hdrext:HDCP-Full-IV-Counter-metadata**
- **urn:ietf:params:rtp-hdrext:HDCP-Short-IV-Counter-metadata**

The below "a=extmap" SDP clause shall be used to indicate that the RTP Extension, attached to the RTP packets, carry the HDCP Header, i.e. the IV Counters, defined by this document.

"a=extmap:<value>["/"<direction>] <URI> <extensionattributes>"
Where:

- <value>: Local ID, 1-14 are allowed (0: reserved for padding, 15: reserved for future use).
- <direction>: "sendonly" (without the quotes)
- <URI>: One of the URNs above

Note: the "a=extmap" SDP clause shall be sent twice, once with the "short URN" and once with the "Full URN".

3.6.2 Video Stream Group

Direct media over RTP may contain multiple Video Streams. Video Streams subject to HDCP Encryption must include the SDP Announcement descriptor defined in Section 3.6.1. SDP Announcement descriptor should be distributed to each Video Stream Receiver.

Only the HDU payload data shall be encrypted, where control, status, management information, metadata information, RTP Header, RTP Header Extension and RTP payload header must not be encrypted.

A complete AKE, Locality Check and SKE procedure is performed with each Video Stream Receiver, prior to enabling HDCP Encryption for any RTP Stream coming from the transmitter. The same k_s and r_{iv} is used for all Video Streams. Encryption may be enabled and disabled separately for each Video Stream that is encrypted by the single HDCP Cipher Block. Encryption enabling and disabling may occur at the two levels:

- Media Encryption Disable:
  The media streaming is still subject to HDCP, but current video frame or Audio packet-time is not encrypted. This case is applicable for AVMUTE periods, where the video stream enters a period of format change. During this period, the IV Counters are frozen and the media is not encrypted. To implement this case, this document provides the "Frz" bit in the HDCP Full Header (Figure 5).

- HDCP Disable:
  A video stream may be subject to HDCP or may be sent clear, as decided by the video source. When the video source moves from a low-value video to a video that is subject to HDCP, or vice versa, an updated SDP record is sent from HDCP Transmitter to the HDCP Receiver to update the SDP Announcement indicating whether HDCP header is carried (or not) by the RTP extension.

The number of 16-byte encrypted block integrated to an RTP packet should guarantee that the maximum length allowed for an RTP packet is not violated. The 16 Byte encryption block boundary must be aligned with the start of HDU, sub-HDU and RTP packet. If the last block in the encrypted RTP packet is less than 16 Bytes, only the encrypted payload bytes must be transmitted; i.e., the unused key stream bits produced by the AES-CTR module must be discarded, and not carried over to a subsequent packet.

Note: This constraint simplifies packet assembly and parsing.

Note: For RTP Streams, only in the RTP packet containing the end of the HDU or end of sub-HDU does the possibility exist that the last block in the packet might be less than 16 Bytes.

Section 3.6.3 of the HDCP Interface Indepdendent Adaptation Specification, Revision 2.3 is removed from this amendment.

3.7 Uniqueness of k_s and r_{iv}

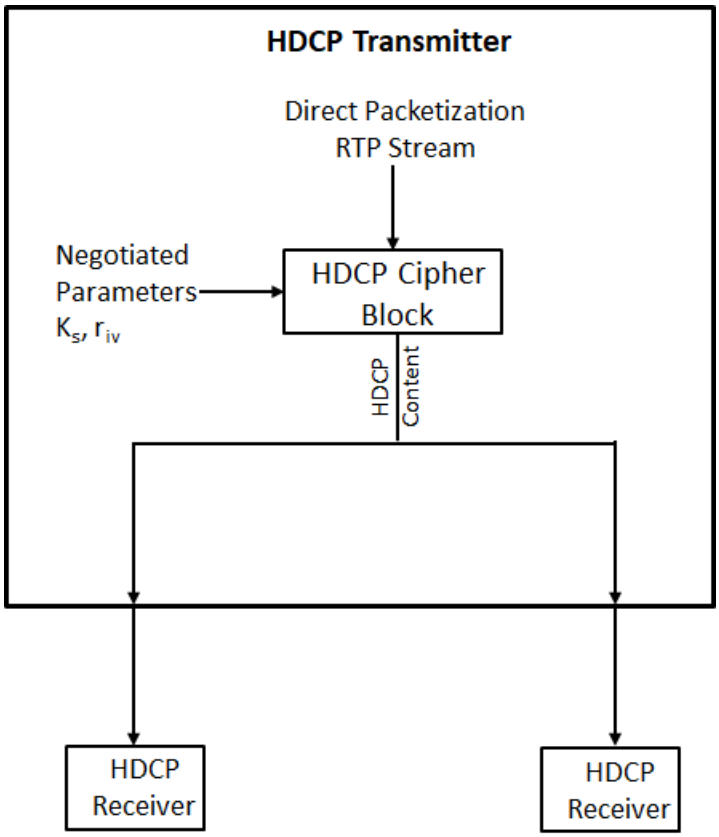

Figure 3.4. k_s and r_{iv} Shared across HDCP Links

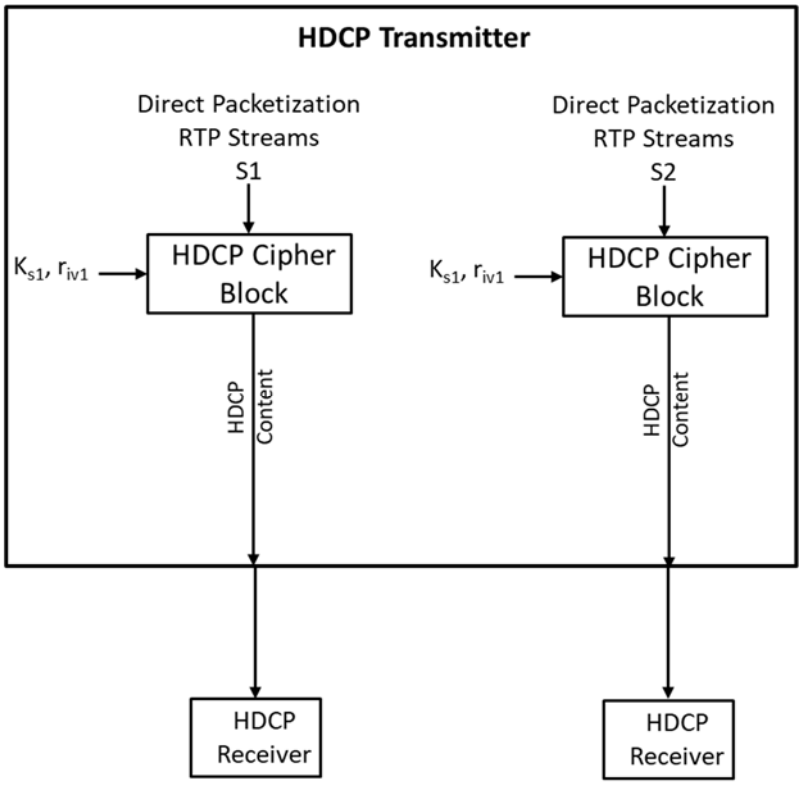

Figure 3.5. Unique k_s and r_{iv} across HDCP Links

4 Authentication Protocol Messages

4.3.15 RepeaterAuth_Stream_Manage (Transmitter to Receiver)

Content Streams may be comprised of audio and video RTP streams. Each RTP stream is assigned a streamCtr value by the HDCP Transmitter which is used during HDCP Encryption of the RTP stream. The ContentStreamID, derived from the RTP stream Identifier, IP Address [5], UDP Ports [4], and RTP Payload Type [8]), for each RTP stream is associated with its corresponding streamCtr in the RepeaterAuth_Stream_Manage message.

Syntax	No. of Bytes	Identifier
RepeaterAuth_Stream_Manage{		
msg_id	1	uint
seq_num_M	3	uint
k	2	uint
for(j=0;j<k;j++) {		
streamCtrj	4	uint
ContentStreamIDj	7	uint
Type	1	uint
}		
}		

Table 4.19. RepeaterAuth_Stream_Manage Payload

Parameter	No. of Bytes	Description
ContentStreamID	7	Content Stream Identification. This parameter must be assigned a concatenated value as follows. ContentStreamID[55:24] = Destination IP Address (32-bit) ContentStreamID[23:8] = Destination UDP Port (16-bit) ContentStreamID[7] = 0 ContentStreamID[6:0] = PT (Payload Type) field of the RTP Header (7-bit)
Type	1	0x00 : Type 0 Content Streams. May be transmitted by the HDCP Repeater to all HDCP Devices. 0x01 : Type 1 Content Streams. Must not be transmitted by the HDCP Repeater to HDCP 1.x-compliant Devices and HDCP 2.0-compliant Repeaters 0x02 – 0xFF : Reserved for future use only. Content Streams with reserved Type values must be treated similar to Type 1 Content Streams i.e. must not be transmitted by the HDCP Repeater to HDCP 1.x-compliant Devices and HDCP 2.0-compliant Repeaters

Table 4.20. RepeaterAuth_Stream_Manage Parameters

Note: To guarantee uniqueness of the ContentStreamID, the 32-bit Destination IP Address shall be:

- IPv4: a 32-bit IPv4 Multicast Address.
- IPv6: the 32-LSbits of an IPv6 Multicast Address.