

Superseded by New Specification CCCP2.0

OpenCable™ POD Copy Protection System

OC-SP-PODCP-IF-I10-030707

**ISSUED
SPECIFICATION**

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 2000-2003 Cable Television Laboratories, Inc. All rights reserved.

Document Status Sheet

Document Control Number:	OC-SP-PODCP-IF-I10-030707			
Document Title:	OpenCable™ POD Copy Protection System			
Revision History:	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			
Date:	July 7, 2003			
Responsible Editor:	Jeff Hamilton			
Status:	Work in Progress	Draft	Issued	Closed
Distribution Restrictions:	author only	CL/Member	CL/Member/ Vendor	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.

Contents

1	INTRODUCTION	1
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
2	SYSTEM OVERVIEW (INFORMATIVE)	2
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
2.5	Identifying Fraudulent Devices and Disabling of Services	2
3	HOST AUTHENTICATION MECHANISMS	3
3.1	Protocol Components	3
3.1.1	X.509 Version 3 Certificate	3
3.1.2	Device Parameters	3
3.1.3	System Parameters	3
3.1.4	Processing Basics	3
3.2	Host-POD Binding and Registration	3
3.2.1	ID Report-back Mechanism	3
3.2.2	Authentication Phase 1 – Certificate Verification & DH Key Exchange	4
3.2.3	Authentication Phase 2 – Headend Report Back	4
3.2.4	Authentication Phase 3 – Authentication Key Verification	4
3.2.5	Headend Report Back Methods	5
3.2.6	Handling of Interrupts to the Authentication Phases	5
3.3	Power-up Re-Authentication	5
3.4	POD Operation with Multiple Hosts	5
3.5	Host Operation with Multiple PODs	5
4	CRYPTOGRAPHIC FUNCTIONS	6
4.1	Authentication Key Generation	6
4.2	Copy Protection Key Generation	6
4.2.1	Basic Key Generation Protocol	6
4.2.2	POD Module Copy Protection Key	6
4.2.3	Host Copy Protection Key	7
4.3	Copy Protection Key Refresh	7

4.3.1	Key Session Period	7
4.3.2	Key Refresh Period	7
4.4	Diffie-Hellman Key Exchange Algorithm.....	7
4.5	SHA-1 Secure Hash Algorithm.....	7
4.6	Random Number Generation	7
4.7	DFAST Algorithm	7
4.8	RSA Digital Signatures	8
5	HOST SERVICE REVOCATION MECHANISMS	9
5.1	System Issues	9
5.2	Revocation Circumstances	9
5.3	Fraudulent POD and Host Identification	9
5.4	CA System Revocation & Selective Denial of Services.....	9
5.5	The Revocation Process	9
5.6	Implementation in the Headend.....	9
6	COPY CONTROL INFORMATION (CCI)	10
7	TRANSPORT ENCRYPTION FROM POD TO HOST	11
7.1	MPEG Scrambling	11
7.2	Transport Processing	11
7.3	Timing of Scrambling Mode Transitions.....	11
7.4	CP-Scrambling as a Function of CA-Scrambling and EMI Value.....	11
7.5	Debug Modes Prohibited	11
8	HOST, POD, & HEADEND MESSAGING PROTOCOLS	11
8.1	Message Protocol Overview	11
8.2	POD & Host Common Messages	11
8.2.1	Copy Protection Resource Identifier.....	11
8.3	One-way System Message Protocol.....	12
8.3.1	Protocol Flow Overview.....	12
8.3.2	Host Authentication Messages.....	13
8.3.3	Host Authentication Key Verification Messages.....	13
8.4	Two-way System POD CPS Message Protocol	13
8.4.1	Protocol Flow Overview.....	13
8.4.2	Host Authentication Protocol Implementation	13
8.5	CCI Simple Authentication Tunnel Protocol (SATP) Messages.....	15
APPENDIX A	LUHN CHECK DIGIT (NORMATIVE)	16
APPENDIX B	APPLYING CPKEY TO DES ENGINE (NORMATIVE).....	17

**APPENDIX C POD COPY PROTECTION X.509 CERTIFICATE PROFILE AND
MANAGEMENT (NORMATIVE)18**

APPENDIX D REVISION HISTORY (INFORMATIVE)19

List of Tables

Table 1 – Length of Keys and Parameters Used in the Key Generation 6

Table 2 – Copy Protection Resource Class 11

This page is intentionally left blank.

1 INTRODUCTION

1.1 Scope

The POD 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> 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-HOSTPOD-IF-I13-030707, OpenCable Host-POD Interface Specification, July 7, 2003, Cable Television Laboratories, Inc., <www.opencable.com>
- [5] OC-SP-SEC-I02-030707, OpenCable System Security Specification, July 7, 2003, Cable Television Laboratories, Inc., www.opencable.com

1.3 Acronyms and Abbreviations

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

1.4 Copy Protection System Components

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

1.5 Implementation Outline

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

1.6 Historical Perspective

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

2.2 Device Authentication

The POD Copy Protection System complies with Section 2.2 of SCTE 41 [1].

2.3 Key Exchange and Interface Encryption

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

2.4 Data Channel Protection

The POD Copy Protection System complies with Section 2.4 of SCTE 41 [1].

2.5 Identifying Fraudulent Devices and Disabling of Services

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

3 HOST AUTHENTICATION MECHANISMS

3.1 Protocol Components

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

3.1.1 X.509 Version 3 Certificate

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

3.1.2 Device Parameters

The POD Copy Protection System complies with section 3.1.2 of SCTE 41 [1].

3.1.3 System Parameters

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

3.1.4 Processing Basics

The POD Copy Protection System complies with Section 3.1.4 of SCTE 41 [1] with the exception that the first line of step 15 shall be modified to:

“15. If headend validation of the POD 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.”

3.2 Host-POD Binding and Registration

The POD 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 POD 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 POD module shall always include a POD-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 POD module 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 POD module shall define the Host as unidirectional. The POD module shall also have a means of determining if the system it is resident in is unidirectional.

Following power-up if the POD module determines it has not bound to the Host and is either in a unidirectional system or is inserted into a unidirectional Host, the POD module 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 POD module 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 POD module 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. This display shall be referred to as the “POD-Host binding information screen”.

The POD-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 POD requests display, e.g., at initial insertion and binding on or command by the headend.

The POD shall support display of the POD-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 data shall include:

An example of a displayed message follows, where the following information is reported back to the CAS:

1. POD_ID (13 decimal digits including the Luhn digit)
2. Host_ID (13 decimal digits including the Luhn digit)

In order to start service for this device
please contact SuperVision Cable at
1-800-555-8888

POD ID: xxx-xxx-xxx-xxxx

Host ID: xxx-xxx-xxx-xxxx

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

In order to support a Host request to display the POD-Host binding information screen, the POD shall support the “POD-Host binding information application” with application_type = 0x01. The POD-Host binding information application shall be defined strictly as display of the POD-Host binding information screen as defined in this section above. The POD shall only support one (1) POD-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 POD Copy Protection System complies with Section 3.2.2 of SCTE 41 [1].

3.2.3 Authentication Phase 2 – Headend Report Back

The POD Copy Protection System complies with Section 3.2.3 of SCTE 41 [1].

3.2.4 Authentication Phase 3 – Authentication Key Verification

The POD 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 POD 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 POD 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 POD-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 POD module shall display the POD and Host ID's, for example:

POD ID: xxx-xxx-xxx-xxxx
Host ID: xxx-xxx-xxx-xxxx

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

3.2.5.3 Two-way System POD and Host ID Validation

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

3.2.6 Handling of Interrupts to the Authentication Phases

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

3.3 Power-up Re-Authentication

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

3.4 POD Operation with Multiple Hosts

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

3.5 Host Operation with Multiple PODs

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

4.2 Copy Protection Key Generation

The POD 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

The POD Copy Protection System complies with Section 4.2.1 of SCTE 41 [1].

4.2.2 POD Module 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 *K_s*:

$$K_s = \text{SHA-1} [\text{AuthKey}_P \mid \{DHKey\} \mid N_{Host} \mid N_{module}]_{\text{LSB } 128}$$

Detailed information on how to generate *AuthKey_P* is described in section 4.1. SHA-1 is used as a cryptographic compression function to generate a seed with the proper 128-bit length for the input to the DFAST engine. The DFAST algorithm is applied to *K_s* to produce the 56-bit value of the Copy Protection Key, also known as *K_{s_dfast}*:

$$\text{CP-Key} = K_{s_dfast} = \text{DFAST} [K_s]$$

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 _H , AuthKey _P)	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 <i>K_s</i>	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 POD and Host.
Copy Protection Key (<i>K_{s_dfast}</i>)	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 (*AuthKeyH*), the 1024-bit shared secret DH key (*DHKey*), and the random numbers exchanged in the key generation. This key is named *Ks*:

$$Ks = \text{SHA-1} [AuthKeyH | \{DHKey\} | N_Host | N_module] \text{ the first 16 bytes}$$

Detailed information on how to generate *AuthKeyH* is described in section 4.1. SHA-1 is used as a cryptographic compression function to generate a seed 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 POD Copy Protection System complies with Section 4.3 of SCTE 41 [1].

4.3.1 Key Session Period

The POD Copy Protection System complies with Section 4.3.1 of SCTE 41 [1].

4.3.2 Key Refresh Period

The POD Copy Protection System complies with Section 4.3.2 of SCTE 41 [1].

4.4 Diffie-Hellman Key Exchange Algorithm

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

4.5 SHA-1 Secure Hash Algorithm

The POD Copy Protection specification employs the RSA signature algorithm with SHA-1 for all X.509 digital certificates. The POD 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 must 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 POD 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.
- POD 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 POD Copy Protection System complies with Section 4.6 of SCTE 41 [1].

4.7 DFAST Algorithm

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

4.8 RSA Digital Signatures

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

5 HOST SERVICE REVOCATION MECHANISMS

5.1 System Issues

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

5.2 Revocation Circumstances

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

5.3 Fraudulent POD and Host Identification

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

5.4 CA System Revocation & Selective Denial of Services

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

5.5 The Revocation Process

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

5.6 Implementation in the Headend

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

6 COPY CONTROL INFORMATION (CCI)

The POD Copy Protection System complies with Section 6 in SCTE 41 [1].

7 TRANSPORT ENCRYPTION FROM POD TO HOST

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

7.1 MPEG Scrambling

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

7.2 Transport Processing

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

7.3 Timing of Scrambling Mode Transitions

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

7.5 Debug Modes Prohibited

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

8 HOST, POD, & HEADEND MESSAGING PROTOCOLS

8.1 Message Protocol Overview

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

8.2 POD & Host Common Messages

The POD Copy Protection System complies with Section 8.2 in SCTE 41 [1] with the exception that Table 8.2-C is superseded by Table 2 – Copy Protection Resource Class below.

Table 2 – Copy Protection Resource Class

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

8.2.1 Copy Protection Resource Identifier

The POD Copy Protection System complies with Section 7.1 in the Host POD 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 POD Copy Protection System complies with Section 8.3.1 in SCTE 41 [1].

8.3.1.1 Authentication Protocol Implementation

The POD 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 POD CPS Protocol Steps Full Authentication

1. The POD Module initiates the authentication protocol by sending a challenge request to Host. The POD generates a secret random x , $1 \leq x \leq n-2$, and sends Host message. This message contains the POD 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 0.
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 POD Module. This response is implemented by the object *CP_data_cnf()* object, as detailed in section 8.3.2.2.
3. The POD Module checks if Host Certificate List is valid by:
 - a) Checking the value of the certificate type or format field; and
 - b) The POD 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 POD certificate is valid by:
 1. Checking the value of the certificate type or format field; and
 2. The Host verifies the POD's certificates (*POD_DevCert* and *POD_ManCert*), which is a sequence (chain) of X.509.v3 certificates, with the POD'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 POD extracts *Host_ID* from the Host device certificate.
6. If *POD_DevCert* is valid, then the Host extracts *POD_ID* from the POD device certificate.
7. The POD 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 POD's public key from the POD device certificate, and then uses it to verify the signature: $SIGN_P(DH_pubKeyP)$.
9. The POD 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 POD Module 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 -> POD) Send EMM to authorize the POD; see section 3.2.5.1.
14. (Cable Headend -> POD) Headend sends validated ID(s) back to the POD Module. 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 POD 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 POD 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 POD Module 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 POD and Host. Diffie public Keys and authentication keys are used in the CP key derivation process.

8.3.2 Host Authentication Messages

The POD 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 POD 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 POD. 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 -> POD:

$DH_pubKeyH$, $SIGN_H(DH_pubKeyH)$, *Host_DevCert*, *Host_ManCert*

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

8.3.3 Host Authentication Key Verification Messages

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

8.4 Two-way System POD CPS Message Protocol

8.4.1 Protocol Flow Overview

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

8.4.2 Host Authentication Protocol Implementation

The POD 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 POD CPS Protocol Steps Full Authentication

1. The POD Module initiates the authentication protocol by sending a challenge request to the Host. The POD generates a secret random x , $1 \leq x \leq p-2$, and sends the message. This challenge request contains the POD Certificate List (*POD_DevCert* and *POD_ManCert*), POD Diffie-Hellman public key (*DH_pubKeyP*) and a signature of *DH_pubKeyP*. 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_pubKeyH*), and a signature of the *DH_pubKeyH* to the POD Module. This response is implemented by the object *CP_data_cnf()* object, as detailed in section 8.3.2.2.
3. The POD Module checks if Host certificate is valid by:
 - a. Checking the value of the certificate type or format field; and
 - b. The POD 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 POD certificate is valid by:
 - a. Checking the value of the certificate type or format field; and
 - b. The Host verifies the POD's certificates (*POD_DevCert* and *POD_ManCert*), which is a sequence (chain) of X.509.v3 certificates, with the POD'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 POD device certificate.
6. If *Host_DevCert* is valid, then the POD extracts *Host_ID* from the Host device certificate.
7. The POD 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 POD's public key from the *POD_Cert*, and then uses it to verify the signature: *SIGN_P* (*DH_pubKeyP*).
9. The POD 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. (POD → Cable Headend) POD 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/POD Manufacturer CA information. This check may not occur in real time; see section 3.2.5.2.
12. (Cable Headend → POD) Send EMM to authorize the POD; see section 3.2.5.2.
13. (Cable Headend → POD) Headend sends validated ID(s) back to the POD Module. 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_H* by applying the SHA-1 hash function to shared Diffie-Hellman Secret Key (DHSK), the DH public keys *DH_pubKey_P* and *DH_pubKey_H*, as well as the *Host_ID* and *POD_ID* exchanged in step 1 and 2.

15. The POD computes Authentication Key, $AuthKey_P$ computed by applying the SHA-1 hash function to shared Diffie-Hellman Secret Key (DHSK), the DH public keys DH_pubKey_P and DH_pubKey_H , as well as the Host_ID and POD_ID exchanged and validated in the previous messages.
16. The POD sends a message to Host to request the Authentication Key $AuthKey_H$ 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 $AuthKey_H$ to the POD Module by using the message $CP_data_cnf()$. This response message is detailed in section 8.3.3.2 in SCTE 41 [1].
18. The POD 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 POD Copy Protection System complies with Section 8.5 in SCTE 41 [1].

Appendix A Luhn Check Digit (Normative)

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

Appendix B Applying CPKey to DES Engine (Normative)

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

Appendix C POD Copy Protection X.509 Certificate Profile and Management (Normative)

The POD 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