

# OpenCable™ Interface Specifications

## CableCARD™ Copy Protection System Interface Specification

**OC-SP-CCCP-IF-C01-050331**

**CLOSED**

### **Notice**

This document is the result of a cooperative effort undertaken at the direction of Cable Television Laboratories, Inc. for the benefit of the cable industry and its customers. This document may contain references to other documents not owned or controlled by CableLabs. Use and understanding of this document may require access to such other documents. Designing, manufacturing, distributing, using, selling, or servicing products, or providing services, based on this document may require intellectual property licenses from third parties for technology referenced in the document.

Neither CableLabs nor any member company is responsible to any party for any liability of any nature whatsoever resulting from or arising out of use or reliance upon this document, or any document referenced herein. This document is furnished on an "AS IS" basis and neither CableLabs nor its members provides any representation or warranty, express or implied, regarding the accuracy, completeness, or fitness for a particular purpose of this document, or any document referenced herein.

© Copyright 2000-2005 Cable Television Laboratories, Inc. All rights reserved.

## Document Status Sheet

<b>Document Control Number:</b>	OC-SP-CCCP-IF-C01-050331			
<b>Document Title:</b>	CableCARD™ Copy Protection System Interface Specification			
<b>Revision History:</b>	I01 – Issued 01/07/00 I02 – Issued 04/10/00 I03 – Issued 07/14/00 I04 – Issued 03/07/01 I05 – Issued 04/18/01 I06 – Issued 12/21/01 I07 – Issued 05/24/02 I08 – Issued 11/26/02 I09 – Issued 2/10/03 I10 – Issued 7/7/03 I11 – Issued 9/5/03 I12 – Issued 11/21/03 I13 – Issued 4/2/04 I14 – Issued 8/31/04 I15 – Issued 11/19/04 C01 – Released 3/31/05			
<b>Date:</b>	March 31, 2005			
<b>Responsible Editor:</b>	Jeff Hamilton			
<b>Status:</b>	<i>Work in Progress</i>	Draft	Issued	Closed
<b>Distribution Restrictions:</b>	<i>author only</i>	GL/Member	<i>GL/ Member/ Vendor</i>	Public

**Key to Document Status Codes:**

- Work in Progress
An incomplete document, designed to guide discussion and generate feedback, that may include several alternative requirements for consideration.
- Draft
A document in specification format considered largely complete, but lacking review by Members and vendors. Drafts are susceptible to substantial change during the review process.
- Issued
A stable document, which has undergone rigorous member and vendor review and is suitable for product design and development, cross-vendor interoperability, and for certification testing.
- Closed
A static document, reviewed, tested, validated, and closed to further engineering change requests to the specification through CableLabs.

**Trademarks:**

DOCSIS®, eDOCSIS™, PacketCable™, CableHome®, CableOffice™, OpenCable™, CableCARD™, and CableLabs® are trademarks of Cable Television Laboratories, Inc.

# Contents

<b>1</b>	<b>INTRODUCTION</b>	<b>1</b>
1.1	Scope	1
1.2	References	1
1.3	Acronyms and Abbreviations	1
1.4	Copy Protection System Components	1
1.5	Implementation Outline	1
1.6	Historical Perspective	1
1.7	Related Documents	1
<b>2</b>	<b>SYSTEM OVERVIEW (INFORMATIVE)</b>	<b>2</b>
2.1	NRSS Copy Protection Framework	2
2.2	Device Authentication	2
2.3	Key Exchange and Interface Encryption	2
2.4	Data Channel Protection	2
2.4.1	Copy Control Information	3
2.5	Identifying Fraudulent Devices and Disabling of Services	3
<b>3</b>	<b>HOST AUTHENTICATION MECHANISMS</b>	<b>4</b>
3.1	Protocol Components	4
3.1.1	X.509 Version 3 Certificate	4
3.1.2	Device Parameters	4
3.1.3	System Parameters	4
3.1.4	Processing Basics	4
3.2	CableCARD Binding and Registration	7
3.2.1	ID Report-back Mechanism	7
3.2.2	Authentication Phase 1 – Certificate Verification & DH Key Exchange	8
3.2.3	Authentication Phase 2 – Authentication Key Verification	8
3.2.4	Authentication Phase 3 – Authentication Key Verification	9
3.2.5	Headend Report Back Methods	9
3.2.6	Handling of Interrupts to the Authentication Phases	10
3.3	Power-up Re-Authentication	10
3.4	CableCARD Operation with Multiple Hosts	10
3.5	Host Operation with Multiple CableCARD Devices	10
<b>4</b>	<b>CRYPTOGRAPHIC FUNCTIONS</b>	<b>11</b>
4.1	Authentication Key Generation	11
4.2	Copy Protection Key Generation	11
4.2.1	Basic Key Generation Protocol	11
4.2.2	CableCARD Device Copy Protection Key	11
4.2.3	Host Copy Protection Key	12

<b>4.3 Copy Protection Key Refresh</b> .....	<b>12</b>
4.3.1 CP-Key Session Period .....	12
4.3.2 CP-Key Refresh.....	13
4.3.3 CA System Key Refresh.....	13
4.3.4 Key Refresh Initialization .....	13
4.3.5 Channel Change .....	13
<b>4.4 Diffie-Hellman Key Exchange Algorithm</b> .....	<b>13</b>
<b>4.5 SHA-1 Secure Hash Algorithm</b> .....	<b>14</b>
<b>4.5 Random Number Generation</b> .....	<b>14</b>
<b>4.6 DFAST Algorithm</b> .....	<b>14</b>
<b>4.7 RSA Digital Signatures</b> .....	<b>14</b>
<b>5 HOST SERVICE REVOCATION MECHANISMS</b> .....	<b>15</b>
5.1 System Issues .....	15
5.2 Revocation Circumstances .....	15
5.3 Fraudulent CableCARD and Host Identification.....	15
5.4 CA System Revocation & Selective Denial of Services.....	15
5.5 The Revocation Process .....	15
5.6 Implementation in the Headend.....	15
<b>6 COPY CONTROL INFORMATION (CCI)</b> .....	<b>16</b>
6.1 CCI Definition .....	16
6.1.1 EMI – Digital Copy Control Bits .....	16
6.1.2 APS – Analog Protection System.....	16
6.1.3 Host Set CCI Values .....	16
6.1.4 CIT—Constrained Image Trigger .....	16
6.2 Associating CCI with a Service.....	17
6.3 Conveying CCI from Headend to CableCARD Device .....	17
6.4 Conveying CCI from CableCARD Device to Host.....	17
6.4.1 CCI Delivery Instances .....	17
6.4.2 Authenticated Tunnel Protocol .....	17
<b>7 TRANSPORT ENCRYPTION FROM CABLECARD DEVICE TO HOST</b> .....	<b>19</b>
7.1 MPEG Scrambling .....	19
7.2 Transport Processing .....	19
7.3 Timing of Scrambling Mode Transitions.....	19
7.4 CP-Scrambling as a Function of CA-Scrambling and EMI Value.....	19
7.5 Debug Modes Prohibited .....	19
<b>8 HOST, CABLECARD, &amp; HEADEND MESSAGING PROTOCOLS</b> .....	<b>20</b>
8.1 Message Protocol Overview .....	20
8.2 CableCARD & Host Common Messages.....	20

8.2.1	Copy Protection Resource Identifier.....	20
<b>8.3</b>	<b>One-way System Message Protocol.....</b>	<b>20</b>
8.3.1	Protocol Flow Overview.....	20
8.3.2	Host Authentication Messages.....	22
8.3.3	Host Authentication Key Verification Messages.....	22
<b>8.4</b>	<b>Two-way System CableCARD CPS Message Protocol.....</b>	<b>22</b>
8.4.1	Protocol Flow Overview.....	22
8.4.2	Host Authentication Protocol Implementation.....	22
<b>8.5</b>	<b>CCI Simple Authentication Tunnel Protocol (SATP) Messages.....</b>	<b>23</b>
<b>APPENDIX A</b>	<b>LUHN CHECK DIGIT (NORMATIVE).....</b>	<b>24</b>
<b>APPENDIX B</b>	<b>APPLYING CP KEY TO DES ENGINE (NORMATIVE).....</b>	<b>25</b>
<b>APPENDIX C</b>	<b>CABLECARD COPY PROTECTION X.509 CERTIFICATE PROFILE AND MANAGEMENT (NORMATIVE).....</b>	<b>26</b>
<b>APPENDIX D</b>	<b>REVISION HISTORY (INFORMATIVE).....</b>	<b>27</b>

## List of Tables

Table 1	– Length of Keys and Parameters Used in the Key Generation.....	12
Table 2	– CCI Bit Assignments.....	16
Table 3	– Host Default and Error CCI Values.....	16
Table 4	– CIT Values and Application.....	16
Table 5	– Copy Protection Resource Class.....	20

## List of Figures

Figure 1	– CableCARD-Host CP Operation.....	6
Figure 2	– CP-Key Refresh and CCI Transfer (Informative).....	18

This page is intentionally left blank.

# 1 INTRODUCTION

## 1.1 Scope

The CableCARD Copy Protection System complies with Section 1.1 of SCTE 41 [1].

## 1.2 References

In order to claim compliance with this specification, it is necessary to conform to the following standards and other works as indicated, in addition to the other requirements of this specification. Notwithstanding, intellectual property rights may be required to use or implement such normative references.

- [1] SCTE 41 2003 (Formerly DVS 301) POD Copy Protection Standard, Society of Cable Telecommunications Engineers, <http://www.scte.org/standards/standardsavailable.html>
- [2] DFAST encryption technology (U.S. Patent number 4,860,353 and related know-how) is licensed from CableLabs as part of the OpenCable POD Module Interface Technology and described in CableLabs' license materials. This technology shall be licensed to any interested party under fair and reasonable terms. For licensing information, refer to the OpenCable website <[www.opencable.com](http://www.opencable.com)> or contact CableLabs at +1 (303) 661-9100.
- [3] FIPS PUB 140-2 "Security Requirements for Cryptographic Modules", Section 4.9.2
- [4] OC-SP-CC-IF-I18-041119, OpenCable CableCARD Interface Specification, November 19, 2004, Cable Television Laboratories, Inc., <[www.opencable.com](http://www.opencable.com)>
- [5] OC-SP-SEC-I05-040831, OpenCable System Security Specification, August 31, 2004, Cable Television Laboratories, Inc., <[www.opencable.com](http://www.opencable.com)>

## 1.3 Acronyms and Abbreviations

The CableCARD Copy Protection System complies with Section 1.3 of SCTE 41 [1].

## 1.4 Copy Protection System Components

The CableCARD Copy Protection System complies with Section 1.4 of SCTE 41 [1].

## 1.5 Implementation Outline

The CableCARD Copy Protection System complies with Section 1.5 of SCTE 41 [1].

## 1.6 Historical Perspective

The CableCARD Copy Protection System complies with Section 1.6 of SCTE 41 [1].

## 1.7 Related Documents

See Section 1.7 of SCTE 41 [1].

## 2 SYSTEM OVERVIEW (INFORMATIVE)

### 2.1 NRSS Copy Protection Framework

The CableCARD Copy Protection System complies with Section 2.1 of SCTE 41 [1].

### 2.2 Device Authentication

The CP System requires authentication of the Host and CableCARD Device prior to the CableCARD Device descrambling any CA-scrambled content. The CableCARD Device requests the Host's X.509 Certificate List and the Host provides it. The Host requests the CableCARD Device's X.509 Certificate List and the CableCARD Device provides it. Authentication is based on:

- CableCARD Device being able to verify the signature of the X.509 host device certificate that contains Host\_ID and the CableCARD Device being able to verify the signature of the Host Manufacturer's XCA certificate; and
- Host being able to verify the signature of the X.509 CableCARD device certificate that contains the POD\_ID and the Host being able to verify the signature of the CableCARD Manufacturer's XCA certificate.
- CableCARD Device and Host proving they each hold the private key paired with the public key embedded in the X.509 certificate by signing a DH session key and sending it to the other device for signature verification.
- CableCARD Device and Host proving that they can derive the authentication key.
- The Host\_ID and POD\_ID extracted from the X.509 certificates are not included in the CRL, as checked in the Headend.

Additionally, the CableCARD Device will validate the authenticity of the X.509 Host Certificate and the Host will validate the authenticity of the X.509 CableCARD Certificate using the License Administrator's public verification keys that are delivered by the CA System to the CableCARD Device and Host in an authenticated manner. If the Validated\_Host\_ID value is the same as the Host\_ID in the authenticated X.509 Host Device Certificate and the Validated\_POD\_ID value is the same as the POD\_ID in the authenticated X.509 CableCARD Device Certificate, the Host and CableCARD Device can continue with the key exchange process described in Section 3.

### 2.3 Key Exchange and Interface Encryption

The CableCARD Copy Protection System complies with Section 2.3 of SCTE 41 [1].

### 2.4 Data Channel Protection

The CableCARD Copy Protection System complies with Section 2.4 of SCTE 41 [1], with the exception that Section 2.4.1 of SCTE 41 is replaced by Section 2.4.1 below.

### **2.4.1 Copy Control Information**

Copy control information (CCI) is passed from the CableCARD Device to the Host across the data channel to inform the Host device of the level of copy protection required. The CCI is sent in the clear to the Host device, but the integrity of the information is maintained by authenticating the CCI using a simple protocol.

The one-byte CCI field contains information that the Host uses to control copying of content. Two EMI bits control copying on Host digital outputs (that support EMI), two APS bits control copying on analog outputs, one bit as a Constrained Image Trigger, and three bits are reserved.

## **2.5 Identifying Fraudulent Devices and Disabling of Services**

The CableCARD Copy Protection System complies with Section 2.5 of SCTE 41 [1].

## 3 HOST AUTHENTICATION MECHANISMS

### 3.1 Protocol Components

The CableCARD Copy Protection System complies with Section 3.1 of SCTE 41 [1].

#### 3.1.1 X.509 Version 3 Certificate

The CableCARD Copy Protection System complies with Section 3.1.1 of SCTE 41 [1].

#### 3.1.2 Device Parameters

The CableCARD Copy Protection System complies with Section 3.1.2 of SCTE 41 [1].

#### 3.1.3 System Parameters

The CableCARD Copy Protection System complies with Section 3.1.3 of SCTE 41 [1].

#### 3.1.4 Processing Basics

The POD Copy Protection System (CPS) comprises the following basic steps:

1. The Host shall report copy protection as a resource during the profile inquiry process (see Section 3.2.2). Failure to do so constitutes a failure of the copy protection system (see Section 8.2.1). After the copy protection resource has been reported, the CableCARD Device shall disable CA decryption and set authenticated status to false.
2. The CableCARD Device shall open a session to the copy protection resource, Section 8.2.1. Failure to open a valid session constitutes a failure of the copy protection system (see Section 3.2.2).
3. The CableCARD Device shall send a CP\_open\_req APDU to the Host; see Section 8.2.2.1 of [1].
4. The Host shall respond with the CP\_open\_cnf APDU within 5 seconds. Failure to respond to any request within 5 seconds constitutes a failure of the Host and shall cause the CableCARD Device to set the IIR flag [4].
5. The Host shall respond with the System 2 bit set in CP\_system\_id\_bitmask; see Section 8.2.2.2 of [1]. Failure to do so constitutes a failure of the copy protection system (see Section 3.2.2).
6. If the CableCARD Device contains a valid Authentication key in its non-volatile memory, it shall request the Host to send its AuthKeyH.
7. The Host shall respond with its AuthKeyH, if available. If it is not available, then it shall transmit a value of all 0's. A value of all 0's shall be recognized by the CableCARD Device as an invalid AuthKeyH.
8. The CableCARD Device shall compare the received AuthKeyH with its stored AuthKeyP. If the authentication keys match, then the certificates are considered valid, the CableCARD Device shall set authenticated status to true, preserve the DH Key and authentication keys, only the Copy Protection Key is regenerated, and continue with step 17. If the authentication keys do not match the CableCARD Device shall set Headend Validated to false in non-volatile memory and the CableCARD Device and Host shall continue with step 9.
9. The CableCARD Device shall send its POD Certificate Data (POD\_DevCert and POD ManCert), the newly generated DH public key (DH\_pubKeyP), and the Diffie-Hellman key signature. In this message, the CableCARD Device also requests that the Host deliver its Host Certificate Data (Host\_DevCert and Host\_ManCert) and signed DH public key (DH\_pubKeyH).
10. The Host shall reply to the CableCARD request with its Host Certificate Data (Host\_DevCert and Host\_ManCert) and newly generated and signed DH public key.

11. The CableCARD Device shall verify the Host\_DevCert, the Host\_ManCert, the signature on the Diffie-Hellman public key (DH\_pubKeyH) and extract the Host\_ID from the Host certificate. If verification fails, this constitutes a failure of the copy protection system (see Section 3.2.2).
12. The Host module shall verify the POD\_DevCert, the POD\_ManCert, the signature on the Diffie-Hellman public key (DH\_pubKeyP) and extract the POD\_ID from the CableCARD certificate. If verification fails, this constitutes a failure of the copy protection system (see Section 3.2.2).
13. After these exchanges, both the CableCARD Device and the Host calculate their respective authentication key, AuthKeyP and AuthKeyH.
14. The CableCARD Device shall request AuthKeyH and compare it to AuthKeyP. If they are not equal, this constitutes a failure of the copy protection system (see Section 3.2.2). If they are equal, then the CableCARD Device and Host shall store the derived authentication key into non-volatile memory, set authenticated status to true, and enable CA-descrambling of content marked with EMI=00.
15. If headend validation of the CableCARD Device and Host ID is complete, go to step 18. Otherwise, the following depends upon what path is available to report device IDs to the headend.
  - a. If possible, i.e., in a system with an active return data channel or telco return path, the CableCARD Device shall send the POD\_ID and Host\_ID to the cable headend via an upstream OOB private CA message or telco modem. The CableCARD Device also stores the Host\_ID and POD\_IDs in non-volatile memory so they can be compared with the validated IDs received back from the headend. Status of this transmission shall be stored in non-volatile memory to prevent unnecessary retransmissions.
  - b. In systems in which no automated return path is available, the CableCARD Device sends a display message as defined in Section 3.2.5.1.
16. The headend CA System records the pairing between Host\_ID and POD\_ID, and looks for the Host\_ID in the revocation list. If not found, the CA System re-sends the Host\_ID and POD\_ID back to the CableCARD Device in a private authenticated CA System ID validation message. This message may be sent to the CableCARD Device substantially later in time than when the POD\_ID and Host\_ID are reported to the headend.
17. Until the CableCARD and Host ID headend validation is completed the CableCARD Device shall CA decrypt only those services with an EMI value of 00.

[Asynchronously and independent of this process, the CA system typically sends an EMM to the CableCARD Device to authorize appropriate services. Some of these services may have EMI values of 01, 10, or 11. The CableCARD Device shall not CA-descramble services with these EMI values until headend ID validation is complete, even if these services are otherwise authorized in an EMM.]
18. The CableCARD Device authenticates the ID validation message and compares the received Host\_ID and POD\_ID with the IDs extracted from the Host device certificate and the CableCARD device certificate, respectively. If they match, the CableCARD Device shall store the Headend Validated True in non-volatile memory and allow CA decryption of high value content with EMI values of 01, 10, or 11, if so authorized by an EMM. If they do not match, the CableCARD Device shall continue to limit its CA decryption to those services with EMI values of 00 only.
19. If the headend CA system receives a new revocation list, it shall examine all previously reported IDs and if there are any matches, it shall notify the cable operator.
20. If a CableCARD Device reports a failure of the copy protection system to the headend CA system, the headend CA system shall notify the cable operator.

The following flowcharts show an implementation of the above steps:

### POD-Host CP Operation

Note: This chart is an informative illustration of a possible compliant solution. Other solutions are permissible within the requirements of this document.

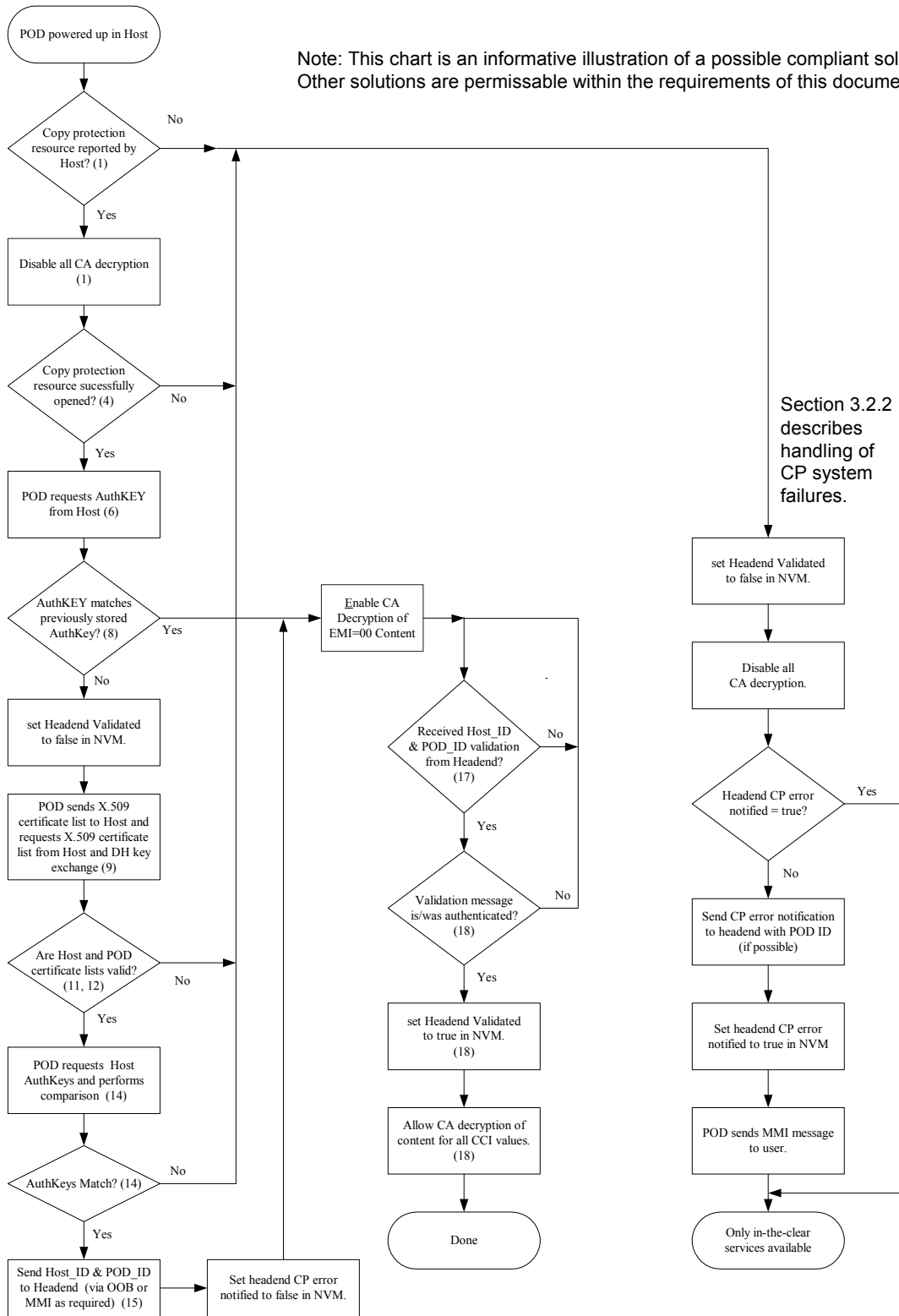


Figure 1 – CableCARD-Host CP Operation

## 3.2 CableCARD Binding and Registration

The CableCARD Copy Protection System complies with Section 3.2 of SCTE 41 [1], with the exception that Sections 3.2.1, 3.2.5, and 3.2.6 are replaced by Sections 3.2.1, 3.2.5, and 3.2.6 below.

### 3.2.1 ID Report-back Mechanism

The Host\_ID and POD\_ID must be reported to the service provider before the CableCARD Device will provide High Value content to the Host. The retailer may perform this service for the subscriber.

In a system with two-way RF or telco return functionality (Host, Cable plant or phone line, and Headend all support compatible connections) the Host\_ID and POD\_ID may be sent to the Headend in an authenticated CA System message.

For one-way Cable systems, unidirectional Hosts, or any system without an automatic report-back mechanism, the POD and Host ID's must be reported manually. The CableCARD Device shall always include a CableCARD-Host binding information menu item in the application info APDU for display of the Host\_ID and POD\_ID to the user (see below). The CableCARD Device shall determine if the Host is unidirectional by sending the oob\_tx\_tune\_req() APDU and receiving the oob\_tx\_tune\_cnf() APDU. If the status\_field is a 0x01 (RF transmitter not physically available), then the CableCARD Device shall define the Host as unidirectional. The CableCARD Device shall also have a means of determining if the system it is resident in is unidirectional.

Following power-up if the CableCARD Device determines it has not bound to the Host and is either in a unidirectional system or is inserted into a unidirectional Host, the CableCARD Device shall open a session to the Host's MMI resource (if not already open), and send an open MMI dialog request.

If the Host is in an off state or any non-video viewing state, it shall deny the dialog open request. When the Host is in a video viewing state, it shall grant the open MMI dialog request. The Host\_ID and POD\_ID SHALL be converted from binary to decimal by first truncating the POD\_ID to its 40 LSBs, separating the manufacturer number as the 10 (remaining) MSBs and the unit number as the 30 LSBs. Each number SHALL be converted to decimal separately and then concatenated into a single 12 digit decimal value. The Luhn digit SHALL be calculated on this composite decimal value and appended to the right end of the value. The CableCARD Device shall then send a message containing the Host\_ID and POD\_ID to the Host in the clear. This message shall contain the Host\_ID and POD\_ID, each with a Luhn check digit (described in Appendix A) appended, in decimal format (digits 0-9) so that they are easy to read and speak and can be entered from a touch-tone telephone keypad. The Host shall display the message and confirm to the CableCARD Device that the message has been displayed to the customer.

Hosts require the ability to display the POD\_ID and Host\_ID to the subscriber for manual reporting to the service provider as generated by the CableCARD Device in the form of an MMI screen. This display shall be referred to as the "CableCARD-Host binding information screen".

The CableCARD-Host binding information screen shall be displayed only if:

1. The subscriber requests display of this message in a host menu,
2. Headend validation has not been received and the user selects a program with copy protection active (CCI ≠ 0),
3. The CableCARD Device requests display, e.g., at initial insertion and binding on or command by the headend.

The CableCARD Device shall support display of the CableCARD-Host binding information screen consisting of the service provider contact telephone number and all of the pertinent data that the subscriber is required to report back to the service provider CAS. This numeric data shall include:

1. POD\_ID (13 decimal digits including the Luhn digit) formatted as follows: M-MMU-UUU-UUU-UUL where the digits are all decimal defined by the fields M = PHICA assigned CableCARD Manufacturer number, U= Manufacturer assigned unit number, L = Luhn check digit<sup>1, 2</sup>.
2. Host\_ID (13 decimal digits including the Luhn digit) formatted as follows: M-MMU-UUU-UUU-UUL where the digits are all decimal defined by the fields M = PHICA assigned Host Manufacturer number, U= Manufacturer assigned unit number, L = Luhn check digit<sup>1, 2</sup>.
3. A phone number to contact the MSO, defined in the example by the X.

An example of a displayed message follows, where the following information is reported back to the CAS<sup>3</sup>:

In order to start cable service for this device please contact customer service at  
X-XXX-XXX-XXXX

CableCARD ID: M-MMU-UUU-UUU-UUL

Host ID: M-MMU-UUU-UUU-UUL

Additional CableCARD data may be displayed, at the discretion of the CableCARD and CA system vendor.

In order to support a Host request to display the CableCARD-Host binding information screen, the CableCARD Device shall support the “CableCARD-Host binding information application” with application\_type = 0x01. The CableCARD-Host binding information application shall be defined strictly as display of the CableCARD-Host binding information screen as defined in this Section above. The CableCARD Device shall only support one (1) CableCARD-Host binding information application and therefore only one application with application\_type = 0x01.

### 3.2.2 Authentication Phase 1 – Certificate Verification & DH Key Exchange

The CableCARD Copy Protection System complies with Section 3.2.2 of SCTE 41[1].

### 3.2.3 Authentication Phase 2 – Authentication Key Verification

A long-term “Authentication Key” (AuthKeyP and AuthKeyH) is derived based on the information exchanged between the CableCARD Device and Host during the first step of authentication. This Authentication Key is calculated as a function of the Host\_ID, the POD\_ID, and the Diffie Hellman public keys

Both the CableCARD Device and the Host calculate an AuthKey, as described in the Cryptographic Functions section (Section 4). The CableCARD Device sends a request message to the Host to request the Authentication Key derived by the Host. If the CableCARD Device confirms that its derived Authentication Key is the same as the one received from the Host, then the CableCARD Device accepts that Host as authenticated and enables CA-descrambling of content marked with EMI=00. This derived Authentication Key shall be stored in non-volatile

<sup>1</sup> Note: It is the sequence of the defined values that is specified. The use of dashes as the delimiter is shown with an example placement to be consistent with the examples used elsewhere in this specification and SCTE-41.

<sup>2</sup> See the OpenCable System Security Specification [5] for details on use of binary and decimal values in the POD\_ID and Host\_ID.

<sup>3</sup> The text portion of this screen is shown as an example only; there is no implied requirement to duplicate the exact wording or formatting shown. All three numeric fields MUST be included as defined above.

memory, and can be used later in the calculation of the Copy Protection Key. If a matching AuthKey has not been received within five seconds of the request message, the CableCARD Device shall display the MMI message and report back to the Headend as described above in section 3.2.2.

Authentication at this step is achieved based on the Host being able to prove that it can derive the same AuthKey as the CableCARD Device.

### **3.2.4 Authentication Phase 3 – Authentication Key Verification**

The CableCARD Copy Protection System complies with Section 3.2.4 of SCTE 41 [1].

### **3.2.5 Headend Report Back Methods**

#### **3.2.5.1 One-way System Device Registration and Validation**

The CableCARD Copy Protection System complies with Section 3.2.5.1 of SCTE 41 [1] with the addition of step 4 which shall be:

4. The user must telephone the service provider and read the Host\_ID and POD\_ID from the screen to a custom service representative (CSR). Additional data that is present on the screen, may also be read from the screen, if requested by the CSR.

#### **3.2.5.2 Manual Return Authentication – Error and Other Conditions**

The CableCARD Copy Protection System complies with Section 3.2.5.2 of SCTE 41 [1] with the exception that the last heading, “Normal Diagnostic Operation”, and the paragraph and example display message following it shall be replaced by:

##### ***Host Request for Binding Information***

If the Host requests the CableCARD-Host binding information screen and both the Host\_ID and POD\_ID have been received and authenticated, independent of whether the Host or system are unidirectional or bidirectional, then the CableCARD Device shall display the CableCARD-Host binding information screen as described above in Section 3.2.1<sup>4</sup>:

In order to start cable service for this device  
please contact customer service at:  
X-XXX-XXX-XXXX

CableCARD ID: M-MMU-UUU-UUU-UUL

Host ID: M-MMU-UUU-UUU-UUL

Additional CableCARD data may be displayed, at the discretion of the CableCARD and CA system vendor.

#### **3.2.5.3 Two-way System CableCARD and Host ID Validation**

The CableCARD Copy Protection System complies with Section 3.2.5.3 of SCTE 41 [1].

---

<sup>4</sup> The text portion of this screen is shown as an example only; there is no implied requirement to duplicate the exact wording or formatting shown. All three numeric fields MUST be included as defined above.

### **3.2.6 Handling of Interrupts to the Authentication Phases**

The CableCARD Device shall not allow any interruption during the authentication phases, such as a power outage, RF connection outage, CableCARD Device removal, etc., to cause any loss of copy protection functionality.

### **3.3 Power-up Re-Authentication**

The CableCARD Copy Protection System complies with Section 3.3 of SCTE 41 [1].

### **3.4 CableCARD Operation with Multiple Hosts**

The CableCARD Copy Protection System complies with Section 3.4 of SCTE 41 [1].

### **3.5 Host Operation with Multiple CableCARD Devices**

The CableCARD Copy Protection System complies with Section 3.5 of SCTE 41 [1].

## 4 CRYPTOGRAPHIC FUNCTIONS

The text and figure in this section are replaced by the text and figure in Section 4 of SCTE 41 [1].

### 4.1 Authentication Key Generation

The CableCARD Copy Protection System complies with Section 4.1 of SCTE 41 [1].

### 4.2 Copy Protection Key Generation

The CableCARD Copy Protection System complies with Section 4.2 of SCTE 41, with the exception that Sections 4.2.2 and 4.2.3 in SCTE are replaced by Sections 4.2.2 and 4.2.3 below.

#### 4.2.1 Basic Key Generation Protocol

- 1) CableCARD Device checks whether a previously derived authentication key is already stored in dedicated non-volatile memory. If such an  $AuthKey_P$  is present, then continue to the next step. Otherwise, restart the whole authentication process as detailed in Section 3.2.
- 2) The CableCARD Device confirms that the current IDs match the previously stored IDs. If they are the same, the CableCARD Device shall proceed with the key generation process; otherwise, the CableCARD Device shall restart the whole authentication process as detailed in Section 3.2.
- 3) The CableCARD Device generates its 64 bit random number ( $N\_module$ ).
- 4) The CableCARD Device sends this  $N\_module$  and its ID ( $POD\_ID$ ) in the clear to the Host.
- 5) The Host generates its 64 bits random number ( $N\_Host$ ).
- 6) The Host sends  $N\_Host$  and its  $Host\_ID$  in the clear to the CableCARD Device.
- 7) The CableCARD Device computes the Copy Protection Key based on long-term keys and newly exchanged random number using the SHA-1 hash function and the DFAST algorithm, as described in the following section.
- 8) The Host computes the Copy Protection Key also based on long-term keys and newly exchanged random number using the SHA-1 hash function and the DFAST algorithm, as described in the following section.

#### 4.2.2 CableCARD Device Copy Protection Key

The SHA-1 function is first used to hash the long-term keys,  $AuthKey_P$  and the  $DHKey$ , and the random numbers exchanged for key generation. The result is named  $Ks$ :

$$Ks = \text{SHA-1} [AuthKey_P | DHKey | N\_Host | N\_module]_{MSB\ 128}$$

Truncating the 160-bit SHA-1 output to its 128 MSBs, left-most bits, generates a seed,  $Ks$ , with the proper 128-bit length for the input to the DFAST engine. The DFAST algorithm is applied to  $Ks$  to produce the 56-bit value of the Copy Protection Key, also known as  $Ks\_dfast$ :

$$CP\text{-Key} = Ks\_dfast = \text{DFAST} [ Ks ]$$

DFAST details are specified in a separate document; contact the PHICA. Table 1 defines the size of keys, as well as the parameters used to derive them.

**Table 1 – Length of Keys and Parameters Used in the Key Generation**

Key or Variable	Size (bits)	Description
Nonces (N_Host, N_module)	64 bits each	Random numbers used to refresh the CP-Key.
Authentication Keys (AuthKey <sub>H</sub> , AuthKey <sub>P</sub> )	160 bits each	Results from the Host authentication process. It is a long-term key, and is stored in a non-volatile memory.
Diffie-Hellman shared secret Key (DHSK)	1024 bits	DH shared secret key represented as an octet string (PKCS format). It is a long-term key, and is stored in non-volatile memory.
SHA-1ed Key Ks	128 bits	The first 128 bits (16 bytes) of the 160 bit SHA-1 output, where the SHA-1 input is the DHKey, Authentication Key, and nonces from CableCARD Device and Host.
Copy Protection Key (Ks_dfast)	56 bits	DFAST output, final encryption and decryption Key.

### 4.2.3 Host Copy Protection Key

The Host computes its SHA-1 key based on the Authentication Key (*AuthKey<sub>H</sub>*), the 1024-bit shared secret DH key (*DHKey*), and the random numbers exchanged in the key generation. This key is named *Ks*:

$$Ks = \text{SHA-1} [AuthKeyH | DHKey | N\_Host | N\_module]_{MSB\ 128}$$

Truncating the 160-bit SHA-1 output to its 128 MSBs, left-most bits, generates a seed, *Ks*, with the proper length for the DFAST engine. The DFAST algorithm is applied to *Ks* to produce the 56-bit value of the Copy Protection Key.

## 4.3 Copy Protection Key Refresh

The CP key shall refresh periodically as initiated by the CableCARD Device. The CA System will set the refresh period with a parameter, *max\_key\_session\_period*, transmitted to the CableCARD Device by the CA System with maximum security.

For each single CP\_Key refresh: after the CableCARD Device initiates a CP key refresh cycle it shall start a Key Refresh timer. The CableCARD Device shall stop scrambling the selected program during the synchronization of keys. It shall start to encrypt again on the earlier of: successful completion of the authenticated CP key refresh cycle, or transmitting unencrypted data for one second. The CCI will not be changed during this 1 second period.

Each CP Key refresh shall recalculate the content key using a new pair of nonces (N\_Host, N\_module) exchanged between the CableCARD Device and Host.

Note that the CableCARD Device requests the Host's Authentication Key at every power up or hard reset if it has a valid Authentication Key stored in non-volatile memory. The CableCARD Device compares the received AuthKey<sub>H</sub> to its stored AuthKey<sub>P</sub> to detect if it has been inserted into a new Host or if the Host has been bound to a different CableCARD Device. If the authentication keys match, then the CableCARD Device shall initiate a CP Key refresh. (If a valid AuthKey<sub>P</sub> is not found, the CableCARD Device initiates a full binding process.)

### 4.3.1 CP-Key Session Period

The key session period is the period of time in which the CableCARD Device and Host utilize the same key for copy protection and CCI calculations before the CableCARD Device initiates a CP-Key refresh. There is a

maximum length of this period, `max_key_session_period`, programmable by the CA System. The CableCARD Device shall implement a timer which is not dependent on the program selected by the Host, and is reset at the completion of each CP-Key refresh. If this timer reaches the value of the `max_key_session_period`, the CableCARD Device shall initiate a CP Key Refresh. The `max_key_session_period` shall be implemented as a 16-bit value with a resolution of 10 seconds (one decasecond). If the value of `max_key_session_period` is zero, then the maximum key session period is unlimited. The Host is not aware of `max_key_session_period`.

#### 4.3.2 CP-Key Refresh

The CableCARD Device controls the timing of the CP-Key refresh. When the CableCARD Device sends its nonce to the Host in the `CP_data_req()` message, the CableCARD Device starts a Key Refresh Timer. When the Host receives the `CP_data_req()`, the Host generates its nonce and sends it to the CableCARD Device in the `CP_data_cnf()` message. The Host shall be implemented such that it shall transmit a `CP_data_cnf()` message within one second of receiving a `CP_data_req()` message.

The CableCARD Device and Host shall start the calculation of the Copy Protection keys when that Host issues the `CP_data_cnf()`. The CableCARD Device and Host shall both be implemented such that each calculates its Copy Protection key within eight seconds. The CableCARD Device shall send the `CP_sync_req()` to the Host when the Key Refresh Timer reaches nine seconds. Until this time, the CableCARD Device and Host shall utilize the “old” key for copy protection and CCI calculations. This timing ensures that both the CableCARD Device and Host have a minimum of eight seconds to complete key calculation. The CableCARD `CP_sync_req()` message indicates that the CableCARD Device has completed calculation of the Copy Protection key. The Host shall issue the `CP_sync_cnf()` message when it has received the `CP_sync_req()` message and has completed calculation of the Host Copy Protection key.

In a single CP\_Key operation, when the CableCARD Device issues the `CP_sync_req()` message, the CableCARD Device shall turn off scrambling of the MPEG content output and set the `CP_transport_scrambling_control_field` to 00. The Host receives cleartext packets and will recognize these packets as unencrypted according to MPEG rules. When the Key Refresh Timer reaches ten seconds, the CableCARD Device may commence scrambling of MPEG packets with the new CP key. If the key refresh has not completed when the Key Refresh Timer reaches ten seconds, the CableCARD Device shall disable CA decryption of copy protected content until the current or a retry CP\_Key refresh is completed by reception of the `CP_sync_cnf()` message. The Card shall commence scrambling of the MPEG packets with the new CP Key at the completion of the CP\_Key Refresh.

In dual CP\_Key operation all protected content shall be scrambled throughout the CP\_Key generation and change process. No one-second clear period shall occur. This is the primary objective of the dual key capability.

#### 4.3.3 CA System Key Refresh

The CableCARD Copy Protection System complies with Section 4.3.3 in SCTE 41 [1].

#### 4.3.4 Key Refresh Initialization

The CableCARD Copy Protection System complies with Section 4.3.4 in SCTE 41 [1].

#### 4.3.5 Channel Change

When a channel change occurs, the Host shall reset CCI to the default value as shown in Section 6.1.3. The Host shall immediately begin using the actual CCI value when it is received from the CableCARD Device. Channel change shall not cause a key refresh to occur.

### 4.4 Diffie-Hellman Key Exchange Algorithm

The CableCARD Copy Protection System complies with Section 4.4 of SCTE 41 [1].

## 4.5 SHA-1 Secure Hash Algorithm

The CableCARD Copy Protection specification employs the RSA signature algorithm with SHA-1 for all X.509 digital certificates. The CableCARD Copy Protection specification uses F4 (65537 decimal, 010001 Hex) as the public exponent for its signing operation. The CableLabs Device Root Certificate shall employ a modulus length of 2048 bits for signing the Manufacturer CA certificates it issues. Manufacturer CA certificates shall employ signature key modulus lengths of 2048 bits for signing the device certificates. The device certificates shall employ a modulus length of 1024 bits for CableCARD and Host devices.

The following functions and operations use the SHA-1 algorithm:

- Host Certificate Signature Verification: the signature algorithm is based on the RSA digital signature scheme defined in FIPS 180-1, which uses the SHA-1 primitive.
- CableCARD Certificate Signature Verification: the signature algorithm is based on the RSA digital signature scheme defined in FIPS 180-1, which uses the SHA-1 primitive.
- Authentication key generation as described in Section 4.1 of SCTE 41 [1].
- Copy Protection Key generation, as described in Section 4.2 of SCTE 41 [1].

## 4.6 Random Number Generation

The CableCARD Copy Protection System complies with Section 4.6 of SCTE 41 [1].

## 4.7 DFAST Algorithm

The CableCARD Copy Protection System complies with Section 4.7 of SCTE 41 [1].

## 4.8 RSA Digital Signatures

The CableCARD Copy Protection System complies with Section 4.8 of SCTE 41 [1].

## **5 HOST SERVICE REVOCATION MECHANISMS**

### **5.1 System Issues**

The CableCARD Copy Protection System complies with Section 5.1 of SCTE 41 [1].

### **5.2 Revocation Circumstances**

The CableCARD Copy Protection System complies with Section 5.2 of SCTE 41 [1].

### **5.3 Fraudulent CableCARD and Host Identification**

The CableCARD Copy Protection System complies with Section 5.3 of SCTE 41 [1].

### **5.4 CA System Revocation & Selective Denial of Services**

The CableCARD Copy Protection System complies with Section 5.4 of SCTE 41 [1].

### **5.5 The Revocation Process**

The CableCARD Copy Protection System complies with Section 5.5 of SCTE 41 [1].

### **5.6 Implementation in the Headend**

The CableCARD Copy Protection System complies with Section 5.6 of SCTE 41 [1].

## 6 COPY CONTROL INFORMATION (CCI)

The CableCARD Copy Protection System complies with Section 6 in SCTE 41 [1], with the exception that Section 6.1.3 is added, and Sections 6.1 and 6.4.2 are replaced as shown below.

### 6.1 CCI Definition

CCI is a single byte, 8 bit, field conveyed from CableCARD Device to Host. Five of the eight bits are defined. The remaining three are reserved. The reserved bits shall be set to zero by the CableCARD Device as shown in Table 2. The Host shall use the reserved bit values received from the CableCARD Device only for execution of the Authenticated Tunnel Protocol described below. The Host shall ignore the reserved bit values thereafter.

**Table 2 – CCI Bit Assignments**

CCI Bits #	7	6	5	4	3	2	1	0
CableCARD sets to	0	0	0	CIT	APS1	APS0	EMI1	EMI0
Host interprets as	rsvd	rsvd	rsvd	CIT	APS1	APS0	EMI1	EMI0

#### 6.1.1 EMI – Digital Copy Control Bits

The CableCARD Copy Protection System complies with Section 6.1.1 in SCTE 41 [1].

#### 6.1.2 APS – Analog Protection System

The CableCARD Copy Protection System complies with Section 6.1.2 in SCTE 41 [1].

#### 6.1.3 Host Set CCI Values

During periods when the CCI for a program is not yet known to the Host, e.g., immediately after a channel change, the Host shall use a default value for CCI as defined in Table 3 and shall start a 10-second timer. If the Host has not yet successfully completed the CCI delivery protocol when the timer reaches ten (10) seconds, the Host shall change CCI to the Error CCI Value as defined in Table 3.

**Table 3 – Host Default and Error CCI Values**

CCI Bits #	7	6	5	4	3	2	1	0
CCI Bit Assignment	rsvd	rsvd	rsvd	CIT	APS1	APS0	EMI1	EMI0
Default CCI Value	0	0	0	0	0	0	1	1
Error CCI Value	0	0	0	1	0	0	1	1

#### 6.1.4 CIT—Constrained Image Trigger

The Host shall control Image Constraint of high definition analog component outputs according to the value of CIT as shown in Table 4.

**Table 4 – CIT Values and Application**

CIT Value	Image Constraint Application
0	No Image Constraint asserted

1	Image Constraint required
---	---------------------------

## 6.2 Associating CCI with a Service

The CableCARD Copy Protection System complies with Section 6.2 in SCTE 41 [1].

## 6.3 Conveying CCI from Headend to CableCARD Device

The CableCARD Copy Protection System complies with Section 6.3 in SCTE 41 [1].

## 6.4 Conveying CCI from CableCARD Device to Host

The CableCARD Copy Protection System complies with Section 6.4 in SCTE 41 [1], with the exception that Section 6.4.2 is replaced by Section 6.4.2 below.

### 6.4.1 CCI Delivery Instances

The CableCARD Copy Protection System complies with Section 6.4.1 in SCTE 41 [1].

### 6.4.2 Authenticated Tunnel Protocol

The “authenticated tunnel protocol” is a means of verifying delivery of valid CCI from CableCARD Device to Host. The CableCARD Device and Host shall jointly execute the steps below once for each transfer of CCI. Any failure of the steps described below shall result in a failed CCI delivery. If the protocol is not completed before the one-second time-out expires the CableCARD Device shall disable CA-descrambling of copy protected content and the Host shall set CCI to the Host default CCI value until the CCI delivery protocol completes successfully.

The CableCARD Device shall check the status of CP-Key refresh before initiating this tunnel protocol. If the CP-Key Refresh Timer is between 8 and 10 seconds the CableCARD Device shall delay CCI transfer until CP-Key refresh is complete. See informative Figure 2.

Step 1. The CableCARD Device generates a new random number CCI\_N\_module and starts a 1-second time-out.

Step 2. The CableCARD Device sends CCI\_N\_module, program\_number, and a request for CCI\_N\_Host.

Step 3. The Host generates a new random number CCI\_N\_Host.

Step 4. The Host replies with CCI\_N\_Host and program\_number (received in step 2 above).

Step 5. The CableCARD Device calculates two values: CCI\_auth to authenticate CCI delivery, and CCI\_ack to authenticate Host acknowledgment of receipt, as:

$$\text{CCI\_auth} = \text{SHA-1}(\text{CCI} | \text{CP-Key} | \text{CCI\_N\_module} | \text{CCI\_N\_Host} | \text{program\_number} )$$

$$\text{CCI\_ack} = \text{SHA-1}(\text{CCI} | \text{CP-Key} | \text{CCI\_N\_module} | \text{CCI\_N\_Host} )$$

Step 6. The CableCARD Device transmits CCI\_auth, CCI, and program\_number to the Host.

Step 7. The Host calculates CCI\_auth using the received CCI value and compares it with the CCI\_auth value received from the CableCARD Device. Failed equivalence generates an error condition and the Host sets CCI to theError CCI Value, as shown in Table 3.

Step 8. The Host shall begin controlling its outputs based on valid CCI within one second.

Step 9. The Host calculates CCI\_ack and sends it to the CableCARD Device.

Step 10. The CableCARD Device compares the received CCI\_ack with the value calculated in step 5 above. Failed equivalence generates an error condition.

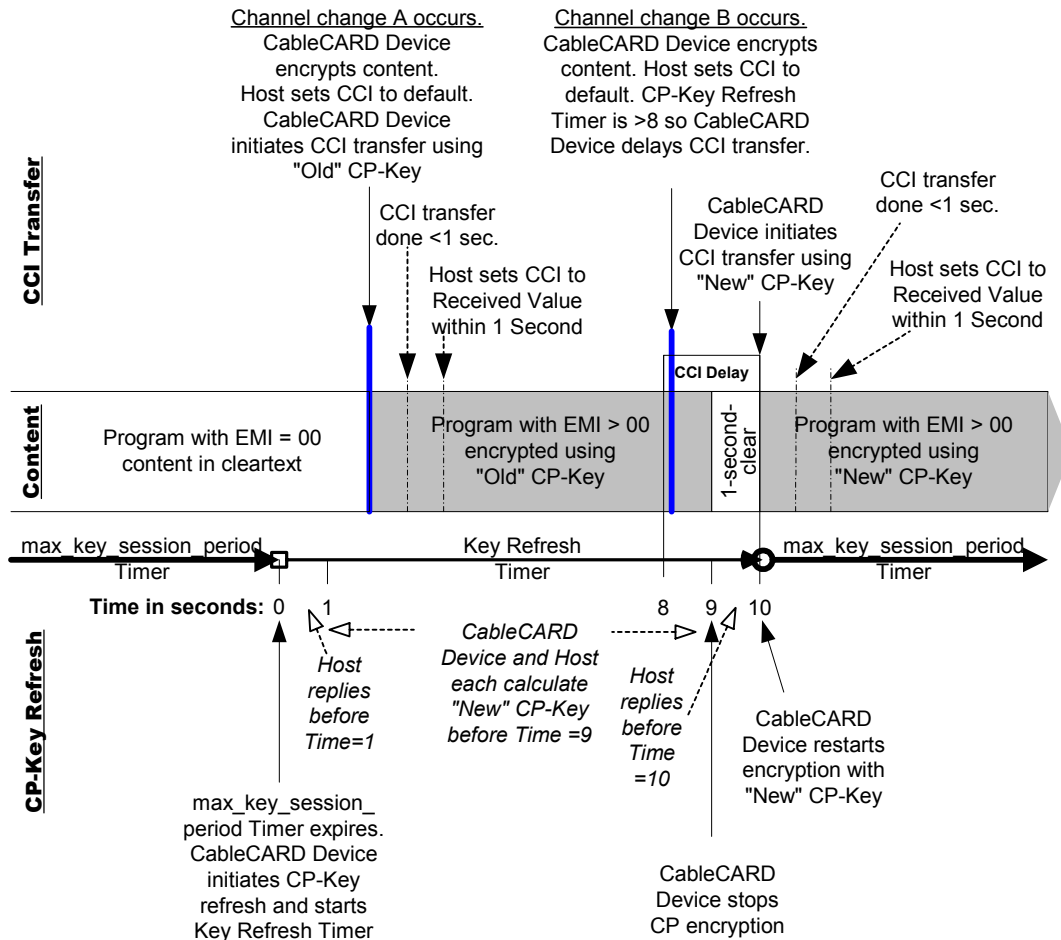


Figure 2 – CP-Key Refresh and CCI Transfer (Informative)

## **7 TRANSPORT ENCRYPTION FROM CABLECARD DEVICE TO HOST**

The CableCARD Copy Protection System complies with Section 7 in SCTE 41 [1], with the exception of Section 7.5 below.

### **7.1 MPEG Scrambling**

The CableCARD Copy Protection System complies with Section 7.1 in SCTE 41 [1].

### **7.2 Transport Processing**

The CableCARD Copy Protection System complies with Section 7.2 in SCTE 41 [1].

### **7.3 Timing of Scrambling Mode Transitions**

The CableCARD Copy Protection System complies with Section 7.3 in SCTE 41 [1].

### **7.4 CP-Scrambling as a Function of CA-Scrambling and EMI Value.**

The CableCARD Copy Protection System complies with Section 7.4 in SCTE 41 [1].

### **7.5 Debug Modes Prohibited**

Production CableCARD Devices or Hosts shall have no debug modes that bypass or compromise security provisions of this interface.

## 8 HOST, CABLECARD, & HEADEND MESSAGING PROTOCOLS

### 8.1 Message Protocol Overview

The CableCARD Copy Protection System complies with Section 8.1 in SCTE 41 [1].

### 8.2 CableCARD & Host Common Messages

The CableCARD Copy Protection System complies with Section 8.2 in SCTE 41 [1] with the exception that Table 8.2-C is superseded by Table 5 – Copy Protection Resource Class below.

**Table 5 – Copy Protection Resource Class**

Resource	Class	Type	Version	Identifier
Copy Protection	176	3	1	00B000C1

#### 8.2.1 Copy Protection Resource Identifier

The CableCARD Copy Protection System complies with Section 7.1 in the CableCARD Interface Specification [4]. This supersedes the reference to EIA-679-B, part B as defined in Section 8.2.1.1 of SCTE 41 [1].

### 8.3 One-way System Message Protocol

#### 8.3.1 Protocol Flow Overview

The CableCARD Copy Protection System complies with Section 8.3.1 in SCTE 41 [1].

##### 8.3.1.1 Authentication Protocol Implementation

The CableCARD Copy Protection System complies with Section 8.3.1.1 in SCTE 41 [1], with the addition of Section 8.3.1.1.1 below.

##### 8.3.1.1.1 One-Way System CableCARD CPS Protocol Steps Full Authentication

1. The CableCARD Device initiates the authentication protocol by sending a challenge request to Host. The CableCARD Device generates a secret random  $x$ ,  $1 \leq x \leq n - 2$ , and sends Host message. This message contains the CableCARD Certificate List Data (*POD\_DevCert* and *POD\_ManCert*), a signature of the Diffie-Hellman public key, and the Diffie-Hellman public key *DH\_pubKeyP*. The request is implemented by the *CP\_data\_req()* object, as defined in APDU layer. The *CP\_data\_req()* message used here is detailed in Section 8.3.2.1.
2. After receiving *CP\_data\_req()*, Host generates a secret random  $y$ ,  $1 \leq y \leq n - 2$ , and sends its reply with its Host Certificate List (*Host\_DevCert* and *Host\_ManCert*), a signature of the combined Host Diffie-Hellman public keys and Diffie-Hellman public key *DH\_pubKeyH* to the CableCARD Device. This response is implemented by the object *CP\_data\_cnf()* object, as detailed in Section 8.3.2.2.
3. The CableCARD Device checks if Host Certificate List is valid by:
  - a. Checking the value of the certificate type or format field; and

- b. The CableCARD Device verifies the Host's certificates (*Host\_DevCert* and *Host\_ManCert*), which is a sequence (chain) of X.509.v3 certificates, with the Host's device certificate signature first followed by its Manufacturer Device CA's certificate signature second, and the CableLabs Manufacturer Root CA's certificate signature last, assuming the Host has the Device Root CA's certificate pre-loaded.
4. The Host checks if CableCARD certificate is valid by:
  - a. Checking the value of the certificate type or format field; and
  - b. The Host verifies the CableCARD Device's certificates (*POD\_DevCert* and *POD\_ManCert*), which is a sequence (chain) of X.509.v3 certificates, with the CableCARD Device's certificate signature first followed by its Manufacturer Device CA's certificate signature second, and the CableLabs Manufacturer Root CA's certificate signature last, assuming the Host has the Device Root CA's certificate pre-loaded.
5. If *Host\_DevCert* is valid, then the CableCARD Device extracts *Host\_ID* from the Host device certificate.
6. If *POD\_DevCert* is valid, then the Host extracts *POD\_ID* from the CableCARD device certificate.
7. The CableCARD Device extracts the Host's public key from the Host device certificate, and then uses it to verify the signature:  $SIGN_H(DH\_pubKeyH)$
8. The Host extracts the CableCARD Device's public key from the CableCARD device certificate, and then uses it to verify the signature:  $SIGN_P(DH\_pubKeyP)$ .
9. The CableCARD Device and the Host verify the RSA signature on these received messages and this proves that the messages were signed using the appropriate private key.
10. The CableCARD Device opens an MMI dialog to present the *POD\_ID* and *Host\_ID* to the subscriber, typically with a telephone number, for manual communication to the headend.
11. (Host → Cable Headend) The end-user will call the service provider and report the *Host\_ID* and *POD\_ID* and any additional data on the screen, if requested by the CSR, via telephone; see Section 3.2.5.1. Detailed operational requirements and message syntax are outside the scope of this document.
12. (Cable Headend CRL Checking: second phase of authentication) Check if *Host\_ID* and *POD\_ID* are in the CRLs. Validate the *Host\_ID* and *POD\_ID*. This check may not occur in real time; see Section 3.2.5.1.
13. (Cable Headend → CableCARD Device) Send EMM to authorize the CableCARD Device; see Section 3.2.5.1.
14. (Cable Headend → CableCARD Device) Headend sends validated ID(s) back to the CableCARD Device. Detailed operational requirements and message protocol/type/syntax are outside the scope of this document. See Section 3.2.5.1.
15. Host computes its Authentication Key *AuthKeyH* as described in Section 4.1 using shared Diffie-Hellman Secret Key (DHSK), the DH public keys *DH\_pubKeyP* and *DH\_pubKeyH*, as well as the *Host\_ID* and *POD\_ID* exchanged in the steps 1 and 2.
16. The CableCARD Device computes its Authentication Key, *AuthKeyP*, as described in Section 4.1 using shared Diffie-Hellman Secret Key (DHSK), the DH public keys *DH\_pubKeyP* and *DH\_pubKeyH*, as well as the *Host\_ID* and *POD\_ID* exchanged and validated in the previous messages.
17. The CableCARD Device sends a message to Host to request the *Authentication Key AuthKeyH* computed by the Host. This request message is implemented by the *object CP\_data\_req()* object, as detailed in Section 8.3.3.1 in SCTE 41 [1].
18. The Host sends its response *AuthKeyH* to the CableCARD Device by using the message *CP\_data\_cnf()*. This response message is detailed in Section 8.3.3.2 in SCTE 41 [1].
19. Complete Diffie-Hellman protocol to derive the shared DH key (*DH\_pubKeyP* and *DH\_pubKeyH*) in both CableCARD Device and Host. Diffie public Keys and authentication keys are used in the CP key derivation process.

### 8.3.2 Host Authentication Messages

The CableCARD Copy Protection System complies with Section 8.3.2 in SCTE 41 [1], with the exception that Section 8.3.2.2 of SCTE 41 is replaced by Section 8.3.2.2 below.

#### 8.3.2.1 *CP\_data\_req()* Syntax in Host Authentication Request Message

The CableCARD Copy Protection System complies with Section 8.3.2.1 in SCTE 41 [1].

#### 8.3.2.2 *CP\_data\_cnf()* Syntax in Host Authentication Response Message

This APDU object is issued by the Host to send its response data to the CableCARD Device. Host's certificate list (*Host\_DevCert* and *Host\_ManCert*), a signature of the Host Diffie-Hellman public key, and Diffie-Hellman public key (*DH\_pubKeyH*) are included in this message.

Host -> CableCARD Device:

$DH\_pubKey_H$ ,  $SIGN_H(DH\_pubKey_H)$ , *Host\_DevCert*, *Host\_ManCert*

The CableCARD Copy Protection System complies with Table 8.3-D in SCTE 41 [1].

### 8.3.3 Host Authentication Key Verification Messages

The CableCARD Copy Protection System complies with Section 8.3.3 in SCTE 41 [1].

## 8.4 Two-way System CableCARD CPS Message Protocol

### 8.4.1 Protocol Flow Overview

The CableCARD Copy Protection System complies with Section 8.4.1 in SCTE 41 [1].

### 8.4.2 Host Authentication Protocol Implementation

The CableCARD Copy Protection System complies with Section 8.4.2 in SCTE 41 [1], with the exception that Section 8.4.2.1 of SCTE 41 is replaced by Section 8.4.2.1 below.

#### 8.4.2.1 Two-way System CableCARD CPS Protocol Steps Full Authentication

1. The CableCARD Device initiates the authentication protocol by sending a challenge request to the Host. The CableCARD Device generates a secret random  $x$ ,  $1 \leq x \leq p-2$ , and sends the message. This challenge request contains the CableCARD Certificate List (*POD\_DevCert* and *POD\_ManCert*), CableCARD Diffie-Hellman public key ( $DH\_pubKey_P$ ) and a signature of  $DH\_pubKey_P$ . The request is implemented by the *CP\_data\_req()* object, as defined in APDU layer. The *CP\_data\_req()* message is detailed in Section 8.3.2.1.
2. After receiving *CP\_data\_req()*, Host generates a secret random  $y$ ,  $1 \leq y \leq p-2$ , and sends its reply with its Host Certificate List (*Host\_DevCert* and *Host\_ManCert*), Host Diffie-Hellman public key ( $DH\_pubKey_H$ ), and a signature of the  $DH\_pubKey_H$  to the CableCARD Device. This response is implemented by the object *CP\_data\_cnf()* object, as detailed in Section 8.3.2.2.
3. The CableCARD Device checks if Host certificate is valid by:
  - a. Checking the value of the certificate type or format field; and
  - b. The CableCARD Device verifies the Host's certificates (*Host\_DevCert* and *Host\_ManCert*), which is a sequence (chain) of X.509.v3 certificates, with the Host's certificate signature first followed by its Manufacturer Device CA's certificate signature second, and the CableLabs Manufacturer Root CA's certificate signature last, assuming the Host has the Device Root CA's certificate pre-loaded.
4. The Host checks if CableCARD certificate is valid by:

- a. Checking the value of the certificate type or format field; and
  - b. The Host verifies the CableCARD Device's certificates (*POD\_DevCert* and *POD\_ManCert*), which is a sequence (chain) of X.509.v3 certificates, with the CableCARD Device's certificate signature first followed by its Manufacturer Device CA's certificate signature second, and the Project Device CA's certificate signature last, assuming the Host has the Device Root CA's certificate pre-loaded.
5. If *POD\_DevCert* is valid, then the Host extracts *POD\_ID* from the CableCARD device certificate.
  6. If *Host\_DevCert* is valid, then the CableCARD Device extracts *Host\_ID* from the Host device certificate.
  7. The CableCARD Device extracts the Host's public key from the *Host\_Cert*, and then uses it to verify the signature:  $SIGN_H(DH\_pubKeyH)$ .
  8. The Host extracts the CableCARD Device's public key from the *POD\_Cert*, and then uses it to verify the signature:  $SIGN_P(DH\_pubKeyP)$ .
  9. The CableCARD Device and the Host verify the RSA signature on these received messages and this proves that the messages were signed using the appropriate private key.
  10. (CableCARD Device → Cable Headend) CableCARD Device sends at least the *POD\_ID* and *Host\_ID* to the headend in an authenticated message.
  11. (Cable Headend CRL Checking: second phase of authentication) Check if *POD\_ID* and *Host\_ID* are in the CRLs. Validate the *POD\_ID* and *Host\_ID* along with the Host/CableCARD Manufacturer CA information. This check may not occur in real time; see Section 3.2.5.2.
  12. (Cable Headend → CableCARD Device) Send EMM to authorize the CableCARD Device; see Section 3.2.5.2.
  13. (Cable Headend → CableCARD Device) Headend sends validated ID(s) back to the CableCARD Device. Detailed operational requirements and message protocol/type/syntax are outside the scope of this document. See Section 3.2.5.2.
  14. Host computes its Authentication Key *AuthKey<sub>H</sub>* by applying the SHA-1 hash function to shared Diffie-Hellman Secret Key (DHSK), the DH public keys *DH\_pubKey<sub>P</sub>* and *DH\_pubKey<sub>H</sub>*, as well as the *Host\_ID* and *POD\_ID* exchanged in step 1 and 2.
  15. The CableCARD Device computes Authentication Key, *AuthKey<sub>P</sub>* computed by applying the SHA-1 hash function to shared Diffie-Hellman Secret Key (DHSK), the DH public keys *DH\_pubKey<sub>P</sub>* and *DH\_pubKey<sub>H</sub>*, as well as the *Host\_ID* and *POD\_ID* exchanged and validated in the previous messages.
  16. The CableCARD Device sends a message to Host to request the Authentication Key *AuthKey<sub>H</sub>* computed by the Host. This request message is implemented by the object *CP\_data\_req()* object, as detailed in Section 8.3.3.1 in SCTE 41 [1].
  17. The Host sends its response *AuthKey<sub>H</sub>* to the CableCARD Device by using the message *CP\_data\_cnf()*. This response message is detailed in Section 8.3.3.2 in SCTE 41 [1].
  18. The CableCARD Device and Host complete the Diffie-Hellman protocol to derive the shared DH key (DHKey). The DHKey is used in the CP key derivation process.

## 8.5 CCI Simple Authentication Tunnel Protocol (SATP) Messages

The CableCARD Copy Protection System complies with Section 8.5 in SCTE 41 [1].

## Appendix A Luhn Check Digit (Normative)

The Luhn check digit is calculated over decimal values using the following algorithm<sup>5</sup>.

1. Double the value of alternate digits beginning with the first right hand digit (least significant digit) and moving left.
2. Add the individual digits comprising the products obtained in step 1 to each of the unaffected digits in the original number.
3. Subtract the total obtained in step 2 from the next higher number ending in 0. This is equivalent to calculating the “tens complement” of the low order digit of the total. If the total obtained in step 2 is a number ending in 0, then the check digit is 0.

Example:

For the 12-digit decimal value, 004,992,739,871, number the digits starting from the right (least significant digit):

digit #: 12, 11, 10, 9, 8, 7, 6, 5, 4, 3, 2, 1

‘odd’ digits: 0, 9, 2, 3, 8, 1

‘even’ digits: 0, 4, 9, 7, 9, 7

1. Multiply each odd # digit by 2:  
0, 9, 2, 3, 8, 1 → 0, 18, 4, 6, 16, 2
2. Add each individual digit of the products above to the ‘even’ digits:  
 $[0 + 1 + 8 + 4 + 6 + 1 + 6 + 2] + [0 + 4 + 9 + 7 + 9 + 7] = 64$
3. Subtract the sum from the next higher number ending in zero. If the result is 10, the check digit is zero.  
 $70 - 64 = 6$

The Luhn check digit for this example is “6”.

<sup>5</sup> Further information was available at <http://staff.sem.el.fi/~kribe/document/luhn.htm> as of 14 July 2000.

## **Appendix B Applying CP Key to DES Engine (Normative)**

This Appendix is replaced by Appendix B of SCTE 41 [1].

## **Appendix C CableCARD Copy Protection X.509 Certificate Profile and Management (Normative)**

The CableCARD Copy Protection System complies with Section 5 of the OpenCable System Security Specification [5].

## Appendix D Revision History (Informative)

Revision	Number	Description	Date
INT01	IS-POD-CP-INT01-000107	Editorial, table of POD input/output encryption vs. CCI, define POD 'Passthrough' state,	01/07/00
INT02	IS-POD-CP-INT02-000410	Incorporate ECN numbers: 32, 61, 62, 63, 64, 66, 70, 71, and 89.	04/10/00
INT03	IS-POD-CP-INT03-000714	ECN 31 to 32 above, new ECNs 65, 67, 68, 73 to 77, 113, 114a, 116, 121, 122, 140. Errors corrected in APDU numerical values of Section 8.	07/14/00
INT04	IS-POD-CP-INT04-010212	ECN-00097a, ECN-00112, ECN-00141, ECN-000148, ECN-000154, ECN-00176, ECN-00194, ECN-00202	03/07/01
INT05	IS-POD-CP-INT05-010515	ECN-00196	04/18/01
INT06	OC-SP-PODCP-IF-I06-011221	ECN-01225	12/21/01
I07	OC-SP-POD-CP-IF-I07-020524	ECN-01214a (Clarify Key Refresh Timeline) ECN-02-0242a (changes DTLA to X.509 certificates) ECN-02-0246a (Major edit to refer to SCTE 41 2001 standard)	05/24/02
I08	OC-SP-POD-CP-IF-I08-021126	ECN-02-0282, 02-0274, 02-0275, 02-0320	11/26/02
I09	OC-SP-PODCP-IF-I09-030210	ECN-02-0350; 02-0351; 02-0352	2/10/03
I10	OC-SP-PODCP-IF-I10-030707	ECN-03-0385, ECN-03-0396r1, ECN-03-0412, ECN-03-0419, ECN-03-0423, ECN-03-0427, ECN-03-0441	7/7/03
I11	OC-SP-CCCP-IF-I11-030905	Contains editorial changes, updating POD to CableCARD	9/5/03
I12	OC-SP-CCCP-IF-I12-031121	ECN-03-0453, ECN-03-0501, ECN-03-0505	11/21/03
I13	OC-SP-CCCP-IF-I13-040402	CCCP-IF-N-03.0530-5, CCCP-IF-N-04.0573-3	4/2/04
I14	OC-SP-CCCP-IF-I14-040831	CCCP-IF-N-04.0582-3, CCCP-IF-N-04.0638-2	8/31/04
I15	OC-SP-CCCP-IF-I15-041119	CCCP-IF-N-04.0697-1	11/5/04
C01	OC-SP-CCCP-IF-C01-050331	CCCP-IF-N-04.0732-2	3/15/05