

Superseded

Data-Over-Cable Service Interface Specifications

Baseline Privacy Interface Specification

SP-BPI-I02-990319

INTERIM

Notice

This document is a cooperative effort undertaken at the direction of Cable Television Laboratories, Inc. (CableLabs®) for the benefit of the cable industry in general. Neither CableLabs nor any member company is responsible for any liability of any nature whatsoever resulting from or arising out of use or reliance upon this specification by any party. This document is furnished on an "AS IS" basis and neither CableLabs nor its members provide any representation or warranty, express or implied, regarding its accuracy, completeness, or fitness for a particular purpose. Distribution of this document is restricted pursuant to the terms of separate agreements negotiated with each of the parties to whom this document has been furnished.

© Copyright 1999 Cable Television Laboratories, Inc.
All rights reserved.

Document Status Sheet

| | | | |
|------------------------------------------------------------|----------------------------|-----------------------------|-----------------------|
| Document Control Number: SP-BPI-I02-990319 | | | |
| Reference: Baseline Privacy Interface Specification | | | |
| Revision History: | | | |
| I01 | 9/22/97 | Released for publication | |
| I02 | 3/19/99 | Re-released for publication | |
| Date: March 19, 1999 | | | |
| Status Code: | Work in Process | Draft | Interim |
| | | | Released |
| Distribution Restrictions: | CableLabs-only | GL/ Authors | GL/ Vendor |
| | | | Public |

Key to Document Status Codes

- Work in Process** 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 MCNS and vendors. Drafts are susceptible to substantial change during the review process.

- Interim** A document which has undergone rigorous MCNS and vendor review, suitable for use by vendors to design in conformance with, and suitable for field testing.

- Released** A stable document, reviewed, tested and validated, suitable to enable cross-vendor interoperability.

| | | |
|----------|--------------------------------------------------------------------|-----------|
| 1 | SCOPE AND PURPOSE | 1 |
| 1.1 | SCOPE | 1 |
| 1.2 | REQUIREMENTS | 1 |
| 1.3 | BACKGROUND | 1 |
| 1.3.1 | <i>Service Goals</i> | 1 |
| 1.3.2 | <i>Reference Architecture</i> | 2 |
| 2 | BASELINE PRIVACY OVERVIEW | 7 |
| 2.1 | OPERATIONAL OVERVIEW | 8 |
| 2.1.1 | <i>Cable Modem Initialization</i> | 8 |
| 2.1.2 | <i>Cable Modem Key Update Mechanism</i> | 9 |
| 3 | VARIABLE-LENGTH PACKET PDU FORMAT | 11 |
| 4 | BASELINE PRIVACY KEY MANAGEMENT (BPKM) PROTOCOL | 15 |
| 4.1 | STATE MODELS | 15 |
| 4.1.1 | <i>Introduction</i> | 15 |
| 4.1.2 | <i>Authorization State Machine</i> | 17 |
| 4.1.3 | <i>TEK State Machine</i> | 25 |
| 4.2.1 | <i>Packet Formats</i> | 33 |
| 4.2.2 | <i>BPKM Attributes</i> | 40 |
| 5 | KEY USAGE | 55 |
| 5.1 | CABLE MODEM | 55 |
| 5.2 | CMTS..... | 55 |
| 5.3 | MULTICAST KEY HANDLING..... | 57 |
| 6 | CRYPTOGRAPHIC METHODS | 59 |
| | APPENDIX A. : TFTP CONFIGURATION FILE EXTENSIONS | 61 |
| | APPENDIX B. : EXAMPLE MESSAGES AND PDUS (INFORMATIVE) | 67 |
| | APPENDIX C. : REFERENCES | 85 |
| | APPENDIX D. : REVISIONS | 87 |

| | | |
|-------------|-------------------------------------------------------------------|----|
| FIGURE 1-1. | TRANSPARENT IP TRAFFIC THROUGH THE DATA-OVER-CABLE SYSTEM | 2 |
| FIGURE 1-2. | DATA-OVER-CABLE REFERENCE ARCHITECTURE | 3 |
| FIGURE 3-1. | FORMAT OF ETHERNET/802.3 PACKET PDU WITH PRIVACY EH ELEMENT | 11 |
| FIGURE 4-1. | AUTHORIZATION STATE MACHINE FLOW DIAGRAM..... | 18 |
| FIGURE 4-2. | TEK STATE MACHINE FLOW DIAGRAM..... | 26 |

| | | |
|-------------|-----------------------------------------------------------------------|----|
| TABLE 3-1. | BASELINE PRIVACY EH ELEMENT FORMATS..... | 13 |
| TABLE 4-1. | AUTHORIZATION FSM STATE TRANSITION MATRIX | 19 |
| TABLE 4-2. | TEK FSM STATE TRANSITION MATRIX | 27 |
| TABLE 4-3. | BASELINE PRIVACY KEY MANAGEMENT MAC MESSAGES..... | 33 |
| TABLE 4-4. | BASELINE PRIVACY KEY MANAGEMENT MESSAGE CODES..... | 34 |
| TABLE 4-5. | AUTHORIZATION REQUEST ATTRIBUTES..... | 36 |
| TABLE 4-6. | AUTHORIZATION REPLY ATTRIBUTES | 36 |
| TABLE 4-7. | AUTH REJ ATTRIBUTES | 37 |
| TABLE 4-8. | KEY REQUEST ATTRIBUTES | 37 |
| TABLE 4-9. | KEY REPLY ATTRIBUTES..... | 38 |
| TABLE 4-10. | KEY REJECT ATTRIBUTES | 39 |
| TABLE 4-11. | AUTHORIZATION INVALID ATTRIBUTES | 40 |
| TABLE 4-12. | TEK INVALID ATTRIBUTES | 40 |
| TABLE 4-13. | BPKM ATTRIBUTE TYPES..... | 41 |
| TABLE 4-14. | ATTRIBUTE VALUE DATA TYPES | 42 |
| TABLE 4-15. | TEK-PARAMETERS SUB-ATTRIBUTES..... | 50 |
| TABLE 4-16. | ERROR-CODE ATTRIBUTE CODE VALUES | 53 |
| TABLE A-1. | RECOMMENDED OPERATIONAL RANGES FOR BPI CONFIGURATION PARAMETERS | 64 |
| TABLE A-2. | SHORTENED BPI PARAMETER VALUES FOR PROTOCOL TESTING..... | 65 |

1 Scope and Purpose

1.1 Scope

The intent of this specification is to describe a simple Data Privacy function for CMTS - CM communications in the Data-Over-Cable system. While there exists a requirement for secure communications over the cable network in order to protect broadcast content and other high value data transmissions, this specification defines a privacy function that provides a means of Data Privacy protection from a service provider's network access devices.

1.2 Requirements

Throughout this document, the words that are used to define the significance of particular requirements are capitalized. These words are:

- “MUST” This word or the adjective “REQUIRED” means that the item is an absolute requirement of this specification.
- “MUST NOT” This phrase means that the item is an absolute prohibition of this specification.
- “SHOULD” This word or the adjective “RECOMMENDED” means that there may exist valid reasons in particular circumstances to ignore this item, but the full implications should be understood and the case carefully weighed before choosing a different course.
- “SHOULD NOT” This phrase means that there may exist valid reasons in particular circumstances when the listed behavior is acceptable or even useful, but the full implications should be understood and the case carefully weighed before implementing any behavior described with this label.
- “MAY” This word or the adjective “OPTIONAL” means that this item is truly optional. One vendor may choose to include the item because a particular marketplace requires it or because it enhances the product, for example; another vendor may omit the same item.

Other text is descriptive or explanatory.

1.3 Background

1.3.1 Service Goals

Cable operators are interested in deploying high-speed data communications systems on cable television systems that are capable of supporting a wide variety of services. Services under

consideration by cable operators include packet telephony service, video conferencing service, T1/frame relay equivalent service, and many others. To this end, CableLabs' member companies decided to prepare a series of interface specifications that will permit the early definition, design, development and deployment of data-over-cable systems on a uniform, consistent, open, non-proprietary, multi-vendor interoperable basis.

The intended service will allow transparent bi-directional transfer of Internet Protocol (IP) traffic, between the cable system headend and customer locations, over an all-coaxial or hybrid fiber/coax (HFC) cable television network. This is shown in simplified form in Figure 1-1.

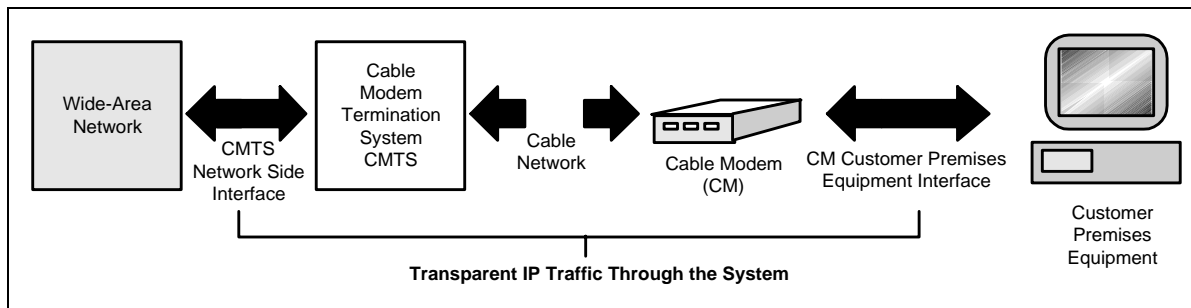


Figure 1-1. Transparent IP Traffic Through the Data-Over-Cable System

The transmission path over the cable system is realized at the headend by a CMTS, and at each customer location by a CM. At the headend (or hub), the interface to the data-over-cable system is called the Cable Modem Termination System - Network-Side Interface (CMTS-NSI) and is specified in [MCNS3]. At the customer locations, the interface is called the cable-modem-to-customer-premise-equipment interface (CMCI) and is specified in [MCNS4]. The intent is for the MCNS operators to transparently transfer IP traffic between these interfaces, including but not limited to datagrams, DHCP, ICMP, and IP Group addressing (broadcast and multicast).

1.3.2 Reference Architecture

The reference architecture for the data-over-cable services and interfaces is shown in Figure 1-2.

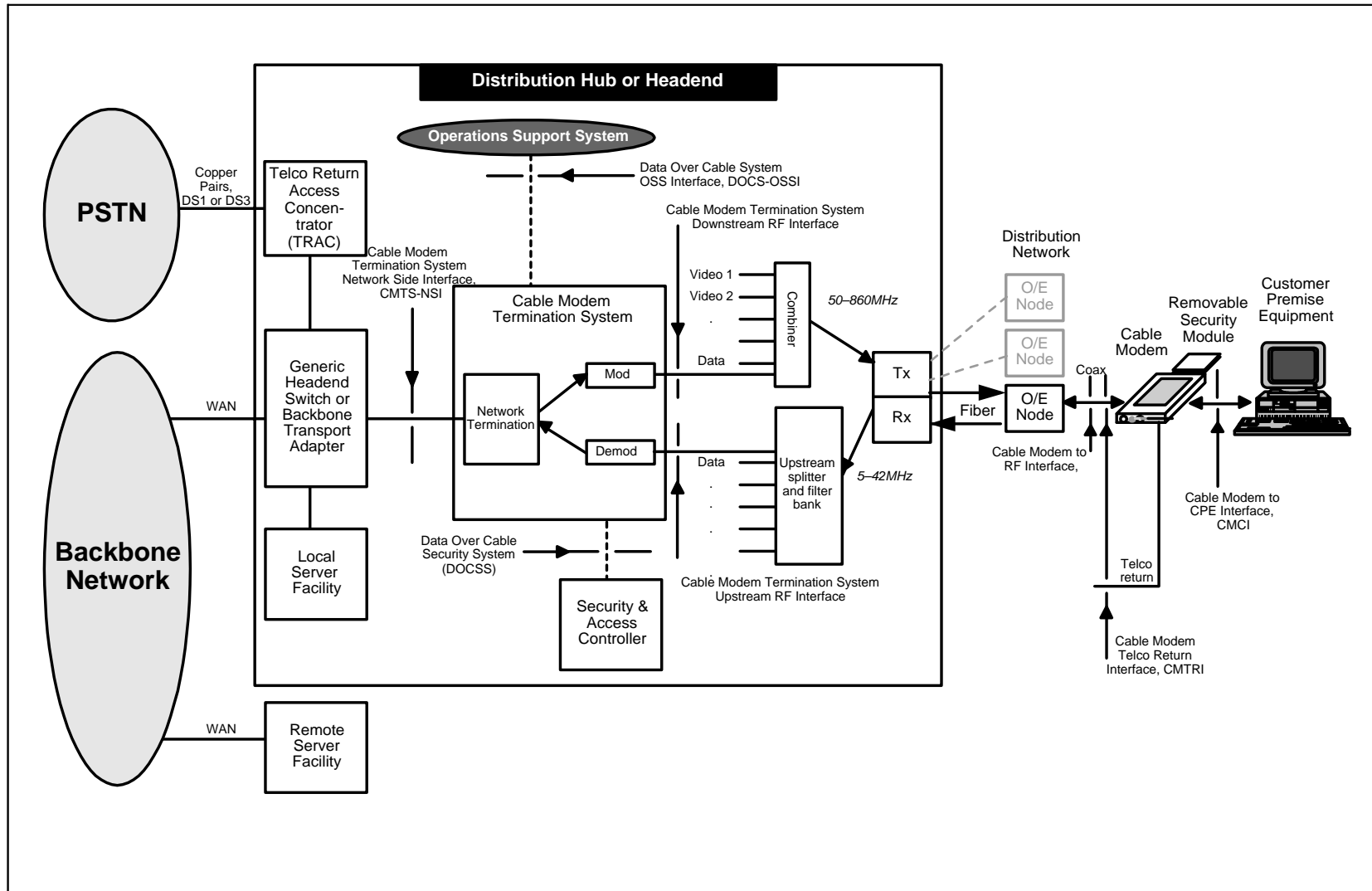


Figure 1-2. Data-over-Cable Reference Architecture

1.3.2.1 Categories of Interface Specification

The basic reference architecture of Figure 1-2 involves three categories of interfaces.

a) Category 1

Data Interfaces - These are the CMCI [MCNS4] and CMTS-NSI [MCNS3], corresponding respectively to the cable modem to customer-premises-equipment (CPE) interface (for example between the customer's computer and the cable modem), and the cable modem termination system network side interface between the cable modem termination system and the data network.

b) Category 2

Operations Support System Interfaces - OSSI - This is the element management layer interface between the network elements and higher level OSSs which support the basic business processes and are documented in [MCNS5].

Telephony Return Interface - CMTRI - This is the interface between the cable modem and a telephone return path, for use in cases where the return path is not provided or not available via the HFC network and is documented in [MCNS6].

c) Category 3

RF Interfaces -

- Between the cable modem and the HFC network, [MCNS1].
- Between the CMTS and the HFC network, in the downstream direction (traffic toward the customer) [MCNS1].
- Between the CMTS and the HFC network, in the upstream direction (traffic from the customer) [MCNS1].

Security requirements -

- The Data-Over-Cable Baseline Privacy interface specification is this document.

1.3.2.2 Data-Over-Cable Interface Documents

A list of the documents in the Data-Over-Cable Interface Specifications family is provided below. For update, please refer to URL <http://www.cablemodem.com>.

| Designation | Title |
|--------------------|---------------------------------------------------------------------|
| SP-BPI | Baseline Privacy Interface Specification |
| SP-CMCI | Cable Modem-to-Customer Premises Equipment Interface Specification |
| SP-CMTRI | Cable Modem Telephony Return Interface Specification |
| SP-CMTS-NSI | Cable Modem Termination System Network Side Interface Specification |
| SP-OSSI | Operations Support System Interface Specification |
| SP-OSSI-RF | Operations Support System Interface Radio Frequency MIB |
| SP-OSSI-TR | Operations Support Systems Interface Telephony Return MIB |
| SP-RFI | Radio Frequency Interface Specification |
| TR-OSSF | Operations Support System Framework |

Key to Designation:

SP Specification

TR Technical Report (provides a context for understanding and applying the specification.)

This page intentionally left blank.

2 Baseline Privacy Overview

Baseline Privacy provides cable modem users with data privacy across the RF network. It provides user data privacy by encrypting traffic flows between CM and CMTS.

A secondary goal of Baseline Privacy is to provide cable operators with basic protection from theft of service. To provide *strong* protection against theft of service, the CMTS would need to authenticate client CMs: the CMTS would need to establish a unique, verifiable identity for a cable modem and authenticate key requests claiming to be from that cable modem. In addition, the CMTS would need to link the cable modem identity to a paying subscriber, and to the data services that subscriber is authorized to access.

Since data privacy is Baseline Privacy's principal service goal, and given neither CM nor CMTS authentication are prerequisite for providing user data privacy, Baseline Privacy's key distribution protocol does not authenticate CM and CMTS (i.e., it does not employ authentication mechanisms such as passwords or digital signatures). In the absence of authentication, Baseline Privacy provides basic protection of service by insuring that a modem, uniquely identified by its 48-bit IEEE MAC address, can only obtain keying material for services it is authorized to access. Since it does not authenticate CMs, it cannot protect against an attacker employing a *cloned* modem, masquerading as an authorized modem.

Baseline Privacy security services are defined as a set of extended services within the MCNS Media Access Control (MAC) sublayer. Packet header information specific to Baseline Privacy is located in a Privacy Extended Header (EH) element in the MAC Extended Header, as defined in [MCNS1]. Figure 3-1 depicts an MCNS MAC Packet Data PDU containing the Privacy EH element. Two new MAC management message types, BPKM-REQ and BPKM-RSP, have been defined to support the Baseline Privacy Key Management (BPKM) protocol.

Baseline Privacy uses the Cipher Block Chaining (CBC) mode of the US Data Encryption Standard (DES) algorithm [FIPS-46, FIPS-46-1, FIPS-74, FIPS-81] to encrypt the Packet PDU field in both upstream and downstream RF MAC Packet Data PDUs. The MCNS MAC headers of these Packet Data PDUs **MUST NOT** be encrypted. The payloads, as well as headers, of MAC management messages **MUST** be sent in the clear to facilitate registration, ranging, and normal operation of the MAC sublayer.

Baseline Privacy extends the definition of the MAC sublayer's Service ID (SID). The MCNS Radio Frequency Interface Specification [MCNS1] defines a SID as a mapping between CM and CMTS for the purposes of upstream bandwidth allocation and class-of-service management. In this context, the SID only has upstream significance. When Baseline Privacy is in operation, the SID also identifies a particular security association and, thus, has both upstream and downstream significance. A downstream multicast traffic flow, then, which normally would have no SID associated with it, will have an associated SID when Baseline Privacy is operational. The Privacy Extended Header Element includes the SID associated with the MAC Packet Data PDU; the SID, in combination with other components of the extended header element, identifies to a modem the keying material required to decrypt the MAC PDU's Packet Data field.

Baseline Privacy's key management protocol runs between CM and CMTS; CMs use the protocol to obtain authorization and traffic keying material (pertaining to a particular SID) from the CMTS, and to support periodic reauthorization and key refresh. The key management protocol uses RSA [RSA, RSA1], a public-key encryption algorithm, and the Electronic Codebook (ECB) mode of DES [FIPS-81] to secure key exchanges between CM and CMTS. CMs MUST have factory-installed RSA private/public key pairs, or provide an internal algorithm to generate such key pairs dynamically. If a CM relies on an internal algorithm to generate its private/public key pair, the CM MUST generate the key pair prior to its first Baseline Privacy establishment, defined in Section 2.1.1. Internal key pair generation MUST be a one-time-only operation: once a key pair is generated, it MUST be retained for the operational life of the CM.

A SID's keying material (DES key and CBC Initialization Vector) has a limited lifetime. When the CMTS delivers SID keying material to a CM, it also provides the CM with that material's remaining lifetime. It is the responsibility of the CM to request new keying material from the CMTS before the set of keying material the CM currently has expired at the CMTS.

2.1 Operational Overview

2.1.1 Cable Modem Initialization

[MCNS1] divides cable modem initialization into the following sequence of tasks:

- scan for downstream channel and establish synchronization with the CMTS
- obtain transmit parameters
- perform ranging
- establish IP connectivity (DHCP)
- establish time of day
- transfer operational parameters (download parameter file via TFTP)
- CMTS Registration

Baseline Privacy establishment follows CMTS registration.

If a CM is to run Baseline Privacy, its parameter file, downloaded during the transfer of operational parameters, MUST include Baseline Privacy Configuration Settings. These additional configuration settings are defined in Appendix A.

Upon completing CMTS registration, the CMTS will have assigned Service IDs (SIDs) to the registering CM that match the CM's class-of-service provisioning. If a CM is configured to run Baseline Privacy, CMTS registration is immediately followed by initialization of the CM's Baseline Privacy security functions.

Baseline Privacy initialization begins with the CM sending the CMTS an authorization request, containing data identifying the CM (e.g., MAC address), the CM's RSA public key,

and a list of zero or more assigned unicast SIDs that have been configured to run Baseline Privacy. (The list would be empty if a cable modem was configured to only run Baseline Privacy on multicast SIDs.)

If the CMTS determines the requesting CM is authorized for these services, the CMTS responds with an authorization reply containing a list of SIDs (both unicast and multicast) that the CM is permitted to run Baseline Privacy on. The reply also includes an authorization key from which CM and CMTS derive the keys needed to secure a CM's subsequent requests for per-SID traffic encryption keys, and the CMTS's responses to these requests. The authorization key is encrypted with the receiving cable modem's public key.

After successfully completing authorization with the CMTS, the cable modem sends key requests to the CMTS, requesting traffic encryption keys to use with each of its Baseline Privacy SIDs. A CM's traffic key requests are authenticated using a keyed hash (the HMAC algorithm [RFC2104]); the message authentication key is derived from the authorization key obtained during the earlier authorization exchange. The CMTS responds with key replies, containing the traffic encryption keys; the keys are DES encrypted with a key encryption key derived from the authorization key. Like the Key Requests, Key Replies are authenticated with a keyed hash, where the message authentication key is derived from the authorization key.

2.1.2 Cable Modem Key Update Mechanism

The traffic encryption keys which the CMTS provides to client CMs have a limited lifetime. The CMTS delivers a key's remaining lifetime, along with the key value, in the key replies it sends to its client CMs. The CMTS controls which keys are current by flushing expired keys and generating new keys. It is the responsibility of individual cable modems to insure the keys they are using match those the CMTS is using. Cable modems do this by tracking when a particular SID's key is scheduled to expire and issuing a new key request for the latest key prior to that expiration time.

In addition, cable modems are required to periodically reauthorize with the CMTS; as is the case with traffic encryption keys, an authorization key has a finite lifetime which the CMTS provides the CM along with the key value. It is the responsibility of individual cable modems to reauthorize and obtain (1) a new authorization key and (2) a current list of supported SIDs before the CMTS expires their current authorization key.

Baseline Privacy initialization and key update is implemented within the Baseline Privacy Key Management protocol, defined in detail in Section 4.

This page intentionally left blank.

3 Variable-Length Packet PDU Format

Figure 3-1 depicts the format of an MCNS variable-length Packet Data PDU with a Privacy Extended Header (EH) element and encrypted Packet PDU payload. The first 12 octets of the Packet PDU, containing the Ethernet/802.3 destination and source addresses (DA/SA), are not encrypted. Transmitting a frame's destination and source addressing in the clear provides vendors with greater flexibility in how they integrate encryption/decryption with MAC functionality; e.g., vendors have freedom to choose between filtering on DA/SA or SID first. The Packet PDU's Ethernet/802.3 CRC is encrypted.



Figure 3-1. Format of Ethernet/802.3 Packet PDU with Privacy EH Element

The CMTS includes the Privacy EH element in all downstream Packet Data PDUs it encrypts under Baseline Privacy. Similarly, a CM includes the Privacy EH element in all upstream Packet Data PDUs it encrypts under Baseline Privacy.

The Privacy Extended Header element employs two EH element type values, BPI_UP and BPI_DOWN, for use with upstream and downstream Packet Data PDUs, respectively. [MCNS1] defines the specific EH element type values assigned to BPI_UP and BPI_DOWN.

The Privacy EH element's value field is 4-bytes long. The high-order 4 bits of the value field's first byte contains a key sequence number, KEY_SEQ. Recall that the keying material associated with a Privacy-enabled SID has a limited lifetime, and that it is the cable modem's responsibility to request and obtain updated keying material from the CMTS before the current material expires. The CMTS distributes its latest keying material in response to a

client CM's traffic key requests. The CMTS manages a 4-bit key sequence number independently for each SID and distributes this key sequence number along with the SID's keying material. The CMTS increments the key sequence number with each new generation of keying material. The Privacy EH element includes this sequence number, along with the SID, to identify the specific generation of that SID's keying material being used to encrypt the attached Packet PDU. Being a 4-bit quantity, the sequence number wraps around to 0 when it reaches 15.

Comparing a received frame's key sequence number with the "current" key sequence number, a CM or CMTS can easily recognize a loss of key synchronization with its peer. A CM **MUST** maintain a minimum of two keys (and their respective CBC Initialization Vectors) for each Privacy-enabled SID. Keeping on-hand at least the two most recent key generations is necessary to maintain uninterrupted multicast service during a multicast SID's key transition.

The 4 bits following KEY_SEQ contain a protocol version number, which for this initial release will be set to 1.

The next two bytes contain the 2 bits of encryption status and the 14-bit SID. The ENABLE encryption status bit indicates whether encryption is enabled or disabled for that PDU. If the ENABLE bit is 0, the Packet PDU is not encrypted and the Privacy EH element **MUST** be ignored (with the exception of the optional piggybacked bandwidth request—see below). The TOGGLE bit **MUST** match the state of the Least Significant Bit (LSB) of KEY_SEQ, the Key Sequence Number.

The RF MAC protocol [MCNS1] defines a Request EH element for piggybacking a bandwidth request on a data transmission. Baseline Privacy defines an additional mechanism for piggybacking bandwidth requests: the final byte of Baseline Privacy's upstream EH element (EH element type = BPI_UP) is used to carry an optional piggybacked bandwidth allocation request. If there is a piggybacked request, the byte represents the number of requested mini-slots or ATM cells. The 14-bit SID within the Privacy EH element identifies the Service ID making the request. If there is no piggybacked request within the Privacy EH element, the request byte is set to zero. A piggybacked request within the Privacy EH element **MUST** be processed regardless of the status of the ENABLE bit.

In downstream packets (extender header element type = BPI_DOWN) the final byte is reserved and set to zero.

When operating Baseline Privacy, the two EH Element types, one for upstream, one for downstream, **MUST** be used on encrypted Packet Data PDUs. Table 3-1 summarizes the contents of the two Baseline Privacy EH Elements

Table 3-1. Baseline Privacy EH Element Formats

| EH_TYPE | EH_LEN | EH_VALUE |
|------------------------------|---------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| BPI__UP See [MCNS1] | 4 | KEY_SEQ (4 bits), Version (4 bits), SID (2 bytes), Request [piggyback] (1 byte) [CM --> CMTS] KEY_SEQ field (4 bits): Key sequence number Version field (4 bits) is defined as: 0x1 SID field is defined as: Bit[15]: ENABLE: 1..Encryption enabled; 0..Encryption Disabled Bit[14]: TOGGLE: 1..Odd Key; 0..Even Key Bit[13:0]: Service ID. Request field contains the number of mini-slots requested for upstream bandwidth. |
| BPI__DOWN See [MCNS1] | 4 | KEY_SEQ (4 bits), Version (4 bits), SID (2 bytes), Reserved (1 byte) [CMTS --> CM] KEY_SEQ field (4 bits) :Key sequence number Version field (4 bits) is defined as: 0x1 SID field is defined as: Bit[15]: ENABLE: 1..Encryption enabled; 0..Encryption Disabled Bit[14]: TOGGLE: 1..Odd Key; 0..Even Key Bit[13:0]: Service ID. Reserved field is set to 0. |

The format of the extended header element remains the same whether or not encryption is used for that particular frame and whether or not there is a piggybacked request. In the case that encryption is not used for that PDU, ENABLE, Bit[15] of the SID field, MUST be set to zero. If there is a piggybacked request, the request byte represents the number of mini-slots or ATM cells required.

If Baseline Privacy Encryption is being applied to a particular traffic session (either a unicast bidirectional flow or a multicast downstream flow), then all Packet Data PDUs associated with that flow, including any upstream Packet Data PDUs transmitted in contention intervals, MUST include a Privacy Extended Header element. If there are multiple Extended Header elements present, the Data Privacy Header element MUST be first.

In the case of encrypted transmissions in an upstream contention interval, the SID in the Data Privacy Header MUST identify the security association; it MUST NOT be set to the Request/Data contention interval's well-known Multicast Service ID.

If Privacy is not enabled on a particular traffic flow (e.g., a downstream multicast service), the Privacy Extended Header element **SHOULD NOT** be used.

4 Baseline Privacy Key Management (BPKM) Protocol

4.1 State Models

4.1.1 Introduction

The BPKM protocol is specified by two separate, but interdependent, state models: an authorization state model (the Authorization state machine) and an operational service key state model (the Traffic Encryption Key, or *TEK* state machine). These state models are for explanatory purposes only, and should not be construed as constraining an actual implementation.

Cable modem authorization, controlled by the Authorization state machine, is the process of obtaining:

- the list of SIDs identifying the particular encrypted traffic services a cable modem is authorized to access
- an authorization key, from which a key encryption key (KEK) and message authentication keys are derived

The KEK is a DES encryption key that the CMTS uses to encrypt the traffic encryption keys (TEKs) it sends to the modem. Traffic encryption keys are used for encrypting user data traffic. CM and CMTS use message authentication keys to authenticate, via a keyed message digest, the key requests and responses they exchange.

After achieving authorization, a cable modem periodically seeks re-authorization with the CMTS; reauthorization is managed by the CM's Authorization state machine. A CM must maintain its authorization status with the CMTS in order to be able to refresh aging traffic encryption keys. TEK state machines manage the refreshing of traffic encryption keys.

A cable modem begins authorization by sending an Authorization Request message to its CMTS. This is a request for an authorization key, and for the identities of the security associations (i.e., the SIDs) the CM is authorized to participate in. The Authorization Request includes:

- a convenient cable modem identifier unique to that cable modem (e.g., a cable modem MAC)
- the cable modem's public key
- a list of zero or more unicast SIDs corresponding to provisioned class-of-service settings configured for Baseline Privacy. The CMTS will have assigned these SIDs to the CM during RF MAC registration.

Note that it is not practical to use a cable modem's public key as a cable modem identifier. While the public key is unique to that cable modem, it is long, i.e., 768 bits, and cannot conveniently be used by CMTS software as a table index or CM “handle.”

In response to an Authorization Request message, a CMTS generates a new authorization key, encrypts it with the cable modem's public key, and sends it back to the CM in an Authorization Reply message. The authorization reply includes:

- the new authorization key encrypted with the CM's public key
- a 4-bit key sequence number, used to distinguish between successive generations of authorization keys (with each reauthorization, a new authorization key is generated)
- a list of SIDs that the CM is authorized to maintain keying information for (including the unicast SID(s) presented in the authorization request)

The CMTS, in responding to a CM's Authorization Request, will determine whether the requesting cable modem, identified by the cable modem id (e.g., MAC address), is authorized for basic unicast services, and what additional multicast services the cable modem's user has subscribed for.

Upon achieving authorization, a CM starts a separate TEK state machine for each of its authorized SIDs. Each TEK state machine operating within the CM is responsible for managing the keying material associated with its respective SID. TEK state machines periodically send Key Request messages to the CMTS, requesting a refresh of keying material for their respective SIDs. A Key Request includes:

- a cable modem identifier unique to that cable modem
- the SID whose keying material is being requested
- an HMAC keyed message digest, authenticating the Key Request

The CMTS responds to a Key Request with a Key Reply message, containing current keying material for a specific SID. This keying material includes:

- the DES-encrypted traffic encryption key
- a CBC initialization vector
- a key sequence number
- a key lifetime
- an HMAC keyed message, authenticating the Key Reply

The traffic encryption key (TEK), in the Key Reply, is DES (ECB mode) encrypted with a key encryption key (KEK), derived from the authorization key.

A TEK state machine remains active as long as:

- the CM is authorized to operate in the CMTS's security domain; i.e., it has a valid authorization key, and
- the CM is authorized to participate in that particular Security Association; i.e. CMTS continues to provide fresh keying material during re-key cycles.

The parent Authorization state machine stops *all* of its child TEK state machines when the CM receives from the CMTS an Authorization Reject during a reauthorization cycle. Individual TEK state machines can be started or stopped during a reauthorization cycle if a CM's SID authorizations changed between successive re-authorizations.

Communication between Authorization and TEK state machines occurs through the passing of events and protocol messaging. The Authorization state machine generates events (i.e., Stop, Authorized, Authorization Pending, and Authorization Complete events) that are targeted at its child TEK state machines. TEK state machines do not target events at their parent Authorization state machine. The TEK state machine affects the Authorization state machine indirectly through the messaging a CMTS sends in response to a modem's requests: a CMTS MAY respond to a TEK machine's Key Requests with a failure response (i.e., Authorization Invalid message) that will be handled by the Authorization state machine.

4.1.2 Authorization State Machine

The Authorization state machine consists of five states and seven distinct events (including receipt of messages) that can trigger state transitions. The Authorization finite state machine (FSM) is presented below in a graphical format, as a state flow model (Figure 4-1), and in a tabular format, as a state transition matrix (Table 4-1).

The state flow diagram depicts the protocol messages transmitted and internal events generated for each of the model's state transitions; however, the diagram does not indicate additional internal actions, such as the clearing or starting of timers, that accompany the specific state transitions. Accompanying the state transition matrix is a detailed description of the specific actions accompanying each state transition; the state transition matrix **MUST** be used as the definitive specification of protocol actions associated with each state transition.

The following legend applies to the KEK state flow diagram in Figure 4-1.

- Ovals are states.
- Events are in *italics*.
- Messages are in normal font.
- State transitions (i.e. the lines between states) are labeled with <what causes the transition>/<messages and events triggered by the transition>. So "*timeout*/Auth Request" means that the state received a "timeout" event and sent an Authorization Request ("Auth Request") message. If there are multiple events or messages before the slash "/" separated by a comma, *any* of them can cause a transition. If there are multiple events or messages listed after the slash, *all* of the specified actions must accompany the transition.

The Authorization state transition matrix presented in Table 4-1 lists the five Authorization machine states in the top-most row and the seven Authorization machine events (includes message receipts) in the left-most column. Any cell within the matrix represents a specific combination of state and event, with the next state (the state transitioned to) displayed within

the cell. For example, cell 3-B represents the receipt of an Authorization Reply (Auth Reply) message when in the Authorize Wait (Auth Wait) state. Within cell 3-B is the name of the next state, "Authorized." Thus, when a CM's Authorization state machine is in the Authorize Wait state and an Authorization Reply message is received, the Authorization state machine will transition to the Authorized state. In conjunction with this state transition, several protocol actions must be taken; these are described in the listing of protocol actions, under the heading 3-B, in Section 4.1.2.5.

A shaded cell within the state transition matrix implies that either the specific event cannot or should not occur within that state, and if the event does occur, the state machine **MUST** ignore it. For example, if an Authorization Reply message arrives when in the Authorized state, that message should be ignored (cell 3-C). The CM **MAY**, however, in response to an improper event, log its occurrence, generate an SNMP event, or take some other vendor-defined action. These actions, however, are not specified within the context of the Authorization state machine, which simply ignores improper events.

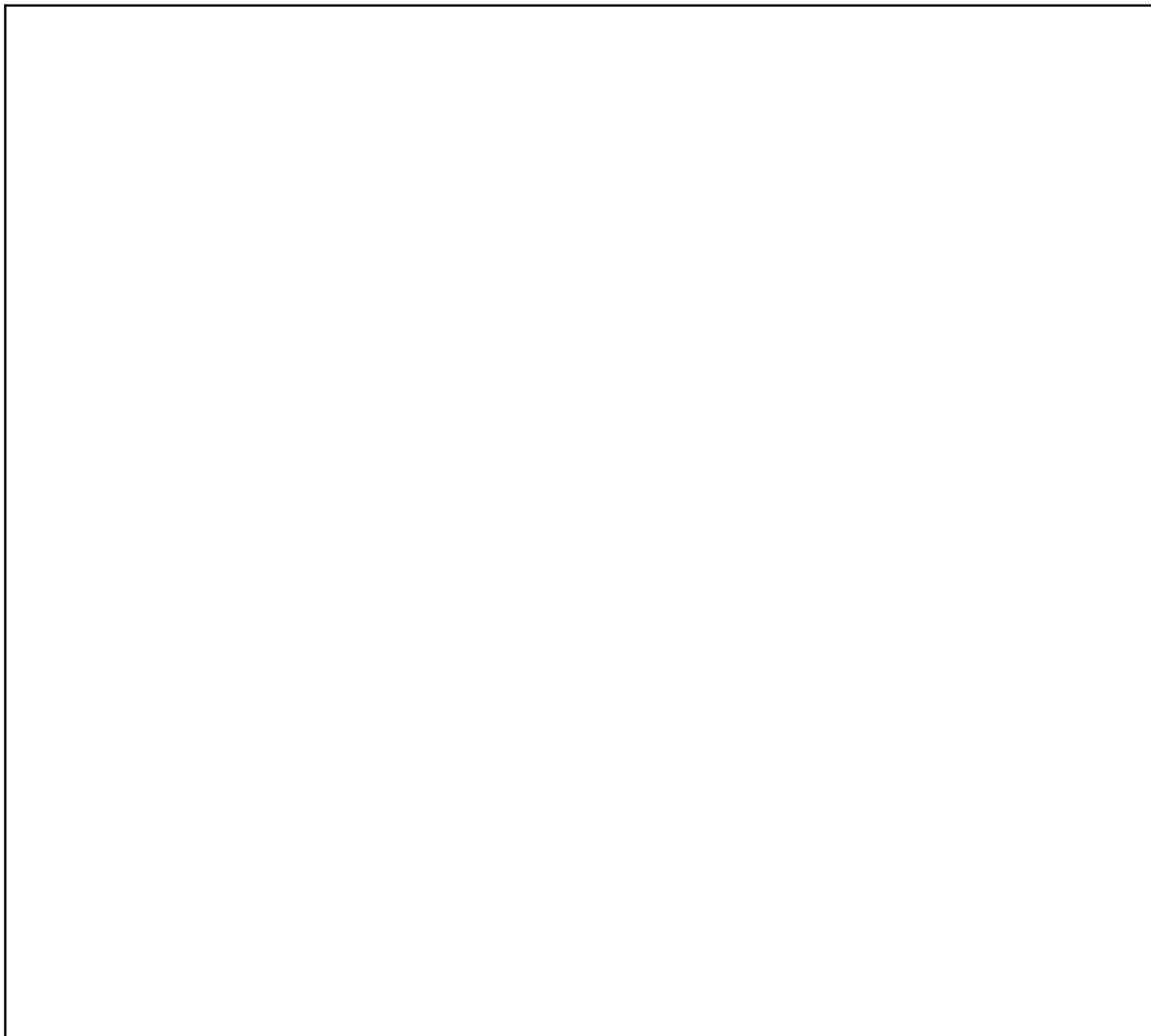


Figure 4-1. Authorization State Machine Flow Diagram

Table 4-1. Authorization FSM State Transition Matrix

| <i>State</i> | (A) Start | (B) Auth Wait | (C) Authorized | (D) Reauth Wait | (E) Auth Reject Wait |
|------------------------------------------|--------------|------------------------|-------------------|------------------------|-------------------------------|
| (1) <i>Provisioned</i> | Auth Wait | | | | |
| (2) <i>Auth Reject</i> | | Auth Reject Wait | | Auth Reject Wait | |
| (3) <i>Auth Reply</i> | | Authorized | | Authorized | |
| (4) <i>Timeout</i> | | Auth Wait | | Reauth Wait | Start |
| (5) <i>Auth Grace Timeout</i> | | | Reauth Wait | | |
| (6) <i>Auth Invalid</i> | | | Reauth Wait | Reauth Wait | |
| (7) <i>Reauth</i> | | | Reauth Wait | | |

4.1.2.1 States

4.1.2.1.1 Start

This is the initial state of the FSM. No resources are assigned to or used by the FSM in this state—e.g., all timers are off, and no processing is scheduled.

4.1.2.1.2 Authorize Wait (Auth Wait)

The CM has received the “Provisioned” event indicating that it has completed RF MAC registration with the CMTS. In response to receiving the event, the CM has sent the Authorize Request message to the CMTS and is waiting for the reply.

4.1.2.1.3 Authorized

The CM has received a Authorization Reply message which contains a list of valid SIDs for this CM. At this point, the modem has a valid authorization key and SID list. Transition into this state triggers the creation of one TEK FSM for each of the CM’s privacy-enabled SIDs.

4.1.2.1.4 Reauthorize Wait (Reauth Wait)

The CM has an outstanding re-authorization request. The CM was either about to time out its current authorization or received an indication (an Authorization Invalid message from the CMTS) that its authorization was no longer valid. The CM sent an Authorization Request message to the CMTS and is waiting for a response.

4.1.2.1.5 Authorize Reject Wait (Auth Reject Wait)

The CM received an Authorization Reject message in response to its last Authorization Request. In response to receiving this reject message, the CM set a timer and transitioned to the Authorize Reject Wait state. The CM remains in this state until the timer expires.

4.1.2.2 Messages

Note that the message formats are defined in detail in Section 4.2.

4.1.2.2.1 Authorization Request (Auth Request)

Request an authorization key and list of authorized SIDs. Sent from CM to CMTS.

4.1.2.2.2 Authorization Reply (Auth Reply)

Receive an authorization key and list of authorized SIDs. Sent from CMTS to CM. The authorization key is encrypted with the CM's public key.

4.1.2.2.3 Authorization Reject (Auth Reject)

Attempt to authorize was rejected. Sent from the CMTS to the CM.

4.1.2.2.4 Authorization Invalid (Auth Invalid)

The CMTS can send an Authorization Invalid message to a client CM as:

- an unsolicited indication, or
- a response to a message received from that CM

In either case, the Authorization Invalid message instructs the receiving CM to re-authorize with its CMTS.

The CMTS responds to a Key Request with an Authorization Invalid message if (1) the CMTS does not recognize the CM as being authorized (i.e., no valid authorization key associated with cable modem) or (2) verification of the Key Request's keyed message digest (in HMAC-Digest Attribute) failed. Note that the Authorization Invalid *event*, referenced in

both the state flow diagram and the state transition matrix, signifies either the receipt of a Authorization Invalid message or an internally generated event.

4.1.2.3 Events

4.1.2.3.1 Provisioned

The Authorization state machine generates this event upon entering the Start state if the RF MAC has completed initialization, i.e., CMTS registration. If the RF MAC initialization is not complete, the CM sends a Provisioned event to the Authorization FSM upon completing CMTS registration. The Provisioned event triggers the CM to begin the process of getting its authorization key and TEKs.

4.1.2.3.2 Timeout

A retransmission or wait timer timed out. Generally a request is resent.

4.1.2.3.3 Authorization Grace Timeout

The Authorization grace timer timed out. This timer fires a configurable amount of time (the Authorization “Grace” Time) before the current authorization is supposed to expire, signalling the CM to re-authorize before its authorization actually expires. The Authorization Grace Time is specified in a configuration setting within the TFTP-downloaded parameter file.

4.1.2.3.4 Reauthorize (Reauth)

CM's set of authorized SIDs may have changed. Event generated in response to an SNMP set, [MCNS8], meant to trigger a reauthorization cycle.

4.1.2.3.5 Authorization Invalid (Auth Invalid)

This event can be internally generated by the CM when there is a failure authenticating a Key Reply or Key Reject message, or externally generated by the receipt of an Authorization Invalid message, sent from the CMTS to the CM. A CMTS responds to a Key Request with an Authorization Invalid if verification of the request's message authentication code fails. Both cases indicate CMTS and CM have lost KEK synchronization.

A CMTS may also send a CM an unsolicited Authorization Invalid message to a CM, forcing an Authorization Invalid event.

[Note: the following events are sent by a parent Authorization state machine to its child TEK state machines.]

4.1.2.3.6 Stop

Sent by the Authorization FSM to an active (non -START state) TEK FSM to terminate their traffic keys.

4.1.2.3.7 Authorized

Sent by the Authorization FSM to a non-active (START state), but valid TEK FSM.

4.1.2.3.8 Authorization Pending (Auth Pend)

Sent by the Authorization FSM to a specific TEK FSM to place that TEK FSM in a wait state until the Authorization FSM can complete its re-authorization operation.

4.1.2.3.9 Authorization Complete (Auth Comp)

Sent by the Authorization FSM to a TEK FSM in the Operational Reauthorize Wait (Op Reauth Wait) or Rekey Reauthorize Wait (Rekey Reauth Wait) states to clear the wait state begun by an Authorization Pending event.

4.1.2.4 Parameters

All configuration parameter values are specified in the TFTP-downloaded parameter file (see Appendix A: TFTP Configuration File Extensions).

4.1.2.4.1 Authorize Wait Timeout

Timeout period between sending Authorization Request messages from Authorize Wait state.

4.1.2.4.2 Reauthorize Wait Timeout (Reauth Wait Timeout)

Timeout period between sending Authorization Request messages from Reauthorize Wait state.

4.1.2.4.3 Authorization Grace Time

Amount of time before authorization is scheduled to expire that the CM starts re-authorization.

4.1.2.4.4 Authorize Reject Wait Timeout

Length of time remain in the Authorize Reject Wait state.

4.1.2.5 Actions

Actions taken in association with state transitions are listed by <event/rcvd message> - <state> below:

1-A Start (*Provisioned*) → Auth Wait

- send Authorization Request message to CMTS
- set Authorization Request retry timer to Authorize Wait Timeout

2-B Auth Wait (Auth Reject) → Auth Reject Wait

- clear Authorization Request retry timer
- set a wait timer to Authorize Reject Wait Timeout

2-D Reauth Wait (Auth Reject) → Auth Reject Wait

- clear Authorization Request retry timer
- generate Stop events for all active TEK state machines
- set a wait timer to Authorize Reject Wait Timeout

3-B Auth Wait (Auth Reply) → Authorized

- clear Authorization Request retry timer
- decrypt and record authorization key delivered with Authorization Reply
- start TEK FSMs for all SIDs listed in Authorization Reply
- generate Authorized events for all non-active TEK FSMs (they should all be in Start state having just been started)
- set the Authorization grace timer to go off “Authorization Grace Time” seconds prior to the new authorization key’s scheduled expiration

3-D Reauth Wait (Auth Reply) → Authorized

- clear Authorization Request retry timer
- decrypt and record authorization key delivered with Authorization Reply
- start TEK FSMs for any newly authorized SIDs
- generate Authorized events for newly started FSMs
- generate Authorization Complete events for any currently active TEK FSMs whose corresponding SIDs were listed in Authorization Reply
- generate Stop events for any currently active TEK FSMs whose corresponding SIDs were not listed in Authorization Reply
- set the Authorization grace timer to go off “Authorization Grace Time” seconds prior to the new authorization key’s scheduled expiration

4-B Auth Wait (*Timeout*) → Auth Wait

- send Authorization Request message to CMTS
- set Authorization Request retry timer to Authorize Wait Timeout

4-D Reauth Wait (*Timeout*) → Reauth Wait

- send Authorization Request message to CMTS
- set Authorization Request retry timer to Reauthorize Wait Timeout

4-E Auth Reject Wait (*Timeout*) → Start

- no protocol actions associated with state transition

5-C Authorized (*Authorization Grace Timeout*) → Reauth Wait

- send Authorization Request message to CMTS
- set Authorization Request retry timer to Reauthorize Wait Timeout

6-C Authorized (*Authorization Invalid*) → Reauth Wait

- clear Authorization grace timer
- send Authorization Request message to CMTS
- set Authorization Request retry timer to Reauthorize Wait Timeout
- if the Authorization Invalid event is *not* associated with an unsolicited Authorization Invalid message, generate an Authorization Pending event for the TEK state machine responsible for the Authorization Invalid event (i.e., the TEK FSM that either generated the event, or sent the Key Request message the CMTS responded to with an Authorization Invalid message)

6-D Reauth Wait (*Authorization Invalid*) → Reauth Wait

- if the Authorization Invalid event is *not* associated with an unsolicited Authorization Invalid message, generate an Authorization Pending event for the TEK state machine responsible for the Authorization Invalid event (i.e., the TEK FSM that either generated the event, or sent the Key Request which the CMTS responded to with a Authorization Invalid message)

7-C Authorized (*Reauth*) → Reauth Wait

- clear Authorization grace timer
- send Authorization Request message to CMTS
- set Authorization Request retry timer to Reauthorize Wait Timeout

4.1.3 TEK State Machine

The TEK state machine consists of six states and nine events (including receipt of messages) that can trigger state transitions. Like the Authorization state machine, the TEK state machine is presented in both a state flow diagram and a state transition matrix. And as was the case for the Authorization state machine, the state transition matrix **MUST** be used as the definitive specification of protocol actions associated with each state transition.

Shaded states in Figure 4-2 (Operational, Rekey Wait, and Rekey Reauthorize Wait) have valid keying material and encrypted traffic can be passed.

The Authorization state machine starts an independent TEK state machine for each of its authorized SIDs.

Table 4-2. TEK FSM State Transition Matrix

| State Event or Rcvd Message | (A) Start | (B) Op Wait | (C) Op Reauth Wait | (D) Op | (E) Rekey Wait | (F) Rekey Re-auth Wait |
|--------------------------------------|--------------|-------------------|--------------------------|---------------|--------------------------|---------------------------------|
| (1) Stop | | Start | Start | Start | Start | Start |
| (2) Authorized | Op Wait | | | | | |
| (3) Auth Pend | | Op Reauth Wait | | | Rekey Re-auth Wait | |
| (4) Auth Comp | | | Op Wait | | | Rekey Wait |
| (5) TEK Invalid | | | | Op Wait | Op Wait | Op Reauth Wait |
| (6) Timeout | | Op Wait | | | Rekey Wait | |
| (7) TEK Grace Timeout | | | | Rekey Wait | | |
| (8) Key Reply | | Opera- tional | | | Opera- tional | |
| (9) Key Reject | | Start | | | Start | |

4.1.3.1 States

4.1.3.1.1 Start

This is the initial state of the FSM. No resources are assigned to or used by the FSM in this state—e.g., all timers are off, and no processing is scheduled.

4.1.3.1.2 Operational Wait (Op Wait)

The TEK state machine has sent its initial request (Key Request) for its SID's keying material (traffic encryption key and CBC initialization vector), and is waiting for a reply from the CMTS.

4.1.3.1.3 Operational Reauthorize Wait (Op Reauth Wait)

The wait state the TEK state machine is placed in if it does not have valid keying material while the Authorization state machine is in the middle of a reauthorization cycle.

4.1.3.1.4 Operational

The CM has a valid TEK for the associated SID.

4.1.3.1.5 Rekey Wait

The TEK grace timer has expired and the CM has requested a replacement key for this SID. Note that the current key has not expired and can still be used for encrypting and decrypting data traffic.

4.1.3.1.6 Rekey Reauthorize Wait (Rekey Reauth Wait)

The wait state the TEK state machine is placed in if the TEK state machine has valid traffic keying material, has an outstanding request for the latest keying material, and the Authorization state machine initiates a reauthorization cycle.

4.1.3.2 Messages

Note that the message formats are defined in detail in Section 4.2.

4.1.3.2.1 Key Request

Request a TEK for this SID. Sent by the CM to the CMTS and authenticated with keyed message digest. The message authentication key is derived from the authorization key.

4.1.3.2.2 Key Reply

Response from the CMTS with a TEK for this SID. Sent by the CMTS to the CM, it includes the traffic encryption key, DES encrypted with a key encryption key derived from the authorization key. The Key Reply message is authenticated with a keyed message digest; the authentication key is derived from the authorization key.

4.1.3.2.3 Key Reject

Response from the CMTS to the CM to indicate this SID is no longer valid and no key will be sent. The Key Reject message is authenticated with a keyed message digest; the authentication key is derived from the authorization key

4.1.3.2.4 TEK Invalid

The CMTS sends a CM this message if it determines that the CM encrypted an upstream Packet Data PDU with an invalid TEK; i.e., a SID's TEK key sequence number, contained within the received packet's Baseline Privacy Extended Header element, is out of the CMTS's range of known, valid sequence numbers for that SID.

4.1.3.3 Events

4.1.3.3.1 Stop

See Section 4.1.2.3.6.

4.1.3.3.2 Authorized

See Section 4.1.2.3.7.

4.1.3.3.3 Authorization Pending (Auth Pend)

See Section 4.1.2.3.8.

4.1.3.3.4 Authorization Complete (Auth Comp)

See Section 4.1.2.3.9.

4.1.3.3.5 TEK Invalid

This event can be triggered by either a CM's data packet decryption logic, or by the receipt of a TEK Invalid message from the CMTS.

A CM's data packet decryption logic triggers a TEK Invalid message if it recognizes a loss of TEK key synchronization between itself and the encrypting CMTS; i.e., a SID's TEK key sequence number, contained within the received, downstream packet's Baseline Privacy Extended Header element, is out of the CM's range of known sequence numbers for that SID.

A CMTS sends a CM a TEK Invalid message, triggering a TEK Invalid event within the CM, if the CMTS's decryption logic recognizes a loss of TEK key synchronization between itself and the CM.

4.1.3.3.6 Timeout

A retry timer timeout. Generally, the particular request is retransmitted.

4.1.3.3.7 TEK Grace Timeout

The TEK grace timer timed out. This timer fires a configurable amount of time (the TEK “Grace” Time) before the TEK is supposed to expire, signalling the CM to re-key its TEK before the TEK actually expires. The TEK Grace Time is specified in a configuration setting within the TFTP-downloaded parameter file, and is the same across all SIDs.

4.1.3.4 Parameters

All configuration parameter values are specified in TFTP downloaded parameter file (see Appendix A: TFTP Configuration File Extensions).

4.1.3.4.1 Operational Wait Timeout

Timeout period between sending of Key Request messages from the Op Wait state.

4.1.3.4.2 Rekey Wait Timeout

Timeout period between sending of Key Request messages from the Rekey Wait state.

4.1.3.4.3 TEK Grace Time

Amount of time before TEK expires that the CM begins re-keying.

4.1.3.5 Actions

1-B Op Wait (*Stop*) → Start

- clear Key Request retry timer
- terminate TEK FSM

1-C Op Reauth Wait (*Stop*) → Start

- terminate TEK FSM

1-D Operational (*Stop*) → Start

- clear TEK grace timer, which is timer set to go off “TEK Grace Time” seconds prior to the TEK’s scheduled expiration time
- terminate TEK FSM

- remove SID keying material from key table

1-E Rekey Wait(*Stop*) → Start

- clear Key Request retry timer
- terminate TEK FSM
- remove SID keying material from key table

1-F Rekey Reauth Wait(*Stop*) → Start

- terminate TEK FSM
- remove SID keying material from key table

2-A Start (*Authorized*) → Op Wait

- send Key Request Message to CMTS
- set Key Request retry timer to Operational Wait Timeout

3-B Op Wait (*Auth Pend*) → Op Reauth Wait

- clear Key Request retry timer

3-E Rekey Wait (*Auth Pend*) → Rekey Reauth Wait

- clear Key Request retry timer

4-C Op Reauth Wait (*Auth Comp*) → Op Wait

- send Key Request message to CMTS
- set Key Request retry timer to Operational Wait Timeout

4-F Rekey Reauth Wait (*Auth Comp*) → Rekey Wait

- send Key Request message to CMTS
- set Key Request retry timer to Rekey Wait Timeout

5-D Operational (*TEK Invalid*) → Op Wait

- clear TEK grace timer
- send Key Request message to CMTS
- set Key Request retry timer to Operational Wait Timeout
- remove SID keying material from key table

5-E Rekey Wait (*TEK Invalid*) → Op Wait

- clear Key Request retry timer
- send Key Request message to CMTS

- set Key Request retry timer to Operational Wait Timeout
- remove SID keying material from key table

5-F Rekey Reauth Wait (*TEK Invalid*) → Op Reauth Wait

- remove SID keying material from key table

6-B Op Wait (*Timeout*) → Op Wait

- send Key Request message to CMTS
- set Key Request retry timer to Operational Wait Timeout

6-E Rekey Wait (*Timeout*) → Rekey Wait

- send Key Request message to CMTS
- set Key Request retry timer to Rekey Wait Timeout

7-D Operational (*TEK Grace Timeout*) → Rekey Wait

- send Key Request message to CMTS
- set Key Request retry timer to Rekey Wait Timeout

8-B Op Wait (Key Reply) → Operational

(Note: Key Reply passed message authentication.)

- clear Key Request retry timer
- process contents of Key Reply message and incorporate new keying material into key database
- set the TEK grace timer to go off “TEK Grace Time” seconds prior to the key’s scheduled expiration

8-E Rekey Wait (Key Reply) → Operational

(Note: Key Reply passed message authentication.)

- clear Key Request retry timer
- process contents of Key Reply message and incorporate new keying material into key database
- set the TEK grace timer to go off “TEK Grace Time” seconds prior to the key’s scheduled expiration

9-B Op Wait (Key Reject) → Start

(Note: Key Reject passed message authentication.)

- clear Key Request retry timer

- terminate TEK FSM

9-E Rekey Wait (Key Reject) → Start

- clear Key Request retry timer
- terminate TEK FSM
- remove SID keying material from key table

4.2 Key Management Message Formats ¹

Baseline Privacy Key Management employs two MAC message types: BPKM-REQ and BPKM-RSP. [MCNS1] defines the specific type values assigned to them.

Table 4-3. Baseline Privacy Key Management MAC Messages

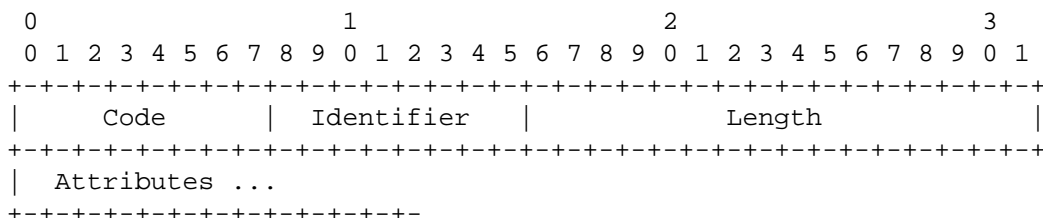
| Type Value | Message Name | Message Description |
|-------------|--------------|----------------------------------------------|
| See [MCNS1] | BPKM-REQ | Privacy Key Management Request [CM -> CMTS] |
| See [MCNS1] | BPKM-RSP | Privacy Key Management Response [CMTS -> CM] |

While these two MAC management message types distinguish between BPKM requests (CM to CMTS) and responses (CMTS to CM), more detailed information about message contents is encoded in the BPKM messages themselves. This maintains a clean separation between privacy management functions and RF MAC upstream bandwidth allocation, timing and synchronization (RF MAC management’s principal responsibilities).

4.2.1 Packet Formats

Exactly one BPKM message is encapsulated in the Management Message Payload field of a MAC management message.

A summary of the BPKM message format is shown below. The fields are transmitted from left to right.



¹. Message formats for the Baseline Privacy Key Management protocol are modeled after those of the Remote Authentication Dial In User Service (RADIUS) protocol, defined in RFC 2058, and an Internet standards track protocol. BPKM, like RADIUS, adheres to a client/server model. Unlike RADIUS, BPKM will not run over UDP/IP. BPKM messages are encapsulated within RF MAC management messages.

Code

The Code field is one octet, and identifies the type of BPKM packet. When a packet is received with an invalid Code field, it SHOULD be silently discarded.

BPKM Codes (decimal) are assigned as follows:

Table 4-4. Baseline Privacy Key Management Message Codes

| Code | BPKM Message Type | MAC Management Message Name |
|--------|-------------------|-----------------------------|
| 0-3 | Reserved | - |
| 4 | Auth Request | BPKM-REQ |
| 5 | Auth Reply | BPKM-RSP |
| 6 | Auth Reject | BPKM-RSP |
| 7 | Key Request | BPKM-REQ |
| 8 | Key Reply | BPKM-RSP |
| 9 | Key Reject | BPKM-RSP |
| 10 | Auth Invalid | BPKM-RSP |
| 11 | TEK Invalid | BPKM-RSP |
| 12-255 | Reserved | - |

Identifier

The Identifier field is one octet. A CM uses the identifier to match a CMTS's responses to the CM's requests.

The CM MUST change (e.g., increment, wrapping around to 0 after reaching 255) the Identifier field whenever it issues a BPKM Authorization Request or Key Request message. The Identifier MUST be incremented for request retransmissions triggered by retry timeouts as well as for the initial Authorization Request or Key Request issued upon entering the respective Wait state.

The Identifier field in a CMTS's BPKM response message MUST be a copy of the Identifier field of the BPKM request message the CMTS is responding to. The Identifier field in TEK Invalid messages, which are not sent in response to BPKM requests, MUST be set to zero. The Identifier field in unsolicited Authorization Invalid messages MUST be set to zero.

On reception of a BPKM response message, the CM associates the message with a particular state machine (the Authorization state machine in the case of Authorization Replies, Authorization Rejects, and Authorization Invalids; a particular TEK state machine in the case of Key Replies, Key Rejects and TEK Invalids). A CM MUST keep track of the Identifier of its latest, pending Authorization request. The CM MUST silently discard Authorization Replies and Authorization Rejects whose Identifier fields do not match those of the pending requests.

A CM MAY keep track of the Identifier of its latest, pending Key Request. The CM MAY silently discard Key Replies and Key Rejects whose Identifier fields do not match those of the pending requests.

Length

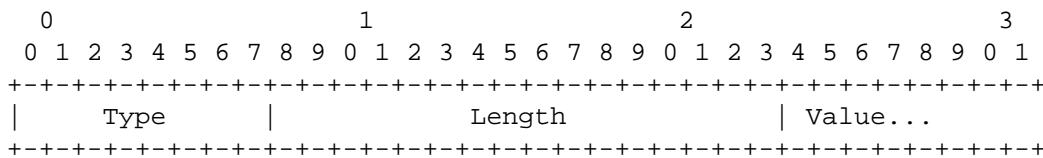
The Length field is two octets. It indicates the length of the Attribute fields in octets. The length field does not include the Code, Identifier and Length fields. Octets outside the range of the Length field MUST be treated as padding and ignored on reception. If the packet is shorter than the Length field indicates, it SHOULD be silently discarded. The minimum length is 0 and maximum length is 1490.

Attributes

BPKM Attributes carry the specific authentication, authorization and key management data exchanged between client and server. Each BPKM packet type has its own set of required and optional Attributes. Unless explicitly stated, there are no requirements on the ordering of attributes within a BPKM message.

The end of the list of Attributes is indicated by the Length of the BPKM packet.

Attributes are type/length/value (TLV) encoded, as shown below. The fields are transmitted from left to right.



Packet formats for each of the BPKM messages are described below. The descriptions list the BPKM attributes contained within each BPKM message type. The Attributes themselves are described in Section 4.2.2. Unspecified attributes shall be ignored on receipt, and skipped over while scanning for recognized attributes.

The CMTS MUST silently discard all requests that do not contain ALL required attributes. The CM MUST silently discard all responses that do not contain ALL required attributes.

4.2.1.1 Authorization Request (Auth Request)

Code: 4

Attributes:

Table 4-5. Authorization Request Attributes

| Attribute | Contents |
|--------------------|-----------------------------------------------------------|
| CM-Identification | contains information used to identify cable modem to CMTS |
| (zero or more) SID | CM unicast SID obtained during RF MAC CMTS registration |

The CM-Identification attribute contains a set of data that identifies the requesting cable modem to the CMTS. Note that the CMTS is in all likelihood using only a single item in the CM-Identification attribute (e.g., CM MAC address) as a CM handle. While a specific item could be selected for inclusion in the Authorization Request message, including the entire CM-Identification attribute for client identification provides vendors with greater flexibility in the headend's system design.

A SID attribute contains an RF MAC Service ID, which in Baseline Privacy also serves as a security association identifier. A cable modem includes in its Authorization Request those unicast SIDs, obtained during RF MAC registration with the CMTS, that are provisioned for Baseline Privacy.

4.2.1.2 Authorization Reply (Auth Reply)

Sent by CMTS to a client CM in response to an Authorization Request, the Authorization Reply message contains an authorization key, the key's lifetime, the key's sequence number, and a list of SIDs identifying the specific data services the requesting cable modem is authorized to access. The authorization key **MUST** be encrypted with the CM's public key. The SID list **MUST** include the unicast SID(s) reported to the CMTS in the corresponding Authorization Request.

Code field: 5

Attributes:

Table 4-6. Authorization Reply Attributes

| Attribute | Contents |
|---------------------|----------------------------------------------------------------------------|
| AUTH-Key | Authorization (AUTH) Key, encrypted with the target client CM's public key |
| Key-Lifetime | Authorization key lifetime |
| Key-Sequence-Number | Authorization key sequence number |
| (one or more) SID | Service ID |

4.2.1.3 Authorization Reject (Auth Reject)

CMTS responds to a CM's authorization request with an Authorization Reject message if the CMTS rejects the CM's authorization request.

Code field: 6

Attributes:

Table 4-7. Auth Rej Attributes

| Attribute | Contents |
|---------------------------|------------------------------------------------------------------------|
| Error-Code | Error code identifying reason for rejection of authorization request |
| Display-String (optional) | Display String providing reason for rejection of authorization request |

4.2.1.4 Key Request

Code: 7

Attributes:

Table 4-8. Key Request Attributes

| Attribute | Contents |
|---------------------|-----------------------------------------------------------|
| CM-Identification | Contains information used to identify cable modem to CMTS |
| Key-Sequence-Number | Authorization key sequence number |
| SID | Service ID (serves as Security Association ID) |
| HMAC-Digest | Keyed SHA message digest |

The HMAC-Digest Attribute is a keyed message digest. The HMAC-Digest Attribute **MUST** be the final Attribute in the Key Request's Attribute list. The message digest is performed over the packet header and all of the Key Request's Attributes, other than the HMAC-Digest, in the order in which they appear within the packet.

Inclusion of the keyed digest allows the CMTS to authenticate the Key Request message. The HMAC-Digest's authentication key is derived from the authorization key. See Section 6, Cryptographic Methods, for details.

4.2.1.5 Key Reply

Code: 8

Attributes:

Table 4-9. Key Reply Attributes

| Attribute | Contents |
|---------------------------|-----------------------------------------------------|
| Key-Sequence-Number | Authorization key sequence number |
| SID | Service ID (serves as Security Association ID) |
| SA-Flag | Security Association Flags |
| TEK-Parameters | Latest generation of key parameters relevant to SID |
| TEK-Parameters (optional) | Prior generation of key parameters relevant to SID |
| HMAC-Digest | Keyed SHA message digest |

The TEK-Parameters Attribute is a compound attribute containing all of the keying material corresponding to a particular generation of SID's TEK. This would include the TEK, encrypted (DES ECB) with a key encryption key derived from the authorization key, the TEK's remaining key lifetime and key sequence number and the DES CBC initialization vector. See Section 4.2.2.13 for details.

Under certain conditions, the CMTS will need to distribute to a client CMTS two generations of keying material (the latest generation and the generation immediately preceding that). In these cases, a Key Reply message will contain two TEK-Parameters Attributes. Section 5, Key Usage, describes the conditions under which the CMTS distributes two successive generations of keying material to a client CM.

The HMAC-Digest Attribute is a keyed message digest. The HMAC-Digest Attribute **MUST** be the final Attribute in the Key Reply's Attribute list. The message digest is performed over the packet header and all of the Key Request's Attributes, other than the HMAC-Digest, in the order in which they appear within the packet.

Inclusion of the keyed digest allows the receiving client to authenticate the Key Reply message and ensure CM and CMTS have synchronized authorization keys. The HMAC-Digest's authentication key is derived from the authorization key. See Section 6, Cryptographic Methods, for details.

4.2.1.6 Key Reject

Receipt of a Key Reject indicates the receiving client CM is no longer authorized for a particular SID.

Code: 9

Attributes:

Table 4-10. Key Reject Attributes

| Attribute | Contents |
|---------------------------|------------------------------------------------------------|
| Key-Sequence-Number | Authorization key sequence number |
| SID | Service ID |
| Error-Code | Error code identifying reason for rejection of Key Request |
| Display-String (optional) | Display string containing reason for Key Reject |
| HMAC-Digest | Keyed SHA message digest |

The HMAC-Digest Attribute is a keyed message digest. The HMAC-Digest Attribute **MUST** be the final Attribute in the Key Reject's Attribute list. The message digest is performed over the packet header and all of the Key Reject's Attributes, other than the HMAC-Digest, in the order in which they appear within the packet.

Inclusion of the keyed digest allows the receiving client to authenticate the Key Reject message and ensure CM and CMTS have synchronized authorization keys. The HMAC-Digest's authentication key is derived from the authorization key. See Section 6, Cryptographic Methods, for details.

4.2.1.7 Authorization Invalid

The CMTS can send an Authorization Invalid message to a client CM as:

- an unsolicited indication, or
- a response to a message received from that CM

In either case, the Authorization Invalid message instructs the receiving CM to re-authorize with its CMTS.

The CMTS sends an Authorization Invalid in response to a Key Request if (1) the CMTS does not recognize the CM as being authorized (i.e., no valid authorization key associated with the requesting cable modem) or (2) verification of the Key Request's keyed message digest (in HMAC-Digest Attribute) failed, indicating a loss of authorization key synchronization between CM and CMTS.

Code: 10

Attributes:

Table 4-11. Authorization Invalid Attributes

| Attribute | Contents |
|---------------------------|---------------------------------------------------------|
| Error-Code | Error code identifying reason for Authorization Invalid |
| Display-String (optional) | Display String describing failure condition |

4.2.1.8 TEK Invalid

The CMTS sends a TEK Invalid message to a client CM if the CMTS determines that the CM encrypted an upstream Packet Data PDU with an invalid TEK; i.e., a SID's TEK key sequence number, contained within the received packet's Baseline Privacy Extended Header element, is out of the CMTS's range of known, valid sequence numbers for that SID.

Code: 11

Attributes:

Table 4-12. TEK Invalid Attributes

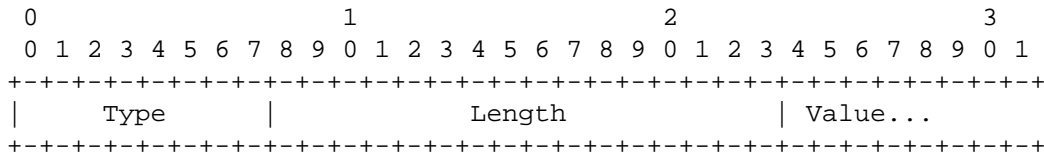
| Attribute | Contents |
|---------------------------|-------------------------------------------------------|
| Key-Sequence-Number | Authorization key sequence number |
| SID | Service ID |
| Error-Code | Error code identifying reason for TEK Invalid message |
| Display-String (optional) | Display string containing vendor-defined information |
| HMAC-Digest | Keyed SHA message digest |

The HMAC-Digest Attribute is a keyed message digest. The HMAC-Digest Attribute **MUST** be the final Attribute in the TEK Invalid's Attribute list. The message digest is performed over the packet header and all of the TEK Invalid's Attributes, other than the HMAC-Digest, in the order in which they appear within the packet.

Inclusion of the keyed digest allows the receiving client to authenticate the TEK Invalid message and ensure CM and CMTS have synchronized authorization keys. The HMAC-Digest's authentication key is derived from the authorization key. See Section 6, Cryptographic Methods, for details.

4.2.2 BPKM Attributes

A summary of the Attribute format is shown below. The fields are transmitted from left to right.



Type

The Type field is one octet. Values of the BPKM Type field are specified below. Note that Type values between 0 and 127 are defined within the Baseline Privacy Specification, values between 128 and 255 are vendor-assigned Attribute Types.

A BPKM server MAY ignore Attributes with an unknown Type.

A BPKM client MAY ignore Attributes with an unknown Type.

Table 4-13. BPKM Attribute Types

| Type | BPKM Attribute |
|---------|---------------------------------|
| 0 | Reserved |
| 1 | Serial-Number |
| 2 | Manufacturer-ID |
| 3 | MAC-Address |
| 4 | RSA-Public-Key |
| 5 | CM-Identification |
| 6 | Display-String |
| 7 | AUTH-KEY |
| 8 | TEK-KEY |
| 9 | Key-Lifetime |
| 10 | Key-Sequence-Number |
| 11 | HMAC-Digest |
| 12 | SID |
| 13 | TEK-Parameters |
| 14 | SA-Flag |
| 15 | DES-CBC-IV |
| 16 | Error-Code |
| 17-126 | Reserved |
| 127 | Vendor-Defined |
| 128-255 | Vendor-assigned attribute types |

Length

The Length field is 2 octets, and indicates the length of this Attribute's Value field, in octets. The length field *does not include* the Type and Length fields ¹. The minimum Attribute Length is 0, the maximum Length is 1487.

Packets containing attributes with invalid lengths SHOULD be silently discarded.

Value

The Value field is zero or more octets and contains information specific to the Attribute. The format and length of the Value field is determined by the Type and Length fields. All multi-octet integer quantities are in network-byte order, i.e., the octet containing the most-significant bits is the first transmitted on the wire.

Note that a “string” does not require termination by an ASCII NUL because the Attribute already has a length field.

The format of the value field is one of five data types.

Table 4-14. Attribute Value Data Types

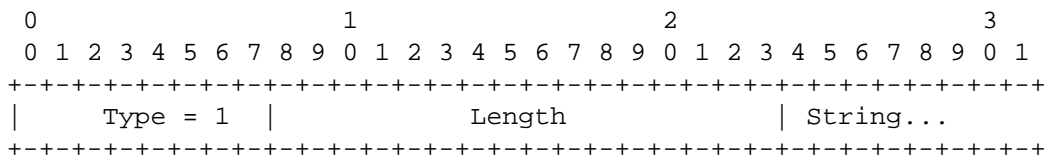
| | |
|----------|--------------------------|
| string | 0 – 1487 octets |
| uint8 | 8-bit unsigned integer |
| uint16 | 16-bit unsigned integer |
| uint32 | 32-bit unsigned integer |
| compound | collection of Attributes |

4.2.2.1 Serial-Number

Description

This Attribute indicates the serial number assigned by the manufacturer to a cable modem device.

A summary of the Serial-Number Attribute format is shown below. The fields are transmitted from left to right.



Type

1 for Serial-Number

Length

>= 0 and =<255

¹. Note that this is consistent with both the TLV encoding employed in the RF MAC’s Extended Header Elements, and the TLV encoding employed for configuration settings in the CM Configuration File [MCNS1]. BPKM’s TLV encoding differs from that employed by the RADIUS protocol, on which BPKM’s basic message structure is based: the Length field of RADIUS attributes includes the Type and Length fields, as well as an attribute’s Value field.

String

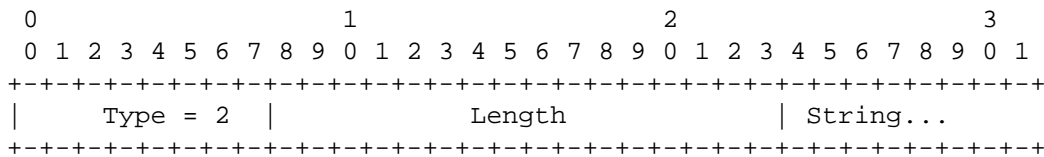
The String field is zero or more octets and contains a manufacturer-assigned serial number.

4.2.2.2 *Manufacturer-ID*

Description

This Attribute identifies the manufacturer. The identifier is 3 octets long and contains the 3-octet Organizationally Unique Identifier (OUI) assigned to applying organizations by the IEEE [IEEE1]. The first two bits of the 3-octet string are set to zero.

A summary of the Manufacturer-ID Attribute format is shown below. The fields are transmitted from left to right.



Type

2 for Manufacturer-ID

Length

3

String

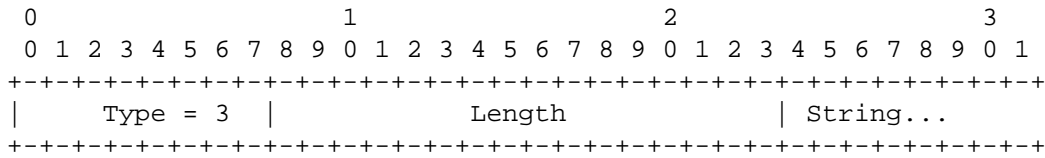
The String field is three octets and contains an IEEE OUI.

4.2.2.3 *MAC-Address*

Description

This Attribute identifies the IEEE MAC address assigned to the CM. Guaranteed to be unique, it is likely to be used as a cable modem handle/index at the CMTS.

A summary of the MAC-Address Attribute format is shown below. The fields are transmitted from left to right.



Type

3 for MAC-Address

Length

6

String

The String field contains a 6-octet MAC address.

4.2.2.4 RSA-Public-Key

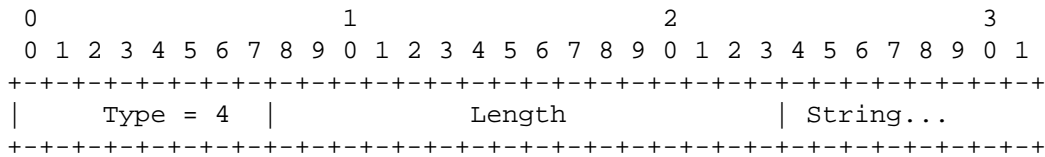
Description

This Attribute is a string attribute containing a BER-encoded RSAPublicKey ASN.1 type, as defined in the RSA Encryption Standard (PKCS #1) [RSA1].

PKCS #1 specifies that an RSA public key consists of both an RSA public modulus and an RSA public exponent; the RSAPublicKey type includes both of these as BER-encoded INTEGER types.

PKCS #1 states that the RSA public exponent may be standardized in specific applications, and the document suggests values of 3 or 65537 (F4). Baseline Privacy standardizes on F4 for a public exponent and employs a 768-bit modulus.

A summary of the Public-Key Attribute format is shown below. The fields are transmitted from left to right.



Type

4 for RSA-Public-Key

Length

106 (length of BER-encoding, using F4 as the public exponent, and a 768-bit public modulus)

String

BER-encoded RSAPublicKey ASN.1 type

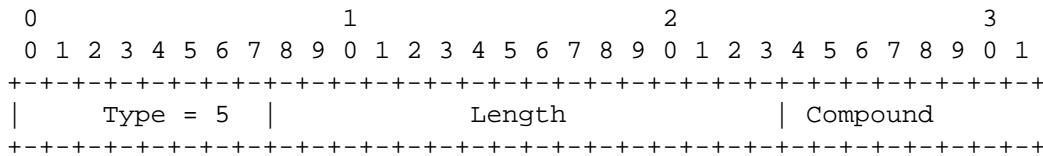
4.2.2.5 CM-Identification

Description

This Attribute is a compound attribute, consisting of a collection of sub-attributes. These sub-attributes contain information that can be used to uniquely identify a cable modem. Sub-attributes **MUST** include:

- Serial-Number
- Manufacturer-ID
- MAC-Address
- RSA-Public-Key

The CM-Identification **MAY** also contain optional Vendor-Defined Attributes. At some point in the future, it may contain a true digital certificate.



Type

5

Length

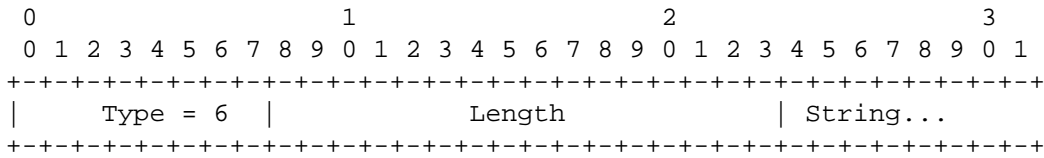
>= 126

4.2.2.6 Display-String

Description

This Attribute contains a textual message. It is typically used to explain a failure response, and might be logged by the receiver for later retrieval by an SNMP manager. Display strings **MUST** be no longer than 128 bytes.

A summary of the Display-String Attribute format is shown below. The fields are transmitted from left to right.



Type

6 for Display String

Length

>=0 and <= 128

String

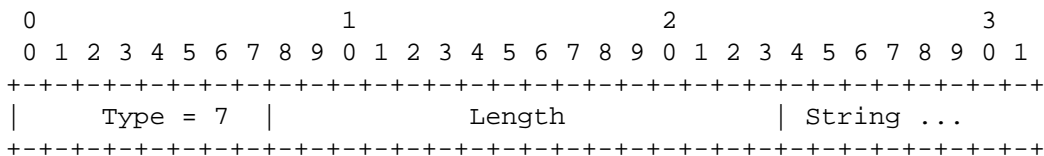
A string of characters. The character string MAY NOT be null-terminated; the length field will identify the end of the string.

4.2.2.7 AUTH-Key

Description

The authorization key is an 8 byte quantity, from which a key encryption key, and two message authentication keys (one for upstream requests, and a second for downstream replies) are derived.

This Attribute contains a 96-octet (768-bit) quantity which is the authorization key encrypted with the CM's RSA public key, as specified in PKCS #1 [RSA1]. The ciphertext produced by the RSA algorithm will be the length of the RSA modulus, i.e., 96 octets.



Type

7 for AUTH-Key

Length

96

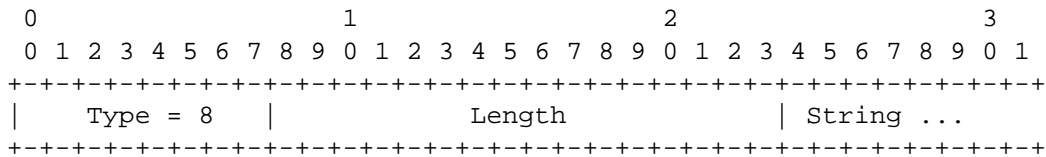
String

96-octet quantity representing an RSA-encrypted authorization key.

4.2.2.8 *TEK-KEY*

Description

This Attribute contains an 8-octet quantity that is a TEK DES key, DES encrypted with a Key Encryption Key derived from the authorization key. TEK keys are encrypted using the Electronic Codebook (ECB) mode of DES. See Section 6, Cryptographic Methods, for details.



Type

8 for TEK-KEY

Length

8

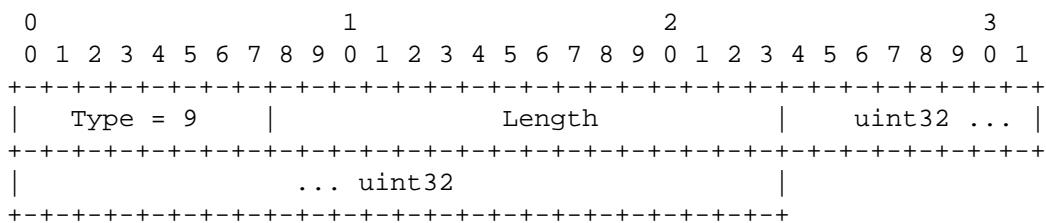
String

64-bit quantity representing a (DES ECB) encrypted traffic encryption key.

4.2.2.9 *Key-Lifetime*

Description

This Attribute contains the lifetime, in seconds, of an authorization key or TEK. It is a 32-bit unsigned quantity representing the number of remaining seconds that the associated key will be valid.



Type

9 for Key-Lifetime

Length

4

uint32

32-bit quantity representing key lifetime

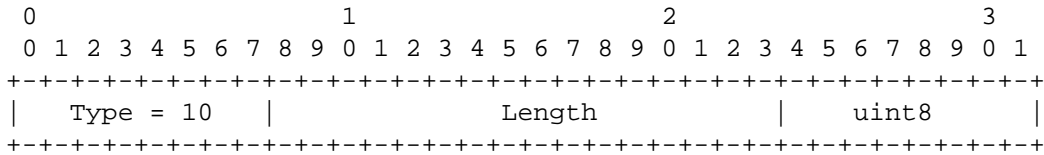
A key lifetime of zero indicates that the corresponding authorization key or traffic encryption key is not valid.

4.2.2.10 Key-Sequence-Number

Description

This Attribute contains a 4-bit sequence number for a TEK or authorization key. The 4-bit quantity, however, is stored in a single octet, with the high-order 4 bits set to 0.

A summary of the Key-Sequence-Number Attribute format is shown below. The fields are transmitted from left to right.



Type

10 for Key-Sequence-Number

Length

1

uint8

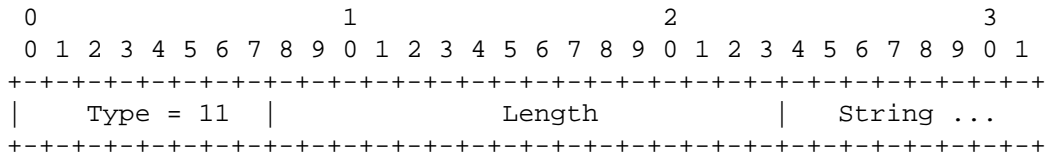
4-bit sequence number

4.2.2.11 HMAC-Digest

Description

This Attribute contains a keyed hash used for message authentication. The HMAC algorithm is defined in [RFC2104]. The HMAC algorithm is specified using a generic cryptographic hash algorithm. Baseline Privacy uses a particular version of HMAC that employs the Secure Hash Algorithm (SHA-1), defined in [SHA].

A summary of the HMAC-Digest Attribute format is shown below. The fields are transmitted from left to right.



Type

11 for HMAC-Digest

Length

20-octets

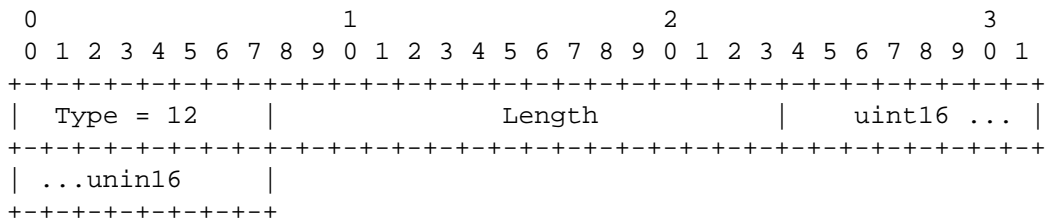
String

A 160-bit (20 octet) keyed SHA hash

4.2.2.12 SID

Description

This Attribute contains a 14-bit SID used by MCNS Privacy as the security association identifier. The two high-order bits will be set to zero.



Type

12 for SID

Length

2

uint16

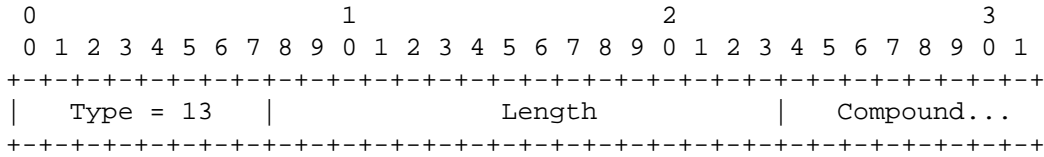
16-bit quantity representing a SID

4.2.2.13 TEK-Parameters

Description

This Attribute is a compound attribute, consisting of a collection of sub-attributes. These sub-attributes represent all security parameters relevant to a particular generation of a SID's TEK.

A summary of the TEK-Parameters Attribute format is shown below. The fields are transmitted from left to right.



Type

13 for TEK-Parameters

Length

42

Compound

The Compound field contains the following sub-Attributes:

Table 4-15. TEK-Parameters Sub-Attributes

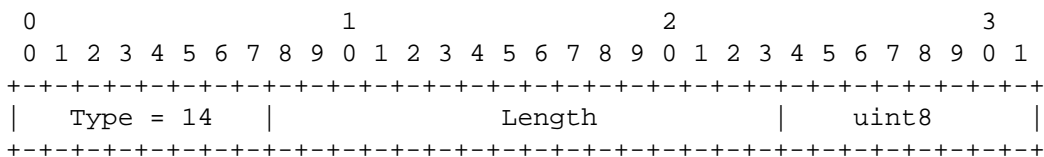
| Attribute | Contents |
|---------------------|--------------------------------------------|
| TEK-KEY | TEK, encrypted (DES-ECB mode) with the KEK |
| Key-Lifetime | TEK Remaining Lifetime |
| Key-Sequence-Number | TEK Sequence Number |
| DES-CBC-IV | Initialization Vector |

4.2.2.14 SA-Flag

Description

This Attribute contains a one-octet flag providing information on a particular security association. Currently, the low order bit is defined to indicate whether the security association corresponds to a multicast service – off (0) indicates unicast, on (1) indicates multicast.

A summary of the SA-Flag Attribute format is shown below. The fields are transmitted from left to right.



Type

14 for SA-Flag

Length

1

uint8

1-octet bit flag

```

0 1 2 3 4 5 6 7
+-----+
|0|0|0|0|0|0|0|M|
+-----+

```

M = 0 indicates unicast

M = 1 indicates multicast

4.2.2.15 DES-CBC-IV

Description

This Attribute contains a 64-bit (8-octet) value specifying a DES-CBC Initialization Vector.

A summary of the HMAC-Digest Attribute format is shown below. The fields are transmitted from left to right.

```

0                               1                               2                               3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-----+-----+-----+-----+
|  Type = 15  |           Length           |  String ...  |
+-----+-----+-----+-----+

```

Type

15 for DES-CBC-IV

Length

8 octets

String

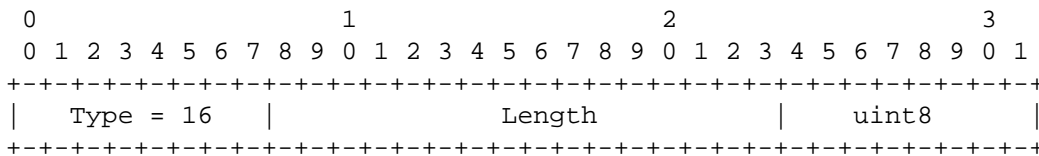
A 64-bit DES-CBC initialization vector.

4.2.2.16 Error-Code

Description

This Attribute contains a one-octet error code providing further information about an Authorization Reject, Key Reject, Authorization Invalid, or TEK Invalid.

A summary of the Error-Code Attribute format is shown below. The fields are transmitted from left to right.



Type

16 for Error-Code

Length

1

uint8

1-octet error code

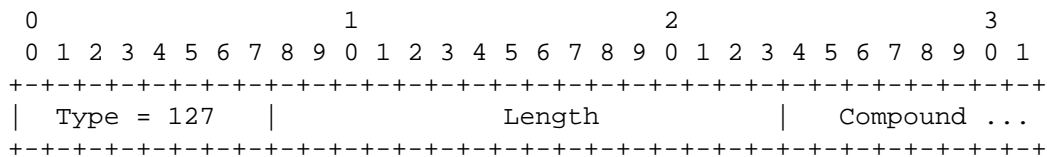
A CMTS MUST include the Error-Code Attribute in all Authorization Reject, Authorization Invalid, Key Reject and TEK Invalid messages. Table 4-16 lists code values for use with this Attribute. The CMTS MAY employ the nonzero error codes (1-5) listed below; it MAY, however, return a code value of zero (0). Error code values other than those defined in Table 4-16 MUST be ignored. Returning a code value of zero sends no additional failure information to the CM; for security reasons, this may be desirable. The BPKM state machines do not interpret Error-Code Attribute values.

Table 4-16. Error-Code Attribute Code Values

| Error Code | Messages | Description |
|------------|---------------------------|----------------------------------------------|
| 0 | all | no information |
| 1 | Auth Reject, Auth Invalid | Unauthorized CM |
| 2 | Auth Reject, Key Reject | Unauthorized SID |
| 3 | Auth Invalid | Unsolicited |
| 4 | Auth Invalid, TEK Invalid | Invalid Key Sequence Number |
| 5 | Auth Invalid | Message (Key Request) authentication failure |

4.2.2.17 Vendor-Defined

The Vendor-Defined Attribute is a compound attribute whose first sub-attribute **MUST** be the Manufacturer-ID Attribute. Subsequent Attribute(s) are user defined, with Type values assigned by the vendor identified by the previous Manufacturer-ID Attribute.



Type

127 for Vendor-Defined

Length

≥ 6

Compound

The first sub-attribute **MUST** be Manufacturer-ID. Subsequent attributes can include both universal Types (i.e., defined within this specification) and vendor-defined Types, specific to the vendor identified in the preceding Manufacturer-ID sub-attribute.

5 Key Usage

5.1 Cable Modem

Message authentication keys (used in calculating and verifying HMAC-Digests) and key encryption keys are derived from the authorization key. A cable modem maintains a single active authorization key.

A CM MUST use its current authorization key when generating a Key Request's HMAC-Digest.

A cable modem MUST use its current authorization key when:

- verifying a Key Reply or Key Reject's HMAC-Digest
- decrypting the TEK-KEY sub-attribute in a Key Reply's TEK-Parameters attribute

A CM MUST be capable of maintaining two successive sets of traffic keying material per authorized SID. Through operation of its TEK state machines, a CM attempts to always maintain a SID's two most recent sets of traffic keying material. A CM, however, need not track the expiration time of a SID's TEKs and delete the expired TEKs from its key tables.

For each of its authorized SIDs, the cable modem:

- MUST use its most recent TEK to encrypt upstream traffic
- MUST be able to decrypt downstream traffic encrypted with either its most recent TEK or its immediate predecessor (i.e., the TEK whose key sequence number is one less than that of the most recent TEK)

The KEY_SEQ field in the Privacy EH element identifies the key sequence number of the TEK used to encrypt the PDU's packet data. The TOGGLE bit in the Privacy EH element, which is equal to the least significant bit of the KEY_SEQ field, assists in distinguishing between two successive key generations.

5.2 CMTS

The CMTS MUST generate a new authorization key for each Authorization Reply it sends. The CMTS MUST always use the most recent authorization key it has sent to a CM. The CMTS uses these authorization keys for:

- verifying the HMAC-Digest in Key Requests received from that CM
- encrypting (ECB mode DES) the TEK in the Key Replies it sends to that CM (TEK contained in TEK-KEY sub-attribute of Key Reply's TEK-Parameters Attribute)
- calculating the HMAC-Digests it writes into Key Replies, Key Rejects and TEK Invalids sent to that CM

For each SID, the CMTS maintains a timer for the associated keying material (TEK and CBC initialization vector). When a TEK's timer expires the CMTS MUST no longer use that TEK for either encryption or decryption. A CMTS MUST be prepared to provide a requesting CM with the next generation of keying material within a TEK's grace period. This grace period begins TEK Grace Time seconds before a TEK's scheduled expiration, and ends when the TEK expires (or is disabled through receipt of an implicit acknowledgment, see below). In between the time that the CMTS generates the new TEK and expires/disables the older TEK, the CMTS has two sets of valid keying material which can be used for encrypting/decrypting the SID's traffic.

During this key transition period, when the CMTS is maintaining two sets of valid keying material, key usage requirements for unicast and multicast SIDs differ. If a SID is unicast:

- the CMTS MUST continue to use the older TEK for encrypting packet data until the CMTS disables the older TEK
- while the older TEK remains active, the CMTS MUST decrypt upstream frames using either the old or new TEK; the KEY_SEQ field in the Baseline Privacy EH element identifies which of the two keys the frame's packet data was encrypted with
- once the CMTS disables the older TEK, the CMTS MUST use the newer TEK for encrypting downstream and decrypting upstream packet data
- the CMTS MUST disable the older TEK when it expires
- the CMTS MAY disable the older TEK, prior to its scheduled expiration time, when it receives a packet from the CM encrypted with the newer TEK (The receipt of an upstream Packet Data PDU encrypted with the newer TEK is an *implicit acknowledgment* that the CM received and installed the newer key.)

Note that once the older TEK times out, the CMTS MUST begin using the newer key, regardless of whether it has seen an implicit acknowledgment. In other words, it is the responsibility of the CM to update its keys in a timely fashion.

The CMTS is responsible for maintaining keying information for unicast and multicast SIDs. The Baseline Privacy Key Management protocol defined in this specification describes a mechanism for synchronizing this keying information between CMTS and its client CMs; it does not specify how a CMTS should generate or maintain keying material. By not requiring a specific procedure or schedule for generating keying material, vendors are afforded a great degree of flexibility in their designs and implementations of BPI within the CMTS.

BPI does specify the relative expiration times of a SID's successive keys. Within the CMTS, TEK expiration times for a unicast SID MUST be scheduled between $N-G$ and N seconds after the scheduled expiration of the previous TEK, where N is the TEK lifetime of the new key, and G is the configured TEK Grace Time. In other words, if the older TEK expires at time T , the scheduled expiration time of the new TEK MUST occur between $T+N-G$ and $T+N$. This insures that the length of time a TEK is actively engaged in the encryption/decryption of data traffic never exceeds the configured TEK lifetime.

In the case of multicast SIDs, the CMTS's key usage policy during this key transition period is as follows:

- the CMTS MUST NOT begin encrypting downstream data traffic with the newer TEK until the older TEK expires
- once the older TEK times out, the CMTS MUST begin using the newer key for encryption

Within the CMTS, TEK expiration times for multicast SIDs MUST be set from the expiration times of the previous TEK; e.g., if the older TEK expires at time T, and the TEK lifetime is N, then the expiration time for the new TEK is T+N.

The lifetime a CMTS sends to a client CM in its Key Replies MUST reflect the remaining lifetime of the requested unicast or multicast key.

5.3 Multicast Key Handling

During a multicast SID's key transition period, a CMTS's Key Replies to requests for the SID's keying data MUST contain two TEK-Parameters Attributes, one for each of the two sets of keying material active during that transition period. This is to insure that a CM that boots up during a multicast SID's key transition period obtains the keying material it needs to decrypt a multicast transmission during that transition period. Note that if the CMTS only distributed the multicast SID's latest key, the CM would not be able to decrypt multicast transmissions associated with that SID until the CMTS began using the newer key, i.e., at the end of the key transition period when the older key expires.

Unicast Key Replies, and multicast Key Replies sent outside of a key transition period MAY contain two SA-Parameter Attributes. One TEK-Parameters Attribute will contain the latest generation of keying material. The Key-Lifetime Sub-attribute within the second TEK-Parameters Attribute indicates whether it carries valid keying material. A Key-Lifetime of zero indicates that the corresponding TEK-Parameters Attribute does not carry valid keying material; a non-zero Key-Lifetime indicates that the older generation of keying material it corresponds to is still active within the CMTS.

If, immediately following CM bootup, the first Key Reply to a multicast SID Key Request contains two active sets of keying material, the CM MUST incorporate both sets of keying material into its key database. If a Key Reply containing two active sets of keying material is (1) associated with a unicast SID, or (2) associated with the rekeying of an operational multicast SID, the older set of keying material MAY be incorporated into the CM's key database or MAY be ignored.

6 Cryptographic Methods

Baseline Privacy MUST use the Cipher Block Chaining (CBC) mode of the US Data Encryption Standard (DES) algorithm [FIPS-46, FIPS-46-1, FIPS-74, FIPS-81] to encrypt the Packet Data field in both upstream and downstream RF MAC Packet Data PDUs. Baseline Privacy MUST operate with either 56-bit or 40-bit DES encryption of user data. The BPKM protocol, however, always generates and distributes 56-bit DES traffic encryption keys. The restriction to 40-bit DES encryption is implemented within hardware. 40-bit DES is identical to 56-bit DES, except that 16 specific bits of the 56-bit key are set to known, fixed values. CM and CMTS hardware running 40-bit DES MUST mask off (to zero) the sixteen leftmost bits of any 56-bit DES key prior to running encryption/decryption operations.

CBC MUST be initialized with an initialization vector that is provided, along with other SID key material, in a CMTS's Key Reply. Chaining is done block to block within a frame and reinitialized on a frame basis in order to make the system more robust to potential frame loss.

Residual termination block processing MUST be used to encrypt the final block of Packet Data PDU payload when the final block of the payload is less than 64 bits. Given a final block having n bits, where n is less than 64, the next-to-last ciphertext block is DES encrypted a second time, using the ECB mode, and the least significant n bits of the result are exclusive ORed with the final n bits of the payload to generate the short final cipher block. In order for the receiver to decrypt the short final cipher block, the receiver DES encrypts the next-to-last ciphertext block, using the ECB mode, and exclusive ORs the leftmost n bits with the short final cipher block in order to recover the short final cleartext block. This encryption procedure is depicted in Figure 9.4 (pg. 195) of [SCHNEIER].

In the special case when the complete Packet Data PDU payload is less than 64 bits, the initialization vector MUST be DES encrypted, and the leftmost n bits of the resulting ciphertext corresponding to the number of bits of the payload MUST be exclusive ORed with the n bits of the payload to generate the short cipher block.

TEK keys in Key Reply messages MUST be encrypted with ECB mode 56-bit DES, using the KEK as the key. The BPKM protocol employs 56-bit DES for encrypting traffic encryption keys, regardless of whether 40-bit or 56-bit DES encryption of packet data is employed.

Authorization keys in Authorization Reply messages MUST be (RSA) public-key encrypted, using the cable modem's public key. Baseline Privacy uses F4 (65537 decimal, or equivalently, 010001 hexadecimal) as its public exponent and a modulus length of 768 bits (96 octets). The RSA public-key encryption process is described in [RSA1].

The keyed hash employed by the HMAC-Digest Attribute MUST use the HMAC message authentication method defined in [RFC 2104]. Baseline Privacy MUST use SHA-1 for its cryptographic hash. SHA-1 is specified in [SHA].

A key encryption key (KEK) and two message authentication keys are derived from a common authorization key. The following defines how these keys are derived:

KEK is the Key Encryption Key used to encrypt Traffic Encryption Keys.

HMAC_KEY_U is the message authentication key used in upstream Key Requests

HMAC_KEY_D is the message authentication key used in downstream Key Replies, Key Rejects and TEK Invalids.

SHA(x|y) denotes the result of applying the SHA function to the concatenated bit strings x and y.

Truncate(x,n) denotes the result of truncating x to its leftmost n bits.

KEK = Truncate(SHA(K_PAD | AUTH_KEY), 64)

HMAC_KEY_U = SHA(H_PAD_U | AUTH_KEY)

HMAC_KEY_D = SHA(H_PAD_D | AUTH_KEY)

Each _PAD_ is a 512 bit string:

K_PAD = 0x53 repeated 64 times.

H_PAD_U = 0x5C repeated 64 times.

H_PAD_D = 0x3A repeated 64 times.

Appendix A: TFTP Configuration File Extensions

All of a CM's Baseline Privacy configuration parameter values are specified in the configuration file TFTP-downloaded by the CM during RF MAC initialization. Baseline Privacy configuration setting fields are included in both the CM MIC and CMTS MIC calculations, and in a CM's registration requests. Refer to [MCNS1] for the order in which Baseline Privacy configuration setting fields are included in the CMTS MIC's MD5 digest.

A.1 Encodings

The following type/length/value encodings for Baseline Privacy configuration settings **MUST** be used in both the configuration file and in RF MAC CM registration requests. All multi-octet quantities are in network-byte order, i.e., the octet containing the most-significant bits is the first transmitted on the wire.

A.1.1 Baseline Privacy Configuration Setting

The presence of this configuration setting serves as an indication that the CM is configured to run Baseline Privacy. Therefore, the configuration setting **MUST** be present if the CM is provisioned to run Baseline Privacy, and **MUST NOT** be present if the CM is provisioned not to run baseline privacy.

This field defines the parameters associated with Baseline Privacy operation. It is composed of a number of encapsulated type/length/value fields. The type fields defined are only valid within the encapsulated Baseline Privacy configuration setting string.

| type | length | value |
|--------|--------|-------|
| BP_CFG | n | |

[MCNS1] defines the specific value of BP_CFG.

A.1.1.1 Internal Baseline Privacy Encodings

A.1.1.1.1 Authorize Wait Timeout

The value of the field specifies retransmission interval, in seconds, of Authorization Request messages from the Authorize Wait state.

| sub-type | length | value |
|----------|--------|-------|
| 1 | 4 | |

Valid Range: 1 - 30

A.1.1.1.2 Reauthorize Wait Timeout

The value of the field specifies retransmission interval, in seconds, of Authorization Request messages from the Authorize Wait state.

| sub-type | length | value |
|----------|--------|-------|
| 2 | 4 | |

Valid Range: 1 - 20

A.1.1.1.3 Authorization Grace Time

The value of this field specifies the grace period for re-authorization, in seconds.

| sub-type | length | value |
|----------|--------|-------|
| 3 | 4 | |

Valid Range: 1 - 1800

A.1.1.1.4 Operational Wait Timeout

The value of this field specifies the retransmission interval, in seconds, of Key Requests from the Operational Wait state.

| sub-type | length | value |
|----------|--------|-------|
| 4 | 4 | |

Valid Range: 1 - 10

A.1.1.1.5 Rekey Wait Timeout

The value of this field specifies the retransmission interval, in seconds, of Key Requests from the Rekey Wait state.

| sub-type | length | value |
|----------|--------|-------|
| 5 | 4 | |

Valid Range: 1 - 10

A.1.1.1.6 TEK Grace Time

The value of this field specifies the grace period for re-keying, in seconds.

| sub-type | length | value |
|----------|--------|-------|
| 6 | 4 | |

Valid Range: 1 - 1800

A.1.1.1.7 Authorize Reject Wait Timeout

The value of this field specifies how long a CM waits in the Authorize Reject Wait state after receiving an Authorization Reject.

| sub-type | length | value |
|----------|--------|-------|
| 7 | 4 | |

A.1.2 Class-of-Service Privacy Enable

This configuration setting is one of the type/length/value fields encapsulated within the Class of Service Configuration Setting (type 4), defined in [MCNS1]. Its value indicates whether Baseline Privacy is enabled for a particular provisioned class of service. If the value field is 1, Baseline Privacy will run on the SID that gets associated with the class of service; if the value field is 0, Baseline Privacy will not run on the associated SID. Note that SIDs are assigned to a provisioned class of service during CMTS registration.

| sub-type | length | on/off |
|---------------|--------|--------|
| CoS_BP_ENABLE | 1 | 1 or 0 |

[MCNS1] defines the specific value of CoS_BP_ENABLE.

A.2 Parameter Guidelines

Below are recommended ranges and values for Baseline Privacy's various configuration and operational parameters. These ranges and default values may change as service providers gain operational experience running Baseline Privacy.

Table A-1. Recommended Operational Ranges for BPI Configuration Parameters

| System | Name | Description | Minimum Value | Default Value | Maximum Value |
|--------|--------------------------|-------------------------------------------------------------------|------------------------|---------------------------|-----------------------------|
| CMTS | Authorization Lifetime | Lifetime, in seconds, CMTS assigns to new authorization key | 1 day (86,400 sec.) | 7 days (604,800 sec.) | 70 days (6,048,000 sec.) |
| CMTS | TEK Lifetime | Lifetime, in seconds, CMTS assigns to new TEK | 30 min. (1800 sec.) | 12 hours (43,200 sec.) | 7 days (604,800 sec.) |
| CM | Authorize Wait Timeout | Auth Req retransmission interval from Auth Wait state | 2 sec. | 10 sec. | 30 sec. |
| CM | Reauthorize Wait Timeout | Auth Req retransmission interval from Reauth Wait state | 2 sec. | 10 sec. | 30 sec. |
| CM | Authorization Grace Time | Time prior to Authorization expiration CM begins re-authorization | 5 min. (300 sec.) | 10 min. (600 sec.) | 30 min. (1800 sec.) |
| CM | Operational Wait Timeout | Key Req retransmission interval from Op Wait state | 1 sec. | 1 sec. | 10 sec. |
| CM | Rekey Wait Timeout | Key Req retransmission interval from Rekey Wait state | 1 sec. | 1 sec. | 10 sec. |
| CM | TEK Grace Time | Time prior to TEK expiration CM begins re-keying | 5 min. (300 sec.) | 10 min. (600 sec.) | 30 min. (1800 sec.) |
| CM | Authorize Reject Wait | Delay before resending Auth Request after receiving Auth Reject | 10 sec. | 60 sec. | 10 min. (600 sec.) |

The valid range (vs. recommended operational range) for Authorization and TEK lifetimes are:

- Authorization Lifetime Valid Range: 1 - 6,048,000 seconds
- TEK Lifetime Valid Range: 1 - 604,800 seconds

Note that valid ranges defined for each of BPI's configuration parameters extend below the recommended operational ranges. For the purposes of protocol testing, it is useful to run the BPI protocol with timer values well below the low end of the recommended operational ranges. The shorter timer values "speed up" BPI's clock, causing BPI protocol state machine events to occur far more rapidly than they would under an "operational" configuration. While BPI implementations need not be designed to operate efficiently at this accelerated BPI pace, the protocol implementation SHOULD operate correctly under these shorter timer values. Table A-2 provides a list of shortened parameter values which are likely to be employed in protocol conformance and certification testing.

Table A-2. Shortened BPI Parameter Values for Protocol Testing

| | |
|--------------------------|-------------------|
| Authorization Lifetime | 5 min. (300 sec.) |
| TEK Lifetime | 3 min. (180 sec.) |
| Authorize Wait Timeout | 1 min. (60 sec.) |
| Reauthorize Wait Timeout | 1 min. (60 sec.) |

The TEK Grace Time **MUST** be less than half the TEK Lifetime; this insures there are at most two active (not expired) TEKs per SID at any time.

This page intentionally left blank.

Appendix B: Example Messages and PDUs (informative)

This appendix presents numerical examples which may be useful to implementors of the specification. The examples walk through a typical key exchange: Authorization Request, Authorization Reply, Key Request, and Key Reply. Details of the cryptographic calculations are provided at each step. The examples also include several Packet PDUs, encrypted using the keying material derived in the example key exchange.

This appendix is informative only and does not constitute any part of the specification.

B.1 Notation

In the examples here, packets are represented as a stream of octets, each octet in hex notation, sometimes with a text annotation. The order of transmission for the octets is left to right, top to bottom. For example, consider the following representation of a packet:

| | |
|-------------|----------------|
| 00 01 02 03 | Description #1 |
| 04 05 | |
| 06 07 08 | Description #2 |

The packet consists of 9 octets, represented in hex notation as “00”, “01”, ..., “08”. The octet represented by “00” is transmitted first, and the octet represented by “08” is transmitted last.

In the discussion of the examples, integer values are represented in either hex notation using an “0x” prefix or in decimal notation with no prefix. For example, the hex notation 0x12345 and the decimal notation 74565 represent the same integer value. All integer values are non-negative. Thus, 0xff represents the integer having value 255, not a negative value.

The BPKM protocol generates and distributes 8-octet keys, without correcting the least significant bit of each octet for parity. Implementations extract a 56-bit key from an 8-octet key by ignoring the value of the least significant bit of each octet. In the examples here, keys are represented without parity correction.

B.2 Authorization Request

The CM sends the following Authorization Request:

| | |
|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------|
| 04 72 00 8b | Auth Request header |
| 05 00 83 | CM-Identification header |
| 01 00 04 31 32 33 34 | Serial Number |
| 02 00 03 55 53 41 | Manufacturer ID |
| 03 00 06 4d 41 43 41 44 44 | MAC Address |
| 04 00 6a 30 68 02 61 00 d3 f4 84 b8 23 ce 70 35 e7 ab 32 30 43 13 ff ff 1b 26 c2 f8 7f e6 e5 0f 22 9a ed 80 13 d8 1d 95 b4 30 87 f0 b5 ab 50 de b1 d8 82 b2 42 a9 67 33 d9 e5 c2 b1 2a 0d 42 58 94 63 0b 11 0e a0 55 96 b1 cf c0 7f 14 97 46 bc 44 13 4b 5e 4a de 46 ce eb c8 b6 35 8a 66 9b 33 c2 23 75 f5 c8 69 79 65 02 03 01 00 01 | RSA Public Key |
| 0c 00 02 22 60 | SID |

The Code field has value 0x04, which identifies this as an Authorization Request packet. The Identifier field has value 0x72; this is an example value. The Length field has value 0x008b (139), which is the number of octets that follow the Length field.

The first attribute is the CM Identification. It is a compound attribute consisting of the following sub-attributes: Serial Number, Manufacturer ID, MAC Address, and RSA Public Key. Example values are shown for these sub-attributes.

The RSA Public Key is BER encoded and is similar to the example in section 2.2 of [RSA2]. The modulus is a 768-bit integer represented using 0x61 (97) octets. In this example, the value of the modulus is:

0x00d3f484b8 ... c8697965.

Notice that 0x00 is the most significant octet of the modulus and 0x65 is the least significant. The exponent is an integer made up of 3 octets and having value 0x010001.

The remaining attributes are the SIDs obtained by the CM during RF registration. In this example, a single SID is included, having value 0x2260.

B.3 Authorization Reply

The CMTS sends the following Authorization Reply:

| | |
|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------|
| 05 72 00 73 | Auth Reply header |
| 07 00 60 ce 7f 8e fe a3 c6 e0 16 bf 31 d9 c9 83 8b c9 f2 6c c6 a5 56 64 65 ac b6 97 78 2b e6 c3 fe dc c9 4b b4 d8 6c 2c dc 87 65 a6 c4 d5 a4 b1 25 b6 e0 ef 76 2a f0 7a 4e 52 b9 0e 7c 18 a7 3b fa 2e 6a bc c0 78 12 de 0e 81 7b 0c b9 68 32 45 55 35 4b 4c ee b1 e2 8c 9d 16 14 a0 10 08 d6 3a c4 c4 80 | Auth Key |
| 09 00 04 00 09 3a 80 | Key Lifetime |
| 0a 00 01 07 | Key Sequence number |
| 0c 00 02 22 60 | SID |

The Code field has value 0x05, which identifies this as an Authorization Reply packet. The Identifier field has value 0x72, matching the Identifier field of the Authorization Request. The Length field has value 0x0073 (115), which is the number of octets that follow the Length field.

The first attribute is the Authorization Key. The attribute contains an authorization key which has been RSA-encrypted using the public key in the Authorization Request message. The RSA-encrypted authorization key is an integer made up of 0x60 (96) octets. In this example, the value of the RSA-encrypted authorization key is:

0xce7f8efe ... 3ac4c480.

Notice that 0xce is the most significant octet of the RSA-encrypted authorization key and 0x80 is the least significant. Details of the RSA encryption calculation are given below.

The second attribute is the Key Lifetime. In this example, the value is 0x00093a80 (604800) seconds, or 7 days.

The third attribute is the Key Sequence Number. In this example, the value is 0x07.

The remaining attributes are the SIDs identifying the specific data services the CM is authorized to access. In this example, a single SID is included, matching the SID in the Authorization Request.

The CM and CMTS each derive a key encryption key and two message authentication keys from the authorization key, using hashing. Details of the hashing calculations are given below. Here are the values of these keys for this example:

| | |
|----------------------------------------|-------------------------------------------------------------|
| Authorization Key | 3b d5 50 60 bd a2 57 c0 |
| Key Encryption Key | 5f 59 05 1d 92 17 d9 83 |
| Message Authentication Key, upstream | eb ff 98 cd 5c d4 57 bb fd 12 b5 65 ff aa f6 89 d4 98 26 14 |
| Message Authentication Key, downstream | 5e 47 69 83 9e ee e4 d0 04 a4 c1 23 80 b0 5a d1 8a c9 2c 9c |

B.3.1 RSA encryption details

The CMTS generates a random authorization key of 8 octets. In this example, the value of the authorization key is:

3b d5 50 60 bd a2 57 c0

The authorization key is formatted into an encryption block of 96 octets as per the PKCS#1 specification [RSA1]:

```
00 02 bc e3 80 66 b6 f5 4c de f1 9b e0 4b e3 85 ba 0e cd e0 7d ff 36
d0 dc 35 7f e3 53 50 66 05 59 2c 76 91 9d 46 23 ec 0e 17 eb 95 65 c1
9e 5d 09 a0 0a a1 ff c7 cd 48 38 ec 52 c2 9f 54 a8 a0 7e 15 bb 5c 9d
77 4e c0 dc 27 f7 9b 67 55 72 9c df cc 87 04 52 74 eb 00 3b d5 50 60
bd a2 57 c0
```

The first octet of the block is 0. The second octet specifies the block type. PKCS#1 specifies a block type value of 2 for public key operations, and thus the Authorization Reply must always use this block type. The next 85 octets are random, nonzero, padding octets, as required for block type 2. The padding is followed by an octet of value 0. The last 8 octets of the block are the authorization key. To perform RSA encryption, the block is interpreted as the integer value:

0x0002bce3 ... bda257c0

Notice that 0x00 is the most significant octet and 0xc0 is the least significant.

The RSA encryption is performed as the operation $Y = M^E \text{ mod } N$, where:

M is the integer value of the RSA encryption block (0x0002bce3 ... bda257c0);

E is the integer value of the exponent of the RSA public key (0x010001);

N is the integer value of the modulus of the RSA public key (0xd3f484b8 ... c8697965);

Y is the integer value of the RSA-encrypted authorization key (0xce7f8efe ... 3ac4c480).

B.3.2 RSA decryption details

Here is a table that lists the private-key parameters that match the RSA public key in the Authorization Request Message:

| Parameter | Property | Value |
|-------------------------|-------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| D (private exponent) | $M^{DE} \bmod N = M$ | 93 40 ce b0 0b 98 51 96 f1 b3 9b 73 b2 36 83 e4 dd 1d 29 d9 71 2a 9d ba cd 24 eb 99 ce af 97 a6 39 00 7a 81 3e 05 9e 72 89 06 4d 8e c1 07 66 3d 78 e9 67 7d ab e5 e8 f3 4c 70 4e 5c fe fb 15 25 8f 37 54 ed da 01 8e c8 de fe 60 f5 a1 bd e8 6c 4f c5 71 61 a6 6f a7 55 c9 29 d8 ca 8b f1 c5 85 |
| P (prime factor) | $N = PQ$ | f1 21 1e d1 49 ff 95 33 e5 60 de 4c ae ae f6 75 11 8b 96 d5 45 45 96 ee 88 a3 3d d9 a1 17 66 42 ae 75 7b c6 18 0a 0d 78 39 b5 5e 99 16 c6 2d db |
| Q (prime factor) | $N = PQ$ | e1 06 cd d0 08 c9 15 09 14 93 8a 8c a5 d7 84 b8 56 15 42 38 b7 e7 8e 89 cb d6 56 9f 59 76 20 10 7f 5e 6e e1 c3 ea 31 41 90 e0 8e 24 47 5e b9 bf |
| D_p (CRT exponent) | $D_p = D \bmod (P - 1)$ | bf bf ea d5 db d6 97 3b d1 a8 9a 9e b8 3b 02 5a 4e 3d 87 10 ca 29 70 c0 f7 7f 78 eb db a2 d3 fb 2a e8 da 28 c9 6d 15 11 0a 33 24 aa f0 e5 60 09 |
| D_q (CRT exponent) | $D_q = D \bmod (Q - 1)$ | 32 02 96 19 06 ea d1 8e fc 10 b2 39 01 de 7c f3 8e c6 18 ba 8c 3c 9d 14 08 c6 30 e8 27 34 b6 79 94 25 03 95 8f 39 ec 0a 7b 4d 3c a9 d9 66 b6 f7 |
| U_p (CRT constant) | $PU_p \bmod Q = 1$ | 55 8c 66 4e ca 5e e6 9c c8 df 47 a6 7a e0 25 e8 7b 9d 4c e9 b0 96 39 0a a4 a2 37 45 c8 5e 5f 9d d1 58 9c 88 fa 75 1c 78 ec b2 3d d6 79 24 d6 f7 |
| U_q (CRT constant) | $QU_q \bmod P = 1$ | 95 75 89 c7 c5 02 81 a5 59 19 8f fa ca 0a bf b8 8d 1b c9 b9 79 fb d1 4f 1f b3 ca 32 30 d7 d1 58 ed 04 64 c5 35 b2 db 21 81 ef b0 fd 6f f1 63 27 |

Each value in the table represents the octets of an integer, with the most significant octet shown first. For example, the private exponent D has the integer value:

0x9340ceb0 ... 8bf1c585.

The CM can decrypt the authorization key with or without using the Chinese Remainder Theorem (CRT). Decryption using the CRT is more complicated, but it may be a faster operation.

To decrypt without using the CRT, the CM performs the operation $M = Y^D \bmod N$. D is the private exponent in the table, and Y and N are as described in the preceding section. The resulting value matches the value of M in the preceding section, that is, it is the integer value that represents the PKCS#1 block formed by the CMTS. The last 8 octets of the block are the authorization key.

To decrypt using the CRT, the CM first computes two intermediate quantities:

$$A = Y^{D_p} \bmod P;$$

$$B = Y^{D_q} \bmod Q.$$

P and Q are the prime factors of the modulus, and D_p and D_q are private exponents related to these factors, all with values shown in the table. The CM computes the value of M in one of two ways:

$$M = A + ((B - A)U_p \bmod Q)P;$$

or

$$M = ((AU_q \bmod P)Q + (BU_p \bmod Q)P) \bmod N.$$

U_p and U_q are constants derived from the prime factors, with values as shown in the table. With either of these computations, the resulting value matches the value that would be computed using the operation $M = Y^D \bmod N$.

B.3.3 Hashing details

The authorization key is hashed using the SHA-1 algorithm [SHA] to produce the Key Encryption Key (KEK), the message authentication key for upstream, and the message authentication key for downstream.

The discussion here represents a hash calculation using a table that shows the input to the hash function and the resulting hash value. For reference, here is such a table that describes the example in Appendix B of [SHA]:

| | |
|------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Hash input | 61 62 63 64 62 63 64 65 63 64 65 66 64 65 66 67 65 66 67 68 66 67 68 69 67 68 69 6a 68 69 6a 6b 69 6a 6b 6c 6a 6b 6c 6d 6b 6c 6d 6e 6c 6d 6e 6f 6d 6e 6f 70 6e 6f 70 71 |
| Hash value | 84 98 3e 44 1c 3b d2 6e ba ae 4a a1 f9 51 29 e5 e5 46 70 f1 |

B.3.3.1 KEK

The KEK is computed using the following hash calculation:

| | |
|------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Hash input | 53 53 53 53 53 53 53 53 53 53 53 53 53 53 53 53 53 53 53 53 53 53 53 53 53 53 53 53 53 53 53 53 53 53 53 53 53 53 53 53 53 53 53 53 53 53 53 53 53 53 53 53 53 53 53 53 53 53 53 3b d5 50 60 bd a2 57 c0 |
| Hash value | 5f 59 05 1d 92 17 d9 83 4e 89 ec 4b 47 7d 8c 69 04 30 99 a9 |

| | |
|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------|
| 07 73 00 a6 | Key Request Header |
| 05 00 83 | CM-Identification header |
| 01 00 04 31 32 33 34 | Serial Number |
| 02 00 03 55 53 41 | Manufacturer ID |
| 03 00 06 4d 41 43 41 44 44 | MAC Address |
| 04 00 6a 30 68 02 61 00 d3 f4 84 b8 23 ce 70 35 e7 ab 32 30 43 13 ff ff 1b 26 c2 f8 7f e6 e5 0f 22 9a ed 80 13 d8 1d 95 b4 30 87 f0 b5 ab 50 de b1 d8 82 b2 42 a9 67 33 d9 e5 c2 b1 2a 0d 42 58 94 63 0b 11 0e a0 55 96 b1 cf c0 7f 14 97 46 bc 44 13 4b 5e 4a de 46 ce eb c8 b6 35 8a 66 9b 33 c2 23 75 f5 c8 69 79 65 02 03 01 00 01 | RSA public key |
| 0a 00 01 07 | Key Sequence Number |
| 0c 00 02 22 60 | SID |
| 0b 00 14 a3 55 a9 c3 61 85 ae a2 8d 20 ed ab c0 f5 6c 4f 2f a1 97 e0 | HMAC digest |

The Code field has value 0x07, which identifies this as a Key Request packet. The Identifier field has value 0x73; this is an example value, obtained by incrementing the Identifier value in the Authorization Request. The Length field has value 0x00a6 (166), which is the number of octets that follow the Length field.

The first attribute is the CM Identification. This is a compound attribute, identical to that in the Authorization Request.

The second attribute is the Key Sequence Number, which identifies the Authorization Key. The value is identical to that in the Authorization Reply.

The third attribute is the SID for which a key is being requested. This SID value was contained in the Authorization Reply.

The final attribute is the HMAC Digest. The digest consists of 20 octets. It is computed using the upstream Message Authentication Key. The digest is performed over all octets of the Key Request packet, excluding the 23 octets of the HMAC Digest attribute itself. Details of the digest calculation are given below.

B.4.1 HMAC digest details

The HMAC digest is computed using the HMAC authentication method defined in [RFC2104], with SHA-1 as the hash function. Example calculations of HMAC using SHA-1 are presented in [RFC2202].

The discussion here represents an HMAC calculation using a table that shows the key, the input to the HMAC function, and the resulting HMAC digest. For reference, here is a table that describes test case #2 of the HMAC-SHA-1 examples in [RFC2202]:

| | |
|-------------|-------------------------------------------------------------------------------------|
| Key | 4a 65 66 65 |
| HMAC input | 77 68 61 74 20 64 6f 20 79 61 20 77 61 6e 74 20 66 6f 72 20 6e 6f 74 68 69 6e 67 3f |
| HMAC digest | ef fc df 6a e5 eb 2f a2 d2 74 16 d5 f1 84 df 9c 25 9a 7c 79 |

The HMAC digest of the Key Request packet is computed using the following HMAC calculation:

| | |
|-------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Key | eb ff 98 cd 5c d4 57 bb fd 12 b5 65 ff aa f6 89 d4 98 26 14 |
| HMAC input | 07 73 00 a6 05 00 83 01 00 04 31 32 33 34 02 00 03 55 53 41 03 00 06 4D 41 43 41 44 44 04 00 6a 30 68 02 61 00 d3 f4 84 b8 23 ce 70 35 e7 ab 32 30 43 13 ff ff 1b 26 c2 f8 7f e6 e5 0f 22 9a ed 80 13 d8 1d 95 b4 30 87 f0 b5 ab 50 de b1 d8 82 b2 42 a9 67 33 d9 e5 c2 b1 2a 0d 42 58 94 63 0b 11 0e a0 55 96 b1 cf c0 7f 14 97 46 bc 44 13 4b 5e 4a de 46 ce eb c8 b6 35 8a 66 9b 33 c2 23 75 f5 c8 69 79 65 02 03 01 00 01 0a 00 01 07 0c 00 02 22 60 |
| HMAC digest | a3 55 a9 c3 61 85 ae a2 8d 20 ed ab c0 f5 6c 4f 2f a1 97 e0 |

The key is the upstream message authentication key. The input consists of all octets of the Key Request packet, excluding the HMAC Digest attribute. The octets of the digest are the contents of the HMAC Digest attribute.

B.5 Key Reply

The CMTS sends the following Key Reply:

| | |
|----------------------------------------------------------------------|-----------------------------------------|
| 08 73 00 48 | Key Reply header |
| 0a 00 01 07 | Key Sequence Number (authorization key) |
| 0c 00 02 22 60 | SID |
| 0e 00 01 00 | SA Flag |
| 0d 00 21 | TEK Parameters header |
| 08 00 08 ab b9 d6 03 23 86 db ce | TEK Key |
| 09 00 04 00 00 a8 c0 | Key Lifetime |
| 0a 00 01 02 | Key Sequence Number (TEK) |
| 0f 00 08 81 0e 52 8e 1c 5f da 1a | DES CBC IV |
| 0b 00 14 ab 85 ee 28 19 b6 00 e6 95 22 94 3c 4a ac a1 e4 ea 7d db 02 | HMAC Digest |

The Code field has value 0x08, which identifies this as a Key Reply packet. The Identifier has 0x73, matching the value in the Key Request. The Length field has value 0x48 (72), which is the number of octets that follow the Length field.

The Key Sequence Number attribute identifies the Authorization Key. It matches the value in the Key Request.

The SID attribute identifies the SID for with a TEK is being supplied. It matches the value in the Key Request.

The SA Flag attribute indicates whether the TEK is for unicast or multicast usage. In this example, the value of 0x00 selects unicast.

The TEK Parameters attribute is a compound attribute consisting of the following sub-attributes: TEK Key, Key Lifetime, Key Sequence Number, and DES CBC IV.

The TEK Key consists of 8 octets. It contains the TEK, encrypted using DES-ECB with the KEK derived from the Authorization Key. Details of the DES-ECB calculation are given below.

The Key Lifetime sub-attribute refers to the TEK. In this example, the value is 0x0000a8c0 (43200) seconds, or 12 hours.

The Key Sequence Number sub-attribute identifies the TEK. In this example, the value is 0x02.

The DES CBC IV sub-attribute consists of 8 octets. It specifies the Initialization Vector to be used with the TEK.

The final attribute is the HMAC Digest. It consists of 20 octets. It is computed in a manner similar to that in the Key Reply, except that the downstream Message Authentication Key is used instead of the upstream key. Details of the HMAC calculation are given below.

After the CM processes the Key Reply packet, the CM and CMTS each share a TEK and IV. Here are the values of these parameters for this example:

| | |
|-----|-------------------------|
| TEK | e6 60 0f d8 85 2e f5 ab |
| IV | 81 0e 52 8e 1c 5f da 1a |

B.5.1 TEK encryption details

The CMTS generates a random TEK of 8 octets. In this example, the value of the TEK is:

e6 60 0f d8 85 2e f5 ab.

The TEK is encrypted using DES-ECB encryption [FIPS-81]. The encryption key is the KEK.

The discussion here represents a DES-ECB encryption using a table that shows the key, the plaintext input, and the ciphertext output. For reference, here is a table that describes the example in Table B1 of [FIPS-81]:

| | |
|------------|-------------------------|
| Mode | ECB |
| Key | 01 23 45 67 89 ab cd ef |
| Plaintext | 4e 6f 77 20 69 73 20 74 |
| Ciphertext | 3f a4 0e 8a 98 4d 48 15 |

Note: [FIPS-81] calls for the least significant bit of each octet in the key to be adjusted so that the octet has odd parity. This is evident in the key in the above example. The BPKM protocol does not require odd parity. BPKM generates and distributes 8-octet DES keys of arbitrary parity, and it requires that implementations ignore the value of the least significant bit of each octet.

The TEK Key field is computed using the following DES-ECB encryption:

| | |
|------------|-------------------------|
| Mode | ECB |
| Key | 5f 59 05 1d 92 17 d9 83 |
| Plaintext | e6 60 0f d8 85 2e f5 ab |
| Ciphertext | ab b9 d6 03 23 86 db ce |

The resulting ciphertext makes up the TEK Key sub-attribute.

B.5.2 HMAC details

The HMAC digest of the Key Reply packet is computed by a method similar to that of the Key Request packet. The key is the downstream message authentication key. Here are the details of the HMAC calculation:

| | |
|-------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Key | 5e 47 69 83 9e ee e4 d0 04 a4 c1 23 80 b0 5a d1 8a c9 2c 9c |
| HMAC input | 08 73 00 48 0a 00 01 07 0c 00 02 22 60 0e 00 01 00 0d 00 21 08 00 08 ab b9 d6 03 23 86 db ce 09 00 04 00 00 a8 c0 0a 00 01 02 0f 00 08 81 0e 52 8e 1c 5f da 1a |
| HMAC digest | ab 85 ee 28 19 b6 00 e6 95 22 94 3c 4a ac a1 e4 ea 7d db 02 |

B.6 Packet PDU encryption

The first 12 octets of the Packet PDU, containing the Ethernet/802.3 destination and source addresses (DA/SA), are not encrypted. The remaining octets of the Packet PDU are encrypted using DES-CBC mode with special handling of residual termination blocks that are less than 64 bits. The combination of DES-CBC and residual block processing ensures that the

encryption does not change the length of the packet. The encryption key is the TEK corresponding to the key sequence number of the packet's Privacy Extended Header.

The specification describes the residual block processing as follows:

“Given a final block having n bits, where n is less than 64, the next-to-last ciphertext block is DES encrypted a second time, using the ECB mode, and the least significant n bits of the result are exclusive ORed with the final n bits of the payload to generate the short final cipher block. ... In the special case where the Packet Data PDU payload is less than 64 bits, the initialization vector is DES encrypted, and the leftmost n bits of the resulting ciphertext corresponding to the number of bits of the payload are exclusive ORed with the n bits of the payload to generate the short cipher block.”

An alternative description of this procedure, which is equivalent to the description in the specification, is as follows:

Given a final block having n bits, where n is less than 64, the n bits are padded up to a block of 64 bits by appending $64-n$ bits of arbitrary value to the right of the n payload bits. The resulting block is DES encrypted using the CFB64 mode, with the next-to-last ciphertext block serving as initialization vector for the CFB64 operation. The leftmost n bits of the resulting ciphertext are used as the short cipher block. ... In the special case where the Packet Data PDU payload is less than 64 bits, the procedure is the same as for a short final block, with the provided initialization vector serving as the initialization vector for the DES-CFB64 operation.

The alternative description produces the same ciphertext as does the description in the specification. In the alternative description, however, no mention is made of combining ECB encryption with exclusive ORing. These operations are internal to CFB64, just as they are internal to CBC. The alternative description is convenient here because it allows residual block processing to be illustrated using CFB64 examples in [FIPS-81].

The Packet PDU includes the DA, SA, and Type/Len fields. In the examples here, no effort is made to use correct values for these fields. As a result, the examples here are not valid packets suitable for transmission. The intent of the examples is to illustrate encryption details only.

In these examples, the TEK and IV are taken from the example Key Reply packet described above.

B.6.1 CBC only

When the number of octets to be encrypted is a multiple of 8, the encryption mode is DES-CBC as defined in [FIPS-81]. The encryption key and IV are as conveyed in the Key Reply packet.

The discussion here represents a DES-CBC encryption using a table that shows the key, IV, plaintext input, and ciphertext output. For reference, here is a table that describes the example in Table C1 of [FIPS-81]:

| | |
|------------|-------------------------------------------------|
| Mode | CBC |
| Key | 01 23 45 67 89 ab cd ef |
| IV | 12 34 56 78 90 ab cd ef |
| Plaintext | 4e 6f 77 20 69 73 20 74 68 65 20 74 69 6d 65 20 |
| Ciphertext | e5 c7 cd de 87 2b f2 7c 43 e9 34 00 8c 38 9c 0f |

Suppose that the Packet PDU, prior to encryption, is as follows:

| | |
|-----------|-------------------------------|
| DA | 01 02 03 04 05 06 |
| SA | f1 f2 f3 f4 f5 f6 |
| Type/Len | 00 01 |
| User Data | 02 03 04 05 06 07 08 09 0a 0b |
| CRC | 88 41 65 06 |

The DES-CBC encryption is performed as follows:

| | |
|------------|-------------------------------------------------|
| Mode | CBC |
| Key | e6 60 0f d8 85 2e f5 ab |
| IV | 81 0e 52 8e 1c 5f da 1a |
| Plaintext | 00 01 02 03 04 05 06 07 08 09 0a 0b 88 41 65 06 |
| Ciphertext | 0d da 5a cb d0 5e 55 67 9f 04 d1 b6 41 3d 4e ed |

The Packet PDU, after encryption, looks like this:

| | |
|-----------|-------------------------------|
| DA | 01 02 03 04 05 06 |
| SA | f1 f2 f3 f4 f5 f6 |
| Type/Len | 0d da |
| User Data | 5a cb d0 5e 55 67 9f 04 d1 b6 |
| CRC | 41 3d 4e ed |

B.6.2 CBC with residual block processing

When the number of octets to be encrypted is greater than 8 and is not a multiple of 8, the encryption mode is a combination of DES-CBC and DES-CFB64.

Encryption begins in DES-CBC mode. DES-CBC is used to process as many complete DES blocks as are present. The encryption key and IV are as conveyed in the Key Reply packet.

After the DES-CBC encryption, there are 1 to 7 octets which have not been encrypted. These octets are encrypted using DES-CFB64 mode. DES-CFB64 is “64-bit Cipher Feedback Mode,” defined in [FIPS-81]. The encryption key is as in the Key Reply packet. The IV is the last 8 octets of ciphertext produced by the DES-CBC processing.

The discussion here represents a DES-CFB64 encryption using a table that shows the key, IV, plaintext input, and ciphertext output. For reference, here is a table that describes the example in Table D3 of [FIPS-81]:

| | |
|------------|-------------------------------------------------|
| Mode | CFB64 |
| Key | 01 23 45 67 89 ab cd ef |
| IV | 12 34 56 78 90 ab cd ef |
| Plaintext | 4e 6f 77 20 69 73 20 74 68 65 20 74 69 6d 65 20 |
| Ciphertext | f3 09 62 49 c7 f4 6e 51 a6 9e 83 9b 1a 92 f7 84 |

Suppose that the Packet PDU, prior to encryption, is as follows:

| | |
|-----------|----------------------------------------|
| DA | 01 02 03 04 05 06 |
| SA | f1 f2 f3 f4 f5 f6 |
| Type/Len | 00 01 |
| User Data | 02 03 04 05 06 07 08 09 0a 0b 0c 0d 0e |
| CRC | 91 d2 d1 9f |

The total number of octets to be encrypted is 19. The first 16 octets are processed using DES-CBC encryption, and the last 3 octets using DES-CFB64 encryption.

The DES-CBC encryption is performed as follows:

| | |
|------------|-------------------------------------------------|
| Mode | CBC |
| Key | e6 60 0f d8 85 2e f5 ab |
| IV | 81 0e 52 8e 1c 5f da 1a |
| Plaintext | 00 01 02 03 04 05 06 07 08 09 0a 0b 0c 0d 0e 91 |
| Ciphertext | 0d da 5a cb d0 5e 55 67 51 47 46 86 8a 71 e5 77 |

The DES-CFB64 encryption is performed as follows:

| | |
|------------|-------------------------|
| Mode | CFB64 |
| Key | e6 60 0f d8 85 2e f5 ab |
| IV | 51 47 46 86 8a 71 e5 77 |
| Plaintext | d2 d1 9f 00 00 00 00 00 |
| Ciphertext | ef ac 88 e8 ee 80 33 14 |

The key is the same as used for the DES-CBC encryption operation. The IV is the last 8 octets of ciphertext generated by the DES-CBC operation.

Notice that 5 octets of value 0 have been appended to the 3 plaintext octets. The values of these appended plaintext octets have no effect on the values of the first 3 ciphertext octets, which are the only ciphertext octets we are interested in. Arbitrary values can be used instead of 0 for the appended plaintext octets.

The Packet PDU, after encryption, looks like this:

| | |
|-----------|----------------------------------------|
| DA | 01 02 03 04 05 06 |
| SA | f1 f2 f3 f4 f5 f6 |
| Type/Len | 0d da |
| User Data | 5a cb d0 5e 55 67 51 47 46 86 8a 71 e5 |
| CRC | 77 ef ac 88 |

B.6.3 Runt frame

When the number of octets to be encrypted is less than 8, the encryption mode is DES-CFB64. The encryption key and IV are as conveyed in the Key Reply packet.

Suppose that the Packet PDU, prior to encryption, is as follows:

| | |
|-----------|-------------------|
| DA | 01 02 03 04 05 06 |
| SA | f1 f2 f3 f4 f5 f6 |
| Type/Len | 00 01 |
| User Data | 02 |
| CRC | 88 ee 59 7e |

The DES-CFB64 encryption is performed as follows:

| | |
|------------|-------------------------|
| Mode | CFB64 |
| Key | e6 60 0f d8 85 2e f5 ab |
| IV | 81 0e 52 8e 1c 5f da 1a |
| Plaintext | 00 01 02 88 ee 59 7e 00 |
| Ciphertext | 17 86 a8 03 a0 85 75 01 |

Notice that an octet of value 0 has been appended to the 7 plaintext octets. The value of this appended plaintext octet has no effect on the values of the first 7 ciphertext octets, which are the only ciphertext octets we are interested in. An arbitrary value can be used instead of 0 for the appended plaintext octet.

The Packet PDU, after encryption, looks like this:

| | |
|-----------|-------------------|
| DA | 01 02 03 04 05 06 |
| SA | f1 f2 f3 f4 f5 f6 |
| Type/Len | 17 86 |
| User Data | a8 |
| CRC | 03 a0 85 75 |

B.6.4 40-bit key

The BPKM protocol always generates and distributes 56-bit DES keys. When 40-bit encryption is required, the 56-bit DES key is converted within an implementation to a 40-bit key by masking off (to zero) 16 of the 56 bits of a TEK.

A TEK has 8 octets, each octet containing 7 bits of key and 1 parity bit. Here is the procedure for converting a TEK to a 40-bit key:

- the first two octets of the TEK are set to 0;
- the two most significant bits of the third octet of the TEK are set to 0;
- the remaining five octets of the TEK are unchanged.

For example, if the TEK distributed by the BPKM protocol is:

```
ff ff ff ff ff ff ff ff,
```

then the conversion to 40 bits yields the TEK

```
00 00 3f ff ff ff ff ff.
```

Except for this conversion of the TEK value, the procedure for 40-bit encryption of a Packet PDU is identical to the case of 40-bit encryption.

To illustrate 40-bit encryption, a previous example of Packet PDU is repeated here, with the TEK converted to 40 bits.

Suppose that the Packet PDU, prior to encryption, is as follows:

| | |
|-----------|----------------------------------------|
| DA | 01 02 03 04 05 06 |
| SA | f1 f2 f3 f4 f5 f6 |
| Type/Len | 00 01 |
| User Data | 02 03 04 05 06 07 08 09 0a 0b 0c 0d 0e |
| CRC | 91 d2 d1 9f |

The total number of octets to be encrypted is 19. The first 16 octets are processed using DES-CBC encryption, and the last 3 octets using DES-CFB64 encryption.

The DES-CBC encryption is performed as follows:

| | |
|------------|-------------------------------------------------|
| Mode | CBC |
| Key | 00 00 0f d8 85 2e f5 ab |
| IV | 81 0e 52 8e 1c 5f da 1a |
| Plaintext | 00 01 02 03 04 05 06 07 08 09 0a 0b 0c 0d 0e 91 |
| Ciphertext | 44 c8 4a 41 14 67 56 a2 dc 64 8f b0 dc 1e 1e 86 |

The key is the TEK conveyed in the Key Reply message, converted to a 40-bit key. The IV is as conveyed in the Key Reply message.

The DES-CFB64 encryption is performed as follows:

| | |
|------------|-------------------------|
| Mode | CFB64 |
| Key | 00 00 0f d8 85 2e f5 ab |
| IV | dc 64 8f b0 dc 1e 1e 86 |
| Plaintext | d2 d1 9f 00 00 00 00 00 |
| Ciphertext | f1 42 aa a3 e4 9b eb 29 |

The key is the same as used for the DES-CBC encryption operation. The IV is the last 8 octets of ciphertext generated by the DES-CBC operation.

The Packet PDU, after encryption, looks like this:

| | |
|-----------|----------------------------------------|
| DA | 01 02 03 04 05 06 |
| SA | f1 f2 f3 f4 f5 f6 |
| Type/Len | 44 c8 |
| User Data | 4a 41 14 67 56 a2 dc 64 8f b0 dc 1e 1e |
| CRC | 86 f1 42 aa |

Appendix C: References

- [FIPS-46] US National Bureau of Standards, “Data Encryption Standard”, Federal Information Processing Standard (FIPS) Publication 46, January 1977.
- [FIPS-46-1] US National Bureau of Standards, “Data Encryption Standard”, Federal Processing Standard (FIPS) Publication 46-1, January 1988.
- [FIPS-74] US National Bureau of Standards, “Guidelines for Implementing and Using the Data Encryption Standard”, Federal Information Processing Standard (FIPS) Publication 74, April 1981.
- [FIPS-81] US National Bureau of Standards, “DES Modes of Operation” Federal Information Processing Standard (FIPS) Publication 81, December 1980.
- [IEEE1] IEEE Std 802-1990, IEEE Standards for Local and Metropolitan Area Networks: Overview and Architecture, December 1990.
- [MCNS1] MCNS Data-over-cable Radio Frequency Interface Specifications SP-RFII01-970326 (SP-RFI) March 26, 1997.
- [MCNS3] MCNS Cable Modem Termination System - Network-Side Interface Specification SP-CMTS-NSII01-960702 (CMTS-NSI) July 2, 1996.
- [MCNS4] MCNS Cable Modem to Customer Premises Equipment Interface Specification SP-CMCI01-960702 (CMCI) July 2, 1996.
- [MCNS5] Data Over Cable Service Operations Support System Interface Specification SP-OSSII01-970404 (OSSI) April 3, 1997.
- [MCNS6] MCNS Cable Modem Telephony Return Interface Specification SP-CMTRII01-970804 (CMTR) August 8, 1997.
- [RFC2104] H. Krawczyk, M. Bellare, R. Canetti, “HMAC: Keyed-Hashing for Message Authentication”, RFC 2104, February 1997.
- [RFC2202] P. Cheng, R. Glenn, “Test cases for HMAC-MD5 and HMAC-SHA-1”, RFC2202, September 1997.
- [RSA] RSA Laboratories, “The Public-Key Cryptography Standards”, RSA Data Security, Inc., Redwood City, CA.
- [RSA1] RSA Laboratories, “PKCS #1: RSA Encryption Standard. Version 1.5”, November 1993.
- [RSA2] RSA Laboratories, “Some Examples of the PKCS Standards,” RSA Data Security,

rity, Inc., Redwood City, CA, November 1, 1993.

[SCHNEIER] B. Schneier, *Applied Cryptography*, Second Edition, John Wiley, New York, 1996.

[SHA] NIST, FIPS PUB 180-1: Secure Hash Standard, April 1995.

Appendix D: Revisions

D.1 ECNs included in SP-BPI-I02-990319

| ECN | Date Accepted |
|-------------|---------------|
| bpi-n-98003 | 4/6/98 |
| bpi-n-98004 | 4/6/98 |
| bpi-n-98005 | 4/6/98 |
| bpi-n-98006 | 4/6/98 |
| bpi-n-98007 | 4/6/98 |
| bpi-n-98008 | 4/6/98 |
| bpi-n-98011 | 4/30/98 |
| bpi-n-98015 | 6/18/98 |
| bpi-n-98019 | 8/12/98 |
| bpi-n-98027 | 10/15/98 |
| bpi-n-98028 | 10/15/98 |
| bpi-n-98029 | 12/9/98 |
| bpi-n-98030 | 12/9/98 |
| bpi-n-98042 | 1/28/98 |

