SMPTE STANDARD

D-Cinema Operations — Log Record Format Specification

Table of Contents

Approved
March 3, 2008

Foreword

SMPTE (the Society of Motion Picture and Television Engineers) is an internationally-recognized standards developing organization. Headquartered and incorporated in the United States of America, SMPTE has members in over 80 countries on six continents. SMPTE's Engineering Documents, including Standards, Recommended Practices and Engineering Guidelines, are prepared by SMPTE's Technology Committees. Participation in these Committees is open to all with a bona fide interest in their work. SMPTE cooperates closely with other standards-developing organizations, including ISO, IEC and ITU.

SMPTE Engineering Documents are drafted in accordance with the rules given in Part XIII of its Administrative Practices.

SMPTE Standard 430-4 was prepared by Technology Committee DC28.

Introduction

Many devices in D-Cinema exhibition environments must generate log records, which can record all manner of events in the normal and exceptional operation of systems. In order to achieve interoperability, it is desirable that log records are presented in a common basic format.

1 Scope

The purpose of this document is to define XML structures and schema for individual log records and sequences of those records in D-Cinema applications. While not all log records require authentication, this specification provides for optional authentication of records and sequences of records, using digital signatures. Authentication requirements are established by separate application specifications. This document does not define a communications protocol, but is limited to specifying the structure and format of log records and their content. This document does not define a format that devices must use internally, but it does define a format that is to be used when log records are interchanged with other devices.

2 Conformance Notation

Normative text is text that describes elements of the design that are indispensable or contains the conformance language keywords: "shall", "should", or "may". Informative text is text that is potentially helpful to the user, but not indispensable, and can be removed, changed, or added editorially without affecting interoperability. Informative text does not contain any conformance keywords.

All text in this document is, by default, normative, except: the Introduction, any section explicitly labeled as "Informative" or individual paragraphs that start with "Note:"

The keywords "shall" and "shall not" indicate requirements strictly to be followed in order to conform to the document and from which no deviation is permitted.

The keywords, "should" and "should not" indicate that, among several possibilities, one is recommended as particularly suitable, without mentioning or excluding others; or that a certain course of action is preferred but not necessarily required; or that (in the negative form) a certain possibility or course of action is deprecated but not prohibited.

The keywords "may" and "need not" indicate courses of action permissible within the limits of the document.

The keyword “reserved” indicates a provision that is not defined at this time, shall not be used, and may be defined in the future. The keyword “forbidden” indicates “reserved” and in addition indicates that the provision will never be defined in the future.

A conformant implementation according to this document is one that includes all mandatory provisions ("shall") and, if implemented, all recommended provisions ("should") as described. A conformant implementation need not implement optional provisions ("may") and need not implement them as described.

3 Normative References

The following standards contain provisions which, through reference in this text, constitute provisions of this standard. At the time of publication, the editions indicated were valid. All standards are subject to revision, and parties to agreements based on this standard are encouraged to investigate the possibility of applying the most recent edition of the standards indicated below.

[XML-SCH-1] XML Schema Part 1: Structures URL http://www.w3.org/TR/xmlschema-1-20041028/

[XML-SCH-2]XML Schema Part 2: Datatypes http://www.w3.org/TR/xmlschema-2-20041028/

Internet Engineering Task Force (IETF) (1997, May) URN Syntax, [WWW document]. URL http://www.ietf.org/rfc/rfc2141.txt

[RFC 3280] Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile URL: http://www.ietf.org/rfc/rfc3280.txt

[xml-c14n] Canonical XML Version 1.0 W3C Recommendation 15 March 2001 http://www.w3.org/TR/xml-c14n/

[xml-dsig] "XML-Signature Syntax and Processing" W3C Recommendation 12 February 2002
http://www.w3.org/TR/xmldsig-core/

[RFC 3275] "XML-Signature Syntax and Processing" http://www.ietf.org/rfc/rfc3275.txt

[DCMLTypes] XML Data Types for Digital Cinema. (SMPTE 433-2008)

[DC-Cert] D-Cinema Digital Certificate (SMPTE 430-2-2006)

[SHA-1] “FIPS PUB 180-1 Secure Hash Standard” http://www.itl.nist.gov/fipspubs/fip180-1.htm

[RFC 2253] “Lightweight Directory Access Protocol (v3): UTF-8 String Representation of Distinguished Names” http://www.ietf.org/rfc/rfc2253.txt

4 Overview (Informative)

The Log Record Format standardizes the reporting format for information that documents the usage and behavior of a Digital Cinema system. This format is intended to be used for communicating log event information at such points in the system where interoperability or a common format is a requirement. This format may be used to construct records that are unidirectional in nature, meaning that a bi-directional communications channel is not required in order to interpret the record content. However, additional efficiencies and features such as dynamic text internationalization can be achieved in a bi-directional communications environment.

In order to provide for maximum flexibility while maintaining interoperability, the Log Record format is designed to support a number of features which may be optional or required depending on any requirements which may be imposed by an application specification which uses this format. This format supports both authenticated and non-authenticated log records for security and non-security logging purposes, respectively. The format also supports record "sequences", or Log Reports, to reduce authentication overhead, and provides for "filtering" of authenticated record sequences in cases where it may be desirable to excise information contained in Log Reports while preserving a record of that information's existence.

In order to accomplish these objectives, the Log Record structure is broken down into three major sub-parts: Header, Body, and Signature, the first of which is required in a Log Record, and the second and third of which are optional. In addition, a number of fields in these sub-parts are optional for Log Records which are not required to support authentication, but available and may be required where authentication is a requirement. In Log Reports, some Body sections may be removed by filtering, so the Header is the only required part of a Log Record in a Log Report.

The Log Record Header contains enough information to identify the source, time, and classification of the log record. It may also include hashes to support data integrity and authentication. Conceptually, the Log Record Header contains just enough information to support automated classification and sorting of log events, without revealing information that may be considered sensitive or non-public.

The Log Record Body contains either the detail of the logged information in human-readable form, or index information to enable the translation of the log detail into human readable form, or both. This includes all available information about the logged event, except that which is conveyed in the Log Header. The Log Record Body is not encrypted, but may be authenticated through a hash of its contents, which is contained in the corresponding Log Record Header, which may be signed by a Log Signature. The (optional) Log Signature provides authentication of the header(s) of the current Log Record, or optionally authentication over a number of log records in a sequence.

This document does not specify the internal format for log records kept within a device. Rather, this document specifies the format for producing authenticated log records, if a device is required to do so.

5 Definitions

5.1 Definition of Terms

- Log Event – Any state or change of state in a system that is desirable to record in a log. Such an event results in the recording of log data.
- Log Data – Log Event information that is recorded and stored in a Log Record.
- Log Record – Standardized XML structure representing a discrete logged event.
- Log Report – Standardized XML structure containing one or more log records.

5.2 Namespace Prefixes

The following table lists the namespace prefixes used in the XSDL and XML examples in this document. Please refer to the normative references in this document and in [DCMLTypes] for specific references to the namespaces referred to in this table. Note that the prefixes themselves are not normative, and that instance documents may assign alternative prefixes in practice.

Prefix	Namespace Reference
xs	XMLSchema
ds	Xmldsig
xsi	XMLSchema-Instance
dcml	DcmlTypes
lr	LogRecord [This Document]

5.3 Log Record Namespace

The types defined in this document shall be included in an XML namespace, whose Namespace Name (URI) shall be:

```
http://www.smpte-ra.org/schemas/430-4/2008/LogRecord/
```

When published, the version number of the XML Schema representation of the namespace shall be set to "1.0", where "1" is the major part of the version number, and "0" is the minor part. If the namespace is subsequently revised to add new types not defined in this standard, but not to modify any of the existing types in a way that breaks reverse compatibility, the minor part of the version number shall be incremented by one. If the namespace is subsequently modified with result of changing any of the types defined in this document in a way that breaks reverse compatibility, the major part of the version number shall be incremented by one. The version number shall be specified by the version attribute the xs:schema element of the XML schema that this namespace represents.

6 Component Data Type

This section describes component data types, which shall be used in the construction of a Log Record. See [XML-SCH-2] and [DCMLTypes] for the definitions of the underlying XML data types.

6.1 Record Text Extension Type

The RecordTextExtension type is meant to contain human readable text and optionally positional parameter replacement codes. The purpose of this data type is to provide for human readable interpretations of record content, in multiple languages where appropriate.

Requesting a set of these description items for a particular language can be done at any time that log records need to be consolidated or interpreted. An XML element based on this type must include one and only one recordTextID element, but may contain more than one recordDescriptionText element in order to express the content in more than one language.

6.1.1 Record Text Extension Fields

The RecordTextExtension Type shall consist of the following elements.

6.1.1.1 Record Text ID

The RecordTextID element shall be a UUID that represents the conceptual information expressed in the RecordDescriptionText field, regardless of the language used to express it.

6.1.1.2 Record Description Text

The RecordDescriptionText element shall be a block of UTF-8 encoded text that describes the meaning of the record in human readable form. The record text may include positional replacement markers in place of which values from the Parameters List of the record may be substituted.

6.1.2 Schema Representation

Expressed in XSDL, the RecordTextExtension Type shall be defined as follows:

```
<xs:complexType name="recordTextExtensionType">
   <xs:sequence>
      <xs:element name="RecordTextID" type="dcml:UUIDType" minOccurs="1"/>
      <xs:element name="RecordDescriptionText" type="dcml:UserTextType"
minOccurs="1" maxOccurs="unbounded"/>
   </xs:sequence>
</xs:complexType>
```

6.2 Event Token Types

In the header and body sections of Log Records, XML "tokens" are used to represent the values of the record fields. The following XML type extends the XML "token" type to include optional scope attributes in order to specify the allowable values for those fields.

6.2.1 Event Type Type

The Event Type Type shall be used to specify types and subtypes of Log Events in Log records. This Type shall be defined in XSDL as follows:

```
<xs:complexType name="eventTypeType">
   <xs:simpleContent>
      <xs:extension base="xs:token">
         <xs:attribute name="scope" type="xs:anyURI" use="required"/>
      </xs:extension>
   </xs:simpleContent>
</xs:complexType>
```

Note that the "scope" attribute is required but not specified, and no default scope is provided. For each application specification that uses this Log Record Format Specification, that specification document must specify one or more scope URIs for this type, and specify the list of allowed tokens defined by that scope.

6.3 Reference ID Types

In the body section of Log Records, it is frequently necessary to identify an entity both by name and by a UUID. The following Types define the framework to be used in expressing these name/value pairs.

6.3.1 Reference ID Type

The Reference ID Type defines a name/value pair which is scoped to indicate the range of possible names for the identifiers. This Type shall be defined in XSDL as follows:

```
<xs:complexType name="ReferenceIDType">
        <xs:sequence>
                <xs:element name="IDName" type="dcml:scopedTokenType"/>
                <xs:element name="IDValue" type="dcml:UUIDType"/>
        </xs:sequence>
</xs:complexType>
```

Reference ID scopes and scope URIs may be defined by other documents as needed, but the specific definitions are outside the scope of this document, as they as are application-specific.

6.3.2 Referenced ID List Type

In the body section of a Log Record, elements of type Reference ID shall be grouped in a list. The list Type shall be defined in XSDL as follows:

```
<xs:complexType name="ReferencedIDListType">
        <xs:sequence>
                <xs:element name="ReferencedID" type="lr:ReferenceIDType"
minOccurs="1" maxOccurs="unbounded"/>
        </xs:sequence>
</xs:complexType>
```

7 Log Record Type Descriptions

A Log Record shall always consist of at least a Log Record Header and a Log Record Body. A Log Record may also contain a Log Signature. This section defines the XML Types that represent Log Record Headers, Bodies, and Signatures.

7.1 Log Record Header Type

This type provides a structural format for the fields, content types and references for a Log Record Header. The Log Event Header shall include the fields described below except where they are optional for the application. Fields marked Optional may be required by other constraints documents for certain applications.

7.1.1 Event ID

The EventID element is a UUID that identifies a specific Log Record. Since Log Record Headers may be processed independently from Log Record Bodies, an identical copy of this UUID shall appear in the Log Record Body to associate the header and body if they become separated. If unique identification of a logged event is required, the contents of the EventID element shall be used for that purpose.

7.1.2 Time Stamp

The Time Stamp element shall indicate the time and date that the event was detected. The Time Stamp shall be represented as an XML "dateTime" type. The Timezone section of the dateTime item shall be present, and shall be the local time zone of the location at which the reporting equipment is installed. [Note: This requires that relative time stamps must be converted to absolute local time at or before the time that the formatted Log Record is generated.]

7.1.3 Event Sequence (optional)

When a Log Report is generated by a device, the device shall set the EventSequence element to a value of its choice for the first Log Record in the sequence, and shall increment the EventSequence value by one for each subsequent record in the sequence. EventSequence numbers shall not be interpreted as uniquely identifying an event, but shall only designate the sequence number of the event in the current sequence, if the record is part of a sequence.

Note: The EventSequence values, used in concert with the chained header hashes, can be used to check that the sequence of Log Headers in a Log Report is complete.

7.1.4 Device Source ID

The Device Source ID shall be an element of type device Identifier List Type (defined in [DCMLTypes]) that identifies the device that generated the event that the Log Record represents. This list shall contain at least one device identifier that identifies the source of the Log Event as recorded in the instant Log Record.

7.1.5 Event Class

The EventClass element shall contain a URI identifying the class of the logged event. The EventClass is intended to express the general classification of the Log Event represented by the Log Record, and to identify the scope or namespace defining the Event Types and SubTypes in that class. This class may be used to facilitate searching and sorting on groups or sequences of Log Records. A particular event may be defined within more than one class, and the same event may be reported under more than one class if required by the application. The EventClass shall be represented in XML using the xs:anyURI type. .The definition of specific Event Classes is outside the scope of this document. Examples of possible classes are listed in the following table.

Example Event Classes (Informative)	
Class	**Class Description**
Security	Event records related to security
Operational	Operational events
Health	Status or change of status of system health items
Maintenance	Maintenance events
Debug	Debugging event records

7.1.6 Event Type

Within an Event Class, this field indicates the type of event that is logged. The event type shall be represented by an Event Type Type. The scope attribute is required, and must be specified by another document that defines Log Records for a particular Class.

Where Event Types are standardized or are published in separate documents, the scope attribute URI of the Event Type shall be defined and shall indicate the allowable values of this field for that class of events.

Informative Note: This document does not specify any EventType scopes. Not all Classes of Events will have all of the possible event types published - for example, Debug Class events will likely be vendor-specific. However, it is required that Event Types for all other classes will be standardized in one or more separate documents. The following table gives examples of possible EventType tokens that might be defined for an Event Class in a separate document.

Example Event Type Tokens (Informative)	
Token	**Description**
power	a power event
timeline	a timeline event
network	a network event
Test	a test sequence has completed.
Playout	an instance of content has been played.

7.1.7 Content Id (optional)

This optional element may be present whenever a Key Delivery Message ID or Composition ID is present in the associated Log Record Body. When this element is present, the EventSequence element of the Log Record Header shall also be present. The Content ID shall be the Composition ID if that element is present in the Body, or the Key Delivery Message ID if the Composition ID is not present.

7.1.8 Previous Header Hash (optional)

If present, this element shall contain the SHA-1 digest of the canonicalized Header section of the previous Log Record in a sequence of log records. Canonicalization shall be performed according to [xml-c14n].

7.1.9 Record Body Hash (optional)

If present, this element shall contain the SHA-1 digest of the canonicalized Log Record Body portion of the event recorded in this log record. Canonicalization shall be performed according to [xml-c14n].

7.1.10 Schema Representation

In XSDL, the Log Record Header shall be represented as follows:

```
<xs:complexType name="logRecordHeaderType">
   <xs:sequence>
      <xs:element name="EventID" type="dcml:UUIDType"/>
      <xs:element name="TimeStamp" type="xs:dateTime"/>
      <xs:element name="EventSequence" type="xs:long" minOccurs="0"/>
      <xs:element name="DeviceSourceID" type="dcml:deviceIdentifierListType"/>
      <xs:element name="EventClass" type="xs:anyURI"/>
      <xs:element name="EventType" type="lr:eventTypeType"/>
      <xs:element name="ContentId" type="dcml:UUIDType" minOccurs="0"/>
      <xs:element name="PreviousHeaderHash" type="ds:DigestValueType"
minOccurs="0"/>
      <xs:element name="RecordBodyHash" type="ds:DigestValueType"
minOccurs="0"/>
   </xs:sequence>
</xs:complexType>
```

7.2 Log Record Body Type

The Log record Body Type provides a means of conveying the detailed record content and description of an individual log record.

7.2.1 Event ID

The value of the Event ID shall be a copy of the Event ID value from the associated Log Record Header.

7.2.2 Event Sub Type (optional)

This optional field provides additional detail about the event type. An Event Sub Type shall be recorded using the same XML Type as the Event Type, except that the scope attribute may be either the same or a different value than the Event Type. Event Sub Type scopes and scope URIs may be defined by other documents as needed, but the specific definitions are outside the scope of this document, as they as are application-specific.

Example Event Sub Type Tokens (Informative)	
Token	**Description**
on	something was turned on
off	something was turned off
open	Opened
close	Closed
error	an error was detected
nominal	an error was not detected

7.2.3 Parameters (optional)

The Parameters element shall be a list of typed numeric or text parameters, associated with identifying tags in each list element. If the Parameters element is present, the list shall contain at least one Parameter.

If a Record Extension ID or a Record Text Extension is provided in the Log Record Body, the parameters in the Parameters element shall be sequenced to match the order of any positional parameter codes included in the Record Description Text element of the Record Text Extension.

The Parameters element, if present, shall be represented using the dcml:parameterListType, which shall contain one or more parameters represented using the dcml:namedParmType. The dcml:namedParmType expresses name/value pairs, with the value member constructed to represent any type.

7.2.4 Exceptions (optional)

The Exceptions element shall be a list of typed numeric or text parameters, associated with identifying tags in each list element. If the Exceptions element is present, the list shall contain at least one Parameter.
The Exceptions element, if present, shall be represented using the dcml:parameterListType, which shall contain one or more parameters represented using the dcml:namedParmType. The dcml:namedParmType expresses name/value pairs, with the value member constructed to represent any type.

Informative Note: While the Parameters and Exceptions elements of the Log Record Body are identical except for the element name, the intended usage is different. The Exceptions element should be used to list unusual or erroneous results associated with an Event, while the Parameters element is intended to provide additional detail about an Event that is not exceptional.

7.2.5 Referenced IDs (optional)

If present, the Referenced IDs element shall be represented using the Referenced ID List Type type. At least one Referenced ID element must be present in this list element, which must be of the Reference ID Type type.

Informative Note: This document does not specify any ID Name scopes for use in the ReferencedID elements. However, it is required that ID Name scopes for these elements will be standardized in one or more separate documents. An example scope might include KDM IDs, CPL IDs, etc.

7.2.6 Record Extension ID (optional)

A Log Record Body may optionally include a Record Extension ID, which refers to a specific Record Description Text item. The same record code value (UUID) may refer to multiple Record Text Extensions, expressed in multiple languages.

7.2.7 Record Text Extension (optional)

In addition to or instead of a Record Extension ID, the logging device may directly include a Record Text Extension structure in the body of the record. The Record Text Extension shall be represented using the Record Text Extension Type defined in Section 6.1 of this document. Positional parameter codes, if present in the record Text Extension value, shall take the form of a UTF-8 percent sign (character code 0x25) followed by one or more digits indicating the position of the replacement value in the Parameters list element. A double percent sign (%%) shall represent a single percent(%) character. Enumeration of parameters shall be counted starting from one, so a positional parameter of zero is meaningless.

7.2.8 Device Description (optional)

The Log Record Body may optionally contain a field of type device Description Type (defined in [DCMLTypes]) that describes the device in some detail. A Logging Device should include this field at least once in a Log Report when a device is first mentioned in a sequence of Log Records. It is unnecessary to repeat this information each time a device is identified in a Log Report or other group of Log Records.

7.2.8.1 Device Type Indication

Any of the device type tokens (e.g. as specified in [DCMLTypes]) may be specified in either the Device Type ID or the Device Subsystem Type ID fields of the Device Description. If the DeviceDescription element is included in the Log Record Body, at least the Device Type ID element shall be included. Some combinations of device type and device subsystem type may not make sense, but the intent is that if a subsystem type is provided, that subsystem is the source of the event, and that one or more of the following conditions hold true:

- The device identified in the Device Type ID contains the device identified in the Device Subsystem Type ID, and is reporting the event on behalf of the subsystem device.
- The device identified in the Device Type ID is proxying log messages for the device identified in the Device Subsystem Type ID, and is reporting the event on behalf of the subsystem device.

In either case, if a Device Subsystem Type ID is reported, that device shall be the original source of the event which the Log record records.

7.2.9 Schema Representation

The Record Body shall be represented in XSDL as follows:

```
<xs:complexType name="logRecordBodyType">
   <xs:sequence>
      <xs:element name="EventID" type="dcml:UUIDType"/>
      <xs:element name="EventSubType" type="lr:eventTypeType" minOccurs="0"/>
      <xs:element name="Parameters" type="dcml:parameterListType" minOccurs="0"
maxOccurs="1"/>
      <xs:element name="Exceptions" type="dcml:parameterListType" minOccurs="0"
maxOccurs="1"/>
      <xs:element name="ReferencedIDs" type="lr:ReferencedIDListType"
minOccurs="0"/>
      <xs:element name="RecordExtensionID" type="dcml:UUIDType" minOccurs="0"/>
      <xs:element name="RecordTextExtension" type="lr:recordTextExtensionType"
minOccurs="0"/>
      <xs:element name="DeviceDescription" type="dcml:deviceDescriptionType"
minOccurs="0"/>
   </xs:sequence>
</xs:complexType>
```

7.3 Log Record Signature Type

If a device is required to generate authenticated Log Records or an authenticated sequence of Log Records (a Log Report), an element of type record Signature Type shall be included (as specified below) to perform that function. Support for this element depends on the presence and proper use of the EventSequence, PreviousHeaderHash, and RecordBodyHash elements of the Log Record Header, so those elements shall be required in the Log Record Header of every authenticated Log Record, and every Log Record in an authenticated Log Report.

This element may contain one of two possible structures depending on whether it appears at the beginning or end of a sequence of Log Records in a Log Report. Appearing as part of the first record in a sequence, its presence indicates that the current log record is the beginning of a sequence of log records to be authenticated as a group.

When a Log Signature appears as part of a Log Record at the end of a sequence of Log Records, it shall contain an end of sequence marker in the HeaderPlacement element and a SHA-1 digest over the header of the record in which it appears. If a Log Record or Report is to be signed, the Log Signature shall appear at the end of an authenticated Log Record or sequence of Log Records, and shall be optional at the beginning of a sequence.

Note: Providing a Log Signature at the beginning of a sequence may facilitate forward authentication processing and random access to sequence start and end points in some applications.

7.3.1 Header Placement

The HeaderPlacement element shall indicate either the beginning of a sequence, the end of a sequence, or both if the sequence is of length one.

Note: A Log Report may contain multiple authenticated sequences of Log Records.

7.3.2 Sequence Length

The SequenceLength element shall contain the length (in Records) of the current sequence of Log Records.

7.3.3 Record Auth Data (Optional)

The RecordAuthData element shall be an instance of the signed Record Auth Data Type, which shall contain the following elements defined in this section. The RecordAuthData element of the record Signature Type shall be optional.

7.3.3.1 Record Header Hash

This element of the signed Record Auth Data Type Type shall contain the SHA-1 digest over the canonicalized Log Record header of the current Log Record, that record of which it is a member. Canonicalization shall be performed according to [xml-c14n].

7.3.3.2 Signer Cert Info

The SignerCertInfo element of the signedRecordAuthDataType type shall be a compound element of type ds:X509IssuerSerialType. The X509IssuerName element of the SignerCertInfo element shall contain a copy of the X.509 Distinguished Name from the device certificate of the device that authenticates the Log Record or Log Report, and that distinguished name shall be compliant with [RFC 2253]. The X509SerialNumber element of the SignerCertInfo element shall contain the X.509 issuer serial number from the device certificate of the device that authenticates the Log Record or Log Report.

7.3.4 Signature (Optional)

This element shall be an XML Signature constructed per [xml-dsig], with the contained Digest Value element containing the SHA-1 digest over the canonicalized RecordAuthData element of the Log Record Signature. Canonicalization shall be performed according to [xml-c14n]. If present, this optional element shall appear in the LogRecordSignature element at the end of an authenticated Log Record or sequence of Log Records.
If present, this Signature element shall contain a KeyInfo element, which shall contain the device certificate of the signing device and its certificate chain.

7.3.5 Schema Representation

In XSDL, the Log Signature Header shall be represented using the record Signature Type as defined in XSDL below. The record Signature Type requires the definition of the placement Token, and the signed Record Auth Data Type, which shall be defined as expressed in XSDL below:

```
<xs:simpleType name="placementToken">
   <xs:restriction base="xs:token">
      <xs:enumeration value="start"/>
      <xs:enumeration value="stop"/>
  </xs:restriction>
</xs:simpleType>

<xs:complexType name="signedRecordAuthDataType">
   <xs:sequence>
      <xs:element name="RecordHeaderHash" type="ds:DigestValueType"/>
      <xs:element name="SignerCertInfo" type="ds:X509IssuerSerialType"/>
   </xs:sequence>
   <xs:attribute name="Id" type="xs:anyURI" use="required"/>
</xs:complexType>
```

```
<xs:complexType name="logRecordSignatureType">
   <xs:sequence>
      <xs:element name="HeaderPlacement" type="lr:placementToken"/>
      <xs:element name="SequenceLength" type="xs:integer"/>
      <xs:element name="RecordAuthData" type="lr:signedRecordAuthDataType"
minOccurs="0"/>
<xs:element ref="ds:Signature" minOccurs="0"/>
   </xs:sequence>
</xs:complexType>
```

7.4 Log Record and Report Types

7.4.1 Log Record Type

A Log Record shall be composed of a Log Record Header at minimum, and may optionally include a Log Record Body and a Log Record Signature. A Log Record shall be defined in XSDL as follows:

```
<xs:complexType name="logRecordType">
    <xs:sequence>
<xs:element name="LogRecordHeader" type="lr:logRecordHeaderType" minOccurs="1"/>
        <xs:element name="LogRecordBody" type="lr:logRecordBodyType"
minOccurs="0"/>
        <xs:element name="LogRecordSignature" type="lr:logRecordSignatureType"
minOccurs="0"/>
    </xs:sequence>
</xs:complexType>
```

7.4.2 Log Report Type

A Log report shall consist of one or more Log Records assembled into a Log Report. If the optional reportDate element is present, it shall contain the date and time at which the report was generated. If the optional reportingDevice element is present, it shall describe the device that produced the Log Report. A Log Report shall be defined in XSDL as follows:

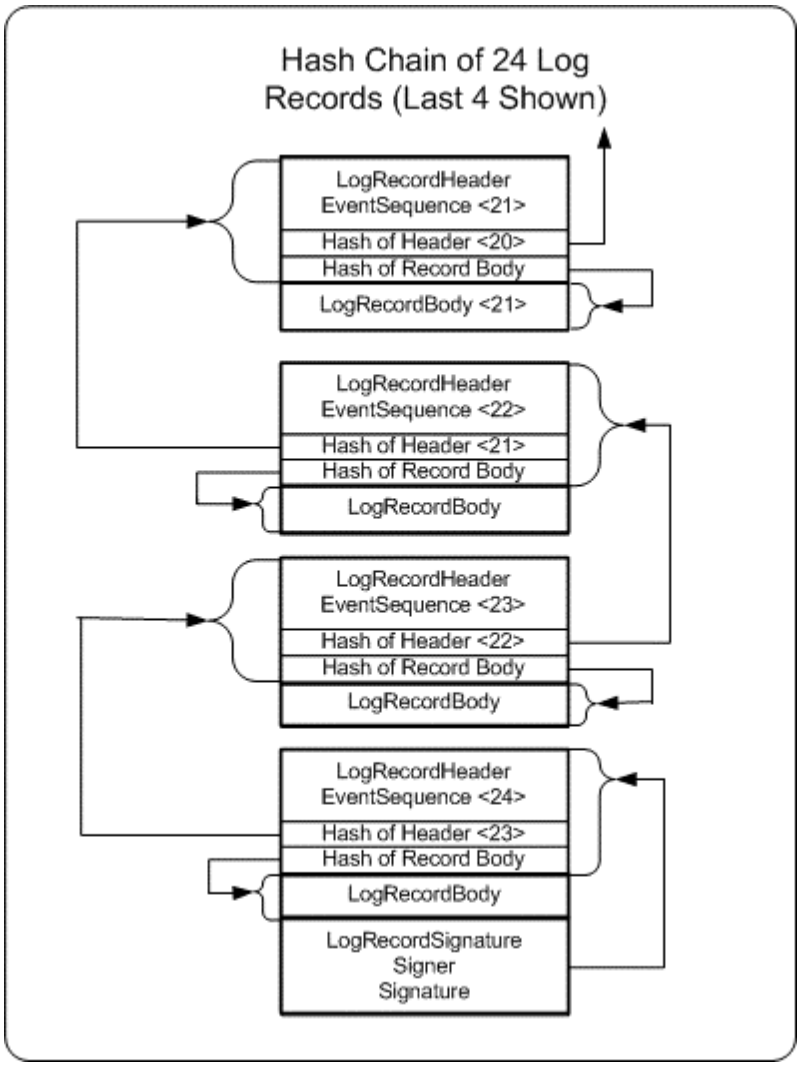
```
<xs:complexType name="logReportType">
   <xs:sequence>
      <xs:element name=”reportDate” type=”xs:dateTime” minOccurs=”0”/>
      <xs:element name=”reportingDevice” type=”dcml:deviceDescriptionType”
minOccurs=”0”/>
      <xs:sequence>
         <xs:element name="LogRecordElement" type="lr:logRecordType"
maxOccurs="unbounded"/>
      </xs:sequence>
   </xs:sequence>
</xs:complexType>
```

8 Schema (Informative)

Note: The Informative Schema has been removed from this document to facilitate maintenance and processing with tools suitable to the purpose. If you are reviewing this document, you should have been provided with an up to date version of the informative schema in electronic form. The name of the file should be "LogRecord.xsd".

9 Log Record Chaining (Informative)

The following diagram illustrates the relationship between Log Record components assembled into an authenticated chain.



Authentication of a sequence may be performed by verifying the RecordHeaderHash element contained in the Log Record Signature on the header of the Log Record that contains that signature, then walking backward through the sequence, verifying each previous Log Header with the Previous Header Hash of the current Log Header. After each header is validated, that header's Record Body Hash field may be used to validate the associated Log Record Body, if present.

10 Example Instances (Informative)

10.1 Example Log Record

The following example illustrates a Log Record, including all of the required fields and some of the optional fields.

Note: This Informative Example has been removed from this document to facilitate maintenance and processing with tools suitable to the purpose. If you are reviewing this document, you should have been provided with an up to date version of the informative example in electronic form. The name of the file should be "SingleLogRecordExample.xml".

10.2 Example Log Report

The following XML document is an example of a simple Log Report of two records, with many of the optional elements omitted for clarity.

Note: This Informative Example has been removed from this document to facilitate maintenance and processing with tools suitable to the purpose. If you are reviewing this document, you should have been provided with an up to date version of the informative example in electronic form. The name of the file should be "LogReportExample.xml".

10.3 Example Authenticated Log Report

The following example illustrates a Log Report with authentication features added. Note that the hashes are not computed correctly, and the final signature is not shown for reasons of practicality and space.

Note: This Informative Example has been removed from this document to facilitate maintenance and processing with tools suitable to the purpose. If you are reviewing this document, you should have been provided with an up to date version of the informative example in electronic form. The name of the file should be "AuthLogReportExampleSigned.xml".

11 Glossary of Acronyms

- CPL – Composition Play List
- KDM – Key Delivery Message
- SE – Security Entity
- SHA-1 – Secure Hash Algorithm revision 1
- SM – Security Manager
- SPB – Secure Processing Block
- TLS – Transport Layer Security protocol.