

SCTE • ISBE[®]

S T A N D A R D S

Digital Video Subcommittee

AMERICAN NATIONAL STANDARD

ANSI/SCTE 67 2017

**Recommended Practice for
Digital Program Insertion for Cable**

NOTICE

The Society of Cable Telecommunications Engineers (SCTE) Standards and Operational Practices (hereafter called “documents”) are intended to serve the public interest by providing specifications, test methods and procedures that promote uniformity of product, interchangeability, best practices and ultimately the long term reliability of broadband communications facilities. These documents shall not in any way preclude any member or non-member of SCTE from manufacturing or selling products not conforming to such documents, nor shall the existence of such standards preclude their voluntary use by those other than SCTE members.

SCTE assumes no obligations or liability whatsoever to any party who may adopt the documents. Such adopting party assumes all risks associated with adoption of these documents, and accepts full responsibility for any damage and/or claims arising from the adoption of such documents.

Attention is called to the possibility that implementation of this document may require the use of subject matter covered by patent rights. By publication of this document, no position is taken with respect to the existence or validity of any patent rights in connection therewith. If a patent holder has filed a statement of willingness to grant a license under these rights on reasonable and nondiscriminatory terms and conditions to applicants desiring to obtain such a license, then details may be obtained from the standards developer. SCTE shall not be responsible for identifying patents for which a license may be required or for conducting inquiries into the legal validity or scope of those patents that are brought to its attention.

Patent holders who believe that they hold patents which are essential to the implementation of this document have been requested to provide information about those patents and any related licensing terms and conditions. Any such declarations made before or after publication of this document are available on the SCTE web site at <http://www.scte.org>.

All Rights Reserved

© Society of Cable Telecommunications Engineers, Inc. 2017
140 Philips Road
Exton, PA 19341

Table of Contents

Title	Page Number
Table of Contents	2
1. Introduction	6
1.1. Executive Summary	6
1.2. Scope	6
1.3. Benefits	6
1.3.1. Improvements in Ad Timing Synchronization	6
2. Informative References	6
2.1. SCTE References	6
2.2. Standards from Other Organizations	7
2.3. Published Materials	7
3. Compliance Notation	8
4. Abbreviations and Definitions	8
4.1. Abbreviations	8
4.2. Definitions	9
5. Overview	10
5.1. Example SCTE 35 Decoder	12
6. Application Guidelines	12
6.1. Practical Boundaries for splice_time() in splice_insert()	12
6.2. System Delays	13
6.2.1. SCTE 104 Insertion Delays	13
6.2.2. Encoder Delays	13
6.2.3. Transmission Delays	13
6.2.4. Splicer/Multiplexor Delays	13
6.3. Splice Time Accuracy	13
6.3.1. Splicer Timing	15
6.4. Splice_event_id Usage and Uniqueness	15
6.5. Use of splice_schedule() Command	17
6.6. Component Splice Mode	17
6.6.1. Erroneous Component Splice Commands	18
6.7. Pre-Roll Functionality - Accomplishing a Pre-Roll Function	18
6.8. Conditional Access and Cue Encryption	19
6.8.1. What to Encrypt	19
6.8.2. Operation in a Cue Insertion Device	19
6.8.3. Operation in an Ad Insertion Device	19
6.8.4. Theory of Operation	20
7. Usage of fields in the Splice_Insert command	24
7.1. Usage of unique_program_id	24
7.1.1. What is a "Program"?	25
7.1.2. What is a "program_id"?	25
7.1.3. Why Should Programs Be Identified and Differentiated?	25
7.1.4. Why Does the Time at Which a Program Is Scheduled Not Identify It?	25
7.1.5. How Will a unique_program_id Alleviate Problems?	25
7.2. Avail Fields Usage	26
7.2.1. What is an Avail?	26
7.2.2. How Many Avails Occur Within a Program?	26
7.2.3. Why is it important to identify the Avails Within a Program?	26
7.2.4. How Do the Avail Fields Provide for This?	27
7.2.5. What Does the avails_expected Field Do?	27
7.2.6. Conditional Avails	27
8. Cue Usage	29
8.1. Starting a Break	29
8.2. Ending a Break	29

8.3.	Spot Sharing Within a Break	30
9.	Creation and Usage of Splice Descriptors	30
9.1.	What are Descriptors	30
9.1.1.	The Problem	31
9.1.2.	The Solution	31
9.2.	Registration	31
9.3.	Creating Compatible Private Descriptors	31
9.4.	Using the avail_descriptor.	32
9.5.	Using the DTMF Descriptor	33
9.5.1.	Pre-roll Timing	33
9.5.2.	DTMF Tone Sequence	34
9.5.3.	SCTE 35 Operating Modes	34
9.6.	Usage of Segmentation Descriptors	34
9.6.1.	Segmentation Descriptor Field Usage	34
9.6.2.	Delivery of Segmentation Descriptors	36
9.6.3.	Processing of Segmentation Descriptors	36
9.6.4.	Specific Use Cases	37
9.6.5.	Identifying Placement Opportunities	38
9.6.6.	Identifying Standalone Advertisements	39
10.	Presentation Time Stamp considerations	42
10.1.	Handling Time Base Discontinuities	42
10.2.	Cascaded Splicing Devices	43
10.2.1.	Restamping Cue Messages	43
10.2.2.	Cue Propagation	44
10.2.3.	Delay	44
10.2.4.	Logical Cascading	44
11.	Command Usage	44
11.1.	Bandwidth Reservation Command	44
11.1.1.	Why Use a Bandwidth Reservation Command?	44
11.2.	Heartbeat Messages	45
11.2.1.	Why Use a Heartbeat Message	45
11.3.	Time Signal Command	45
11.3.1.	Uses for the Time Signal Command	45
11.3.2.	Practical Boundaries for splice_time() in time_signal()	45
12.	Implementing SCTE 35 for Signaling in Linear Content	46
12.1.	System Architecture - Provider	46
12.1.1.	Rights	47
12.1.2.	Programming	47
12.1.3.	Sports	47
12.1.4.	Ads	47
12.1.5.	Scheduling	47
12.1.6.	Automation	48
12.1.7.	Master Control	48
12.1.8.	Live/Servers	48
12.1.9.	SCTE 104	48
12.1.10.	Slate/Alternate Switch/Generator	48
12.1.11.	Encoder	48
12.1.12.	Packager	48
12.1.13.	Origin	49
12.1.14.	CDN	49
12.1.15.	Distributor	49
12.1.16.	Metadata Generation	49
12.1.17.	App Server	49
12.1.18.	Clients	49
12.2.	System Architecture - Distributor	49
12.2.1.	Broadcast/Mezzanine	50

12.2.2.	QAM/Satellite	50
12.2.3.	VOD/nPVR	50
12.2.4.	Transcoder	50
12.2.5.	Packager	51
12.2.6.	Origin	51
12.2.7.	Metadata Reader	51
12.2.8.	Player/App Server	51
12.2.9.	MVPD Distribution	51
12.3.	Extensions to Content Identification for Real Time Signaling	51
12.3.1.	SCTE 35 Guidelines	51
12.3.2.	USAGE of SCTE 35 Restriction Bits	52
12.3.3.	Web Restriction Use Cases	52
12.3.4.	Regional Blackout Use Cases	56
12.3.5.	Alternate Content	57
12.3.6.	Alternate Content Removal	58
12.3.7.	Archive Use Cases	58
12.3.8.	Device Restriction Use Cases	58
13.	Recommendations on carrying SCTE 35 in other than MPEG2 Transport Streams	58
13.1.	General Comments on Transforming SCTE 35	58
13.1.1.	Time Base Conversions	58
13.1.2.	Ad Signaling for RTMP	60
13.1.3.	AD Signaling for Microsoft Smooth Streaming	61
13.1.4.	AD Signaling for MPEG-DASH	61
13.1.5.	Ad Signaling for HLS	64
14.	Additional Information	67
14.1.	Considerations for Evaluation of MPEG-2 Splicing Devices	67
14.2.	Overview	67
14.2.1.	Splicer Technology	67
14.2.2.	Environment	68
14.2.3.	Splicer Performance	71
14.3.	Ad Timing Recommendations	74
14.3.1.	Provider Issues	74
14.3.2.	Distributor Monitoring	75
14.3.3.	Cue Timing Monitoring Points	76

List of Figures

Title	Page Number
Figure 1 - System Overview	11
Figure 2 - Cue Message Insertion Points	16
Figure 3 - DES ECB Example	22
Figure 4 - DES CBC Encryption Example	22
Figure 5 - DES CBC Decryption Example	22
Figure 6 - Triple-DES ECB Encryption Example	23
Figure 7 - Triple-DES ECB Decryption Example	24
Figure 8 - Cascading of Splicer / Server Devices	43
Figure 9 - System Architecture	46
Figure 10 - Example Distributor Architecture	50
Figure 11 - Cue Timing Monitoring Points	76

List of Tables

Title	Page Number
Table 1 - Avail incrementing/skipping Example	28
Table 2 - (of SCTE 35): splice_descriptor()	32

1. Introduction

1.1. Executive Summary

This Recommended Practice is to serve as an informational enhancement to SCTE 35, Digital Program Insertion Cue Message for Cable. SCTE 35 is necessarily brief in many areas in order to maintain conciseness and accuracy. This document serves as a companion to SCTE 35.

1.2. Scope

This document is an informational companion to SCTE 35. It is not in itself a specification or a standard. The information within is intended as guideline information. Where this document contradicts SCTE 35, SCTE 35 takes precedence.

1.3. Benefits

The purpose of this document is to aid splicing equipment designers, ad insertion equipment designers as well as the purchasers and users of such equipment, such as the networks that will originate Cue Messages from their uplink sites. This document is also expected to aid in the system integration of advertising related equipment, both at the Message origination end and at the Message reception end.

SCTE 35 includes content segmentation messages, and this document has been updated to aid the users of these messages. Some of the new devices that will be interpreting the SCTE 35 commands include transcoders, packagers, network PVR and other content manipulation, storage and streaming delivery systems.

There may be crucial information within this document for manufacturers of equipment that pass the Cue Message as part of the MPEG-2 transport stream. An example of such equipment is a rate altering re-multiplexer, which performs complex processing of the stream. When the stream is demultiplexed and processed and then re-multiplexed, it is very important to place the Cue Message in the proper position relative to the video service and relative to nearby time-base discontinuities. Such equipment may also be required to alter the Message before retransmission

1.3.1. Improvements in Ad Timing Synchronization

This document makes recommendations on how to maintain proper time synchronization between servers and splicers. These recommendations include the time synchronization method used between servers and splicers; methods for configuring splicer controlled Ad Insertion Systems; and methods for monitoring the proper timing of Cue Messages.

2. Informative References

The following documents might provide valuable information to the reader but are not required when complying with this document.

2.1. SCTE References

- [SCTE 35] ANSI/SCTE 35 2016 - Digital Program Insertion Cueing Message for Cable.
- [SCTE 30] ANSI/SCTE 30 2015 - Digital Program Insertion Splicing API.
- [SCTE 40] ANSI/SCTE 40 2016 - Digital Cable Network Interface Standard.

- [SCTE 54] ANSI/SCTE 54 2015 - Digital Video Service Multiplex and Transport System Standard for Cable Television.
- [SCTE 104] ANSI/SCTE 104 2015 – Automation System to Compression System Communications Applications Program Interface (API).
- [SCTE 118-1] ANSI/SCTE 118-1 2012 – Program-Specific Ad Insertion - Data Field Definitions, Functional Overview and Application Guidelines.
- [SCTE 118-2] ANSI/SCTE 118-2 2012 – Program-Specific Ad Insertion – Content Provider to Traffic Communication Applications Data Model.
- [SCTE 118-3] ANSI/SCTE 118-3 2012 – Program-Specific Ad Insertion – Traffic System to Ad Insertion System File Format Specification.
- [SCTE 130] ANSI/SCTE 130-1 2013 – Digital Program Insertion – Advertising Systems Interfaces Part 1 – Advertising Systems Overview (Informative).
- [SCTE 172] ANSI/SCTE 172 2011 – Constraints on AVC video coding for digital program insertion.
- [SCTE 224] ANSI/SCTE 224 2015 – Event Scheduling and Notification Interface
- [DVS 839] DVS 839 2014 – SCTE Interfaces for Advanced Advertising – Version 29 15-Jul-2014.

2.2. Standards from Other Organizations

- [13818-1] ITU-T Rec. H.222.0 / ISO/IEC 13818-1 2013 - Information technology – Generic coding of moving pictures and associated audio information: systems
- [13818-2] ITU-T Rec. H.262.0 / ISO/IEC 13818-2 2012 - Information technology – Generic coding of moving pictures and associated audio information: video
- [AVC] ITU-T Rec. [13818-4] ISO/IEC 13818-4 2004 - Information technology — Generic coding of moving pictures and associated audio information — Part 4: Conformance testing
- [312] SMPTE ST 312:2001- Splice Points for MPEG-2 Transport Streams
- [AMF0] "Action Message Format AMF0",
http://download.macromedia.com/pub/labs/amf/amf0_spec_121207.pdf
- [BFX] SMPTE ST 2021-1:2013 - Broadcast Exchange Format
- [DASH] ISO/IEC 23009-1 2013 - Information technology - Dynamic adaptive streaming over HTTP (DASH) - Part 1: Media presentation description and segment formats, 2nd edition
- [EIDR] Entertainment Identification Registry
http://eidr.org/documents/EIDR_2.0_Best_Practices.pdf
- [EIDRCB] EIDR ID Format – “EIDR: ID FORMAT Ver. .2”,
http://eidr.org/documents/EIDR_ID_Format_v1.2.pdf
- [ESAM] OC-SP-ESAM-API- C01 161021 Real-time Event Signaling and Management API
- [ISOBMFF] ISO/IEC 14496-12 2012 - Information technology - Coding of audio-visual objects - Part 12: ISO base media file format, 4th edition
- [OATC CF] OATC Content Feed 2.0 www.oatc.us
- [HLS] “HTTP Live Streaming”, <https://tools.ietf.org/html/draft-pantos-http-live-streaming-20>
- [RFC4648] “The Base16, Base32, and Base64 Data Encodings”
- [XML10] "Extensible Markup Language (XML) 1.0, Fifth Edition",
<http://www.w3.org/TR/REC-xml/>
- [XMLS2] "XML Schema Part 2 Data Type, Second Edition", <http://www.w3.org/TR/xmlschema-2/>

2.3. Published Materials

- No informative references are applicable.

3. Compliance Notation

<i>shall</i>	This word or the adjective “ required ” means that the item is an absolute requirement of this document.
<i>shall not</i>	This phrase means that the item is an absolute prohibition of this document.
<i>forbidden</i>	This word means the value specified shall never be used.
<i>should</i>	This word or the adjective “ <i>recommended</i> ” means that there may exist valid reasons in particular circumstances to ignore this item, but the full implications should be understood and the case carefully weighted before choosing a different course.
<i>should not</i>	This phrase means that there may exist valid reasons in particular circumstances when the listed behavior is acceptable or even useful, but the full implications should be understood and the case carefully weighed before implementing any behavior described with this label.
<i>may</i>	This word or the adjective “ <i>optional</i> ” means that this item is truly optional. One vendor may choose to include the item because a particular marketplace requires it or because it enhances the product, for example; another vendor may omit the same item.
<i>deprecated</i>	Use is permissible for legacy purposes only. Deprecated features may be removed from future versions of this document. Implementations should avoid use of deprecated features.

4. Abbreviations and Definitions

4.1. Abbreviations

ATSC	Advanced Television Systems Committee, www.atsc.org
AVC	Advanced Video Coding AVC
CBC	Cipher Block Chaining
CBR	Constant Bit Rate
CDN	Content Delivery Network
CRC	Cyclic Redundancy Check
DASH	Dynamic Adaptive Streaming over HTTP [DASH]
DES	Data Encryption Standard
DTMF	Dual Tone Multi Frequency
DTS	Decoding Time Stamp [13818-1]
DVB	Digital Video Broadcasting
ECB	Electronic Code Book
ECM	Entitlement Control Message
EMM	Entitlement Management Message
GOP	Group of Pictures
HD-SDI	High-Definition Serial Digital Interface
IDR	Instantaneous Decoder Refresh
MPEG	Moving Picture Experts Group
MPD	Media Presentation Description
MPTS	Multi Program Transport Stream.
MVPD	Multichannel Video Programming Distributor
PES	Packetized Elementary Stream [13818-1]

PID	Packet Identifier
PMT	Program Map Table [13818-1]
PTS	Presentation Time Stamp [13818-1]
SCTE	Society of Cable Telecommunications Engineers Inc., www.scte.org
SDI	Serial Digital Interface
SPTS	Single Program Transport Stream
T-STD	Transport Stream System Target Decoder
TVE	TV Everywhere
uimsbf	Unsigned integer, most significant bit first
VANC	Vertical Ancillary Data
VBR	Variable Bit Rate

4.2. Definitions

Access Unit	The coded representation of a video picture or an audio frame [13818-1]
Ad Insertion System	A combination of servers and splicers used to splice advertisements into digital networks
Advanced Video Coding	Refers specifically to video compression standardized in ISO/IEC 14496-10 [AVC]
Analog Cue Tone	In an analog system, a signal which is usually either a sequence of DTMF tones or a contact closure that denotes to ad insertion equipment that an advertisement avail is about to begin or end
Avail	Time space provided to cable distributors by cable programming services during a program for use by the CATV distributor; the time is usually sold to local advertisers or used for channel self promotion
Break	Avail or an actual insertion in progress
C3	An audience measurement that is specified to include live viewing as well as DVR viewing up to 75 hours from the ad minute original broadcast time
Cipher Block Chaining	This is a specific method of encryption. It is one of the methods used in DES
Cyclic Redundancy Check	A method to verify the integrity of a transmitted Message
Component Splice Mode	A mode of the Cue Message whereby the program_splice_flag is set to '0' and indicates that each PID/component that is intended to be spliced will be listed separately by the syntax that follows. Components not listed in the Message are not be spliced
Cue Message	In the context of this document a message is the contents of any splice_info_section of an SCTE 35 message in a compressed stream
Data Encryption Standard	A method for encrypting data with symmetric keys
Digital Video Broadcasting	An international consortium for the development of digital television systems
Electronic Code Book	This is a specific method of encryption. It is one of the methods used in DES
Entitlement Control Message	These are private conditional access information messages which specify control words and possibly other, typically stream-specific, scrambling and/or control parameters

Entitlement Management Message	These are private conditional access information messages which specify the authorization levels or the services of specific decoders. They may be addressed to single decoders or groups of decoders
Event	A splice event or a Viewing Event as defined below
In Point	A point in the stream, suitable for entry
Media Presentation Description	A formalized description for a [DASH] Media Presentation for the purpose of providing a streaming service
Message	In the context of this document a message is the contents of any splice_info_section
Out Point	A point in the stream, suitable for exit
Packet identifier	A unique 13-bit value used to identify elementary streams of a program in a single or multi-program Transport Stream [13818-1]
payload_unit_start_indicator	A bit in the transport packet header that signals, among other things, that a section begins in the payload that follows [13818-1]
PID stream	A stream of packets with the same PID within a transport stream
Presentation Time	The time that a presentation unit is presented in the system target decoder [13818-1]
Program	A collection of video, audio, and data PID streams which share a common program number within an MPTS [13818-1]
Program In Point	A group of PID stream In Points that correspond in Presentation Time
Program Out Point	A group of PID stream Out Points that correspond in Presentation Time
Program Splice Mode	A mode of the Cue Message whereby the program_splice_flag is set to '1' and indicates that the Message refers to a Program Splice Point and that all PIDs/components of the program are to be spliced
Program Splice Point	A Program In Point or a Program Out Point
Registration Descriptor	Carried in the PMT of a program to indicate that, when signaling Splice Events, splice_info_sections is carried in a PID stream within this program. The presence of the Registration Descriptor signifies a program's compliance with SCTE 35
reserved	The term "reserved", when used in the clauses defining the coded bit stream, indicates that the value may be used in the future for extensions to the standard. Unless otherwise specified in SCTE 35, all reserved bits are set to '1'
Splice Event	An opportunity to splice one or more PID streams
Splice Immediate Mode	A mode of the Cue Message whereby the splicing device chooses the nearest opportunity in the stream, relative to the splice_info_table, to splice. When not in this mode, the Message gives a "pts_time", which is a Presentation Time, for the intended splicing moment
Trigger	In the context of this document a message is the contents of any splice_info_section of an SCTE 224 message in a baseband stream
TV Everywhere	A being authenticated as a subscriber to a distributor, it is the ability to view TV content on the internet in addition to on one's television
Viewing Event	A television program or a span of compressed material within a service; as opposed to a Splice Event, which is a point in time

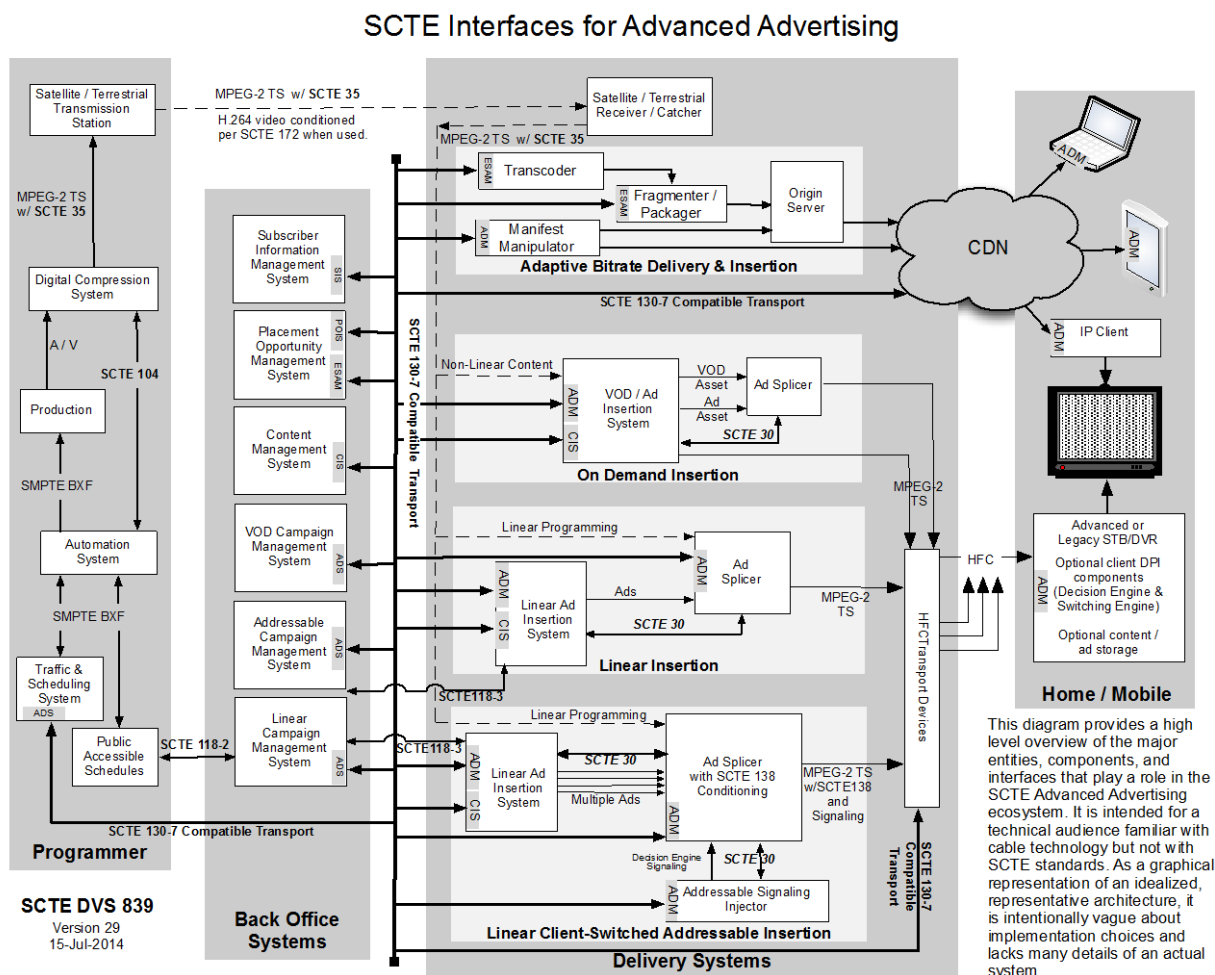
5. Overview

The SCTE 35 standard supports the splicing of MPEG-2 transport streams for the purpose of Digital Program Insertion, which includes insertion of advertisement and other content types. An in-stream

messaging mechanism is defined in SCTE 35 to signal splicing and insertion opportunities. A splicing device is free to ignore Splicing Events signaled by the Cue Message because the Message is not a command to splice, but is an indicator of the presence of an ad Avail. Inserting content in to the Avail is optional.

A Cue Message can also signal other events such as program start/end, content identification and other program or advertisement related events. These events can be used for applications such as network PVR, TV Everywhere, Regional Blackouts, Live VOD Capture or other advanced services.

As shown in the following diagram, Cue Messages are received and acted upon in the cable system headends by splicer and server devices to affect the insertion of local advertisements by splicing the ad bit stream (typically containing the commercial content) into the bit stream of programming content. Similar to SCTE 30, SCTE 35 does not differentiate between a splicing device and a server. When SCTE 35 uses the terms “splicer” or “splicing device” the meaning of the sentence may apply to a splicer/server combination as well. In actual practice it is common for servers (and not splicers) to parse, interpret and initiate action upon Cue Messages. Since splicer and server devices can be combined into one, this document often uses the term server/splicer to denote a device or set of devices that together perform both functions. This block diagram describes the overall functionality and interoperability associated with the systems that accomplish this.



In Figure 1 [DVS 839], a compliant MPEG-2 transport stream (either Multi Program Transport Stream or Single Program Transport Stream) is assumed for the provider stream. No further constraints beyond the inclusion of the defined Cue Messages are placed upon the stream in SCTE 35.

SCTE 35 specifies a technique for carrying notification of upcoming splice points in the transport stream. A splice information table is defined for notifying downstream devices of Splice Events, such as a network Break or return from a network Break. The splice information table, which pertains to a given program, is carried in a separate PID referred to by that program's Program Map Table (PMT). In this way, Splice Event notification can pass through most transport stream remultiplexers without need for special processing. However, remultiplexers may need to obey certain constraints when they carry the Cue Message. These constraints are addressed in SCTE 35 and are elaborated on within this document.

SCTE 35 does not address constraints on splicing devices and SCTE 35 splice_info_table syntax and is not intended to guarantee seamless splicing.

By using PTS to reference In Points and Out Points in the program or component streams frame accurate splicing is possible. This document does offer some advice on how to properly encode or condition the video to aid this. While SCTE 35 does not require any stream conditioning, it can be advantageous to get the best splice results. SCTE 172 defines constraints on insertion materials at In Points and Out Points. SCTE 172 also provides suggestions on splicer behavior at In Points and Out Points

5.1. Example SCTE 35 Decoder

A sample java SCTE 35 decoder application written using the Netbeans framework is available (provided separately). It allows you to enter a Base64 or Hex string of the SCTE 35 mpeg section and will decode it with a printout of the data. It will also convert the Base64 or Hex to the opposite format. This is open source sample code and it does not have extensive error checking. It may be useful to understand how to properly parse a Cue Message. The SCTE website will be updated to list the SCTE 35 decoder beside the listing of the SCTE 67.

6. Application Guidelines

6.1. Practical Boundaries for splice_time() in splice_insert()

How far ahead of the splice must a splice_insert Message be sent, relative to the picture it refers to, in order to be safely responded to by an Ad Insertion System? SCTE 35 requires a minimum of 4 seconds of "pre-roll," which is also termed "arm time."

The "arm time" denotes the time a Cue Message precedes that actual insertion. The arm time is typically in the range of 5 - 8 seconds. This is in line with the pre-roll time for analog cue-tones. The arm time cannot be so short that the Avail passes by before the Ad Insertion System has time to respond. A minimum of 4 seconds is required for "start" Cue Messages (out_of_network_indicator = 1) and a minimum of 4 seconds is recommended for "stop" Cue Messages (out_of_network_indicator = 0). It is not necessary to send a "stop" Cue Message if "Duration" is provided. Manufacturers should provide the option to generate a real time DTMF "stop" cue-tone at the actual switch point for legacy analog insertion systems.

More thought about the possible consequences is required if it is desirable to extend the arm time beyond the recommended eight seconds. To specify a maximum arm time therefore seems premature.

The splice_info_section itself does not have any PTS or DTS value associated with it, as it is not a PES structure.

6.2. System Delays

Most devices that operate on the video and audio components will delay the transport stream from the input to the output of the device. Encoders typically have delays of 0.5 to two or more seconds depending on if a low delay mode or multi-pass encoding mode is used. Splicers/statistical multiplexors also have delays in the same range. The satellite transmission of the transport stream also incurs some delay as well. Converting a transport stream to an adaptive bit-rate format such as [DASH] will incur significant delays both on the preparation and playback/buffering side.

While most providers prefer that there is as small amount of delay as possible, small amounts of additional delay are usually only a concern on live programs that have call-in or simulcast issues.

6.2.1. SCTE 104 Insertion Delays

Being able to insert messages with at least 4 seconds of pre-roll to allow the encoders to handle GOP structures properly, especially in AVC encoding, is a good practice. There are some events that are manually triggered and can lead to an immediate command being inserted. This will likely result in an inaccurate splice with MPEG-2 video or picture artifacts if AVC video is used.

One possible solution is to add a few seconds of HD-SDI delay before the SCTE 224 VANC inserter and craft the Triggers pre-roll value to account for that delay. This would have the effect of adding the additional pre-roll to the message but to the distributor it would be as if the command happened as an immediate command.

6.2.2. Encoder Delays

Encoder delays should be handled by the encoder properly resulting in frame accurate splicing. An encoder can pass the Trigger through immediately as a Cue Message in to the output essentially adding the encode delay to the pre-roll. In all cases the encoder needs to consider the encode delay when converting the SCTE 224 frame to the SCTE 35 PTS time.

6.2.3. Transmission Delays

Transmission delays are a factor when generating Cue Messages, although the variability of receipt and processing within receive sites is a major reason why Cue Messages use the in-stream PTS time.

6.2.4. Splicer/Multiplexor Delays

Section 6.1 describes the suggested minimum pre-roll of 4 seconds for splice insert commands. This allows for one second of message processing before the 3 second minimum to send a SCTE 30 Splice_Request Message. In order to ensure proper splice accuracy and quality these minimums need to be adhered to.

Some splicers and servers using MPEG-2 video will attempt to complete insertions even if the timing is below the minimums as this does represent lost revenue. Some splicers use their delay and/or have faster processing capabilities to allow for lower minimums and theoretically could have slightly negative splice times due to delays.

6.3. Splice Time Accuracy

Although precise splice times can be specified using both the Program Splice Mode and Component Splice Mode a server/splicer may choose to perform splices at other than these precise times. Since most

splicing devices operate in the compressed domain they are typically constrained to only perform splices at specific locations in the stream if the splice is to meet the semantic and syntactic requirements of the MPEG-2 Systems specification [13818-1] and the relevant video coding specification.

See Section 6 of the document for stream requirements for splicing MPEG-2 video. Stream requirements for splicing AVC and other video codecs are described in SCTE 172. While the details of Out Point requirements differ, both video codecs require that In Points be intra-coded pictures with no stream dependencies on pictures prior to the In Point (a closed GOP in MPEG-2, an IDR in AVC). In time shifted environments (for example VOD), where ads may be inserted into content, the Out and In Points in the entertainment content may be coincident and, if coincident, needs to meet the requirements of both an In Point and an Out Point.

If the signaled splice times are not aligned with these locations, the server/splicer may choose nearby In or Out Points, using the splice times as guidelines, if the resulting splice is to be syntactically seamless.

Splicing between AVC streams with differing GOP structures and/or buffer delays may place further requirements on the selection of splice points. Please refer to SCTE 172 for details.

In the case of audio, the video Presentation Time of the splice will not in general align with the audio Presentation Time(s) and, therefore, corresponding audio splices may be executed at times that are "close" to the nearest video Presentation Time. Consequently, it may be necessary for a server/splicer to replace existing audio around the splice point with silence to ensure a clean transition.

Note that specific server/splicer implementations may place further, or fewer, restrictions on In and Out Points depending upon their capabilities. Section 6 of this document discusses various aspects of server/splicer capabilities and performance.

In order for a splice to be visually seamless, that is occurring at the desired location in the content with no visible transition (for example, black frames) the specified splice points should align with the GOP structure of the content. This can be accomplished by various means:

- Encoders that insert splice points under control of an automation system can SCTE 224 adjust the GOP structure around the splice point so that it meets In or Out Point requirements without impacting video quality
- Systems that insert Cue Messages into existing content may re-encode the GOP surrounding the desired splice point so that it meets In or Out Point requirements
- Since splice points will often coincide with scene breaks or changes, encoders that employ scene change detection will often generate streams where GOP structures naturally align with splice points (for example, beginning each new scene with a closed GOP or IDR would enable scene breaks to be used as splice points after encoding).

If the content cannot be encoded such that the desired splice points align with the GOP structure, other techniques can be used such as encoding a number of black frames around the time of the desired splice point to minimize the visual effect if the server/splicer performs the splice at the nearest In or Out Point.

Some server/splicers may be capable of inserting black or repeat frames, and corresponding audio silence, between the nearest valid In or Out Point and the specified splice location to minimize visual effect of adjusting the splice point. If this is used in an environment where an ad is replacing an existing ad in the network feed the server/splicer is required to ensure that the combined length of the black frames and the inserted content does not exceed the length of the replaced ad. Similarly, there is the possibility that the length or GOP structure of the replacement ad may not precisely align with the original. In that event a splicer may insert black frames to align with the network feed GOP structure when returning from the ad.

Alternatively, some splicers may use their buffers and delay to look for an appropriate In Point/Out Point somewhere in their buffer. This technique should be used carefully and every effort to move in opposite directions each time a time-jump is made to even out the offset and not overflow or underflow the buffer.

Thus, behavior when requested splice points do not align with actual splice points will be one factor that differentiates between splicers.

6.3.1. Splicer Timing

This section of the recommended practice considers the splicer control of digital ad insertion. The following recommendation would change the way ad splices are controlled. As specified by SCTE 30, the server controls the splice time by sending a splice request, containing the splice time, to the splicer.

Assumptions for a splicer controlled system

- An ideal system where the Cue Message always points to the correct insertion point.
- Distributors will never need to adjust ad insertion timing at their location.

This section recommends two possible ways of having the splicer control the splice time; one using SCTE 130 and one using SCTE 30.

6.3.1.1. Using SCTE 130

Distributors choosing a splicer centric system can implement an SCTE 130 based Ad Insertion System that behaves as a splicer centric system.

The splicer can also use SCTE 130 to contact an ad decision system (an ad decision system, or ADS as defined in SCTE 130 Part 1) and request what ad to play at the given break. The splicer will need access to the ads in some manner, but this is not a difficult task.

This option requires frame accurate PTS splice time from the provider. There is no requirement for time synchronization with this method.

6.3.1.2. Using the SCTE 30 asset_id_descriptor

Another possible way of deploying a splicer centric system is to utilize the asset_id_descriptor defined in Section 8.5.5 of SCTE 30:asset_id_descriptor() Field Definitions. The asset_id_descriptor is added to a Splice_Request command by the server to tell the splicer what ad to play. If the Splice_Request returns the exact time in the Cue_Request then the splicer should attempt to play (by local file, NFS or some similar method) the ad when it needs it. There is no longer any need for time synchronization with this method.

“The asset_id_descriptor() is an implementation of the splice_API_descriptor() which shall only be used in the Splice_Request message. This descriptor is intended to be used when the asset playout is being performed by the splicer, but can be used to identify the asset being played also. It is suggested for advertising content that the spot id or other identifier be used rather than a fully qualified name. For long form VOD type content, a full asset name will probably be required. How the Server and Splicer manage the name format is not specified in this document.”

6.4. Splice_event_id Usage and Uniqueness

Note: This section also applies to the segmentation_event_id carried in the segmentation_descriptor.

Cue Messages can be created by several means: from information embedded in the original audio / video source, by uplink event trigger systems or by headend event trigger systems. When all of these sources are combined within a service there is potential for collisions in the splice_event_id value chosen for the SCTE 35Cue Message. The 32-bit splice_event_id should therefore be partitioned in the following way to allow each source a unique range of splice_event_id values:

Syntax	Bits	Type
Event_source	4	uimsbf
Event_number	28	uimsbf

Event_source – A user assigned number for the source of the Cue Message.

Event_number – A number chosen by the Event source to identify an instance of the Cue Message.

The splice_event_id serves to identify a particular instance of an opportunity to change the multiplex. At least two distinct splice_events are required to perform one insertion, one to initiate the insertion and one to end it (unless the duration is included in the Message signaling the splice-out).

Each Event is required to have a unique splice_event_id. A splice_event_id cannot be re-used before the splicing opportunity it describes (fully or in part) has completely passed. A splice_event_id is regarded to be in-use from the first Cue Message where it appears until about one second after the splice time that it is associated with. It may be reused for a new Splice Event immediately afterwards. It is not possible to use the same splice_event_id to signal an Avail's start and stop if the stop is signaled before the start has been executed. However, it is possible to reuse the same splice_event_id if the first stop Message is sent after the start has been executed.

The advertising industry uses additional information besides the splice_event_id to identify the actual material being played and the time of its insertion. This information includes: service name, time, ad-agency and spot number.

As long as the Event_source is unique for each point at which Cue Messages can be inserted in the following chain the Event identifiers will not collide:

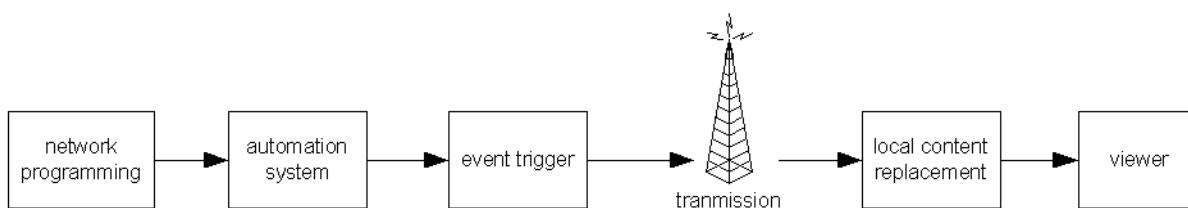


Figure 2 - Cue Message Insertion Points

The following values for Event_source are suggested:

Source of Cue Message	Event_source Value	Example splice_event_id ranges
Cue embedded in original source material	0	0x00000000, 0x00000001 0x0ffffff
Cue created by automation system switching	4	0x40000000, 0x40000001 0x4ffffff
Cue created by manual event trigger system	6	0x60000000, 0x60000001 0x6ffffff
Cue created by local content replacement system	12	0xC0000000, 0xC0000001 0xCffffff

6.5. Use of splice_schedule() Command

The splice_schedule() command is intended for announcements of large schedules. It is not intended for the announcement of single Events. Current implementations of SCTE 35 focus on the support of single Events that do not rely on the use of the splice_schedule() command.

6.6. Component Splice Mode

While Component Splice Mode is part of the SCTE 35 standard, there are currently no known implementations and it should be used with caution.

The primary reason for the Component Splice Mode is to allow the replacement of or the passage of individual elementary streams of any type in a manner consistent with the program content and the advertiser's intent.

For example, during a local commercial Break, it might be useful to allow informational data (e.g. market returns) that is being streamed from the network provider to continue, although the viewer is being shown a locally inserted commercial. Conversely, with an advanced set-top box implementation, one might offer the viewer information to be downloaded for later review as part of the commercial. In this case, if the data content exceeds the duration of the commercial, the advertising data might be continued, although the program has resumed.

The methodology defined in SCTE 35 allows some data types to be passed while allowing others to be blocked or replaced during a Break. The decision which to pass and which to block is made by the Message inserter. A splicing device may choose to behave differently if it knows better or is commanded to do so by an entity within the headend.

It is the intention of the standard, in both the Program Splice Mode and the Component Splice Mode, that the splicer should have a great deal of freedom in choosing the actual Access Units (both video and audio) on which to splice. Some equipment will provide frame accurate splices and some may not.

The signaled splice_time may or may not be on an anchor frame or I-frame and the splicer may need to round off to a frame that makes sense for its capabilities.

There appears to be industry consensus that in Program Splice Mode it makes the most sense to give a Presentation Time of a video frame and then let the splicer choose whichever audio frame is closest or which makes sense for its methods. This wasn't, however, specified in the standard. It is believed that the industry will come up with the best way to use the tools (especially if it differs from what is stated here).

In the Component Splice Mode, things work the same way. A single splice_time should be used (called the default splice_time). It is optional to give a splice_time for each component. There is some concern in the industry that the creator of a Message will try to "help" the splicer by giving the exact time of all components and possibly complicate the job of the splicer (or the splicer may need to ignore the splice_time for some components).

It is understood that SMPTE [312] requires a time per component to define the exact Access Unit for each component to be spliced. The cable industry is looking at a different usage for the time per component approach, however. SCTE 35 allows individual components to start or end at very different times from the normal beginning or ending of the Break. For instance, a "data" component such as a Java applet may have to be downloaded to the set-top box a number of seconds prior to the ad in order to support the ad. This is the primary usage of the unique splice_time for each component.

6.6.1. Erroneous Component Splice Commands

When a Cue Message (in component mode) specifies invalid component tags, the splice should be executed for all correctly identified components. The intent is to let insertion succeed if possible, even if the complete chain of devices results in a violation of the standard. Several potential cases are given below.

In case a device removes one of several audio streams after the Cue Message has been inserted, but fails to update the Cue Messages (either because it is unaware of them or other reasons), the combination of stream and Message at the server/splicer will be invalid. It is, however, still possible for the server/splicer to perform a valid insertion.

In case a device adds a component (e.g. a data channel), but fails to update the Cue Message (same possible reasons as above), it is still possible to perform a valid insertion.

Whenever a discrepancy between actual stream and Cue Message exists, a server/splicer should perform the insertion, in order to add error resilience to the chain.

6.7. Pre-Roll Functionality - Accomplishing a Pre-Roll Function

A Pre-Roll function was necessary in the days of analog ad insertion to allow time for a tape machine to get up to speed by the beginning of the ad Avail. So Analog Cue Tones were often sent about 5 to 8 seconds prior to the Avail (each network used its own chosen time and the consistency was varying). The early arrival of the cue tone has remained unchanged during the days of hybrid ad insertion where tape machines are replaced by MPEG servers and decoders, but the insertion was still analog. This early cue tone is useful for digital ad insertion as well. It gives the ad insertion equipment time to access its ad database, to determine which ad to play next, to start accessing its disk drives and to fill the MPEG decoder pipeline. This however takes much less time and allows for pre-rolls typically in the 4 to 5 second range. However, legacy analog systems remain in operation and should be considered until the industry "sunssets" the requirement to provide DTMF cue tones and differing "splice return" with pre-roll for digital splices versus actual end-time switch points for analog.

The splice_insert() command is constructed such that a pre-roll time can be used. It is also possible to repeat a splice_insert() command to increase the error resilience of the system.

The simplest variant is to send the splice_insert() command once about 8 seconds prior to an Avail. This is similar to the analog insertion case.

An enhancement is to send the splice_insert() command 3 times: at about 8 sec, 6 sec and 4 sec prior to the Avail. This increases the redundancy and prevents lost insertion opportunities. A lost Avail nationwide is a very expensive error. In the case of multiple Messages denoting a single Avail, the splice_event_id field in all such Messages are required to be identical. It is allowable that the splice time be different from Message to Message in the case where the first splice time is approximate and subsequent Messages supply a more accurate splice time. Servers should only act on the last time received that meets the pre-roll constraints.

While servers should be reading the splice_event_id and handling the messages as described in the above paragraph, other implementations are known to exist. Some servers once they receive a valid Cue Message and have ads to play will block all subsequent Cue Messages until playback is finished. If they receive a valid Cue Message with no ads to play, they may see the subsequent Cue Messages and incorrectly advance to the next avail within the window. Some servers have an adjustable timer to block subsequent cues for a number of seconds to prevent this from happening.

6.8. Conditional Access and Cue Encryption

There are no known implementations of the following sections. The message encryption is typically handled by the proprietary conditional access systems in the satellite feed.

6.8.1. What to Encrypt

The format of the SCTE 35 section allows for the payload to be optionally encrypted. This includes all data from the “splice_command_type” through to the “E_CRC_32”. This mechanism allows for any current, or future, commands to be protected. The reasons for protection can range from anti-piracy (e.g. commercial killers), malicious tampering of the data, and privacy when using cascaded ad insertion devices.

The specification requires that an encrypted CRC be present so that any receive device is able to verify that the encrypted data has not been changed since origination. This could be due to noise, but normally this type of corruption is detected by the standard CRC. The encrypted CRC has the primary purpose as a signature to detect that the receiver is authorized to receive the Message. (i.e. that they have the correct control word). It is also used to detect willful modification of the encrypted data. This CRC is not present when the section is sent in-the-clear.

6.8.2. Operation in a Cue Insertion Device

A cue insertion device is used to create splice_info_sections and insert them into a transport stream. When encryption is desired, the cue inserter uses a fixed key entered by a distributor, plus the algorithm selected, to encrypt the section before being transmitted.

The cue insertion device should be able to maintain multiple simultaneous keys for the same program. The reason is that each ad inserter in a cascade could require one or more different keys to decrypt the splice_info_section.

A cue insertion device is only required to implement one of the standard encryption methods. In recommended practice, the cue insertion device should implement all standard algorithms.

6.8.3. Operation in an Ad Insertion Device

An ad insertion device consumes splice_info_sections. When a splice_info_section is encrypted, the ad inserter will only act upon the section if the decryption is successful. This is determined by checking the

encrypted CRC before interpreting the data. A failure will usually mean that the device is simply not authorized to interpret the information that the section contains.

The ad inserter is expected to allow the encrypted splice_info_section to pass through it to the next device. This is true whether the decryption was successful or not. The ad inserter never releases the contents of a decrypted section for security reasons.

Once the section has been decrypted, the device may act on the information it contains, or discard it. This is the same operation that would be performed if the section had been sent in-the-clear.

The ad inserter device should be able to hold 256 different decryption keys. The cw_index field in the section indicates which of these decryption keys should be used. The method of key interchange is out of scope for this document.

For normal operation, it is expected that one or two keys are used for any one pair of send/receive devices. There is a provision for a large number of keys to allow an ad inserter to connect to multiple programs in a transport stream, or to allow cascaded ad inserters to use different ranges of cw_indexes when connecting to the same program.

An Ad Insertion device should implement all defined decryption algorithms in SCTE 35. This allows it to receive encrypted sections from any cue insertion device. Any cue insertion device may choose any standard algorithm to protect the section, and the Ad Inserter is required to be able to decrypt any Message it is authorized to receive.

6.8.4. Theory of Operation

6.8.4.1. Encryption Verses Scrambling

Scrambling means the use of the “standard” DES or DVB Common Scrambling algorithm that is being used for video and audio services.

When deciding how best to encrypt the splice_info_section, one needs to consider that the Ad Insertion functionality is typically not located in a settop box. There has been a provision made for this model, but it is not the primary model. Most systems will employ an Ad Insertion device in a digital headend (or some other distribution point). These standalone devices are typically computers. Using a descrambler would require a custom circuit designed to descramble the stream. Also, since the entire transport packet is scrambled, the ad insertion device has no access to the header data as it does with the current model. Some of the information, such as pts_adjustment, may need to be modified even when the section is not descrambled. There are similar considerations when creating the stream.

The model used for the splice_info_section is closer to the ECM model, than the Elementary Stream model. An ECM section is independently encrypted (and decrypted), and authorized by some external mechanism. In the case of the ECM, it is usually the EMM that controls authorization. In the case of the splice_info_section, it is a manual authorization.

The fixed key model was chosen for simplicity. The key distribution was left undefined and may use any mechanism that gets the key to the decryptor in a secure and timely manner. It is conceivable that a committee may standardize on some type of ECM and/or EMM to distribute the keys in the transport stream.

6.8.4.2. **Standard Encryption Methods**

The standard provides for three types of algorithms. All three algorithms use the DES decryptor as the basic building block. For Triple DES, the DES encryptor is also required. Software implementations of the DES algorithm are readily available.

6.8.4.3. **Section Stuffing**

All DES type algorithms (block encryptors rather than stream encryptors) require the length of the data to be an exact multiple of 8 bytes. To achieve this, stuffing is often required. The `alignment_stuffing` field may be present whether the section is encrypted or not. The number of bytes of stuffing can be determined during the interpretation of the data. The stuffing bytes are never used, so they may take on any value.

To determine the length of the stuffing, one uses the fact that the total length is known from the `section_length` field. Knowing the size of the header, it is possible to determine the start of a `splice_command`. Every command has a size completely determined by the syntax within the command itself. The length is:

$$\langle \text{section_length} \rangle + 3 - \langle \text{end_of_command} \rangle - \langle \text{length_trailing} \rangle$$

where `<length_trailing>` is four for non-encrypted sections and eight for encrypted sections.

Algorithmically, it is easier to simply ignore the stuffing. Work forward to decrypt and interpret the command, and work backward from the end to find the CRCs.

6.8.4.4. **DES Electronic Codebook (ECB) Algorithm**

DES ECB requires a 56-bit key plus 8 parity bits to make up a complete 64-bit key, which is then used by the algorithm to encrypt or decrypt a stream. The DES algorithm is symmetric. The same key is used in both the encryptor and the decryptor. The actual encryption and decryption algorithms are slightly different to allow this symmetry to work.

It is suggested that the full 64-bit key be distributed between the two devices. The key will be entered as a 16 digit hexadecimal number. For example, 0x123456789ABCDEF0 would be distributed. In engineering notation, the leftmost digit is the most significant digit, and the MSB of this nibble is the most significant bit of the key (bit 63). Cipher algorithms generally use FIPS notation to represent keys. In this case, the leftmost bit of the key is numbered Bit 1, through to Bit 64 on the right. Therefore, bit 63 of the distributed key value will be loaded into Bit 1 of the initial key register. Similarly, bit 0 of the key value is loaded into Bit 64 of the key register.

The Electronic Codebook method of encryption uses the *same* key for every 8-byte block of the original Message. Figure 3 gives an example of a simple 3-block Message being encrypted. The arrows represent operations. Across the top, the key is being loaded into the key register. Side to side, the data is being shifted into the encryption register, and across the bottom, the DES algorithm is being applied.

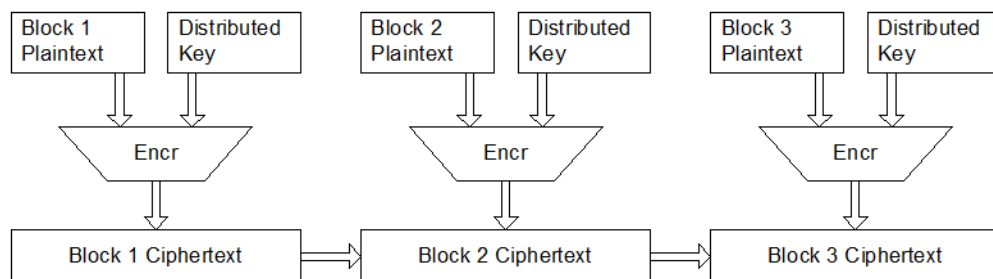


Figure 3 - DES ECB Example

6.8.4.5. DES Cipherblock Chaining (CBC) Algorithm

DES CBC requires a 56-bit key plus 8 parity bits to make up a complete 64-bit key, which is used by to the algorithm to encrypt or decrypt a stream. The same bit ordering rules and key distribution methods can be used for CBC that were used for the ECB algorithm described in section 6.8.4.4.

The Cipherblock Chaining method of encryption uses a different key for every 8-byte block of the original Message.

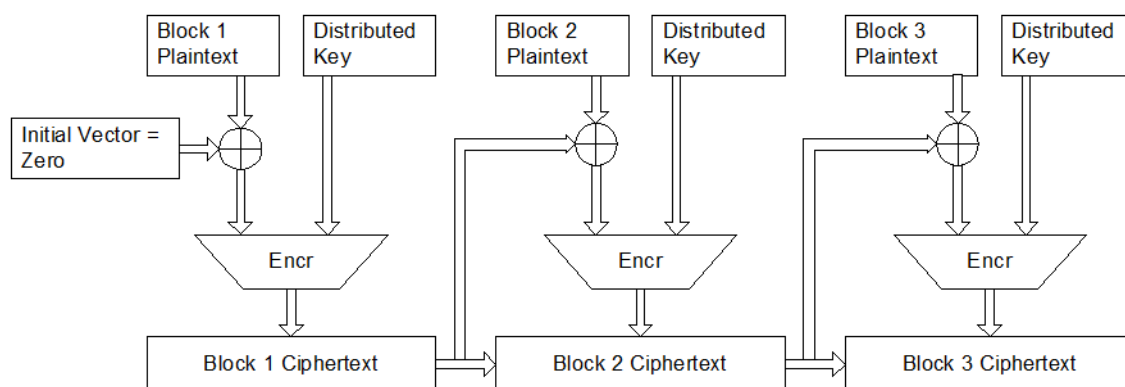


Figure 4 - DES CBC Encryption Example

From Figure 4, one can see that the plaintext data is modified by the result of the previous encryption block. For the first block, one needs to have an “initial vector” to apply to the exclusive-or block. It turns out that the initial vector provides no extra security, whether it is known or not. So, for this system, the initial vector can be zero, which removes the need for it to be distributed with the key.

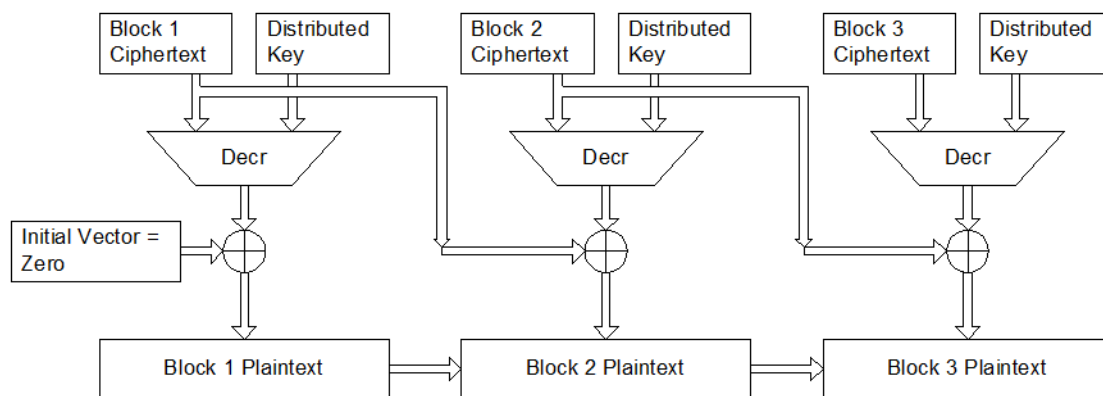


Figure 5 - DES CBC Decryption Example

From Figure 5 one can see the DES CBC decryption is slightly different from encryption. The exclusive-or block requires the previous block of encrypted data to arrive at the proper cipher key for the DES algorithm. In the decryptor, this encrypted block is derived from the transmitted data.

6.8.4.6. Triple-DES ECB Mode (EDE Method) Algorithm

Triple DES uses the standard DES algorithm, but applies the algorithm 3 times for each block of input data. This provides a significant security enhancement. The disadvantage is that distributing a 192-bit key (actually three 64-bit keys) is needed to obtain the security enhancement.

Using the combination of two algorithms and three passes (encryption and decryption are different), it is possible to generate eight different Triple DES variants¹. The variants are designated by a three-letter code. Of these variants, the most common is selected for use in a splicing system. This variant is designated the EDE method, referring to the fact that pass one uses encryption, pass two uses decryption, and pass three uses the encryption algorithm again. For simplicity, the DES ECB mode of operation is used for the Triple-DES algorithm².

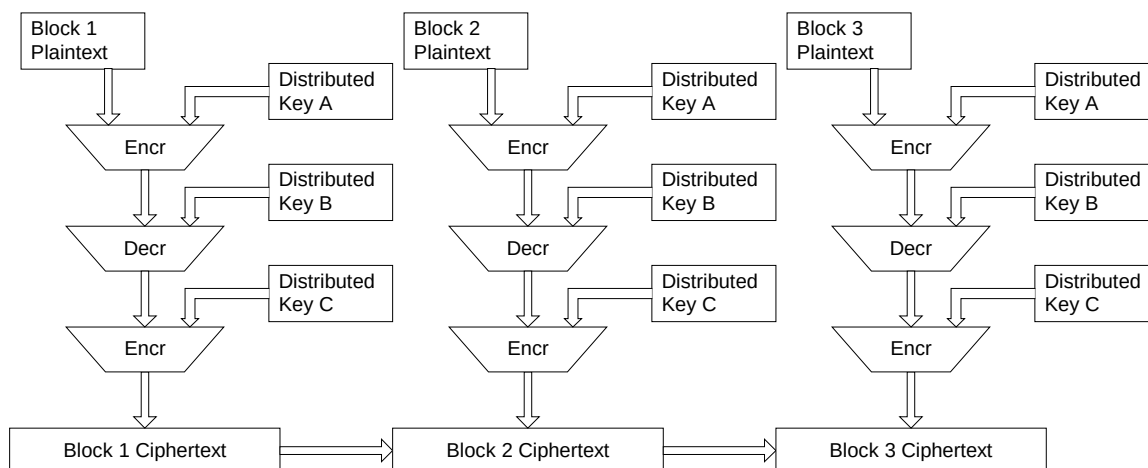


Figure 6 - Triple-DES ECB Encryption Example

Triple-DES decryption requires that the algorithm be applied in the reverse of the encryption algorithm. As a result, the block diagram for decryption is slightly different again. It can be found in Figure 7.

¹ Triple DES also has two key variants. This can be simulated by making KEY A == KEY C.

² Triple DES can also use the cipherblock chaining mode, but this not one of the standard methods.