ATSC Candidate Standard: ATSC 3.0 Security and Service Protection

Doc. S36-016r13
28 October-2016

Advanced Television Systems Committee
1776 K Street, N.W.
Washington, D.C. 20006
202-872-9160

The Advanced Television Systems Committee, Inc., is an international, non-profit organization developing voluntary standards for digital television. The ATSC member organizations represent the broadcast, broadcast equipment, motion picture, consumer electronics, computer, cable, satellite, and semiconductor industries.

Specifically, ATSC is working to coordinate television standards among different communications media focusing on digital television, interactive systems, and broadband multimedia communications. ATSC is also developing digital television implementation strategies and presenting educational seminars on the ATSC standards.

ATSC was formed in 1982 by the member organizations of the Joint Committee on InterSociety Coordination (JCIC): the Electronic Industries Association (EIA), the Institute of Electrical and Electronic Engineers (IEEE), the National Association of Broadcasters (NAB), the National Cable Telecommunications Association (NCTA), and the Society of Motion Picture and Television Engineers (SMPTE). Currently, there are approximately 150 members representing the broadcast, broadcast equipment, motion picture, consumer electronics, computer, cable, satellite, and semiconductor industries.

ATSC Digital TV Standards include digital high definition television (HDTV), standard definition television (SDTV), data broadcasting, multichannel surround-sound audio, and satellite direct-to-home broadcasting.

Note: The user's attention is called to the possibility that compliance with this standard may require use of an invention covered by patent rights. By publication of this standard, no position is taken with respect to the validity of this claim or of any patent rights in connection therewith. One or more patent holders have, however, filed a statement regarding the terms on which such patent holder(s) may be willing to grant a license under these rights to individuals or entities desiring to obtain such a license. Details may be obtained from the ATSC Secretary and the patent holder.

This specification is being put forth as a Candidate Standard by the TG3/S36 Specialist Group. This document is a revision of the Working Draft (S36-016r12) dated 15 August 2016. All ATSC members and non-members are encouraged to review and implement this specification and return comments to cs-editor@atsc.org. ATSC Members can also send comments directly to the TG3/S36 Specialist Group. This specification is expected to progress to Proposed Standard after its Candidate Standard period.

> Note: Cyan highlight and text in *blue italics* identify areas that are under development in the committee. Feedback and comments on these points from implementers is encouraged.

Revision History

Version	Date
Candidate Standard approved	28 October 2016
Revision 1 approved	Insert date here

Table of Contents

1. SCOPE 1
1.1 Organization 1
2. REFERENCES 1
2.1 Normative References 1
2.2 Informative References 3
3. DEFINITION OF TERMS 3
3.1 Compliance Notation 3
3.2 Treatment of Syntactic Elements 3
3.2.1 Reserved Elements 3
3.3 Acronyms and Abbreviation 4
3.4 Terms 4
3.5 Extensibility 4
3.5.1 Backward-compatible Extensibility Mechanisms 4
3.5.2 Non-backward-compatible Extensibility Mechanisms 4
3.5.3 Extensions with Unknown Compatibility 4
3.5.4 Descriptor Processing Considerations 4
4. SYSTEM OVERVIEW 4
4.1 Features 5
4.2 System Architecture 5
4.3 Central Concepts 5
4.4 Security of Surrounding Systems 5
5. SPECIFICATION 5
5.1 Transport Protection 5
5.1.1 Internet Streaming Transport Security 5
5.1.2 Broadcast Transport Security 8
5.1.3 Server Authentication 8
5.1.4 Media Protection 8
5.1.5 Certificates and Certificate Management 8
5.2 ATSC 3.0 Application Code Signing 8
5.3 Certificates and Certificate Management 9
5.3.1 Certificate Profiles 9
5.4 ATSC 3.0 Client Certificate Storage 10
5.5 Certificate Revocation and Status Information 10
5.5.1 Certificate Revocation and Status Information for TLS Server Certificates 11
5.5.2 Certificate Revocation and Status Information for ATSC 3.0 Application Signing Certificates 11
5.6 [Companion Device Pairing] 11
5.7 Content Protection 12
5.7.1 Types of Content Protection 12
5.7.2 Protecting Content from Unauthorized Viewing 12
5.7.3 Protecting Content from Unauthorized Copying and Redistribution 13
5.7.4 Common Encryption 13
5.7.5 Encrypted Media Extensions 14
5.7.6 Extensions to MMT Signaling for CENC Support 15

5.7.7 ROUTE/DASH Support for CENC and EME 20
5.7.8 Required Mechanisms to Support Forensic or Copyright Watermarking 20
5.7.9 Required Mechanisms to Support OTT Content Security 20
5.7.10 Required Mechanisms to Support Home Network Redistribution or Prevention of Home Network Redistribution 20
5.8 Backend Business Systems 20
ANNEX A: ROUTE/DASH CLIENT PROCESSING FOR COMMON ENCRYPTION (CENC) AND ENCRYPTED MEDIA EXTENSIONS (EME) (INFORMATIVE) 21
A.1 Introduction 21
A.1.1 Basic CENC Operation in ROUTE/DASH 21
A.1.2 Solution Framework for DRM and CENC 23
A.1.3 MPD Support for Encryption and DRM Signaling 25

Index of Figures and Tables

Figure 5.1 Storage of CENC related information. 14
Figure 5.2 Encrypted Media Extensions workflow. 15
Figure 5.3 Broadcast license retrieval. 19
Figure A.1 DRM license and key acquisition before start of program in ROUTE/DASH. 22
Figure A.2 DRM license and key acquisition during program delivery in ROUTE/DASH. 23
Figure A.3 CENC-related metadata structure for protection of VoD content by a single key. ... 24
Figure A.4 CENC-related metadata structure for protection of live streaming content. 25

Table 5.1 SI Descriptor 16
Table 5.2 LS Signaling Message 17
Table 5.3 LR Signaling Message 18

ATSC Candidate Standard:
ATSC 3.0 Security and Service Protection

1. SCOPE

This standard specifies the mechanisms for security and service protections in ATSC 3.0 systems.

1.1 Organization

This document is organized as follows:

- Section 1 – Outlines the scope of this document and provides a general introduction.
- Section 2 – Lists references and applicable documents.
- Section 3 – Provides a definition of terms, acronyms, and abbreviations for this document.
- Section 4 – System overview
- Section 5 – Specification
- Annex A – ROUTE/DASH Client Processing for CENC and EME

2. REFERENCES

All referenced documents are subject to revision. Users of this Standard are cautioned that newer editions might or might not be compatible.

2.1 Normative References

The following documents, in whole or in part, as referenced in this document, contain specific provisions that are to be followed strictly in order to implement a provision of this Standard.

[1] IEEE: "Use of the International Systems of Units (SI): The Modern Metric System," Doc. SI 10, Institute of Electrical and Electronics Engineers, New York, N.Y.

[2] ISO/IEC: ISO/IEC 23001-7 Second edition 2015-04-01, "Information technology — MPEG systems technologies — Part 17: Common encryption in ISO base media file format files."

[3] ISO/IEC: ISO/IEC 14496-12 Fourth edition 2012-07-15 Corrected version 2012-09-15, "Information technology — Coding of audio-visual objects — Part 12: ISO base media file format."

[4] W3C: "Encrypted Media Extensions," W3C Editor's Draft 03 September 2015, World Wide Web Consortium, https://w3c.github.io/encrypted-media/.

[5] ATSC: "ATSC Mobile DTV Standard, Part 6 – Service Protection," A/153 Part 6:2011, Advanced Television Systems Committee, Washington, D.C., 23 May 2011.

[6] W3C: "Media Source Extensions", W3C Editor's Draft 14 July 2015, World Wide Web Consortium, https://w3c.github.io/media-source/.

[7] ISO/IEC: "ISO/IEC 23009–1:2014, Information technology — Dynamic adaptive streaming over HTTP (DASH) — Part 1: Media presentation description and segment formats," International Organization for Standardization, Geneva, 2nd Edition, 15 May 2014.

[8] DASH: "Guidelines for Implementation: DASH-IF Interoperability Points", Version 3.3, DASH Industry Forum, Beaverton, OR, 12 June 2016.

[9] DASH: "Guidelines for Implementation: DASH-IF Interoperability Points for ATSC 3.0", Version 0.9, DASH Industry Forum, Beaverton, OR, 3 August 2016.

[10] IETF: "RFC 3279, Algorithms and Identifiers for the Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile," L. Bassham, W. Polk, R. Housley, Internet Engineering Task Force, Fremont, CA, April 2002.

[11] IETF: "RFC 4055, Additional Algorithms and Identifiers for RSA Cryptography for use in the Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile," J. Schaad, B. Kaliski, R. Housley, Internet Engineering Task Force, Fremont, CA, June 2005.

[12] IETF: "RFC 5019, The Lightweight Online Certificate Status Protocol (OCSP) Profile for High-Volume Environments," A. Deacon, R. Hurst, Internet Engineering Task Force, Fremont, CA, September 2007.

[13] IETF: "RFC 5077, Transport Layer Security (TLS) Session Resumption without Server-Side State," J. Salowey, H. Zhou, P. Eronen, H. Tschofenig, Internet Engineering Task Force, Fremont, CA, January 2008.

[14] IETF: "RFC 5246, The Transport Layer Security (TLS) Protocol Version 1.2," T. Dierks, E. Rescorla, Internet Engineering Task Force, Fremont, CA, August 2008.

[15] IETF: "RFC 5280, Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile," D. Cooper, S. Santesson, S. Farrell, S. Boeyen, R. Housley, W. Polk, Internet Engineering Task Force, Fremont, CA, May 2008.

[16] IETF: "RFC 5289, TLS Elliptic Curve Cipher Suites with SHA-256/384 and AES Galois Counter Mode (GCM)," E. Rescorla, Internet Engineering Task Force, Fremont, CA, August 2008.

[17] IETF: "RFC 5480, Elliptic Curve Cryptography Subject Public Key Information," S. Turner, D. Brown, K. Yiu, R. Housley, T. Polk, Internet Engineering Task Force, Fremont, CA, March 2009.

[18] IETF: "RFC 5746, Transport Layer Security (TLS) Renegotiation Indication Extension," E. Rescorla, M. Ray, S. Dispensa, N. Oskov, Internet Engineering Task Force, Fremont, CA, February 2010.

[19] IETF: "RFC 5758, Internet X.509 Public Key Infrastructure: Additional Algorithms and Identifiers for DSA and ECDSA," Q. Dang, S. Santesson, K. Moriarty, D. Brown, T. Polk, Internet Engineering Task Force, Fremont, CA, January 2010.

[20] IETF: "RFC 5869, HMAC-based Extract-and-Expand Key Derivation Function (HKDF)," H. Krawczyk, P. Eronen, Internet Engineering Task Force, Fremont, CA, May 2010.

[21] IETF: "RFC 6066, Transport Layer Security (TLS) Extensions: Extension Definitions," D. Eastlake 3rd, Internet Engineering Task Force, Fremont, CA, January 2011.

[22] IETF: "RFC 6960, X.509 Internet Public Key Infrastructure Online Certificate Status Protocol – OCSP," S. Santesson, M. Myers, R. Ankney, A. Malpani, S. Galperin, C. Adams, Internet Engineering Task Force, Fremont, CA, June 2013.

[23] IETF: "TLS 1.3, The Transport Layer Security (TLS) Protocol Version 1.3," draft-ietf-tls-tls13-14, Internet Engineering Task Force, Fremont, CA,

[24] IETF: "RFC 7539, ChaCha20 and Poly1305 for IETF Protocols," Y. Nir, A. Langley, Internet Engineering Task Force, Fremont, CA, May 2015.

[25] IETF: "CHACHA20_POLY1305, ChaCha20-Poly1305 Cipher Suites for Transport Layer Security (TLS)," draft-ietf-tls-chacha20-poly1305-04, Internet Engineering Task Force, Fremont, CA,

[26] IETF: "ECDHE PSK, ECDHE_PSK with AES-GCM and AES-CCM Cipher Suites for Transport Layer Security (TLS)," draft-mattson-tls-ecdhe-psk-aead-03, Internet Engineering Task Force, Fremont, CA,

[27] W3C: "Widgets-DigSig, XML Digital Signature for Widgets," World Wide Web Consortium, March 2012.

2.2 Informative References

The following documents contain information that may be helpful in applying this Standard.

[28] CTA: "CEA 2053. Receiver Specifications for ATSC 2.0 Security," ANSI/CTA-2053, Consumer Technology Association, Arlington, VA, August 2015.

3. DEFINITION OF TERMS

With respect to definition of terms, abbreviations, and units, the practice of the Institute of Electrical and Electronics Engineers (IEEE) as outlined in the Institute's published standards [1] shall be used. Where an abbreviation is not covered by IEEE practice or industry practice differs from IEEE practice, the abbreviation in question will be described in Section 3.3 of this document.

3.1 Compliance Notation

This section defines compliance terms for use by this document:

shall – This word indicates specific provisions that are to be followed strictly (no deviation is permitted).

shall not – This phrase indicates specific provisions that are absolutely prohibited.

should – This word indicates that a certain course of action is preferred but not necessarily required.

should not – This phrase means a certain possibility or course of action is undesirable but not prohibited.

3.2 Treatment of Syntactic Elements

This document contains symbolic references to syntactic elements used in the audio, video, and transport coding subsystems. These references are typographically distinguished by the use of a different font (e.g., restricted), may contain the underscore character (e.g., sequence_end_code) and may consist of character strings that are not English words (e.g., dynrng).

3.2.1 Reserved Elements

One or more reserved bits, symbols, fields, or ranges of values (i.e., elements) may be present in this document. These are used primarily to enable adding new values to a syntactical structure without altering its syntax or causing a problem with backwards compatibility, but they also can be used for other reasons.

The ATSC default value for reserved bits is '1.' There is no default value for other reserved elements. Use of reserved elements except as defined in ATSC Standards or by an industry standards setting body is not permitted. See individual element semantics for mandatory settings and any additional use constraints. As currently-reserved elements may be assigned values and meanings in future versions of this Standard, receiving devices built to this version are expected to ignore all values appearing in currently-reserved elements to avoid possible future failure to function as intended.

3.3 Acronyms and Abbreviation

The following acronyms and abbreviations are used within this document.

AES – Advanced Encryption Standard
ATSC Advanced Television Systems Committee
CA – Certificate Authority
CEA – Consumer Electronics Association
DNS – Domain Name System
DTCP – Digital Transmission Content Protection
ECDHE – Elliptic Curve Diffie-Hellman Ephemeral key exchange
ECDSA – Elliptic Curve Digital Signature Algorithm
GCM – Galios Counter Method
IP – Internet Protocol
OCSP – Online Certificate Status Protocol
RSA – A method for obtaining digital signatures and public-key cryptosystems (originally proposed by Rivest, Shamir, and Adelman).
SECP – Standard for Efficient Cryptography Elliptic Curve Domain Parameters
SHA – Secure Hash Algorithm
TLS – Transport Layer Security

3.4 Terms

The following terms are used within this document.

ATSC 3.0 Server – any IP-connected device that provides content or other service to an ATSC 3.0 client, and that complies with the normative requirements of this standard.
reserved – Set aside for future use by a Standard.

3.5 Extensibility

3.5.1 Backward-compatible Extensibility Mechanisms

3.5.2 Non-backward-compatible Extensibility Mechanisms

3.5.3 Extensions with Unknown Compatibility

3.5.4 Descriptor Processing Considerations

4. SYSTEM OVERVIEW

<To be added>

4.1 Features

4.2 System Architecture

4.3 Central Concepts

4.4 Security of Surrounding Systems

5. SPECIFICATION

5.1 Transport Protection

Transport Protection provides protection against spoofing or hijacking the delivery of the data. This may include protection of content that is not separately encrypted. Encryption of content in transit will be described in this section.

5.1.1 Internet Streaming Transport Security

5.1.1.1 TLS – Transport Layer Security

ATSC 3.0 clients are expected to implement both TLS 1.3 [23] and TLS 1.2 (RFC 5246 [14]) for Secure Connections over the Interaction Channel. An ATSC 3.0 client is expected to request a connection using TLS 1.3 (ProtocolVersion { 0x03, 0x04 }), but is also expected to accept a server's request to downgrade the connection to TLS 1.2 (ProtocolVersion { 0x03, 0x03 }) in the manner specified in TLS 1.3 Appendix C.

An ATSC 3.0 server, when negotiating a Secure Connection for use with ATSC 3.0 Interaction Channel protocols should comply with TLS 1.3. An ATSC 3.0 server that does not support TLS 1.3 shall respond with a "Server Hello" message specifying a ProtocolVersion { 0x03, 0x03 } (indicating TLS 1.2). The server shall refuse Secure Connection negotiations with clients that do not support a ProtocolVersion equal to or greater than { 0x03, 0x03 } and shall send a protocol_version alert message to the client as described in TLS 1.3 Appendix C (TLS 1.2 Appendix E).

5.1.1.1.1 TLS Cipher Suite Negotiation

ATSC 3.0 servers shall negotiate Secure Connections using one or more of the following Cipher Suites (as specified in RFC 5289 [16]):

```
TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256
TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384
TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256
TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384
```

or one or more of the following Cipher Suites (as specified in RFC 7539 [24] and in CHACHA20_POLY1305 [25]) where these cipher suites are requested by the client:

```
TLS_ECDHE_ECDSA_WITH_CHACHA20_POLY1305_SHA256
TLS_RSA_ECDSA_WITH_CHACHA20_POLY1305_SHA256
```

and in the case of a TLS 1.2 negotiated Secure Connection the additional Cipher Suite:

```
TLS_RSA_WITH_AES_128_CBC_SHA
```

(as specified in RFC 5246 [14]) may be negotiated for, however, the server shall only choose this Cipher Suite as the least preferred of the client's cipher suites (irrespective of the order supplied by the client).

An ATSC 3.0 server that supports TLS 1.3 and has established a TLS 1.3 based Secure Connection shall only negotiate a session resumption request for that TLS 1.3 session from a client that uses one or more of the following Cipher Suites (as specified in ECDHE-PSK [26]):

```
TLS_ECDHE_PSK_WITH_AES_128_GCM_SHA256
TLS_ECDHE_PSK_WITH_AES_256_GCM_SHA384
```

or the following Cipher Suite (as specified in CHACHA20_POLY1305 [25]) where this cipher suite is requested by the client:

```
TLS_ECDHE_PSK_WITH_CHACHA20_POLY1305_SHA256
```

5.1.1.1.2 Elliptic Curve Parameters

An ATSC 3.0 server shall support the following elliptic curve groups: **secp256r1, secp384r1,** and **secp521r1.** An ATSC 3.0 server shall support the **uncompressed** point format.

These elliptic curve groups and point formats are the only ones available in both TLS 1.2 and TLS 1.3. Servers shall decline to establish a connection that does not request one or more of these curve groups or point formats even when negotiating a TLS 1.2 connection.

The client is expected to only negotiate elliptic curve groups and point formats that are required to be supported by an ATSC 3.0 server (even when negotiating a TLS 1.2 connection).

5.1.1.1.3 Signature Algorithms

When a client negotiates a connection it is expected to either to include one or more of the following signature and hash algorithm combinations in its Signature Algorithm Extension:

- **rsa** or **ecdsa** Signature Algorithm with any of **sha256, sha384** or **sha512** Hash Algorithm. An ATSC 3.0 server shall support each of these Signature Algorithm and Hash Algorithm combinations for both TLS 1.3 and TLS 1.2 connections.
- **rsapss** Signature Algorithm with any of **sha256, sha384** or **sha512** Hash Algorithm. An ATSC 3.0 server shall support each of these Signature Algorithm and Hash Algorithm combinations for TLS 1.3 connections.

An ATSC 3.0 client that is negotiating (or renegotiating) a TLS 1.2 connection may omit the Signature Algorithm Extension. When a client does not include a Signature Algorithm Extension, the ATSC 3.0 server shall assume that the client supports the default set of **rsa** and **ecdsa** Signature Algorithms with the **sha1** Hash Algorithm.

5.1.1.1.4 Server Certificate Selection

An ATSC 3.0 server shall only supply certificates with signatures using one of the supported signature and hash algorithm combinations (see Section 5.1.1.1.3 above) that is negotiated by the client (even in the case that the client attempts to negotiate other algorithms) and shall not establish a Secure Connection with certificates that use other algorithms.

When a client requests a connection over TLS 1.2 or TLS 1.3 it can include a Server Name Indication extension as specified in RFC 6066 [21]that contains the fully qualified DNS host name of the server. The ATSC 3.0 server shall use the Server Name Indication where provided by the client to assist in the selection of a suitable server certificate to return to the client in the TLS handshake.

When a client requests a connection over TLS 1.2 or TLS 1.3 it can include a Trusted CA Indication extension as specified in RFC 6066 [21] to provide a list of the trusted root certificates that it holds in its secure store. Such list of trusted root certificates can be either the full list of certificates described at http://www.hbbtv.org/spec/certificates.html, or some subset thereof. The ATSC 3.0 server shall use the Trusted CA Indication extension to assist in the selection of a suitable certificate chain to return to the client in the TLS handshake.

In the case that an ATSC 3.0 server is unable to select a certificate chain that matches the client criteria in either the Server Name Indication extension or the Trusted CA Indication extension, the ATSC 3.0 server shall not establish the connection.

5.1.1.1.5 TLS Certificate Status Request and Response

The client is expected to include the Certificate Status Request extension as specified in RFC 6066 [21] Section 8. The Certificate Status Request extension includes a list of OCSP Responder Identifiers each encoded as a SHA-1 hash of the trusted OCSP responder public key as defined in RFC 6960 [22]. An ATSC 3.0 server shall only supply to the client the OCSP responses that it has received from OCSP responders with responder public keys that are trusted by the client and which are signed using signature algorithms supported by the client. If an ATSC 3.0 server is unable to obtain an OCSP Response for a certificate that it supplies from an OCSP Responder that is identified by the client as a trusted responder, the ATSC 3.0 server shall not establish the connection.

The ATSC 3.0 server shall forward the most recent OCSP Response (see Section 5.5.1 below) for the certificates it uses to establish a connection to the ATSC 3.0 client. The format of the OCSP Response provided by the responder should be limited to the mandatory elements defined in RFC 5019 [12] and no optional elements should be included in the response. When a server is establishing a connection over TLS 1.2, the server shall include the OCSP Response in its Certificate Status handshake message (immediately after its Certificate handshake message) as defined in RFC 6960 [22]. When a server is establishing a connection over TLS 1.3, the server shall include the OCSP Response in the Encrypted Extensions of the Server Hello handshake message.

The ATSC 3.0 client is expected to verify the Certificate Status message provided by the server as specified in RFC 6066 [21] Section 8. A client uses the OCSP Response data that it receives to verify that the certificates that authenticate server connections are valid at the time the connection is established. See CEA 2053 [28].

5.1.1.1.6 TLS Session Resumption

An ATSC 3.0 server that has a newly established TLS 1.3 connection may provide a New Session Ticket message once it has received the client's Finished handshake message. The New Session Ticket message shall include the allow_dhe_resumption flag and shall not include either the allow_early_data flag or the allow_psk_resumption flag. When a client supplies the information from this session ticket in a subsequent Client Hello message to resume the TLS session, the ATSC 3.0 server shall verify that the session ticket is still valid and that the client has selected one of the Cipher Suites available for session resumption (see Section 5.1.1.1.1 above) before resuming the session. The Client Hello message shall not include the early_data extension and the ATSC 3.0 server shall not respond to a Client Hello message that contains early_data thus requiring the client to issue a session resumption Client Hello message without any early data.

An ATSC 3.0 server that has established a TLS 1.2 connection session may support the Session Ticket Extension (RFC 5077 [13]) to allow later resumption of that session. If the ATSC 3.0 server

does not support this extension, then it shall not send an empty Session Ticket Extension to the client that has requested session ticket information.

5.1.1.1.7 TLS Connection Renegotiation

TLS 1.3 does not support connection renegotiation.

An ATSC 3.0 client that is processing a TLS 1.2 handshake is expected to support the Renegotiation Indication extension (RFC 5746 [18]) but is not expected to send a Client Hello handshake message that includes any data in this extension. An ATSC 3.0 server that is processing a TLS 1.2 handshake shall include an empty Renegotiation Indication extension as required by RFC 5746 [18] in the Server Hello message to indicate that it does not support renegotiation. An ATSC 3.0 server that is processing a TLS 1.2 handshake shall not send a Hello Request message to the client to instigate renegotiation of connection parameters.

Teal – Drafting Notes - TBD

5.1.1.2 Source Stream Validation

5.1.1.2.1 DNSSEC – Domain Name System Security Extensions

5.1.1.2.2 DANE- DNS-based Authentication of Named Entities

5.1.2 Broadcast Transport Security

5.1.3 Server Authentication

5.1.3.1 Server Authentication in Internet Streaming Transport

5.1.3.2 Server Authentication in Broadcast Transport

5.1.4 Media Protection

How is this different from “Content Protection”, covered below?

5.1.5 Certificates and Certificate Management

5.1.5.1 Certificates Supported

5.1.5.2 Initial Certificate Synchronization

Requirements for any certificates that must exist on initial power up of devices to “seed” the key distribution and update process, and possible resynchronization via an “Out of Band” process

5.1.5.2.1 Startup/Factory Certificate Requirements

5.1.5.2.2 Out-of-Band Certificate Resynchronization

5.1.5.3 Certificate Management in Internet Streaming or “online” environments

5.1.5.4 Certificate Management in Broadcast (unidirectional dataflow) Environments

5.1.5.4.1 Limited Security May Prevent Selected Service Models

5.2 ATSC 3.0 Application Code Signing

Executable or interpretable code may be cryptographically signed. If signatures are used, they shall be signed as follows:

Signed applications shall be signed as specified in Widgets-DigSig [27] with the addition of a SigningTime Signature Property for each signature as defined below. RSA keys used for signing shall be 2048 bits in length. The SHA-256 digest method shall be used.

> Note: The digital signatures specification allows for zero or one author signatures, and zero, one or more distributor signatures.

The SigningTime signature property shall contain the time at which the signature is generated and shall be formatted as a dateTime XML DataType. The schema definition for the SigningTime signature property is:

<element name="SigningTime" type="xsd:dateTime"/>

5.3 Certificates and Certificate Management

This standard uses the Internet X.509 Public Key Infrastructure Profile (RFC 5280 [15]) as the base profile for certificates used by an ATSC 3.0 TLS server and ATSC 3.0 application signing authority authentication.

The following types of certificate are used by ATSC 3.0 devices during the authentication process:

- One or more root certificates. These are trusted self-signed certificates issued by a trusted certificate authority as the root of trust. Each certificate path validation process completes when a trusted root certificate is reached.
- Certificate authority certificates. These certificates are issued by a trusted root certificate authority or a certificate authority whose certificate path can be validated to a trusted root certificate authority.
- TLS server certificates. These certificates are issued by a trusted certificate authority and are designated for use in server authentication.
- ATSC 3.0 application signer certificates. These certificates are issued by a trusted certificate authority and are designated for use in code signing.
- OCSP responder certificates. These certificates are issued by a trusted certificate authority and are designated for use in OCSP responder authentication.

The client is expected to perform certificate chain validation as specified in RFC 5280 [15] using the certificate status information provided by the ATSC 3.0 server in stapled OCSP Response messages (see Sections 5.1.1.1.5 and 5.5.2) as a reliable source for revocation information.

5.3.1 Certificate Profiles

The profile specified in RFC 5280 [15] is further constrained for certificates used in ATSC 3.0.

5.3.1.1 General

All ATSC 3.0 certificates shall be X.509 version 3 certificates.

All keys contained in ATSC 3.0 certificates shall be either RSA keys with a minimum size of 2048 bits encoded as specified in RFC 3279 [10] or ECDSA keys which use the elliptic curve groups and point format defined above (Section 5.1.1.1.2) and encoded as specified in RFC 5480 [17].

All RSA signatures contained in ATSC 3.0 certificates shall be encoded according to the RSA signature algorithms specified in RFC 3279 [10] and RFC 4055 [11].

All ECDSA signatures contained in ATSC 3.0 certificates shall be encoded according to the ECDSA signature algorithms specified in RFC 5758 [19] and shall use one of the hash algorithms specified above (Section 5.1.1.1.3) for use with the ECDSA signature algorithm.

All ATSC 3.0 certificates shall contain a Key Usage extension containing at least the digitalSignature value and with values constrained as specified in RFC 3279 [10] and RFC 4055 [11].

ATSC 3.0 devices need not process the Authority Information Access or the Subject Information Access extensions.

5.3.1.2 Root Certificate Profile

The RSA key size for any root certificate shall be at least 2048 bits and should be 4096 bits.

The ECDSA key size for any root certificate shall be at least 384 bits.

5.3.1.3 Certificate Authority Certificate Profile

The RSA key size for any certificate authority certificate shall be at least 2048 bits.

The ECDSA key size for any certificate authority certificate shall be at least 256 bits.

5.3.1.4 Server Authentication Certificate Profile

The RSA key size for this certificate shall be at least 2048 bits.

The ECDSA key size for any server authentication certificate shall be at least 256 bits.

The Subject Alternative Name extension shall be present and shall include either the DNS Name or the IP Address of the server being authenticated.

The Extended Key Usage extension shall be present and shall be set to the value id-kp-serverAuth to indicate that the certificate is used in TLS server authentication.

5.3.1.5 ATSC 3.0 Application Signer Certificate Profile

The RSA key size for any application signer certificate shall be at least 2048 bits.

The Key Usage extension shall be marked as critical and shall include only the digitalSignature value.

The Extended Key Usage extension shall be present, marked as critical, and shall be set to the value id-kp-codeSigning to indicate that the certificate is used in the signing of downloadable executable code.

5.3.1.6 OCSP Responder Certificate Profile

The RSA key size for any OCSP responder certificate shall be at least 2048 bits.

The ECDSA key size for any OCSP responder certificate shall be at least 256 bits.

The Extended Key Usage extension shall be present and shall be set to the value id-kp-OCSPSigning to indicate that the certificate is used to sign OCSP responses.

5.4 ATSC 3.0 Client Certificate Storage

See CEA 2053 [28], which describes secure storage of certificates, and the mechanism(s) for modifying certificates used by client devices.

Clients provide secure storage for the following set of certificates:

- The set of trusted root certificates
- The set of trusted signing certificate authority certificates
- The set of trusted OCSP responder certificates

Certificates are changed over time, either by client device code download or by other means.

5.5 Certificate Revocation and Status Information

The management of certificate status is under the control of the issuing authority which works according to their defined certification practices and policies. Each certificate authority that issues certificates used by an ATSC 3.0 server or ATSC 3.0 application signing authority is responsible for the timely supply of certificate status information to the OCSP responder(s). The specific

methods by which this information is made available to the OCSP responder are beyond the scope of this specification.

5.5.1 Certificate Revocation and Status Information for TLS Server Certificates

An ATSC 3.0 server shall request certificate status information from an OCSP responder at least once per minute for each server authentication certificate that it provides as server identification when establishing a TLS connection. The request shall be in the format specified in RFC 6960 [22], shall be unsigned and the only extension included in the request shall be the Preferred Signature Algorithms extension.

> Note: In order to satisfy clients that support different signature algorithms, a server may need to request certificate status information from the same OCSP responder using different values in the Preferred Signature Algorithm extension. In particular, if some clients no longer support SHA-1 hash calculations while other clients do not support SHA-2 hash calculations, the server will need to maintain OCSP Responses signed using each of these hash calculations in order to support the full set of clients.

5.5.2 Certificate Revocation and Status Information for ATSC 3.0 Application Signing Certificates

An ATSC 3.0 application signing authority shall request certificate status information from an OCSP responder for the signing authority certificate that validates the signing key each time that key is used in a signing operation. The OCSP request shall indicate that the preferred signature algorithm to be used by the OCSP responder is RSA with SHA-256.

The SigningTime associated with the ATSC 3.0 application signature and the producedAt time of the corresponding OCSP Response providing the status of the signing authority certificate shall differ by no more than one minute. The ATSC 3.0 application signing authority shall include the OCSP Response in the signed application and should not issue a signed application where the OCSP Response indicates that the status of the signing authority certificate (as specified in RFC 6960 [22]) is other than “good”.

The application signing authority shall include an OCSPResponse child element within each X.509Data element of the digital signature structure (see http://www.w3.org/TR/xmldsig-core1) that contains its signing authority certificate. The OCSPResponse child element shall be included as an “Element from an external namespace” (see http://www.w3.org/TR/xmldsig-core1) and shall be formatted as base64 encoding of the OCSP Response received from the ATSC approved OCSP responder.

A client uses the OCSP Response data that it receives to verify that the certificates that authenticate the application signing authority are valid at the time the application is signed. See CEA 2053 [28].

5.6 [Companion Device Pairing]

An ATSC 3.0 device can establish a pairing relationship with one or more companion devices. This pairing process is used to establish a pre-shared symmetric key that is unique between the ATSC 3.0 device and the companion device. This pre-shared symmetric key can be used to establish a TLS connection between the devices using one of the following Cipher Suites (as specified in ECDHE-PSK [26]):

```
TLS_ECDHE_PSK_WITH_AES_128_GCM_SHA256
TLS_ECDHE_PSK_WITH_AES_256_GCM_SHA384
```

or the following Cipher Suite (as specified in CHACHA20_POLY1305 [25]):

```
TLS_ECDHE_PSK_WITH_CHACHA20_POLY1305_SHA256
```

and using the Elliptic Curve Parameters specified in Section 5.1.1.1.2.

It is recommended that the pre-shared symmetric key is derived in the following manner:

1) The keying material includes elements that are provided by the user (for example a pass phrase) and elements that are unique to the companion device (for example its MAC address).
2) The key derivation function complies with RFC 5869 [20].
3) The derived pre-shared symmetric key is at least 64 octets in length.

The TLS connection can be initiated by either the ATSC 3.0 device or the companion device (acting as the client) and the TLS client handshake request is expected to indicate the use of the TLS 1.3 protocol. Where the device that receives the TLS client handshake request (acting as the server) does not support the TLS 1.3 protocol, that device may offer to downgrade to TLS 1.2 protocol in its TLS server handshake response. The devices are not expected to allow a TLS connection to be downgraded to a protocol other than the TLS 1.2 protocol.

Devices that have established a TLS connection with a companion device are not expected to support TLS Session Resumption (see Section 5.1.1.1.6) for those connections since there is minimal overhead in re-establishing the session from a new TLS client handshake without the expense of storing the session resumption information.

5.7 Content Protection

5.7.1 Types of Content Protection

5.7.1.1 Content May Be “In” or “Out” of Protection

Editor’s note: Be sure to clarify if this is different from simple data based timeout, may need to change 5.7.1.2.1

5.7.1.2 Embedded Content Protection

(e.g. BluRay) This may also include encryption of content before it enters the ATSC defined stream

5.7.1.2.1 Use Embedded Content Protection with Date-based Timeout

5.7.1.2.2 Limits of Embedded Protection

Depending on type of embedded protection the system may not be able to support protection if the underlying content protection scheme is broken

5.7.1.3 OOB or “Backend” Authentication

5.7.2 Protecting Content from Unauthorized Viewing

Sometimes described as Conditional Access, and will protect either full services or components with a given service. [Note: a standard elsewhere may need to specify some forms of protection for user data, such as passwords and credentials.]

5.7.2.1 User Authentication

[Note: Unclear whether this document needs to describe authentication methods or mechanisms. It may be a function of the conditional access/DRM system in use.]

- Local Authentication

- SAML/OpenID/OAuth/SSO etc.
- Device-based authentication
  - End users may be authenticated using receiver-based authenticators. Examples of such are proof-of-possession authenticators such as hardware tokens (e.g., CableCard), or biometric authenticators. Such authenticators can attest to a service provider as to is genuineness or certfiability,
  - Device-based authentication is compatible with many standardized authorization protocols (e.g., see https://tools.ietf.org/html/draft-ietf-oauth-proof-of-possession-03).
  - Device-based authentication can be used to compliment media encryption techniques such as conditional access or DRM if user verification is required as part of the decryption process.

5.7.2.2 Authorization and Authorization Databases

5.7.3 Protecting Content from Unauthorized Copying and Redistribution

5.7.3.1 Required Mechanisms to Prevent Undesired/Illegal Copying of Content After Delivery to an End Device

5.7.4 Common Encryption

ATSC 3.0 uses MPEG-defined ISO Base Media File Format (ISO BMFF) [3] as the media container that will be sent through the broadcast emission to the receiver for consumption. MPEG Common Encryption (CENC) [2] has been specified as a digital rights management system suitable for use with ISO BMFF. Any media that requires DRM encryption shall use MPEG Common Encryption (CENC).

The Common Encryption (`cenc`) protection scheme specifies encryption parameters that can be applied by a scrambling system, along with key mapping methods via common key identifier (KID) for use by different DRM systems, such that the same encrypted version of a file can be handled by different DRM systems which can store proprietary information for licensing and key retrieval in designated metadata boxes of the ISO BMFF file – specifically, the Protection System Specific Header Box (`pssh`) as defined in ISO/IEC 23001-7 [2].

The key advantage of CENC is that by providing a common way to encrypt content, it decouples the content encryption from the key acquisition and thus provides support for multiple DRM systems.

The CENC mechanism only encrypts media samples or parts thereof and leaves the ISOBMFF metadata such as the file and track structure boxes un-encrypted to enable players to recognize and read the file correctly and acquire any required license. CENC supports the encryption of NAL-based video encoding formats such as AVC and HEVC, thus offering sub-sample encryption capability, where only the video data of a sub-sample is encrypted, while the NAL header is not. This flexibility can be used to offer a free preview of the video, enable editing and processing of the video, or provide free access to some service components such as audio. By providing offsets to the encrypted byte ranges inside a sample in an ISOBMFF “mdat” box, players can easily process the file and pass the encrypted chunks to the decryptor for decryption and playback.

In order for decryption to work, CENC provides the following information in the ISOBMFF:

- Key Identifiers (KID): a key ID must be associated with every encrypted sample in a track. In case a single key is used for the whole track.
- Initialization Vectors (IV): the IV, a random number used to initialize an encryption function, is used for randomization and removal of semantics and is essential for strong

protection. For every sample, the IV must be known in order to be able to construct the decryption key.

- License Acquisition Information: information about license acquisition is specific to each DRM system. The player needs to support at least one of the DRM systems that offer access to the encrypted stream.

CENC defines a way to store the previous information in the ISOBMFF. The Key Identifiers may be provided:

- As the default_KID in the track encryption box "tenc", when a single key applies to the whole track,
- As a key for a set of samples that share the same encryption key, provided in a sample grouping structure using the sample group description box "sgpd".

The IV for every sample is provided as part of the sample auxiliary information in the "mdat" box or in the "senc" box together with information about the position of the encrypted chunks.

The license acquisition information is provided as part of the protection system specific header box "pssh", where each DRM system is identified by a SystemID. The "pssh" box also provides a list of the provided Key Identifiers and opaque system-specific information that describes how to acquire the keys identified by the supported key ids.

Figure 5.1 depicts the encrypted track structure.

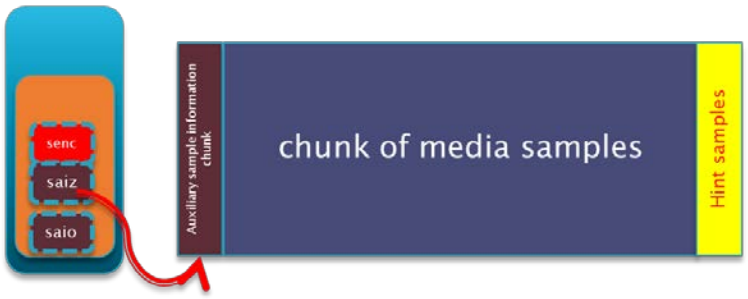

Figure 5.1 Storage of CENC related information.

5.7.5 Encrypted Media Extensions

W3C Encrypted Media Extensions (EME) [4] specifies JavaScript APIs which enable a web application to facilitate the exchange of decryption keys between a device-resident DRM system agent, referred to as the Content Decryption Module (CDM), and a key source or license server located somewhere on the network, to support the playback of encrypted audio and video media content. EME is based on the HTML5 Media Source Extensions specification [6] which enables adaptive bitrate streaming in HTML5 using, for example, MPEG-DASH [7] with MPEG-CENC (Common Encryption) [2] protected content The architecture of EME is depicted in Figure 5.2, which depicts the primary interactions of the EME workflow between the functional entities involved in the detection of encrypted content and the subsequent acquisition of license and key material, to enable content decryption and playout.

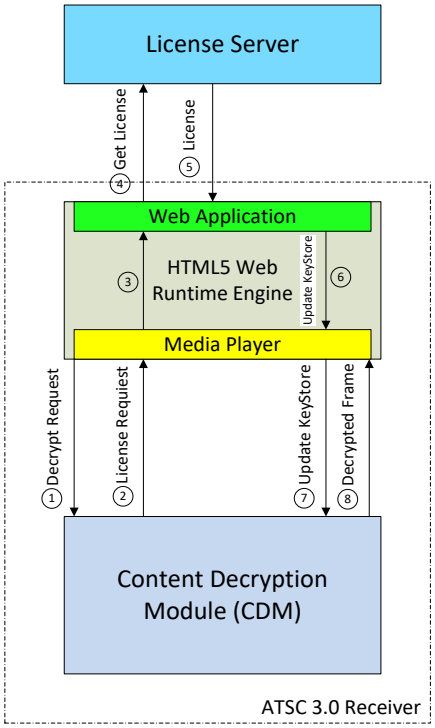

Figure 5.2 Encrypted Media Extensions workflow.

The principal objects in EME are `MediaKeySession` and `MediaKeys`. The web application creates a `MediaKeySession` object, which represents the lifetime of a license and its key(s), by calling `createSession()` on the `MediaKeys` object. The app initiates the request for a license by passing the media data obtained in the `encrypted` event handler to the CDM. In turn, the CDM for the selected DRM system will generate a data blob (license request) and deliver it back to the app, which will then send that request to the license server. The returned license from the server is then passed by the app to the CDM, by using the `update()` method of the `MediaKeySession`. The CDM and/or the browser will use keys stored in the key session to decrypt media samples as they are encountered. The CDM may be either embedded in the web browser, or run in a trusted environment, depending on the required level of security, in passing the decrypted frames to a decoder.

5.7.6 Extensions to MMT Signaling for CENC Support

[Editor's Note: This section (5.7.6) may require significant modification and/or may be superseded by documentation elsewhere (e.g., an S33-1 document, or MPEG).]

5.7.6.1 Security Information (SI) Descriptor

5.7.6.1.1 Introduction

The SI descriptor is an asset descriptor that can be used by the MMT sender to provide information about the content encryption applied to the asset as part of DRM protection or Conditional Access. For content protection using common encryption, the contents of this descriptor are extracted from the "pssh" box that is located as part of the MPU metadata [Editor's Note: need reference to "pssh" box source specification].

5.7.6.1.2 Syntax

The syntax of the SI descriptor is provided in Table 5.1.

Table 5.1 SI Descriptor

Syntax	Value	No. of Bits	Mnemonic
SI_descriptor() {			
descriptor_tag	[TBD]	16	uimsbf
descriptor_length		16	uimsbf
security_system_count	N1	8	uimsbf
reserved	'000 0000'	7	uimsbf
system_provider_url_flag		1	
if (system_provider_url_flag) {			
system_provider_url_length		N2	uimsbf
for (i=0; i<N2; i++) {			
system_provider_url _byte		8	uimsbf
}			
}			uimsbf
for (i=0;i<N1;i++) {			
system_id		16*8	uimsbf
kid_count	N3	16	uimsbf
for(i=0;i<N3;i++) {			
KID		16*8	uimsbf
}			
data_size	N4	32	uimsbf
for(j=0;j<N4;j++) {			
data		8	uimsbf
}			
}			
}			

5.7.6.1.3 Semantics

descriptor_tag – A 16-bit unsigned integer field that shall have the value [TBD], indicating the type of a descriptor.

descriptor_length – A 16-bit unsigned integer field that shall indicate the length in bytes counting from the next byte after this field to the last byte of the descriptor.

security_system_count – An 8-bit unsigned integer field that shall provide the number of DRM or CAS systems information that can process and handle the content protection, access control and rights management.

system_provider_url_flag – A 1-bit field that shall indicate the presence of the URL of a provider for the system. This URL may be used for downloading and installing the system, when needed.

system_provider_url_length – A N2-bit unsigned integer field that shall indicate the length of the URL.

system_provider_url_byte – An 8-bit unsigned integer field that shall indicate the UTF-8 character of a URL associated with the system provider.

system_id – A 16-byte integer field that shall indicate the UUID that uniquely identifies the DRM system.

KID_count – A 16-bit unsigned integer field that shall specify the number of KID entries in the descriptor.

KID – A 16-byte unsigned integer filed that shall identify a key that the data field applies to.

data_size – A 32-bit integer field that shall indicate the size in bytes of the data field.

data – An 8-bit unsigned integer field that shall carry DRM system specific data.

5.7.6.2 License Signaling Message

5.7.6.2.1 Introduction

The license signaling (LS) message carries the license information targeted to a specific receiver or group of receivers. This information is delivered to receivers that do not have a return channel.

5.7.6.2.2 Syntax

The syntax of license signaling (LS) message is defined in Table 5.2.

Table 5.2 LS Signaling Message

Syntax	Value	No. of Bits	Mnemonic
LS_message () {			
message_id		`	uimsbf
version		8	uimsbf
length	N1	32	uimsbf
system_id		16*8	uimsbf
number_of_licenses	N2	32	uimsbf
for(i=0;i<N2;i++) {			
message_payload {			
license_message_hash_length	N3	8	uimsbf
license_message_hash		8*N3	uimsbf
encrypted_license_data_length	N4	`	uimsbf
encrypted_license_data		8*N4	uimsbf
}			
}			
}			

5.7.6.2.3 Semantics

message_id – A 16-bit unsigned integer field that shall indicate the identifier of the LS message.

version – An 8-bit unsigned integer field that shall indicate the version of the LS message.

length – A 32-bit unsigned integer field that shall indicate the length of the LS message in bytes, counting from the beginning of the next field to the last byte of the LS message. The value '0' is not valid for this field.

system_id – A 16-byte unsigned integer field that shall indicate the UUID that uniquely identifies the DRM system.

number_of_licenses – A 32-bit unsigned integer field that shall indicate the number of licenses in the LS message.

license_message_hash_length – An 8-bit unsigned integer field that shall indicate the length in bytes of the license message hash.

license_message_hash – An N3-byte integer field that shall indicate the license message hash code used to identify the target receiver or group of receivers for the enclosed license. The license

message hash is generated by the content decryption module in the same way as the license server. The hash generation algorithm is DRM system specific.

encrypted_license_data_length – A 16-bit unsigned integer filed that shall indicate the length of the encrypted license data.

encrypted_license_data – An N4-byte unsigned integer field that shall indicate the encrypted license that corresponds to the license message of which the hash value is included in this message. The license is encrypted using the certificate of the targeted receiver or group of receivers.

5.7.6.3 License Revocation Message (LR)

5.7.6.3.1 Introduction

The license revocation message is defined to allow the DRM system provide to signal that it has revoked the license for a user or group of users.

5.7.6.3.2 Syntax

The syntax of the LR signaling message is defined as follows:

Table 5.3 LR Signaling Message

Syntax	Value	No. of Bits	Mnemonic
LR_message () {			
message_id		16	uimsbf
version		8	uimsbf
length	N1	32	uimsbf
system_id		16*8	uimsbf
number_of_licenses	N2	32	uimsbf
for(i=0;i<N2;i++) {			
message_payload {			
encrypted_license_challenge_length	N4	16	uimsbf
encrypted_license_challenge		8*N4	uimsbf
}			
}			
}			

5.7.6.3.3 Semantics

message_id – A 16-bit unsignd integer field that shall indicate the identifier of the LR message.

version – An 8-bit unsigned integer that shall indicate the version of the LR message.

length – A 32-bit unsigned integer field that shall indicate the length of the LR message in bytes, counting from the beginning of the next field to the last byte of the LR message. The value ‘0’ is not valid for this field.

system_id – A 16-byte unsigned integer field that shall indicate the UUID that uniquely identifies the DRM system.

number_of_licenses – A 32-bit unsigned integer field that shall indicate the number of licenses in the LS message.

encrypted_license_challenge_length – A 16-bit unsigned integer field that shall indicate the length of the encrypted license challenge.

encrypted_license_challenge – An N4-byte unsigned integer field that shall indicate an encrypted license challenge. The license challenge is encrypted using the certificate of the targeted receiver or group of receivers.

5.7.6.4 CENC and EME over Pure Broadcast Channel for MMT Streaming Service Delivery

When used over a pure broadcast channel without any return channel, the player application should use the message that is returned by EME from the `generateRequest()` method to locate the license that is delivered as part of the MMT signaling and pass it to the CDM through the update method of the MediaKeySession object.

The DRM System delivers the license for every single receiver encrypted with the public key for that receiver and identified by a hash of the message that was returned by the *generateRequest* method. Figure **5.3** depicts the license retrieval over broadcast.

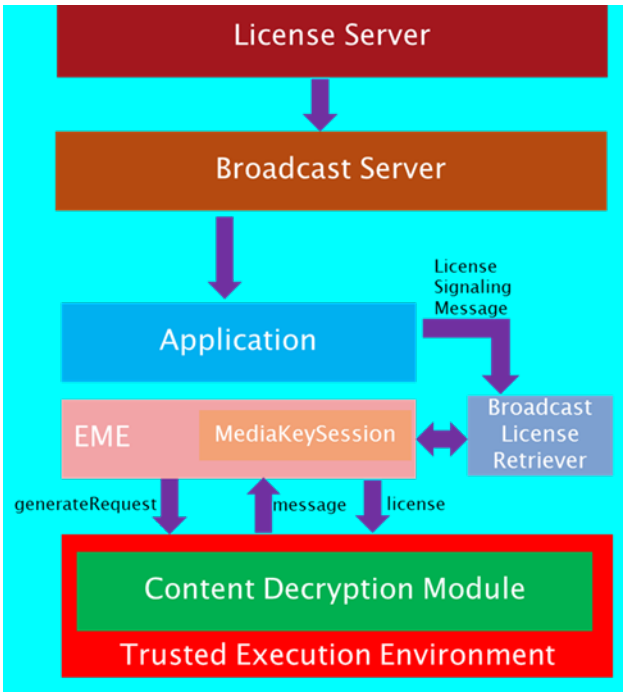

Figure 5.3 Broadcast license retrieval.

The application forwards all broadcast license messages to the Broadcast License Retriever (BLR). The BLR is initially configured with the client certificate, which contains the client's public key. The BLR identifies the messages that are targeted to the receiver through a hash mapping of the request message that was provided by the CDM. When a message targeting the receiver is located, the license is decrypted and passed the MediaKeySession, which on its turn passes it to the CDM for content decryption.

[Editor's Note: It should be verified whether the operation as described above on DRM support for broadcast-only receivers, and which is generically applicable to both ROUTE/DASH and MMTP delivery of streaming services, is supported by commercial DRM systems and license servers in conjunction with MPEG CENC based content protection. The latest DASH-IF ATSC 3.0 Profile states (in Sec. 7.1) that broadcast delivery of individualized licenses (cryptographically bound to a device or user) is not specified by DASH-IF. DRM support for broadcast-only receivers implies that only static and offline methods can be used to provide device credentials (e.g., device ID and certificate) to the DRM system to establish trust in the device and enable subsequent broadcast delivery of initial security key material protected by the device's public key. Complete absence of broadband connectivity precludes dynamic and interactive license/key renewal

transactions, which may be required by CENC-compatible DRM system from the security standpoint.]

5.7.7 ROUTE/DASH Support for CENC and EME

ROUTE/DASH support for CENC may be found in [9], Section 7. Information on the interaction of ROUTE/DASH and EME is provided in Annex A.

5.7.8 Required Mechanisms to Support Forensic or Copyright Watermarking

Be sure to cover how this may affect emission since the stream must carry the watermark from end to end.

5.7.9 Required Mechanisms to Support OTT Content Security

5.7.10 Required Mechanisms to Support Home Network Redistribution or Prevention of Home Network Redistribution

5.8 Backend Business Systems

Editor's Note: Cover backend requirements for both transport and content protection, hopefully certificate management, user management, and other feature support can use common elements where appropriate.

Annex A: ROUTE/DASH Client Processing for Common Encryption (CENC) and Encrypted Media Extensions (EME) (Informative)

A.1 **INTRODUCTION**

This Annex describes the operation of a ROUTE-enabled ATSC receiver when accessing CENC-protected media.

ROUTE/DASH supports the Common Encryption (CENC) framework for multiple DRM systems to protect DASH-formatted streaming service content. ROUTE/DASH includes protection system specific and proprietary signaling information delivered in two way: a) in predetermined locations in the MPD, and b) carried inband to the DASH content, in designated metadata boxes of the ISO BMFF format for movie fragments [3], in accordance to the usage as defined in ISO/IEC 23001-7 [2]. Most of the details can be found in the DASH-IF IOP specification [8], and are compliant to the DASH-IF Broadcast IOP specification [9].

A.1.1 Basic CENC Operation in ROUTE/DASH

This section describes the basic mechanisms of how DASH-formatted streaming content, protected by a DRM system, and delivered by the ROUTE protocol, can be decrypted and played out. It describes, in the context of CENC and EME, the required interactions within the receiver and between the receiver and a license server, for license and key acquisition and subsequent content decryption and playout.

Two alternative methods are described using message/interaction flows. In the first (see Section A.1.1.1), acquisition of the DRM license and content key by the CDM occurs prior to the start of the streaming program delivery. In the second method (as described in Section A.1.1.2), acquisition of the DRM license and content key by the CDM occurs during the program delivery. The first method, by bootstrapping the license and key acquisition prior to the start of the broadcast program, may be preferable over the second in reducing start-up delay for playing out of DRM-protected content, although the actual gains depend on the specific service requirements and practical license acquisition latency over the broadband network.

A.1.1.1 License Acquisition Prior to Program Delivery

Figure A.1 is an example message flow illustrating the method whereby the `ContentProtection` descriptor in the MPD is used to provide the affiliated metadata, such as license server's URL and default KID to the CDM. This triggers the CDM to request and obtain the DRM license, and associated key material prior to the media delivery. When the encrypted media content is later broadcast, the receiver has the necessary decryption keys to render the content immediately.

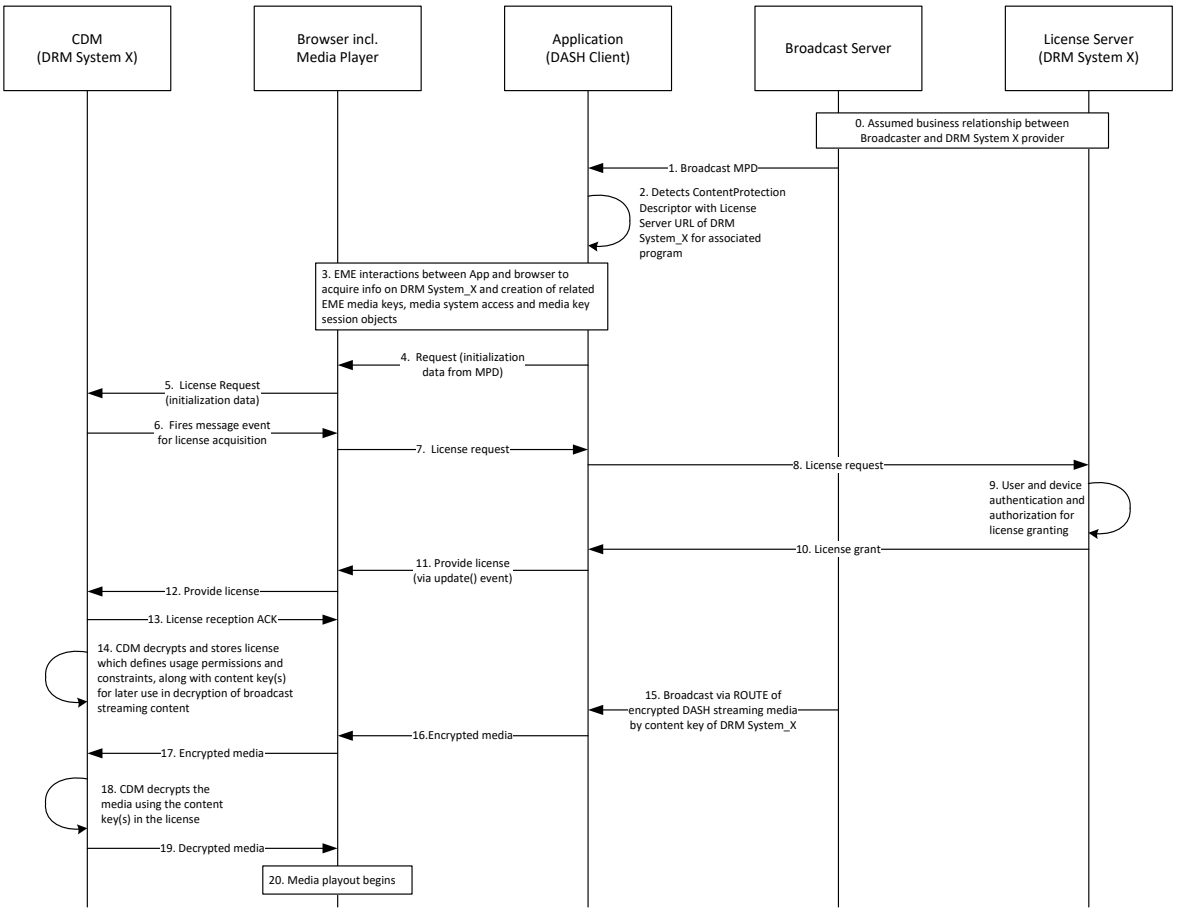

Figure A.1 DRM license and key acquisition before start of program in ROUTE/DASH.

A.1.1.2 License Acquisition During Program Delivery

Figure A.2 is an example message flow illustrating the method whereby the protection system related metadata carried in the DASH Segments, specifically the '`pssh`' box in the '`moov`' or '`moof`' box is used to provide the affiliated metadata, such as license server's URL and default KID to the CDM. During the interval that it takes for the CDM to request and obtain the DRM license, and associated key material, the program cannot be rendered. Due to the greater start-up delay associated with this method, it is suggested that the alternative method in Section A.1.1.1 be employed by the broadcaster.

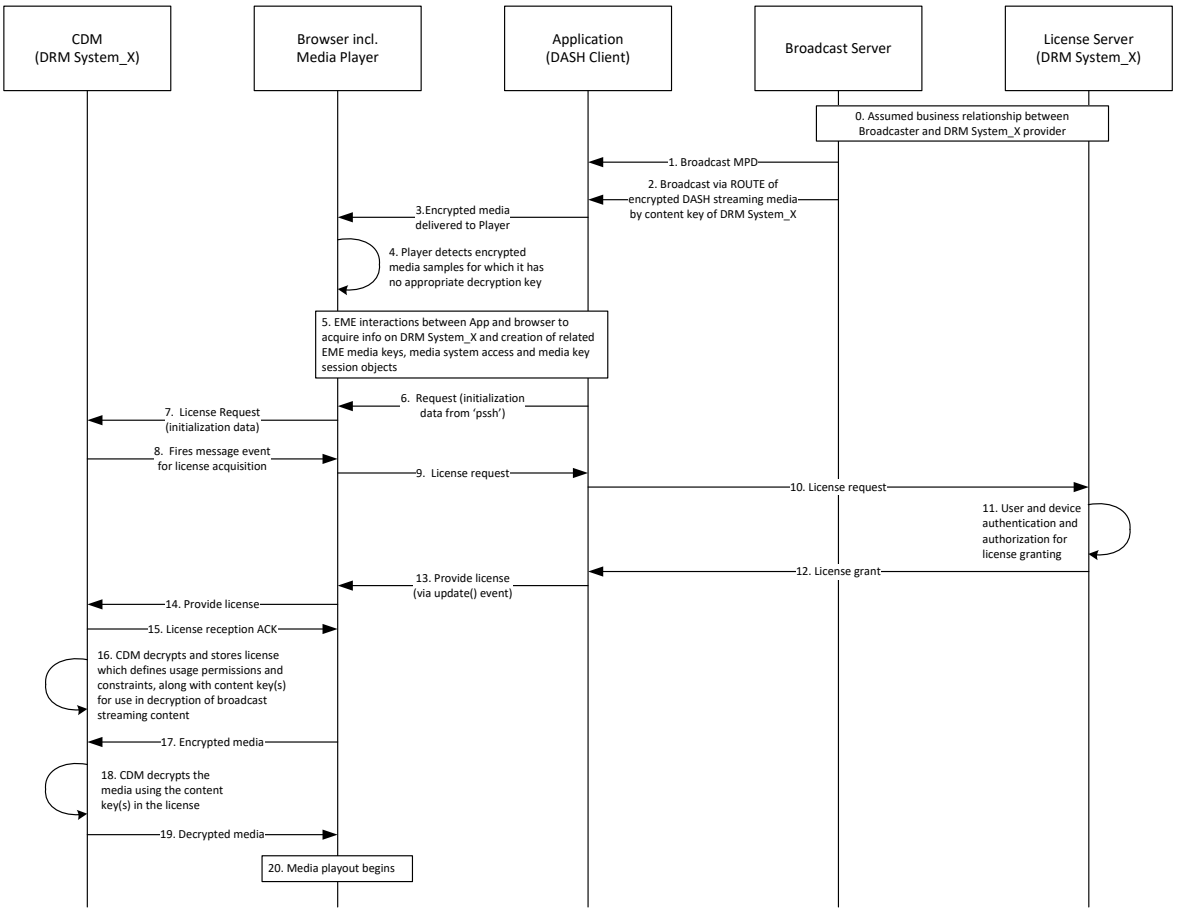

Figure A.2 DRM license and key acquisition during program delivery in ROUTE/DASH.

A.1.2 Solution Framework for DRM and CENC

[Editor's Note: The description in this section should be verified to be aligned with the DASH-IF's Broadcast IOP specification [9], which is under development.]

ISO-IEC 23001-7 [2] represents the normative standard for common encryption in conjunction with ISO BMFF [3], and includes the following technology components used for DRM protection of streaming media carried by ROUTE/DASH:

- Common encryption of NAL structure video and other media with AES-128 CTR mode
- Support for decryption of individual Representations by one or more DRM systems
- Key rotation to enable the change of the content encryption keys over time
- Extension of the **`ContentProtection`** descriptor to enable the signaling of `default_KID` and '`pssh`' parameters in the MPD

The primary DRM related signaling components and tools available for use in ROUTE/DASH are as follows:

1) The **`ContentProtection`** descriptor in the MPD which contains the URI for signaling of the use of Common Encryption or the specific DRM scheme being used.

2) Parameters of the 'tenc' box, carried as part of protection scheme information in the movie box ('moov') of the Initialization Segment, which specify encryption parameters and default_KID. The default_KID information may also be carried out-of-band in the MPD.
3) Signaling of common encryption sample auxiliary information in the form of initialization vectors and subsample encryption ranges, if applicable, using the 'senc box as defined in ISO/IEC 23001-7 [2], or via the SampleAuxiliaryInformationSizesBox ('saiz') and a SampleAuxiliaryInformationOffsetsBox ('saio').
4) 'pssh' license acquisition data or keys for each DRM system in a format that is protection system specific. 'pssh' refers to the Protection System Specific Header box as defined in ISO/IEC 23001-7 [2], and which may be stored in the Initialization Segment or in Media Segments. It may also be present in a **cenc:pssh** element in the MPD. Note that while the presence of **cenc:pssh** information in the MPD increases the MPD size, it may allow faster parsing, earlier access, and addition of DRM systems without content modification.
5) Key rotation to enable modification over time in the entitlement for access to continuous live content. Details on how key rotation operates in the protection of broadcast DASH streaming content can be found in the DASH-IF Interoperability Points documents [8], [9] (*nb.* Chapter ___).

A graphical representation of the box structure pertaining to encryption metadata support for video-on-demand (VoD) content is shown in Figure A.3.

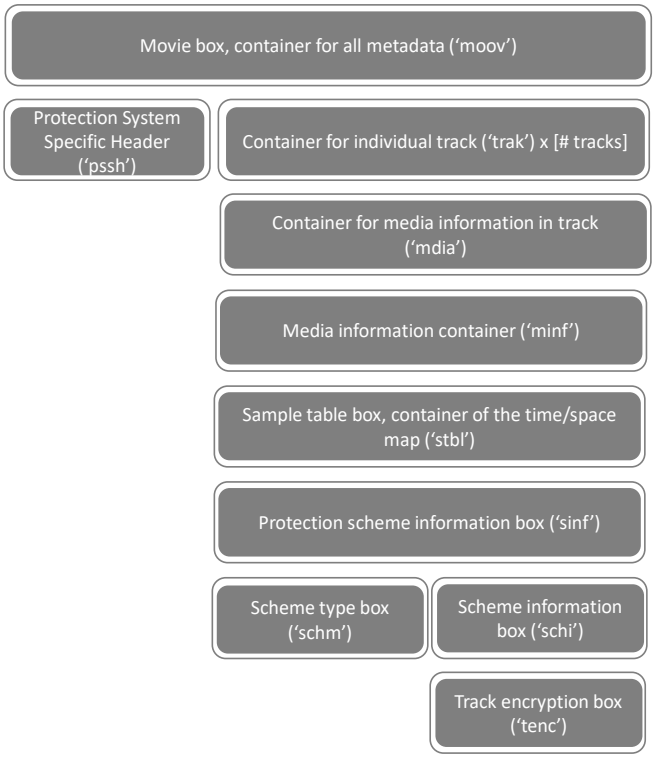

Figure A.3 CENC-related metadata structure for protection of VoD content by a single key.

A graphical representation of the box structure pertaining to encryption metadata support for live streaming content is shown in Figure A.4.

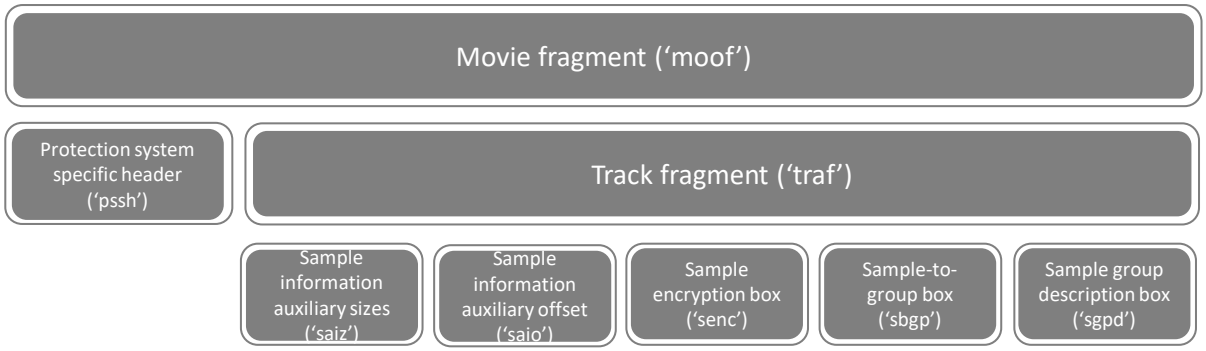

Figure A.4 CENC-related metadata structure for protection of live streaming content.

A.1.3 MPD Support for Encryption and DRM Signaling

[Editor's Note: The description in this section should be verified to be aligned with the DASH-IF's Broadcast IOP specification [9], which is under development.]

The MPD contains signaling of the content encryption and key management methods used to enable the DRM client to determine whether it is capable to play out the content. That information is contained in the **`ContentProtection`** descriptor, of which at least one instance must be present in each **`AdaptationSet`** element describing encrypted content.

A.1.3.1 Use of the Content Protection Descriptor for mp4 Protection Scheme

As specified by MPEG-DASH [7], Representations based on ISO BMFF [3], a **`ContentProtection`** descriptor with `@schemeIdUri` value of "`urn:mpeg:dash:mp4protection:2011`" indicates that the content is encrypted with the scheme as indicated in the `@value` attribute. The file structure of content protection schemes is specified in MPEG-DASH [7], Section 5.8.5.2, and the `@value` is '`cenc`' in denoting the Common Encryption scheme. Such value for the `@schemeIdUri` of the **`ContentProtection`** descriptor along with `@cenc:default_KID` as defined within the "`urn:mpeg:cenc:2013`" extension namespace may be sufficient for the receiver to acquire a DRM license, or identify a previously acquired license that can be used to decrypt the Adaptation Set.

When the `@cenc:default_KID` is present for each Adaptation Set, it allows a player to determine if a new license needs to be acquired for each Adaptation Set by comparing their default_KIDs with each other, and with the default_KIDs of stored licenses. A player can simply compare these KID strings and determine what unique licenses are necessary without interpreting license information specific to each DRM system.

A.1.3.2 Use of Content Protection Descriptor for uuid Scheme

A UUID **`ContentProtection`** descriptor in the MPD may indicate the availability of a particular DRM scheme for license acquisition. An example is shown below:

```
<ContentProtection
schemeIdUri="urn:uuid:xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx"
value="DRMNAME version"/>
```

The `schemeIdUri` uses a UUID URN with the UUID string equal to the registered SystemID for a particular DRM system. This is specified in MPEG DASH [7], Section 5.8.5.2. A list of known DRM System IDs can be found in the DASH identifier repository at: http://www.dashif.org/identifiers/content-protection.

A.1.3.3 Protection System Specific Header Box in the MPD

A '`pssh`' box is defined by each DRM system for use with their registered SystemID, and is nominally stored in the movie box ('`moov`') and additionally may be present in the movie fragment box ('`moof`'). The same box can also be stored in the MPD within a **`ContentProtection`** Descriptor for a UUID scheme using the extension element **`cenc:pssh`** in the "`urn:mpeg:cenc:2013`" namespace, as defined by ISO/IEC 23001-7 [2]. Carrying the **`cenc:pssh`** element and also the `cenc:default_KID` attribute as defined by the same "`urn:mpeg:cenc:2013`" extension namespace, in the MPD, can be useful in supporting key identification, license evaluation, and license retrieval before the availability of Initialization Segments for live content. This enables ATSC receivers, via the broadband network, to be able to acquire license requests prior to the start of the program. Also, spreading out over time license requests avoids potential overloading of the license server due to a high volume of simultaneous license requests from many viewers, starting when at an Initialization Segment containing license acquisition information in '`pssh`' becomes available. With **`cenc:default_KID`** indicated in the mp4protection **`ContentProtection`** descriptor for each Adaptation Set, the DRM client in the receiver can determine whether

- the associated decryption key for the program is available to the viewer (e.g., without purchase or subscription),
- if the key is already downloaded, or
- which license the client should download before the `@availabilityStartTime` of the program, based on the `default_KID` of each Adaptation Set element selected.

End of Document