High-bandwidth Digital Content Protection System

Mapping HDCP to WirelessHD

Revision 2.2

18 July, 2013

Notice

Acknowledgement

Silicon Image and SiBEAM, Inc. have contributed to the development of this specification.

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

January 19th, 2010 - 2.0 Revision. Publication on DCP LLC website

1 Introduction ... 5
1.1 Scope ... 5
1.2 Definitions ... 5
1.3 Overview ... 8
1.4 Terminology ... 9
1.5 References ... 9

2 Authentication Protocol ... 11
2.1 Overview ... 11
2.2 Authentication and Key Exchange ... 12
2.2.1 Pairing ... 16
2.2.2 Pairing key encryption. ... 17
2.3 Locality Check ... 17
2.4 Session Key Exchange ... 19
2.5 Authentication with Repeaters ... 19
2.5.1 Upstream Propagation of Topology Information ... 20
2.5.2 Downstream Propagation of Content Stream Management Information ... 26
2.6 Link Synchronization ... 27
2.7 Key Derivation ... 27
2.8 HDCP Transmitter State Diagram ... 28
2.9 HDCP Receiver State Diagram ... 34
2.10 HDCP Repeater State Diagrams ... 35
2.10.1 Propagation of Topology Errors ... 36
2.10.2 HDCP Repeater Downstream State Diagram ... 36
2.10.3 HDCP Repeater Upstream State Diagram ... 40
2.11 Converters ... 44
2.11.1 HDCP 2 – HDCP 1.x Converters ... 44
2.11.2 HDCP 1.x – HDCP 2 Converters ... 46
2.12 Session Key Validity ... 47
2.13 Random Number Generation ... 48

3 HDCP Encryption ... 49
3.1 Description ... 49
3.2 AV Stream ... 49
3.3 HDCP Cipher ... 49
3.4 Uniqueness of k_s and r_{iv} ... 51

4 Authentication Protocol Messages ... 53
4.1 Abbreviations ... 53
4.2 HDCP Control / Status Stream in WirelessHD ... 53
4.3 Message Format ... 54
4.3.1 AKE_Init (Transmitter to Receiver) ... 54
4.3.2 AKE_Send_Cert (Receiver to Transmitter) ... 54
4.3.3 AKE_No_Stored_km (Transmitter to Receiver) ... 55
4.3.4 AKE_Stored_km (Transmitter to Receiver) ... 55
4.3.5 AKE_Send_rrx (Receiver to Transmitter) ... 55
4.3.6 AKE_Send_H_prime (Receiver to Transmitter) ... 55
4.3.7 AKE_Send_Pairing_Info (Receiver to Transmitter) ... 55
4.3.8 LC_Init (Transmitter to Receiver) ... 56
4.3.9 RTT_Ready (Receiver to Transmitter) ... 56
4.3.10 RTT_Challenge (Transmitter to Receiver) ... 56
4.3.11 RTT_Response (Receiver to Transmitter) ... 56
4.3.12 SKE_Send_Eks (Transmitter to Receiver) ... 56
4.3.13 RepeaterAuth_Send_ReceiverID_List (Receiver to Transmitter) ... 57
4.3.14 RepeaterAuth_Send_Ack (Transmitter to Receiver) ... 58
4.3.15 RepeaterAuth_Stream_Manage (Transmitter to Receiver) ... 58

4.3.16 RepeaterAuth_Stream_Ready (Receiver to Transmitter) 59
4.3.17 Receiver_AuthStatus (Receiver to Transmitter) 59

5 Renewability 61
5.1 SRM Size and Scalability 62
5.2 Updating SRMs 64

Appendix A. Core Functions and Confidentiality and Integrity of Values 65

Appendix B. DCP LLC Public Key 68

Appendix C. Bibliography (Informative) 69

1 Introduction

1.1 Scope

This specification describes a WirelessHD mapping of the High-bandwidth Digital Content Protection (HDCP) system, Revision 2.20. This specification is applied on implementations of the WirelessHD™ interface as explained in subsequent chapters.

In an HDCP System, two or more HDCP Devices are interconnected through an HDCP-protected Interface. The Audiovisual Content flows from the Upstream Content Control Function into the HDCP System at the most upstream HDCP Transmitter. From there the Audiovisual Content encrypted by the HDCP System, referred to as HDCP Content, flows through a tree-shaped topology of HDCP Receivers over HDCP-protected Interfaces. This specification describes a content protection mechanism for: (1) authentication of HDCP Receivers to their immediate upstream connection (i.e., an HDCP Transmitter), (2) revocation of HDCP Receivers that are determined by the Digital Content Protection, LLC, to be invalid, and (3) HDCP Encryption of Audiovisual Content over the HDCP-protected Interfaces between HDCP Transmitters and their downstream HDCP Receivers. HDCP Receivers may render the HDCP Content in audio and visual form for human consumption. HDCP Receivers may be HDCP Repeaters that serve as downstream HDCP Transmitters emitting the HDCP Content further downstream to one or more additional HDCP Receivers.

Unless otherwise specified, the term "HDCP Receiver" is also used to refer to the upstream HDCP-protected interface port of an HDCP Repeater. Similarly, the term "HDCP Transmitter" is also used to refer to the downstream HDCP-protected interface port of an HDCP Repeater.

Except when specified otherwise, HDCP 2.2-compliant Devices must interoperate with other HDCP 2.2-compliant Devices, HDCP 2.1-compliant and HDCP 2.0-compliant Devices connected to their HDCP-protected Interface Ports using the same protocol. HDCP Transmitters must support HDCP Repeaters.

The state machines in this specification define the required behavior of HDCP Devices. The link-visible behavior of HDCP Devices implementing the specified state machines must be identical, even if implementations differ from the descriptions. The behavior of HDCP Devices implementing the specified state machines must also be identical from the perspective of an entity outside of the HDCP System.

Implementations must include all elements of the content protection system described herein, unless the element is specifically identified as informative or optional. Adopters must also ensure that implementations satisfy the robustness and compliance rules described in the technology license. Additionally, HDCP Transmitters may be subject to additional robustness and compliance rules associated with other content protection technologies.

Device discovery and association, and link setup and teardown, is outside the scope of this specification.

1.2 Definitions

The following terminology, as used throughout this specification, is defined as herein:

Audiovisual Content. Audiovisual works (as defined in the United States Copyright Act as in effect on January 1, 1978), text and graphic images, are referred to as *AudioVisual Content*.

Authorized Device. An HDCP Device that is permitted access to HDCP Content is referred to as an *Authorized Device*. An HDCP Transmitter may test if a connected HDCP Receiver is an Authorized Device by successfully completing the following stages of the authentication protocol – Authentication and Key Exchange (AKE) and Locality check. If the authentication protocol successfully results in establishing authentication, then the other device is considered by the HDCP Transmitter to be an Authorized Device.

Content Stream. A *Content Stream* consists of Audiovisual Content received from an Upstream Content Control Function that is to be encrypted and Audiovisual Content received from an Upstream Content Control Function that is encrypted by the HDCP System.

Device Key Set. An HDCP Receiver has a Device Key Set, which consists of its corresponding Device Secret Keys along with the associated Public Key Certificate.

Device Secret Keys. For an HDCP Transmitter, Device Secret Key consists of the secret Global Constant. For an HDCP Receiver, Device Secret Keys consists of the secret Global Constant and the RSA private key. The Device Secret Keys are to be protected from exposure outside of the HDCP Device.

downstream. The term, *downstream*, is used as an adjective to refer to being towards the sink of the HDCP Content. For example, when an HDCP Transmitter and an HDCP Receiver are connected over an HDCP-protected Interface, the HDCP Receiver can be referred to as the *downstream* HDCP Device in this connection. For another example, on an HDCP Repeater, the HDCP-protected Interface Port(s) which can emit HDCP Content can be referred to as its *downstream* HDCP-protected Interface Port(s). See also, *upstream*.

Global Constant. A 128-bit random, secret constant provided only to HDCP adopters and used during HDCP Content encryption or decryption

HDCP 1.x. *HDCP 1.x* refers to, specifically, the variant of HDCP described by Revision 1.00 and higher versions along with their associated errata, if applicable..

HDCP 1.x-compliant Device. An HDCP Device that is designed in adherence to HDCP 1.x, defined above, is referred to as an *HDCP 1.x-compliant Device*.

HDCP 2. *HDCP 2* refers to, specifically, the variant of HDCP mapping for all HDCP protected interfaces described by Revision 2.00 and higher versions along with their associated errata, if applicable.

HDCP 2.0. *HDCP 2.0* refers to, specifically, the variant of HDCP mapping described by Revision 2.00 of this specification along with its associated errata, if applicable.

HDCP 2.0-compliant Device. An HDCP Device that is designed in adherence to HDCP 2.0 is referred to as an *HDCP 2.0-compliant Device*.

HDCP 2.1. *HDCP 2.1* refers to, specifically, the variant of HDCP mapping described by Revision 2.10 of HDCP Interface Independent Adaptation Specification along with its associated errata, if applicable.

HDCP 2.1-compliant Device. An HDCP Device that is designed in adherence to HDCP 2.1 is referred to as an *HDCP 2.1-compliant Device.*

HDCP 2.2. *HDCP 2.2* refers to, specifically, the variant of HDCP mapping described by Revision 2.20 of this specification along with its associated errata, if applicable.

HDCP 2.2-compliant Device. An HDCP Device that is designed in adherence to HDCP 2.2 is referred to as an *HDCP 2.2-compliant Device.*

HDCP Content. *HDCP Content* consists of Audiovisual Content that is protected by the HDCP System. *HDCP Content* includes the Audiovisual Content in encrypted form as it is transferred from an HDCP Transmitter to an HDCP Receiver over an HDCP-protected Interface, as well as any translations of the same content, or portions thereof. For avoidance of doubt, Audiovisual Content that is never encrypted by the HDCP System is not *HDCP Content*.

HDCP Device. Any device that contains one or more HDCP-protected Interface Port and is designed in adherence to HDCP is referred to as an *HDCP Device*.

HDCP Encryption. *HDCP Encryption* is the encryption technology of HDCP when applied to the protection of HDCP Content in an HDCP System.

HDCP Receiver. An HDCP Device that can receive and decrypt HDCP Content through one or more of its HDCP-protected Interface Ports is referred to as an *HDCP Receiver*.

HDCP Repeater. An HDCP Device that can receive and decrypt HDCP Content through one or more of its HDCP-protected Interface Ports, and can also re-encrypt and emit said HDCP Content through one or more of its HDCP-protected Interface Ports, is referred to as an *HDCP Repeater*. An *HDCP Repeater* may also be referred to as either an HDCP Receiver or an HDCP Transmitter when referring to either the upstream side or the downstream side, respectively.

HDCP Session. An *HDCP Session* is established between an HDCP Transmitter and HDCP Receiver with the transmission or reception of r_{tx} as part of the authentication initiation message, AKE_Init. The established *HDCP Session* remains valid until it is aborted by the HDCP Transmitter or a new *HDCP Session* is established, which invalidates the *HDCP Session* that was previously established, by the transmission or reception of a new r_{tx} as part of the AKE_Init message.

HDCP System. An *HDCP System* consists of an HDCP Transmitter, zero or more HDCP Repeaters and one or more HDCP Receivers connected through their HDCP-protected interfaces in a tree topology; whereas the said HDCP Transmitter is the HDCP Device most upstream, and receives the Audiovisual Content from one or more Upstream Content Control Functions. All HDCP Devices connected to other HDCP Devices in an *HDCP System* over HDCP-protected Interfaces are part of the *HDCP System*.

HDCP Transmitter. An HDCP Device that can encrypt and emit HDCP Content through one or more of its HDCP-protected Interface Ports is referred to as an *HDCP Transmitter*.

HDCP. *HDCP* is an acronym for High-bandwidth Digital Content Protection. This term refers to this content protection system as described by any revision of this specification and its errata.

HDCP-protected Interface Port. A logical connection point on an HDCP Device that supports an HDCP-protected Interface is referred to as an *HDCP-protected Interface Port*. A single connection can be made over an *HDCP-protected Interface Port*.

HDCP-protected Interface. An interface for which HDCP applies is described as an *HDCP-protected Interface*.

Master Key. A 128-bit random, secret cryptographic key negotiated between the HDCP Transmitter and the HDCP Receiver during Authentication and Key Exchange and used to pair the HDCP Transmitter with the HDCP Receiver.

Public Key Certificate. Each HDCP Receiver is issued a Public Key Certificate signed by DCP LLC, and contains the Receiver ID and RSA public key corresponding to the HDCP Receiver.

Receiver Connected Indication. An indication to the HDCP Transmitter that an active receiver has been connected to it. The format of the indication or the method used by the HDCP Transmitter to connect to or disconnect from a receiver is outside the scope of this specification.

Receiver_Disconnected Indication. An indication to the HDCP Transmitter that the receiver has been disconnected from it. The format of the indication or the method used by the HDCP Transmitter to connect to or disconnect from a receiver is outside the scope of this specification.

Receiver ID. A 40-bit value that uniquely identifies the HDCP Receiver. It has the same format as an HDCP 1.x KSV i.e. it contains 20 ones and 20 zeroes.

Session Key. A 128-bit random, secret cryptographic key negotiated between the HDCP Transmitter and the HDCP Receiver during Session Key exchange and used during HDCP Content encryption or decryption.

Upstream Content Control Function. The HDCP Transmitter most upstream in the HDCP System receives Audiovisual Content to be protected from the *Upstream Content Control Function*. The *Upstream Content Control Function* is not part of the HDCP System, and the methods used, if any, by the *Upstream Content Control Function* to determine for itself the HDCP System is correctly authenticated or permitted to receive the Audiovisual Content, or to transfer the Audiovisual Content to the HDCP System, are beyond the scope of this specification. On a personal computer platform, an example of an *Upstream Content Control Function* may be software designed to emit Audiovisual Content to a display or other presentation device that requires HDCP.

upstream. The term, *upstream*, is used as an adjective to refer to being towards the source of the HDCP Content. For example, when an HDCP Transmitter and an HDCP Receiver are connected over an HDCP-protected Interface, the HDCP Transmitter can be referred to as the *upstream* HDCP Device in this connection. For another example, on an HDCP Repeater, the HDCP-protected Interface Port(s) which can receive HDCP Content can be referred to as its *upstream* HDCP-protected Interface Port(s). See also, *downstream*.

1.3 Overview

HDCP is designed to protect the transmission of Audiovisual Content between an HDCP Transmitter and an HDCP Receiver. The HDCP Transmitter may support simultaneous connections to HDCP Receivers through one or more of its HDCP-protected interface ports. The system also allows for HDCP Repeaters that support downstream HDCP-protected Interface Ports. The HDCP System allows up to four levels of HDCP Repeaters and as many as 32 total HDCP Devices, including HDCP Repeaters, to be connected to an HDCP-protected Interface port.

Figure 1.1. illustrates an example connection topology for HDCP Devices.

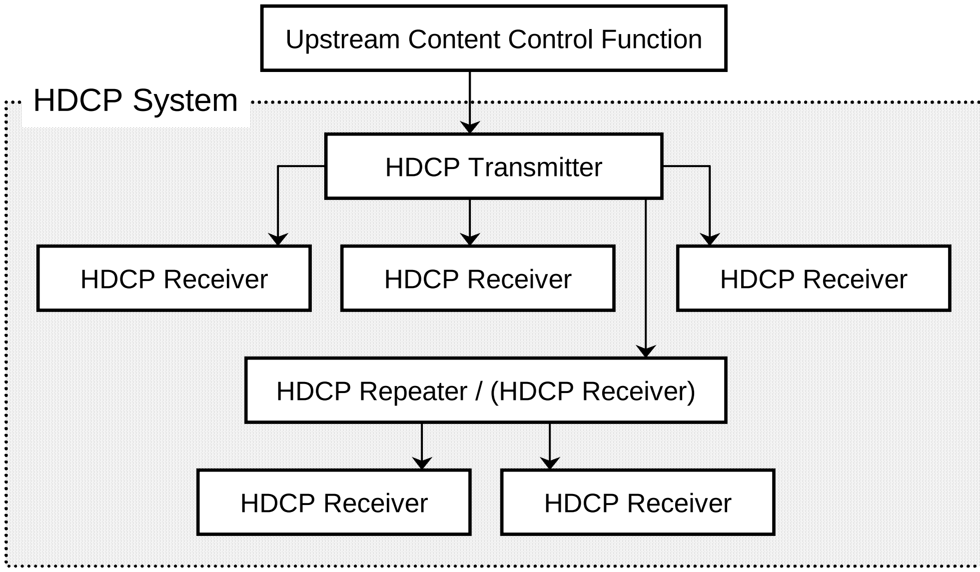

Figure 1.1. Sample Connection Topology of an HDCP System

There are three elements of the content protection system. Each element plays a specific role in the system. First, there is the authentication protocol, through which the HDCP Transmitter verifies that a given HDCP Receiver is licensed to receive HDCP Content. The authentication protocol is implemented between the HDCP Transmitter and its corresponding downstream HDCP Receiver. With the legitimacy of the HDCP Receiver determined, encrypted HDCP Content is transmitted between the two devices based on shared secrets established during the authentication protocol. This prevents eavesdropping devices from utilizing the content. Finally, in the event that legitimate devices are compromised to permit unauthorized use of HDCP Content, renewability allows an HDCP Transmitter to identify such compromised devices and prevent the transmission of HDCP Content.

This document contains chapters describing in detail the requirements of each of these elements. In addition, a chapter is devoted to describing the cipher structure that is used in the encryption of HDCP Content.

1.4 Terminology

Throughout this specification, names that appear in italic refer to values that are exchanged during the HDCP cryptographic protocol. C-style notation is used throughout the state diagrams and protocol diagrams, although the logic functions AND, OR, and XOR are written out where a textual description would be more clear.

This specification uses the big-endian notation to represent bit strings so that the most significant bit in the representation is stored in the left-most bit position. The concatenation operator '||' combines two values into one. For eight-bit values a and b, the result of $(a \| b)$ is a 16-bit value, with the value a in the most significant eight bits and b in the least significant eight bits.

1.5 References

[1]. Digital Content Protection (DCP) LLC, High-bandwidth Digital Content Protection System, Revision 1.4, July 8, 2009.

[2]. Digital Content Protection (DCP) LLC, HDCP Interface Independent Adaptation Specification, Revision 2.2, October 16, 2012.

[3]. WirelessHD Specification, Revision 1.1.

[4]. National Institute of Standards and Technology (NIST), *Advanced Encryption Standard (AES)*, FIPS Publication 197, November 26, 2001.

[5]. RSA Laboratories, *RSA Cryptography Standard*, PKCS #1 v2.1, June 14, 2002.

[6]. National Institute of Standards and Technology (NIST), *Secure Hash Standard (SHS)*, FIPS Publication 180-2, August 1, 2002.

[7]. Internet Engineering Task Force (IETF), *HMAC: Keyed-Hashing for Message Authentication*, Request for Comments (RFC) 2104, February 1997.

[8]. National Institute of Standards and Technology (NIST), Recommendation for Random Number Generation Using Deterministic Random Bit Generators, Special Publication 800-90, March 2007.

2 Authentication Protocol

2.1 Overview

The HDCP authentication protocol is an exchange between an HDCP Transmitter and an HDCP Receiver that affirms to the HDCP Transmitter that the HDCP Receiver is authorized to receive HDCP Content. It is comprised of the following stages

- Authentication and Key Exchange (AKE) – The HDCP Receiver's public key certificate is verified by the HDCP Transmitter. A Master Key k_m is exchanged.

- Locality Check – The HDCP Transmitter enforces locality on the content by requiring that the Round Trip Time (RTT) between a pair of messages is not more than 1 ms.

- Session Key Exchange (SKE) – The HDCP Transmitter exchanges Session Key k_s with the HDCP Receiver.

- Authentication with Repeaters – The step is performed by the HDCP Transmitter only with HDCP Repeaters. In this step, the repeater assembles downstream topology information and forwards it to the upstream HDCP Transmitter.

Successful completion of AKE and locality check stages affirms to the HDCP Transmitter that the HDCP Receiver is authorized to receive HDCP Content. At the end of the authentication protocol, a communication path is established between the HDCP Transmitter and HDCP Receiver that only Authorized Devices can access.

All HDCP Devices contain a 128-bit secret Global Constant denoted by lc_{128}. All HDCP Devices share the same Global Constant. lc_{128} is provided only to HDCP adopters.

The HDCP Transmitter contains the 3072-bit RSA public key of DCP LLC denoted by $kpub_{dcp}$.

The HDCP Receiver is issued 1024-bit RSA public and private keys. The public key is stored in a Public Key Certificate issued by DCP LLC, denoted by $cert_{rx}$. Table 2.1. gives the fields contained in the certificate. All values are stored in big-endian format.

Name	**Size (bits)**	**Bit position**	**Function**
Receiver ID	40	4175:4136	Unique receiver identifier. It has the same format as an HDCP 1.x KSV i.e. it contains 20 ones and 20 zeroes
Receiver Public Key	1048	4135:3088	Unique RSA public key of HDCP Receiver denoted by $kpub_{rx}$. The first 1024 bits is the big-endian representation of the modulus n and the trailing 24 bits is the big-endian representation of the public exponent e
Protocol Descriptor	4	3087:3084	Protocol descriptor field. Possible values are 0x0 or 0x1. 0x2 – 0xF – Reserved for future use
Reserved	12	3083:3072	Reserved for future definition. Must be 0x0000
DCP LLC Signature	3072	3071:0	A cryptographic signature calculated over all preceding fields of the certificate. RSASSA-PKCS1-v1_5 is the signature scheme used as defined by PKCS #1 V2.1: RSA Cryptography Standard. SHA-256 is the underlying hash function

Table 2.1. Public Key Certificate of HDCP Receiver

The secret RSA private key is denoted by $kpriv_{rx}$. The computation time of RSA private key operation can be reduced by using the Chinese Remainder Theorem (CRT) technique. Therefore, it is recommended that HDCP Receivers use the CRT technique for private key computations.

2.2 Authentication and Key Exchange

Authentication and Key Exchange (AKE) is the first step in the authentication protocol.

Figure 2.1 and Figure 2.2**Error! Reference source not found.** illustrates the AKE. The HDCP Transmitter (*Device A*) can initiate authentication at any time, even before a previous authentication exchange has completed. The HDCP Transmitter initiates a new HDCP Session by sending a new r_{tx} as part of the authentication initiation message, AKE_Init. Message formats are defined in Section 4.3.

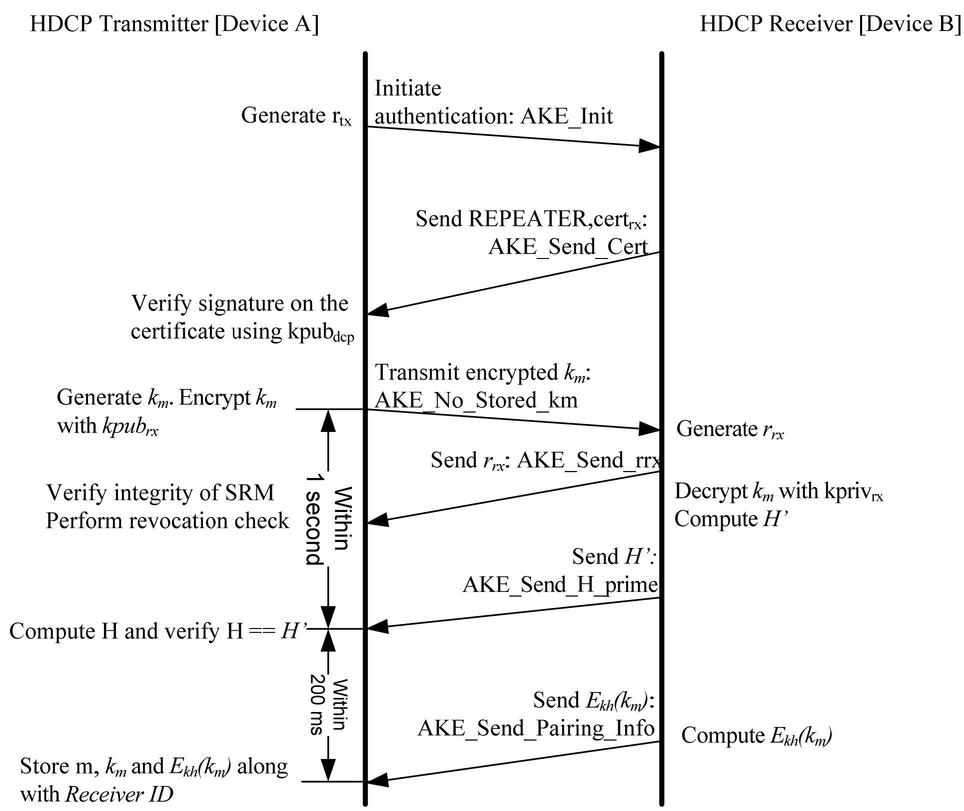

Figure 2.1. Authentication and Key Exchange (Without Stored k_m)

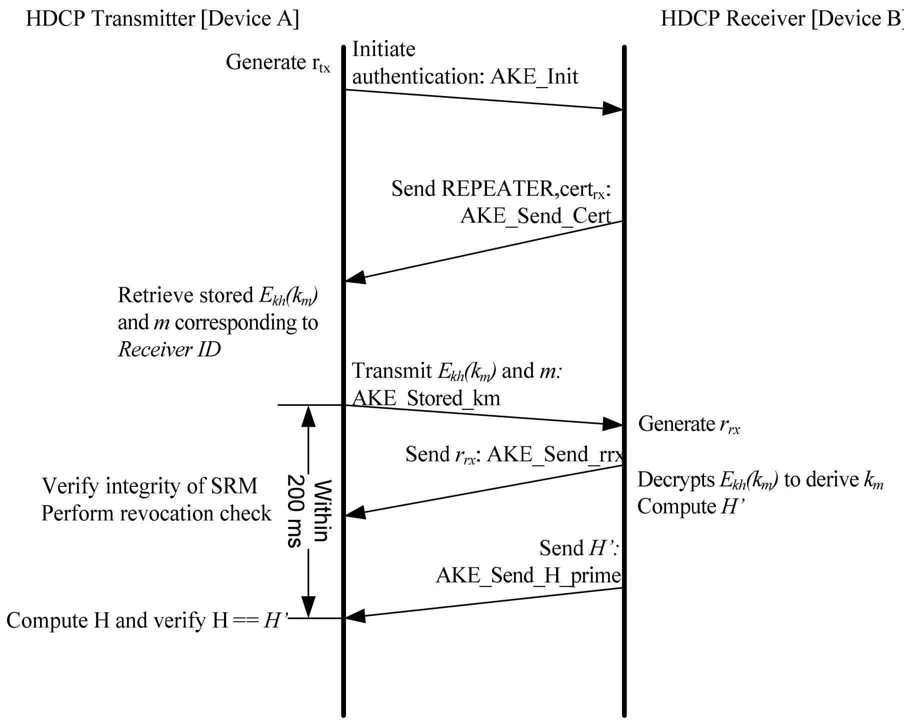

Figure 2.2. Authentication and Key Exchange (With Stored k_m)

The HDCP Transmitter

- Initiates authentication by sending the initiation message, AKE_Init, containing a 64-bit pseudo-random value (r_{tx}).

  Note: The HDCP Transmitter uses mechanisms in the WirelessHD specification that are outside the scope of the HDCP Specification to determine whether the HDCP Receiver is an HDCP 2.0-compliant Device. See section 4.2 for more details.

- Receives AKE_Send_Cert from the receiver containing REPEATER and $cert_{rx}$ values. REPEATER indicates whether the connected receiver is an HDCP Repeater.

- Extracts *Receiver ID* from $cert_{rx}$
  - o If the HDCP Transmitter does not have a 128-bit Master Key k_m stored corresponding to the *Receiver ID* (See Section 2.2.1)
    - Verifies the signature on the certificate using $kpub_{dcp}$. Failure of signature verification constitutes an authentication failure and the HDCP Transmitter aborts the authentication protocol.
    - Generates a pseudo-random 128-bit Master Key k_m. Encrypts k_m with $kpub_{rx}$ ($E_{kpub}(km)$) and sends AKE_No_Stored_km message to the receiver containing the 1024-bit $E_{kpub}(km)$. RSAES-OAEP encryption scheme must be used as defined by PKCS #1 V2.1: RSA Cryptography Standard. SHA-256 is the underlying hash function. The mask generation function used is MGF1 which uses SHA-256 as its underlying hash function.

- Verifies integrity of the System Renewability Message (SRM). It does this by checking the signature of the SRM using $kpub_{dcp}$. Failure of this integrity check constitutes an authentication failure and causes the HDCP Transmitter to abort authentication protocol.

  The top-level HDCP Transmitter checks to see if the *Receiver ID* of the connected device is found in the revocation list. If the *Receiver ID* of the connected HDCP Device is found in the revocation list, authentication fails and the authentication protocol is aborted. SRM integrity check and revocation check are performed only by the top-level HDCP Transmitter.

- Receives AKE_Send_rrx message from the receiver containing the 64-bit pseudo-random value (r_{rx}).

- Performs key derivation as explained in Section 2.7 to generate 256-bit k_d. $k_d = dkey_0 \| dkey_1$, where $dkey_0$ and $dkey_1$ are derived keys generated when ctr = 0 and ctr = 1 respectively. $dkey_0$ and $dkey_1$ are in big-endian order.

- Computes 256-bit H = HMAC-SHA256(r_{tx} XOR (REPEATER || Receiver_HDCP2_VERSION || Transmitter_HDCP2_VERSION), k_d) if the Protocol Descriptor field in $cert_{rx}$ is equal to 0x1. HMAC-SHA256 is computed over r_{tx} XOR REPEATER concatenated with Receiver_HDCP2_VERSION and Transmitter_HDCP2_VERSION and the key used for HMAC is k_d. Transmitter_HDCP2_VERSION is XORed with the least significant byte of r_{tx}. All values are in big-endian order.

  Computes 256-bit H = HMAC-SHA256(r_{tx} XOR REPEATER, k_d) if the Protocol Descriptor field in $cert_{rx}$ is equal to 0x0. HMAC-SHA256 is computed over r_{tx} XOR REPEATER and the key used for HMAC is k_d. REPEATER is XORed with the least significant byte of r_{tx}.

- Receives AKE_Send_H_prime message from the receiver containing the 256-bit *H'*. This message must be received within one second after sending $E_{kpub}(km)$ (AKE_No_Stored_km) to the receiver. Authentication fails and the authentication protocol is aborted if the message is not received within one second or there is a mismatch between H and *H'*.

o If the HDCP Transmitter has a 128-bit Master Key k_m stored corresponding to the *Receiver ID* (See Section 2.2.1)

  - Sends AKE_Stored_km message to the receiver with the 128-bit $E_{kh}(k_m)$ and the 128-bit *m* corresponding to the *Receiver ID* of the HDCP Receiver

  - Verifies integrity of the System Renewability Message (SRM). It does this by checking the signature of the SRM using $kpub_{dcp}$. Failure of this integrity check constitutes an authentication failure and causes the HDCP Transmitter to abort the authentication protocol.

    The top-level HDCP Transmitter checks to see if the *Receiver ID* of the connected device is found in the revocation list. If the *Receiver ID* of the connected HDCP Device is found in the revocation list, authentication fails and the authentication protocol is aborted.

- Receives AKE_Send_rrx message from the receiver containing the 64-bit pseudo-random value (r_{rx}) from the receiver.

- Performs key derivation as explained in Section 2.7 to generate 256-bit k_d. k_d = $dkey_0 \parallel dkey_1$, where $dkey_0$ and $dkey_1$ are derived keys generated when ctr = 0 and ctr = 1 respectively. $dkey_0$ and $dkey_1$ are in big-endian order.

- Computes 256-bit H = HMAC-SHA256(r_{tx} XOR (REPEATER || Receiver_HDCP2_VERSION || Transmitter_HDCP2_VERSION), k_d) if the Protocol Descriptor field in $cert_{rx}$ is equal to 0x1. HMAC-SHA256 is computed over r_{tx} XOR REPEATER concatenated with Receiver_HDCP2_VERSION and Transmitter_HDCP2_VERSION and the key used for HMAC is k_d. Transmitter_HDCP2_VERSION is XORed with the least significant byte of r_{tx}. All values are in big-endian order.

  Computes 256-bit H = HMAC-SHA256(r_{tx} XOR REPEATER, k_d) if the Protocol Descriptor field in $cert_{rx}$ is equal to 0x0. HMAC-SHA256 is computed over r_{tx} XOR REPEATER and the key used for HMAC is k_d. REPEATER is XORed with the least significant byte of r_{tx}.

- Receives AKE_Send_H_prime message from the receiver containing the 256-bit *H'*. This message must be received within 200 ms after sending the AKE_Stored_km message to the receiver. Authentication fails and the authentication protocol is aborted if the message is not received within 200 ms or there is a mismatch between H and *H'*.

The HDCP Receiver

- Sends AKE_Send_Cert message in response to AKE_Init

- Generates and sends 64-bit r_{rx} as part of the AKE_Send_rrx message immediately after receiving either AKE_No_Stored_km or AKE_Stored_km message from the transmitter.

  - If AKE_No_Stored_km is received, the HDCP Receiver

    - Decrypts k_m with $kpriv_{rx}$ using RSAES-OAEP decryption scheme.

    - Performs key derivation as explained in Section 2.7 to generate 256-bit k_d. k_d = $dkey_0 \parallel dkey_1$, where $dkey_0$ and $dkey_1$ are derived keys generated when ctr = 0 and ctr = 1 respectively. $dkey_0$ and $dkey_1$ are in big-endian order.

    - Computes *H'* = HMAC-SHA256(r_{tx} XOR (REPEATER || Receiver_HDCP2_VERSION || Transmitter_HDCP2_VERSION), k_d) if the Protocol Descriptor field in its public key certificate, $cert_{rx}$, is equal to 0x1 and the HDCP Transmitter is not HDCP2.0-compliant.

      Computes *H'* = HMAC-SHA256(r_{tx} XOR REPEATER, k_d) if the Protocol Descriptor field in its public key certificate, $cert_{rx}$ is equal to 0x0 or the HDCP Transmitter is HDCP2.0-compliant.

The HDCP Receiver sends AKE_Send_H_prime message immediately after computation of *H'* to ensure that the message is received by the transmitter within the specified one second timeout at the transmitter.

- If AKE_Stored_km is received, the HDCP Receiver
  - Computes 128-bit k_h = SHA-256($kpriv_{rx}$)[127:0]
  - Decrypts received $E_{kh}(k_m)$ using k_h as key to the AES module as described in Section 2.2.2 to derive k_m.
  - Performs key derivation as explained in Section 2.7 to generate 256-bit k_d. $k_d = dkey_0 \| dkey_1$, where $dkey_0$ and $dkey_1$ are derived keys generated when ctr = 0 and ctr = 1 respectively. $dkey_0$ and $dkey_1$ are in big-endian order.
  - Computes *H'* = HMAC-SHA256(r_{tx} XOR (REPEATER \|\| Receiver_HDCP2_VERSION \|\| Transmitter_HDCP2_VERSION), k_d) if the Protocol Descriptor field in its public key certificate, $cert_{rx}$, is equal to 0x1 and the HDCP Transmitter is not HDCP2.0-compliant.

    Computes *H'* = HMAC-SHA256(r_{tx} XOR REPEATER, k_d) if the Protocol Descriptor field in its public key certificate, $cert_{rx}$ is equal to 0x0 or the HDCP Transmitter is HDCP2.0-compliant.

    The HDCP Receiver sends AKE_Send_H_prime message immediately after computation of *H'* to ensure that the message is received by the transmitter within the specified 200 ms timeout at the transmitter.

On a decryption failure of k_m with $kpriv_{rx}$, the HDCP Receiver does not send *H'* and simply lets the timeout occur on the HDCP Transmitter.

2.2.1 Pairing

To speed up the AKE process, pairing must be implemented between the HDCP Transmitter and HDCP Receiver in parallel with AKE. When AKE_No_Stored_km message is received from the transmitter, it is an indication to the receiver that the transmitter does not have k_m stored corresponding to the receiver. In this case, after computing *H'*, the HDCP Receiver

- Computes 128-bit k_h = SHA-256($kpriv_{rx}$)[127:0].
- Encrypts k_m to produce 128-bit $E_{kh}(k_m)$ using method described in Section 2.2.2. Sends AKE_Send_Pairing_Info to the transmitter containing the 128-bit $E_{kh}(k_m)$.

On receiving AKE_Send_Pairing_Info message, the HDCP Transmitter may persistently store *m* (which is r_{tx} concatenated with r_{rx} ($r_{tx} \| r_{rx}$)), k_m and $E_{kh}(k_m)$ along with *Receiver ID*.

If AKE_Send_Pairing_Info is not received by the HDCP Transmitter within 200 ms of the reception of AKE_Send_H_prime, authentication fails and the authentication protocol is aborted.

Note: The HDCP Transmitter may store in its non-volatile storage *m*, k_m and $E_{kh}(k_m)$ along with corresponding *Receiver ID*s of all HDCP Receivers with which pairing was implemented by the HDCP Transmitter.

If the HDCP Receiver is HDCP 2.0-compliant, the HDCP Transmitter must not store pairing information (*m*, k_m and $E_{kh}(k_m)$ and *Receiver ID*) corresponding to the receiver.

If the HDCP Transmitter is not HDCP 2.0-compliant, the HDCP Receiver constructs 128-bit m by concatenating r_{tx} with r_{rx}: $m=r_{tx}||r_{rx}$. Both values are in big-endian order.

If the HDCP Transmitter is HDCP 2.0-compliant, 128-bit m is constructed by the HDCP Receiver by appending 64 0s to r_{tx}: $m=r_{tx}||$0x0000000000000000. r_{tx} is in big-endian order.

2.2.2 Pairing key encryption.

k_m encryption and decryption is performed using the following method:

1) Initialize 64-bit A to upper half k_m[127:64] for encryption and $E_{kh}(k_m)$[127:64] for decryption. Initialize 64-bit B to lower half k_m[63:0] for encryption and $E_{kh}(k_m)$[63:0] for decryption. Initialize 64-bit R[63:3] to m[127:67] XOR m[63:3]. Set R[2:0] = 0 for encryption and R[2:0] = 5 for decryption.

2) Do six times:

    a) AES-encrypt (A || R) with key k_h producing C.
    b) Update B = B XOR C[127:64] XOR C[63:0].
    c) Exchange A and B.
    d) If encrypting, increment R by 1 and if decrypting, decrement R by 1.

3) For encryption, $E_{kh}(k_m)$ is (B || A). For decryption, k_m is (B || A). (B becomes bits 127:64 of the result while A becomes bits 63:0 of the result). Note: the mapping of the halves is in the opposite order of step 1.

Figure 2.3 illustrates the encryption of k_m with k_h.

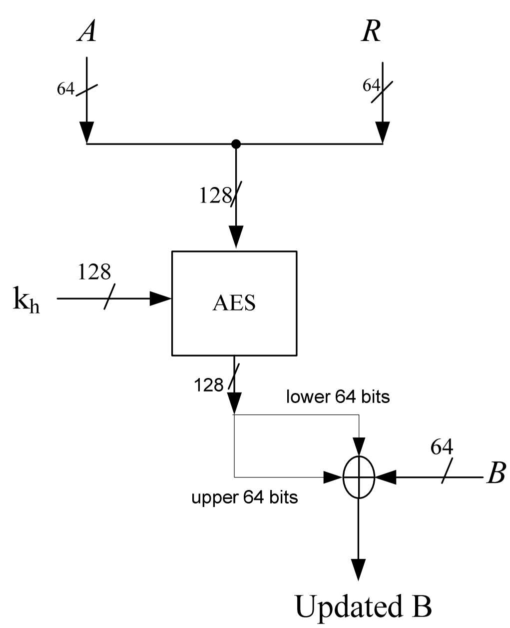

Figure 2.3. $E_{kh}(k_m)$ Computation

2.3 Locality Check

Locality check is performed after AKE and pairing. The HDCP Transmitter initiates locality check by sending a 64-bit pseudo-random nonce r_n to the downstream receiver. The HDCP Transmitter

- Initiates locality check by sending LC_Init message containing a 64-bit pseudo-random nonce r_n to the HDCP Receiver.

- Computes 256-bit L = HMAC-SHA256(r_n , k_d XOR r_{rx}) if the HDCP Transmitter is HDCP 2.0-compliant, where HMAC-SHA256 is computed over r_n and the key used for HMAC is k_d XOR r_{rx}, where r_{rx} is XORed with the least-significant 64-bits of k_d.

  Computes 256-bit L = HMAC-SHA256($r_n \| r_n$, k_d XOR r_{rx}) if the HDCP Transmitter is not HDCP 2.0-compliant, where HMAC-SHA256 is computed over $r_n \| r_n$ and the key used for HMAC is k_d XOR r_{rx}, where r_{rx} is XORed with the least-significant 64-bits of k_d.

- On receiving the RTT_Ready message from the receiver, the transmitter sends an RTT_Challenge message containing the least significant 128-bits of L.

- Sets its watchdog timer to 1 ms. Locality check fails if the watchdog timer expires before RTT_Response message is received.

- On receiving RTT_Response message the HDCP Transmitter compares the received value with the most significant 128-bits of L and locality check fails if there is a mismatch.

An HDCP Repeater initiates locality check on all its downstream HDCP-protected interface ports by sending unique r_n values to the connected HDCP Devices.

Figure 2.4 illustrates locality check between the HDCP Transmitter and HDCP Receiver.

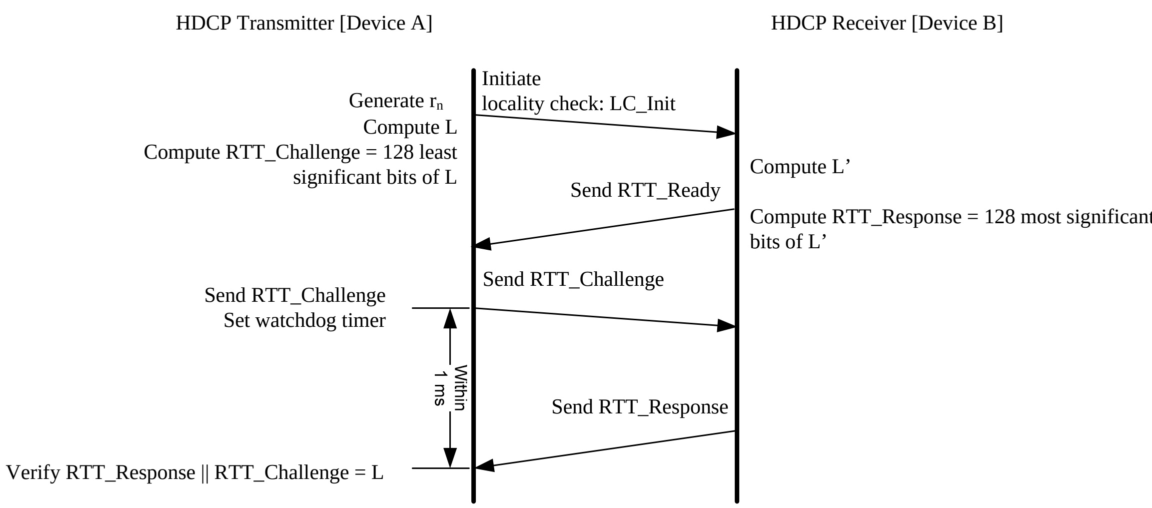

Figure 2.4. Locality Check between HDCP Transmitter and HDCP Receiver

The HDCP Receiver

- Computes 256-bit *L'* = HMAC-SHA256(r_n , k_d XOR r_{rx}) if Transmitter_HDCP2_VERSION = 0x0.

- Computes 256-bit *L'* = HMAC-SHA256($r_n \| r_n$, k_d XOR r_{rx}) if Transmitter_HDCP2_VERSION is not equal to 0x0.

- Sends RTT_Ready message to the transmitter when *L'* calculation is complete and the receiver is ready for the RTT Challenge.

- On receiving the RTT_Challenge message from the transmitter, if the value received in the RTT_Challenge message matches the least significant 128 bits of *L'*, the receiver sends an RTT_Response message containing the most significant 128-bits of *L'*.

In the case of a locality check failure due to expiration of the watchdog timer or due to mismatch of the most significant 128-bits of L and *L'* at the HDCP Transmitter, locality check may be reattempted by the HDCP Transmitter for a maximum of 1023 additional attempts (for a maximum allowed 1024 total trials) with the transmission of an LC_Init message containing a new r_n. Failure of locality check on the first attempt and subsequent zero or more reattempts results in an authentication failure and the authentication protocol is aborted.

2.4 Session Key Exchange

Successful completion of AKE and locality check stages affirms to the HDCP Transmitter that the HDCP Receiver is authorized to receive HDCP Content. Session Key Exchange (SKE) is initiated by the HDCP Transmitter after a successful locality check. The HDCP Transmitter sends encrypted Session Key to the HDCP Receiver at least 200 ms before enabling HDCP Encryption and beginning the transmission of HDCP Content. HDCP Encryption may be enabled 200 ms after the transmission of the encrypted Session Key to the HDCP Receiver and at no time prior. Content encrypted with the Session Key k_s starts to flow between the HDCP Transmitter and HDCP Receiver. HDCP Encryption must be enabled only after successful completion of AKE, locality check and SKE stages.

HDCP allows for multiple simultaneous streams within a given HDCP session. Each stream is identified by StreamIndex present in each WirelessHD MAC packet header. The HDCP session key is not specific to a particular stream. An HDCP Session begins when CONNECT_RESPONSE (Section 10.4.4.2 of [3]) is transmitted for the first active stream and ends when DISCONNECT_NOTIFY (Section 10.4.4.4 of [3]) is transmitted. Within a session, there can be multiple streams. The same session key can be used by the HDCP Transmitter to encrypt different streams with different StreamIndex values. The HDCP session must be terminated when Secure Packet Counter (40-bit parameter), present in WirelessHD MAC packet header rolls back to zero. Note that under normal conditions this is a very long time; at 200ms per 7-sub-packet packet, this is over one year.

During SKE, the HDCP Transmitter

- Generates a pseudo-random 128-bit Session Key k_s and 64-bit pseudo-random number r_{iv}.
- Performs key derivation as explained in Section 2.7 to generate 128-bit $dkey_2$ where $dkey_2$ is the derived key when ctr =2.
- Computes 128-bit $E_{dkey}(k_s) = k_s$ XOR ($dkey_2$ XOR r_{rx}), where r_{rx} is XORed with the least-significant 64-bits of $dkey_2$.
- Sends SKE_Send_Eks message containing $E_{dkey}(k_s)$, r_{iv} to the HDCP Receiver.

On receiving SKE_Send_Eks message, the HDCP Receiver

- Performs key derivation as explained in Section 2.7 to generate 128-bit $dkey_2$ where $dkey_2$ is the derived key when ctr =2.
- Computes $k_s = E_{dkey}(k_s)$ XOR ($dkey_2$ XOR r_{rx})

2.5 Authentication with Repeaters

The HDCP Transmitter executes authentication with repeaters after Session Key exchange and only when REPEATER is 'true', indicating that the connected HDCP Receiver is an HDCP Repeater. Authentication with repeaters stage is used for the upstream propagation of topology information and the downstream propagation of Content Stream management information as explained in Section 2.5.1 and Section 2.5.2 respectively. Authentication with repeaters may be implemented by the HDCP Transmitter in parallel with the flow of encrypted content and Link Synchronization. The Link Synchronization process is explained in Section 2.6.

2.5.1 Upstream Propagation of Topology Information

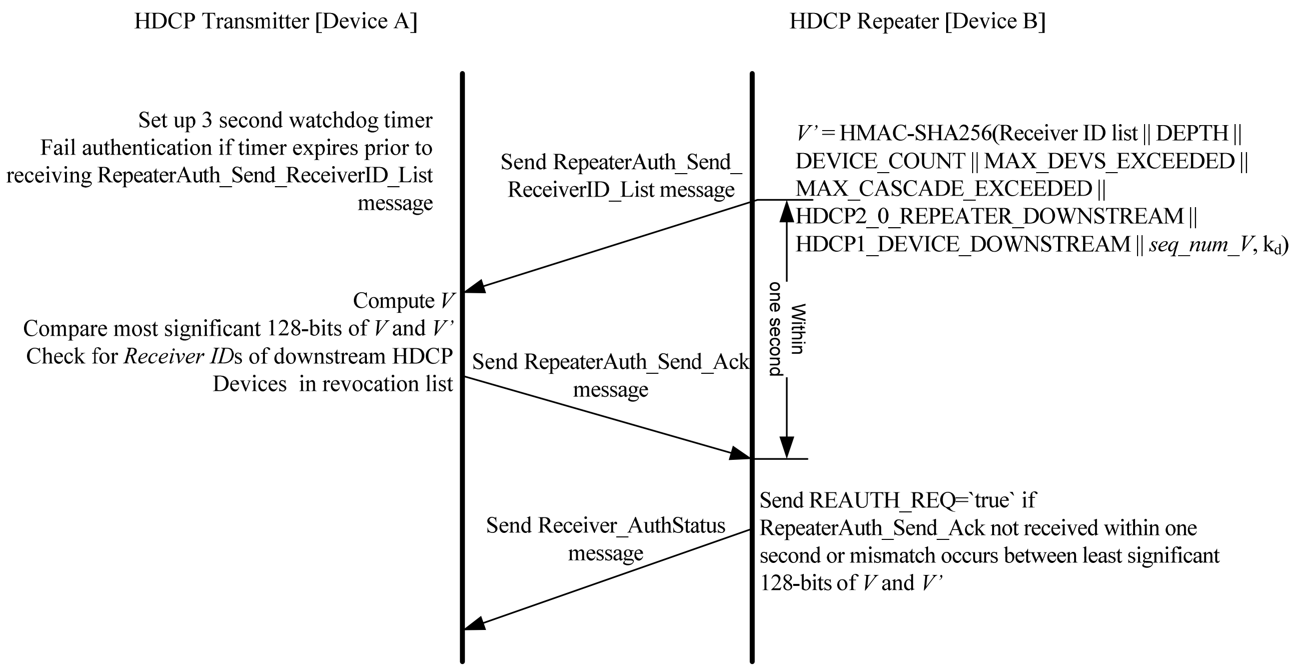

Figure 2.5. Upstream Propagation of Topology Information

Figure 2.5 illustrates the upstream propagation of topology information. This stage assembles a list of all downstream *Receiver IDs* connected to the HDCP Repeater through a permitted connection tree, enabling revocation support upstream. This stage is implemented after successful completion of Session Key Exchange. This stage is used to assemble the latest topology information at the beginning of the HDCP Session immediately following an SKE or on subsequent changes to the topology due to connect or disconnect of an HDCP Receiver or HDCP Repeater.

HDCP Repeaters assemble the list of all connected downstream HDCP Receivers as the downstream HDCP-protected Interface Ports of the HDCP Repeater successfully complete the authentication protocol with connected HDCP Receivers. The list is represented by a contiguous set of bytes, with each *Receiver ID* occupying five bytes stored in big-endian order. The total length of the Receiver ID list is five bytes times the total number of connected and active downstream HDCP Devices, including downstream HDCP Repeaters, with which the HDCP Repeater has successfully completed the authentication protocol. This total number is represented in the RepeaterAuth_Send_ReceiverID list message by the DEVICE_COUNT value. An HDCP-protected Interface Port with no active device connected adds nothing to the list. Also, the *Receiver ID* of the HDCP Repeater itself at any level is not included in its own Receiver ID list. An HDCP-protected Interface Port connected to an HDCP Receiver that is not an HDCP Repeater adds the *Receiver ID* of the connected HDCP Receiver to the list. HDCP-protected Interface Ports that have an HDCP Repeater connected add the Receiver ID list received from the connected downstream HDCP Repeater, plus the *Receiver ID* of the connected downstream HDCP Repeater itself.

In order to add the Receiver ID list of the connected downstream HDCP Repeater, it is necessary for the HDCP Repeater to verify the integrity of the list. If the connected HDCP Repeater is not an HDCP 2.0-compliant Device, the HDCP Repeater verifies the integrity of the list by computing *V* and checking the most significant 128-bits of *V* against the most significant 128 bits of *V*' received as part of the RepeaterAuth_Send_ReceiverID_List message from the connected downstream HDCP Repeater. If the connected HDCP Repeater is an HDCP 2.0-compliant Device, the HDCP Repeater verifies the integrity of the list by computing *V* and comparing *V* against *V*'. If the values do not match, the downstream Receiver ID list integrity check fails, and the HDCP Repeater must not add the Receiver ID list received from the downstream HDCP Repeater to its Receiver ID list.

When the HDCP Repeater has assembled the complete list of *Receiver ID*s of connected and active HDCP Devices with which the HDCP Repeater has successfully completed the authentication protocol, it computes the 256-bit verification value *V*'.

An HDCP Repeater connected to an HDCP 2.0-compliant upstream HDCP Transmitter and an HDCP Transmitter connected to an HDCP 2.0-compliant HDCP Repeater computes respective *V*' and *V* values as given below. HMAC-SHA256 is computed over the concatenation of Receiver ID list, DEPTH, DEVICE_COUNT, MAX_DEVS_EXCEEDED and MAX_CASCADE_EXCEEDED received as part of the RepeaterAuth_Send_ReceiverID_List message. The key used for HMAC is k_d.

V' (or *V*) = HMAC-SHA256(Receiver ID list || DEPTH || DEVICE_COUNT || MAX_DEVS_EXCEEDED || MAX_CASCADE_EXCEEDED, k_d)

An HDCP Repeater connected to an upstream HDCP Transmitter that is not HDCP 2.0-compliant and an HDCP Transmitter connected to an HDCP Repeater that is not HDCP 2.0-compliant computes respective *V*' and *V* values as given below. HMAC-SHA256 is computed over the concatenation of Receiver ID list, DEPTH, DEVICE_COUNT, MAX_DEVS_EXCEEDED, MAX_CASCADE_EXCEEDED, HDCP2_0_REPEATER_DOWNSTREAM, HDCP1_DEVICE_DOWNSTREAM and *seq_num_V* received as part of the RepeaterAuth_Send_ReceiverID_List message. The key used for HMAC is k_d.

V' (or *V*) = HMAC-SHA256(Receiver ID list || DEPTH || DEVICE_COUNT || MAX_DEVS_EXCEEDED || MAX_CASCADE_EXCEEDED || HDCP2_0_REPEATER_DOWNSTREAM || HDCP1_DEVICE_DOWNSTREAM || *seq_num_V*, k_d)

Receiver ID list is formed by appending downstream *Receiver ID*s in big-endian order. When the Receiver ID list, *V*', DEPTH, DEVICE_COUNT , and if applicable, HDCP2_0_REPEATER_DOWNSTREAM and HDCP1_DEVICE_DOWNSTREAM are available, the HDCP Repeater sends RepeaterAuth_Send_ReceiverID_List message to the upstream HDCP Transmitter. The HDCP Repeater sends *V*' if the upstream transmitter is HDCP 2.0-compliant and the most significant 128-bits of *V*' if the upstream transmitter is not HDCP 2.0-compliant.

The HDCP Repeater initializes *seq_num_V* to 0 at the beginning of the HDCP Session i.e. after r_{tx} is received. It is incremented by one after the transmission of every RepeaterAuth_Send_ReceiverID_List message. *seq_num_V* must never be reused during an HDCP Session for the computation of *V* (or *V*'). If *seq_num_V* rolls over, the HDCP Transmitter must detect the roll-over in the RepeaterAuth_Send_ReceiverID_List received from the HDCP Repeater and the transmitter must disable HDCP Encryption if encryption is enabled, restart authentication by the transmission of a new r_{tx} as part of the AKE_Init message.

When an HDCP Repeater receives HDCP2_0_REPEATER_DOWNSTREAM = 'true' or HDCP1_DEVICE_DOWNSTREAM = 'true' from a downstream HDCP Repeater, it must propagate this information to the upstream HDCP Transmitter by setting the corresponding values to 'true' in the RepeaterAuth_Send_ReceiverID_List message.

If HDCP2_0_REPEATER_DOWNSTREAM = 'true' or HDCP1_DEVICE_DOWNSTREAM = 'true', the Upstream Content Control Function may instruct the most upstream HDCP Transmitter to abort the transmission of certain HDCP encrypted Type 1 Content Streams. The most upstream HDCP Transmitter must be prepared to process the request and immediately cease the transmission of specific Content Streams as instructed by the Upstream Content Control Function.

Whenever the RepeaterAuth_Send_ReceiverID_List message is received, the HDCP Transmitter verifies the integrity of the Receiver ID list by computing *V* and comparing either *V* and *V'* (if the connected HDCP Repeater is HDCP 2.0-compliant) or the most significant 128-bits of *V* and *V'* (if the connected HDCP Repeater is not HDCP 2.0-compliant). If the values do not match, authentication fails, the authentication protocol is aborted and HDCP Encryption is disabled.

On successful verification of Receiver ID list and topology information, i.e. if the values match, none of the reported *Receiver IDs* are in the current revocation list (in the case of the most upstream HDCP Transmitter), the HDCP Transmitter does not detect a roll-over of *seq_num_V*, the downstream topology does not exceed specified maximums (explained below) and the HDCP Repeater is not HDCP 2.0-compliant, the HDCP Transmitter (including downstream port of HDCP Repeater) sends the least significant 128-bits of *V* to the HDCP Repeater as part of the RepeaterAuth_Send_Ack message. Every RepeaterAuth_Send_ReceiverID_List message from the repeater to the transmitter must be followed by a RepeaterAuth_Send_Ack message from the transmitter to repeater on successful verification of Receiver ID list and topology information by the transmitter.

The RepeaterAuth_Send_Ack message must be received by the HDCP Repeater within one second from the transmission of the RepeaterAuth_Send_ReceiverID_List message to the HDCP Transmitter if the HDCP Transmitter is not HDCP 2.0-compliant and the downstream topology does not exceed specified maximums. A match between the least significant 128-bits of *V* and *V'* indicates successful upstream transmission of topology information. If a mismatch occurs or the RepeaterAuth_Send_Ack message is not received by the repeater within one second, the HDCP Repeater must send the Receiver_AuthStatus message with the REAUTH_REQ set to 'true' and must transition in to an unauthenticated state (See Section 2.10.3).

If the upstream HDCP Transmitter receives a Receiver_AuthStatus message with REAUTH_REQ set to 'true', it may initiate re-authentication with the HDCP Repeater by the transmission of a new r_{tx}.

After transmitting the SKE_Send_Eks message, the HDCP Transmitter, having determined that REPEATER received earlier in the protocol is 'true', sets a three second watchdog timer. If the RepeaterAuth_Send_ReceiverID_List message is not received by the HDCP Transmitter within a maximum-permitted time of three seconds after transmitting SKE_Send_Eks message, authentication of the HDCP Repeater fails. With this failure, the HDCP Transmitter disables HDCP Encryption and aborts the authentication protocol with the HDCP Repeater.

When an HDCP Receiver (including HDCP Repeater) is connected to the HDCP Repeater or when a connected, active HDCP Receiver with which the HDCP Repeater has successfully completed the authentication protocol is disconnected from the HDCP Repeater and the upstream HDCP Transmitter is not HDCP 2.0-compliant, the HDCP Repeater must send the RepeaterAuth_Send_ReceiverID_List message to the upstream HDCP Transmitter which must include the *Receiver IDs* of all connected and active downstream HDCP Receivers with which the HDCP Repeater has successfully completed the authentication protocol. This enables upstream propagation of the most recent topology information after changes to the topology without interrupting the transmission of HDCP Content.

Refer to Table 2.2 for the HDCP Repeater upstream and downstream propagation time.

The HDCP Repeater propagates topology information upward through the connection tree to the HDCP Transmitter. An HDCP Repeater reports the topology status variables DEVICE_COUNT and DEPTH. The DEVICE_COUNT for an HDCP Repeater is equal to the total number of connected downstream HDCP Receivers and HDCP Repeaters. The value is calculated as the sum of the number of directly connected downstream HDCP Receivers and HDCP Repeaters plus the sum of the DEVICE_COUNT received from all connected HDCP Repeaters. The DEPTH status for an HDCP Repeater is equal to the maximum number of connection levels below any of the downstream HDCP-protected Interface Ports. The value is calculated as the maximum DEPTH reported from downstream HDCP Repeaters plus one (accounting for the connected downstream HDCP Repeater).

In Figure 2.6, R1 has three downstream HDCP Receivers connected to it. It reports a DEPTH of one and a DEVICE_COUNT of three.

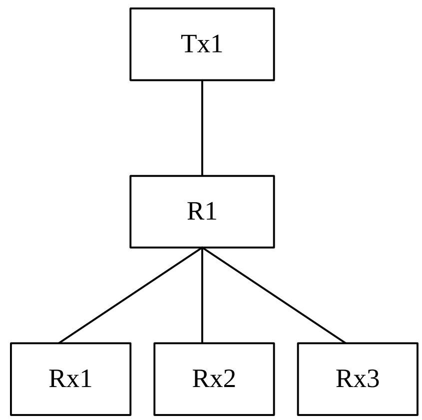

Figure 2.6. DEPTH and DEVICE_COUNT for HDCP Repeater

In Figure 2.7, R1 reports a DEPTH of two and a DEVICE_COUNT of four.

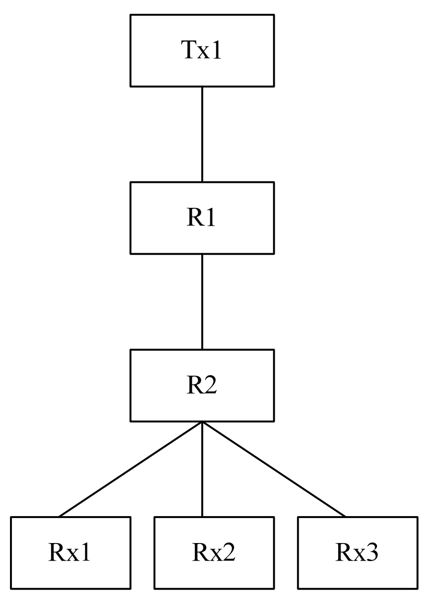

Figure 2.7. DEPTH and DEVICE_COUNT for HDCP Repeater

HDCP Repeaters must be capable of supporting DEVICE_COUNT values of up to 31 and DEPTH values of up to 4. If the computed DEVICE_COUNT for an HDCP Repeater exceeds 31, the error is referred to as MAX_DEVS_EXCEEDED error. The repeater sets MAX_DEVS_EXCEEDED = 'true' in the RepeaterAuth_Send_ReceiverID_List message. If the computed DEPTH for an HDCP Repeater exceeds four, the error is referred to as MAX_CASCADE_EXCEEDED error. The repeater sets MAX_CASCADE_EXCEEDED = 'true' in the RepeaterAuth_Send_ReceiverID_List message. When an HDCP Repeater receives a MAX_DEVS_EXCEEDED or a MAX_CASCADE_EXCEEDED error from a downstream HDCP Repeater, it must propagate the error to the upstream HDCP Transmitter and must not transmit *V'* (or the most significant 128-bits of *V'*), DEPTH, DEVICE_COUNT, Receiver ID list and if applicable, HDCP2_0_REPEATER_DOWNSTREAM and HDCP1_DEVICE_DOWNSTREAM.

Authentication fails if the topology maximums are exceeded. HDCP Encryption is disabled and the authentication protocol is aborted. The top-level HDCP Transmitter, having already performed SRM integrity check during AKE, proceeds to see if the *Receiver ID* of any downstream device from the Receiver ID list is found in the current revocation list, and, if present, authentication fails, HDCP Encryption is disabled and authentication protocol is aborted.

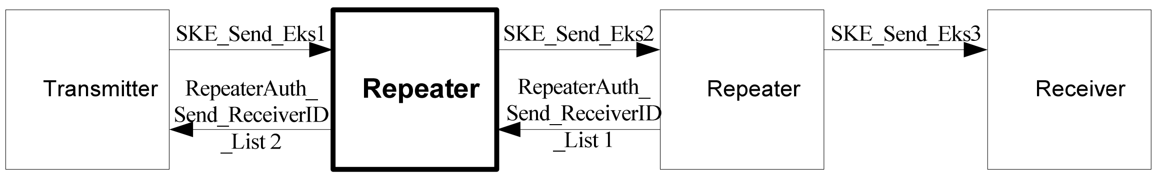

Figure 2.8. HDCP Repeater Protocol Timing Requirements

From	To	Max Delay	Conditions and Comments
SKE_Send_Eks1 Session key received from Upstream HDCP Transmitter	SKE_Send_Eks 2 k_s generated by HDCP Repeater transmitted downstream	100 ms	Downstream propagation time.
SKE_Send_Eks3 k_s transmitted to all downstream HDCP-protected Interface Ports	RepeaterAuth_Send_ReceiverID_List1 *Receiver IDs* and topology information transmitted upstream	200 ms	Upstream propagation time when no downstream HDCP Repeaters are attached (no downstream Receiver ID lists to process)
RepeaterAuth_Send_ReceiverID_List1	RepeaterAuth_Send_ReceiverID_List2	200 ms	
Downstream *Receiver IDs* and topology information received	*Receiver IDs* and topology information transmitted upstream		
Upstream propagation time when one or more HDCP Repeaters are attached. From latedoSKE_Send_Eks1Upstream HDCP Transmitter transmits k_s	RepeaterAuth_Send_ReceiverID_List2 Upstream HDCP Transmitter receives RepeaterAuth_Send_ReceiverID_List message	1.2 seconds	For the Maximum of four repeater levels, 4 * (100 ms + 200 ms)

Table 2.2 HDCP Repeater Protocol Timing Requirements

Table 2.2 specifies HDCP Repeater timing requirements that bound the worst-case propagation time for the Receiver ID list. A maximum delay of three seconds has been provided to receive the RepeaterAuth_Send_ReceiverID_List message by the upstream transmitter to account for authentication delays due to the presence of downstream receivers that have not been paired with the upstream HDCP Repeater. Note that because each HDCP Repeater does not know the number of downstream HDCP Repeaters, it must use the same three-second timeout used by the upstream HDCP Transmitter for receiving the RepeaterAuth_Send_ReceiverID_List message.

2.5.2 Downstream Propagation of Content Stream Management Information

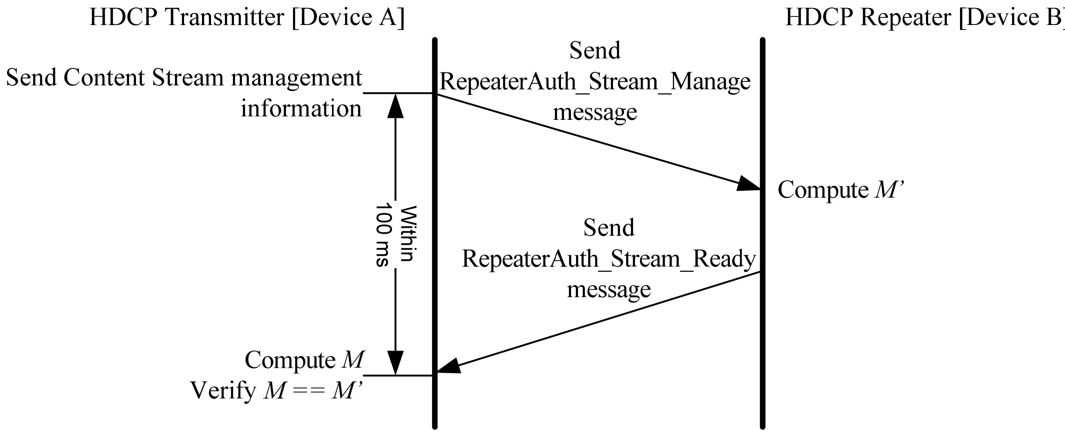

Figure 2.9. Downstream Propagation of Content Stream Management Information

The HDCP Transmitter may transmit multiple Content Streams to an HDCP Receiver during an HDCP Session. The HDCP Transmitter may use the same Session Key, k_s, negotiated during the HDCP Session for HDCP Encryption of the Content Streams.

The HDCP Transmitter propagates Content Stream management information, which includes Type values assigned to Content Streams, using the RepeaterAuth_Stream_Manage message to the attached HDCP Repeater only if the attached HDCP Repeater is not an HDCP 2.0-compliant Device. The HDCP Transmitter executes this step after successful completion of Session Key Exchange and before beginning the transmission of a Content Stream after HDCP Encryption to the HDCP Repeater. The RepeaterAuth_Stream_Manage message from an HDCP Transmitter to the attached HDCP Repeater identifies restrictions, as specified by the Upstream Content Control Function, on the transmission of Content Streams to specific devices.

Type values are assigned to all Content Streams by the most upstream HDCP Transmitter based on instructions received from the Upstream Content Control Function. The exact mechanism used by the Upstream Content Control Function to instruct the HDCP Transmitter is outside the scope of this specification. Type 0 Content Streams (see Section 4.3.15) may be transmitted by the HDCP Repeater to all HDCP Devices. Type 1 Content Streams (see Section 4.3.15) must not be transmitted by the HDCP Repeater through its HDCP-protected Interface Ports connected to HDCP 1.x-compliant Devices and HDCP 2.0-compliant Repeaters.

The most upstream HDCP Transmitter must not transmit Type 1 Content Streams to HDCP 1.x-compliant Devices and HDCP 2.0-compliant Repeaters as instructed by the corresponding Upstream Content Control Function.

The HDCP Transmitter must send the RepeaterAuth_Stream_Manage message specifying Type values assigned to Content Streams, to the attached HDCP Repeater at least 100ms before the transmission of the corresponding Content Streams after HDCP Encryption. The HDCP Transmitter must only send the RepeaterAuth_Stream_Manage message corresponding to encrypted Content Streams it will transmit to the HDCP Repeater. The HDCP Transmitter initializes *seq_num_M* to 0 at the beginning of the HDCP Session i.e. after r_{tx} is sent. It is incremented by one after the transmission of every RepeaterAuth_Stream_Manage message.

On receiving the RepeaterAuth_Stream_Manage message, the HDCP Repeater computes *M'* as given below. HMAC-SHA256 is computed over the concatenation of STREAMID_TYPE (see Section 4.3.15) and *seq_num_M* values received as part of the RepeaterAuth_Stream_Manage message. All values are in big-endian order. The key used for HMAC is SHA256(k_d). *seq_num_M* must never be reused during an HDCP Session for the computation of *M'* (or *M*). If *seq_num_M* rolls over, the HDCP Transmitter must disable HDCP Encryption if encryption is enabled, restart authentication by the transmission of a new r_{tx} as part of the AKE_Init message.

M' (or *M*) = HMAC-SHA256(STREAMID_TYPE || *seq_num_M*, SHA256(k_d)).

M' must be sent by the HDCP Repeater to the HDCP Transmitter as part of the RepeaterAuth_Stream_Ready message.

The HDCP Transmitter must receive the RepeaterAuth_Stream_Ready message within 100 ms after the transmission of RepeaterAuth_Stream_Manage message. Every RepeaterAuth_Stream_Manage message from the transmitter to the repeater must be followed by a RepeaterAuth_Stream_Ready message from the repeater to the transmitter.

When the RepeaterAuth_Stream_Ready message is received, the HDCP Transmitter verifies the integrity of the message by computing *M* and comparing this value to *M'*. If *M* is equal to *M'*, the HDCP Transmitter may transmit the Content Streams identified in the corresponding RepeaterAuth_Stream_Manage message. If the RepeaterAuth_Stream_Ready message is not received within 100 ms or if *M* is not equal to *M'*, the HDCP Transmitter must not transmit the Content Streams identified in the corresponding RepeaterAuth_Stream_Manage message.

An HDCP Repeater connected to an HDCP 2.0-compliant Transmitter or an HDCP 1.x-compliant Transmitter will not receive the RepeaterAuth_Stream_Manage message from the transmitter. In this case, the HDCP Repeater must assign a Type value of 0x00 to all Content Streams received from the HDCP Transmitter.

The HDCP Repeater must in turn propagate the received Content Stream management information using the RepeaterAuth_Stream_Manage message further downstream.

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. The HDCP Transmitter uses the STREAM_START_NOTIFY message to indicate the StreamIndex for the stream that will use HDCP 2, as negotiated in the CONNECT_REQUEST and CONNECT_RESPONSE messages. The HDCP Receiver uses the StreamIndex in the MAC header and the CP Header Present bit to identify data that has HDCP Encryption enabled and that the sub-packet payload is encrypted. (Refer to [3] for details about StreamIndex and CP Header Present bit). The CP Header Present bit must be set to one in the MAC Control header if HDCP Encryption is enabled for the audio and video sub-packet payloads and must be set to zero if HDCP Encryption is disabled for the audio and video sub-packet payloads. Once encrypted content starts to flow, a periodic Link Synchronization is performed to maintain cipher synchronization between the HDCP Transmitter and the HDCP Receiver.

The HDCP Transmitter must ensure that it supports both protected and non-protected video streams to HDCP Receiver. For example supporting non-protected menu video. This can be done either at MAC sub-packet level for a given StreamIndex (CP can be disabled for duration of sub-packet) or a separate StreamIndex can be configured and sent in a separate MAC frame.

Link Synchronization is achieved every time a MAC packet header is transmitted, by the inclusion of the Secure Packet Counter in the MAC header and the CP Header Present bit is set to one in the MAC header. The HDCP Receiver updates its *inputCtr* (as explained in Section 0) corresponding to the Secure Packet Counter value received from the transmitter and local count of 128-bit video blocks for AES cipher.

2.7 Key Derivation

Key derivation is illustrated in Figure 2.10.

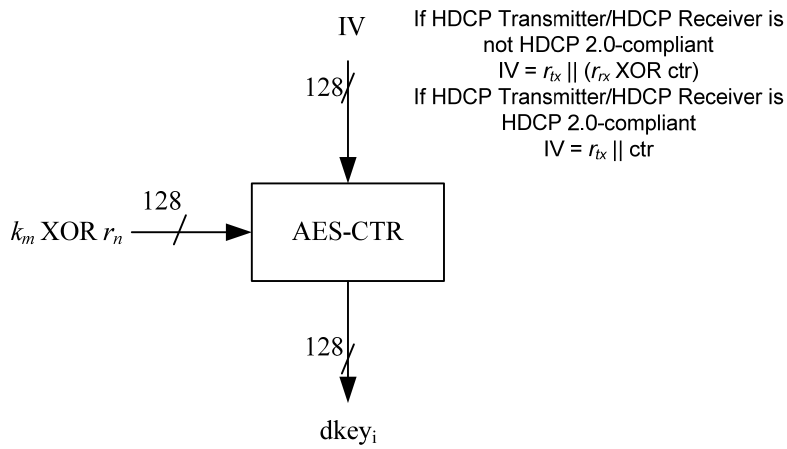

Figure 2.10. Key Derivation

The HDCP Transmitter sets IV = r_{tx} || (r_{rx} XOR ctr) if HDCP Receiver is not HDCP 2.0-compliant. If HDCP Receiver is HDCP 2.0-compliant, the HDCP Transmitter sets IV = r_{tx} || ctr. All values are in big-endian order.

The HDCP Receiver sets IV = r_{tx} || (r_{rx} XOR ctr) if HDCP Transmitter is not HDCP 2.0-compliant. If HDCP Transmitter is HDCP 2.0-compliant, the HDCP Receiver sets IV = r_{tx} || ctr. All values are in big-endian order.

ctr is a 64-bit counter and is initialized to 0 at the beginning of the HDCP Session i.e. after r_{tx} is sent or received. It is incremented by one after every derived key computation. $dkey_i$ is the 128-bit derived key when ctr = i. ctr must never be reused during an HDCP Session.

r_n is initialized to 0 during AKE i.e. during the generation of $dkey_0$ and $dkey_1$. It is set to a pseudo-random value during locality check as explained in Section 2.3. The pseudo-random r_n is XORed with the least-significant 64-bits of k_m during generation of $dkey_2$.

2.8 HDCP Transmitter State Diagram

As explained in Section 1.3, the HDCP Transmitter may support simultaneous connections to HDCP Receivers through one or more of its HDCP-protected interface ports. The HDCP Transmitter state diagram is implemented independently on each HDCP-protected interface port.

The HDCP Transmitter Link State Diagram and HDCP Transmitter Authentication Protocol State Diagram (Figure 2.11 and Figure 2.12) illustrate the operation states of the authentication protocol for an HDCP Transmitter that is not an HDCP Repeater. For HDCP Repeaters, the downstream (HDCP Transmitter) side is covered in Section 2.10.2.

Transmitter's decision to begin authentication is dependent on events such as detection of an HDCP Receiver, availability of premium content or other implementation dependent details in the transmitter. In the event of authentication failure, an HDCP Receiver must be prepared to process subsequent authentication attempts. The HDCP Transmitter may cease to attempt authentication for transmitter-specific reasons, which include receiving a Receiver Disconnected Indication or after a certain number of authentication re-attempts by the transmitter.

The transmitter must not initiate authentication unless it determines that the receiver is HDCP-capable as described in the setup and discovery part of section 10.6.4 of WirelessHD specification

([3]). Section 10.6.4 of WirelessHD specification also describes how the HDCP transmitter determines the connect/disconnect status of the HDCP Receiver.

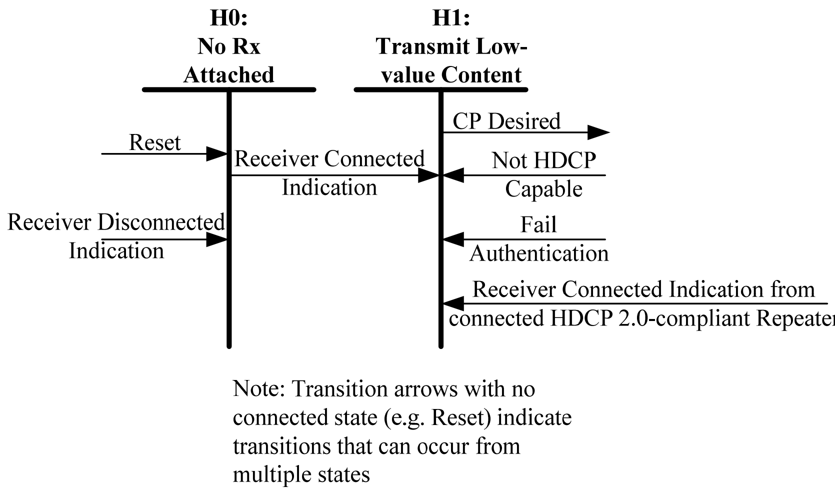

Figure 2.11. HDCP Transmitter Link State Diagram

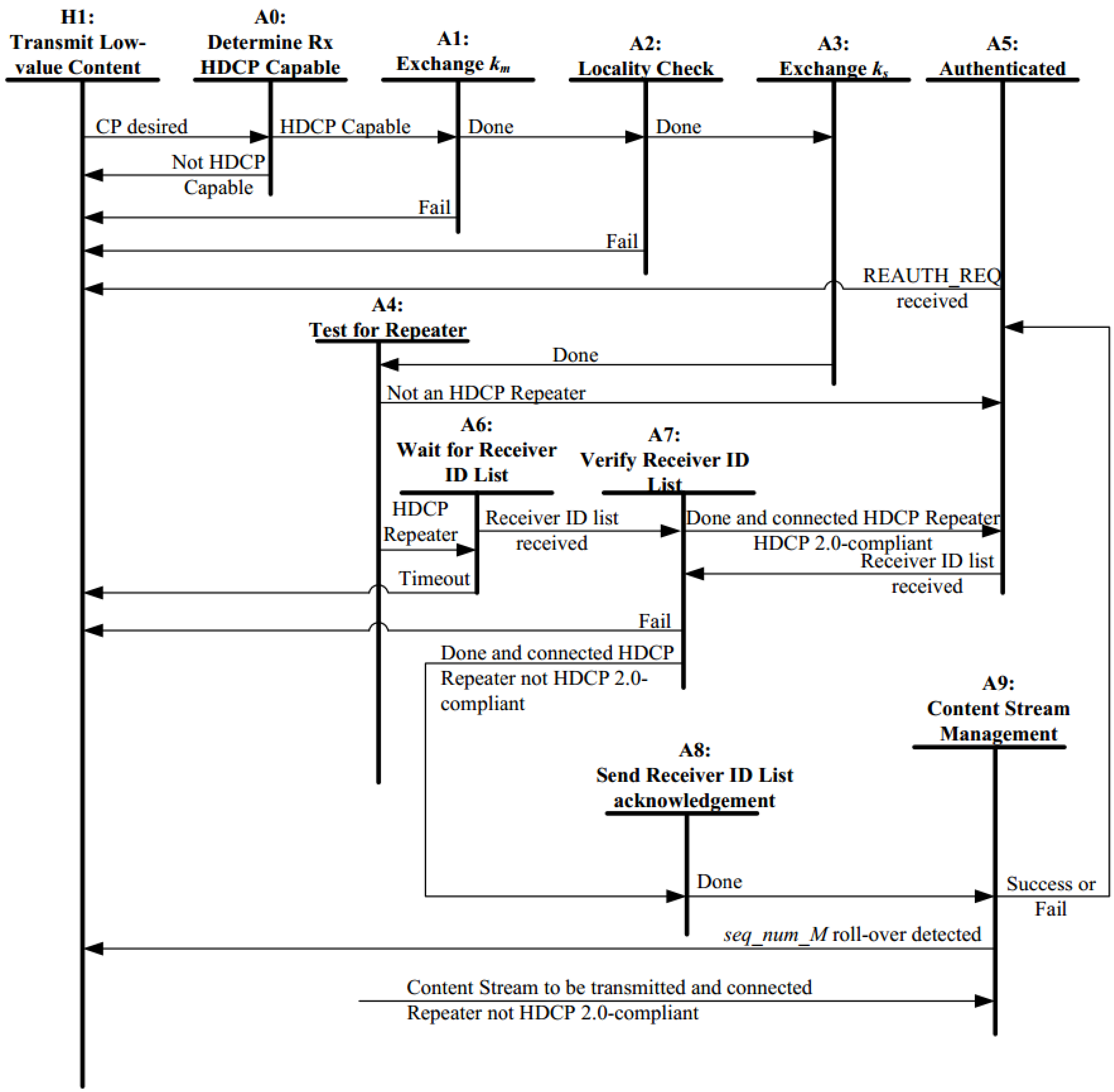

Figure 2.12. HDCP Transmitter Authentication Protocol State Diagram

Transition Any State:H0. Reset conditions at the HDCP Transmitter or disconnect of the connected HDCP capable receiver cause the HDCP Transmitter to enter the No Receiver Attached state.

Transition H0:H1. The detection of a sink device (through Receiver Connected Indication) indicates to the transmitter that a sink device is connected and ready to display the received content. When the receiver is no longer active, the transmitter is notified through Receiver Disconnected Indication.

State H1: Transmit Low-value Content. In this state the transmitter should begin sending an unencrypted signal with HDCP Encryption disabled. The transmitted signal can be a low value content or informative on-screen display. This will ensure that a valid video signal is displayed to the user before and during authentication. At any time a receiver connected indication received from the connected HDCP 2.0-compliant HDCP Repeater causes the transmitter to transition in to this state.

Transition H1:A0. If content protection is desired by the Upstream Content Control Function, then the HDCP Transmitter should immediately attempt to determine whether the receiver is HDCP capable.

State A0: Determine Rx HDCP Capable. The transmitter determines that the receiver is HDCP capable as part of the setup and discovery procedures described in Section 10.6.4 in the WirelessHD specification ([3]). If state A0 is reached when content protection is desired by the Upstream Content Control Function, authentication must be started immediately by the transmitter if the receiver is HDCP capable. A valid video screen is displayed to the user with encryption disabled during this time.

Transition A0:H1. If the receiver is not HDCP capable, the transmitter continues to transmit low value content or informative on-screen display.

Transition A0:A1. If the receiver is HDCP capable, the transmitter initiates the authentication protocol.

State A1: Exchange k_m. In this state, the HDCP Transmitter initiates authentication by sending AKE_Init message containing r_{tx} to the HDCP Receiver. It receives AKE_Send_Cert from the receiver containing REPEATER and $cert_{rx}$.

If the HDCP Transmitter does not have k_m stored corresponding to the *Receiver ID*, it generates $E_{kpub}(km)$ and sends $E_{kpub}(km)$ as part of the AKE_No_Stored_km message to the receiver after verification of signature on $cert_{rx}$. It performs integrity check on the SRM and checks to see whether the *Receiver ID* of the connected HDCP Device is in the revocation list. It receives AKE_Send_rrx message containing r_{rx} from the receiver. It computes H, receives AKE_Send_H_prime message from the receiver containing *H'* within one second after sending AKE_No_Stored_km to the receiver and compares *H'* against H.

If the HDCP Transmitter has k_m stored corresponding to the *Receiver ID*, it sends AKE_Stored_km message containing $E_{kh}(k_m)$ and *m* to the receiver, performs integrity check on the SRM, checks to see whether the *Receiver ID* of the connected HDCP Device is in the revocation list and receives r_{rx} as part of AKE_Send_rrx message from the receiver. It computes H, receives AKE_Send_H_prime message from the receiver containing *H'* within 200 ms after sending AKE_Stored_km to the receiver and compares *H'* against H.

If the HDCP Transmitter does not have a k_m stored corresponding to the *Receiver ID*, it implements pairing with the HDCP Receiver as explained in Section 2.2.1.

Transition A1:H1. This transition occurs on failure of signature verification on $cert_{rx}$, failure of SRM integrity check, if *Receiver ID* of the connected HDCP Device is in the revocation list or if there is a mismatch between H and *H'*. This transition also occurs if AKE_Send_H_prime message is not received within one second after sending AKE_No_Stored_km or within 200 ms after sending AKE_Stored_km to the receiver.

Transition A1:A2. The HDCP Transmitter implements locality check after successful completion of AKE and pairing.

State A2: Locality Check. In this state, the HDCP Transmitter implements the locality check as explained in Section 2.3 with the HDCP Receiver.

Transition A2:H1. This transition occurs on one or more consecutive locality check failures. Locality check fails when the most significant 128-bits of *L'* are not received within 1 ms, or on a mismatch between the most significant 128-bits of L and *L'*.

Transition A2:A3. The HDCP Transmitter implements SKE after successful completion of locality check.

State A3: Exchange k_s. The HDCP Transmitter sends encrypted Session Key, $E_{dkey}(k_s)$, and r_{iv} to the HDCP Receiver as part of the SKE_Send_Eks message. It may enable HDCP Encryption 200 ms after sending encrypted Session Key. HDCP Encryption must be enabled only after successful completion of AKE, locality check and SKE stages.

Transition A3:A4. This transition occurs after completion of SKE.

State A4: Test for Repeater. The HDCP Transmitter evaluates the REPEATER value that was received in State A1.

Transition A4:A5. REPEATER is 'false' (the HDCP Receiver is not an HDCP Repeater).

State A5: Authenticated. At this time, and at no prior time, the HDCP Transmitter has completed the authentication protocol.

A periodic Link Synchronization is performed to maintain cipher synchronization between the HDCP Transmitter and the HDCP Receiver.

Transition A4:A6. REPEATER is 'true' (the HDCP Receiver is an HDCP Repeater).

State A6: Wait for Receiver ID List. The HDCP Transmitter sets up a three-second watchdog timer after sending SKE_Send_Eks.

Transition A6:H1. The watchdog timer expires before the RepeaterAuth_Send_ReceiverID_List is received.

Transition A6:A7. RepeaterAuth_Send_ReceiverID_List message is received.

State A7: Verify Receiver ID List. If a transition in to this state occurs from State A6, the watchdog timer is cleared. If both MAX_DEVS_EXCEEDED and MAX_CASCADE_EXCEEDED are not 'true', computes *V*. If the connected HDCP Repeater is HDCP 2.0-compliant, compares *V* and *V'*. If the connected HDCP Repeater is not HDCP 2.0-compliant, compares the most significant 128-bits of *V* and *V'*. The *Receiver ID*s from the Receiver ID list are compared against the current revocation list.

Transition A7:H1. This transition is made if a mismatch occurs between *V* and *V'* (if the connected HDCP Repeater is HDCP 2.0-compliant) or the most significant 128-bits of *V* and *V'* (if the connected HDCP Repeater is not HDCP 2.0-compliant). This transition is also made if any of the *Receiver ID*s in the Receiver ID list are found in the current revocation list or if the HDCP Transmitter detects a roll-over of *seq_num_V* (if the repeater is not HDCP 2.0-compliant). A MAX_CASCADE_EXCEEDED or MAX_DEVS_EXCEEDED error also causes this transition.

Transition A7:A5. This transition occurs if the connected HDCP Repeater is HDCP 2.0-compliant, on successful verification of *V* and *V'*, none of the reported *Receiver ID*s are in the current revocation list, and the downstream topology does not exceed specified maximums.

Transition A7:A8. This transition occurs if the connected HDCP Repeater is not HDCP 2.0-compliant, on successful verification of the most significant 128-bits of *V* and *V'*, none of the reported *Receiver IDs* are in the current revocation list, the HDCP Transmitter does not detect a roll-over of *seq_num_V* and the downstream topology does not exceed specified maximums.

State A8: Send Receiver ID list acknowledgement. The HDCP Transmitter sends the least significant 128-bits of *V* to the HDCP Repeater as part of the RepeaterAuth_Send_Ack message.

The RepeaterAuth_Send_Ack message must be received by the HDCP Repeater within one second from the transmission of the RepeaterAuth_Send_ReceiverID_List message to the HDCP Transmitter.

Transition A8:A9. This transition occurs after the RepeaterAuth_Send_Ack message has been sent to the repeater.

Transition A5:H1. This transition occurs if a Receiver_AuthStatus message with the REAUTH_REQ set to 'true' is received.

Transition A5:A7. This transition occurs whenever a RepeaterAuth_Send_ReceiverID_List message is received from the connected HDCP Repeater that is not HDCP 2.0-compliant.

State A9: Content Stream Management. This stage is implemented if Content Stream is to be transmitted and the connected HDCP Repeater is not HDCP 2.0-compliant. The HDCP Transmitter sends the RepeaterAuth_Stream_Manage message specifying Type values assigned to Content Streams, to the attached HDCP Repeater at least 100ms before the transmission of the corresponding Content Streams after HDCP Encryption. It must receive the RepeaterAuth_Stream_Ready message from the HDCP Repeater within 100 ms after the transmission of RepeaterAuth_Stream_Manage message and verifies *M'*. This step fails if the RepeaterAuth_Stream_Ready message is not received within 100 ms or if *M* is not equal to *M'*.

This stage may be implemented in parallel with the upstream propagation of topology information (State A4, State A6, State A7 and State A8) and with the flow of encrypted content and Link Synchronization (State A5). This state may be implemented asynchronously from the rest of the state diagram. A transition in to this state may occur from State A4, State A5, State A6, State A7 or State A8 if Content Stream is to be transmitted and the connected HDCP Repeater is not HDCP 2.0-compliant. Also, the transition from State A9 must return to the appropriate state to allow for undisrupted operation.

Note: The HDCP Transmitter must not transmit Type 1 Content Streams to HDCP 1.x-compliant Devices and HDCP 2.0-compliant Repeaters as instructed by the corresponding Upstream Content Control Function.

Transition A9:A5. This transition occurs on success or failure of the Content Stream management stage.

Transition A9:H1. This transition occurs if *seq_num_M* rolls over.

Note: Since Link Synchronization (State A5) may be implemented in parallel with the upstream propagation of topology information (State A4, State A6, State A7 and State A8) and Content Stream management (State A9) stages, the link synchronization process (i.e. State A5) may be implemented asynchronously from the rest of the state diagram. The transition into State A5 may occur from any state for which encryption is currently enabled. Also, the transition from State A5 returns to the appropriate state to allow for undisrupted operation.

The HDCP Transmitter may support simultaneous connections to HDCP Receivers through one or more of its HDCP-protected interface ports. It may share the same Session Key and r_{iv} across all its HDCP-protected interface ports, as explained in Section 3.4. However, the HDCP Transmitter must ensure that each connected HDCP Receiver receives distinct k_m and r_{tx} values.

2.9 HDCP Receiver State Diagram

The operation states of the authentication protocol for an HDCP Receiver that is not an HDCP Repeater are illustrated in

Figure 2.13. For HDCP Repeaters, the upstream (HDCP Receiver) side is covered in Section 2.10.3.

The HDCP Receiver must be ready to re-authenticate with the HDCP Transmitter at any point in time. In particular, the only indication to the HDCP Receiver of a re-authentication attempt by the HDCP Transmitter is the reception of an r_{tx} as part of the AKE_Init message from the HDCP Transmitter.

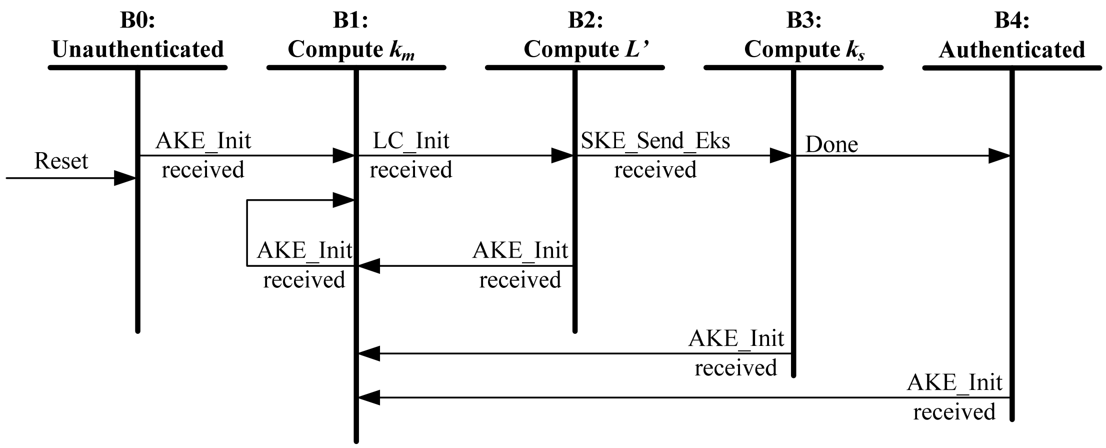

Figure 2.13. HDCP Receiver Authentication Protocol State Diagram

Transition Any State:B0. Reset conditions at the HDCP Receiver cause the HDCP Receiver to enter the unauthenticated state.

State B0: Unauthenticated. The HDCP Receiver is awaiting the reception of r_{tx} from the HDCP Transmitter to trigger the authentication protocol.

Transition B0:B1. r_{tx} is received as part of the AKE_Init message from the HDCP Transmitter.

State B1: Compute k_m. In this state, the HDCP Receiver sends AKE_Send_Cert message in response to AKE_Init, generates and sends r_{rx} as part of AKE_Send_rrx message. If AKE_No_Stored_km is received, it decrypts k_m with kpriv$_{rx}$, calculates H'. It sends AKE_Send_H_prime message immediately after computation of H' to ensure that the message is received by the transmitter within the specified one second timeout at the transmitter.

If AKE_Stored_km is received, the HDCP Receiver decrypts $E_{kh}(k_m)$ to derive k_m and calculates H'. It sends AKE_Send_H_prime message immediately after computation of H' to ensure that the message is received by the transmitter within the specified 200 ms timeout at the transmitter.

If AKE_No_Stored_km is received, this is an indication to the HDCP Receiver that the HDCP Transmitter does not contain a k_m stored corresponding to its *Receiver ID*. It implements pairing with the HDCP Transmitter as explained in Section 2.2.1.

Transition B1: B1. Should the HDCP Transmitter send an AKE_Init while the HDCP Receiver is in State B1, the HDCP Receiver abandons intermediate results and restarts computation of k_m.

Transition B1: B2. The transition occurs when r_n is received as part of LC_Init message from the transmitter.

State B2: Compute *L'*. The HDCP Receiver computes *L'* required during locality check and sends RTT_Response message after receiving the RTT_Challenge message from the transmitter.

Transition B2: B1. Should the HDCP Transmitter send an AKE_Init while the HDCP Receiver is in State B2, the HDCP Receiver abandons intermediate results and restarts computation of k_m.

Transition B2: B3. The transition occurs when SKE_Send_Eks message is received from the transmitter.

State B3: Compute k_s. The HDCP Receiver decrypts $E_{dkey}(k_s)$ to derive k_s.

Transition B3: B1. Should the HDCP Transmitter send an AKE_Init while the HDCP Receiver is in State B3, the HDCP Receiver abandons intermediate results and restarts computation of k_m.

Transition B3: B4. Successful computation of k_s transitions the receiver into the authenticated state.

State B4: Authenticated. The HDCP Receiver has completed the authentication protocol. Periodically, it updates its *inputCtr* using the Secure Packet Counter value received from the transmitter.

Transition B4: B1. Should the HDCP Transmitter send an AKE_Init while the HDCP Receiver is in State B4, the HDCP Receiver abandons intermediate results and restarts computation of k_m.

2.10 HDCP Repeater State Diagrams

The HDCP Repeater has one HDCP-protected Interface connection to an upstream HDCP Transmitter and one or more HDCP-protected Interface connections to downstream HDCP Receivers. The state diagram for each downstream connection (Figure 2.14 and Figure 2.15) is substantially the same as that for the host HDCP Transmitter (Section 2.8), with the exception, that the HDCP Repeater is not required to check for downstream *Receiver IDs* in a revocation list.

When the upstream HDCP-protected interface port of the HDCP Repeater is in an unauthenticated state, it signals the detection of an active downstream HDCP Receiver to the upstream HDCP Transmitter by propagating the Receiver Connected Indication to the upstream HDCP Transmitter.

Whenever authentication is initiated by the upstream HDCP Transmitter by sending AKE_Init, the HDCP Repeater immediately initiates authentication on all its downstream HDCP-protected interface ports if its downstream ports are in an unauthenticated state.

The HDCP Repeater may cache the latest Receiver ID list and topology information received from its downstream ports. Whenever authentication is attempted by the upstream transmitter by sending an r_{tx} value, the HDCP Repeater may propagate the cached Receiver ID list upstream without initiating a re-authentication on all its downstream ports.

The HDCP Repeater must generate unique k_m values for HDCP Devices connected to each of its downstream HDCP-protected Interface Ports.

The HDCP Repeater may transmit the same session key, k_s to all its authenticated and active downstream HDCP-protected Interface Ports before beginning the transmission of HDCP Content to any of its downstream ports. After beginning the transmission of HDCP Content by the HDCP Repeater to any of its downstream ports, subsequent connection of a new HDCP Receiver to its downstream port must result in (a) a unique session key, k_s, exchanged with that HDCP Receiver or (b) a new authentication attempt with all its downstream HDCP-protected Interface ports and subsequent exchange of the same session key, k_s, to all its authenticated and active downstream HDCP-protected Interface Ports.

If an HDCP Repeater has no active downstream HDCP Devices, it must authenticate as an HDCP Receiver with REPEATER set to 'false' if it wishes to receive HDCP Content, but must not pass HDCP Content to downstream devices.

2.10.1 Propagation of Topology Errors

MAX_DEVS_EXCEEDED and MAX_CASCADE_EXCEEDED: HDCP Repeaters must be capable of supporting DEVICE_COUNT values of up to 31 and DEPTH values of up to 4. If the computed DEVICE_COUNT for an HDCP Repeater exceeds 31, the error is referred to as MAX_DEVS_EXCEEDED error. The repeater sets MAX_DEVS_EXCEEDED = 'true' in the RepeaterAuth_Send_ReceiverID_List message. If the computed DEPTH for an HDCP Repeater exceeds four, the error is referred to as MAX_CASCADE_EXCEEDED error. The repeater sets MAX_CASCADE_EXCEEDED = 'true' in the RepeaterAuth_Send_ReceiverID_List message. When an HDCP Repeater receives a MAX_DEVS_EXCEEDED or a MAX_CASCADE_EXCEEDED error from a downstream HDCP Repeater, it must propagate the error to the upstream HDCP Transmitter and must not transmit *V'* and Receiver ID list.

2.10.2 HDCP Repeater Downstream State Diagram

In this state diagram and its following description, the downstream (HDCP Transmitter) side refers to the HDCP Transmitter functionality within the HDCP Repeater for its corresponding downstream HDCP-protected Interface Port.

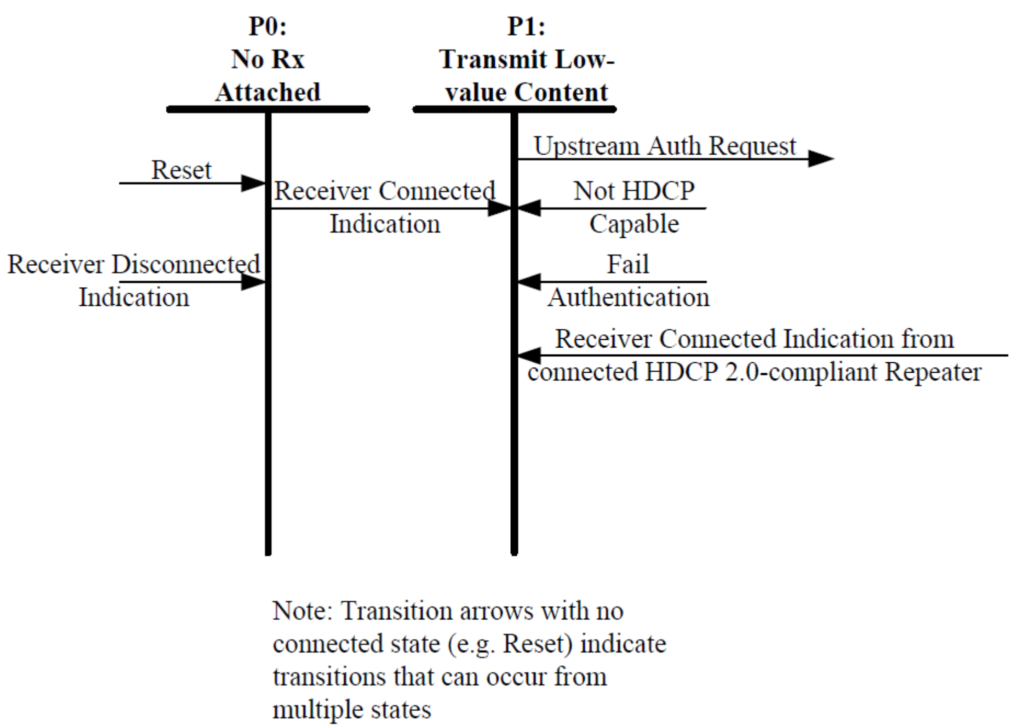

Figure 2.14. HDCP Repeater Downstream Link State Diagram

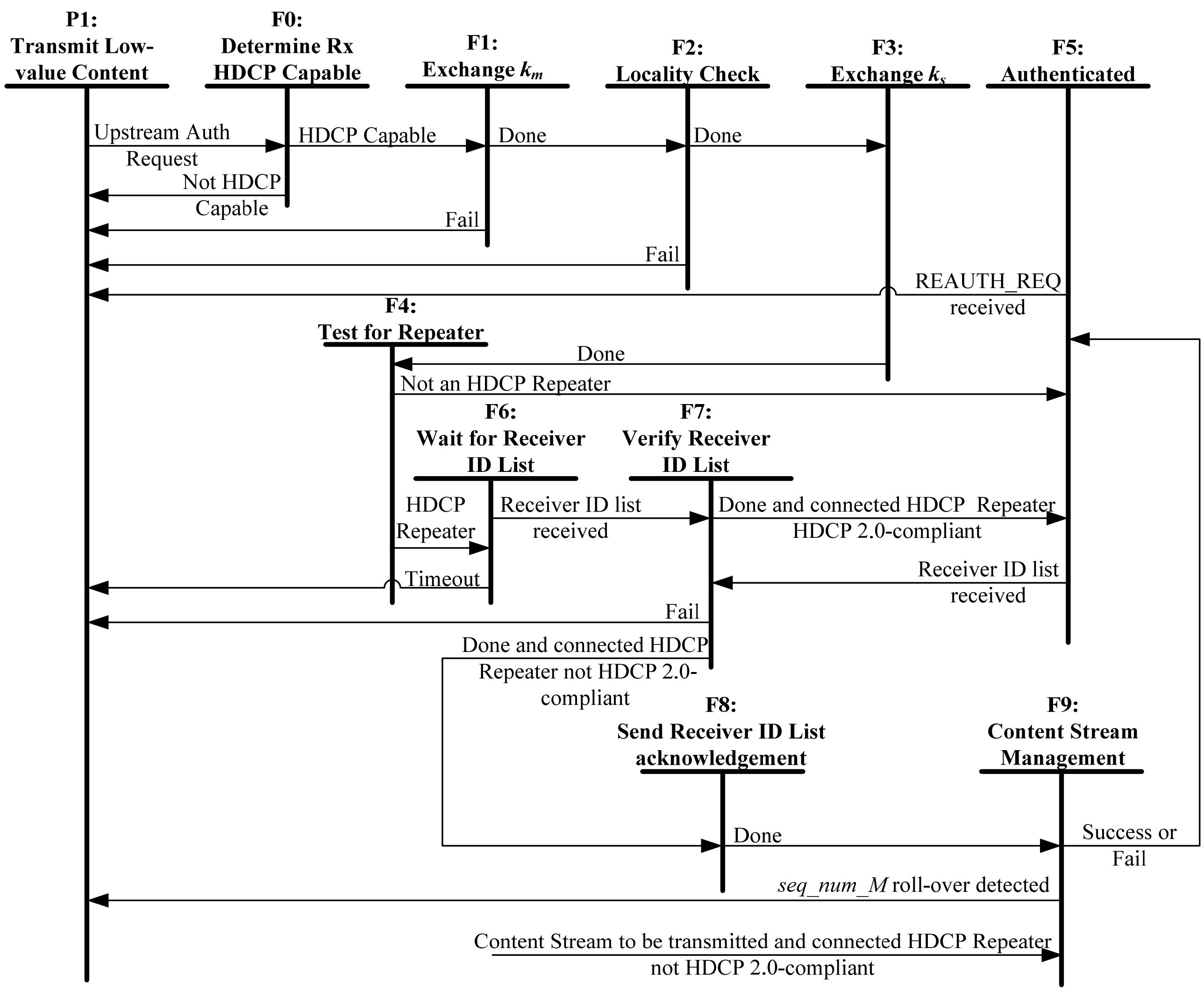

Figure 2.15. HDCP Repeater Downstream Authentication Protocol State Diagram

Transition Any State:P0. Reset conditions at the HDCP Repeater or disconnect of the connected HDCP capable receivers cause the HDCP Repeater to enter the No Receiver Attached state for this port.

Transition P0:P1. The detection of a sink device (through Receiver Connected Indication) indicates that the receiver is available and active (ready to display received content). When the receiver is no longer active, the downstream (HDCP Transmitter) side is notified through Receiver Disconnected Indication.

State P1: Transmit low-value content. In this state the downstream side should begin sending the unencrypted video signal received from the upstream HDCP Transmitter with HDCP Encryption disabled. At any time a Receiver Connected Indication received from the connected HDCP 2.0-compliant HDCP Repeater causes the downstream side to transition in to this state.

Transition P1:F0. Upon an Upstream Authentication Request, the downstream side should immediately attempt to determine whether the receiver is HDCP capable.

State F0: Determine Rx HDCP Capable. The downstream side determines that the receiver is HDCP capable as part of the setup and discovery procedures defined in Section 10.6.4 of the WirelessHD specification ([3]). If state F0 is reached upon an Upstream Authentication Request, authentication must be started immediately by the downstream side if the receiver is HDCP capable. A valid video screen is displayed to the user with encryption disabled during this time.

Note: The downstream side may initiate authentication before an Upstream Authentication Request is received.

Transition F0:P1. If the receiver is not HDCP capable, the downstream side continues to transmit low value content or informative on-screen display received from the upstream HDCP Transmitter.

Transition F0:F1. If the receiver is HDCP capable, the downstream side initiates the authentication protocol.

State F1: Exchange k_m. In this state, the downstream side initiates authentication by sending AKE_Init message containing r_{tx} to the HDCP Receiver. It receives AKE_Send_Cert from the receiver containing REPEATER and $cert_{rx}$.

If the downstream side does not have k_m stored corresponding to the *Receiver ID*, it generates $E_{kpub}(km)$ and sends $E_{kpub}(km)$ as part of the AKE_No_Stored_km message to the receiver after verification of signature on $cert_{rx}$. It receives AKE_Send_rrx message containing r_{rx} from the receiver. It computes H, receives AKE_Send_H_prime message from the receiver containing *H'* within one second after sending AKE_No_Stored_km to the receiver and compares *H'* against H.

If the downstream side has k_m stored corresponding to the *Receiver ID*, it sends AKE_Stored_km message containing $E_{kh}(k_m)$ and m to the receiver and receives r_{rx} as part of AKE_Send_rrx message from the receiver. It computes H, receives AKE_Send_H_prime message from the receiver containing *H'* within 200 ms after sending AKE_Stored_km to the receiver and compares *H'* against H.

If the downstream side does not have a k_m stored corresponding to the *Receiver ID*, it implements pairing with the HDCP Receiver as explained in Section 2.2.1.

Transition F1:P1. This transition occurs on failure of signature verification on $cert_{rx}$ or if there is a mismatch between H and *H'*. This transition also occurs if AKE_Send_H_prime message is not received within one second after sending AKE_No_Stored_km or within 200 ms after sending AKE_Stored_km to the receiver.

Transition F1:F2. The downstream side implements locality check after successful completion of AKE and pairing.

State F2: Locality Check. In this state, the downstream side implements the locality check with the HDCP Receiver as explained in Section 2.3.

Transition F2:P1. This transition occurs on one or more consecutive locality check failures. Locality check fails when the most significant 128-bits of *L'* are not received within 1 ms, or on a mismatch between the most significant 128-bits of L and *L'*.

Transition F2:F3. The downstream side implements SKE after successful completion of locality check.

State F3: Exchange k_s. The downstream side sends encrypted Session Key, $E_{dkey}(k_s)$, and r_{iv} to the HDCP Receiver as part of the SKE_Send_Eks message. It may enable HDCP Encryption 200 ms after sending encrypted Session Key. HDCP Encryption must be enabled only after successful completion of AKE, locality check and SKE stages.

Transition F3:F4. This transition occurs after completion of SKE.

State F4: Test for Repeater. The downstream side evaluates the REPEATER value that was received in State F1.

Transition F4:F5. REPEATER is 'false' (the HDCP Receiver is not an HDCP Repeater).

State F5: Authenticated. At this time, and at no prior time, the downstream side has completed the authentication protocol.

A periodic Link Synchronization is performed to maintain cipher synchronization between the downstream side and the HDCP Receiver.

Transition F4:F6. REPEATER is 'true' (the HDCP Receiver is an HDCP Repeater).

State F6: Wait for Receiver ID List. The downstream side sets up a three-second watchdog timer after sending SKE_Send_Eks.

Transition F6:P1. The watchdog timer expires before the RepeaterAuth_Send_ReceiverID_List is received.

Transition F6:F7. RepeaterAuth_Send_ReceiverID_List message is received.

State F7: Verify Receiver ID List. If a transition in to this state occurs from State F6 the watchdog timer is cleared. If both MAX_DEVS_EXCEEDED and MAX_CASCADE_EXCEEDED are not 'true', computes *V*. If the connected HDCP Repeater is HDCP 2.0-compliant, compares *V* and *V'*. If the connected HDCP Repeater is not HDCP 2.0-compliant, compares the most significant 128-bits of *V* and *V'*. The *Receiver ID*s from this port are added to the Receiver ID list for this HDCP Repeater. The upstream HDCP Transmitter must be informed if topology maximums are exceeded.

Transition F7:P1. This transition is made if a mismatch occurs between *V* and *V*' (if the connected HDCP Repeater is HDCP 2.0-compliant) or the most significant 128-bits of *V* and *V'* (if the connected HDCP Repeater is not HDCP 2.0-compliant). This transition is also made if the downstream side detects a roll-over of *seq_num_V* (if the repeater is not HDCP 2.0-compliant). A MAX_CASCADE_EXCEEDED or MAX_DEVS_EXCEEDED error also causes this transition.

Transition F7:F5. This transition is made if the connected HDCP Repeater is HDCP 2.0-compliant, on successful verification of *V* and *V*' and the downstream topology does not exceed specified maximums.

Transition F7:F8. This transition occurs if the connected HDCP Repeater is not HDCP 2.0-compliant, on successful verification of the most significant 128-bits of *V* and *V'*, the downstream side does not detect a roll-over of *seq_num_V* and the downstream topology does not exceed specified maximums.

State F8: Send Receiver ID list acknowledgement. The downstream side sends the least significant 128-bits of *V* to the HDCP Repeater as part of the RepeaterAuth_Send_Ack message.

The RepeaterAuth_Send_Ack message must be received by the HDCP Repeater within one second from the transmission of the RepeaterAuth_Send_ReceiverID_List message to the downstream side.

Transition F8:F9. This transition occurs after the RepeaterAuth_Send_Ack message has been sent to the repeater.

Transition F5:P1. This transition occurs if a Receiver_AuthStatus message with the REAUTH_REQ set to 'true' is received.

Transition F5:F7. This transition occurs whenever a RepeaterAuth_Send_ReceiverID_List message is received from the connected HDCP Repeater that is not HDCP 2.0-compliant.

State F9: Content Stream Management. This stage is implemented if Content Stream is to be transmitted and the connected HDCP Repeater is not HDCP 2.0-compliant. The downstream side propagates the Content Stream management information, received from the upstream transmitter, using the RepeaterAuth_Stream_Manage message to the attached HDCP Repeater at least 100ms before the transmission of the corresponding Content Streams after HDCP Encryption. If the upstream transmitter is HDCP 2.0-compliant or HDCP 1.x-compliant, the downstream side will not receive the RepeaterAuth_Stream_Manage message from the upstream transmitter and assigns a Type value of 0x00 to all Content Streams received from the upstream transmitter and propagates the Content Stream management information using the RepeaterAuth_Stream_Manage message.

The downstream side must receive the RepeaterAuth_Stream_Ready message from the HDCP Repeater within 100 ms after the transmission of RepeaterAuth_Stream_Manage message and verifies *M'*. This step fails if the RepeaterAuth_Stream_Ready message is not received within 100 ms or if *M* is not equal to *M'*.

This stage may be implemented in parallel with the upstream propagation of topology information (State F4, State F6, State F7 and State F8) and with the flow of encrypted content and Link Synchronization (State F5). This state may be implemented asynchronously from the rest of the state diagram. A transition in to this state may occur from State F4, State F5, State F6, State F7 or State F8 if Content Stream is to be transmitted and the connected HDCP Repeater is not HDCP 2.0-compliant and the Content Stream management information is received from the upstream HDCP Transmitter. Also, the transition from State F9 must return to the appropriate state to allow for undisrupted operation.

Note: Type 1 Content Streams must not be transmitted by the downstream side through its HDCP-protected Interface Ports connected to HDCP 1.x-compliant Devices and HDCP 2.0-compliant Repeaters.

Transition F9:F5. This transition occurs on success or failure of the Content Stream management stage.

Transition F9:P1. This transition occurs if *seq_num_M* rolls over.

Note: Since Link Synchronization may be implemented in parallel with the upstream propagation of topology information (State F4, State F6, State F7 and State F8) and Content Stream management (State F9) stages, the link synchronization process (i.e. State F5) may be implemented asynchronously from the rest of the state diagram. The transition into State F5 may occur from any state for which encryption is currently enabled. Also, the transition from State F5 returns to the appropriate state to allow for undisrupted operation.

2.10.3 HDCP Repeater Upstream State Diagram

The HDCP Repeater upstream state diagram, illustrated in 2.16, makes reference to states of the HDCP Repeater downstream state diagram. In this state diagram and its following description, the upstream (HDCP Receiver) side refers to the HDCP Receiver functionality within the HDCP Repeater for its corresponding upstream HDCP-protected Interface Port.

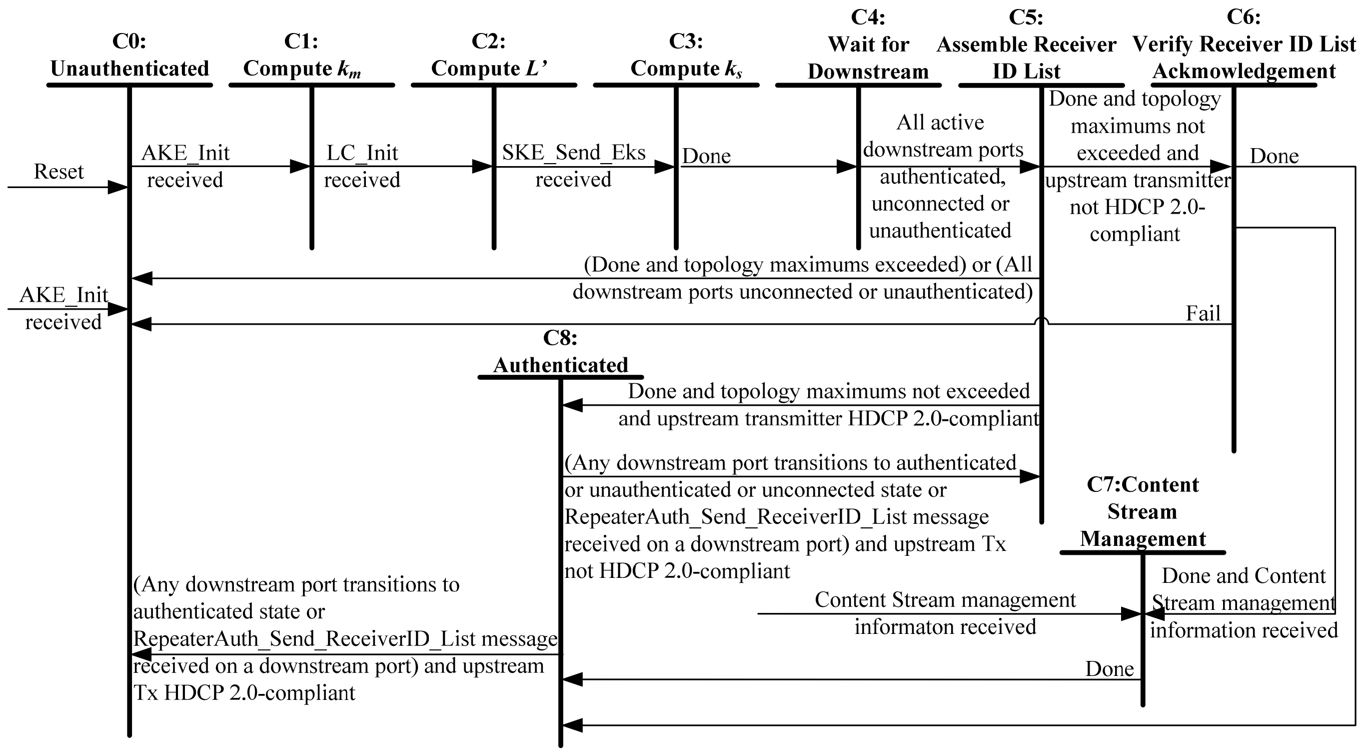

Figure 2.16. HDCP Repeater Upstream Authentication Protocol State Diagram

Transitions Any State:C0. Reset conditions at the HDCP Repeater cause the HDCP Repeater to enter the unauthenticated state. Re-authentication is forced any time AKE_Init is received from the connected HDCP Transmitter, with a transition through the unauthenticated state.

State C0: Unauthenticated. The device is idle, awaiting the reception of r_{tx} from the HDCP Transmitter to trigger the authentication protocol.

Transition C0:C1. r_{tx} is received as part of the AKE_Init message from the HDCP Transmitter.

State C1: Compute k_m. In this state, the upstream (HDCP Receiver) side sends AKE_Send_Cert message in response to AKE_Init, generates and sends r_{rx} as part of AKE_Send_rrx message. If AKE_No_Stored_km is received, it decrypts k_m with kpriv$_{rx,}$ calculates *H'*. It sends AKE_Send_H_prime immediately after computation of *H'* to ensure that the message is received by the transmitter within the specified one second timeout at the transmitter.

If AKE_Stored_km is received, the upstream side decrypts $E_{kh}(k_m)$ to derive k_m and calculates *H'*. It sends AKE_Send_H_prime message immediately after computation of *H'* to ensure that the message is received by the transmitter within the specified 200 ms timeout at the transmitter

If AKE_No_Stored_km is received, this is an indication to the upstream side that the HDCP Transmitter does not contain a k_m stored corresponding to its *Receiver ID*. It implements pairing with the HDCP Transmitter as explained in Section 2.2.1.

Transition C1:C2. The transition occurs when r_n is received as part of LC_Init message from the transmitter.

State C2: Compute *L'*. The upstream side computes *L'* required during locality check and sends RTT_Response message after receiving the RTT_Challenge message from the transmitter.

Transition C2: C3. The transition occurs when SKE_Send_Eks message is received from the transmitter.

State C3: Compute k_s. The upstream side decrypts $E_{dkey}(k_s)$ to derive k_s.

Transition C3: C4. Successful computation of k_s causes this transition.

State C4: Wait for Downstream. The upstream state machine waits for all downstream HDCP-protected Interface Ports of the HDCP Repeater to enter the unconnected (State P0), unauthenticated (State P1), or the authenticated state (State F5).

Transition C4:C5. All downstream HDCP-protected Interface Ports with connected HDCP Receivers have reached the state of authenticated, unconnected or unauthenticated state.

State C5: Assemble Receiver ID List. The upstream side assembles the list of all connected downstream topology HDCP Devices as the downstream HDCP-protected Interface Ports reach terminal states of the authentication protocol. An HDCP-protected Interface Port that advances to State P0, the unconnected state, or P1, the unauthenticated state, does not add to the list. A downstream HDCP-protected Interface Port that arrives in State F5 that has an HDCP Receiver that is not an HDCP Repeater connected, adds the *Receiver ID* of the connected HDCP Receiver to the list. Downstream HDCP-protected Interface Ports that arrive in State F5 that have an HDCP Repeater connected will cause the Receiver ID list read from the connected HDCP Repeater, plus the *Receiver ID* of the connected HDCP Repeater itself, to be added to the list.

Note: The upstream side may add the Receiver ID list read from the HDCP Repeater connected to the downstream HDCP-protected Interface port, plus the Receiver ID of the connected HDCP Repeater itself to the list after the downstream port has transitioned in to State F8.

When the Receiver ID list for all downstream HDCP Receivers has been assembled, the upstream side computes DEPTH, DEVICE_COUNT and the upstream *V'* and sends RepeaterAuth_Send_ReceiverID_List message to the upstream HDCP Transmitter. In the case of a MAX_DEVS_EXCEEDED or a MAX_CASCADE_EXCEEDED error, it does not transmit *V'* (or the most significant 128-bits of *V'*), DEPTH, DEVICE_COUNT, Receiver ID list and if applicable, HDCP2_0_REPEATER_DOWNSTREAM and HDCP1_DEVICE_DOWNSTREAM. When an HDCP Repeater receives a MAX_DEVS_EXCEEDED or MAX_CASCADE_EXCEEDED error from a downstream HDCP Repeater, it is required to inform the upstream HDCP Transmitter.

If any downstream port connected to an HDCP Repeater receives HDCP2_0_REPEATER_DOWNSTREAM = 'true' or HDCP1_DEVICE_DOWNSTREAM ='true', the upstream side sets the corresponding values to 'true' in the RepeaterAuth_Send_ReceiverID_List message to the upstream HDCP Transmitter.

Transition C5:C0. This transition occurs if RepeaterAuth_Send_ReceiverID_List message has been sent to the upstream HDCP Transmitter and topology maximums are exceeded i.e. on a MAX_DEVS_EXCEEDED or MAX_CASCADE_EXCEEDED error. This transition also occurs if all downstream HDCP-protected Interface Ports have reached the state of unconnected or unauthenticated.

Transition C5:C6. RepeaterAuth_Send_ReceiverID_List message has been sent to the upstream HDCP Transmitter and topology maximums are not exceeded and upstream transmitter is not HDCP 2.0-compliant.

Transition C5:C8. RepeaterAuth_Send_ReceiverID_List message has been sent to the upstream HDCP Transmitter and topology maximums are not exceeded and upstream transmitter is HDCP 2.0-compliant.

State C6. Verify Receiver ID list acknowledgement. In this state, the upstream side receives the RepeaterAuth_Send_Ack message from the upstream transmitter and compares the least significant 128-bits of *V* and *V*'. A match between the least significant 128-bits of *V* and *V*' indicates successful upstream transmission of topology information. The RepeaterAuth_Send_Ack message must be received by the upstream side within one second from the transmission of the RepeaterAuth_Send_ReceiverID_List message to the upstream transmitter if the transmitter is not HDCP 2.0-compliant.

Transition C6:C0. This transition occurs if the RepeaterAuth_Send_Ack message is not received by the upstream side within one second or on a mismatch between the least significant 128-bits of *V* and *V*'. If this transition occurs, the upstream side must send the Receiver_AuthStatus message with the REAUTH_REQ set to 'true' to the upstream transmitter.

Transition C6:C7. This transition occurs if the RepeaterAuth_Send_Ack message is received by the upstream side within one second, on a successful match between the least significant 128-bits of *V* and *V*' and if Content Stream management information is received from the upstream transmitter.

Transition C6:C8. This transition occurs if the RepeaterAuth_Send_Ack message is received by the upstream side within one second and on a successful match between the least significant 128-bits of *V* and *V*'.

State C7: Content Stream Management. On receiving the RepeaterAuth_Stream_Manage message, the upstream side computes *M*' and sends it to the upstream Transmitter as part of the RepeaterAuth_Stream_Ready message.

This stage may be implemented in parallel with the upstream propagation of topology information (State C4, State C5 and State C6) and with the flow of encrypted content and link synchronization (State C8). This state may be implemented asynchronously from the rest of the state diagram. A transition in to this state may occur from State C4, State C5, State C6 or State C8 if Content Stream management information is received from the upstream transmitter. Also, the transition from State C7 may return to the appropriate state to allow for undisrupted operation.

The upstream side must be prepared to implement this stage in parallel with the upstream propagation of topology information and with the flow of encrypted content and link synchronization if these stages are implemented in parallel by the upstream transmitter.

Transition C7:C8. This transition occurs after RepeaterAuth_Stream_Ready message has been sent to the upstream transmitter.

State C8: Authenticated. The upstream side has completed the authentication protocol. Periodically, it updates its *inputCtr* using the Secure Packet Counter value received from the transmitter.

Transition C8:C5. This transition occurs only if the upstream HDCP Transmitter is not HDCP 2.0-compliant and on detection of any changes to the topology.

This transition occurs when a downstream port that was previously in the unauthenticated (State P1) or unconnected (State P0) state transitions in to the authenticated (State F5) state. For example, the transition may occur when a new HDCP Receiver is connected to a downstream port, that previously had no receivers connected, and the downstream port completes the authentication protocol with the HDCP Receiver.

This transition also occurs when a downstream port that was previously in an authenticated state transitions in to an unauthenticated on unconnected state. For example, the transition may occur when an active, authenticated HDCP Receiver attached to the downstream port is disconnected.

Reception of a RepeaterAuth_Send_ReceiverID_List message on a downstream port from the connected downstream HDCP Repeater also causes this transition.

Transition C8:C0. This transition occurs only if the upstream HDCP Transmitter is HDCP 2.0-compliant and on detection of any changes to the topology.

This transition occurs when a downstream port that was previously in the unauthenticated (State P1) or unconnected (State P0) state transitions in to the authenticated (State F5) state. For example, the transition may occur when a new HDCP Receiver is connected to a downstream port, that previously had no receivers connected, and the downstream port completes the authentication protocol with the HDCP Receiver.

Reception of a RepeaterAuth_Send_ReceiverID_List message on a downstream port from the connected downstream HDCP Repeater also causes this transition.

If this transition occurs, the upstream side must propagate a Receiver Connected Indication to the upstream HDCP Transmitter.

Note: Since Link Synchronization may be implemented in parallel with the upstream propagation of topology information (State C4, State C5 and State C6) and Content Stream management (State C7), the link synchronization process (i.e. State C8) may be implemented asynchronously from the rest of the state diagram. The transition into State C8 may occur from any state for which encryption is currently enabled. Also, the transition from state C8 may return to the appropriate state to allow for undisrupted operation.

The upstream side must be prepared to implement the link synchronization process in parallel with the upstream propagation of topology information and Content Stream management if these stages are implemented in parallel by the upstream transmitter.

2.11 Converters

2.11.1 HDCP 2 – HDCP 1.x Converters

HDCP 2 – HDCP 1.x converters are HDCP Repeaters with an HDCP 2 compliant interface port on the upstream (HDCP Receiver) side and one or more HDCP 1.x compliant interface ports on the downstream (HDCP Transmitter) side.

The HDCP 1.x compliant downstream side implements the state diagram explained in the corresponding HDCP 1.x specification (See Section 1.5).

The HDCP 2 compliant upstream side implements the state diagram as explained in Section 2.10.3 with these modifications.

- **State C5: Assemble Receiver ID List**. The upstream side assembles the list of all connected downstream topology HDCP Devices as the downstream HDCP-protected Interface Ports reach terminal states of the authentication protocol. An HDCP-protected Interface Port that advances to the unconnected state or the unauthenticated state does not add to the list. A downstream HDCP-protected Interface Port that arrives in an authenticated state that has an HDCP Receiver that is not an HDCP Repeater connected, adds the *Bksv* of the connected HDCP Receiver to the Receiver ID list. Downstream HDCP-protected Interface Ports that arrive in an authenticated state that have an HDCP Repeater connected will cause the KSV list read from the connected HDCP Repeater, plus the *Bksv* of the connected HDCP Repeater itself, to be added to the list. KSVs are used in place of *Receiver IDs* and are added to the Receiver ID list in big-endian order.

When the Receiver ID list (comprising KSVs of connected downstream HDCP 1.x Receivers, where the KSVs are added to the list in big-endian order) for all downstream HDCP Receivers has been assembled, the upstream side computes DEPTH, DEVICE_COUNT and the upstream *V'* and sends RepeaterAuth_Send_ReceiverID_List message to the upstream HDCP Transmitter. In the case of a MAX_DEVS_EXCEEDED or a MAX_CASCADE_EXCEEDED error, it does not transmit *V'* (or the most significant 128-bits of *V'*), DEPTH, DEVICE_COUNT, Receiver ID list and if applicable, HDCP2_0_REPEATER_DOWNSTREAM and HDCP1_DEVICE_DOWNSTREAM. When an HDCP Repeater receives a MAX_DEVS_EXCEEDED or MAX_CASCADE_EXCEEDED error from a downstream HDCP Repeater, it is required to inform the upstream HDCP Transmitter.

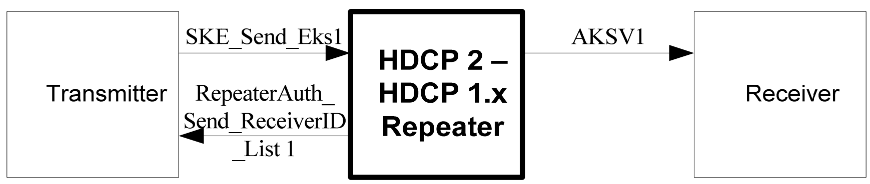

Figure 2.17. HDCP 2 – HDCP 1.x Repeater Protocol Timing with Receiver Attached

From	To	Max Delay	Conditions and Comments
SKE_Send_Eks1 Session Key received from Upstream HDCP Transmitter	AKSV1 HDCP Repeater's *Aksv* transmitted downstream	100 ms	Downstream propagation time.
AKSV1 HDCP Repeater's *Aksv* transmitted downstream	RepeaterAuth_Send_ReceiverID_List1 *Receiver IDs* and topology information transmitted upstream	200 ms	Upstream propagation time when no downstream HDCP Repeaters are attached (no downstream KSV lists to process)

Table 2.3. HDCP 2 – HDCP 1.x Repeater Protocol Timing with Receiver Attached

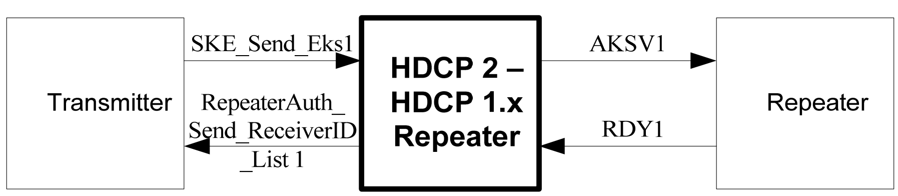

Figure 2.18. HDCP 2 – HDCP 1.x Repeater Protocol Timing with Repeater Attached

From	To	Max Delay	Conditions and Comments
SKE_Send_Eks1 Session key received from	AKSV1 HDCP Repeater's *Aksv* transmitted	100 ms	Downstream propagation time.

Upstream HDCP Transmitter	downstream		
RDY1 Downstream *Receiver IDs* and topology information received	RepeaterAuth_Send_ReceiverID_List1 *Receiver ID*s and topology information transmitted upstream	200 ms	Upstream propagation time when one or more HDCP 1.x-compliant Repeaters are attached. From latest downstream READY. (downstream KSV lists must be processed)

Table 2.4. HDCP 2 – HDCP 1.x Repeater Protocol Timing with Receiver Attached

2.11.2 HDCP 1.x – HDCP 2 Converters

HDCP 1.x – HDCP 2 converters are HDCP Repeaters with an HDCP 1.x compliant interface port on the upstream (HDCP Receiver) side and one or more HDCP 2 compliant interface ports on the downstream (HDCP Transmitter) side.

An HDCP l.x – HDCP 2 Converter, that receives a DEVICE_COUNT value equal to 1 and DEPTH equal to 3 from its downstream port, must set DEVICE_COUNT value equal to 1 and DEPTH equal to 7 on its upstream port for transmission to the HDCP l.x-compliant upstream transmitter.

The HDCP 1.x compliant upstream side implements the state diagram explained in the corresponding HDCP 1.x specification (See Section 1.5).
The HDCP 2 compliant downstream side implements the state diagram as explained in Section 2.10.2 with these modifications.

- **State F7: Verify Receiver ID List**. If a transition in to this state occurs from State F6, the watchdog timer is cleared. If both MAX_DEVS_EXCEEDED and MAX_CASCADE_EXCEEDED are not 'true', computes *V*. If the connected HDCP Repeater is HDCP 2.0-compliant, compares *V* and *V'*. If the connected HDCP Repeater is not HDCP 2.0-compliant, compares the most significant 128-bits of *V* and *V'*. The *Receiver ID*s from this port are used in place of KSVs and are added to the KSV list for this HDCP Repeater. KSV list is constructed by appending *Receiver ID*s in little-endian order. The upstream HDCP Transmitter must be informed if topology maximums are exceeded.

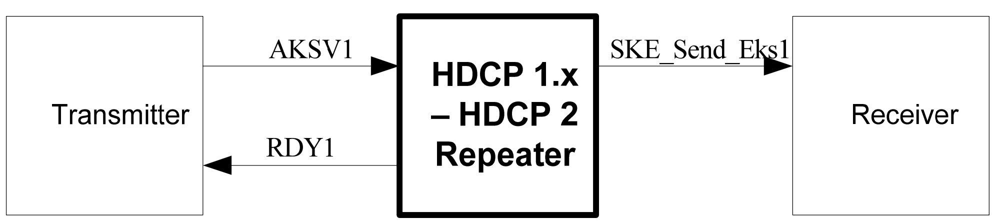

Figure 2.19. HDCP 1.x – HDCP 2 Repeater Protocol Timing with Receiver Attached

From	To	Max Delay	Conditions and Comments
AKSV1 Upstream HDCP Transmitter's *Aksv* received	SKE_Send_Eks1 k_s generated by HDCP Repeater transmitted	400 ms	Downstream propagation time.

	downstream		
SKE_Send_Eks1 k_s, generated by HDCP Repeater transmitted downstream	RDY1 Upstream READY asserted	500 ms	Upstream propagation time when no downstream HDCP Repeaters are attached (no downstream Receiver ID lists to process)

Table 2.5. HDCP 1.x – HDCP 2 Repeater Protocol Timing with Receiver Attached

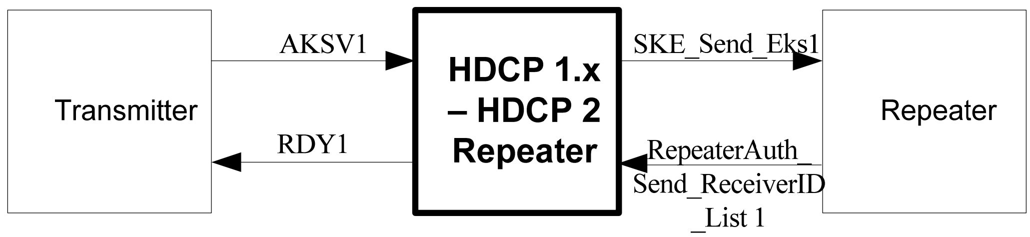

Figure 2.20. HDCP 1.x – HDCP 2 Repeater Protocol Timing with Repeater Attached

From	To	Max Delay	Conditions and Comments
AKSV1 Upstream HDCP Transmitter's *Aksv* received	SKE_Send_Eks1 k_s generated by HDCP Repeater transmitted downstream	400 ms	Downstream propagation time.
RepeaterAuth_Send_ReceiverID_List 1 Downstream *Receiver IDs* and topology information received	RDY1 Upstream READY asserted	500 ms	Upstream propagation time when one or more HDCP Repeaters are attached. From latest downstream RepeaterAuth_Send_ReceiverID_List message. (downstream Receiver ID lists must be processed)

Table 2.6. HDCP 1.x – HDCP 2 Repeater Protocol Timing with Repeater Attached

2.12 Session Key Validity

When HDCP Encryption is disabled, the transmitter and receiver ceases to perform HDCP Encryption (Section 0) and stops incrementing the *inputCtr*.

If HDCP Encryption was disabled, from its enabled state, due to the detection of Receiver Connected Indication, Receiver Disconnected Indication or authentication failures, the HDCP Transmitter expires the Session Key. The HDCP Transmitter initiates re-authentication by the transmission of a new r_{tx}. In all other cases, where HDCP Encryption was disabled, from its enabled state, while the link was still active and authenticated (for e.g., HDCP Encryption may be briefly disabled during transmission of low value content), the HDCP Transmitter need not expire the Session Key. The HDCP Transmitter may maintain the encryption parameters used during the HDCP Session i.e. *inputCtr* value after the last HDCP Encryption operation (after which HDCP Encryption was disabled), k_s and r_{iv}. When encryption is re-enabled, HDCP Encryption may be applied seamlessly, without requiring re-authentication, by using the same encryption parameters.

If HDCP Encryption was disabled, from its enabled state, the HDCP Receiver must maintain k_s and r_{iv} used during the HDCP Session. If encryption was re-enabled, without intervening re-authentication requests from the transmitter, the HDCP Receiver must use the same k_s and r_{iv}. It must update its *inputCtr* with the *inputCtr* value received from the transmitter. (See Section 2.6 on Link Synchronization).

2.13 Random Number Generation

Random number generation is required both in the HDCP Transmitter logic and in the HDCP Receiver logic. Counter mode based deterministic random bit generator using AES-128 block cipher specified in NIST SP 800-90 is the recommended random number generator. The minimum entropy requirement for random values that are not used as secret key material (i.e. r_{tx}, r_{rx}, r_{iv}, r_n) is 40 random bits out of 64-bits. This means that a reasonable level of variability or entropy is established if out of 1,000,000 random (r_{tx}, r_{rx}, r_{iv} or r_n) values collected after the first authentication attempt (i.e. after power-up cycles on the HDCP Transmitter or HDCP Receiver logic), the probability of there being any duplicates in this list of 1,000,000 random values is less than 50%.

For randomly generated secret key material (k_m, k_s) the minimum entropy requirement is 128-bits of entropy (i.e. the probability of there being any duplicates in the list of 2^64 secret values (k_m or k_s) collected after power-up and first authentication attempt on the HDCP Transmitter logic is less than 50%).

A list of possible entropy sources that may be used for generation of random values used as secret key material include

- a true Random Number Generator or analog noise source, even if a poor (biased) one
- a pseudo-random number generator (PRNG), seeded by a true RNG with the required entropy, where the state is stored in non-volatile memory after each use. The state must be kept secret. Flash memory or even disk is usable for this purpose as long as it is secure from tampering.

A list of possible entropy sources that may be used for generation of random values not used as secret key material include

- timers, network statistics, error correction information, radio/cable television signals, disk seek times, etc.
- a reliable (not manipulatable by the user) calendar and time-of-day clock. For example, some broadcast content sources may give reliable date and time information.

3 HDCP Encryption

3.1 Description

Figure 3.1 shows how HDCP fits in to the WirelessHD protocol stack. The link consists of two constituent links: a unidirectional high-speed stream transporting the AV Content, and a lower-speed bidirectional link used for control / status.

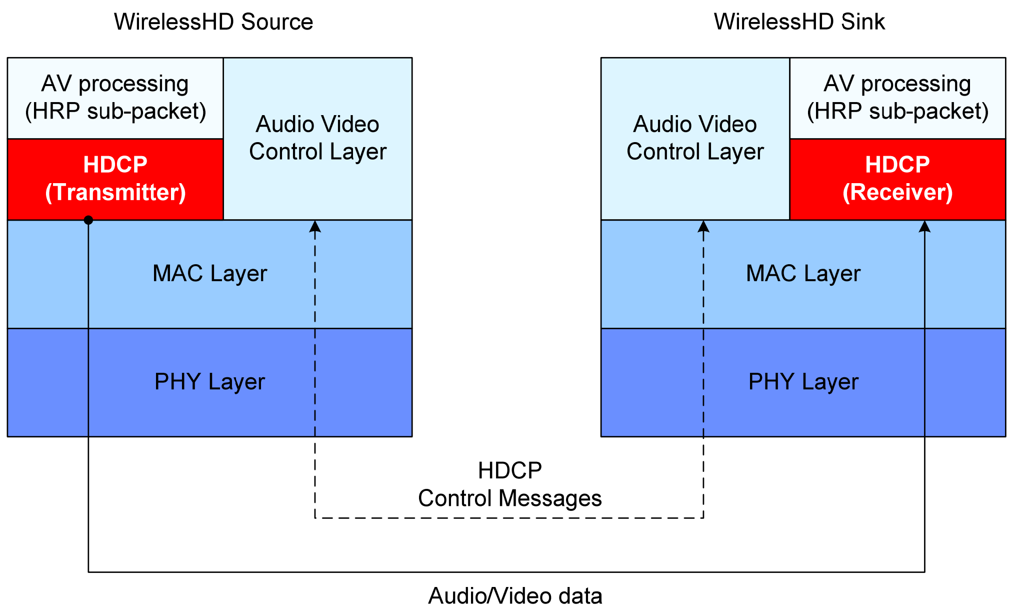

Figure 3.1. WirelessHD with HDCP Block Diagram (Informative)

Video in the HDCP Transmitter is assumed to be a stream of pixel samples, together with any associated audio or data streams, they are carried over the WirelessHD channel as HRP packetized streams specified in [3].

3.2 AV Stream

WirelessHD AV streams are packetized and carried in the sub-packet(s) payload. The HRP packet payload consists of sub-packets. Each sub-packet contains either Video, Audio or Data. Only the Audio and Video sub-packets are subject to HDCP Encryption. Audio and Video may be in compressed or uncompressed form

3.3 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**.

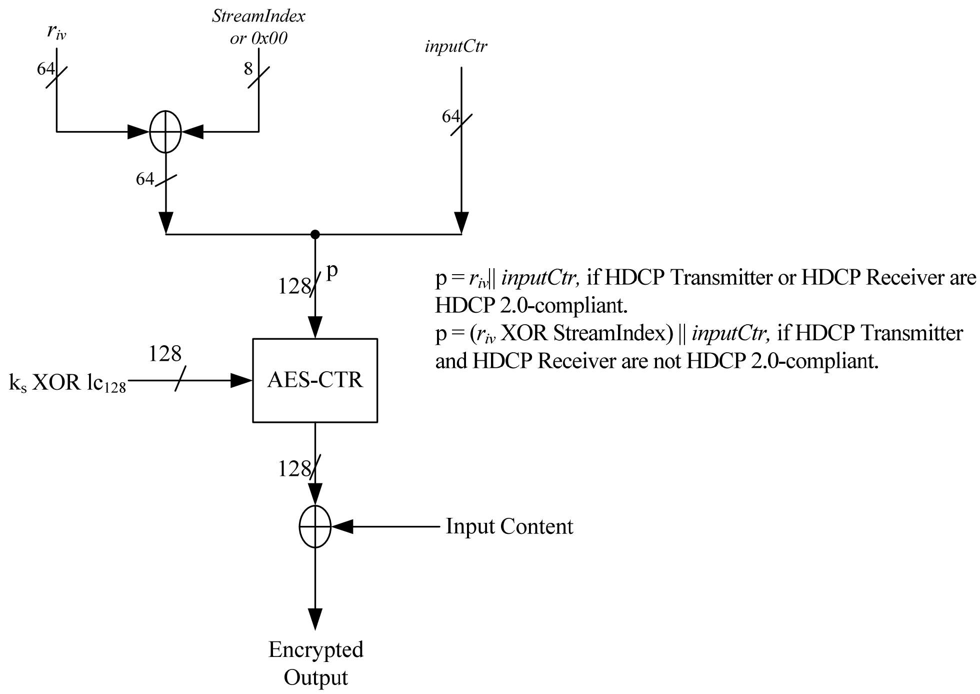

Figure 3.2. HDCP Cipher Structure

k_s is the 128-bit Session Key which is XORed with lc_{128}.

p = r_{iv}|| *inputCtr*, if HDCP Transmitter or HDCP Receiver are HDCP 2.0-compliant. All values are in big-endian order.

p = (r_{iv} XOR StreamIndex) || *inputCtr*, if HDCP Transmitter and HDCP Receiver are not HDCP 2.0-compliant. StreamIndex is XORed with the least-significant byte of r_{iv}. All values are in big-endian order.

To reduce errors inherent in the wireless link and bandwidth overhead, *inputCtr* is derived in both the HDCP Transmitter and HDCP Receiver.

inputCtr is a 64-bit counter. It is derived from the Secure Packet Counter by appending 24 0s to the 40-bit Secure Packet Counter. The Secure Packet Counter is transmitted to the HDCP Receiver in the MAC Header during Link Synchronization. Local values of inputCtr are incremented for each 128-bit block of AES data. The inputCtr value will be re-synchronized at the beginning of each MAC packet.

inputCtr[63:24] = Secure Packet Counter*;* *inputCtr*[23:0] = 0 – incremented locally for each 128-bit AES data block

During any given HDCP Session, there may be multiple simultaneous streams each identified by StreamIndex field. (Refer to [3] for details about StreamIndex. Each HRP MAC packet contains portion of one WirelessHD stream, which is identified by the StreamIndex field in the MAC header of the HRP packet. The StreamIndex value is distinct for each WirelessHD stream.

HDCP Encryption must be applied to HRP Sub-packet payloads; HRP Headers and HRP sub-packet Headers 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* is incremented by one following encryption of 128-bit block of payload data. The value of *inputCtr* must never be reused for a given set of encryption parameters i.e. k_s and r_{iv}.

The 16 Byte encryption block boundary must be aligned with the start of the each sub-packet payload (if present) in the HRP packet. If the last block in an encrypted sub-packet payload 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 sub-packet payload.

Bit ordering is such that the least significant byte of the 16 Byte (128-bit) key stream produced by AES-CTR module is XORed with the first byte in time in the 16 Byte payload data block.

3.4 Uniqueness of k_s and r_{iv}

HDCP Receivers and HDCP Repeaters with multiple inputs may share the same Public Key Certificates and Private Keys across all inputs. The HDCP Transmitter (including downstream side of HDCP Repeater) must negotiate distinct k_m with each directly connected downstream HDCP Device. While r_{tx} used during each HDCP Session is required to be fresh, transmitters with multiple downstream HDCP links must ensure that each link receives a distinct r_{tx} value.

As illustrated in Figure 3.3, HDCP Transmitters, including downstream side of HDCP Repeaters, with multiple downstream HDCP links may share the same k_s and r_{iv} across those links only if HDCP Content from the same HDCP cipher block is transmitted to those links.

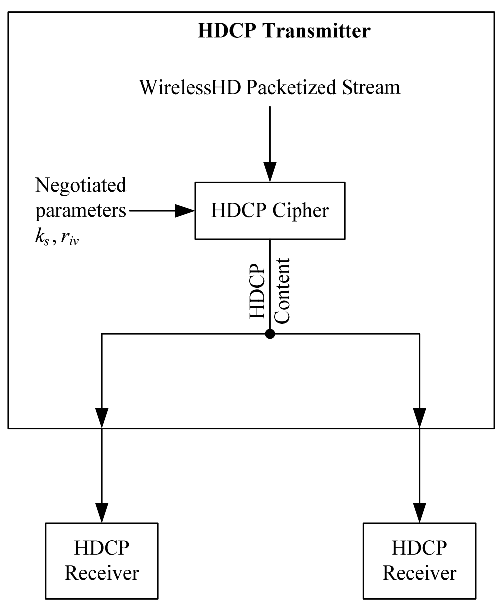

Figure 3.3. k_s and r_{iv} Shared across HDCP Links

As illustrated in Figure 3.4, HDCP Transmitters, including downstream side of HDCP Repeaters, with multiple downstream HDCP links must ensure that each link receives distinct k_s and r_{iv} values if HDCP Content from different HDCP cipher blocks is transmitted to those links.

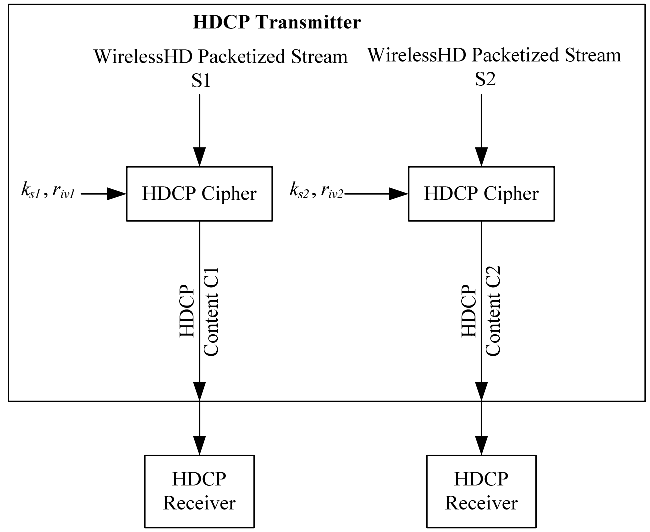

Figure 3.4. Unique k_s and r_{iv} across HDCP Links

4 Authentication Protocol Messages

4.1 Abbreviations

byte – a digital word 8 bits in length.

uint – unsigned integer, an integral number of bytes in length.

bool – a parameter one byte in length. The parameter is ‘true’ if the least-significant bit is non-zero, and false otherwise.

4.2 HDCP Control / Status Stream in WirelessHD

WirelessHD specification provides a bidirectional control mechanism for passing HDCP authentication protocol messages between transmit and receive devices. Please refer to [3] for details.

Each Control/Status message begins with a msg_id field. Valid values of msg_id are shown in Table 4.1..

Message Type	msg_id Value
Null message	1
AKE_Init	2
AKE_Send_Cert	3
AKE_No_Stored_km	4
AKE_Stored_km	5
AKE_Send_rrx	6
AKE_Send_H_prime	7
AKE_Send_Pairing_Info	8
LC_Init	9
RTT_Ready	10
RTT_Challenge	11
RTT_Response	12
SKE_Send_Eks	13
RepeaterAuth_Send_ReceiverID_List	14
RepeaterAuth_Send_Ack	15
RepeaterAuth_Stream_Manage	16
RepeaterAuth_Stream_Ready	17
Receiver_AuthStatus	18
Reserved	19-31

Table 4.1. Values for msg_id

A reliable, bidirectional packet protocol is used to transport messages used for the HDCP authentication protocol from the HDCP Transmitter to the HDCP Receiver, and vice versa.

Prior to the initiation of HDCP session, during the WirelessHD device capability exchange as defined in 10.4.3 of [3], the WirelessHD device supporting the HDCP2.2 protocol shall indicate the revision of HDCP protocol supported by setting the HDCP 2 revision field in CP_SUPPORT format data field of the DEVICE_CAPABILITY_REQUEST or DEVICE_CAPABILITY_RESPONSE messages of WirelessHD MAC protocol to the value 0x2. The HDCP 2 revision field in CP_SUPPORT format data field of the DEVICE_CAPABILITY_REQUEST or DEVICE_CAPABILITY_RESPONSE messages received shall be used to identify the HDCP 2 revision implemented by the device and is assigned to Transmitter_HDCP2_VERSION and Receiver_HDCP2_VERSION when received from a device transmitting or receiving the content stream respectively. Both

Transmitter_HDCP2_VERSION and Receiver_HDCP2_VERSION shall be considered as "byte" when used in computations defined by this specification.

During connection procedure of WirelessHD protocol the HDCP 2 revision field in CP_SUPPORT format data field of CONNECT_REQUEST and CONNECT_RESPONSE messages shall be set to smallest value among Transmitter_HDCP2_VERSION and Receiver_HDCP2_VERSION.

Each message must be carried in an HDCP_2_CONTROL AVC message, as described in [3]. Each AVC message payload commences with a msg_id specifying the message type, followed by parameters specific to each message. RTT_Challenge is sent with the RTT Test MAC command with L[127..0] in the Data field. RTT_Response is sent with the RTT Response MAC command with *L'*[255..128] in the Data field.

Note:

- The use of the Null message and Reserved values for msg_id are not defined in this specification. HDCP Devices must be capable of receiving Null message and messages with reserved msg_id values and must ignore these messages.

4.3 Message Format

4.3.1 AKE_Init (Transmitter to Receiver)

Syntax	No. of Bytes	Identifier
AKE_Init {		
msg_id	1	uint
r_{tx}[63..0]	8	uint
}		

Table 4.2. AKE_Init Payload

4.3.2 AKE_Send_Cert (Receiver to Transmitter)

The HDCP Receiver sets REPEATER to 'true' if it is an HDCP Repeater and 'false' if it is an HDCP Receiver that is not an HDCP Repeater. When REPEATER = 'true', the HDCP Receiver supports downstream connections as permitted by the Digital Content Protection LLC license.

Syntax	No. of Bytes	Identifier
AKE_Send_Cert {		
msg_id	1	uint
REPEATER	1	bool
$cert_{rx}$[4175..0]	522	uint
}		

Table 4.3. AKE_Send_Cert Payload

4.3.3 AKE_No_Stored_km (Transmitter to Receiver)

Syntax	No. of Bytes	Identifier
AKE_No_Stored_km {		
msg_id	1	uint
$E_{kpub}\_k_m$[1023..0]	128	uint
}		

Table4.4. AKE_No_Stored_km Payload

4.3.4 AKE_Stored_km (Transmitter to Receiver)

Syntax	No. of Bytes	Identifier
AKE_Stored_km{		
msg_id	1	uint
$E_{kh}\_k_m$[127..0]	16	uint
m[127..0]	16	uint
}		

Table 4.5. AKE_Stored_km Payload

4.3.5 AKE_Send_rrx (Receiver to Transmitter)

Syntax	No. of Bytes	Identifier
AKE_Send_rrx{		
msg_id	1	uint
r_{rx}[63..0]	8	uint
}		

Table 4.6. AKE_Send_rrx Payload

4.3.6 AKE_Send_H_prime (Receiver to Transmitter)

Syntax	No. of Bytes	Identifier
AK_Send_H_prime{		
msg_id	1	uint
H'[255..0]	32	uint
}		

Table 4.7. AKE_Send_H_prime Payload

4.3.7 AKE_Send_Pairing_Info (Receiver to Transmitter)

Syntax	No. of Bytes	Identifier
AKE_Send_Pairing_Info{		
msg_id	1	uint
$E_{kh}\_k_m$[127..0]	16	uint
}		

Table 4.8. AKE_Send_Pairing_Info Payload

4.3.8 LC_Init (Transmitter to Receiver)

Syntax	No. of Bytes	Identifier
LC_Init {		
msg_id	1	uint
r_n[63..0]	8	uint
}		

Table 4.9. LC_Init Payload

4.3.9 RTT_Ready (Receiver to Transmitter)

Syntax	No. of Bytes	Identifier
RTT_ Ready{		
msg_id	1	uint
}		

Table 4.10. RTT_Ready Payload

4.3.10 RTT_Challenge (Transmitter to Receiver)

Syntax	No. of Bytes	Identifier
RTT_Challenge{		
msg_id	1	uint
L[127..0]	16	uint
}		

Table 4.11. RTT_Challenge Payload

4.3.11 RTT_Response (Receiver to Transmitter)

Syntax	No. of Bytes	Identifier
RTT_Response{		
msg_id	1	uint
L'[255..128]	16	uint
}		

Table 4.12. RTT_Response Payload

4.3.12 SKE_Send_Eks (Transmitter to Receiver)

Syntax	No. of Bytes	Identifier
SKE_Send_Eks{		
msg_id	1	uint
$E_{dkey}\_k_s$[127..0]	16	uint
r_{iv}[63..0]	8	uint
}		

Table 4.13. SKE_Send_Eks Payload

4.3.13 RepeaterAuth_Send_ReceiverID_List (Receiver to Transmitter)

The HDCP Repeater constructs the RepeaterAuth_Send_ReceiverID_List message as given in Table 4.14, if the upstream HDCP Transmitter is an HDCP 2.0-compliant Device, else the HDCP Repeater constructs the message as given in Table 4.15.

Receiver ID list is constructed by appending *Receiver ID*s in big-endian order.

Receiver ID list = $Receiver\ ID_0$ || $Receiver\ ID_1$ || … || $Receiver\ ID_{n-1}$, where n is the DEVICE_COUNT.

If the computed DEVICE_COUNT for an HDCP Repeater exceeds 31, the repeater sets MAX_DEVS_EXCEEDED = 'true'. If the computed DEPTH for an HDCP Repeater exceeds four, the repeater sets MAX_CASCADE_EXCEEDED = 'true'. If topology maximums are not exceeded, MAX_DEVS_EXCEEDED = 'false' and MAX_CASCADE_EXCEEDED = 'false'.

The HDCP Repeater sets HDCP2_0_REPEATER_DOWNSTREAM = 'true' if an HDCP 2.0-compliant repeater is attached to any one of its downstream ports, else it sets HDCP2_0_REPEATER_DOWNSTREAM = 'false'.

The HDCP Repeater sets HDCP1_DEVICE_DOWNSTREAM = 'true' if an HDCP 1.x-compliant Device i.e. an HDCP 1.x-compliant Receiver or an HDCP 1.x-compliant Repeater is attached to any one of its downstream ports, else it sets HDCP1_DEVICE_DOWNSTREAM = 'false'.

Syntax	No. of Bytes	Identifier
RepeaterAuth_Send_ReceiverID_List{		
msg_id	1	uint
MAX_DEVS_EXCEEDED	1	bool
MAX_CASCADE_EXCEEDED	1	bool
if (MAX_DEVS_EXCEEDED != 1 &&		
MAX_CASCADE_EXCEEDED != 1) {		
DEVICE_COUNT	1	uint
DEPTH	1	uint
V'[255..0]	32	uint
for (j=0; j< DEVICE_COUNT; j++) {		
$Receiver\_ID_j$[39..0]	5	uint
}		
}		
}		

Table 4.14. RepeaterAuth_Send_ReceiverID_List Payload

Syntax	No. of Bytes	Identifier
RepeaterAuth_Send_ReceiverID_List{		
msg_id	1	uint
MAX_DEVS_EXCEEDED	1	bool
MAX_CASCADE_EXCEEDED	1	bool
if (MAX_DEVS_EXCEEDED != 1 && MAX_CASCADE_EXCEEDED != 1) {		
DEVICE_COUNT	1	uint
DEPTH	1	uint
HDCP2_0_REPEATER_DOWNSTREAM	1	bool
HDCP1_DEVICE_DOWNSTREAM	1	bool
seq_num_V	3	uint
V′[255..128]	16	uint
for (j=0; j< DEVICE_COUNT; j++) {		
$Receiver\_ID_j$[39..0]	5	uint
}		
}		
}		

Table 4.15. RepeaterAuth_Send_ReceiverID_List Payload

4.3.14 RepeaterAuth_Send_Ack (Transmitter to Receiver)

Syntax	No. of Bytes	Identifier
RepeaterAuth_Send_Ack{		
msg_id	1	uint
V[127..0]	16	uint
}		

Table 4.16. RepeaterAuth_Send_Ack Payload

4.3.15 RepeaterAuth_Stream_Manage (Transmitter to Receiver)

Content Streams are assigned a Type value by the most upstream HDCP Transmitter based on instructions received from the Upstream Content Control Function.

Content Streams may be comprised of audio and video elementary streams. Each elementary stream is assigned a *StreamIndex* value by the HDCP Transmitter. The ContentStreamID, assigned to each elementary stream is associated with its corresponding *StreamIndex* in the RepeaterAuth_Stream_Manage message.

Elementary streams, identified by the *StreamIndex* value are assigned the same Type value that is assigned to the corresponding Content Stream by the HDCP Transmitter. All elementary streams transmitted by the HDCP Transmitter to the HDCP Repeater, after HDCP Encryption, are assigned Type values.

The *StreamIndex* assigned to an elementary stream is followed by its corresponding ContentStreamID value and its assigned Type value 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++) {		
$StreamIndex_j$	1	uint
$\mathrm{ContentStreamID}_j$	2	uint
Type	1	uint
}		
}		

Table 4.17. RepeaterAuth_Stream_Manage Payload

STREAMID_TYPE = $StreamIndex_1$ \|\| $\mathrm{ContentStreamID}_1$ \|\| Type \|\| $StreamIndex_2$ \|\| $\mathrm{ContentStreamID}_2$ \|\|Type \|\|… \|\| $StreamIndex_k$ \|\| $\mathrm{ContentStreamID}_k$ \|\| Type

StreamIndex assigned to an elementary stream is concatenated with its corresponding ContentStreamID value and its assigned Type value. All values are in big-endian order.

k is the number of elementary streams that are being transmitted by the HDCP Transmitter to the attached HDCP Repeater during the HDCP Session.

Parameter	No. of Bytes	Description
ContentStreamID	2	Content Stream Identification. This parameter must be assigned by the Upstream Content Control Function.
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 4.18. RepeaterAuth_Stream_Manage Parameters

4.3.16 RepeaterAuth_Stream_Ready (Receiver to Transmitter)

Syntax	No. of Bytes	Identifier
RepeaterAuth_Stream_Ready{		
msg_id	1	uint
M'[255..0]	32	uint
}		

Table 4.19. RepeaterAuth_Stream_Ready Payload

4.3.17 Receiver_AuthStatus (Receiver to Transmitter)

LENGTH parameter is the size of the Receiver_AuthStatus message in bytes. An HDCP 2.2-compliant Receiver will set the LENGTH parameter equal to four bytes i.e. the combined size of the msg_id, LENGTH and REAUTH_REQ parameters. An HDCP 2.2-compliant transmitter that receives a Receiver_AuthStatus message with the LENGTH parameter greater than four bytes must read the msg_id, LENGTH and REAUTH_REQ parameters and must ignore the remaining parameters.

Syntax	No. of Bytes	Identifier
Receiver_AuthStatus{		
msg_id	1	uint
LENGTH	2	uint
REAUTH_REQ	1	bool
}		

Table 4.20. Receiver_AuthStatus Payload

5 Renewability

It is contemplated that an authorized participant in the authentication protocol may become compromised so as to expose the RSA private keys it possesses for misuse by unauthorized parties. In consideration of this, each HDCP Receiver is issued a unique Receiver ID which is contained in $cert_{rx}$. Through a process defined in the HDCP Adopter's License, the Digital Content Protection LLC may determine that an HDCP Receiver's RSA private key, $kpriv_{rx}$, has been compromised. If so, it places the corresponding Receiver ID on a revocation list that the HDCP Transmitter checks during authentication.

The HDCP Transmitter is required to manage system renewability messages (SRMs) carrying the Receiver ID revocation list. The validity of an SRM is established by verifying the integrity of its signature with the Digital Content Protection LLC public key, which is specified by the Digital Content Protection LLC.

For interoperability with HDCP 1.x, KSVs of revoked HDCP 1.x devices will be included in the HDCP 2 SRM, in addition to the HDCP 1.x SRM. Similarly, *Receiver IDs* of revoked HDCP 2 devices will be included in the HDCP 1.x SRM, in addition to the HDCP 2 SRM.

The SRMs are delivered with content and must be checked when available. The *Receiver IDs* must immediately be checked against the SRM when a new version of the SRM is received. Additionally, devices compliant with HDCP 2.0 and higher must be capable of storing at least 5kB of the SRM in their non-volatile memory. The process by which a device compliant with HDCP 2.0 or higher updates the SRM stored in its non-volatile storage when presented with a newer SRM version is explained in Section 5.2.

5.1 SRM Size and Scalability

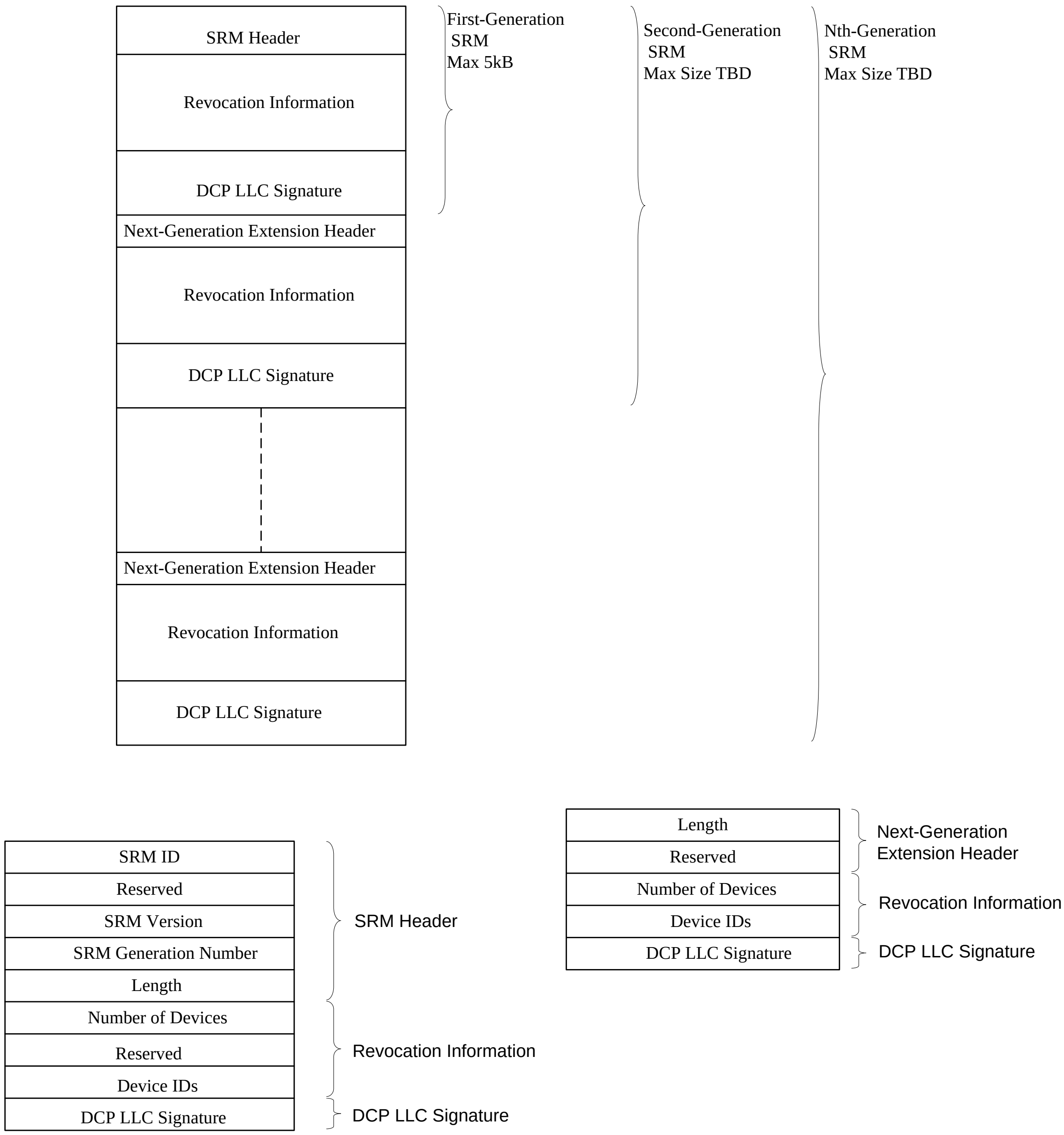

Figure 5.1. SRM Generational Format

As illustrated in

Figure 5.1, the size of the First-Generation HDCP SRM will be limited to a maximum of 5kB. The actual size of the First-Generation SRM is 5116 bytes. For scalability of the SRM, the SRM format supports next-generation extensions. By supporting generations of SRMs, an HDCP SRM can, if required in future, grow beyond the 5kB limit to accommodate more *Receiver IDs*. Next-generation extensions are appended to the current-generation SRM in order to ensure backward compatibility with devices that support only previous-generation SRMs.

Table 5.1. gives the format of the HDCP 2 SRM. All values are stored in big endian format.

Name	Size (bits)	Function
SRM ID	4	A value of 0x9 signifies that the message is for HDCP2. All other values are reserved. SRMs with values other than 0x9 must be ignored.
HDCP2 Indicator	4	A value of 0x1 signifies that the message is for HDCP2
Reserved	8	Reserved for future definition. Must be 0x00
SRM Version	16	Sequentially increasing unique SRM numbers. Higher numbered SRMs are more recent
SRM Generation Number	8	Indicates the generation of the SRM. The generation number starts at 1 and increases sequentially
Length	24	Length in bytes and includes the combined size of this field (three bytes) and all following fields contained in the first-generation SRM i.e. size of this field, Number of Devices field, Reserved (22 bits) field, Device IDs field and Digital Content Protection LLC signature field (384 bytes) in the first-generation SRM
Number of Devices	10	Specifies the number (N1) of *Receiver IDs* / KSVs contained in the first-generation SRM
Reserved	22	Reserved for future definition. All bits set to 0
Device IDs	40 * N1 Max size for this field is 37760 (4720 bytes)	40-bit *Receiver IDs* / KSVs
DCP LLC Signature	3072	A cryptographic signature calculated over all preceding fields of the SRM. RSASSA-PKCS1-v1_5 is the signature scheme used as defined by PKCS #1 V2.1: RSA Cryptography Standard. SHA-256 is the underlying hash function

Table 5.1. System Renewability Message Format

Each subsequent next-generation extensions to the first-generation SRM will have the following fields.

Name	Size (bits)	Function
Length	16	Length in bytes and includes the combined size of this field (two bytes) and all following fields contained in this next-generation extension i.e. size of this field, Number of Devices field, Reserved (6 bits) field, Device IDs field and Digital Content Protection LLC signature field (384 bytes) in this next-generation SRM
Reserved	6	Reserved for future definition. All bits set to 0
Number of Devices	10	Specifies the number (N2) of *Receiver IDs* / KSVs contained in this next generation extension
Device IDs	40 * N2	40-bit *Receiver IDs* / KSVs
DCP LLC Signature	3072	A cryptographic signature calculated over all preceding fields of the SRM. RSASSA-PKCS1-v1_5 is the signature scheme used as defined by PKCS #1 V2.1: RSA Cryptography Standard. SHA-256 is the underlying hash function

Table 5.2. Next-generation extension format

5.2 Updating SRMs

The stored HDCP SRM must be updated when a newer version of the SRM is delivered with the content. The procedure for updating an SRM is as follows:

1. Verify that the version number of the new SRM is greater than the version number of the SRM currently stored in the device's non-volatile storage

2. If the version number of the new SRM is greater (implying that it is a more recent version), verify the signature on the new SRM

On successful signature verification, replace the current SRM in the device's non-volatile storage with the new SRM. If, for instance, the device supports only second-generation SRMs and the new SRM is a third-generation SRM, the device is not required to store the third-generation extension. Devices compliant with HDCP 2.0 or higher must be capable of storing at least 5kB (actual size is 5116 bytes) of the SRM (First-Generation SRM).

Appendix A. Core Functions and Confidentiality and Integrity of Values

Table A.1 identifies the requirements of confidentiality and integrity for values within the protocol. A *confidential* value must never be revealed. The *integrity* of many values in the system is protected by fail-safe mechanisms of the protocol. Values that are not protected in this manner require active measures beyond the protocol to ensure integrity. Such values are noted in the table as requiring integrity. Core Functions must be implemented in Hardware. The values used by Core Functions, along with the corresponding Core Functions by which they are used, are identified in the table.

Value	**Confidentiality Required[±]?**	**Integrity Required[±]?**	**Value used by Core Functions?**	**Core Function**
lc_{128}	Yes	Yes	Yes	HDCP Encryption and Decryption
$kpub_{dcp}$	No	Yes	No	N/A
$cert_{rx}$	No	No	No	N/A
$kpub_{rx}$	No	Yes	No	N/A
Receiver ID	No	Yes	No	N/A
$kpriv_{rx}$	Yes	Yes	Yes	Handling of Device Secret Key, during AKE, in plaintext form
r_{tx}	No	Yes[*]	Yes	
r_{iv}	No	Yes[*]	Yes	N/A
REPEATER	No	Yes	No	N/A
r_{rx}	No	Yes[**]	Yes	N/A
k_m	Yes	Yes[*]	Yes	Handling of Master Key, during AKE (including Pairing) and Key Derivation, in plaintext form
k_d	Yes	Yes[*]	No	N/A
$dkey_0$, $dkey_1$	Yes	Yes[*]	No	N/A

[±] According to the robustness rules in the HDCP Adopter's License

[*] Only within the transmitter

[**] Only within the receiver

$dkey_2$	Yes	Yes*	Yes	Handling of information or materials during Key Derivation and SKE, including but not limited to cryptographic keys used to encrypt or decrypt HDCP Core Keys (k_s), from which HDCP Core Keys could reasonably be derived
ctr	No	Yes*	Yes	N/A
H	Yes	Yes	No	N/A
H'	No	No	No	N/A
m	No	No	Yes	N/A
k_h	Yes	Yes	Yes	Handling of information or materials during Pairing, including but not limited to cryptographic keys used to encrypt or decrypt HDCP Core Keys (*km*), from which HDCP Core Keys could reasonably be derived
r_n	No	Yes*	Yes	N/A
L	Yes	Yes	No	N/A
L'	No	No	No	N/A
k_s	Yes	Yes*	Yes	Handling of Session Key, during SKE and HDCP Encryption/Decryption, in plaintext form
V	Yes	Yes	No	N/A
V'	No	No	No	N/A
Receiver ID list	No	Yes	No	N/A
DEPTH	No	Yes	No	N/A
DEVICE_COUNT	No	Yes	No	N/A
MAX_DEVS_EXCEEDED	No	Yes	No	N/A
MAX_CASCADE_EXCEEDED	No	Yes	No	N/A

inputCtr	No	Yes*	Yes	HDCP Encryption and Decryption
p	No	Yes*	Yes	HDCP Encryption and Decryption

Table A.1. Core Functions and Confidentiality and Integrity of Values

Appendix B. DCP LLC Public Key

Table B.1 gives the production DCP LLC public key.

Parameter	Value (hexadecimal)
Modulus n	B0E9 AA45 F129 BA0A 1CBE 1757 28EB 2B4E 8FD0 C06A AD79 980F 8D43 8D47 04B8 2BF4 1521 5619 0140 013B D091 9062 9E89 C227 8ECF B6DB CE3F 7210 5093 8C23 2983 7B80 64A7 59E8 6167 4CBC D858 B8F1 D4F8 2C37 9816 260E 4EF9 4EEE 24DE CCD1 4B4B C506 7AFB 4965 E6C0 0083 481E 8E42 2A53 A0F5 3729 2B5A F973 C59A A1B5 B574 7C06 DC7B 7CDC 6C6E 826B 4988 D41B 25E0 EED1 79BD 3985 FA4F 25EC 7019 23C1 B9A6 D97E 3EDA 48A9 58E3 1814 1E9F 307F 4CA8 AE53 2266 2BBE 24CB 4766 FC83 CF5C 2D1E 3AAB AB06 BE05 AA1A 9B2D B7A6 54F3 632B 97BF 93BE C1AF 2139 490C E931 90CC C2BB 3C02 C4E2 BDBD 2F84 639B D2DD 783E 90C6 C5AC 1677 2E69 6C77 FDED 8A4D 6A8C A3A9 256C 21FD B294 0C84 AA07 2926 46F7 9B3A 1987 E09F EB30 A8F5 64EB 07F1 E9DB F9AF 2C8B 697E 2E67 393F F3A6 E5CD DA24 9BA2 7872 F0A2 27C3 E025 B4A1 046A 5980 27B5 DAB4 B453 973B 2899 ACF4 9627 0F7F 300C 4AAF CB9E D871 2824 3EBC 3515 BE13 EBAF 4301 BD61 2454 349F 733E B510 9FC9 FC80 E84D E332 968F 8810 2325 F3D3 3E6E 6DBB DC29 66EB
Public Exponent e	03

Table B.1. DCP LLC Public Key

Appendix C. Bibliography (Informative)

These documents are not normatively referenced in this specification, but may provide useful supplementary information.

WirelessHD specification. See www.wirelessHD.org