INTERNATIONAL TELECOMMUNICATION UNION

ITU-T

TELECOMMUNICATION
STANDARDIZATION SECTOR
OF ITU

J.191

Implementor's Guide

(April 2003)

SERIES J: CABLE NETWORKS AND TRANSMISSION OF TELEVISION, SOUND PROGRAMME AND OTHER MULTIMEDIA SIGNALS

Miscellaneous

Implementor's Guide for
ITU-T Recommendation J.191

IP feature package to enhance cable modems

Implementor's Guide

During the Study Group 9 meeting (Geneva, 31 March – 4 April 2003) it was agreed to publish the present implementor's guide in view of correcting editorial faults in the current Recommendation J.191.

(Based on Temporary Document 125(revision to COM 9-C 55) presented during the meeting by the Rapporteur for Question 14/9)*

The following editorial corrections to the Implementer's Guide have been made:

1. The format of the IETF references is now consistent.
2. The J.112 references also includes Annex C
3. The use of the term "CableHome" has been changed to "IPCable2Home"in several places.

———

Industry interest in home networking has been very strong during the past year. I have observed that Recommendation J.191 served as the starting point for product development and deployment by the industry. In the process of developing products, a number of defects and opportunities for addressing omissions have emerged, which suggests that changes to J.191 are needed. I have had visibility to the issues that have been discussed within the industry and to the solutions that have been offered. This has led to continuing work on the home networking question during the interim period using the electronic means provided by the ITU. This material has been gathered together to create an Implementer's Guide that describes the changes required to update J.191 to make it of greater value to home networking product development. Since it is possible that the Study Group will want to approve this Implementer's Guide at the April meeting, it is attached to this report.

Although it seems this material is best covered by an Implementer's Guide, I will leave the matter of the form and timing of publication to the Study Group. My primary concern as Rapporteur is that we have correctly captured the technical material; hence I am requesting that the experts of Q14 review the attachment to be sure it properly reflects the current state of home networking development.

Attachment: ITU-T Recommendation J.191 – Implementor's Guide

***Contact:**	James Dahl CableLabs USA	Tel: +1 303 661 3861 Fax: +1 303 661 3810 Email:jdahl@cablelabs.com

ITU-T Recommendation J.191
Implementor's Guide

I. Introduction

In developing equipment per J.191, several problem areas have been identified that need correcting or enhancing in order to produce equipment that will serve the market need. Below are the text changes necessary to resolve the problem issues.

II. Changed text: Deletions are in red strike through; Additions are in blue double underline. Text in green is items that have been moved.

1. Summary:

This Recommendation provides a set of IP based features that may be added to a Cable Modem or incorporated into a stand alone device, that will enable cable operators to provide an additional set of enhanced services to their customers including support for IPCablecom Quality of Service (QoS), enhanced security, additional management and provisioning features, and improved addressing and packet handling.

2. Scope

This Recommendation provides a set of IP based features that may be added to a Cable Modem or incorporated into a stand alone device, that will enable cable operators to provide an additional set of enhanced services to their customers including support for IPCablecom Quality of Service (QoS), enhanced security, additional management and provisioning features, and improved addressing and packet handling. This Recommendation implements the IPCable2Home Domain defined in Recommendation J.190.

3. Add the following normative references:

FIPS 180-1 *Secure Hash Algorithm*, Department of Commerce, April, 1995.

FIPS 186 *Digital Signature Standard*, NIST, 18 May 1994

IETF RFC 1901, *Introduction to Community-based SNMPv2*, January 1996.

IETF RFC 1905 *Protocol Operations for Version 2 of the Simple Network Management Protocol (SNMPv2)*, January 1996.

IETF RFC 1906, *Transport Mappings for Version 2 of the Simple Network Management Protocol (SNMPv2)*, January 1996.

IETF RFC 2315 *PKCS #7: Cryptographic Message Syntax Version 1.5*, March 1998

IETF RFC 2437 *PKCS #1: RSA Cryptography Specifications Version 2.0*, October 1998

IETF RFC 2851 *Textual Conventions for Internet Network Addresses*, June 2000.

3a. One reference needs to be amended to read: ITU-T Recommendation J.112 Annexes B&C

4. Add the following informative references:

ITU-T Recommendation J.190 (06/02) *Architecture of MediaHomeNet that supports cable based services*

IETF RFC 1949, Scalable Multicast Key Distribution, May 1996.

IETF RFC 2979, Behavior of and Requirements for Internet Firewalls, October 2000.

IETF RFC 3235, Network Address Translator (NAT) - Friendly Application Design Guidelines, January 2002.

ICSA, Inc., "Firewall Buyer's Guide", 1998, www.icsalabs.com.

5. Section 5 is the most impacted part of the document. Below is the complete section 5.

5 ~~IP Feature Package Requirements,~~ Reference Architecture~~, and Overview~~

This Recommendation provides a set of IP based features that may be added to a Cable Modem, or implemented in a stand alone device, that will enable cable operators to provide an additional set of enhanced services to their customers. These IP based features reside in a logical element called the Portal Service (PS or just Portal). A ~~Cable Modem~~device that contains these enhanced features is referred to as ~~an IP enhanced Cable Modem (IPCM)~~a Residential Gateway, which is an implementation of ~~a J.190 HA device class. As~~IPCable2Home as described in J.~~190, the HA device class includes both Cable Modem functionality as well as Portal Services functionality~~190. .

Major areas and features are as follows:

Management and ~~provisioning~~Provisioning

- Remote management and configuration of the ~~PS~~Residential Gateway device
- Simple Residential Gateway management proxy for IP based home devices ~~(e.g. a PC)~~
- Hands off provisioning for ~~the PS~~Residential Gateway devices

Addressing and Packet Handling

~~One to one address translation for home devices~~

- One to many address translation for home devices
- One to one address translation for home devices
- Non translated addressing for home devices (for NAT phobic applications)
- HFC traffic protection from in-home device intra-communications
- Home addressing support during HFC outage
- Simple DNS server in the ~~PS~~Residential Gateway

Quality of ~~service~~Service (QoS)

- ~~Transparent~~ Residential Gateway device transparent bridging functionality for IPCablecom QoS messaging from/to~~/from~~ IPCablecom compliant applications~~.~~

Security

- ~~PS~~ Residential Gateway device authentication
- Secure Residential Gateway management messages
- Secure download of configuration and software files

- Secure QoS on the HFC link
- Remote ~~PS~~Residential Gateway firewall management

Communication across the WAN and LAN is IPv4 based, leveraging specific protocols defined throughout the remainder of this document. Compliant devices MUST implement version 4 of the Internet Protocol suite (IPv4).

~~5.1~~ The remainder of this section examines the Reference Architecture from six perspectives:

- Logical view (Section 5.1)
- Functional view (Section 5.2)
- Messaging Interface view (Section 5.3)
- Informational view (Section 5.4)
- Operational view (Section 5.5)
- Physical Interface view (Section 5.6)

5.1 Logical Reference Architecture

As shown in Figure 5-1, this section introduces the logical concepts of the IPCable2Home domain, logical elements, and the Home Access (HA) device class.

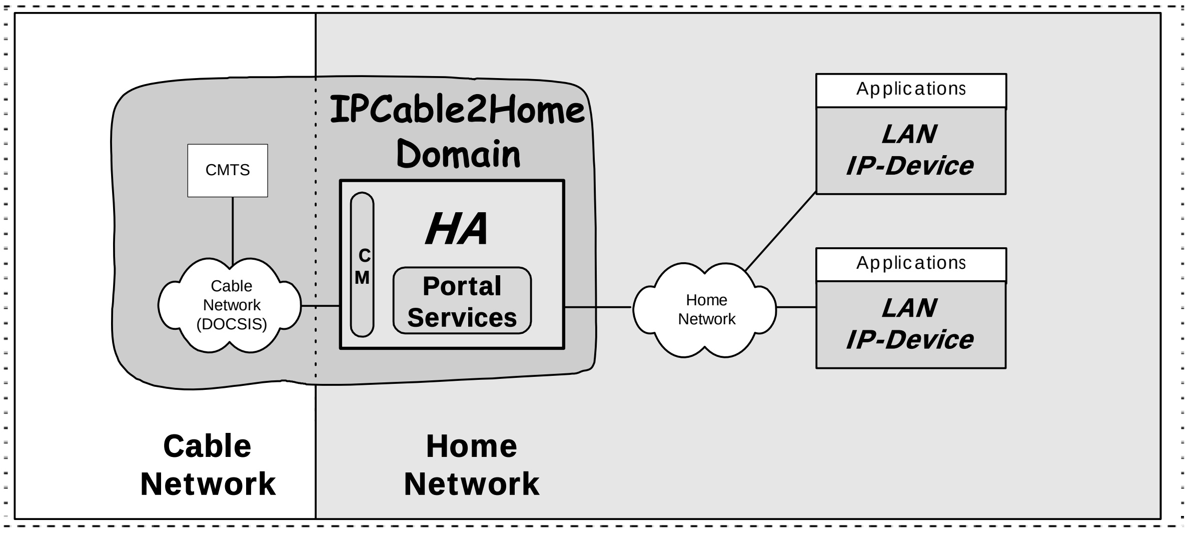

Figure 1. **Key Logical Concepts**

~~5.1.1 Portal Service~~

5.1.1 IPCable2Home Domains

The ~~Portal Service~~IPCable2Home domain represents the set of network elements that are compliant with this Recommendation, and is diagrammatically represented as a shaded region in Figure 1. This region serves as a visual tool to clearly identify those elements within the home network that are compliant. Elements that reside within the IPCable2Home domain (i.e., compliant elements) are directly manageable by operators.

5.1.2 Logical Elements

The architectural framework introduces the concept of logical elements. IPCable2Home logical elements are logically bounded functional entities that can generate and respond to IPCable2Home compliant messages. IPCable2Home logical elements operate at the network protocol layer and above, thus remaining independent of any particular physical network technology. They also include the ability to gather and communicate information as needed to manage and deliver services over IPCable2Home networks. This Recommendation defines a single logical entity known as the Portal Services (PS) element.

5.1.2.1 Portal Services (PS)

A portal is a logical element that provides in-premise and aggregated security, management, provisioning, and addressing services. Three portal service sets of functions are defined. They are the management set of functions, the Quality of Service (QoS) set of functions~~,~~ and the security set of functions. The PS logical element forms the foundation of the logical reference architecture.

5.1.3 Device Classes

The architecture framework also uses the concept of device classes to lend tangible context to the logical elements and combinations of these logical elements. The IPCable2Home concept of device class places no restrictions on physical devices or combinations of logical elements within physical devices. Device classes provide an informative way of depicting collections of logical elements but are not considered definitive or restrictive.

In IPCable2Home, the HA device class represents the physical location of the PS logical element and it enables the network elements within the IPCable2Home domain to interact with LAN IP Devices. The HA device has a single Cable Modem RF-compliant interface, a single PS logical element, and may have zero or more LAN IP interfaces.

This Recommendation also refers to LAN IP Devices. A LAN IP Device is representative of a typical IP device expected to reside on home networks, and is assumed to contain a TCP/IP stack as well as a DHCP client.

5.1.3.1 Embedded PS and Standalone PS

The two primary components of the HA, the Cable Modem (CM) and the Portal Services (PS) element, may physically interface to one another in a variety of ways. It is the nature of this physical interface between the CM and PS that distinguishes the Embedded PS from the Standalone PS.

A standalone CM connects to its subtending CPE using Ethernet over an industry standard physical interface: 10BaseT, USB or PCI bus. A standalone PS, therefore, connects to the CM using one of these interfaces. A PS connecting to a CM via any other interface will be considered an Embedded PS. Given this definition, it is possible that a PS might reside within the same physical enclosure as a CM, yet still be considered a Standalone PS.

The CM and the PS are considered to be separate elements in both the Standalone and Embedded cases, and they respond to unique management addresses. In the Embedded case, the CM and PS may share hardware and software components, but from the management prospective they are separate entities.

Figure 5-2 illustrates both the Standalone and Embedded PS. In both of these cases the combination of a CM and a PS is considered to embody the concept of the HA device.

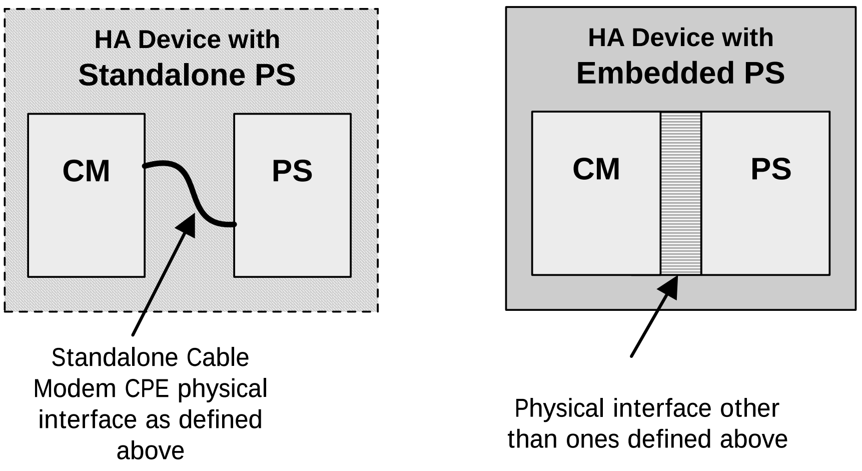
HA Device with
Standalone PS
CM
PS
Standalone Cable Modem CPE physical interface as defined above
HA Device with
Embedded PS
CM
PS
Physical interface other than ones defined above

Figure 1a Standalone and Embedded PS

~~5.1.2~~ 5.1.4 Address Realms

An Address Realm is defined as "a network domain in which the network addresses are uniquely assigned to entities such that datagrams can be routed to them" [RFC 2663]. Within this Recommendation, address realms are categorized as WAN address realms and LAN address realms. (See Figure 2).

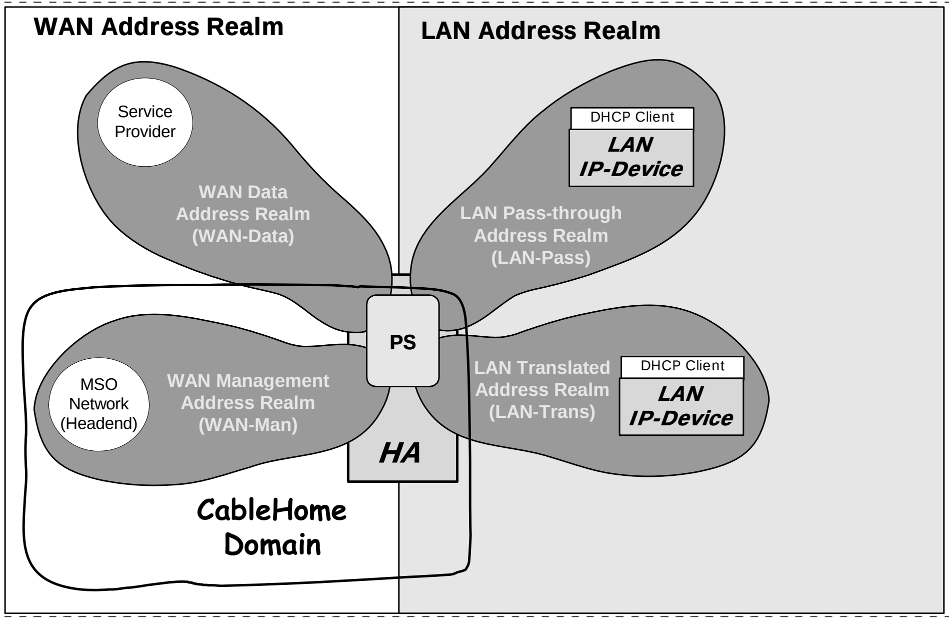
WAN Address Realm
LAN Address Realm
Service Provider
WAN Data Address Realm (WAN-Data)
DHCP Client
LAN IP-Device
LAN Pass-through Address Realm (LAN-Pass)
PS
MSO Network (Headend)
WAN Management Address Realm (WAN-Man)
LAN Translated Address Realm (LAN-Trans)
DHCP Client
LAN IP-Device
HA
CableHome Domain

Figure 2. **Address Realms**

WAN addresses reside in one of two realms: the WAN Management Address Realm (WAN-Man) or the WAN Data Address Realm (WAN-Data). LAN addresses also reside in one of two realms: LAN Passthrough Address Realm (LAN-Pass) or LAN Translated Address Realm (LAN-Trans). The properties of these addressing realms are as follows:

- The WAN Management Address Realm (WAN-Man) is intended to carry network management traffic on the cable network between the network management system and the PS element. Typically, addresses in this realm will reside in private IP address space.

- The WAN Data Address Realm (WAN-Data) is intended to carry subscriber application traffic on the cable network and beyond, such as traffic between LAN IP Devices and Internet hosts. Typically, addresses in this realm will reside in public IP address space.
- The LAN Translated Address Realm (LAN-Trans) is intended to carry subscriber application and management traffic on the home network between LAN IP Devices and the PS element. Typically, addresses in this realm will reside in private IP address space, and can typically be reused across subscribers.
- The LAN Passthrough Address Realm (LAN-Pass) is intended to carry subscriber application traffic, such as traffic between LAN IP Devices and Internet hosts, on the home ~~link~~network, the cable network, and beyond. Typically, addresses in this realm will reside in public IP address space.

On the LAN side, the addresses in the LAN Passthrough Address Realm (LAN-Pass) are directly extracted from the addresses in WAN Data Address Realm. These are used by LAN IP Devices and applications such as IPCablecom services that are intolerant of address translation and require a globally routable IP address. Additionally on the LAN side, LAN IP Devices may use translated addresses from the LAN Translated Address Realm (LAN-Trans).

5.2 IPCable2Home Functional Reference Model

IPCable2Home Functions are services (layer-3 and above) defined for IPCable2Home These Functions are located within the PS, LAN IP Devices, and the Headend. There are IPCable2Home Functions for each of the major specification areas: Provisioning and Management, Security, and Quality of Service. The Functions for Provisioning and Management, Security, and QoS are briefly introduced in the following three subsections.

~~5.2~~ 5.2.1 Management Functions

To support the provisioning and management of IP LAN-Devices within the home, three Management Functions classes are defined:

- Management Server Functions
- Management Client Functions
- Management ~~Service~~ Portal Functions

Several of the Management Server Functions reside within the ~~headend~~Headend (HE). Management Client Functions are typically found within LAN IP Devices. Management ~~Service~~ Portal Functions are located within the PS logical element ~~of the Cable Modem~~ and may include server-like, client-like, and relay-like functionality to aggregate and translate messages between the ~~headend~~Headend and LAN IP Devices. Examples of Management Server, PS~~,~~ and Client functions are introduced in Table 1, Table2, and Table 3 and are illustrated in Figure 3.

Table 1. Management Server Function Description

Management Server Functions	Description
Headend DHCP Server	The DHCP server is a ~~headend~~Headend component that provides address information for the WAN-Man and WAN-Data address realms to the PS
~~Headend DNS Server~~	~~The headend DNS server is a back office component used to map between ASCII domain names and IP addresses~~
Headend Management Messaging server	The ~~headend~~Headend management messaging, download, event notification servers including protocols such as SNMP, SYSLOG, and TFTP

Table 2. Management ~~&~~and Provisioning ~~PS~~Portal Function Description

Management Portal Functions	Description
Cable Address Portal (CAP)	Within the PS, the CAP interconnects the WAN and LAN address realms for data traffic. (See CAT/Passthrough)
Cable Address Translation (CAT)	A sub-function of the CAP, a CAT translates addresses on the WAN-Data side of the CAP to addresses within a single logical subnet on the LAN-Trans side.
Passthrough	A sub-function of the CAP, the Passthrough function bridges packets on the WAN-Data side of the CAP to the LAN-Pass side unchanged.
Cable Management Portal (CMP)	The function that provides an interfaces between the ~~headend~~operator and the PS - database.
Cable DHCP Portal (CDP)	Address information functions (e.g. those transmitted via DHCP) including a server for the LAN realm and a client for the WAN realms
Cable Naming Portal (CNP)	The CNP provides a simple DNS service for LAN IP Devices requiring naming services.
Cable Testing Portal (CTP)	The CTP provides a remote means to initiate pings and loopbacks within the LAN.

Table 3. Management Client Function Description

Management Client Functions	Description
LAN IP Device DHCP Client	The Cable DHCP client function is a in- home component used during the LAN IP Device provisioning process to dynamically request IP addresses and other logical element configuration information.
LAN IP Device Loopback responder	Within LAN IP Device, the loopback responder loops data sourced from the CTP loopback function back to the CTP loopback function.

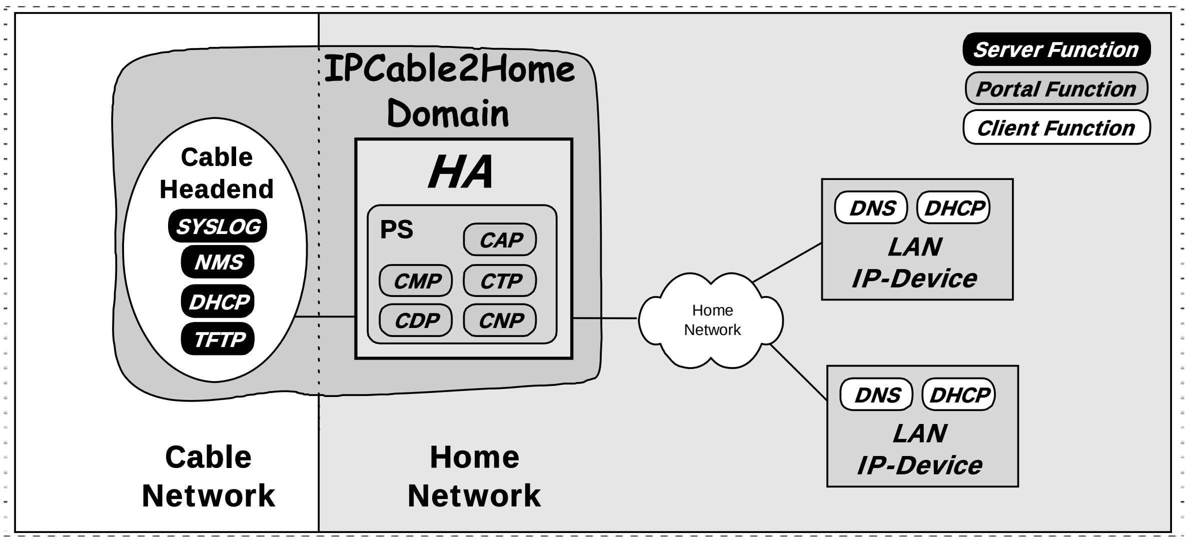

Figure 3. **Management Elements**

~~5.3~~ 5.2.2 Security Functions

~~Security functions are categorized as Security Portal Functions or Server Functions. The relationship between the different security elements and their classification as Server and Portal functions is presented in Figure 4 and described in Table 4 and Table 5.~~

To support the IPCable2Home security requirements, two classes of Security Functions are defined:

- Security Server Functions (Kerberos, Key Distribution Center)
- Security Portal Functions

Security Server Functions reside within the Headend (HE), and the Security Portal Functions consist of client-like functions residing within the PS. Examples of Security Server and Security Portal functions are introduced in Table 4 and Table 5, and are illustrated in Figure 5.

Table 4. Security Portal Function Description

Security Portal Functions	Description
Cable Security Portal (CSP)	The CSP ~~acts as a portal for security material for all of the other security functions within the PS. The CSP communicates on the WAN side with a Security Server (Key Distribution Center, KDC)~~communicates with Headend security servers, and includes functions that provide client side participation in the authentication, key exchange and certificate management processes defined by IPCable2Home. Other security functions include management message security, participation in secure download processes, and remote firewall management.
Firewall (FW)	The Firewall provides ~~protection of~~functionality that protects the home ~~IP environment~~network from malicious attack.

Table 5-Security Server Function Description

Security Server Functions	Description
Headend KDC Servers	The Headend KDC servers provide security services to the CSP and include functions that participate in the ~~head-end provides for authentication services and key distribution for the home. They communicate with the CSP function to establish these services.~~authentication and key exchange processes defined by IPCable2Home

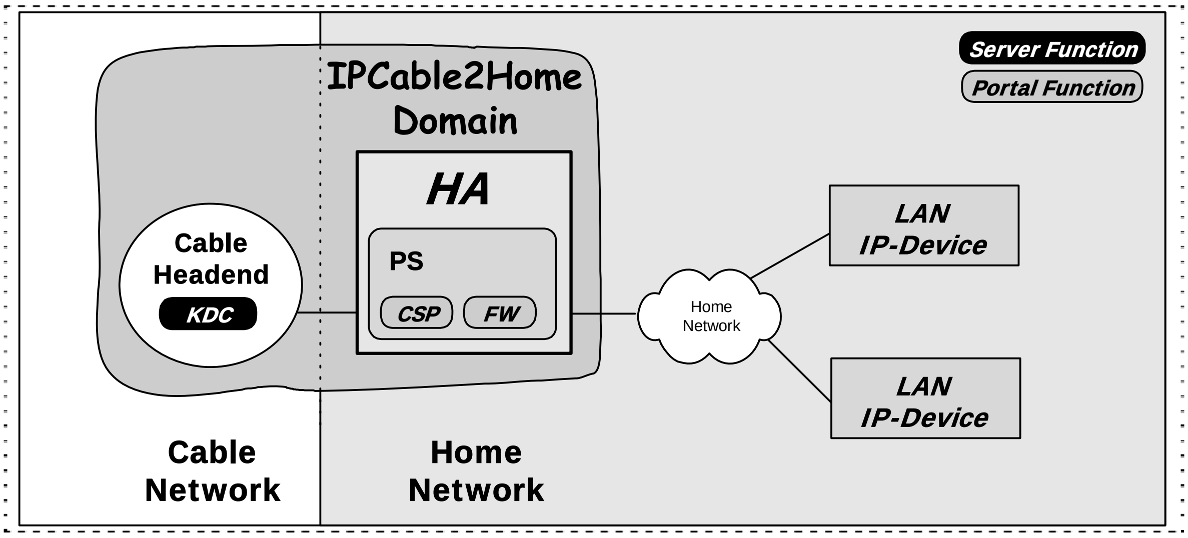

Figure 4 Security Elements

~~5.4~~ 5.2.3 QoS Functions

The QoS architecture is composed of a single PS based functional entity known as the ~~Cable~~IPCable2Home QoS Portal (CQP). The CQP provides transparent bridging for QoS messaging between IPCablecom applications and the IPCablecom QoS infrastructure on the cable network.

~~5.5~~ 5.3 IPCable2Home Messaging Interface Model

The communication between the functions in ~~the~~IPCable2Home network elements and LAN IP Devices occurs on messaging interfaces. The types of messaging interfaces are differentiated by the elements that are involved in the communication. The Messaging interfaces are illustrated in Figure 5.

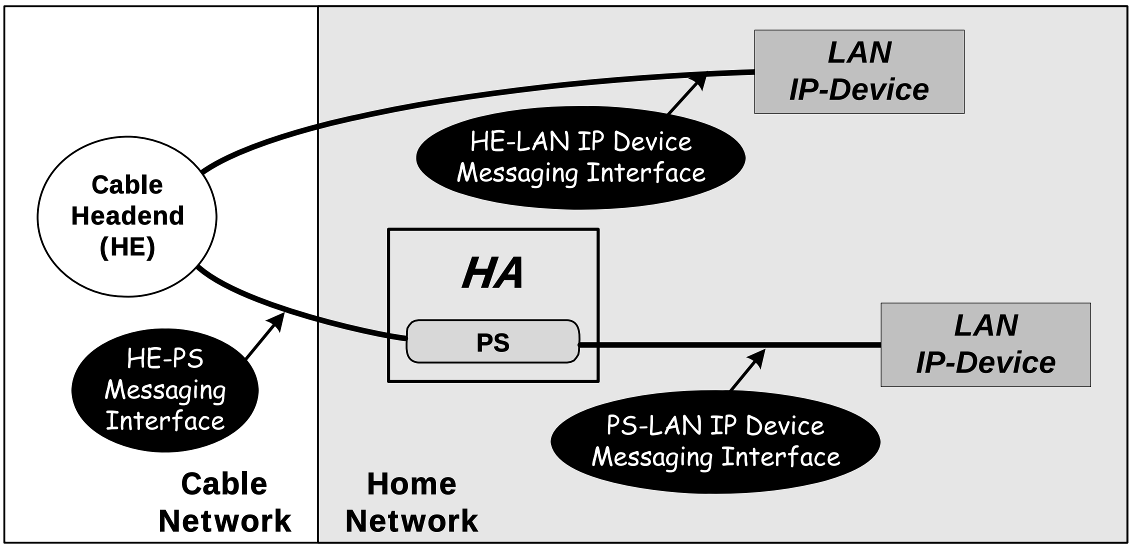
LAN IP-Device
HE-LAN IP Device Messaging Interface
Cable Headend (HE)
HA
PS
LAN IP-Device
HE-PS Messaging Interface
PS-LAN IP Device Messaging Interface
Cable Network
Home Network

Figure 5. Reference Interfaces

The IPCable2Home Messaging interfaces are summarized in Table 6.

Table 6. Valid Interface Paths for Each Functionality

		Interface		
Functionality	**Protocol**	**HE-PS**	**HE-LAN IP Dev**	**PS-LAN IP Dev**
Name service	DNS	Unspecified	Unspecified	~~Unspecified~~This Recommendation
Software Download	TFTP	This Recommendation	Unspecified	Unspecified
Address Acquisition	DHCP	This Recommendation	Unspecified	This Recommendation
Management (single) (Bulk)	SNMP TFTP	This Recommendation This Recommendation	Unspecified	Unspecified
Event Notification	SNMP SYSLOG	This Recommendation This Recommendation	Unspecified	Unspecified
QoS	IPCablecom QoS Protocols	Unspecified	IPCablecom	Unspecified
Security (key distribution)	Kerberos	This Recommendation	Unspecified	Unspecified
Security (authentication)	Kerberos	This Recommendation	Unspecified	Unspecified
Ping	ICMP	This Recommendation	Unspecified	This Recommendation
Loopback/Echo	UDP/TCP	Unspecified	Unspecified	This Recommendation

~~5.6~~ 5.4 IPCable2Home Information Reference Model

The operation of the management model is based upon a store of information maintained in the ~~Portal~~PS by the various ~~Portal~~PS functions (CAP, CDP, CMP, etc.). These functions must have a means of interacting via information exchange, and the ~~Portal~~PS Database is a conceptual entity that represents a store for this information. The ~~Portal~~PS-Database is not an actual specified database per se, but rather a tool to aid in the understanding of the information that is exchanged between the various elements.

Figure 6 shows the relationship between the database and the ~~Portal~~PS functions, Table 7 describes the typical information associated with each of these functions. Figure ~~7~~ shows a detailed example implementation indicating the set of information, the functions that derive the information, and the relationships between the functions and the information.

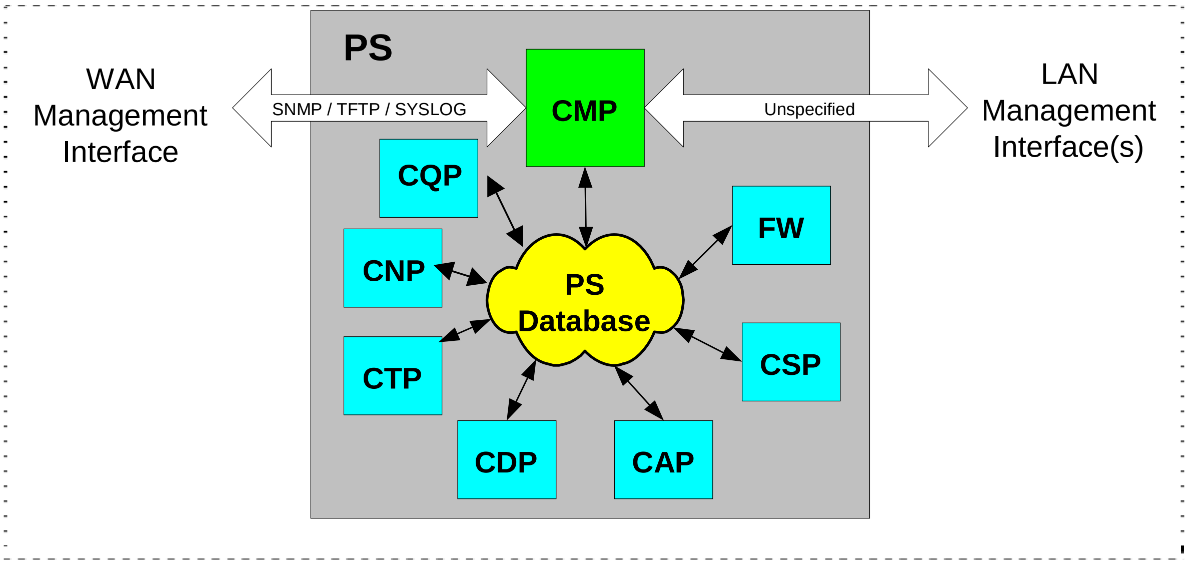

Figure 6. **Portal Function and Database Relationship**

The ~~Portal~~PS Database stores a myriad of data relationships. The CMP provides the WAN management interface (SNMP) to the ~~Portal~~PS database. The functions within the ~~Portal~~PS enter and revise data relationships in the ~~Portal~~PS Database. Additionally, the Functions within the ~~Portal~~PS may retrieve information from the ~~Portal~~PS Database that is maintained by other IPCable2Home Functions within the ~~Portal.~~ PS.

Table 7. Typical ~~Portal~~PS Database information examples

Name	Usage (in general)
CDP Information	Information associated with addresses acquired and allocated via DHCP
CAP information	Information associated with IPCable2Home address translation mappings
CMP information	Information associated with the state of the management functions
CTP information	Information associated with results of LAN test performed by the CMP
CNP information	Information associated with LAN IP Device name resolution
USFS information	Information associated with the Upstream Selective Forwarding Switch function

CSP information	Information associated with authentication, key exchange, etc.
Firewall information	Information associated with the behavior of the Firewall (rule set) and firewall logging.
Event information	Information associated with the local log for all general events, traps, etc.

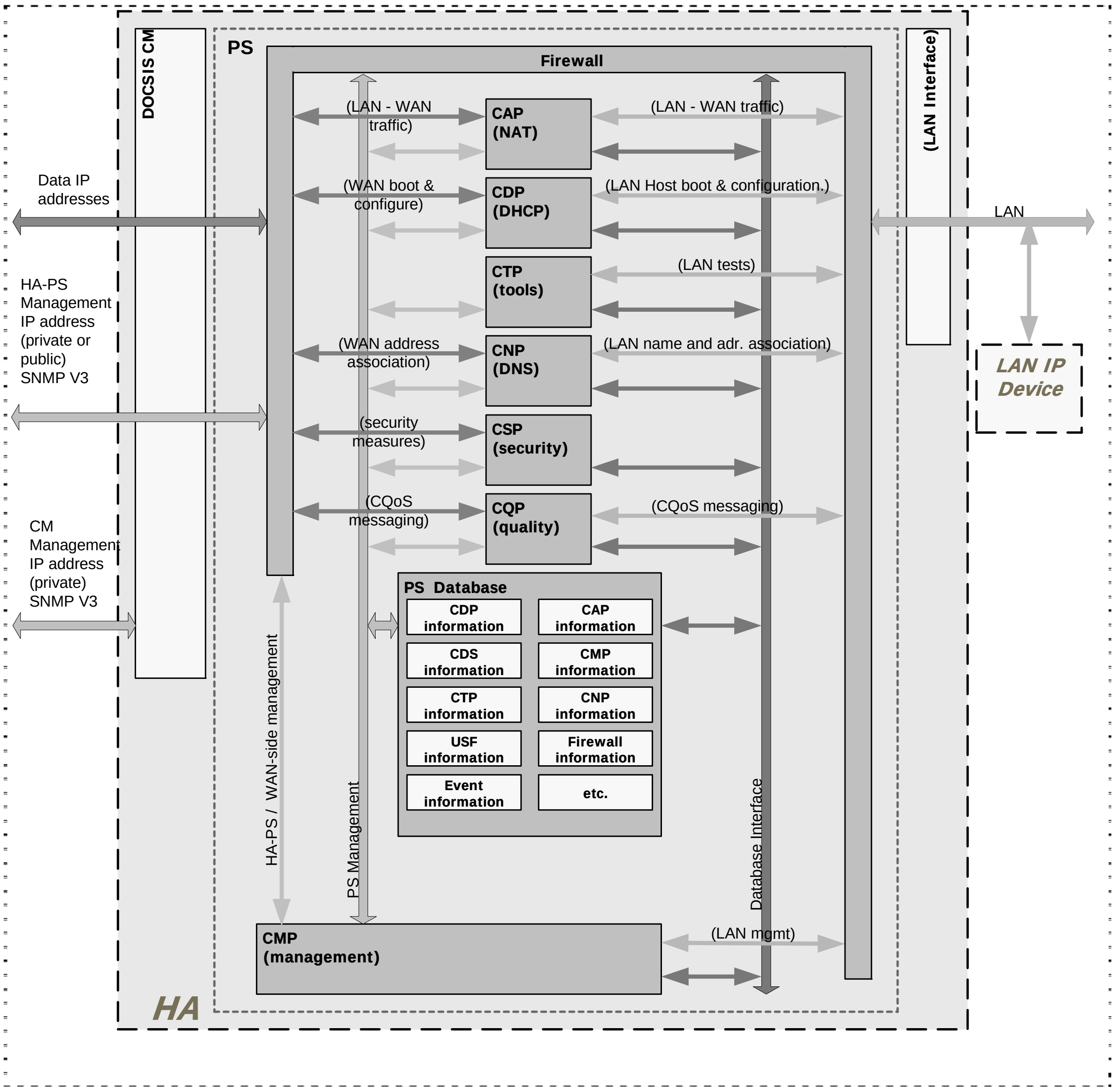

Figure 7. Portal Database Detailed Example Implementation

The ~~Portal~~PS is managed from the WAN via the CMP, and to a large degree this involves access to the information in the ~~Portal~~PS Database. Management is used for initialization and provisioning of the WAN side network elements, and diagnostics or status of the LAN~~ side~~. The diagnostics may rely on the CTP to get better visibility into the current state of the LAN. Connectivity and rudimentary network performance can be measured.

The CNP is the LAN Domain Name System (DNS) manager. All LAN-Trans LAN IP Devices are configured by the CDP to use the CNP as the primary Name Server. The CNP resolves textual host names of LAN IP Devices, returning their corresponding IP addresses and in addition, refers LAN IP Devices to external DNS servers for requests that cannot be answered from local information. ~~The CNP only responds to DNS queries on the LAN-Trans Realm.~~

The CDP contains the address functions to support the DHCP server in the LAN-Trans realm and a DHCP client in the WAN realms.

The CAP creates address translation mappings between the WAN-Data and LAN-Trans address realms. The CAP is also responsible for Upstream Selective Forwarding Switch decisions to preserve HFC upstream channel (WAN) bandwidth from the local LAN only traffic. Finally, the CAP contains the Passthrough function, which bridges traffic between the LAN and WAN address realms.

The CSP is provides ~~PORTAL~~PS authentication capabilities as well as key exchange activities.

The CQP is part of a system that enables IPCablecom Quality of Service (QoS) through the ~~portal~~PS. The CQP, acting as a transparent bridge, forwards IPCablecom compliant QoS messaging between IPCablecom applications and the IPCablecom QoS infrastructure.

~~The firewall is implementation specific, and this Recommendation does not specify the details of firewall implementation.~~

~~5.7~~ 5.5 IPCable2Home Operational Models

~~This enhanced~~The functionality of the Portal Services element is compatible with a variety of cable network infrastructures, which are accommodated by a number of different PS operational modes. These various operating modes enable the PS to function properly within a Cable Modem infrastructure, and within an Extended IPCable2Home infrastructure. The Extended IPCable2Home infrastructure builds upon ~~a cable modem infrastructure~~the Cable Modem infrastructures to enable additional services, and incorporates a number of capabilities that are similar to those within ~~an~~a IPCablecom provisioning system.

For the purpose of configuration, the ~~Portal~~PS may operate within one of two provisioning modes:

- The DHCP Provisioning Mode
- The SNMP Provisioning Mode

If the PS does not receive the information needed to make a provisioning mode determination it will operate with factory default parameters.

When the PS is operating within the DHCP Provisioning Mode, it can operate in one of two Network Management sub-modes:

- NmAccess Mode
- Coexistence Mode

Figure 8 illustrates the various PS operational modes along with the associated triggers for each. See Section 6.3.6.1.1.

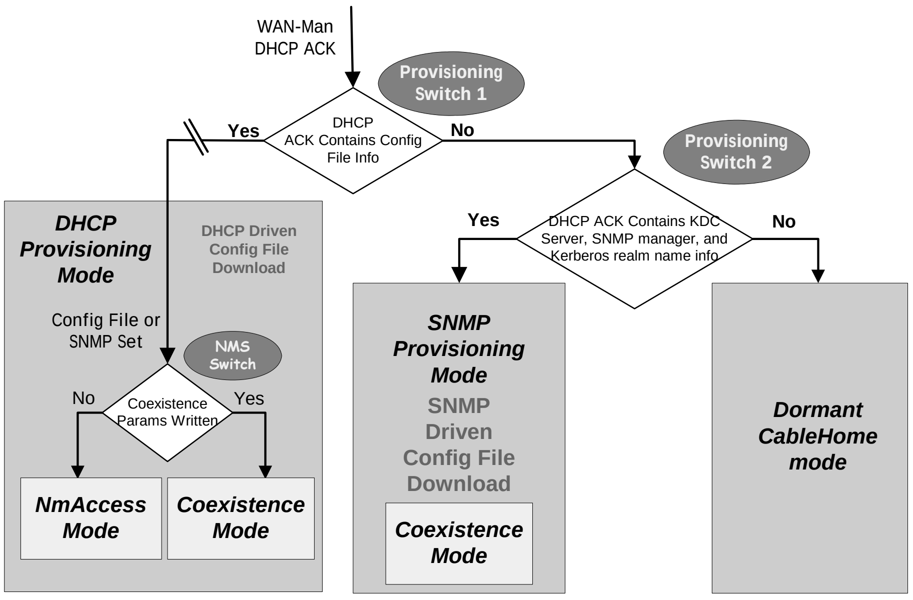

Figure 8. Portal Operational Modes

If ~~Portal~~PS Configuration File information (server location and file name) is provided, and Kerberos server information is not provided, to the ~~Portal~~PS in the DHCP ~~OFFER~~ACK issued by the cable network DHCP server, the ~~Portal~~PS will operate in DHCP Provisioning Mode. When in DHCP Provisioning Mode, the ~~Portal~~PS may operate in one of two Network Management Modes (NmAccess and Coexistence). Within DHCP Provisioning Mode, the ~~Portal~~PS will operate in NmAccess Network Management Mode by default, but can be configured by the NMS to operate in Coexistence Mode.

If ~~Portal~~Kerberos server information is provided, and PS Configuration File information is not provided, to the ~~Portal~~PS in the DHCP ~~OFFER~~ACK issued by the cable network DHCP server, the ~~Portal~~PS will operate in SNMP Provisioning Mode. When operating in the SNMP Provisioning Mode, information and triggers for ~~Portal~~PS Configuration File download are provided by the NMS via SNMP messaging. As opposed to the DHCP Provisioning Mode, the network management behavior does not vary within this mode.

If neither Kerberos server information nor PS Configuration File information is provided to the PS in the DHCP ACK issued by the cable network DHCP server, the PS will default to Dormant IPCable2Home mode, in which it uses factory default configuration parameters for operation.

Table 8 ~~describes the capabilities impacted by each operational mode described above.~~describes the infrastructures within which each PS mode is intended to operate.

Table 8 PS Infrastructures

Mode	Capability Directly Effected	Intended Infrastructure
SNMP Provisioning Mode	Configuration file download.	Extended IPCable2Home Infrastructure
DHCP Provisioning Mode	Configuration file download.	DOCSIS 1.0 and 1.1 Infrastructures
DHCP Provisioning Mode: NmAccess Mode	SNMP version used between NMS and PS	DOCSIS 1.0 Infrastructure (SNMP v1/v2)

DHCP Provisioning Mode: SNMP Coexistence Mode	SNMP version used between NMS and PS	DOCSIS 1.1 and Extended IPCable2Home Infrastructures (SNMP v3)
Dormant IPCable2Home Mode	SNMP manageability from WAN interface	Any cable network infrastructure that does not support IPCable2Home provisioning and management.

~~These various operational modes are meant to accommodate a variety of infrastructures from a back office server perspective, including various SNMP versions, and various types of security servers. More details can be found in sections 13.1 – 13.3.~~

5.6 IPCable2Home Physical Interfaces

There are many types of physical interfaces that may be implemented on a device containing PS functionality. Several are described in the following list:

- WAN Networking Interfaces, which include the Radio Frequency Interface (RFI) as described by J.112 (or J.122) for the Embedded PS case, and other WAN Networking Interfaces, intended for WAN connection, in the Standalone PS case.
- LAN Networking Interfaces for connection to LAN IP Devices.
- Hardware Test Interfaces, such as JTAG and other proprietary approaches, which are part of the silicon and don't always have software controls to turn the interfaces off. These interfaces are hardware state machines that sit passively until their input lines are clocked with data. Though these interfaces can be used to read and write data, they require an intimate knowledge of the chips and the board layout and are therefore difficult to "attack". Hardware test interfaces MAY be present on a device implementing PS functionality. Hardware test interfaces MUST NOT be either labeled or documented for customer use.
- Management Access Interfaces, also called console ports, which are communications paths (usually RS-232, but could be Ethernet, etc.) and debugging software that interact with a user. The software prompts the user for input and accepts commands to read and write data to the PS. If the software for this interface is disabled, the physical communications path is disabled. A PS MUST NOT allow access to PS functions via a Management Access Interface. Access to PS functions MUST only be allowed via interfaces specifically prescribed by this Recommendation, e.g., operator-controlled access via SNMP.
- Read-only Diagnostic Interfaces can be implemented in many ways and are used to provide useful debug, trouble-shooting, and PS status information to users. A PS MAY have Read-only Diagnostic Interfaces.
- Some products might choose to implement higher layer functions (such as customer premise data network functions) that could require configuration by a user. A PS MAY provide the ability to configure non-IPCable2Home functions. Management interface (read/write) access to PS functions MUST NOT be allowed through the mechanism used for configuring non-IPCable2Home functions.

6. Near the end of section 6.2.2, make the following changes:

The CMP and CTP functional elements reside within the PS. The PS logical element may be embedded or stand alone, relative to the cable modem functionality, as described in Section 5.

~~The CM and PS~~In both Embedded PS and Stand-alone PS cases, from the management perspective, the CM and PS are separate and independent management entities, and no data sharing between CM and PS is implied, except for the case of software image download to ~~a PS. The~~an Embedded PS. In the Embedded PS case, the cable modem's docsDevSoftware objects are accessed to set up, initiate, and monitor the download of a single combined software image. Because of this management independence, the CM and PS MUST respond to different and independent management IP addresses. CM MIB Objects are only visible when the manager accesses them through the CM management IP address, and are not visible via the PS management IP address (and vice-versa). The SNMP access rights to the PS and CM entities MUST be set independently.~~ This does not preclude the~~ The use of a single SNMP agent for Embedded PS case is not precluded.

7. In section 6.3.3, make the following changes:

As described in Section ~~5.7,~~5.5, when in SNMP ~~provisioning mode, the PS operates using: 1) SNMPv3 protocol, 2) supports USM and VACM, and 3)~~Provisioning Mode, the PS defaults to operating in SNMPv3 Coexistence Mode with SNMPv1 and SNMPv2 not enabled, and uses Kerberos to distribute keying material. User-based Security Model (USM) [RFC 2574] and View-based Access Control Model (VACM) [RFC 2575] are supported to allow the cable operator to implement management policy for access to IPCable2Home-specified MIBs.

As described in ~~section 5.7,~~Section 5.5, when in DHCP ~~provisioning mode~~Provisioning Mode, the PS ~~can~~defaults to operate in~~ either of the other two network management modes,~~ NmAccess Table mode~~ and~~, but can be configured by the cable operator to operate in SNMPv3 Coexistence ~~mode~~Mode. In NmAccessTable mode, management access is controlled by the NmAccessTable of ~~[~~RFC 2669~~]~~ and the SNMPv1/v2c protocols are supported. ~~In~~If the PS is configured to operate in SNMPv3 Coexistence ~~mode~~Mode, management access is controlled as described in ~~[~~RFC ~~2576],~~2576, the SNMPv1/ v2c/v3 protocols are supported, USM and VACM are ~~possible~~supported, and SNMPv3 keying material is distributed using ~~[~~RFC 2786~~]~~ and TLVs in the PS Configuration File.

If the PS does not receive SNMP Provisioning Mode or DHCP Provisioning mode decision parameters and falls back to Dormant IPCable2Home mode, it disables SNMP access from its WAN interfaces and responds to any SNMPv1 or SNMPv2c message received through any LAN interface.

Table ~~11~~6-3 contains definitions for terms that are specific to the CMP.

~~Table 11.~~ **Table 6-3 Definition of Terms**

Management-control	Read or write access to a set of parameters that control or monitor the behavior of the PS.
PS Database	A set of parameters that controls or monitors the behavior of the PS element readable by the WAN management system. It can be thought of as a repository of information describing the current state of the PS.
User	As defined in SNMP ~~[RC2574 -~~(section 2.1~~]~~ of RFC 2574), a User has a name associated with it, associated security definitions and access to a View.
View	A View is a set of MIB objects and the access rights to those objects. Each View has a name and it is associated with a User ~~[~~(section 2.4 of RFC 2575~~ - 2.4]~~).
Ultimate Authorization	The single authority that establishes, modifies, or deletes User IDs, authentication keys, encryption keys, and access rights to the PS Database. ~~Ultimate Authorization MAY be switched between a User in the cable network NMS and a User in the home but SHOULD NOT be both.~~ This User is entrusted with all security management operations.
Maintenance User	A User that typically performs only read-only operations on the PS database. This is typically used for performance monitoring and accounting.
Administrator User	A User that typically performs both read and write operations on the PS database. These operations are used for Configuration and Fault Management.

8. Near the end of section 6.3.3 make the following insertion:

The NMS management tools use SNMP to access and manage objects in the PS. If the PS is operating in SNMPv3 Coexistence Mode, SNMPv3 provides NMS operator User authentication to the PS, view-based access to the management information base (MIB) objects in the PS, and encryption of management messages if requested.

9. In section 6.3.4 make the following changes:

The ~~CMP MUST provide Management control to the WAN through SNMP v3 [RFC 2571, RFC 2572].The CMP MUST implement ICMP [RFC 0792] and reply to ICMP Echo Requests from the NMS.~~PS MUST implement ICMP Echo and Echo Reply Message types (Type 8 and Type 0) and ICMP Timestamp and Timestamp Reply Message types (Type 13 and Type 14) as described in RFC 792, and reply appropriately to Ping requests received on any interface.

If the PS is operating in DHCP Provisioning Mode (indicated by a value of '1' in cabhPsDevProvMode) the CMP MUST default to using SNMPv1/v2c for management messaging with the NMS and follow rules for NmAccess mode and Coexistence Mode, described in Section 6.3.6.1.

If the PS is operating in SNMP Provisioning Mode (indicated by a value of '2' in cabhPsDevProvMode), the CMP MUST use SNMPv3 for management messaging with the NMS, following rules described in Section 6.3.6.2.

~~The CMP MUST be able to grant Ultimate Authorization to either the LAN Administrator or the Cable WAN Administrator (PS Administrator).~~

~~The~~When the PS is operating in SNMP Coexistence Mode, the default Ultimate Authorization setting MUST be WAN Administrator (PS Administrator).~~ The Ultimate Authorization setting MAY be overwritten via SNMP access or a configuration file.~~

When the PS is operating in Dormant IPCable2Home mode as described in Section 5.5 and in Section 7.2.3.3 and as indicated by a value of ‘3’ in cabhPsDevProvMode, the PS MUST NOT accept or process any SNMP message received through any WAN interface.

When the PS is operating in Dormant IPCable2Home mode as described in Section 5.5 and in Section 7.2.3.3 and as indicated by a value of ‘3’ in cabhPsDevProvMode, the PS MUST accept and process SNMP messages received through any LAN interface according to docsDevNmAccessTable settings (ref.: Section 6.3.6.1) or according to View-based Access Control Model settings (ref.: Section 6.3.6.3).

10. Also in section 6.3.4, make the following deletion:

~~The PS needs to report, through sysDescr fields, all of the information necessary to determine what SW the PS is capable of being upgraded to.~~ If any of the required sysDescr fields are not applicable, the SysDescr MUST report “NONE” as the value. For example, a PS with no BOOTR will report BOOTR: NONE.

11. In section 6.3.6 add the following text:

Section 5.5 introduced two provisioning modes, (DHCP Provisioning Mode and SNMP Provisioning Mode) and two network management modes (NmAccessTable Mode and SNMPv3 Coexistence Mode) that the PS is required to support. Sections 7.2.3.3, 7.3.3.2, and 7.3.3.3 provide additional detail about PS operation in each of the two provisioning modes.

12. In section 6.3.6.1.1 make the following changes:

~~***a)*** Following receipt of DHCP ACK, the PS operating~~Initial operation of the PS configured for DHCP Provisioning Mode can be thought of as having three steps: (1) behavior of the PS after it has been configured for DHCP Provisioning

Mode, but before its network management mode has been configured via the PS Configuration File, (2) determination of the network management mode, and (3) behavior of the PS after its network management mode has been configured. Rules of operation for each of these steps follow:

1. Once the PS has been configured to operate in DHCP Provisioning Mode (indicated by a cabhPsDevProvMode value of '1' (DHCPmode)), but before it has been configured for a network management mode, the PS MUST operate as follows:

   - ~~SNMPv1/v2c read-only Access to all MIB variables, which are required to be in view during SNMPv1/v2c operation, is allowed from the LAN. No access is allowed from the WAN, to prevent unauthorized management access before the PS is configured via the PS Configuration File.~~
   - ~~SNMPv1/v2c packets are accepted which contain any community string.~~
   - All ~~SNMPv3~~SNMP packets are dropped.
   - ~~Access SHOULD be prohibited to any MIB variable that would allow determination of the PS WAN-Man IP address, like the MIB-2 IpAddrTable.~~
   - None of the SNMPv3 MIBs (Community MIB, TARGET-MIB, VACM-MIB, USM-MIB, NOTIFICATION-MIB) are accessible~~, except that they may be set from the PS Configuration File~~ to the SNMP manager in the NMS.
   - None of the elements in the SNMP-USM-DH-OBJECTS-MIB is accessible ~~except that they may be set from the~~to the SNMP manager in the NMS.
   - The PS Configuration File~~.~~ specified in the DHCP OFFER is downloaded and processed.
   - Successful processing of all MIB elements in the PS Configuration File MUST be completed before beginning the calculation of the public values in the USMDHKickstart Table~~.~~

~~b)~~ 2. If a PS is operating in DHCP Provisioning Mode, the content of the PS Configuration File determines the network management mode, as described below:

   - The PS is in SNMPv1/v2c docsDevNmAccess mode if the PS Configuration File contains ONLY docsDevNmAccess Table setting for SNMP access control.
   - If the PS Configuration File does not contain SNMP access control items (docsDevNmAccessTable or snmpCommunityTable or TLV 34.1/34.2 or TLV38), then the PS is in NmAccess mode.
   - If the PS Configuration File contains snmpCommunityTable setting and/or TLV type 34.1 and 34.2 and/or TLV type 38, then the PS is in SNMP Coexistence Mode. In this case, any entries made to the docsDevNmAccessTable are ignored.

~~c)~~ 3. After completion of the provisioning process described in Section 13.2 (indicated by the value 'pass' (1) in cabhPsDevProvState), the PS operates in one of two network management modes. The network management mode is determined by the contents of the PS Configuration File as described above. Rules for PS operation for each of the two network management modes follow:

   NmAccess Mode ~~(using docsDevNmAccess Table)~~ using SNMPv1/v2c

   - ~~Only~~ The PS MUST process SNMPv1/v2c packets ~~are processed~~and drop SNMPv3 packets.
   - ~~SNMPv3 packets are dropped.~~
   - docsDevNmAccessTable controls access and trap destinations as described in ~~[RFC 2669]~~RFC 2669. The PS MUST enforce the management access policy, as defined by the NmAccess Table, for any access to the specified MIB objects, regardless of the interface or access protocol used.
   - None of the SNMPv3 MIBs (Community MIB, TARGET-MIB, VACM-MIB, USM-MIB, NOTIFICATION-MIB) is accessible.

   When the PS is operating in SNMP v1/v2c NmAccess mode it MUST support the capability of sending traps as specified by the following MIB object (proposed MIB extension to the docsDevNmAcess table):

   DocsDevNmAccessTrapVersion OBJECT-TYPE

   SYNTAX INTEGER {

DisableSNMPv2trap(1),

EnableSNMPv2trap(2),

}

MAX-ACCESS read-create

STATUS current

DESCRIPTION

“Specifies the TRAP version that is sent to this NMS. Setting this object to disableSNMPv2trap (1) causes the trap in SNMPv1 format to be sent to particular NMS. Setting this object to EnableSNMPv2trap(2) causes the trap in SNMPv2 format be sent to particular NMS”

DEFVAL { Disable SNMPv2trap }

::={docsDevNmAcessEntry 8}

Coexistence Mode using SNMPv1/v2c/v3

When in SNMPv3 Coexistence Mode, the PS MUST support the “SNMPv3 Initialization” and “DH Key Changes” requirements specified in Section 11.3.3.1.2. These requirements include calculation of USM Diffie-Hellman Kickstart Table public parameters. The following rules for PS operation apply during and after calculation of the public parameters (values) as indicated:

During calculation of USMDHKickstartTable public values:

- The PS MUST NOT allow any SNMP access from the WAN.
- The PS MAY continue to allow access from the LAN with the limited access as configured by USM MIB, community MIB and VACM-MIB.

After calculation of USMDHKickstartTable public values:

- The PS MUST send the cold start or warm start trap to indicate that the PS is now fully SNMPv3 manageable.
- SNMPv1/v2c/v3 Packets are processed as described by ~~[RFC 2571] and [~~RFC 2571, RFC 2572, RFC 2573, RFC 2574, RFC 2575, and RFC ~~2576].~~2576.
- docsDevNmAccessTable is not accessible.
- Access control and trap destinations are determined by the snmpCommunityTable, Notification MIB, Target MIB, VACM-MIB, and USM-MIB. The PS MUST enforce the management access policy, as defined by the VACM View configured by the cable operator, for any access to the specified MIB objects, regardless of the interface or access protocol used.
- Community MIB controls the translation of SNMPv1/v2c packet community string into security name which select entries in the USM MIB. Access control is provided by the VACM MIB.
- USM MIB and VACM MIB controls SNMPv3 packets.
- Trap destinations are specified in the Target MIB and Notification MIB.

In case of failure to complete SNMPv3 initialization for a User (i.e., NMS cannot access the PS via SNMPv3 PDU), the USM User Table for that User MUST be deleted, the PS is in Coexistence Mode, and the PS will allow SNMPv1/v2c access if and only if the community MIB entries (and related entries) are configured.

13. Delete all of section 6.3.6.1.2

14. Change section 6.3.6.2:

If the PS is operating in SNMP Provisioning Mode following DHCP ACK (as indicated by a value '2' (SNMPmode) for cabhPsDevProvMode), it operates in ~~the network management mode~~SNMPv3 Coexistence Mode using SNMPv~~3, USM and VACM, and~~3 by default for exchanging management messages with the NMS, and uses Kerberos for exchanging key material ~~(as described in Section 6.3.3)~~with the KDC, following rules described in this section.

15. Make the following changes to section 6.3.6.2.1

The management controls defined for IPCable2Home are in the CMP function of the PS~~ element~~. Settings, based on management mode, define the access rights that are granted to a ~~command generator~~User for access to the ~~PS~~Portal Services database, through IPCable2Home-specified MIBs, via SNMP from the cable network NMS. A single ~~command generator~~User is defined by the this Recommendation.

Figure 12 illustrates some ~~example~~possible management Views ~~using SNMPv3.~~for the PS. A WAN Administrator View (PS Administrator view) and a WAN Administrator User (PS Administrator user) are defined by IPCable2Home. Other Views and Users, such as the WAN Maintenance View, the LAN Administrator View, or the LAN User View can be established by the Ultimate Authorization (PS Administrator), following rules defined in ~~[~~RFC 2574~~]~~ and ~~[~~RFC ~~2575].~~2575.

16 Make the following changes to section 6.3.6.2.2

SNMP Access Control, per ~~[~~RFC ~~2575],~~2575, will be used ~~for the WAN side Views~~to control access to specified MIB objects, regardless of the interface through which the request arrives. The View-based Access Control Model (VACM) [RFC 2575] defines a set of services that can be used for checking access rights. VACM Groups define the rights to access the CMP.

17. Make the following changes to section 6.3.6.2.3

To protect against replay attacks, a ~~real ~~time of day clock is utilized to provide timestamps for messaging. Management messaging security requirements are specified in Section 11.3.3.

18. Delete section 6.3.6.3

19. Make the following changes to section 6.3.6.4 (the section is also to be renumbered to 6.3.6.3):

To provide controlled access to management information and the creation of distinct management realms for a PS operating in SNMP v3 Coexistence Mode, View- based Access Control Model (VACM) MUST be employed as defined by ~~[~~RFC ~~2575].~~2575.

The WAN Administrator View MUST be implemented in ~~the PS~~a Portal Services element. Default Views other than the WAN Administrator View MUST NOT be available on the PS. Other Views MAY be created by the Ultimate Authorization through the cable network NMS by configuring the VACM MIB.

20. Make the following changes to section 6.3.7

~~With the exception of the SNMP group of MIB 2, USM MIB, and VACM MIB, which are directly accessed by the SNMP Agent in the PS (CMP), AND the DOCSIS Cable Device MIB for the case of software download to an PS, the PS MUST maintain separate instances of PS-specified MIBs from the cable modem. Information accessed from the PS Database through the PS WAN-Man address MUST be separate and distinct from information accessed via the CM management address.~~

In the Embedded PS, the cable modem management entity and PS management entity (CMP) MUST respond to different and independent management IP addresses. Cable Modem and IPCable2Home specify some of the same MIB objects but if a compliant cable modem and a IPCable2Home-compliant PS Element are embedded in the same device, each is required to maintain its own, separate instance of specified MIB objects, accessible through different management IP addresses, with the exception of the SNMP group of MIB 2 and SNMPv2 MIB, which MAY be common to and shared between the cable modem and the Portal Services Element, and MAY be accessible through either the cable modem management IP address or the PS management IP address.

In the Embedded PS, software download of the single image of the combined cable modem software and Portal Services software, is controlled by the cable modem. The docsDevSoftware Group of objects [RFC 2669] MUST NOT be implemented for the Embedded PS, i.e., this group of objects MUST only be accessible through the cable modem management IP address in an Embedded PS.

The docsDevSoftware Group of objects MUST be implemented in a Standalone PS. Modification of the docsDevSoftware objects (as specified in Section 11.3.7) by the cable operator for the purpose of downloading the standalone PS software image MUST result in proper secure software download operation.

In the Embedded PS, cable modem MIB objects MUST only be visible and accessible when the manager access them through the cable modem management IP address, and MUST NOT be visible or accessible via the PS management IP address (PS WAN-Man IP address), with the exception of the SNMP group of MIB 2 and the SNMPv2 MIB which are allowed to be shared between the CM and PS management entities.

In the Embedded PS, IPCable2Home-specified MIB objects MUST only be visible and accessible when the manager accesses them through the PS management IP address (PS WAN-Man IP address), and MUST NOT be visible or accessible via the cable modem management IP address, with the exception of the SNMP group of MIB 2 and the SNMPv2 MIB which are allowed to be shared between the CM and PS management entities.

21. In section 6.3.9 make the following changes.

Configuration settings MUST be processed in the same order that they appear in the PS Configuration File. The CMP MUST be capable of accepting a series of TLV parameters contained in a PS Configuration File. ~~There is no preconceived state of the PS when a PS Configuration File is received. The process of loading and executing a PS Configuration File may interrupt data processing in the PS.~~ The CMP MUST disregard any configuration setting for which no valid database parameter exists.

For SNMP sets in the PS Configuration File, the PS MUST treat all SNMP variable bindings (Varibinds) in the PS Configuration File as if they were received in a single SNMP PDU. If duplicate Varibinds are received in the PS Configuration File, then the PS MUST stop the provisioning process.

The objects defined by TLVs that are passed in the PS Configuration File and are not supported or cannot be written in the particular PS implementation, MUST be ignored. The CMP MUST disregard any unknown TLV.

The size of the PS Configuration File~~, the~~ MUST be updated in the MIB object cabhPsDevProvConfigFileSize. The number of TLVs processed (i.e., the TLVs that are intended to change the PS configuration per their own Value field) and the number of TLVs ignored (i.e., the TLVs intended to change the PS configuration per their own value fields that are not successful) MUST be updated in the MIB objects~~: cabhPsDevProvConfigFileSize,~~ cabhPsDevProvConfigTLVProcessed and cabhPsDevProvConfigTLVRejected, respectively. Configuration parameter Types 255 (End-of-Data Marker), 0 (Pad

Configuration Setting), and Type and Length field pairs that encompass sub-TLVs do not specify values in Value fields of their own and thus are not counted by cabhPsDevProvConfigTLVProcessed and cabhPsDevProvConfigTLVRejected.

PS Configuration File requirements are specified in Section 7.3.

The CMP MUST exchange TFTP messages only through the PS WAN-Man Interface.

The CMP MUST reject any configuration file not received through the PS WAN-Man Interface.

22. In section 6.4.3.1 make the following changes:

This function is used to get a rough measure of the throughput performance across the link between the PS and a LAN IP Device. It sends a burst of packets between the PS and the LAN IP Device under test, and the round trip ~~delay~~ time is measured for the burst. Generally speaking, the NMS operator fills in a few parameters and triggers the function, and results are stored in the PS Database for later retrieval through the CTP MIB.

The Connection Speed function relies on the LAN IP Devices to have a "loop-back function" or "echo-service" embedded. The Internet Assigned Numbers Authority (IANA) has assigned the echo service port 7 for both TCP and UDP [RFC 347]. The default value of the source IP address ~~is always that~~(cabhCtpConnSrcIp) is the same as the value of the PS LAN default gateway (cabhCdpServerRouter). The value of cabhCtpConnSrcIp can be set to any valid PS WAN-Data IP address or to any valid PS LAN Interface IP address. The PS WAN-Man IP address is not used as the source IP address for a CTP tool since when a PS WAN-Man IP address is present but a PS WAN-Data IP address is not, the PS is operating in Passthrough Primary Packet-handling mode and the cable operator can test LAN IP Devices directly from the NMS console if desired. This test feature only works on LAN IP Devices in the LAN Trans address realm~~.~~ that implement the Echo Service function as described in RFC 347.

23. in section 6.4.3.2 make the following changes:

The CTP MUST ignore, and exclude from the cabhCtpPingNumRecv count, any Echo Reply received after cabhCtpPingTimeOut expires.

Section 6.4.4 lists the parameters and responses for the Ping Tool~~. Note that the time for the request reply is not stored, since typical frame propagation time in the home may be faster than standard time units (mS) can accurately measure. For performance metering, the Connection Speed Tool should be used.~~.

24. section 6.4.4 is reworked as follows:

6.4.4.1 Connection Speed Tool

The CTP MUST implement the Connection Speed Tool~~ with parameters listed below, where the angle brackets indicate the CTP MIB object. Numbers in square brackets are the options or the lower and upper bounds of the parameter range, and the number in parenthesis is the default value:~~, AND MUST comply with the default values and value ranges defined for the Connection Speed Tool-specific objects of the Cable CTP MIB.

The CTP MUST transmit the bytes of test data as fast as possible when running the Connection Speed Tool.

The CTP MUST use Port 7 as the Destination Port when running the Connection Speed Tool.

The Connection Speed Tool MUST NOT generate packets out any WAN Interface.

When the NMS triggers the CTP to initiate the Connection Speed Tool by setting cabhConnControl = start(1), the CTP MUST do the following:

- reset the timer
- set cabhCtpConnStatus = running(2)

~~<cabhCtpConnSrcIp> (equal to the value of cabhCdpServerRouter) - the LAN IP address used as the source of the Connection Speed Tool~~

~~<cabhCtpConnDestIp> - the LAN IP address used as the destination of the Connection Speed Tool. NOTE: can be set to any valid IPv4 address, to find LAN IP Devices in the LAN-Trans address realm.~~

~~<cabhCtpConnProto> [UDP (1), TCP (2)] (UDP) - the protocol used for the Connection Speed Tool~~

~~<cabhCtpConnPort> [1 - 65535] (7) - the port used for the Connection Speed Tool. NOTE: IANA reserves port 7 for this use. Other ports may be useful.<cabhCtpConnNumPkts> [1 - 255] (1) - the number of packets to send for the Connection Speed Tool~~

- transmit the number of packets equal to the value of cabhCtpConnNumPkts, each of the size equal to the value of cabhCtpConnPktSize, to the IP address equal to the value of cabhCtpConnDestIp and port number 7, using the protocol specified by cabhCtpConnProto

~~<cabhCtpConnPktSize> [64 - 1518] (64) - the size of the test frames for the Connection Speed Tool in bytes~~

~~<cabhCtpConnTimeOut> [0 - 600000] (600000) - the timeout value, in milliseconds, for the response to the Connection Speed Tool. NOTE: a value of zero indicates no timeout and can be used for TCP only.~~

~~<cabhCtpConnControl> [notRun (1), start (2), abort (3)] - control for the Connection Speed test~~

~~<cabhCtpConnStatus> [running (1), complete (2), aborted (3)] - status of the Connection Speed test~~

~~<cabhCtpConnPktsSent> [1 - 255] - the number of packets sent during the Connection Speed test~~

~~<cabhCtpConnPktsRecv> [0 - 255] - the number of packets received during the Connection Speed test. NOTE: this value allows the operator to determine whether the timeout was achieved (PktsSent > PktsRecv) due to packet loss, assuming the timeout was properly calculated. This pair of parameters was included to support detection of UDP packet loss. Under normal operation, PktsRecv is equal to PktsSent.~~

~~<cabhCtpConnAvgRTT> [0 - 600000] - the resulting average of round trip time for acknowledged packets in milliseconds~~

~~<cabhCtpConnMaxRTT> [0 - 600000] - the resulting maximum of round trip times for acknowledged packets in milliseconds~~

~~<cabhCtpConnMinRTT> [0 - 600000] - the resulting minimum of round trip times for acknowledged packets in milliseconds~~

~~<cabhCtpConnNumIcmpError> [0 - 255] - the number of ICMP errors. NOTE: The value may include net or host "prohibited" or "unreachable". This parameter is null by default, or when no errors occur.~~

~~<cabhCtpConnIcmpError> [0 - 255] - the last ICMP error~~

- initiate the timer with the first bit transmitted
- terminate the timer when the last bit is received back from the target LAN IP Device OR when the value of the timer is equal to the value of cabhCtpConnTimeOut, whichever occurs first
- when the timer is terminated, set cabhCtpConnStatus = complete(3) AND report the appropriate event (refer to Annex B - CTP Events)
- store the value of the timer (in milliseconds) in cabhCtpConnRTT
- if the value of the timer is equal to the value of cabhCtpConnTimeOut before the last bit is received from the target LAN IP Device, report the appropriate event (refer to Annex B - CTP Events)
- calculate the throughput as defined in the requirement below and store the value in cabhCtpConnThroughput

If the Connection Speed Tool is terminated by the NMS setting the object cabhCtpConnControl = abort(2) or for any other reason before the last bit is received from the target LAN IP Device OR before the timer is terminated, the CTP MUST set cabhCtpConnStatus = aborted(4) AND report the appropriate event (refer to Annex B - CTP Events).

When the CTP runs the Connection Speed Tool, it MUST determine the average round-trip throughput between the PS and the LAN IP Device whose address is passed in cabhCtpConnDestIp (the target LAN IP Device) in kilobits per second, round the number to the nearest whole integer, AND store the result in cabhCtpConnThroughput:

The payload of the packets transmitted when the Connection Speed Tool is running SHOULD NOT be all zeroes or all ones.

The CTP MUST reset cabhCtpConnPktsSent, cabhCtpConnPktsRecv, cabhCtpConnRTT and cabhCtpConnThroughput each to a value of 0 when the connection Speed Tool is initiated (i.e., when the value of cabhCtpConnControl is set to start(1)).

Connection Speed Tool RTT is measured at the PS as the time from the first bit of the first sent packet to the last bit of the last received packet. RTT is only valid if the number of received packets is equal to the number of transmitted packets.

The CTP MUST allow the Connection Speed Tool destination IP address (cabhCtpConnDestIp) to be set to any valid IPv4 address of any LAN IP Device accessible through any LAN Interface of the PS running the CTP Connection Speed Tool.

Setting the Connection Speed Tool control object, cabhCtpConnControl, with the value start(1) MUST result in the execution of the Connection Speed Tool.

Setting the Connection Speed Tool control object, cabhCtpConnControl, with the value abort(2) MUST result in the termination of the Connection Speed Tool.

The default value of cabhCtpConnStatus is notRun(1), which indicates that the Connection Speed Tool has never been executed.

The CTP MUST set the value of cabhCtpConnStatus to running(2) if the Tool has been instructed to start, has not been terminated, and if the Connection Speed Timer has not timed out.

The CTP MUST set the value of cabhCtpConnStatus to complete(3) when the last packet sent by the Connection Speed Tool is received by the CTP.

The CTP MUST set the value of cabhCtpConnStatus to aborted(4) if the Connection Speed Tool is terminated after it is initiated, by an SNMP set of the value abort(2) to the object cabhCtpConnControl or if the test is otherwise terminated before the last packet sent by the Connection Speed Tool is received AND before the Connection Speed Tool timer (cabhCtpConnTimeOut) expires.

The CTP MUST set the value of cabhCtpConnStatus to timedOut(5) if the Connection Speed Tool timer (cabhCtpConnTimeOut) expires before the last packet sent by the Connection Speed Tool is received by the CTP.

The CTP MUST NOT use any IP address for the Connection Speed Tool source IP address (cabhCtpConnSrcIp) except a current, valid PS WAN-Data IP address (i.e., an active cabhCdpWanDataAddrIp object value) OR a current, valid PS LAN Interface IP address. If an invalid value is configured for cabhCtpConnSrcIp, the CTP MUST treat the execution of the test as an aborted case and set the Connection Speed Tool status object cabhCtpConnStatus to 'aborted' and report the appropriate event (see Table B-1).

6.4.4.2 Ping Tool

The CTP MUST implement the CTP Ping Tool, AND MUST comply with the ~~parameters listed below, where the angle brackets indicate the CTP MIB object, numbers in brackets are the lower and upper bounds of the parameter range, and the number in parenthesis is the default value:~~default values and value ranges defined for the Ping Tool-specific objects of the Cable CTP MIB.

~~<cabhCtpPingSrcIp> (equal to the value of cabhCdpServerRouter) - the LAN IP address used as the source of the Remote Ping Tool~~

~~<cabhCtpPingDestIp> - the LAN IP address used as the destination of the Remote Ping Tool~~

~~<cabhCtpPingProto> [icmp (1)] (icmp) - the protocol used for the Remote Ping Tool~~
~~<cabhCtpPingNumPkts> [1 - 4] (1) - the number of packets to send to each host for the Remote Ping test~~

~~<cabhCtpPingPktSize> [64 - 1518] (64) - the size of the test frames for the Remote Ping test in bytes~~
~~<cabhCtpPingTimeBetween> [0 - 600000] (1000) - the time between sending one packet and the next during the~~

~~Remote Ping test in milliseconds~~

~~<cabhCtpPingTimeOut> [0 - 600000] (5000) - the time out for response of sending a single ping during the Remote Ping test in milliseconds~~

~~<cabhCtpPingControl> [notRun(1), start (2), abort (3)] - control for the Remote Ping test~~

~~<cabhCtpPingStatus> [running (1), complete (2), aborted (3)] - status of the Remote Ping test~~

~~<cabhCtpPingNumSent> [0 - 254] - the number of pings sent during the Remote Ping test~~

~~<cabhCtpPingNumRecv> [0 - 254] - the number of pings received during the Remote Ping test~~

When the NMS triggers the CTP to initiate the Ping Tool by setting cabhPingControl = start(1), the CTP MUST do the following:

- reset the timeout timer. The timeout value for this timer is the value of cabhCtpPingTimeOut.
- set cabhCtpPingStatus = running(2)
- issue as many Pings (ICMP requests) as specified by the value cabhCtpPingNumPkts, to the IP address defined by the value of cabhCtpPingDestIp, using the value of cabhCtpPingSrcIp as the source address of each request. The size of each test frame issued is the value of cabhCtpPingPktSize.
- if the value of cabhCtpPingNumPkts is greater than 1, wait the amount of time defined by the value of cabhCtpPingTimeBetween between each Ping request issued by the CTP
- initiate the timeout timer with the first bit transmitted
- terminate the timeout timer when the last bit of the last reply (total of cabhCtpPingNumPkts replies) is received back from the target LAN IP Device

If the CTP recevies all Ping replies before the timeout timer expires, the CTP MUST set cabhCtpPingStatus = complete(3) AND report the appropriate event (refer to Annex B - CTP Events).

If the Ping Tool is terminated by the NMS setting the object cabhCtpPingControl = abort(2) or for any other reason before the last bit is received from the target LAN IP Device AND before the timer is terminated, the CTP MUST set cabhCtpPingStatus = aborted(4) AND report the appropriate event (refer to Annex B - CTP Events).

If the timeout timer expires before the last bit is received from the target LAN IP Device, the CTP MUST set cabhCtpPingStatus = timedOut(5) AND report the appropriate event (refer to Annex B - CTP Events).

When the CTP runs the Ping Tool, it MUST determine the average round-trip time between the PS and the LAN IP Device whose address is passed in cabhCtpPingDestIp (the target LAN IP Device), over the number of Ping requests defined by cabhCtpPingNumPkts, AND store the result in cabhCtpPingAvgRTT. When the CTP runs the Ping Tool, it MUST determine the minimum and maximum round-trip times between the PS and the target LAN IP device, for the set of Ping requests defined by cabhCtpPingNumPkts, and store the values in cabhCtpPingMinRTT and cabhCtpPingMaxRTT, respectively.

If an ICMP error occurs during execution of the Ping Tool, the CTP MUST increment the value of cabhCtpPingNumIcmpError AND log the error in cabhCtpPingIcmpError. The last ICMP error that occurs will over-write the previous one written.

The payload of the packets transmitted when the Ping Tool is running SHOULD NOT be all zeroes or all ones

The CTP MUST reset cabhCtpPingNumSent, cabhCtpPingNumRecv, cabhCtpPingAvgRTT, cabhCtpPingMaxRTT, cabhCtpPingMinRTT, cabhCtpPingNumIcmpError and cabhCtpPingIcmpError each to a value of 0 when the Ping Tool is initiated (i.e., when the value of cabhCtpPingControl is set to start(1)).

Ping Tool RTT is measured at the PS as the time from the last bit of each packet transmitted by the CTP Ping Tool, to the time when the last bit of that packet is received.

The CTP MUST allow the Ping Tool destination IP address (cabhCtpPingDestIp) to be set to any valid IPv4 address of any LAN IP Device accessible through any LAN Interface of the PS running the CTP Ping Tool.

The Ping Tool MUST NOT generate packets out any WAN Interface.

The CTP MUST NOT use any IP address for the Ping Tool source IP address (cabhCtpPingSrcIp) except a current, valid PS WAN-Data IP address (i.e., an active cabhCdpWanDataAddrIp object value) OR a current, valid PS LAN Interface IP address. If an invalid value is configured for cabhCtpPingSrcIp, the CTP MUST treat the execution of the test as an aborted case and set the Ping Tool status object cabhCtpPingStatus to "aborted" and report the appropriate event (see Table B-1).

25. The first paragraph of section 6.5.1 is changed as follows:

The PS MUST generate asynchronous events that indicate important events and situations as specified ~~in~~(refer to Annex B). Events can be stored in an internal event LOG, stored in non-volatile memory, reported to other SNMP entities (as TRAP or INFORM SNMP messages), or sent as a SYSLOG event message to ~~a pre-defined SYSLOG server~~the SYSLOG server whose IP address is passed in DHCP Option 7 of the DHCP OFFER received from the Headend DHCP server through the PS WAN-Man Interface.

26. Make the following changes to section 6.5.2

- ~~MAC Address - describes the MAC address of the device~~ PS WAN-Man-MAC address - describes the MAC address of the PS Element used for management of the box
- PS WAN-Data-MAC address - describes the MAC address of the PS Element optionally used for data

27. Add new section 6.5.4

6.5.4 Secure Software Download Event Reporting

Table B-1 in Annex B, Format and Content for Event, SYSLOG and SNMP Trap, describes events associated with Portal Services software upgrades, in three categories: Software Upgrade Initialization (SW UPGRADE INIT), Software Upgrade General Failure, and Software Upgrade Success. These events apply only to the standalone PS, since software upgrade (also referred to as secure software download) for an embedded PS is controlled and managed by the cable modem. Section 11.3.7.1, Software Download Into Embedded or Standalone PS Elements defines requirements for secure software download for the two classes of Portal Services elements. The embedded PS, as defined in Section 5.1.3.1, Embedded PS and Standalone PS, MUST NOT generate events categorized in Table B-1, Defined Events for IPCable2Home as "Software Upgrade Initialization" (SW UPGRADE INIT) events, "Software Upgrade General Failure" (SW UPGRADE GENERAL FAILURE) events, or "Software Upgrade Success" (SW UPGRADE SUCCESS) events.

28. In section 7.2.2 add the following paragraph after the bulleted items:

The Portal Services maintains two hardware addresses, one of which is to be used to acquire an IP address for management purpose, the other could be used for the acquisition of one or more IP address(es) for data. To prevent hardware address spoofing, the PS does not allow either of the two hardware addresses to be modified.

29. In section 7.2.2.1 the bulleted items need enhancing:

The number of addresses supplied by the CDS to LAN IP Devices is controllable by the NMS system. The behavior of the CDS when a cable operator settable limit is exceeded is also configurable via the NMS. Possible CDS actions when the limit is exceeded include (1) assign a LAN-Trans IP address and treat the WAN to LAN CAT interconnection as would normally occur if the limit had not been exceeded and (2) do not assign an address to requesting LAN IP devices. An address threshold setting of 0 indicates the maximum threshold possible for the LAN-Trans IP address pool defined by the pool “start” (cabhCdpLanPoolStart) and “end” (cabhCdpLanPoolEnd) values.

- In the absence of time of day information from the Time of Day (ToD) server, the CDS uses the PS default starting time of 0 (January 1, ~~1900~~1970), updates the Expire Time for any active leases in the LAN-Trans realm to re-synchronize with DHCP clients in LAN IP Devices, and maintains leases based on that starting point until the PS synchronizes with the Time of Day server in the cable network.
- ~~Upon~~ During the PS ~~re-boot or reset~~Boot process, the CDS remains inactive until activated by the PS ~~after successful download of the PS Configuration File or after 5 unsuccessful attempts by the PS to download the PS Configuration File, whichever occurs first. The CDS is thereby prevented from granting DHCP leases in the LAN-Trans realm until there has been a reasonable opportunity for the cable operator to update LAN-Trans lease parameters such as cabhCdpServerLeaseTime, cabhCdpLanPoolStart, and cabhCdpLanPoolEnd.~~ .

- If the PS Primary Packet-handling mode (cabhCapPrimaryMode) has been set to Passthrough AND the PS provisioning process has completed (as indicated by cabhPsDevProvState = pass(1)), then the CDS is disabled.

30. also in section 7.2.2.1, make the following changes:

Throughout this document, the terms ~~**Automatic Allocation,**~~ Dynamic Allocation~~,~~ and Manual Allocation are used as defined in ~~[RFC 2131]. Automatic Allocation of IP addresses within a LAN-Trans address realm will be permanent, and the CDS may reuse Automatic addresses if all available addresses have been allocated.~~RFC 2131. The CDS Provisioned DHCP Options, cabhCdpServer objects in the CDP MIB, are ~~used by the CDS to indicate the DHCP options~~DHCP Options that can be provisioned by the NMS, and are offered by the CDS to LAN IP devices assigned a LAN-Trans address. CDS Provisioned DHCP Options, cabhCdpServer objects,

31. One more change in section 7.2.2.1

3. An indication that the address was allocated either manually (via the CMP) or ~~automatically~~dynamically (via the CDP) (cabhCdpLanAddrConfig)

~~The CDS uses the MAC address to identify LAN IP Devices.~~

The CDS stores information about the identification of a LAN IP Device in the object cabhCdpLanAddrClientID. The first priority for the value to be stored in this object is the Client ID value passed by the LAN IP Device in DHCP Option 61, Client Identifier. If no value is passed in Option 61, the CDS stores the value passed in the *chaddr* field of the DHCP DISCOVER message issued by the LAN IP Device.

32. Make the following changes to section 7.2.2.2

The Vendor Specific Information option (DHCP Option 43) further identifies the type of device and its capabilities. It describes the type of component that is making the request (embedded or standalone, CM or PS), the components that are contained in the device (CM, MTA, PS, etc.), the device serial number, and also allows device specific parameters.

Details of the requirements for supporting DHCP options 60 and 43 are in Table 19 and Table 20. ~~.~~ Details related to other optional and mandatory DHCP options are provided in Table 21.

The WAN-Data IP Address count parameter of the CDP MIB (cabhCdpWanDataIpAddrCount) is the number of IP address leases the CDC is required to attempt to acquire for the WAN side of NAT and NAPT mappings. The default value of cabhCdpWanDataIpAddrCount is zero, which means that, by default, the CDC will acquire only a WAN-Man IP address.

33. Add the following paragraph to the end of section 7.2.2.2.1

The PS is required to have two hardware addresses: one to be used to uniquely identify the logical WAN interface associated with the WAN-Man IP address (WAN-Man hardware address) and the other to be used to uniquely identify the logical WAN interface associated with WAN-Data IP addresses (WAN-Data hardware address).

34. Major overhaul is required starting at section 7.2.2.2.2 and continuing through to the end of section 7.2

7.2.2.2.2 WAN Address Modes

In order to enable compatibility with as many cable operator provisioning systems as possible, the CDC will support the following configurable WAN ~~address modes:~~Address Modes:

WAN Address Mode 0: The PS Element makes use of a single WAN IP Address, acquired via DHCP using the WAN-Man hardware address. The PS Element has one WAN-Man IP Interface and zero WAN-Data IP Interfaces. This Address Mode is only applicable when the PS Primary Packet-handling Mode (cabhCapPrimaryMode) is set to Passthrough (refer to Section 8.3.2). The cable operator's Headend DHCP server typically needs no software modifications to support this Address Mode. In WAN Address Mode 0, the value of cabhCdpWanDataIpAddrCount is zero.

WAN Address Mode 1: The PS Element makes use of a single WAN IP Address, acquired via DHCP using the WAN-Man hardware address. The PS Element has one WAN-Man IP Interface and one WAN-Data IP Interface~~, which share a common MAC Address~~. These two Interfaces share a single, common IP address. This ~~is the factory default configuration of the PS Element~~Address Mode is only applicable when the PS Primary Packet-handling Mode (cabhCapPrimaryMode) is set to NAPT. The cable operator's Headend DHCP server typically needs no software modifications to support this ~~provisioning mode~~Address Mode. In WAN Address Mode 1, the value of cabhCdpWanDataIpAddrCount is zero.

WAN Address Mode 2: The PS Element ~~makes use of two or more different WAN IP Addresses~~acquires a WAN-Man IP address using the unique WAN-Man hardware address, and is subsequently configured by the NMS to request one or more unique WAN-Data IP Address(es). The PS Element ~~could~~will have one WAN-Man and one or more WAN-Data IP Interface(s)~~, which~~. All WAN-Data IP addresses will share a common ~~MAC Address. These~~hardware address that is unique from the WAN-Man hardware address. The two or more Interfaces ~~would~~(one WAN-Man and one or more WAN-Data) each ~~have their~~has its own, unshared IP address. The CDP is configured by the cable operator to operate in WAN Address Mode 2 by writing a nonzero value to cabhCdpWanDataIpAddrCount, via the PS Configuration File or an SNMP set-request. This Address Mode is applicable when the PS Primary Packet-handling Mode (cabhCapPrimaryMode) is set to NAPT or NAT. The cable operator's Headend DHCP server ~~may~~might need software modification to include support ~~assigning~~for Client IDs (DHCP Option 61) so that it can assign multiple IP ~~Addresses to a single MAC Address. In this mode, Headend DHCP server will need to support IP assignment based on Client ID (option 61), as well as MAC Address.~~addresses to the single WAN-Data hardware address.

~~WAN Address Mode 2 is triggered by writing a unique Client ID string into the cabhCdpWanDataAddrClientId entry of the CDP MIBs cabhCdpWanDataAddrTable, for each Wan-Data interface to be used. To support this provisioning mode, the cable operator will need to provide (via NMS, config file, or manual customer entry via a proprietary interface) the PS~~

~~Element with a unique Client ID string for each WAN-Data IP Interface.~~

There are four potential scenarios for WAN-Data IP addresses:

1. The PS is configured to request zero WAN-Data IP addresses. No WAN-Data Client IDs are needed.
2. The PS is configured to request one or more WAN-Data IP addresses and there are no operator-configured cabhCdpWanDataAddrClientId entries in the CDP MIB. The PS is required to auto-generate as many unique WAN-Data Client IDs as the value of cabhCdpWanDataIpAddrCount.
3. The PS is configured to request one or more WAN-Data IP addresses and there are at least as many operator-configured cabhCdpWanDataAddrClientId entries as the value of cabhCdpWanDataIpAddrCount, i.e., the operator has provisioned enough WAN-Data Client ID values. The PS does not auto-generate any Client IDs.
4. The PS is configured to request one or more WAN-Data IP addresses and there are fewer operator-configured cabhCdpWanDataAddrClientId entries than the value of cabhCdpWanDataIpAddrCount, i.e., the operator has provisioned some but not provisioned enough WAN-Data Client ID values. The PS is required to auto-generate enough additional unique WAN-Data Client IDs to bring the total number of unique WAN-Data Client IDs to the value of cabhCdpWanDataIpAddrCount.

If the cable operator desires for the PS to acquire one or more WAN-Data IP addresses, that are distinct from the WAN-Man IP address, the procedure is as follows. For all WAN Address Modes, the PS first requests a WAN-Man IP address using the WAN-Man hardware address. The procedure described below assumes the PS has already acquired a WAN-Man IP address:

1. The cable operator optionally provisions the PS with unique specific Client IDs, by writing values to the cabhCdpWanDataAddrClientId entries of the CDP MIB's cabhCdpWanDataAddrTable, via the PS Configuration File or SNMP set-request message(s).
2. The cable operator configures the CDP to operate in WAN Address Mode 2 by writing cabhCdpWanDataIpAddrCount to a nonzero value through the PS Configuration File or SNMP set-request message.
3. After the CDP has been configured to operate in WAN Address Mode 2 as described in step 2), the PS checks to see if Client ID values have been provisioned by the NMS as described in step 1). If a number of Client ID values greater than or equal to the value of cabhCdpWanDataIpAddrCount have been provisioned, the PS uses these values in DHCP Option 61 when requesting the WAN-Data IP address(es). If Client ID values have not been provisioned, i.e., if the cabhCdpWanDataAddrClientId entries do not exist, or if the number of Client ID values provisioned is less than the value of cabhCdpWanDataIpAddrCount, the PS generates a number of unique Client ID values such that in combination with the provisioned Client IDs, the total number of unique Client IDs equals the value of cabhCdpWanDataIpAddrCount. The PS generates Client ID values by using the WAN-Data hardware address alone for the first requested WAN-Data IP address, and by concatenating the WAN-Data hardware address with a count that is 8 bits in length for the second and all subsequent WAN-Data IP addresses. If no Client IDs have been provisioned by the NMS, the first 8-bit count value is 0x02 (indicating the second requested WAN-Data IP address), the second count value is 0x03, and so on.

   Example for the case when no Client IDs have been provisioned by the NMS:

   Given WAN-Data hardware address 0xCDCDCDCDCDCD

   PS-generated Client ID for the first requested WAN-Data IP address: 0xCDCDCDCDCDCD

   PS-generated Client ID for the second requested WAN-Data IP address: 0xCDCDCDCDCDCD02

   PS-generated Client ID for the third requested WAN-Data IP address: 0xCDCDCDCDCDCD03

   PS-generated Client ID for the nth requested WAN-Data IP address: 0xCDCDCDCDCDCDn (n=<0xFF)

If some Client IDs have been provisioned by the NMS but the number is less than the value of cabhCdpWanDataIpAddrCount, the PS generates additional Client IDs as needed to bring the total number of Client IDs to the value of cabhCdpWanDataIpAddrCount. The PS will generate these additional Client IDs values by appending an 8-bit count value to the WAN-Data hardware address, starting with 0x02, unless that would duplicate a provisioned Client ID. If the Client IDs provisioned by the NMS follow the same format (hardware address with 8-bit count value), the PS is required to use a unique count value so as to not duplicate a provisioned Client ID.

Example for the case when Client IDs have been provisioned by the NMS (three provisioned Client ID values, cabhCdpWanDataIpAddrCount = 5):

Given WAN-Data hardware address 0xCDCDCDCDCDCD

First provisioned Client ID for the first WAN-Data IP address: 0x0A0A0A0A0A1A

Second provisioned Client ID for the second WAN-Data IP address: 0x0A0A0A0A0A2A

Third provisioned Client ID for the third WAN-Data IP address: 0x0A0A0A0A0A3A

First Client ID generated by the PS for the fourth requested WAN-Data IP address: 0xCDCDCDCDCDCD02

Second Client ID generated by the PS for the fifth requested WAN-Data IP address: 0xCDCDCDCDCDCD03

4. The PS adds the Client ID values it generates as cabhCdpWanDataAddrClientId entries to the end of the cabhCdpWanDataAddrTable.

5. The PS (CDC) requests (repeating the DHCP DISCOVER process as needed) as many unique WAN-Data IP addresses as the value of cabhCdpWanDataIpAddrCount specifies, using the WAN-Data hardware address in the chaddr field of the DHCP message and the Client ID value(s) from step 3) in DHCP Option 61, beginning with the first cabhCdpWanDataAddrClientId entry of the cabhCdpWanDataAddrTable. The CDC is not permitted to request more WAN-Data IP addresses than the value of cabhCdpWanDataIpAddrCount, even if the number of provisioned Client IDs is greater than the value of cabhCdpWanDataAddrTable.

7.2.3 Cable DHCP Portal Requirements

7.2.3.1 CDP Requirements~~.~~

In both the Embedded and Standalone configurations, the PS MUST implement two unique WAN hardware addresses: the PS WAN-Man hardware address and the PS WAN-Data hardware address. The numerical value of the PS WAN-Data hardware address MUST follow sequentially the numerical value of the PS WAN-Man hardware address. The PS WAN-Man and PS WAN-Data hardware addresses MUST persist once they are set at the factory. The PS MUST NOT permit the modification of its factory-set PS WAN-Man and PS WAN-Data hardware addresses.

In both the Embedded PS and Standalone PS cases, the PS element MUST have WAN interface hardware addresses that are distinct from the cable modem's hardware address.

7.2.3.2 CDS Requirements

The CDS behavior MUST be in accordance with the Server requirements of RFC 2131 section 4.3.

The CDS MUST support Dynamic and Manual address allocation in accordance with RFC 2131 section 1.

~~CDP Manual~~CDS Manual IP address allocation MUST be supported using CDP ~~Table (~~MIB's cabhCdpLanAddrTable~~)~~ entries created via the NMS system or ~~config~~PS Configuration file.

In support of Dynamic IP address allocation, the CDS MUST be capable of creating, modifying and deleting cabhCdpLanAddrTable entries for devices allocated a LAN-Trans address.

Provisioned CDP LAN Address Management Table (cabhCdpLanAddrTable) entries MUST be retained during a cable outage and MUST persist after a PS power cycle. The CDS MUST be able to provide DHCP addressing services to LAN IP Devices when enabled by the PS, independent of the WAN connectivity state.

~~7.2.3.2 CDS Requirements.~~

~~The CDS behavior MUST be in accordance with the Server requirements of [RFC 2131] section 4.3.~~

~~The CDS MUST support Automatic, Dynamic, and Manual address allocation in accordance with [RFC 2131] section 1.~~

Upon PS reset or re-boot, the CDS MUST NOT exchange DHCP messages with LAN IP Devices until the CDS is activated by the PS~~, following successful download of the PS Configuration File or following 5 successive unsuccessful PS Configuration File download attempts, whichever occurs first.~~.

The PS MUST activate the CDS, i.e., the CDS MUST begin responding to DHCP DISCOVER and DHCP REQUEST messages received through any PS LAN Interface, in any of the following conditions (see also Figure 13-2 IPCable2Home Provisioning Modes):

- When the PS is operating in DHCP provisioning mode, after the CDC has received a PS WAN-Man IP address lease and the PS has received and properly processed a PS configuration file
- When the PS is operating in SNMP provisioning mode, after the CDC has received a PS WAN-Man IP address lease, has authenticated with the Key Distribution Center (KDC) server, and has successfully enrolled with the NMS
- When the first CDC attempt to acquire a PS WAN-Man IP address lease fails
- When the PS is operating in DHCP provisioning mode and the first attempt to download or to process the PS configuration file fails
- When the PS is operating in SNMP provisioning mode and the attempt to authenticate with the KDC server fails
- When the PS is operating in SNMP provisioning mode and is triggered to download a PS configuration file before CDS operation is initiated, and the first attempt to download or to process the PS configuration file fails

The CDS MUST assign a unique, available IP address from the range of addresses ~~and deliver DHCP configuration parameters only to LAN IP Devices receiving an address in the LAN-Trans address realm~~beginning with cabhCdpLanPoolStart and ending with cabhCdpLanPoolEnd, to each LAN-IP Device in the LAN-Trans realm that requests an IP address using DHCP, if the number of IP addresses already assigned by the CDS is less than the value of cabhCdpLanTransThreshold.

~~Automatic addresses allocation with in a LAN-Trans address realm MUST be permanent and the CDS MAY reuse Automatic addresses if all available addresses have been allocated.~~

~~The CDS MUST use the hardware (MAC) address of LAN IP Devices as their client identifier value.~~

~~The CDS MUST support the CDP MIB including all objects in the cabhCdpLanAddrTable, cabhCdpLanPool objects, cabhCdpServer objects, and cabhCdpLanTrans objects.~~

~~The CDS MUST support the DHCP options indicated as mandatory in the CDS Protocol Support column of Table 18.~~

~~The CDS MUST support NMS provisioning of the options indicated as Mandatory in the CDS Mgmt Support column of Table 18.~~

~~The CDS DHCP options indicated as Mandatory in the CDS Cable Outage Retention column of Table 18 MUST be retained during a cable service outage.~~

~~The CDS DHCP options indicated as Mandatory in the CDS Power Outage Persistent column of Table 18 MUST Persist after a CDP power cycle.~~

~~The CDS MUST support offering the default values indicated in the CDS Factory Defaults column of Table 18, if the DHCP option has not been provisioned.~~

~~If the PS Primary Packet-handling mode (cabhCapPrimaryMode) has been set to Passthrough, then the CDS MUST be disabled.~~

~~Table 18. CDS DHCP Options~~

~~Option Number~~	~~Option Function~~	~~CDS Protocol Support (M)andatory or (O)ptional~~	~~CDS Mgmt Support (M)andatory or (O)ptional~~	~~CDS Factory Defaults~~	~~CDS Cable Outage Retention (M)andatory~~	~~CDS PowerOutage Persistent (M)andatory~~	~~MIB Object Name~~
~~0~~	~~Pad~~	~~M~~	~~-~~	~~N/A~~	~~N/A~~	~~N/A~~	~~N/A~~
~~255~~	~~End~~	~~M~~	~~M~~	~~N/A~~	~~N/A~~	~~N/A~~	~~N/A~~
~~1~~	~~Subnet Mask~~	~~M~~	~~M~~	~~255.255.255.0~~	~~M~~	~~M~~	~~cabhCdpServer SubnetMask~~
~~2~~	~~Time Offset~~	~~M~~	~~O~~	~~0~~	~~N/A~~	~~N/A~~	~~cabhCdpServer TimeOffset~~
~~3~~	~~Router Option~~	~~M~~	~~M~~	~~192.168.0.1~~	~~M~~	~~M~~	~~cabhCdpServer Router~~
~~6~~	~~Domain Name Server~~	~~M~~	~~M~~	~~192.168.0.1~~	~~M~~	~~M~~	~~cabhCdpServer DnsAddress~~
~~7~~	~~Log Server~~	~~M~~	~~M~~	~~0.0.0.0~~	~~M~~	~~M~~	~~cabhCdpServer SyslogAddress~~
~~12~~	~~Host Name~~	~~M~~	~~O~~	~~N/A~~	~~N/A~~	~~N/A~~	~~N/A~~
~~15~~	~~Domain Name~~	~~M~~	~~M~~	~~Null String~~	~~M~~	~~M~~	~~cabhCdpServer DomainName~~
~~23~~	~~Default Time-to-live~~	~~M~~	~~M~~	~~255~~	~~M~~	~~M~~	~~cabhCdpServer TTL~~
~~26~~	~~Interface MTU~~	~~M~~	~~M~~	~~1520~~	~~M~~	~~M~~	~~cabhCdpServer InterfaceMTU~~
~~43~~	~~Vendor Specific Information~~	~~M~~	~~M~~	~~Vendor Selected~~	~~M~~	~~M~~	~~cabhCdpServer VendorSpecific~~
~~50~~	~~Requested IP Address~~	~~M~~	~~N/A~~	~~N/A~~	~~N/A~~	~~N/A~~	~~N/A~~
~~51~~	~~IP Address Lease Time~~	~~M~~	~~M~~	~~60~~	~~M~~	~~M~~	~~cabhCdpServer LeaseTime~~
~~54~~	~~Server Identifier~~	~~M~~	~~M~~	~~192.168.0.1~~	~~M~~	~~M~~	~~cabhCdpServer DhcpAddress~~
~~55~~	~~Parameter Request List~~	~~M~~	~~N/A~~	~~N/A~~	~~N/A~~	~~N/A~~	~~N/A~~
~~60~~	~~Vendor Class Identifier~~	~~M~~	~~N/A~~	~~N/A~~	~~N/A~~	~~N/A~~	~~N/A~~
~~61~~	~~Client-identifier~~	~~M~~	~~N/A~~	~~N/A~~	~~N/A~~	~~N/A~~	~~N/A~~

~~In support of Automatic address allocation, the CDS MUST be capable of creating, modifying and deleting CDP Table entries for devices allocated a LAN-Trans address.~~

If the value of cabhCdpLanTransThreshold is 0, the CDS MUST treat the threshold as if it has been assigned the largest value possible for the current LAN-Trans IP address pool size (as defined by the LAN-Trans IP address pool start (cabhCdpLanPoolStart) and end (cabhCdpLanPoolEnd) values).

The CDS MUST maintain the Address Count parameter (cabhCdpLanTransCurCount) indicating the number of active LAN-Trans address ~~assigned~~leases granted to LAN IP devices~~.~~

The Address Count MUST increase each time a lease for a LAN-Trans address is granted to a LAN IP Device and MUST decrease each time a LAN-Trans address is released or a LAN-Trans address lease expires.

The CDS MUST compare the Address Count parameter (cabhCdpLanTransCurCount) to the Address Threshold parameter (cabhCdpLanTransThreshold) after assigning a LAN-Trans address. If the Address Count parameter (cabhCdpLanTransCurCount) exceeds the Address Threshold parameter (cabhCdpLanTransThreshold), a notification MUST be generated as in accordance with the event reporting mechanism defined in Section 6.5 and Annex B. While the Address Count parameter (cabhCdpLanTransCurCount) exceeds the Address Threshold parameter (cabhCdpLanTransThreshold), the CDS MUST be capable of the following threshold exceeded actions for the next DHCP DISCOVER from the LAN: assign a LAN-Trans addresses as normal or do not assign an address.

~~The~~If cabhCdpLanTranCurCount equals or exceeds cabhCdpLanTransThreshold AND a LAN IP Device requests and additional IP address lease, the specific action taken by the CDS MUST be as indicated by the Threshold Exceeded Action (cabhCdpLanTransAction) provisioned parameter.

The CDS MUST assign IP addresses and deliver DHCP configuration parameters listed in Table 18 for which the CDS has a valid value, only to LAN IP Devices receiving an address in the LAN-Trans address realm.

If the cable operator provisions values for a row in the cabhCdpLanAddrTable, the PS (CDS) MUST offer a lease for (i.e., attempt to assign) the provisioned cabhCdpLanAddrIp IP address, to the LAN IP Device whose hardware address corresponds to the provisioned cabhCdpLanAddrClientID, in response to a DHCP DISCOVER received from that LAN IP Device.

When the CDS assigns an active lease for an IP address to a LAN IP Device, the CDP MUST remove that address from the pool of IP addresses available for assignment to LAN IP Devices.

If the CDS receives a lease request from a LAN IP device that it cannot satisfy due to the unavailability of addresses from the IP address pool (defined by cabhCdpLanPoolStart and CabhCdpLanPoolEnd), it must notify the event in accordance to Annex B and the event reporting mechanism defined in Section 6.5.

If a LAN IP Device in the LAN-Trans realm provides a value in DHCP Option 61 (Client ID) in its DHCP DISCOVER, the CDS MUST use this value as its cabhCdpLanAddrClientID entry.

If a LAN IP Device in the LAN-Trans realm provides a value in DHCP Option 61 (Client ID) in its DHCP DISCOVER, the CDS MUST use this value as its cabhCdpLanAddrClientID entry.

Each LAN IP Device client ID (cabhCdpLanAddrClientId) MUST be stored in hexadecimal number format.

The CDS MUST support the IPCable2Home CDP MIB including all objects in the cabhCdpLanAddrTable, cabhCdpLanPool objects, cabhCdpServer objects, and cabhCdpLanTrans objects.

The CDS MUST support the DHCP options indicated as mandatory in the CDS Protocol Support column of

Table 18.

The CDS MUST support NMS provisioning of the options indicated as Mandatory in the CDS Mgmt Support column of Table 18.

The CDS DHCP options indicated as Mandatory in the CDS Cable Outage Retention column of Table 18 MUST be retained during a cable service outage.

The CDS DHCP options indicated as Mandatory in the CDS Power Outage Persistent column of Table 18 MUST Persist after a CDP power cycle.

The CDS MUST support offering the default values indicated in the CDS Factory Defaults column of Table 18, if the DHCP option has not been provisioned.

If the PS Primary Packet-handling mode (cabhCapPrimaryMode) has been set to Passthrough AND the PS provisioning process has completed (as indicated by cabhPsDevProvState = pass(1)), then the CDS MUST be disabled.

The CDS MUST NOT respond to DHCP messages that are received through, or send DHCP messages through, any WAN Interface.

The CDS MUST NOT deliver any DHCP option with null value to any LAN IP Device.

Table 18 CDS DHCP Options

~~7.2.3.3~~

Option Number	Option Function	CDS Protocol Support (M)andatory or (O)ptional	CDS Mgmt Support (M)andatory or (O)ptional	CDS Factory Defaults	CDS Cable Outage Retention (M)andatory	CDS Power Outage Persistent (M)andatory	MIB Object Name
0	Pad	M	-	N/A	N/A	N/A	N/A
255	End	M	M	N/A	N/A	N/A	N/A
1	Subnet Mask	M	M	255.255.255.0	M	M	cabhCdpServer SubnetMask
2	Time Offset	M	O	0	N/A	N/A	cabhCdpServer TimeOffset
3	Router Option	M	M	192.168.0.1	M	M	cabhCdpServer Router
6	Domain Name Server	M	M	192.168.0.1	M	M	cabhCdpServer DnsAddress
7	Log Server	M	M	0.0.0.0	M	M	cabhCdpServer SyslogAddress
12	Host Name	M	O	N/A	N/A	N/A	N/A
15	Domain Name	M	M	Null String	M	M	cabhCdpServer DomainName
23	Default Time-to-live	M	M	255	M	M	cabhCdpServer TTL
26	Interface MTU	M	M	N/A	M	M	cabhCdpServer InterfaceMTU

43	Vendor Specific Information	M	M	Vendor Selected	M	M	cabhCdpServer VendorSpecific
50	Requested IP Address	M	N/A	N/A	N/A	N/A	N/A
51	IP Address Lease Time	M	M	3600 seconds	M	M	cabhCdpServer LeaseTime
54	Server Identifier	M	M	192.168.0.1	M	M	cabhCdpServer DhcpAddress
55	Parameter Request List	M	N/A	N/A	N/A	N/A	N/A
60	Vendor Class Identifier	M	N/A	N/A	N/A	N/A	N/A
61	Client-identifier	M	N/A	N/A	N/A	N/A	N/A

7.2.3.3 CDC Requirements

The CDC behavior MUST be in accordance with the Client requirements of ~~[RFC 2131].~~RFC 2131.

The ~~PS element MUST have a WAN interface hardware address that is distinct from the cable modem~~CDC MUST attempt to acquire a PS WAN-Man IP address during the PS boot process.

The CDC MUST use the PS WAN-Man hardware address in the *chaddr* field AND in DHCP Option 61, in the DHCP DISCOVER and DHCP REQUEST messages, when requesting a WAN-Man IP address from the Headend DHCP server.

If the ~~CDC receives, in the DHCP response~~ ~~[RFC 2131] from the DHCP server in the cable network, a~~ value of cabhCdpWanDataIpAddrCount is zero, the PS MUST use the WAN-Man IP Address for the WAN-Man and WAN-Data Interfaces.

If the value of cabhCdpWanDataIpAddrCount is greater than zero, the PS MUST request the same number of unique WAN-Data IP address(es) from the Headend DHCP server as the value of cabhCdpWanDataIpAddrCount.

The PS (CDC) MUST NOT attempt to acquire more WAN-Data IP addresses than the value of cabhCdpWanDataIpAddrCount.

The CDC MUST use a unique cabhCdpWanDataAddrClientId in DHCP Option 61 for each WAN-Data IP address requested from the Headend DHCP server.

Each WAN Data client ID (cabhCdpWanDataAddrClientId) MUST be stored in hexadecimal number format.

The CDC MUST use the WAN-Data hardware address as the value in the DHCP message *chaddr* field for each WAN-Data IP address requested from the Headend DHCP server.

When the CDC requests WAN-Data IP addresses from the Headend DHCP server, the CDC MUST use cabhCdpWanDataAddrClientId entries for DHCP Option 61 in the order the entries appear in the cabhCdpWanDataAddrTable, beginning with the first entry.

If a nonzero value is configured for cabhCdpWanDataIpAddrCount, and if the number of cabhCdpWanDataAddrClientId entries is less than the value of cabhCdpWanDataIpAddrCount, the PS MUST generate as many unique WAN-Data Client IDs as needed to bring the total number of cabhCdpWanDataAddrClientId entries to the value of cabhCdpWanDataIpAddrCount, and add each generated entry to the end of the cabhCdpWanDataAddrTable.

If the PS generates WAN-Data Client IDs, the first cabhCdpWanDataAddrClientId entry of the cabhCdpWanDataAddrTable MUST be the WAN-Data hardware address.

If the PS generates WAN-Data Client IDs, any cabhCdpWanDataAddrClientId entry generated by the PS other than the first entry of the cabhCdpWanDataAddrTable MUST be the WAN-Data hardware address with an 8-bit count value appended to the end, beginning with 0x02, unless that value already exists as a cabhCdpWanDataAddrClientId entry, in which case the PS MUST generate the Client ID as the WAN-Data hardware address appended with the next available 8-bit count value.

If during the process of acquiring or renewing a lease for the PS WAN-Man IP address the CDC receives, in the DHCP ACK [RFC 2131] from the DHCP server in the cable network, a valid IP address in the 'siaddr' field AND a valid file name in the 'file' field AND does not receive DHCP Option 177 sub-~~ option 51, the PS MUST~~ option 3, sub-option 6, OR sub-option 51 (valid combination 1), the PS MUST set cabhPsDevProvMode to '~~1' (DHCP Mode).~~

~~If the CDC receives, from the DHCP server in the cable network, a valid IP address for DHCP Option 177 sub-option 51 AND~~ 1' (DHCP Mode) and attempt to synchronize time of day with the ToD server as described in Section 7.4.3.

If during the process of acquiring or renewing a lease for the PS WAN-Man IP address the CDC receives a DHCP ACK from the DHCP server in the cable network containing DHCP Option 177 with a valid IP address (SNMP Entity's address) in sub-option 3, a valid Kerberos realm name in sub-option 6, AND a valid IP address (Kerberos server IP address) in sub-option 51, AND does not receive a valid IP address in the 'siaddr' field AND does not receive a valid file name in the 'file' field (valid combination 2), the PS MUST set cabhPsDevProvMode~~ to '2' (SNMP Mode).~~

~~If the CDC receives, in the DHCP message [RFC 2131] from the DHCP server in the cable network, DHCP Option 177 sub-option 51 AND a valid IP address in the 'siaddr' field, OR if the CDC receives DHCP Option 177 sub-option 51 AND a valid file name in the 'file' field, the PS MUST~~ to '2' (SNMP Mode) AND the PS MUST initiate operation of the CDS AND attempt to synchronize time of day with the ToD server and to authenticate with the KDC server as described in Section 11.

If during the process of acquiring or renewing a lease for the PS WAN-Man IP address the CDC receives, in the DHCP ACK [RFC 2131] from the DHCP server in the cable network, any combination of DHCP Option 177 sub-options 3, 6, and 51, 'siaddr' field, and 'file' field other than the two valid combinations described above, the PS has received an invalid DHCP configuration, and the PS MUST log an error in the local log and re-broadcast a ~~DHCP DISCOVER~~ DHCP DISCOVER message (i.e., restart the provisioning sequence in the event of this invalid condition)~~.~~

~~If the CDC does not receive DHCP Option 177 sub-option 51 AND does not receive a valid IP address in the 'siaddr' field AND does not receive a valid file name in the 'file' field, the PS MUST log an error in the local log and re-broadcast a DHCPDISCOVER message (i.e., restart the provisioning sequence in the event of this invalid condition).~~ – repeating the entire DHCP lease acquisition process up to 5 times.

If on its fifth attempt to aquire or renew a lease for the PS WAN-Man IP address the CDC does not receive one of the two valid combinations of DHCP options in a DHCP ACK message, the PS MUST do the following on the assumption that it is connected via a cable modem to a cable data network that does not support IPCable2Home provisioning (Dormant IPCable2Home mode):

- Disable the SNMP agent (CMP) for WAN interface access. Leave the SNMP agent enabled for messages received through the LAN interface (i.e., for SNMP messages addressed to the PS Server Router address).
- Disable the TFTP client
- Disable SYSLOG event reporting
- Accept the offered (CPE) IP address lease and use it as the PS WAN-Data address in the CAP Mapping Table, including assigning the address to cabhCdpWanDataAddrIp and populating the other entries of the CDP WAN-Data Address Table (cabhCdpWanDataAddrTable). The PS will thereby be operating in WAN Address Mode 1 (ref.: Section 7.2.2.2.2 WAN Address Modes)
- Terminate the provisioning timer
- Set the value of cabhPsDevProvMode to dormantCHmode(3)
- Set the value of cabhPsDevProvState to fail(3)
- Enable the CDS
- Enable CAP and USFS functionality
- Enable the CNP
- Enable the Firewall

The DHCP Option 43, sub-option 11 is a device specific parameter. It indicates whether an address is being requested in the PS WAN Management or PS WAN Data realm. Table ~~18~~7-4 indicates ~~the~~ how the values for DHCP Option 43, sub-option 11 MUST be set for these interfaces.

The CDC MUST implement the Vendor Class Identifier Option (DHCP option 60) as specified in Table 20. and Table 21.

Table 19. DHCP Option 43, Sub-option 11 Values

Element Id	Description ~~&~~and Comments
PS WAN-Man = 0x01	Identifies the request for a WAN-Man realm address.
PS WAN-Data = 0x02	Identifies the request for a WAN-Data realm address

~~The~~In the case of an Embedded PS with cable modem, the cable modem and PS element each send separate DHCP requests. Table 20 describes how the CDC MUST set the contents of options 60 and 43 for the PS when the PS element is embedded with a cable modem, and separate PS WAN Management and PS WAN Data addresses are requested.

Table 20. DHCP Options for Embedded PS WAN-Man and WAN-Data Address Requests

DHCP Request Options	Value	Description
~~PS~~Embedded Portal Services DHCP Request for WAN Management Address		
CPE Option 60	"~~PS~~IPCable2Home"	
CPE Option 43 sub-option 1	request sub-option vector	List of sub-options (within option 43) to be returned by server. None defined.
CPE Option 43 sub-option 2	"EPS"	Embedded PS
CPE Option 43 sub-option 3	"ECM:EPS"	List of embedded devices (Embedded CM and embedded PS)
CPE Option 43 sub-option 4	e.g.,"123456"	CM/PS Device serial number
CPE Option 43 sub-option 5	e.g.,"v3.2.1"	CM/PS Hardware Version Number
CPE Option 43 sub-option 6	e.g.,"v1.0.2"	CM/PS Software Version Number

CPE Option 43 sub-option 11	PS WAN-Man (0x01)	Defines that an address is being requested in the PS WAN Management realm
CPE Option 43 sub-option 12	e.g., "ABC Inc. CM-PS123…"	CM/PS System Description from sysDescr
CPE Option 43 sub-option 13	e.g., "CM-PS123-1.0.2…"	CM/PS Firmware Rev from docsDevSwCurrentVers
CPE Option 43 sub-option 14	e.g., "1.2.3…"	Firewall Policy File Version from cabhSecFwPolicyFileCurrentVersion
~~PS~~Embedded Portal Services DHCP Request for WAN-Data Address		
CPE Option 60	"~~PS~~IPCable2Home"	
CPE Option 43 sub-option 1	request sub-option vector	List of sub-options (within option 43) to be returned by server. None defined.
CPE Option 43 sub-option 2	"EPS"	Embedded PS
CPE Option 43 sub-option 3	"ECM:EPS"	List of embedded devices (Embedded CM and embedded PS)
CPE Option 43 sub-option 4	e.g.,"123456"	CM/PS Device serial number
CPE Option 43 sub-option 11	PS WAN-Data (0x02)	Defines that an address is being requested in the PS WAN-Data realm

Table 21 describes to what the CDC MUST set the contents of options 60 and 43, when the PS is a standalone device.

Table 21 DHCP Options for Stand-alone PS WAN-Man and WAN-Data Address Requests

DHCP Request Options	**Value**	**Description**
Stand-alone Portal Services DHCP Request for WAN Management Address		
CPE Option 60	"IPCable2Home"	
CPE Option 43 sub-option 1	request sub-option vector	List of sub-options (within option 43) to be returned by server. None defined.
CPE Option 43 sub-option 2	"SPS"	Stand-alone PS
CPE Option 43 sub-option 3	"SPS"	List of Embedded devices (Standalone PS only)
CPE Option 43 sub-option 4	e.g., "123456"	PS Device serial number
CPE Option 43 sub-option 5	e.g.,"v3.2.1"	PS Hardware Version Number
CPE Option 43 sub-option 6	e.g.,"v1.0.2"	PS Software Version Number
CPE Option 43 sub-option 11	PS WAN-Man (0x01)	Defines that an address is being requested in the PS WAN Management realm
CPE Option 43 sub-option 12	e.g., "ABC Inc. PS123…"	PS System Description from sysDescr

CPE Option 43 sub-option 13	e.g., "PS123-1.0.2..."	PS Firmware Rev from docsDevSwCurrentVers
CPE Option 43 sub-option 14	e.g., "1.2.3..."	Firewall Policy File Version from cabhSecFwPolicyFileCurrentVersion
Standalone I Portal Services DHCP Request for WAN-Data Address		
CPE Option 60	"IPCable2Home"	
CPE Option 43 sub-option 1	request sub-option vector	List of sub-options (within option 43) to be returned by server. None defined.
CPE Option 43 sub-option 2	"SPS"	Stand-alone PS
CPE Option 43 sub-option 3	"SPS"	List of Embedded devices (Stand-alone PS only)
CPE Option 43 sub-option 4	e.g., "123456"	PS Device serial number
CPE Option 43 sub-option 11	PS WAN-Data (0x02)	Defines that an address is being requested in the PS WAN-Data realm

The PS MUST include at least the Hardware version, Boot ROM image version, Vendor name, Software version, and Model number in the sysDescr object (from [RFC 1907]). The value of the docsDevSwCurrentVers MIB object MUST contain the same Software version information as that contained in the Software version information included in the sysDescr object.

The format of the specific information contained in the sysDescr MUST be as follows:

To report	Format of each field
Hardware Version	HW_REV: <Hardware version>
Vendor Name	VENDOR: <Vendor name>
Boot ROM	BOOTR: <Boot ROM Version>
Software Version	SW_REV: <Software version>
Model Number	MODEL: <Model number>

Each type-value pair MUST be separated internally with a colon ":" followed by a blank space and separated externally by a semi-colon ";" followed by a blank space. For instance, a sysDescr of a PS of vendor X, hardware version 5.2, Boot ROM version 1.4, SW version 2.2, and model number X would appear as follows:

any text<<HW_REV: 5.2; VENDOR: X; BOOTR: 1.4; SW_REV 2.2; MODEL: X>>any text

The PS MUST report in the sysDescr at least all of the information necessary in determining what software and firewall policy versions the PS is capable of loading. If any fields are not applicable, the PS MUST report "NONE" as the value. For example, a PS with no BOOTR will report "BOOTR: NONE". When a PS and a CM are embedded in the same device, the sysDescr and docsDevSwCurrentVers objects of the PS MUST report the same values as those of the CM.

The CDC MUST support the DHCP ~~options~~Options indicated as mandatory in the CDC Protocol Support column in Table 21.

Table 21 ~~represents~~ lists the DHCP ~~options~~Options that are mandatory and optional for the CDC to support. DHCP ~~options~~Options listed as mandatory in Table 21 MUST be included in DHCP DISCOVER and DHCP REQUEST messages sent by the CDC to the cable network DHCP server.

Table 21. CDC DHCP Options

Option Number	Option Function	CDC Protocol Support (M)andatory
0	Pad	M
255	End	M
1	Subnet Mask	M
2	Time Offset Option	M
3	Router Option	M
4	Time Server Option	M
6	Domain Name Server	M
7	Log Server (syslog)	M
12	Host Name	M
15	Domain Name	M
23	Default Time-to-live	M
26	Interface MTU	M
43	Vendor Specific Information	M
50	Requested IP Address	M
51	IP Address Lease Time	M
54	Server Identifier	M
55	Parameter Request List	M
60	Vendor Class identifier	M
61	Client-identifier	M
177	Suboption 3 - Service Provider's SNMP Entity Address	M
177	Suboption 6 –Kerberos Realm Name of the Provisioning Realm	M
177	Suboption 51 - Kerberos Server IP address	M

The PS MUST support a Service Provider's SNMP Entity Address (DHCP Option 177 Sub-option 3) configured as an IPv4 address.

Whenever the first PS WAN-Data interface does not have a current DHCP lease, that first PS WAN-Data interface MUST default to the following IP parameters:

~~(This IP address is used for the WAN mapping for the Dynamic NAPT tuple. This address cannot be used for NAT mapping because WAN side of NAT mapping is persistent. It also cannot be used for Passthrough addresses, which are assigned from the service provider's IP address pool.)~~

~~Management IP address: 192.168.100.5~~

“Fallback” WAN-Data IP address: 192.168.100.5

Netmask:	255.255.255.0
Default Gateway:	192.168.100.1

The purpose for the "Fallback" WAN-Data IP address is to enable access to the cable modem's diagnostic IP address (192.168.100.1) from a LAN IP Device. The "Fallback" WAN-Data IP address MUST only be used as the WAN IP address portion of the Dynamic NAT or NAPT tuple of a C-NAT and C-NAPT address mapping, respectively. If the PS is operating in WAN Address Mode 2 and is required to attempt to acquire multiple WAN-Data IP address leases AND the PS is unable to acquire the leases after issuing three DHCP DISCOVER messages (in accordance with DHCP retry procedures specified in Section 7.2.3.3, CDC Requirements), the PS MUST use the "Fallback" WAN-Data IP address as the WAN portion of each Dynamic NAT tuple, until the PS acquires the necessary WAN-Data IP address lease(s) from a DHCP server through a PS WAN interface.

The "Fallback" WAN-Data IP address MUST NOT be used when the PS is configured to operate in Passthrough Primary Packet-handling mode.

The PS MUST NOT use the "Fallback" WAN-Data IP address for any C-NAT or C-NAPT mappings when the PS has a current PS WAN-Man and PS WAN-Data IP address lease. If a DHCP server on the PS WAN interface offers a lease to the PS (CDC) for the IP address 192.168.100.5, i.e., the same address as the "Fallback" WAN-Data IP address, the PS (CDC) MAY accept the lease and use the address as the WAN-Data IP address for a C-NAT or C-NAPT mapping.

Even when using the 192.168.100.5 default WAN-Data IP address, the CDC MUST continue to perform a DHCP DISCOVER every 10 seconds until a valid DHCP lease is granted to that PS WAN-Data interface (or the WAN- Man interface, if the WAN-Man and WAN-data are sharing one IP address).

When a PS is acquiring a WAN-Management IP address for its WAN-Man interface, the CDC MUST always insert its WAN hardware address into the Client ID (DHCP option 61) field in the DHCP Discover message.

When a PS operating in WAN Address Mode 2 (as described in Section 7.2.2.2) is acquiring a WAN-Data IP address for a WAN-Data interface that will use an IP address distinct from the WAN-Man interface, the CDC MUST include the Client Identifier option (cabhCdpWanDataAddrClientId) in the DHCP Discover message. To enable these unique ~~Wan~~WAN-Data Client IDs, the CDC MUST enable the NMS system to create cabhCdpWanDataAddrClientId entries in the cabhCdpWanDataAddrTable.

If a PS is operating in WAN Address Mode 2 (as described in Section 7.2.2.2) the CDC MUST attempt to obtain an IP address, via DHCP, for each unique client ID (cabhCdpWanDataAddrClientId) in the cabhCdpWanDataAddrTable~~.~~, up to the limit defined by cabhCdpWanDataIpAddrCount.

The CDC MUST continue to ~~broadcast its DHCP DISCOVER message (in accordance with [RFC 2131]) until it receives an address and DHCP ACK. The specific timeout for DHCP server access is implementation dependent. However, the CDC MUST NOT broadcast DHCP DISCOVER more than 3 times in any 30 second period. At minimum, the CDC MUST broadcast DHCP DISCOVER at least once per 30 second interval, until it successfully acquires an address.~~retransmit the broadcast DHCP DISCOVER message implementing a randomized exponential backoff algorithm consistent with that described in RFC 2131. The CDC MUST transmit up to 5 DHCP DISCOVER messages (one initial plus 4 retransmission attempts) before resetting the backoff timer value to ZERO and repeating the process.

If the CDC is successful in acquiring the WAN-Man IP address (i.e., receives a DHCP ACK from a DHCP server via the PS WAN-Man Interface) on its first attempt, AND if the PS is operating in DHCP Provisioning Mode, the PS MUST attempt Time of Day time synchronization with the ToD server by issuing a ToD request as described in Section 7.4.3, before attempting to download the PS Configuration File.

If the CDC ~~does not receive a DHCP OFFER after 5 attempts to broadcast a DHCP DISCOVER message, the~~

~~PS MUST~~is unsuccessful in acquiring the WAN-Man IP address (i.e., the DHCP request times out in accordance with RFC 2131) on its first attempt, the PS MUST trigger the CDS (i.e., initiate CDS operation ~~of the CDS~~), so that the CDS can serve DHCP requests from LAN IP Devices in the LAN-Trans realm ~~can be served with IP addresses.~~.

The CDC MUST only respond to DHCP messages that are received through, or send DHCP messages through, a WAN Interface.

When the last remaining WAN-Data DHCP lease expires, the CDC MUST clear all cabhCdpWanDataAddrDnsIp entries from the cabhCdpWanDataAddrServerTable.

Until the cabhPsDevProvState MIB has a value of 'pass' (1) indicating that the provisioning process is complete, the PS MUST block incoming traffic on the WAN interface that is not in response to a LAN-to-WAN request from the PS element itself or a LAN IP device. This will help protect against potential hacker attacks during the provisioning process when the PS firewall is disabled.

35. One small change near the end of section 7.3.2

It is significant to note that a PS operating in SNMP Provisioning Mode does not need a PS Configuration File loaded before it can operate. It is expected that a PS operating in SNMP Provisioning Mode will initialize itself to a known state and a PS could run for a lifetime without having a PS Configuration File loaded. However, a PS will accept and process a PS Configuration File when one is provided.

36. Add the following text to section 7.3.3

A PS operating in DHCP Provisioning Mode MUST download and process a PS Configuration File.

A PS operating in SNMP Provisioning Mode MUST be capable of operating without a PS Configuration File, but MUST be capable of downloading and processing a PS Configuration File if triggered as described in Section 7.3.3.2.

MIB object settings passed in the PS Configuration File take precedence over and MUST over-write existing MIB object settings.

37. In section 7.3.3.1 make the following changes

The configuration settings MUST follow each other directly in the file, which is a stream of octets (no record markers). The ~~file length MUST be~~PS MUST be capable or properly receiving and processing a configuration file that is padded to an integral number of 32-bit words, AND be able to properly receive and process a configuration file that is not padded to an integral number of 32-bit words. See Section 7.3.3.1.1 for a definition of the pad. Configuration settings are divided into three types:

AND

Each~~ PS~~ Portal Services element MUST support and a PS Configuration File MAY include configuration parameter Types 0, 4, 9, 10, 17, 21, 28, 32, 33, 34, 38 and 255, which are described in this section.

AND

The Length value for each Type described in Sections ~~7.3.3.1.1 – 7.3.3.1.10~~7.3.3.1.1, 7.3.3.1.2, 7.3.3.1.3, 7.3.3.1.4, 7.3.3.1.5, 7.3.3.1.6, 7.3.3.1.7, and 7.3.3.1.8 is the actual length in octets of the Value field.

38. Delete sections 7.3.3.1.2 and 7.3.3.1.5. Renumber remaining sections.

39. Section 7.3.3.1.2 (renumbered) needs minor modification:

The filename of the software upgrade file for the ~~PS~~IPCable2Home device. The filename is a fully qualified directory- path name. The file is expected to reside on a TFTP server identified in a configuration setting option.

Type	**Length**	**Value**
9	Variable~~**~~	filename

~~** Length MUST NOT cause resulting MAC management message to exceed the maximum allowed size.CA-Certificate.~~

40. In section 7.3.3.1.5 (renumbered) delete the note:

~~** Length MUST NOT cause resulting MAC management message to exceed the maximum allowed size.~~

41. Delete the following from section 7.3.3.1.6 (renumbered)

~~If the length of the M-CVC exceeds 65,535 bytes, the M-CVC MUST be fragmented into two or more successive Type 32 elements. Each fragment, except the last, MUST be 65,535 bytes in length. The PS reconstructs the M-CVC by concatenating the contents (Value of the TLV) of successive Type 32 elements in the order in which they appear in the configuration file. For example, the first byte following the length field of the second Type 32 element is treated as if it immediately follows the last byte of the first Type 32 element.~~

42. Delete the following from section 7.3.3.1.7 (renumbered)

~~If the length of the C-CVC exceeds 65,535 bytes, the C-CVC MUST be fragmented into two or more successive Type 33 elements. Each fragment, except the last, MUST be 65,535 bytes in length. The PS reconstructs the CCVC by concatenating the contents (Value of the TLV) of successive Type 33 elements in the order in which they appear in the configuration file. For example, the first byte following the length field of the second Type 33 element is treated as if it immediately follows the last byte of the first Type 33 element.~~

43. Add the following section:

7.3.3.1.11 PS Message Integrity Check (PS MIC)

Type	Length	Value
53	20	A 160-bit (20 octet) SHA hash

This parameter contains a hash (PS MIC) calculated by a Secure Hash Algorithm (SHA-1) defined in FIPS 180-1. This TLV is only used in the configuration file immediately before the end of data marker.

44. Section 7.3.3.2 experiences significant regonfiguration and additions:

PS Configuration File Download Trigger for DHCP Provisioning Mode:

If the PS receives the TFTP server address in the 'siaddr' field and the PS Configuration File name in the 'file' field of the DHCP OFFER, the PS MUST combine the TFTP server address and PS Configuration File name to form a URL-encoded value and write that value into cabhPsDevProvConfigFile~~. The PS Configuration hash appended to the PS Configuration File name MUST NOT be included in the URL- encoded value.~~.

Download of the PS Configuration File, by a PS operating in DHCP Provisioning Mode, is triggered by the presence of the PS Configuration File location (TFTP server IP address) and name in the DHCP message issued to the PS (CDC) by the DHCP server in the cable network. Refer to Section 7.2.3.3.

If the PS is operating in DHCP Provisioning Mode (as indicated by the value of cabhPsDevProvMode), after the PS (CDC) receives a DHCPACK from the DHCP server in the cable network, the PS MUST issue a TFTP Get request to the server identified in the DHCP message 'siaddr' field to download the file identified in the DHCP message 'file' field.

The PS MUST issue TFTP Get request messages through the PS WAN-Man Interface only.

Modification of cabhPsDevProvConfigFile MUST NOT trigger a PS operating in DHCP Provisioning Mode to download a configuration file. A PS operating in DHCP Provisioning Mode MUST treat cabhPsDevProvConfigFile as a read-only object.

The PS MUST reject any PS Configuration File that is received through any Interface except the PS WAN-Man Interface.

PS Configuration File Download Trigger for SNMP Provisioning Mode:

If the PS is operating in SNMP Provisioning Mode (as indicated by the value of cabhPsDevProvMode), PS Configuration File download MUST NOT occur before completion of the SNMP v3 authentication process (refer to Section ~~11~~11, Security for details about the SNMP authentication process).

If the PS is operating in SNMP Provisioning Mode (as indicated by the value of cabhPsdevProvMode), the PS element MUST NOT initiate a PS Configuration File download if a valid value for cabhPsDevProvConfigHash (PSDev MIB) has not been provisioned by the NMS.

~~If~~Once the PS~~ is~~ operating in SNMP Provisioning Mode (as indicated by the value of cabhPsDevProvMode) ~~AND the cabhPsDevProvConfigHash object from the PSDev MIB has a valid value, the PS Configuration File download MUST be triggered when an SNMP set- request message, addressed to the PS WAN-Man interface, contains a valid value for the cabhPsDevProvConfigFile PSDev MIB object.~~ issues a TFTP request to download a PS Configuration file (subject to conditions described in other requirements, below), the PS MUST complete the download phase. When the PS

(CMP) has successfully downloaded the requested PS Configuration File, it MUST process the file before issuing a TFTP request for another PS Configuration File.

A signaling mechanism is necessary to inform the management entity that the PS is currently processing a configuration file. The PS Dev MIB object cabhPsDevProvConfigFileStatus is defined to serve as this signaling mechanism.

If a PS (CMP) is not currently requesting, downloading, or processing a configuration file, it MUST set cabhPsDevProvConfigFileStatus = idle(1). When the PS (CMP) has issued a TFTP request for a configuration file specified in cabhPsDevProvConfigFile, it MUST set cabhPsDevProvConfigFileStatus = busy(2). When the PS (CMP) completes the processing of the PS Configuration File, the PS MUST set cabhPsDevProvConfigFileStatus = idle(1).

The PS (CMP) MUST attempt to download and process the configuration file whose name and address are specified in cabhPsDevProvConfigFile when it receives an SNMP set request message for the cabhPsDevProvConfigFile object, if the following conditions are true:

- the PS is operating in SNMP Provisioning Mode
- the cabhPsDevProvConfigHash object has a valid value, AND
- cabhPsDevProvConfigFileStatus = idle(1)

The format of cabhPsDevProvConfigFile MUST be a URL- encoded TFTP server IP address and configuration file name.

If the PS (CMP) operating in SNMP Provisioning Mode receives an SNMP set request from the NMS to update the value of cabhPsDevProvConfigFile AND cabhPsDevProvConfigFileStatus = busy(2) OR if the cabhPsDevProvConfigHash object does not have a valid value, then the PS MUST reject the set request.

Post-trigger Operation:

Once triggered, the PS MUST use an ~~[~~RFC 1350~~]~~ compliant TFTP client to download the PS Configuration Files.

If the PS Configuration File is properly authenticated, when the TFTP download of the PS Configuration File is complete, the PS MUST process the TLVs contained within the file. Refer to Section 6.3.9 for a description of how the CMP processes the Configuration File.

If the PS is triggered to download a PS Configuration File AND the TFTP Get request times out, i.e., if the PS is not successful in downloading the PS Configuration File, on the first attempt, the PS MUST initiate operation of the CDS AND report the appropriate event (refer to Annex B - "TFTP Errors Before Provisioning Complete").

If the PS is operating in DHCP Provisioning Mode and fails to successfully download the PS Configuration File OR the CMP fails to successfully process all TLVs, the PS MUST report the appropriate event (refer to Annex B - "TFTP Errors Before Provisioning Complete"), AND re-start the initialization process beginning with the CDC issuing a DHCP DISCOVER message.

If the PS is operating in SNMP Provisioning Mode AND is triggered to download a PS Configuration File AND fails to successfully download the PS Configuration File OR if the CMP fails to successfully process all TLVs, the PS MUST report the appropriate event (refer to Annex B - "TFTP Errors Before Provisioning Complete"), wait a period of time defined by the adaptive timeout algorithm described in the requirement below, retrieve the PS Configuration File information from cabhPsDevProvConfigFile, AND re-issue the TFTP request using the PS Configuration file name and address retrieved.

Refer to Section 6.3.9 for additional PS Configuration File processing requirements.

45. Section 7.3.3.3 also experiences considerable change;

This section defines the procedure for authenticating the PS Configuration File.

~~A hash calculation is used to authenticate the PS Configuration File. The NMS calculates the hash of the PS Configuration File then sends the resulting hash value to the PS element. The identity of the NMS that generated the PS Configuration File is authenticated by comparing the hash of the PS Configuration File that was generated by the NMS and transported to the PS element against the hash (calculated by the PS) on the PS Configuration File downloaded from the TFTP server. The identity of the PS element requesting the file is not required.~~

The ~~security~~ algorithm used to ~~authenticate~~check the PS Configuration File Hash depends upon the provisioning mode of the PS element (see Section ~~5.7~~5.5). There are two types of provisioning modes, DHCP Provisioning Mode and SNMP Provisioning mode. The following sections describe the security algorithms and requirements needed to ~~authenticate~~check the PS Configuration File Hash based on the provisioning mode of the PS element. The PS element MUST support both security algorithms specified in ~~Section~~Sections 7.3.3.3.1 and 7.3.3.3.2.

__7.3.3.3.1 PS Configuration File Authentication Algorithm for DHCP Provisioning Mode__

The procedure for ~~authentication~~checking of the PS Configuration File hash by the PS element in DHCP Provisioning Mode follows:

1. When the ~~NMS~~Config File Generator of the Provisioning System creates a new PS Configuration File or modifies an existing file, the ~~NMS~~Config File Generator will create a SHA- 1 hash of the ~~entire content~~contents of the PS Configuration File, taken as a byte string. The end of data marker and any padding that follow it are not included in the hash calculation.

~~2. The NMS appends the hash value to the PS Configuration File name that is sent to the PS element in the DHCP Offer (see Section 7.2.3.3 and Section 13.2). The delimiter used between the PS Configuration File name and hash value is the '@' character (e.g., “configfile1.txt@23423487987345”). The PS element updates the cabhPsDevProvConfigHash MIB object with the received hash value.~~

2. The Config File Generator adds the hash value, calculated in Step 1, to the PS Configuration File as the last TLV setting (immediately before the end of data marker) using a type 53 TLV. The PS Configuration File is then made available to the appropriate TFTP server.

3. The PS element downloads the ~~named file from the configured TFTP server.~~PS Configuration File.

4. The PS MUST update the cabhPsDevProvConfigHash MIB object with the hash value from the hash TLV created in steps 1 and 2. The hash value MUST be stored in hexadecimal number format.

~~4.~~ 5. The PS element MUST compute a SHA-1 hash over the ~~entire content~~contents of the PS Configuration File ~~and compare~~excluding the ~~computed~~ hash ~~to~~TLV (used to configure the ~~hash in~~ cabhPsDevProvConfigHash MIB object~~.~~), the end of data marker, and any padding that follows. If the computed ~~and configured hash values~~ hash and the value of the cabhPsDevProvConfigHash MIB object are the same, the PS Configuration File ~~is authenticated~~integrity is verified and the configuration file MUST be processed; otherwise, the file MUST be rejected.

~~5. When authentication is successful, the PS element MUST use the PS Configuration File contents for its configuration.~~

__7.3.3.3.2 Configuration File Authentication Algorithm for SNMP Provisioning Mode__

The procedure for ~~authentication of~~checking the PS Configuration File Hash by the PS element in SNMP Provisioning Mode follows:

1. When the ~~NMS~~Config File Generator of the Provisioning System creates a new PS Configuration File or modifies an existing file, the ~~NMS~~Config File Generator will create a SHA- 1 hash of the entire content of the PS Configuration File, taken as a byte string. The end of data marker and any padding that follow it are not included in the hash calculation.
2. The NMS sends the hash value calculated in step 1 to the PS element via SNMP SET~~ and~~. The PS updates ~~the~~its cabhPsDevProvConfigHash MIB object with the new value.
3. The NMS sends the Name and location of the PS Configuration File via SNMP SET~~ and~~. The PS updates ~~the~~its cabhPsDevProvConfigFile MIB object ~~(this triggers the TFTP download, see Section 7.3.3.2)~~with the new value.
4. The PS element downloads the named file from the configured TFTP server. If the PS Configuration File contains TLV type 53 the PS MUST ignore it.
5. The PS element MUST compute a SHA-1 hash over the ~~entire content~~contents of the PS Configuration File ~~and compare~~excluding the TLV 53 if it exists, the end of data marker and any padding that follows. If the computed hash ~~to~~and the ~~hash in~~value of the cabhPsDevProvConfigHash MIB object~~. If the computed and configured hash values~~ are the same, the PS Configuration File ~~is authenticated~~integrity is verified and the configuration file MUST be processed; otherwise, the file MUST be rejected.

~~6. When authentication is successful, the PS element MUST use the PS Configuration File contents for its configuration.~~

46. Make the following changes to section 7.3.3.4

Once triggered to download a PS Configuration File, the PS element MUST continue to attempt to download the specified PS Configuration File from the specified location until the PS Configuration File is successfully downloaded and the hash successfully computed as described in Section 7.3.3.3. ~~The specific timeout for TFTP server access is implementation dependent. However, the PS MUST NOT attempt to access the TFTP server more than 3 times in any 5 minute period. At minimum, the PS MUST attempt at least once per 5 minute interval to download the PS Configuration File, until the PS Configuration File is successfully downloaded.~~Retry requirements for TFTP server access are described later in this section.

AND

The PS MUST use an adaptive timeout for TFTP based on binary exponential backoff as described ~~in [RFC 1123] and [RFC 2349].~~below, if the first attempt is not successful, until the PS (CMP) successfully receives the requested file from the TFTP server in the Headend OR until the PS is reset:

- each retry is 2^n second(s) following the previous attempt, where n = [0, 1, 2, 3, 4, or 5]
- n = 0 for the first retry, then is incremented by one for each subsequent attempt until n = 5
- if the CMP does not successfully acquire the requested file following the attempt with n = 5, n is to be reset to 0 and the process repeated.

47. In section 7.4.2 make the following changes:

Once the ~~Wan~~WAN-Man IP stack begins use of the IP address it received from DHCP, it should send an ~~[~~RFC 868~~]~~ time query to the ToD Server. If the ToD server responds with a valid response, the PS operating in DHCP Provisioning Mode will begin using this time of day for event ~~messaging~~message time stamps and security functions. When the PS is operating in SNMP Provisioning Mode, it will use the time of day provided by the Key Distribution Center for event message time stamps and security functions.

48. Section 7.4.3 requires significant enhancement:

Upon reset, the ~~PS element~~Portal Services Element MUST initialize its time to 0 (0:0.0 January 1, ~~1900) in accordance with [RFC 868]~~1970).

The ~~PS element~~Portal Services Element operating in DHCP Provisioning Mode MUST attempt Time of Day time synchronization with the ToD server indicated by the DHCP Option 4, that is received in the DHCP Offer made to the WAN-Man interface following acquisition of a WAN-Man DHCP lease.

The PS MUST combine the time retrieved from the ToD server with the time offset provided by DHCP Option 2, to create the current local time.

The ~~PS element~~Portal Services Element operating in DHCP Provisioning Mode MUST make use of the current local time calculated from the time retrieved from the ToD server and time offset received by DHCP Option 2 for ~~event messaging and security~~any functions requiring time of day, and which need only be accurate to the nearest second.

The Portal Services Element operating in SNMP Provisioning Mode MUST make use of the current local time provided by the Key Distribution Center server for any functions requiring time of day.

The priority for the system time of day clock for an Embedded PS is as follows:

- First priority: time of day acquired from the KDC server
- Second priority: time of day acquired from the ToD server
- Third priority: time of day acquired from the cable modem
- Fourth priority: time initialized to January 1, 1970

If an Embedded PS operating in SNMP Provisioning Mode acquires time of day from the KDC server, it MUST use this value for the system time of day clock, even if this means overwriting the system time acquired by the CM.

An Embedded PS operating in DHCP Provisioning Mode MUST use the most recent valid time of day acquired from the ToD server for the system time of day clock, even if this means overwriting the system time acquired by the CM.

If an Embedded PS is unable to acquire time of day from the KDC server OR from the ToD server, it MUST use time of day acquired by the cable modem for the system time of day clock.

If an Embedded PS is unable to acquire time of day from the KDC server OR from the ToD server, AND is unable to acquire valid time of day from the cable modem, it MUST use time of day initialized in the boot process to January 1, 1970 for the system time of day clock.

The priority for the system time of day clock for a Standalone PS is as follows:

- First priority: time of day acquired from the KDC server
- Second priority: time of day acquired from the ToD server
- Third priority: time initialized to January 1, 1970

If a Standalone PS operating in SNMP Provisioning Mode acquires time of day from the KDC server, it MUST use this value for the system time of day clock.

A Standalone PS operating in DHCP Provisioning Mode MUST use the most recent valid time of day acquired from the ToD server for the system time of day clock.

If a Standalone PS is unable to acquire time of day from the KDC server OR from the ToD server, it MUST use time of day initialized in the boot process to January 1, 1970 for the system time of day clock.

The PS element MUST continue to attempt to communicate with the Time of Day server, until local time is established. The specific timeout for Time of Day Requests is implementation dependent. However, the PS Time of Day client MUST NOT exceed more than 3 ToD requests in any 5 minute period. At minimum, the PS Time of Day client MUST issue at least 1 ToD request per 5 minute period, until local time is established.

If the ToD server does not respond with a valid response the PS MUST do the following, not necessarily in the order listed:

- set the value of cabhPsDevTodSyncStatus to '2' (ToD access failed),
- if there are active leases in the LAN-Trans realm as indicated by a nonzero value for cabhCdpLanTransCurCount, set cabhCdpLanAddrCreateTime to the current time and set cabhCdpLanAddrExpireTime to the value of cabhCdpLanAddrCreateTime plus the value of cabhCdpServerLeaseTime for each active lease (Expire Time = CreateTime + LeaseTime),
- log the failure and generate a standard event defined in Annex B, and
- continue to retry communication with the ToD server until local time is established~~.~~, and
- attempt to download the PS Configuration File as described in Section 7.3.3.2.

If the ~~PS successfully synchronizes its time reference with the ToD server in the cable network,~~ToD server does respond with a valid response the PS MUST do the following, not necessarily in the order listed:

- set the value of cabhPsDevTodSyncStatus to '1' (ToD ~~synchronization successful).~~access succeeded),
- if there are active leases in the LAN-Trans realm as indicated by a nonzero value for cabhCdpLanTransCurCount, set cabhCdpLanAddrCreateTime to the current time and set cabhCdpLanAddrExpireTime to the value of cabhCdpLanAddrCreateTime plus the value of cabhCdpServerLeaseTime for each active lease (Expire Time = CreateTime + LeaseTime),
- attempt to download the PS Configuration File as described in Section 7.3.3.2.

If the value of cabhPsDevTodSyncStatus is '1', i.e., if local time has already been established, it is not necessary for the Time of Day client to issue a ToD request.

The PS MUST send and receive ToD messages only through a WAN-Man Interface.

49. In section 8.1.2 delete the third bullet.

~~•The PS Packet Handling function may forward broadcast traffic to all LAN and WAN-Data interfaces transparently. Throttling of broadcast traffic is not required. It is assumed that the DOCSIS cable modem has the capability to filter broadcast IP traffic.~~

50. In section 8.2.2.2 add the following paragraphs:

not have an existing Mapping. If no available IP address exists in the WAN-Data IP address pool, the Dynamic C-NAT Mapping can not be created, and this traffic is dropped, and an event is generated (see Annex B).

The LAN-Trans IP address portion of the Dynamically created C-NAT Mapping tuples is provided by the pool of IP addresses defined by the cable operator in the CDP MIB. The CAP enters the tuple of the unique WAN-Data IP address and a unique LAN-Trans IP address in the CAP Mapping Table, along with other parameters including WAN and LAN Port numbers, the Mapping Method, and the transport protocol used for the Mapping. The port number will not be translated by the CAP for C-NAT Mappings: the source and destination port numbers in the UDP or TCP header will be unchanged. The CAP will enter the value 0 into the WAN and LAN port number entries of the CAP Mapping Table. The 0-value port number entry will serve two purposes: (1) indicate to the CAP that the port numbers are not to be translated, and (2) indicate to anyone reading the CAP Mapping Table that this is a C-NAT mapping, thereby providing a distinction between C-NAT Mappings (port number 0) and C-NAPT Mappings (nonzero port number).

Dynamic C-NAT Mappings for UDP traffic are destroyed when an inactivity timeout period,

AND

layer functions, including the C-NAT/C-NAPT Transparent Routing function.

It is assumed that when the PS is in Routing mode (C-NAT/C-NAPT), that it will process broadcast traffic in accordance with RFCs 919, 922, 1812, and 2644. It is also assumed that when the PS is in Passthrough Mode, that broadcast traffic will be bridged to all interfaces.

When the PS is in Mixed Bridging/Routing Mode, and receives broadcast traffic sourced from a device in Passthrough Table, the PS is expected to bridge the broadcast to all interfaces. When the PS is in Mixed Bridging/Routing Mode, and receives broadcast traffic on any WAN interface, the PS is expected to bridge the broadcast to all LAN interfaces.

It should be noted that the USFS functionality (Section 8.2.2.3) is applied in each of the three primary packet-

51. make the following changes to section 8.2.2.4

The CAP supports WAN-to-LAN Multicast traffic by transparently bridging downstream IGMP messaging [RFC 2236] and downstream IP Multicast packets. In addition, when in C-NAT/C-NAPT Transparent Routing Mode, the CAP performs address translation on upstream IGMP messages sourced by LAN IP Devices residing in the LAN-Trans domain. The CAP forwards

52. section 8.3.1 requires enhancing:

All logical IP interfaces on the ~~PS~~Portal Services element MUST be compliant with ~~[~~RFC ~~1122], Sections~~1122, sections 3 and 4, to enable standard communication with Internet Hosts.

The CAP MUST support WAN-to-LAN Multicast traffic~~,~~ by transparently bridging WAN-to-LAN IGMP messaging and WAN-to-LAN IP Multicast packets as defined in ~~[RFC 2236]~~RFC 2236.

If the Primary Packet-handling Mode, cabhCapPrimaryMode, is set to Passthrough, all LAN-to-WAN IGMP messaging MUST be transparently bridged.

If the Primary Packet-handling Mode, cabhCapPrimaryMode, is set to C-NAPT, the source IP address for all LAN-to-WAN IGMP messages, sourced from LAN IP Devices residing in the LAN-Trans Domain, MUST be translated to the WAN-Data IP address being used for C-NAPT mappings, and then forwarded out to the WAN.

If the Primary Packet-handling Mode, cabhCapPrimaryMode, is set to C-NAT, the source IP address for all LAN-to-WAN IGMP messages - sourced from LAN IP Devices residing in the LAN-Trans Domain that have an IP address that is part of an existing C-NAT mapping - MUST be translated to the WAN-Data IP address being used in that C-NAT mapping, and then forwarded out to the WAN.

53. Section 8.3.2 requires enhancing:

The CAP MUST set the WAN and LAN port numbers (cabhCapMappingWanPort and cabhCapMappingLanPort, respectively) of the CAP Mapping Table equal to zero for each Dynamic C-NAT Mapping it creates.

If the cable operator creates or changes a row in the CAP Mapping Table, i.e., if a row is created via the static mapping method (cabhCapMappingMethod = static(1)), AND the port number objects of the row (cabhCapMappingLanPort and cabhCapMappingWanPort) are not specified, the CAP MUST enter zero for cabhCapMappingLanPort and cabhCapMappingWanPort for that row.

The CAP MUST NOT translate the port number for any packet whose IP address appears in the CAP Mapping Table with a port number of zero.

If the Primary Packet-handling Mode, cabhCapPrimaryMode, is set to C-NAPT, the CAP MUST make certain there exists a current WAN IP address (with a current DHCP lease from Headend provisioning) before attempting to use this IP address as part of a C-NAPT Mapping. If the CAP is unable to create a C-NAPT Mapping, due to not having a current WAN IP Address or due to port number depletion, it ~~must~~MUST generate a standard event (as defined in Annex B).

LAN-to-LAN uni-cast traffic MUST never be routed or bridged out a WAN interface.

When the DHCP lease of a WAN-Data IP address - that is part of C-NAT or C-NAPT mapping - expires, all mappings associated with that IP address MUST be deleted from cabhCapMappingTable.

54. Change a bullet item in section 9.1.1

~~•Refer~~• Via recursive queries to remote DNS servers, provide answers to LAN DNS clients ~~to Headend DNS servers,~~when queried for resolution of non-local hostnames.

55. make the following changes to section 9.2.2.1

- Resolves hostnames for LAN IP Devices, returning their corresponding IP addresses.
- ~~•Refers LAN IP-Devices to external DNS servers~~• Provide DNS answers, via recursive queries beginning with a Headend DNS server, for queries that cannot be resolved via local PS information. This action occurs only when WAN DNS server information is available in the PS. Otherwise, the CNP returns an error indicating that the name cannot be resolved at this time.

Making the CNP the primary DNS server on the ~~customer premis~~LAN avoids the need to reconfigure LAN IP Devices when the state of the WAN connection changes. It also permits changing external DNS server assignment without LAN IP Device reconfiguration.

56. Make the following changes to section 9.2.2.2

When queried to resolve a hostname, the CNP performs the lookup process shown in Figure ~~21.~~9-1. The CNP responds to initial standard DNS queries [RFC 1035], directed to cabhCdpServerDnsAddress, for all name lookups. ~~If~~It is the responsibility of the CNP ~~responds with a referral~~to make recursive queries to external DNS servers~~, it is assumed to be the responsibility of the LAN IP Device to send a query directly to the referred server~~ – beginning with the cabhCdpWanDataAddrDnsIp entries in the CDP's cabhCdpWanDataAddrServerTable – when queried by a LAN IP Device and to respond to that LAN IP Device with either an answer or an error message.

The CNP relies on the CDP's cabhCdpLanAddrTable, to learn the hostnames associated with the current IP addresses of active LAN IP Devices. As long as a LAN IP Device maintains an active DHCP lease with the CDP and has provided a hostname to the CDP (as part of its IP address acquisition process) its name can be resolved by the CNP. If the hostname requested for resolution cannot be found in the cabhCdpLanAddrTable, the CNP ~~returns a DNS referral which points to an~~performs recursive queries to external DNS ~~server~~servers (of which the initial one is learned by the CDC via DHCP options).~~ The IP address of the external DNS server is the last cabhCdpWanDataAddrDnsIp entry in the CDP's cabhCdpWanDataAddrServerTable.~~

57. Section 9.3 needs the following changes:

The CNP MUST ~~operate at least in non-~~support recursive mode, as defined in [RFC 1034].

The CNP answers name queries, ~~using only~~beginning with local information within the PS, and its response messages MUST contain an ~~error, an~~ answer~~, or a referral to an external DNS server.~~ or an error.

The CNP MUST only respond to DNS queries addressed to cabhCdpServerDnsAddress.

The CNP MUST NOT respond to any DNS queries addressed to the PS WAN-Man and WAN-Data IP addresses.

Upon receiving an initial hostname resolution query from a LAN IP Device, the CNP MUST access the CDP's cabhCdpLanAddrTable to look up hostnames associated with IP addresses that are leased to LAN IP Devices.

Regardless of the ~~state~~existence of ~~the~~any cabhCdpWanDataAddrDnsIp ~~entry~~entries in the CDP's cabhCdpWanDataAddrServerTable, if the hostname can be resolved by the CNP from local data, the CNP MUST respond to the hostname resolution query with the IP address of the named LAN IP Device.

~~When functioning as a Non-recursive DNS server: if~~If the queried hostname can not be resolved by the CNP from local data AND the ~~last cabhCdpWanDataAddrDnsIP entry in the~~ CDP's cabhCdpWanDataAddrServerTable is populated with at least one cabhCdpWanDataAddrDnsIP entry, the CNP MUST ~~respond~~attempt to resolve the hostname ~~resolution~~ query ~~with a referral to an external DNS server, represented by the IP address contained in the cabhCdpWanDataAddrDnsIp object~~via recursive queries to external DNS servers, starting with queries to DNS servers represented by cabhCdpWanDataAddrDnsIP entries in the cabhCdpWanDataAddrServerTable.

If the hostname can not be resolved by the CNP from local data AND the cabhCdpWanDataAddrDnsIp object is not populated, the CNP MUST respond to the hostname resolution query with the appropriate error specified by ~~[RFC 1035].~~RFC 1035.

~~When the last remaining Wan-Data DHCP lease expires, the CDC MUST clear all cabhCdpWanDataAddrDnsIp entries from the cabhCdpWanDataAddrServerTable.~~

58. In section 10.2.1, add an additional item to the table:

QoS 4	CQoS must support both the Embedded PS and Stand Alone PS HA configurations.

59. Section 10.2.2.1 requires modification:

The Portal Services (PS) element is a logical element that contains network addressing, management, security, and QoS portal components that provide translation functions between the HFC network and the home network. The PS resides in HA devices only (see Section 5). The QoS component is referred to as the Cable Quality-of-Service Portal (CQP).

10.2.2.1.1 CQP Component

The PS element includes a Cable Quality-of-Service Portal (CQP) component. The CQP acts as a CQoS portal for IPCablecom compliant applications. Its primary function is to forward QoS messaging between the CMTS and IPCablecom Applications.

10.2.2.1.2 Standalone PS Configuration

This Recommendation does not define QoS requirements between a PS and a CM, and thus functions for maintaining data session priorities and avoiding contention due to asynchronous access by multiple devices will not be specified. It is recommended that this interface be a high bandwidth, dedicated PS-to-CM connection (not shared with other devices) to minimize QoS packet jitter due to multi-device contention.

60. Make the following changes to section 11.3.1.1

~~The~~In SNMP provisioning mode, the PS Element, the NMS (i.e. SNMP Manager) and KDC~~ enabled authentication~~ MUST follow the specification for Kerberos/PKINIT as defined in ~~[J.170]. The~~ITU-T J.170 sections 6.4 and 6.5, unless otherwise noted in this specification. The IPCable2Home KDC is equivalent to or the same as the IPCablecom ~~operator~~MSO KDC (IPCablecom specifies the use of several KDCs). The IPCable2Home specification uses the term Network Management Systems (NMS) to provide SNMP functionality. In referencing the IPCablecom suite of specifications, it is noted that IPCablecom uses the term provisioning server to denote SNMP functionality. The reader should be aware that this SNMP functionality in general should be compatible within both specifications, however they are not identical as IPCablecom and IPCable2Home specific information is specified. The PS element MUST act as the client to the KDC. In the IPCablecom Security Specification the MTA is the client~~. It~~ and it is expected that IPCable2Home implementations will use the client functionality specified for the MTA for the PS element. The PS element makes use of Kerberos for SNMP. The certificates used in PKINIT for IPCable2Home are specified in the PKI Section of this document. Where IPCablecom specifies an MTA device certificate, this Recommendation provides a certificate for the PS Element (PS Element Certificate), and implementations of PS Elements MUST include the PS Element Certificate.

~~IPCablecom Security for the~~The following sections for Kerberos functionality ~~does~~from J.170 do not apply to this Recommendation:

- section 6.4.2.1.3 Pre-Authenticator for Provisioning Sever Location
- section 6.4.6 MTA Principal Names
- section 6.4.7 Mapping of MTA MAC Address to MTA FQDN
- section 6.4.9 Service Key Versioning~~, Section~~
- section 6.4.10 Kerberos Cross-Realm Operation~~, Section 6.4.11~~
- section 6.5.2.1 Rekey ~~Message, Section 6.5.4~~Messages
- section 6.5.3 Kerberized IPSec~~, Section 6.5.6~~
- section 6.4.5 Kerberos Server Locations and Naming Conventions~~, Section 6.4.6.3 for the CMS~~

61. In section 11.3.1.2 delete the las bullet item:

~~•Replace MTA-FQDN-Map as defined in the IPCablecom Security Specification with FQDN-Map~~

62. Add new section 11.3.1.3:

11.3.1.3 IPCable2Home profile for Kerberos Server Locations and Naming Conventions

Kerberos Realm names MAY use the same syntax as a domain name, however Kerberos Realms MUST be in all capitals. Kerberos Realm details MUST be followed according to J.170, appendix B.

The KDC conventions listed in J.170, section 6.4.5.2 are considered informative for this Recommendation with the expectation that the KDC will perform the necessary functions in the back office to exchange the appropriate information with the NMS (provisioning server or SNMP manager). The PS element has provided the KDC with the provisioning server IP address in the AS Request as the necessary information to make appropriate contact between the KDC and provisioning server.

A PS Element principal name MUST be of type NT-SRV-INST with exactly two components, where the first component MUST be the string "PSElement" (not including the quotes) and the second component MUST be the WAN-Man-MAC address:

PSElement/<WAN-Man-MAC>

where <WAN-Man-MAC> is the WAN Management MAC address of the PS Element. The format the <WAN-Man-MAC> MUST be "XX:XX:XX:XX:XX:XX" (not including the quotes) where X is a hexadecimal character of the MAC address. Hexadecimal characters a-f MUST be in lower case.

63. Make the following changes to section 11.3.2.1.3

The extensions (subjectKeyIdentifier, authorityKeyIdentifier, KeyUsage, and BasicContraints~~, Signature Algorithm, SubjectName and IssuerName~~) MUST follow ~~[~~RFC ~~3280].~~3280. Any other certificate extensions MAY also be included as non-critical. The encoding tags are [c:critical, n:non-critical; m:mandatory, o:optional] and these are identified in the table for each certificate.

64. Modify as shown below:

11.3.2.1.3.2 authorityKeyIdentifier

The authorityKeyIdentifier extension included in all certificates as required by ~~[~~RFC 3280~~]~~ MUST include the subjectKeyIdentifier from the issuer's certificate (see ~~[~~RFC 3280~~]~~) with the exception of root certificates.

65. Add new section:

11.3.2.1.6 serialNumber

The serial number MUST be a unique, positive integer assigned by the CA to each certificate (i.e., the issuer name and serial number identify a unique certificate). CAs MUST force the serialNumber to be a non-negative integer. The Manufacturer SHOULD NOT impose or assume a relationship between the serial number of the certificate and the serial number of the modem to which the certificate is issued.

Given the uniqueness requirements above, serial numbers can be expected to contain long integers. Certificate users MUST be able to handle serialNumber values up to 20 octets. Conformant CAs MUST NOT use serialNumber values longer than 20 octets.

66. Make the following changes to section 11.3.2.2

The ~~Certificate Hierarchies are generic in nature and applicable to all applications needing certificates. This means the basic infrastructure~~certificate hierarchies described in this document can apply to all ITU projects needing certificates. Each project may adopt this hierarchy as there is an opportunity to move to a more generic, shared certificate structure. Also each project may need to make specific adjustments in the requirements for that particular project. It is a goal to create a PKI which can be re-used for every ~~application (DOCSIS, IPCablecom, PS)~~project. There may be differences in the end-entity certificates required for each project, but in the cases where end-entity certificates overlap, one end-entity certificate could be used ~~to support the overlap~~for several services within the cable infrastructure. For example, IPCablecom requires a KDC for the service provider and ~~this Recommendation can take advantage of a KDC that supports IPCablecom to provide mutual authentication~~IPCable2Home also requires a KDC for the service provider. If the service provider is running both network architectures on their systems, they can use the same KDC and the same KDC certificate for communication on both systems, i.e., IPCablecom and ~~this application~~IPCable2Home. In this case, ~~this application's~~the IPCable2Home KDC is equivalent to or the same as the IPCablecom ~~operator~~MSO KDC (IPCablecom specifies the use of several KDCs).

67. Add the following note at the end of Table 32, Table 40 and Table 41:

The Company Name in the Organization (O) field MAY be different than the Company Name (CN) in the Common Name field.

68. *11.3.2.2.2.2 Code Verification CA Certificate*

The Code Verification CA Certificate MUST be verified as part of a certificate chain containing the Code Verification Root CA Certificate, Code Verification CA Certificate and the Code Verification Certificate. ~~There MAY be more than one Code Verification CA~~A Stand-Alone PS MUST only support one CVC CA at a time.

69. Make the following changes to section 11.3.2.2.2.3.

This certificate MUST be verified as part of the certificate chain containing the Code Verification Root CA Certificate, the Code Verification CA Certificate, and the Code Verification Certificates.

Table 36. Manufacturer Code Verification Certificate

Manufacturer Code Verification Certificate	

Subject Name Form	C=<country>~~,~~
	O=<CompanyName>~~,~~
	[~~S~~ST=<state/province>]~~,~~
	[L=<city>]~~,~~
	CN=<CompanyName> Mfg CVC

Intended Usage	The Code Verification CA issues this certificate to each authorized Manufacturer. It is used in the policy set by the cable operator for secure software download. The CompanyName in the O and CN fields may be different.

Signed By	Code Verification CA
Validity Period	2 years
Modulus Length	1024, 1536, 2048
Extensions	keyUsage[c,m](digitalSignature, keyEncipherment), extendedKeyUsage [c,m] (id-kp-codeSigning), authorityKeyIdentifier [n,m]

70. Make the following changes to section 11.3.2.2.2.4 *Code Verification Certificate*

The Code Verification Certificate MUST be verified as part of a certificate chain containing the Code Verification Root CA Certificate, the Code Verification CA Certificate, and the Code Verification Certificate.

Table 37 Code Verification Certificate

Code Verification Certificate	

Subject Name Form	C= <country>~~;~~
	O=~~;~~
	CN= CVC

Intended Usage	The Code Verification CA issues this certificate. It is used to authenticate certified code. It is used in the policy set by the cable operator for secure software download.
Signed By	Code Verification ~~Root~~ CA
Validity Period	2 years
Modulus Length	1024, 1536, 2048
Extensions	keyUsage[c,m](digitalSignature, keyEncipherment), extendedKeyUsage [c,m] (id-kp-codeSigning), authorityKeyIdentifier [n,m]

71. Make the following changes to section 11.3.2.2.2.5

The Service Provider Code Verification Certificate MUST be verified as part of a certificate chain containing the Code Verification Root CA Certificate, the Code Verification CA Certificate, and the Service Provider Code Verification Certificate.

Table 38 Service Provider Code Verification Certificate

Service Provider Code Verification Certificate	

Subject Name Form	C=<country>~~;~~
	O=<CompanyName>~~;~~
	[~~S~~ST=<state/province>~~;~~]
	[L=<city>~~;~~]
	CN=<CompanyName> Service Provider CVC

Intended Usage	The Code Verification CA issues this certificate to each authorized Service Provider. It is used in the policy set by the cable operator for secure software download. The CompanyName in the O and CN fields may be different.

Signed By	Code Verification ~~Root~~ CA
Validity Period	2 years
Modulus Length	1024, 1536, 2048
Extensions	keyUsage[c,m](digitalSignature, keyEncipherment), extendedKeyUsage [c,m] (id-kp-codeSigning), authorityKeyIdentifier [n,m]

72. Table 42 in section 11.3.2.2.3.4 needs modification:

Table 42 KDC Certificate

KDC Certificate	

Subject Name Form	C=<country>~~,~~
	O=<Company Name>~~,~~
	[OU=<Local System Name>~~,] OU=<KDC>, ~~]
	OU= Key Distribution Center
	~~CN=<IP Address of the KDC server>~~CN=<DNS Name>

Intended Usage	This certificate is issued either by the Service Provider CA or the Local System CA. It is used to authenticate the identity of the KDC to the Kerberos clients during PKINIT exchanges. This certificate is passed to the PS element inside the PKINIT reply.
Signed By	Service Provider CA or the Local System CA
Validity Period	20 years
Modulus Length	1024, 1536, 2048
Extensions	keyUsage[~~n~~c,o](digitalSignature~~, keyEncipherment),~~) authorityKeyIdentifier~~, The keyUsage extension is optional. When it is used it SHOULD be marked as non-critical~~[n,m](keyIdentifier=<subjectKeyIdentifier value from CA certificate>) subjectAltName [n,m] (see ~~IPCablecom Security specification).~~ [PKT-SEC], appendix C)

73. Section 11.3.3.1.2 needs major overhauling:

In order to establish SNMPv3 ~~keysPS compliant CM MUST support the "SNMPv3 Initialization" described below.~~keys, all SNMP interfaces MUST utilize the SNMPv3 initialization and key changes procedure described below..

To support SNMPv3 initialization and key changes the PS element MUST also be capable of receiving TLVs of type 34, 34.1, and 34.2 as defined in section B.C.1.2.8 of the Radio Frequency Interface specification, [J.112] and implement the key-change mechanism specified in RFC 2786 which includes the usmDHKickstartTable MIB object.

11.3.3.1.2.1 SNMPv3 Initialization

~~(Note: The cable modem is designated as having "very-secure" security posture in the context of RFC-2574 Appendix A and RFC-2575 Appendix A. This means that default usmUser and vacmAccess entries defined in RFC-2574 Appendix A and RFC-2575 Appendix A MUST NOT be present.)~~

~~1.~~ For each of up to 5 different security names, the ~~Manager~~Ultimate Authorization (PS Administrator) generates a pair of numbers~~:a.~~ ~~Manager~~. First, the PS Administrator generates a random number Rm .

~~b.~~ ~~Manager~~Then, the CH Administrator uses the DH equation to translate Rm to a public number z. The equation is as follows:

z = g ^ Rm MOD p

where g is ~~the~~ from the set of Diffie-Hellman parameters, and p is the prime from those parameters.

~~2.~~ ~~CM configuration file~~The PS Configuration File is created to include ~~(security name, public number) pair and CM~~the (security name, public number) pair. The PS MUST support a minimum of 5 pairs. For example:

TLV type 34.1 (~~SnmpV~~SNMPv3 Kickstart Security Name) = ~~docsisManager~~PS Administrator

TLV type 34.2 (~~SnmpV~~SNMPv3 Kickstart Public Number) = z

~~During the CM boot up process, the above values (security name, public number) will (MUST) be populated in the usmDHKickstartTable.~~

~~At this point:~~
~~usmDHKickstartMgrpublic.1 = "z" (octet string)~~
~~usmDHKickstartSecurityName.1 = "docsisManager"~~

The PS MUST support the VACM entries defined in Section 6.3.6.3. Only VACM entries specified by the corresponding security name in the PS Configuration File MUST be active.

During the PS boot process, the above values (security name, public number) MUST be populated in the usmDHKickstartTable.

At this point:

usmDHKickstartMgrpublic.1 = "z" (octet string)

usmDHKickstartSecurityName.1 = "PS Administrator"

When usmDHKickstartMgrpublic.n is set with a valid value during the registration, a corresponding row is created in the usmUserTable with the following values:

usmUserEngineID: localEngineID

usmUserName: usmDHKickstartSecurityName.n value

usmuserSecurityName: usmDHKickstartSecurityName.n value

~~usmUserCloneForm~~usmUserCloneFrom: ZeroDotZero

usmUserAuthProtocol: usmHMACMD5AuthProtocol

usmuserAuthKeyChange: (derived from set value)

usmUserOwnAuthKeyChange: (derived from set value)

usmUserPrivProtocol: usmDESPrivProtocol

usmUserPrivKeyChange: (derived from set value)

usmUserOwnPrivKeyChange: (derived from set value

usmUserPublic~~: ""~~

usmUserStorageType: permanent

usmUserStatus: active

Note: For (~~CM~~PS) dhKickstart entries in usmUserTable, Permanent means it MUST be written to but not deleted and is not saved across reboots.

After the ~~CM has registered with the AN.~~ PS has completed initialization (indicated by a value of '1' (pass) for cabhPsDevProvState):

1. The PS generates a random number xa for each row populated in the usmDHKickstartTable which has a non -zero length usmDHKickstartSecurityName and usmDHKickstartMgrPublic.
2. The PS uses DH equation to translate xa to a public number c (for each row identified above).

   c = g ^ xa MOD p

where g is the from the set of Diffie-~~Helman~~Hellman parameters, and p is the prime from those parameters.

At this point:

usmDHKickstartMyPublic.1 = "c" (octet string)

usmDHKickstartMgrPublic.1 = "z" (octet string)

usmDHKickstartSecurityName.1 = "~~docsisManager"~~PS Administrator"

3. The PS calculates shared secret sk where sk = z ^xa mod p.
4. The PS uses sk to derive the privacy key and authentication key for each row in usmDHKickstartTable and sets the values into the usmUserTable.

   As specified in RFC~~-~~ 2786, the privacy key and the authentication key for the associated username, "~~docsisManager~~PS Administrator" in this case, is derived from sk by applying the key derivation function PBKDF2 defined in PKCS#5 v2.0.

```
privacy key <---        PBKDF2( salt =  0xd1310ba6,
                     iterationCount = 500,
                     keyLength = 16,
                     prf = id-hmacWithSHA1 )
authentication  key  <----        PBKDF2( salt = 0x98dfb5ac,
                        iterationCount = 500,
                        keyLength = 16 (usmHMACMD5AuthProtocol),
                        prf = id-hmacWithSHA1 )
```

   At this point the ~~CM~~PS (CMP) has completed its SNMPv3 initialization process and MUST allow appropriate access level to a valid securityName with the correct authentication key and/or privacy key.

   ~~Compliant CM~~The PS MUST properly populate keys to appropriate tables as specified by the SNMPv3 -related RFCs and RFC~~-~~ 2786.

5. The following describes the process that the manager uses to derive ~~CM'~~the PS's unique authentication key and privacy key.

   The SNMP manager accesses the contents of the usmDHKickstartTable using the security name of ~~'dhKickstart' with no authentication.~~'dhKickstart' with no authentication.

~~Compliant CM MUST provide preinstalled entries in the USM table and VACM tables to correctly create user 'dhKickstart' of security level noAuthnoPriv that has read only access to system group and usmDHkickstartTable.~~

The PS MUST provide pre-installed entries in the USM table and VACM tables to correctly create user 'dhKickstart' of security level noAuthNoPriv that has read-only access to system group and usmDHkickstartTable.

If the PS is in Coexistence Mode and is configured to use SNMPv3 the Group specification for the dhKickstart View MUST be implemented as follows:

dhKickstart Group

vacmGroupName 'dhKickstart'

vacmAccessContextPrefix ''

vacmAccessSecurityModel 3 (USM)

vacmAccessSecurityLevel NoAuthNoPriv

vacmAccessContextMatch exact

vacmAccessReadViewName 'dhKickstartView'

vacmAccessWriteViewName ''

vacmAccessNotifyViewName ''

vacmAccessStorageType permanent

vacmAccessStatus active

The VACM View for the dhKickstart view MUST be implemented as follows:

dhKickstartView subtree 1.3.6.1.2.1.1 (System Group) and 1.3.6.1.3.101.1.2.1 (usmDHkickstartTable)

The SNMP manager gets the value of ~~CM'~~the PS's usmDHKickstartMypublic number associated with the ~~security name that~~securityName for which the manager wants to derive authentication and privacy keys ~~for. With the manager's knowledge of~~. Using the private random number, the manager can calculate the DH shared secret. From that shared secret, the manager can derive operational authentication and confidentiality keys for the ~~security name~~securityName that the manager is going to use to communicate with the ~~CM~~PS.

11.3.3.1.2.2 Diffie-Hellman Key Changes

~~To support SNMPv3 initialization and key changes the PS element MUST also be capable of receiving TLVs of type 34, 34.1, and 34.2 as defined in section B.C.1.2.8 in the DOCSIS Radio Frequency Interface specification, [J.112 2001] and implement~~The PS MUST support the key-change mechanism specified in ~~[RFC 2786] which includes the usmDHKickstartTable MIB object.~~RFC 2786.

74. More changes relative to security.

11.3.3.2.1 SNMPv3 Encryption Algorithms

The encryption Transform Identifiers ~~to be used by the~~for Kerberized SNMPv3 key management ~~to negotiate an encryption algorithm for use by SNMPv3 are the same ones~~MUST be followed as defined in section 6.3.1 in ~~[~~J.~~170].~~170.

11.3.3.2.2 SNMPv3 Authentication Algorithms

The authentication ~~Transform Identifiers to be used by the~~algorithms for Kerberized SNMPv3 key management ~~to negotiate a message authentication algorithm for use by SNMPv3 in are the same ones~~MUST be followed as defined in section 6.3.2 in ~~[~~J.~~170].~~170.

11.3.3.2.3 Kerberized SNMPv3

The Kerberized key management profile specific for SNMPv3 ~~is the same profile~~MUST be followed as defined in section ~~6.5.7~~6.5.4 in ~~[~~J.~~170].~~170.

75. Delete the introductory text in section 11.3.4.2:

~~This section describes the IPCablecom Secured DQoS Architecture in order to discuss how these messages interact with the firewall in the PS. Within DQoS, the Multimedia Terminal Adaptor (MTA) communicates with the CMTS and Call Management Server (CMS) to establish the necessary QoS for its IPCablecom services. The MTA is embedded with the DOCSIS CM. Below are a table and diagram of the devices, the communication protocol and the security protocol for DQoS.~~

76. Section 11.3.5.1 requires enhancement:

The security policy defines the desired level of security/functionality for a subscriber's firewall. More than one may exist to choose from. The files containing the corresponding rule set for these security policies are maintained on an operator file server. The PS MUST use an ~~[~~RFC 1350~~]~~ compliant TFTP client to download the firewall rule set configuration file.~~ To authenticate the rule set file download, the authentication algorithm defined in Section 7.3.3.3.2 MUST be used with the corresponding hash and file name management parameters defined in Section 11.3.5.2 below.~~

~~Using the management interface of the Security MIB, the operator configures the security policy rule set file parameters listed in Section 11.3.5.2 and then follows the procedure defined in Section 7.3.3.3.2 to download and authenticate the file. If the download is a success, the security policy rule set file MUST be "activated" on the firewall. If the authentication fails, the policy rule set MUST be discarded.~~

The Firewall Configuration File download MUST be triggered when the value of the cabhSecFwPolicyFileURL MIB object changes by either the PS Configuration File or by a SNMP SET command. If the value of the cabhSecFwPolicyFileURL MIB object does not change, the Firewall Configuration File download MUST NOT be triggered.

The procedure for checking the integrity of the Firewall Configuration File by the PS element follows:

1. The Firewall Config File Generator will create a SHA-1 hash of the entire contents of the Firewall Configuration File, taken as a byte string.
2. The provisioning system sends the hash value calculated in step 1 to the PS element in one of two ways:

   a) modifies the value of the cabhSecFwPolicyFileHash MIB object via a type 28 TLV in the PS Configuration File. The firewall configuration file hash value MUST be stored in hexadecimal number format.

   b) sends an SNMP SET to update the cabhSecFwPolicyFileHash MIB object. The firewall configuration file hash value MUST be stored in hexadecimal number format.
3. The provisioning system sends the Name and location of the Firewall Configuration File to trigger the

download of the Firewall Configuration File in one of two ways:

a) modifies the cabhSecFwPolicyFileURL MIB object via a type 28 TLV in the PS Configuration File

b) sends an SNMP SET to update the cabhSecFwPolicyFileURL MIB object.

4. If the cabhSecFwPolicyFileOperStatus is not inProgress(1) and a value has been set for the cabhSecFwPolicyFileHash MIB object and the value of the cabhSecFwPolicyFileURL MIB object is updated then the PS element MUST immediately download the named file from the configured TFTP server.

5. The PS element MUST compute a SHA-1 [FIPS 186] hash over the entire contents of the Firewall Configuration File and compare the computed hash to the hash represented by the value of the cabhSecFwPolicyFileHash MIB object. If the computed hash and the value of the cabhSecFwPolicyFileHash MIB object are the same, the integrity of the Firewall Configuration File is verified and the Firewall Configuration File MUST be used, otherwise the file MUST be rejected.

Successful download of the Firewall Configuration File is defined as complete and correct reception by the PS element the contents of the Firewall Configuration File within the TFTP timeout period and computation by the PS the hash value for the Firewall Configuration File with no errors resulting from the computation.

The Firewall Configuration File policy settings MUST be persistent across reboots of the PS element.

Activation and deactivation of the PS firewall is controlled by the cabhSecFwPolicyEnable MIB object. If the value of cabhSecFwPolicyEnable is enable(1) the PS firewall MUST be activated after, and not before, the cabhPsDevProvState MIB has a value of 'pass' (1) indicating that the provisioning process is complete. This will provide the ability to change the firewall policy via a power cycle of the PS when the WAN management access has been accidentally restricted. The PS firewall MUST NOT be enabled if the value of cabhSecFwPolicyEnable is disable(2).

The cabhSecFwPolicyCurrentVersion MIB MUST always reflect the version of the policy installed on the PS regardless of whether it is currently enabled or disabled in the cabhSecFwPolicyEnable MIB.

77. Modify section 11.3.5.3 Firewall Event Log

The firewall MUST be capable of logging the following types of events:

TYPE 1: attempts from both private and public clients to traverse the Firewall that violate the Security Policy.

TYPE 2: identified Denial of Service attack attempts.

TYPE 3: changes made to any of the ~~active~~following firewall ~~policy or firewall configuration parameters.~~management parameters:

- cabhSecFwPolicyFileURL
- cabhSecFwPolicyFileCurrentVersion
- cabhSecFwPolicyFileEnable

The choice of which types of firewall events actually get logged is configured through the Security MIB interface as described in Section ~~11.3.5.2.~~11.3.5.4.

The firewall MUST log events associated with the download via TFTP of the firewall policy file as appropriate. Refer to Annex B, Table B-1, Defined Events for IPCable2Home (CSP Process, Firewall TFTP sub-process).

ALSO

In the last paragraph of the section, change days to hours (twice)

78. It is necessary to add some default values in section 11.3.5.4

The following management parameters MUST be implemented in the PS as defined by the Security MIB to monitor/configure firewall event logging:

cabhSecFwEventType1Enable - Enables or disables logging of type 1 firewall event messages. Default = disable (2)

cabhSecFwEventType2Enable - Enables or disables logging of type 2 firewall event messages. Default = disable (2)

cabhSecFwEventType3Enable - Enables or disables logging of type 3 firewall event messages. Default = disable (2)

cabhSecFwEventAttackAlertThreshold - If the number of type 1 or 2 hacker attacks exceeds this threshold in the period defined by the cabhSecFwEventAttackAlertPeriod object, a firewall message event MUST be logged with priority level 4. The default is set to the highest allowed integer value. This MIB MUST be ignored if the cabhSecFwEventAttackAlertPeriod is set to 0 and an event message MUST NOT be sent.Default = 65535

cabhSecFwEventAttackAlertPeriod - Indicates the period to be used in past ~~days~~hours for the cabhSecFwEventAttackAlertThreshold object. Default = 0

79. Section 11.3.6 requires some minor modification:

The Standalone PS MUST support the following software download support MIBs defined in ~~[~~RFC 2669~~]~~:

AND

The Standalone PS MUST support the following software download support MIBs defined in [~~J.112-B, Annex O~~draft-ietf-ipcdn-bpiplus-mib-05]:

AND

The Standalone PS MUST support the following configuration download support MIB:

80. Section 11.3.7 requires the following changes:

~~The~~A Standalone PS ~~element in a device~~Element MUST be capable of remotely downloading a software image over the network. As described in Section 6.3.7, secure software download to an Embedded PS is controlled by the cable modem. The new software image would allow the operator to improve performance, accommodate new functions and features, correct design deficiencies, and to allow a migration path ~~for~~of IPCable2Home devices as this ~~Specification~~Recommendation evolves. The software download capability MUST allow the functionality of the PS element to be changed without requiring that cable system personnel physically visit and reconfigure each unit.The Standalone PS secure software download process addresses the following primary system requirements:

AND

- A Co-Signer (operator or ~~PS~~Certification body) MAY counter sign the code file in addition to the manufacturer's signature.

AND

- (Optional): The PS element SHOULD be able to update the CVC Root CA ~~Public Key~~Certificate stored in the device.

AND

- (Optional): The PS element SHOULD be able to update the Service Provider Root CA Certificate stored in the device.

AND

The optional downloading of the Service Provider Root CA Certificate, CVC Root CA ~~Public Key~~Certificate, CVC CA Certificate, and/or the Manufacturer CA Certificate as a part of the Code File ~~makes it possible to~~are clearly ~~discriminate~~separated from the code image ~~from~~and the other parameters in the code download file. It ~~also makes it~~is possible to change the ~~Root CVC Public Key, the CVC CA Certificate, the Manufacturer CA Certificates or SignedData parameters in the code download file without disrupting or changing the code image that the PS element will receive. This~~ Service Provider Root CA Certificate, CVC Root CA Certificate, CVC CA Certificate, and/or the Manufacturer CA Certificate understood by the PS element by including the new certificates in the code image. Inclusion of the Manufacturer CVC Certificate and/or a co-signer CVC and corresponding CVS permits the PS element to verify that the code image has not been altered ~~even though the code download file changed due to changes in the CVC Root CA Public Key, the CVC~~since the Service Provider Root CA Certificate, CVC Root CA Certificate, CVC CA Certificate, and/or the Manufacturer CA ~~Certificates~~Certificate or SignedData parameters are appended to the code image.

81. 11.3.7.1 Software Download into Embedded or Standalone PS Elements

As shown in the Figure 11-6 below, a Complaint Home Access (HA) device may implement the cable modem and the PS Element as separate entities or embedded as defined in Section 5.1.3.1.

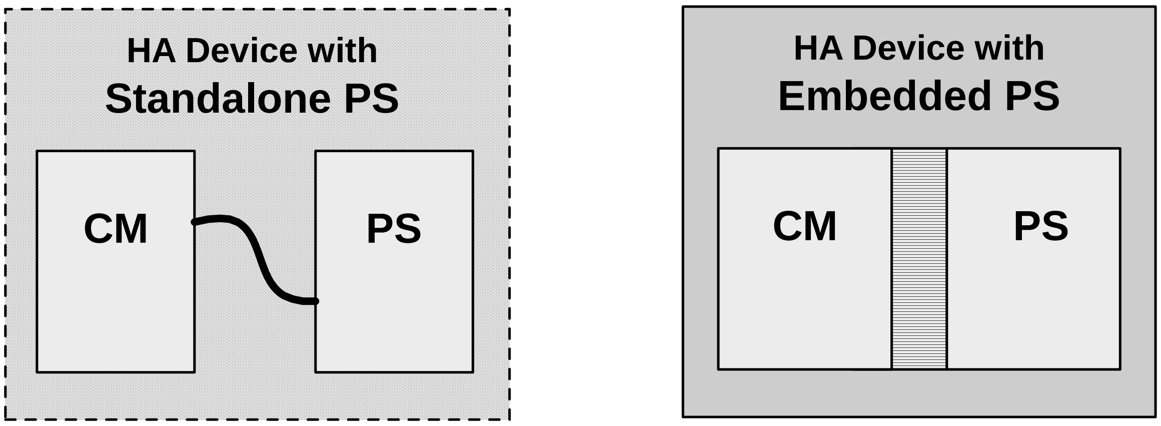

Figure 11-6 HA Device

For IPCable2Home:

- ~~Since~~ If the PS Element is embedded with a cable modem, the PS/CM image MUST be a single image, and the software download MUST be performed only by the cable modem.

- If the PS Element is composed of separate stand alone entities, then the software download for the IPCable2Home elements MUST be performed by the PS Element as described below.

82. Section 11.3.7.2.1 needs modification:

When ~~downloading the CVC Root CA Public Key and/or CA Certificates (e.g., a CVC CA Certificate and/or a Manufacturer CA Certificate)~~certificates are downloaded as a part of the Code File, the certificates MAY be contained in the ~~RootCAPublicKey field and/or the CACerts~~ fields ~~respectively~~ as specified in the Table 45 ~~respectively,~~ and separated from the actual code image contained in the CodeImage field.

Table 45 Code File Structure

Code File	**Description**
PKCS#7 Digital Signature {	
ContentInfo	
ContentType	SignedData
SignedData ()	EXPLICT signed-data content value: includes CVS and ITU-T X.509 compliant CVSs
} end PKCS#7 Digital Signature	
SignedContent {	
Download Parameters {	Mandatory TLV format (Type 28). (Length is zero if there is no sub-TLVs).
MfgCACerts ()	Optional TLV for one or more DER-encoded certificate(s) each formatted according to the Manufacturer CA-Certificate TLV format (Type 17).
clabServProvRootCACert ()	Optional TLV for one DER-encoded certificate formatted according to the Service Provider Root CA-Certificate TLV Format (Type 50).
clabCVCRootCACert ()	Optional TLV for one DER-encoded certificate formatted according to the CVC Root CA CA-Certificate TLV Format (Type 51).
clabCVCCACertificate ()	Optional TLV for one DER-encoded certificate formatted according to the CVC CA-Certificate TLV Format (Type 52).
}	
CodeImage ()	Upgrade code image.
} end SignedContent	

83. The changes appropriate for section 11.3.7.2.1.2 are:

The signed content field of the code file contains the code image and the download parameters field, which possibly contains additional optional items~~ – a CVC Root CA Public Key and CA Certificates (e.g., a~~ Service Provider Root CA Certificate,Certification Testing Labortory (CTL) CVC Root CA Certificate,CTL CVC CA

Certificate, and/or ~~a~~the Manufacturer CA Certificate~~)~~.

AND

If included in the signed content field, ~~the CVC Root CA Public Key~~a certificate is intended to replace the ~~CVC Root CA Public Key~~ certificate currently stored in the PS element. If the code download and installation is successful, then the PS element MUST replace its currently stored ~~CVC Root CA Public Key~~ certificate with the ~~CVC Root CA Public Key~~ new certificate received in the signed content field. This new ~~CVC Root CA Public key~~ certificate will then be used for subsequent ~~CVC~~ verification.

~~If included in the signed content field, the CA Certificate(s) is intended to replace the CA Certificate(s) currently stored in the PS element. For example, if the code download and installation is successful and the CACert contained a Manufacturer CA Certificate, then the PS element MUST replace its currently stored Manufacturer Certificate(s) with the Manufacturer Certificate(s) received in the signed content field.~~

84. The changes for section 11.3.7.2.1.3 are:

The ~~PKCS#7~~[RFC 2315] digital signature uses the RSA Encryption Algorithm [~~RSA 3~~RFC 2437] with SHA-1 [FIPS 186]. The ~~RSA key modulus for code signing is 2048 bits in length. The~~ PS element MUST be able to verify code file signatures ~~that are signed using this modulus size~~. The public exponent is F_4 (65537 decimal).

85. Add four new sections:

11.3.7.2.1.4 Manufacturer CA-Certificate

This Attribute is a string attribute containing an X.509 CA Certificate, as defined in ITU-T X.509.

Type	Length	Value
17	Variable	X.509 CA Certificate (DER-encoded ASN.1)

11.3.7.2.1.5 Service Provider Root CA-Certificate

This Attribute is a string attribute containing an X.509 Service Provider Root CA Certificate, as defined in ITU-T X.509. This certificate must be used by the PS Element in SNMP provisioning mode for mutual authentication.

Type	Length	Value
50	Variable	X.509 CA Certificate (DER-encoded ASN.1)

11.3.7.2.1.6 CVC Root CA CA-Certificate

This Attribute is a string attribute containing an X.509 CVC Root CA Certificate, as defined in ITU-T X.509. This certificate must be used by the standalone PS Element in the secure software downloading process.

Type	Length	Value
51	Variable	X.509 CA Certificate (DER-encoded ASN.1)

11.3.7.2.1.7 CVC CA-Certificate

This Attribute is a string attribute containing an X.509 CVC CA Certificate, as defined in ITU-T X.509. This certificate must be used by the standalone PS Element in the secure software downloading process.

Type	**Length**	**Value**
52	Variable	X.509 CA Certificate (DER-encoded ASN.1)

86. Section 11.3.7.3.1 needs one small addition:

For secure software download, the format used for the CVC is ITU-T X.509 compliant. However, the X.509 structure has been restricted to ease the processing a PS element does to validate the certificate and extract the public key used to verify the CVS. The CVC MUST be DER encoded and exactly match the structure shown in Table 47 except for any change in order required to DER encode (e.g., the ordering of SET OF attributes). The PS element SHOULD reject the CVC if it does not match the DER encoded structure represented in Table 47 The DER encoding MUST meet the requirements of Section 11.3.2 IPCable2Home Public Key Infrastructure (PKI).

87. Add the followisng to the end of section -11.3.7.5.1 Manufacturer Initialization;

The Manufacturer MUST initialize the following certificates into the Standalone PS Element's non-volatile memory:

- Service Provider Root CA Certificate
- CVC Root CA Certificate
- CVC CA Certificate
- Manufacturer CA Certificate
- PS Element Certificate

The Manufacturer MUST initialize the following certificates into the Embedded PS Element's non-volatile memory:

- Service Provider Root CA Certificate
- Manufacturer CA Certificate
- PS Element Certificate

88. Make the following changes near the end of section 11.3.7.6.1.

3. Validate the CVC issuer signature using the CTL CVC CA Public Key held by the PS element.

4. Validate the CTL CVC CA signature using the CTL CVC Root CA Public Key held by the PS element. Verification of the signature will authenticate the source and validate trust in the CVC parameters.

~~4.~~ 5. Update the PS element's current value of cvcAccessStart corresponding to the CVC's subject organizationName (i.e., manufacturer or co-signer) with the validity start time value from the validated CVC. If the validity start time value is greater than the PS element's current value of codeAccessStart, update the PS element's codeAccessStart value with the validity start time value. The PS element SHOULD discard any remnants of the co-signer CVC.

89. Make the following changes near the end of section 11.3.7.6.2

3. Validate the CVC issuer signature using the CTL CVC CA Public Key held by the PS element.

4. Validate the CVC issuer signature using the CTL CVC Root CA Public Key held by the PS element.

Verification of the signature will authenticate the certificate and confirm trust in the CVC's validity start time.

5. Update the current value of the subject's cvcAccessStart values with the validated CVC's validity start time value. If the validity start time value is greater than the PS element's current value of codeAccessStart, update the PS element's codeAccessStart value with the validity start value.~~ All certificate parameters EXCEPT for the validity start time are no longer needed and SHOULD be discarded.~~

90. Modifications to section 11.3.7.7.1 are:

Code Verification Certificates (CVCs) are signed and issued by the Certification Testing Laboratory (CTL) CVC CA. The CVC MUST be exactly as specified in ~~section~~Section 11.3.7.3. The CTL CVC CA MUST not sign any CVC unless it is identical to the format specified in ~~that section.~~Section 11.3.7.3. Before signing a CVC, the CTL CVC CA MUST verify that the certificate request is authentic.

The CTL CVC CA will be responsible for registering names of authorized CVC subscribers. CVC Subscribers include PS Element manufacturers and operator's that will co-sign code images. It is the responsibility of the CTL CVC CA to guarantee that the organization name of every CVC Subscriber is different. The following guidelines MUST be enforced when assigning organization names for code file co- signers:

AND

~~To conserve storage space, the PS element MAY internally represent the code co-signer's name in an~~In any alternate format ~~as long as~~ all the information ~~is~~MUST be maintained and the original format ~~can~~MUST be reproduced; e.g., as a 32-bit nonzero integer, with an integer value of 0 representing the absence of a code-signer.

91. Modify as described below:

11.3.7.7.1.2 Operator Requirements

When an operator receives software upgrade code files from a manufacturer the operator ~~Should~~should validate the code image using the CTL CVC CA Public Key. This will allow the operator to verify that the code image is as built by the trusted manufacturer. The operator can re-verify the code file at any time by repeating the process.

If an operator wants to exercise the option of co-signing the code image destined for a device on their network, the operator MUST obtain a valid CVC from the CTL CVC CA.

When signing a code file, the operator MUST co-sign the file content according to the PKCS#7 signature standard, and include their operator CVC as defined in Section 11.3.7.2.1.1. This ~~application~~Recommendation does not require an operator to co-sign code files; but when the operator follows all the rules defined in this specification for preparing a code file, the PS element MUST accept it.

92. Make the following change to section 11.3.7.9

3. The PS element MUST validate the certificate signature using the CTL CVC CA Public Key held by the PS element. In turn, the CTL CVC CA Certificate signature is validated by the CTL CVC Root CA Public Key held by the PS element. Verification of the signature will authenticate the source of the public code verification key (CVK) and confirm trust in the key.~~ Once trust has been established in the manufacturer's CVK, the remaining certificate parameters EXCEPT for the validity start time are no longer needed and SHOULD be discarded.~~

AND

c. The PS element MUST validate the certificate signature using the CTL CVC CA Public Key held by the PS element. In turn, the CTL CVC CA certificate signature is validated by the ~~CVC~~ CTLCVC Root CA Public Key held by the PS element. Verification of the signature will authenticate the source of the co- signer's public code verification key (CVK) and confirm trust in the key~~. Once trust has been established in the co-signer's CVK, the remaining certificate parameters EXCEPT for the validity start time are no longer needed and SHOULD be discarded~~.

93. Add new section:

11.3.9 Cryptographic Algorithms

11.3.9.1 SHA-1

The IPCable2Home implementation of SHA-1 MUST use the SHA-1 hash algorithm as defined in FIPS 180-1.

94. In section 12, change the name “Remote Connection Speed Test” to “Connection Speed Tool” and change the name “Remote Ping Test” to “Ping Tool” as the new names better describe what is being provided. This requires several changes scattered throughout the section.

95. In section 13 make the following change:

Provisioning of the cable modem ~~functionality~~element of an embedded PS is separate and distinct from PS provisioning, and is out of scope ~~or~~for this Recommendation. The reader is referred to ~~DOCSIS specifications~~cable modem Recommendations for descriptions of cable modem provisioning.

96. In section 13.1 make the following changes:

the PS CDC, during the DHCPOFFER phase of the initialization process. DHCP Provisioning Mode is intended to enable the PS to operate on ~~an~~a J.112 infrastructure ~~that does not include advanced IPCablecom features~~with little or no changes to the DOCSIS network.

SNMP Provisioning Mode in the PS is triggered when the DHCP server in the cable network does NOT provide values for 'siaddr' and 'file', and when the cable network DHCP server DOES send DHCP option 177 sub-option 51. SNMP Provisioning Mode is intended to enable the PS to take advantage of advanced features of a IPCablecom infrastructure.

Not all error conditions are shown in Figure 13-2. Refer to Section 7.2.3.3 for a description of PS behavior in the event of incorrect Provisioning Mode decision criteria.

97. Section 13.2 needs the following changes:

Section 7.2.2.2.2 describes ~~two~~three WAN Address Modes supported for the acquisition of IP addresses by the PS from the DHCP server in the cable network.

If the PS makes the determination that it is to operate in DHCP Provisioning Mode, it will use the PS Configuration File information passed in the DHCP message as a trigger to download the PS Configuration

File, as described in Section ~~.~~7.2 PS Configuration File download is a requirement for the PS operating in DHCP Provisioning Mode but is optional for the PS operating in SNMP Provisioning Mode. ~~After the initial PS Configuration File download triggered by the DHCP message fields, the NMS may initiate post-provisioning configuration by issuing an SNMP Set-request to cabhPsDevProvConfigHash and cabhPsDevProvConfigFile MIB object as described in Section 7.3.~~

In DHCP Provisioning Mode the PS (CMP) defaults to using NmAccess mode for management message exchange with the NMS, but the NMS can optionally configure the CMP for Coexistence ~~mode~~Mode. These management messaging modes are described in Section 6.3.3.

Figure 40 and Table 48 describe the sequence of messages needed to initialize a PS operating in DHCP Provisioning Mode. The process for provisioning ~~for the~~ a PS operating in DHCP Provisioning Mode is the same for the PS embedded with a cable modem as it is for the stand-alone PS. The provisioning for the Embedded PS MUST NOT occur before the cable modem provisioning process. The stand-alone PS management provisioning SHOULD occur immediately after power-up/reset.

98. Table 48 requires the following modifications:

Flow Step	PS WAN-Man Provisioning: DHCP Provisioning Mode	Normal Sequence	Failure Sequence
CHPSWMD-1	DHCP Broadcast Discover The CDP (CDC) MUST send a broadcast DHCP DISCOVER message~~.~~ to acquire the WAN-Man IP address as described in Section 7.2.3. The DHCP DISCOVER broadcast by the CDP (CDC) MUST include mandatory options listed in Table ~~21.~~ 7-7. The PS MUST start the Provisioning Timer using the starting value accessible via cabhPsDevProvTimer AND set cabhPsDevProvState to status 'inProgress' (2) when the CDC sends a broadcast DHCP DISCOVER.	Begin provisioning sequence.	If unsuccessful per DHCP protocol report an error and continue to retry DHCP Broadcast Discover until successful (return to step CHPSWMD-1). ~~After 5 retries~~If unsuccessful on the first attempt to acquire a WAN-Man IP address, the PS initiates operation of the CDS as specified in Section 7.2.3.3.
CHPSWMD-2	DHCP OFFER The DHCP OFFER issued by the DHCP server in the cable network is expected to include no IPCable2Home option code 177 with sub-~~option~~options 3, 6, and 51 AND is expected to include PS configuration file information in the siaddr and file fields of the DHCP message. ~~The PS modifies cabhPsDevProvMode based on information received in the DHCP OFFER~~ (ref.: Section 7.2.3.3)~~.~~	CHPSWMD-2 MUST occur after CHPSWMD-1 completion.	If failure per DHCP protocol return to CHPSWMD-1 and report an error.
CHPSWMD-4	DHCP ACK The DHCP server sends the CDP a DHCP ACK message which contains the IPv4 address of the PS. The PS modifies cabhPsDevProvMode based on information received in the DHCP ACK (ref.: Section 7.2.3.3). The PS MUST store the Time of Day server address in cabhPsDevTimeServerAddr.	CHPSWMD-4 MUST occur after CHPSWMD-3 completion.	If failure per DHCP protocol return to CHPSWMD-1 and report an error. If the expected configuration file information is not

			received in the DHCP ACK after 5 attempts, the PS operates in “Dormant IPCable2Home mode” as described in Sections 5.5 and 7.2.3.3.
CHPSWMD-8	TFTP server sends PS Configuration File After the PS Configuration File is received, the hash ~~of the configuration file is calculated and compared to the value appended to the PS Configuration File name (ref. Section 7.3.3.3).~~is checked. Refer to Section 7.3.3.3. The PS Configuration File is then processed. Refer to Section 7.3.3 for PS Configuration File contents. Optionally, the IP Address~~/FQDN~~ of the firewall Configuration ~~File TFTP~~FileTFTP server, the firewall Configuration File filename~~,~~ and the hash of the firewall ~~configuration file, and the encryption key (if the firewall configuration file is encrypted)~~Configuration File are included in ~~the PS~~thePS Configuration File if there is a firewall Configuration File to be loaded, and this is the method selected to specify it.	CHPSWM-8 MUST occur after CHPSWM-7 completion.	If the TFTP download fails, report an error and return to CHPSWMD- 7 (continue to retry PS Configuration File download). If processing of the PS Configuration File produces an error, continue with CHPSWMD-9 and report the error as an event. If the Provisioning Timer expires before PS Configuration File is successfully downloaded, the PS MUST report an error and return to CHPSWMD- 1.
CHPSWMD-10	TFTP server sends firewall configuration file (optional) If step CHPSWMD-9 occurs, the TFTP Server sends the PS a TFTP Response containing the requested file. After the firewall configuration file is received the hash of the configuration file is calculated and compared to the value received in the PS Configuration File. ~~If encrypted, the file is decrypted.~~ The file is then processed. Refer to Section 11.3.5.	CHPSWMD-10 MUST occur after CHPSWMD-9 completion.	If the TFTP fails, continue with PS operation but report an error and continue to retry CHPSWMD-9. If processing of the firewall configuration file produces an error, continue and report the error as an event.

99. In section 13.3 make the following changes:

The following diagram illustrates message flows that are to be used to accomplish the provisioning of the PS when it operates in SNMP Provisioning Mode. The provisioning process for the PS WAN-Man interface is the same for the Embedded PS as it is for the Stand-alone PS. The Standalone PS provisioning SHOULD occur immediately after power-up/reset.

100. Table 49 needs the following changes:

Flow Step	PS WAN-Man Provisioning: SNMP Provisioning Mode	Normal Sequence	Failure Sequence
CHPSWMS- 1	DHCP Broadcast Discover The CDP (CDC) MUST send a broadcast DHCP DISCOVER message~~.~~ to acquire the WAN-Man IP address as described in Section 7.2.3. The DHCP DISCOVER broadcast by the CDP (CDC) MUST include mandatory options listed in Table ~~21.~~7-1. The PS MUST start the Provisioning Timer using the starting value accessible via cabhPsDevProvTimer AND set cabhPsDevProvState to status 'inProgress' (2) when the CDC sends a broadcast DHCP DISCOVER.	Begin provisioning sequence.	If failure per DHCP protocol report an error and continue to retry DHCP Broadcast Discover until successful (return to CHPSWMS-1). ~~After 5 retries the PS initiates~~If the first attempt to acquire an address lease from the Headend DHCP server fails, initiate operation of the CDS as specified in ~~Section~~Table 7.2.3.3.
CHPSWMS- 5	~~Time of Day (TOD)~~AS Request ~~per [RFC 868]~~[1] ~~The PS sends a ToD request to the address stored in cabhPsDevServerTime as required in Section 7.4.2~~The PS MUST send the AS Request message to the operator KDC to request a Kerberos ticket	CHPSWMS-5 MUST occur after CHPSWMS-4 completion.	~~Continue with~~Return to CHPSWMS-~~6.~~1. PS initiates operation of CDS.
CHPSWMS- 6	~~TOD Response~~AS Reply ~~The ToD server is expected to reply with the current time in UTC format.~~The AS Reply Message is received from the operator KDC containing the Kerberos ticket	CHPSWMS-6 MUST occur after CHPSWMS-5 completion.	~~Continue with CHPSWMS-7, report an error, and return to CHPSWMS-5 (continue to retry TOD until successful)~~Return to CHPSWMS-1. PS initiates operation of CDS.
CHPSWMS- 7	~~AS~~TGS Request~~1~~ ~~The~~If the PS obtained a Ticket Granting Ticket (TGT) during step CHPSWMS-6, the PS MUST send the ~~AS~~TGS Request message to the operator KDC server whose address was passed to ~~request a Kerberos ticket~~the PS (CDC) in DHCP Option 177 sub-option 51.	CHPSWMS-7 MUST occur after CHPSWMS-~~5 completion. CHPSWMS-7 MAY occur before CHPSWMS-~~6 completion.	Return to CHPSWMS- 1. PS initiates operation of CDS.
CHPSWMS- 8	~~AS~~TGS Reply The ~~AS~~TGS Reply ~~Message~~message containing the ticket is received from the operator KDC ~~containing the Kerberos ticket~~.	CHPSWMS-8 MUST occur after CHPSWMS-7 completion.	Return to CHPSWMS- 1. PS initiates operation of CDS.
CHPSWMS- 9	~~TGS~~AP Request ~~If PS obtained Ticket Granting Ticket (TGT) in PS WAN-Man Interface provisioning process step CHPSWMS-10, the TGS Request message MUST be sent to the operator KDC.~~The PS MUST send the AP Request message to the NMS (SNMP manager) to request keying information for SNMPv3, as described in Table 11.3.3.2.	CHPSWMS-9 MUST occur after CHPSWMS-8 completion.	Return to CHPSWMS- 1. PS initiates operation of CDS.

~~1 Steps CHPSWMS-7-CHPSWMS-10 are optional in some cases. Refer to Section 11 for details.~~

~~CHPSWMS-10~~	~~TGS Reply~~ ~~The TGS Reply message containing the ticket is received from the operator KDC.~~	~~CHPSWMS-10 MUST occur after CHPSWMS-9 completion.~~	~~Return to CHPSWMS-1.~~
~~CHPSWMS-11~~	~~AP Request~~ ~~The AP Request message MUST be sent to the Provisioning Server to request the keying information for SNMPv3.~~	~~CHPSWMS-11 MUST occur after CHPSWMS-10 completion.~~	~~Return to CHPSWMS-1.~~
CHPSWMS-~~12~~10	AP reply The AP Reply message is received from the ~~Provisioning Server~~NMS containing the keying information for SNMPv3.Note: The SNMPv3 keys MUST be established and the associated SNMPv3 tables populated before the next step. The keys and tables are established using the information in the AP Reply (see Section ~~11.4~~11.3 for additional detail.)	CHPSWMS-~~12~~10 MUST occur after CHPSWMS-~~11~~9 completion.	Return to CHPSWMS- 1. PS initiates operation of CDS.
CHPSWMS-~~17~~15	SNMP Get Response Iterative: The PS MUST reply to the NMS Get-request or Get-bulk request messages with a Get Response for each Get Request.After all the Gets, or the GetBulk, finish, the NMS sends the requested data to the provisioning application.	If CHPSWMS-~~17~~14 occurs, CHPSWMS-15 MUST occur after CHPSWMS-~~16 completion~~14 completes.	N/A
CHPSWMS-~~18~~16	Configuration File Create Optional: The provisioning system uses information from PS provisioning steps CHPSWMS-~~14~~12 and CHPSWMS-~~15~~13 to create a PS configuration file. The provisioning system runs a hash on the contents of the configuration file. The hash is sent to the PS in the next step.	If CHPSWMS-~~18~~15 occurs, CHPSWMS-16 MUST occur after CHPSWMS-~~17 completion~~15 completes.	N/A
CHPSWMS-~~19~~17	SNMP Set The provisioning system might instruct the NMS to send an SNMP Set message to the PS containing the IP Address of the TFTP server, the PS Configuration File filename and the hash of the configuration file as described in Section 7.3.3.2 (SNMP Provisioning Mode). Optionally, the IP Address of the Firewall Configuration File TFTP server, the Firewall Configuration File filename~~,~~ and the hash of the firewall ~~configuration file, and the encryption key (if the firewall configuration file is encrypted)~~Configuration File are included in the SNMP set if there is a firewall Configuration File to be loaded, and this method is selected to specify it.	If CHPSWMS-~~19~~16 occurs, CHPSWMS-17 MUST occur after CHPSWMS-~~18 completion~~16 completes.	Return to CHPSWMS- 1 if the set was received, but there was a processing error.
CHPSWMS-~~20~~18	TFTP Request If the NMS triggers the PS to download a PS Configuration File as described in Section 7.3.3.2, the PS MUST send the TFTP Server a TFTP Get Request to request the specified PS Configuration File.	If CHPSWMS-~~20~~17 occurs, CHPSWMS-18 MUST occur after CHPSWMS-~~19 completion~~17 completes.	Continue with CHPSWMS-~~21.~~19.

CHPSWMS-~~21~~19	TFTP server sends Configuration File After the PS receives the PS Configuration File, the PS calculates the hash of the PS Configuration File and compares it to the value received in step CHPSWMS-19. The PS then processes the PS Configuration File. Refer to Section 7.3.3 for PS Configuration File contents. Optionally, the IP Address~~/FQDN~~ of the Firewall Configuration File TFTP server, the Firewall Configuration File filename~~,~~, and the hash of the firewall configuration file~~, and the encryption key (if the firewall configuration file is encrypted)~~ are included in the PS Configuration File if there is a firewall Configuration File to be loaded, and this is the method selected to specify it.	If CHPSWMS-~~21~~18 occurs, CHPSWMS-19 MUST occur after CHPSWMS-~~20 completion~~18 completes.	If the TFTP download fails, report an error, proceed to CHPSWMS-~~23,~~20, and continue to retry CHPSWMS-~~20~~18 (continue to retry PS Configuration File download). If processing of the Configuration File produces an error, continue and report the error as an event.
CHPSWMS-~~22~~20	TFTP Request - Firewall Configuration File (optional) The PS sends the Firewall Configuration TFTP Server a TFTP Get Request to request the specified firewall configuration data file.	If CHPSWMS-~~22~~19 occurs, If CHPSWMS-20 occurs, it MUST occur after CHPSWMS-~~21 completion~~19 completes.	Return to CHPSWMS- 1.
CHPSWMS-~~23~~21	TFTP server sends Firewall Configuration File The TFTP Server sends the PS a TFTP Response containing the requested file. After the PS receives the Firewall Configuration File the PS calculates the hash of the Firewall Configuration File and compares it to the value received in step CHPSWMS-21. ~~If encrypted, the file is decrypted.~~ The file is then processed. Refer to Section ~~for description of PS configuration file contents~~11.3 for additional detail.	If CHPSWMS-~~23~~20 occurs, CHPSWMS-21 MUST occur after CHPSWMS-~~22 completion~~20 completes.	If the TFTP download fails, continue with PS operation but report an error and continue to retry CHPSWMS-~~22.~~20. If processing of the firewall configuration file produces an error, continue and report the error as an event.

Notes to Table 13-2:

1. Steps CHPSWMS-5-CHPSWMS-8 are optional in some cases. Refer to Section 11 for details.
2. The SNMP Get and following SNMP Get Response operations are optional, depending on whether additional information is required to form a PS Configuration File, and also depending on whether a PS Configuration File is needed.

101. Change section 13.3.3

For the PS operating in SNMP Provisioning Mode only, the provisioning enrollment inform~~, defined in Annex B,~~ (cabhPsDevProvEnrollTrap) enables the Provisioning Server to determine that the PS is ready for the PS Configuration File.

In either DHCP Provisioning Mode or SNMP Provisioning Mode the provisioning complete trap (cabhPsDevInitTrap)~~, defined in Annex B,~~ indicates whether the provisioning sequence has completed successfully or not.

102. Renumber section 13.4 to 13.3.4 and section 13.4.1 to 13.3.5. Section 13.5 becomes 13.4 etc.

103. Make the following addition to section 13.4 (renumbered)

The following diagrams illustrate the message flows that are to be used to accomplish the provisioning of PS

WAN-Data addresses. The provisioning process for the PS WAN-Data addresses is the same for the PS embedded with a cable modem as it is for the stand-alone PS.

104. Add the following note benith Table 51.

Notes to Table 51:

1. If the client is aware of its previous IP address (e.g., following reboot), it may omit the DHCPDISCOVER and proceed with step 3.
2. If the client is located on a non-broadcast network it is expected to unicast the message to the DHCP Server.
3. If the client is located on a non-broadcast network it is expected that it will unicast the message to the PS.

105. Add the following note benith Table 52

Notes to Table 13-7:

1. If the client is located on a non-broadcast network it must unicast the message to the DHCP Server or DHCP Relay Agent in the cable network.

106. The existing Annex A needs to be put into a better format and enhanced; Hence it is easier to replace the whole thing.

Annex A MIB Objects

This annex lists all required MIB objects, as indicated in Section 6.3.7.

MIB NAME/Parameter	Max-Access	Persistent	# of Persistent Entries
mib-2			
system			
sysDescr	read-only	N/A	N/A
sysObjectID	read-only	N/A	N/A
sysUpTime	read-only	N/A	N/A
sysContact	read-write	Yes	1
sysName	read-write	Yes	1
sysLocation	read-write	Yes	1
sysServices	read-only	N/A	N/A
interfaces [RFC 2863]			
ifNumber	read-only	N/A	N/A
ifTable/ifEntry			
ifIndex	read-only	N/A	N/A
ifDescr	read-only	N/A	N/A
ifType	read-only	N/A	N/A
ifMtu	read-only	N/A	N/A
ifSpeed	read-only	N/A	N/A
ifPhysAddress	read-only	N/A	N/A
ifAdminStatus	read-write	N/A	N/A
ifOperStatus	read-only	N/A	N/A
ifLastChange	read-only	N/A	N/A
ifInOctets	read-only	N/A	N/A
ifInUcastPkts	read-only	N/A	N/A
ifInDiscards	read-only	N/A	N/A
ifInErrors	read-only	N/A	N/A
ifInUnknownProtos	read-only	N/A	N/A
ifOutOctets	read-only	N/A	N/A
ifOutUcastPkts	read-only	N/A	N/A
ifOutDiscards	read-only	N/A	N/A
ifOutErrors	read-only	N/A	N/A

	ip [RFC 2011]		
ipForwarding	read-write	No	N/A
ipDefaultTTL	read-write	No	N/A
ipInReceives	read-only	N/A	N/A
ipInHdrErrors	read-only	N/A	N/A
ipInAddrErrors	read-only	N/A	N/A
ipForwDatagrams	read-only	N/A	N/A
ipInUnknownProtos	read-only	N/A	N/A
ipInDiscards	read-only	N/A	N/A
ipInDelivers	read-only	N/A	N/A
ipOutRequests	read-only	N/A	N/A
ipOutDiscards	read-only	N/A	N/A
ipOutNoRoutes	read-only	N/A	N/A
ipReasmTimeout	read-only	N/A	N/A
ipReasmReqds	read-only	N/A	N/A
ipReasmOKs	read-only	N/A	N/A
ipReasmFails	read-only	N/A	N/A
ipFragOKs	read-only	N/A	N/A
ipFragFails	read-only	N/A	N/A
ipFragCreates	read-only	N/A	N/A
ipNetToMediaTable/ipNetToMediaEntry			
ipNetToMediaIfIndex	read-create	No	N/A
ipNetToMediaPhyAddress	read-create	No	N/A
ipNetToMediaNetAddress	read-create	No	N/A
ipNetToMediaType	read-create	No	N/A
	icmp		
icmpInMsgs	read-only	N/A	N/A
icmpInErrors	read-only	N/A	N/A
icmpInDestUnreachs	read-only	N/A	N/A
icmpInTimeExcds	read-only	N/A	N/A
icmpInParmProbs	read-only	N/A	N/A
icmpInSrcQuenchs	read-only	N/A	N/A
icmpInRedirects	read-only	N/A	N/A
icmpInEchos	read-only	N/A	N/A
icmpInEchosReps	read-only	N/A	N/A

icmpInTimestamps	read-only	N/A	N/A
icmpInTimestampsReps	read-only	N/A	N/A
icmpInAddrMasks	read-only	N/A	N/A
icmpInAddrMaskReps	read-only	N/A	N/A
icmpOutMsgs	read-only	N/A	N/A
icmpOutErrors	read-only	N/A	N/A
icmpOutDestUnreachs	read-only	N/A	N/A
icmpOutTimeExcds	read-only	N/A	N/A
icmpOutParmProbs	read-only	N/A	N/A
icmpOutSrcQuenchs	read-only	N/A	N/A
icmpOutRedirects	read-only	N/A	N/A
icmpOutEchos	read-only	N/A	N/A
icmpOutEchosReps	read-only	N/A	N/A
icmpOutTimestamps	read-only	N/A	N/A
icmpOutTimestampReps	read-only	N/A	N/A
icmpOutAddrMasks	read-only	N/A	N/A
icmpOutAddrMaskReps	read-only	N/A	N/A
udp [RFC 2013]			
udpInDatagrams	read-only	N/A	N/A
udpNoPorts	read-only	N/A	N/A
udpInErrors	read-only	N/A	N/A
udpOutDatagrams	read-only	N/A	N/A
udpTable/udpEntry			
udpLocalAddress	read-only	N/A	N/A
udpLocalPort	read-only	N/A	N/A
transmission [draft-ietf-ipcdn-bpiplus-mib-05]			
docsIfMib			
docsBpi2MIB			
docsBpi2MIBObjects			
docsBpi2CmObjects			
docsBpi2CmCertObjects			
docsBpi2CmDeviceCertTable/docsBpi2CmDeviceCertEntry			
docsBpi2CmDeviceCmCert	read-write	Yes	5
docsBpi2CmDeviceManufCert	read-only	N/A	N/A

docsBpi2CodeDownloadGroup			
docsBpi2CodeDownloadStatusCode	read-only	N/A	N/A
docsBpi2CodeDownloadStatusString	read-only	N/A	N/A
docsBpi2CodeMfgOrgName	read-only	N/A	N/A
docsBpi2CodeMfgCodeAccessStart	read-only	N/A	N/A
docsBpi2CodeMfgCvcAccessStart	read-only	N/A	N/A
docsBpi2CodeCoSignerOrgName	read-only	N/A	N/A
docsBpi2CodeCoSignerCodeAccessStart	read-only	N/A	N/A
docsBpi2CodeCoSignerCvcAccessStart	read-only	N/A	N/A
docsBpi2CodeCvcUpdate	read-write	Yes	1
snmp [RFC 1905]			
snmpInPkts	read-only	N/A	N/A
snmpInBadVersions	read-only	N/A	N/A
snmpInBadCommunityNames	read-only	N/A	N/A
snmpInBadCommunityUses	read-only	N/A	N/A
snmpInASNParseErrs	read-only	N/A	N/A
snmpEnableAuthenTraps	read-write	No	N/A
snmpSilentDrops	read-only	N/A	N/A
ifMIB [RFC 2863]			
ifMIBOjects			
ifXTable/ifXEntry			
ifName	read-only	N/A	N/A
ifInMulticastPkts	read-only	N/A	N/A
ifInBroadcastPkts	read-only	N/A	N/A
ifOutMulticastPkts	read-only	N/A	N/A
ifOutBroadcastPkts	read-only	N/A	N/A
ifLinkUpDownTrapEnable	read-write	No	N/A
ifHighSpeed	read-only	N/A	N/A
ifPromiscuousMode	read-write	N/A	N/A
ifConnectorPresent	read-only	N/A	N/A
ifAlias	read-write	No	N/A
ifCounterDiscontinuitityTime	read-only	N/A	N/A

	docsDev [RFC 2669]		
	docsDevMIBObjects		
docsDevNmAccessTable/docsDevNmAccessEntry			
docsDevNmAccessIndex	not-accessible	N/A	N/A
docsDevNmAccessIp	read-create	No	N/A
docsDevNmAccessIpMask	read-create	No	N/A
docsDevNmAccessCommunity	read-create	No	N/A
docsDevNmAccessControl	read-create	No	N/A
docsDevNmAccessInterfaces	read-create	No	N/A
docsDevNmAccessStatus	read-create	No	N/A
docsDevNmAccessTrapVersion	read-create	No	N/A
	docsDevSoftware		
docsDevSwServer	read-write	Yes	1
docsDevSwFilename	read-write	Yes	1
docsDevSwAdminStatus	read-write	No	1
docsDevSwOperStatus	read-only	N/A	N/A
docsDevSwCurrentVers	read-only	N/A	N/A
	docsDevEvent		
docsDevEvControl	read-write	No	N/A
docsDevEvSyslog	read-write	No	N/A
docsDevEvThrottleAdminStatus	read-write	No	N/A
docsDevEvThrottleInhibited	read-only	N/A	N/A
docsDevEvThrottleThreshold	read-write	No	N/A
docsDevEvThrottleInterval	read-write	No	N/A
docsDevEvControlTable/docsDevEvControlEntry			
docsDevEvPriority	not-accessible	N/A	N/A
docsDevEvReporting	read-write	No	N/A
docsDevEventTable/docsDevEventEntry			
docsDevEvIndex	not-accessible	N/A	N/A
docsDevEvFirstTime	read-only	Yes	1
docsDevEvLastTime	read-only	Yes	1
docsDevEvCounts	read-only	Yes	1

docsDevEvLevel	read-only	Yes	1
docsDevEvId	read-only	Yes	1
docsDevEvText	read-only	Yes	1
private			
enterprises			
cableLabs			
clabProject			
clabProjCableHome			
cabhPsDevMib			
cabhPsDevBase			
cabhPsDevDateTime	read-write	No	N/A
cabhPsDevResetNow	read-write	No	N/A
cabhPsDevSerialNumber	read-only	N/A	N/A
cabhPsDevHardwareVersion	read-only	N/A	N/A
cabhPsDevWanManMacAddress	read-only	N/A	N/A
cabhPsDevWanDataMacAddress	read-only	N/A	N/A
cabhPsDevTypeIdentifier	read-only	N/A	N/A
cabhPsDevSetToFactory	read-write	No	N/A
cabhPsDevTodSyncStatus	read-only	N/A	N/A
cabhPsDevProvMode	read-only	N/A	N/A
cabhPsDevProv			
cabhPsDevProvisioningTimer	read-write	No	N/A
cabhPsDevProvConfigFile	read-write	No	N/A
cabhPsDevProvConfigHash	read-write	No	N/A
cabhPsDevProvConfigFileSize	read-only	N/A	N/A
cabhPsDevProvConfigFileStatus	read-only	N/A	N/A
cabhPsDevProvConfigTLVProcessed	read-only	N/A	N/A
cabhPsDevProvConfigTLVRejected	read-only	N/A	N/A
cabhPsDevProvSolicitedKeyTimeout	read-write	Yes	1
cabhPsDevProvState	read-only	N/A	N/A
cabhPsDevProvAuthState	read-only	N/A	N/A
cabhPsDevTimeServerAddrType	read-only	N/A	N/A
cabhPsDevTimeServerAddr	read-only	N/A	N/A

	cabhSecMib		
	cabhSecFwObjects		
	cabhSecFwBase		
cabhSecFwPolicyFileEnable	read-write	No	N/A
cabhSecFwPolicyFileURL	read-write	Yes	1
cabhSecFwPolicyFileHash	read-write	No	N/A
cabhSecFwPolicyFileOperStatus	read-only	N/A	N/A
cabhSecFwPolicyFileCurrentVersion	read-only	N/A	N/A
	cabhSecFwLogCtl		
cabhSecFwEventType1Enable	read-write	No	N/A
cabhSecFwEventType2Enable	read-write	No	N/A
cabhSecFwEventType3Enable	read-write	No	N/A
cabhSecFwEventAttackAlertThreshold	read-write	No	N/A
cabhSecFwEventAttackAlertPeriod	read-write	No	N/A
cabhSecCertObjects			
cabhSecCertPsCert	read-only	Yes	1
	cabhCapMib		
	cabhCapObjects		
	cabhCapBase		
cabhCapTcpTimeWait	read-write	Yes	1
cabhCapUdpTimeWait	read-write	Yes	1
cabhCapIcmpTimeWait	read-write	Yes	1
cabhCapPrimaryMode	read-write	No	N/A
cabhCapSetToFactory	read-write	No	N/A
	cabhCapMap		
cabhCapMappingTable/cabhCapMappingEntry			
cabhCapMappingIndex	not-accessible	Yes[1]	16
cabhCapMappingWanAddrType	read-create	Yes[1]	16
cabhCapMappingWanAddr	read-create	Yes[1]	16
cabhCapMappingWanPort	read-create	Yes[1]	16
cabhCapMappingLanAddrType	read-create	Yes[1]	16
cabhCapMappingLanAddr	read-create	Yes[1]	16
cabhCapMappingLanPort	read-create	Yes[1]	16

cabhCapMappingMethod	read-only	N/A	16
cabhCapMappingProtocol	read-create	Yes[1]	16
cabhCapMappingRowStatus	read-create	Yes	16
cabhCapPassthroughTable/cabhCapPassthroughEntry			
cabhCapPassthroughIndex	not-accessible	Yes	16
cabhCapPassthroughMACAddr	read-create	Yes	16
cabhCapPassthroughRowStatus	read-create	Yes	16

1. cabhCapMappingEntry objects are persistent if provisioned by the NMS and non-persistent if created dynamically based on outbound traffic. Refer to Section 8.3.2.2.

cabhCdpMib

cabhCdpObjects

cabhCdpBase

cabhCdpSetToFactory	read-write	No	N/A
cabhCdpLanTransCurCount	read-only	N/A	N/A
cabhCdpLanTransThreshold	read-write	No	N/A
cabhCdpLanTransAction	read-write	No	N/A
cabhCdpWanDataIpAddrCount	read-write	No	N/A

cabhCdpAddr

cabhCdpLanAddrTable/cabhCdpLanAddrEntry			
cabhCdpLanAddrIpType	not-accessible	Yes	16
cabhCdpLanAddrIp	not-accessible	Yes	16
cabhCdpLanAddrClientID	read-create	Yes	16
cabhCdpLanAddrLeaseCreateTime	read-only	Yes	16
cabhCdpLanAddrLeaseExpireTime	read-only	Yes	16
cabhCdpLanAddrMethod	read-only	Yes	16
cabhCdpLanAddrHostName	read-only	Yes	16
cabhCdpLanAddrRowStatus	read-create	Yes	16
cabhCdpWanDataAddrTable/cabhCdpWanDataAddrEntry			
cabhCdpWanDataAddrIndex	not-accessible	N/A	N/A
cabhCdpWanDataAddrClientId	read-create	No	N/A
cabhCdpWanDataAddrIpType	read-only	N/A	N/A
cabhCdpWanDataAddrIp	read-only	N/A	N/A
cabhCdpWanDataAddrRenewalTime	read-only	N/A	N/A
cabhCdpWanDataAddrRowStatus	read-create	No	N/A
cabhCdpWanDataAddrSeverTable/cabhCdpWanDataAddrSeverEntry			

cabhCdpWanDataAddrDnsIpType	not-accessible	N/A	N/A
cabhCdpWanDataAddrDnsIp	not-accessible	N/A	N/A
cabhCdpWanDataAddrDnsRowStatus	read-create	No	N/A
	cabhCdpServer		
cabhCdpLanPoolStartType	read-write	Yes	1
cabhCdpLanPoolStart	read-write	Yes	1
cabhCdpLanPoolEndType	read-write	Yes	1
cabhCdpLanPoolEnd	read-write	Yes	1
cabhCdpServerNetworkNumberType	read-write	Yes	1
cabhCdpServerNetworkNumber	read-write	Yes	1
cabhCdpServerSubnetMaskType	read-write	Yes	1
cabhCdpServerSubnetMask	read-write	Yes	1
cabhCdpServerTimeOffset	read-write	Yes	1
cabhCdpServerRouterType	read-write	Yes	1
cabhCdpServerRouter	read-write	Yes	1
cabhCdpServerDnsAddressType	read-write	Yes	1
cabhCdpServerDnsAddress	read-write	Yes	1
cabhCdpServerSyslogAddressType	read-write	Yes	1
cabhCdpServerSyslogAddress	read-write	Yes	1
cabhCdpServerDomainName	read-write	Yes	1
cabhCdpServerTTL	read-write	Yes	1
cabhCdpServerInterfaceMTU	read-write	Yes	1
cabhCdpServerVendorSpecific	read-write	Yes	1
cabhCdpServerLeaseTime	read-write	Yes	1
cabhCdpServerDhcpAddressType	read-write	Yes	1
cabhCdpServerDhcpAddress	read-write	Yes	1
	cabhCtpMib		
	cabhCtpObjects		
	cabhCtpBase		
cabhCtpSetToFactory	read-write	No	N/A
	cabpCtpConnSpeed		
cabhCtpConnSrcIpType	read-write	No	N/A
cabhCtpConnSrcIp	read-write	No	N/A
cabhCtpConnDestIpType	read-write	No	N/A

cabhCtpConnDestIp	read-write	No	N/A
cabhCtpConnProto	read-write	No	N/A
cabhCtpConnNumPkts	read-write	No	N/A
cabhCtpConnPktSize	read-write	No	N/A
cabhCtpConnTimeOut	read-write	No	N/A
cabhCtpConnControl	read-write	No	N/A
cabhCtpConnStatus	read-only	N/A	N/A
cabhCtpConnPktsSent	read-only	N/A	N/A
cabhCtpConnPktsRecv	read-only	N/A	N/A
cabhCtpConnRTT	read-only	N/A	N/A
cabhCtpConnThroughput	read-only	N/A	N/A
	cabhCtpPing		
cabhCtpPingSrcIpType	read-write	No	N/A
cabhCtpPingSrcIp	read-write	No	N/A
cabhCtpPingDestIpType	read-write	No	N/A
cabhCtpPingDestIp	read-write	No	N/A
cabhCtpPingNumPkts	read-write	No	N/A
cabhCtpPingPktSize	read-write	No	N/A
cabhCtpPingTimeBetween	read-write	No	N/A
cabhCtpPingTimeOut	read-write	No	N/A
cabhCtpPingControl	read-write	No	N/A
cabhCtpPingStatus	read-only	N/A	N/A
cabhCtpPingNumSent	read-only	N/A	N/A
cabhCtpPingNumRecv	read-only	N/A	N/A
cabhCtpPingAvgRTT	read-only	N/A	N/A
cabhCtpPingMinRTT	read-only	N/A	N/A
cabhCtpPingMaxRTT	read-only	N/A	N/A
cabhCtpPingNumIcmpError	read-only	N/A	N/A
cabhCtpPingIcmpError	read-only	N/A	N/A
	experimental		
	snmpUSMDHObjectsMIB [RFC 2786]		
	usmDHKeyObjects		
	usmDHPublicObjects		
usmDHPParamaters	read-write	No	N/A
usmDHUserKeyTable/usmDHUserKeyEntry			

usmDHUserAuthKeyChange	read-create	No	N/A
usmDHUserOwnAuthKeyChange	read-create	No	N/A
usmDHUserPrivKeyChange	read-create	No	N/A
usmDHUserOwnPrivKeyChange	read-create	No	N/A
usmDHKickstartGroup			
usmDHKickstartTable/usmDHKickstartEntry			
usmDHKickstartIndex	not-accessible	No	N/A
usmDHKickstartMyPublic	read-only	N/A	N/A
usmDHKickstartMgrPublic	read-only	N/A	N/A
usmDHKickstartSecurityName	read-only	N/A	N/A
snmpV2			
snmpModules			
snmpMIB			
snmpMIBObjects			
snmpSet			
snmpSetSerialNo	read-write	No	N/A
snmpFrameworkMIB [RFC 2571]			
snmpEngine			
snmpEngineID	read-only	N/A	N/A
snmpEngineBoots	read-only	Yes	1
snmpEngineTime	read-only	N/A	N/A
snmpEngineMaxMessageSize	read-only	N/A	N/A
snmpMPDMIB [RFC 2572]			
snmpMPDObjects			
snmpMPDStats			
snmpUnknownSecurityModels	read-only	N/A	N/A
snmpInvalidMsgs	read-only	N/A	N/A
snmpUnknownPDUHandlers	read-only	N/A	N/A
snmpTargetMIB [RFC 2573]			
snmpTargetObjects			
snmpTargetSpinLock	read-write	No	N/A
snmpTargetAddrTable/snmpTargetAddrEntry			

snmpTargetAddrName	not-accessible	No	N/A
snmpTargetAddrTDomain	read-create	No	N/A
snmpTargetAddrTAddress	read-create	No	N/A
snmpTargetAddrTimeout	read-create	No	N/A
snmpTargetAddrRetryCount	read-create	No	N/A
snmpTargetAddrTagList	read-create	No	N/A
snmpTargetAddrParams	read-create	No	N/A
snmpTargetAddrStorageType	read-create	No	N/A
snmpTargetAddrRowStatus	read-create	No	N/A
snmpTargetParamsTable/snmpTargetParamsEntry			
snmpTargetParamsName	not-accessible	No	N/A
snmpTargetParamsMPModel	read-create	No	N/A
snmpTargetParamsSecurityModel	read-create	No	N/A
snmpTargetParamsSecurityName	read-create	No	N/A
snmpTargetParamsSecurityLevel	read-create	No	N/A
snmpTargetParamsStorageType	read-create	No	N/A
snmpTargetParamsRowStatus	read-create	No	N/A
snmpUnavailableContexts	read-only	N/A	N/A
snmpUnknownContexts	read-only	N/A	N/A

snmpNotificationMIB [RFC 2573]
snmpNotifyObjects

snmpNotifyTable/snmpNotifyEntry			
snmpNotifyName	not-accessible	No	N/A
snmpNotifyTag	read-create	No	N/A
snmpNotifyType	read-create	No	N/A
snmpNotifyStorageType	read-create	No	N/A
snmpNotifyRowStatus	read-create	No	N/A
snmpNotifyFilterProfileTable/snmpNotifyFilterProfileEntry			
snmpNotifyFilterProfileName	read-create	No	N/A
snmpNotifyFilterProfileStorType	read-create	No	N/A
snmpNotifyFilterProfileRowStatus	read-create	No	N/A

snmpNotifyFilterTable/snmpNotifyFilterEntry			
snmpNotifyFilterSubtree	not-accessible	No	N/A
snmpNotifyFilterMask	read-create	No	N/A
snmpNotifyFilterType	read-create	No	N/A
snmpNotifyFilterStorageType	read-create	No	N/A
snmpNotifyFilterRowStatus	read-create	No	N/A
	snmpUsmMIB [RFC 2574]		
	usmStats		
usmStatsUnsupportedSecLevels	read-only	N/A	N/A
usmStatsNotInTimeWindows	read-only	N/A	N/A
usmStatsUnknownUserNames	read-only	N/A	N/A
usmStatsUnknownEngineIDs	read-only	N/A	N/A
usmStatsWrongDigests	read-only	N/A	N/A
usmStatsDecryptionErrors	read-only	N/A	N/A
	usmUser		
usmUserSpinLock	read-write	No	N/A
usmUserTable/usmUserEntry			
usmUserEngineID	not-accessible	N/A	N/A
usmUserName	not-accessible	N/A	N/A
usmUserSecurityName	read-only	N/A	N/A
usmUserCloneFrom	read-create	No	N/A
usmUserAuthProtocol	read-create	No	N/A
usmUserAuthKeyChange	read-create	No	N/A
usmUserOwnAuthKeyChange	read-create	No	N/A
usmUserPrivProtocol	read-create	No	N/A
usmUserPrivKeyChange	read-create	No	N/A
usmUserOwnPrivKeyChange	read-create	No	N/A
usmUserPublic	read-create	No	N/A
usmUserStorageType	read-create	No	N/A
usmUserStatus	read-create	No	N/A

SNMP-VIEW-BASED-ACM-MIB [RFC 2575]

snmpVacmMIB

vacmMIBObjects

vacmContextTable/vacmContextEntry			
vacmContextName	read-only	No	N/A
vacmSecurityToGroupTable/vacmSecurityToGroupEntry			
vacmSecurityModel	not-accessible	No	N/A
vacmSecurityName	not-accessible	No	N/A
vacmGroupName	read-create	No	N/A
vacmSecurityToGroupStorageType	read-create	No	N/A
vacmSecurityToGroupStatus	read-create	No	N/A
vacmAccessTable/vacmAccessEntry			
vacmAccessContextPrefix	not-accessible	No	N/A
vacmAccessSecurityModel	not-accessible	No	N/A
vacmAccessSecurityLevel	not-accessible	No	N/A
vacmAccessContextMatch	read-create	No	N/A
vacmAccessReadViewName	read-create	No	N/A
vacmAccessWriteViewName	read-create	No	N/A
vacmAccessNotifyViewName	read-create	No	N/A
vacmAccessStorageType	read-create	No	N/A
vacmAccessStatus	read-create	No	N/A

vacmMIBViews

vacmViewSpinLock	read-write	No	N/A
vacmViewTreeFamilyTable/vacmViewTreeFamilyEntry			
vacmViewTreeFamilyViewName	not-accessible	No	N/A
vacmViewTreeFamilySubtree	not-accessible	No	N/A
vacmViewTreeFamilyMask	read-create	No	N/A
vacmViewTreeFamilyType	read-create	No	N/A
vacmViewTreeFamilyStorageType	read-create	No	N/A
vacmViewTreeFamilyStatus	read-create	No	N/A

snmpCommunityMIB [RFC 2576]

snmpCommunityMIBObjects

snmpCommunityTable/snmpCommunityEntry			
snmpCommunityIndex	not-accessible	No	N/A

snmpCommunityName	read-create	No	N/A
snmpCommunitySecurityName	read-create	No	N/A
snmpCommunityContextEngineID	read-create	No	N/A
snmpCommunityContextName	read-create	No	N/A
snmpCommunityTransportTag	read-create	No	N/A
snmpCommunityStorageType	read-create	No	N/A
snmpCommunityStatus	read-create	No	N/A
snmpTargetAddrExtTable/snmpTargetAddrExtEntry			
snmpTargetAddrTMask	read-create	No	N/A
snmpTargetAddrMMS	read-create	No	N/A
clabSecCertObject			
clabSrvcPrvdrRootCACert	read-only	N/A	N/A
clabCVCRootCACert	read-only	N/A	N/A
clabCVCCACert	read-only	N/A	N/A
clabMfgCVCCert	read-only	N/A	N/A

107. In Table B1 of Annex B, many of the entries in the far right column (Trap Name) have a space in the name (done to look better, I suspect). There are no spaces in the Trap Names. The spaces need to be removed.

AND

Make the following changes to Table B1:

PROCESS	SUB-PROCESS	PS PRIORITY	EVENT TEXT	MESSAGE NOTES AND DETAILS	Error Code SET	EventID	TRAP NAME
Init	TFTP	Critical	TFTP file complete - but failed ~~Message Integrity~~SHA-1 hash check~~ MIC~~	For SYSLOG only: append: File name = <P1> P1 = filename of TFTP file	D08.0	68000800	
CDP	CDS	Notice	Unable to obtain all WAN-Data IP addresses the PS was configured to obtain		P02.0	80000200	cabhPsDevCdpWanDataIpTrap

CDP	CDS	Notice	Unable to provision DHCP LAN client- IP address pool exhausted		P03.0	80000300	cabhPsDevCdpLanIpPoolTrap
CSP	Firewall TFTP	Critical	TFTP download of firewall policy file failed: request sent, no response	P1 = requested firewall policy file URL	P130.0	80013000	cabhPsDevCSPTrap
CSP	Firewall TFTP	Critical	TFTP failed - firewall policy file not found	P1 = requested firewall policy file URL	P131.0	80013100	cabhPsDevCSPTrap
CSP	Firewall TFTP	Critical	TFTP failed - invalid firewall policy file	P1 = requested firewall policy file URL	P132.0	80013200	cabhPsDevCSPTrap
CSP	Firewall TFTP	Critical	Firewall policy file download complete but failed SHA-1 has check	P1 = requested firewall policy file URL, P2 = firewall policy file has value	P133.0	80013300	cabhPsDevCSPTrap
CSP	Firewall TFTP	Critical	Firewall policy file download exceeded maximum allowable number of TFTP retries	P1 = requested firewall policy file URL	P134.0	80013400	cabhPsDevCSPTrap
CSP	Firewall TFTP	Notice	Firewall policy file TFTP download success	P1 = requested firewall policy file URL For SYSLOG only: append: Retry limit = <P2> P2 = maximum allowable number of retry attempts	P135.0	80013500	cabhPsDevCSPTrap
CTP Events							

CTP	Connection Speed Tool	Notice	Connection Speed Tool test completed successfully	P1 = IP address of source P2 = IP address of destination P3 = protocol P4 = throughput	P301.0	80030100	cabhPsDevCtpTrap
CTP	Connection Speed Tool	Notice	Connection Speed Tool test timed out	P1 = IP address of source P2 = IP address of destination P3 = protocol P4 = value of timer (millisec)	P302.0	80030200	cabhPsDevCtpTrap
CTP	Connection Speed Tool	Notice	Connection Speed Tool test aborted	P1 = IP address of source P2 = IP address of destination P3 = protocol P4 = value of timer (millisec)	P303.0	80030300	cabhPsDevCtpTrap
CTP	Ping Tool	Notice	Ping Tool test completed successfully	P1 = IP address of source P2 = IP address of destination P3 = average round trip time	P320.0	80032000	cabhPsDevCtpTrap
CTP	Ping Tool	Notice	Ping Tool test timed out	P1 = IP address of source P2 = IP address of destination P3 = number of requests sent P4 = number of responses received	P321.0	80032100	cabhPsDevCtpTrap

CTP	Ping Tool	Notice	Ping Tool test aborted	P1 = IP address of source P2 = IP address of destination P3 = number of requests sent P4 = number of responses received	P322.0	80032200	cabhPsDevCtpTrap

Notes to Table B-1:

* Software upgrade (secure software download) events apply to stand-alone Portal Services only. Software upgrade is controlled by the cable modem in an embedded PS, so software upgrade event reporting is managed by the cable modem in an embedded PS. For more information, refer to Section 11.3.7.1 Software Download Into Embedded or Standalone PS Elements.

AND

Replace the entire section B1 with the following :

All traps specified by IPCable2Home are defined in the PS DEV MIB specification, [Annex E.1].

108. Add the following at the end of Annex D

RFC 3235, Network Address Translator (NAT)-Friendly Application Design Guidelines, outlines a number of guidelines for creating applications in such a manner that they will not be compromised when running in the presence of Network Address Translation functionality. It is strongly recommended that developers of applications that will to run within a IPCable2Home environment adhere to these guidelines.

109. Several figures need to be replaced:

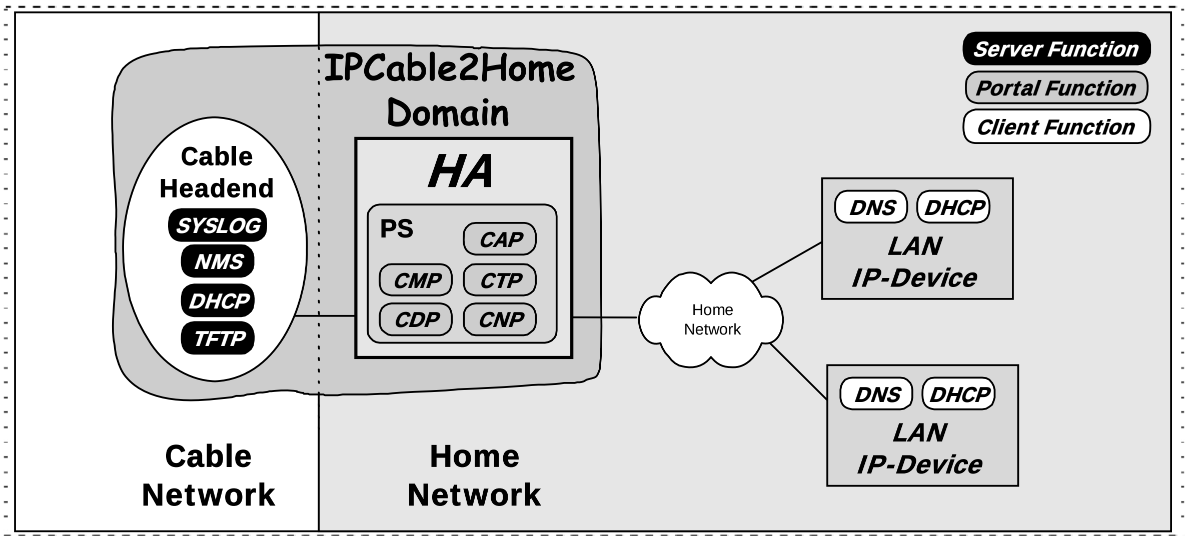

Figure 3 Management Elements

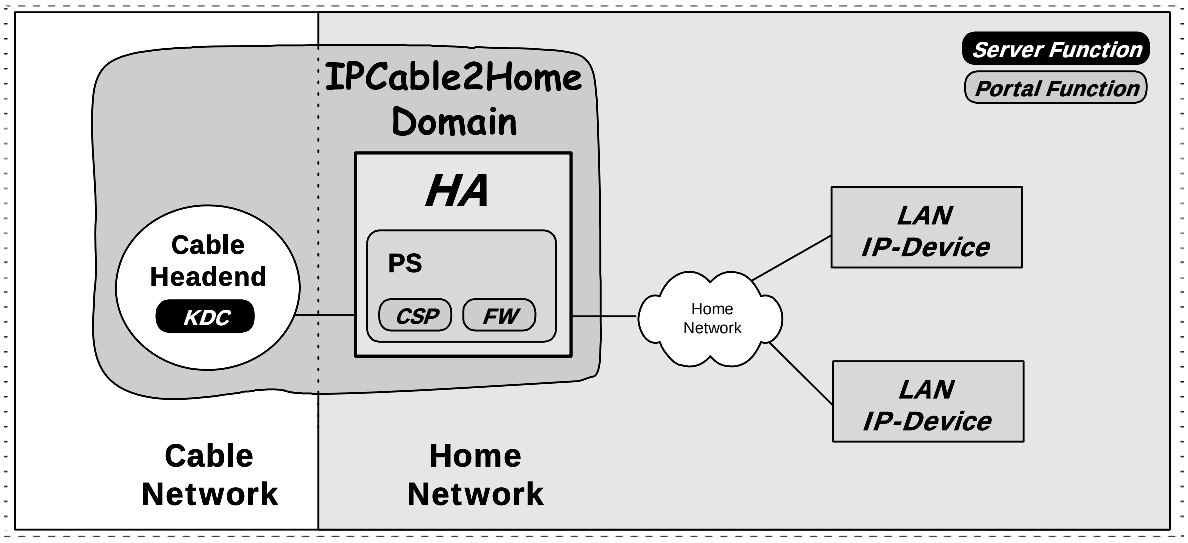

Figure 4 Security Elements

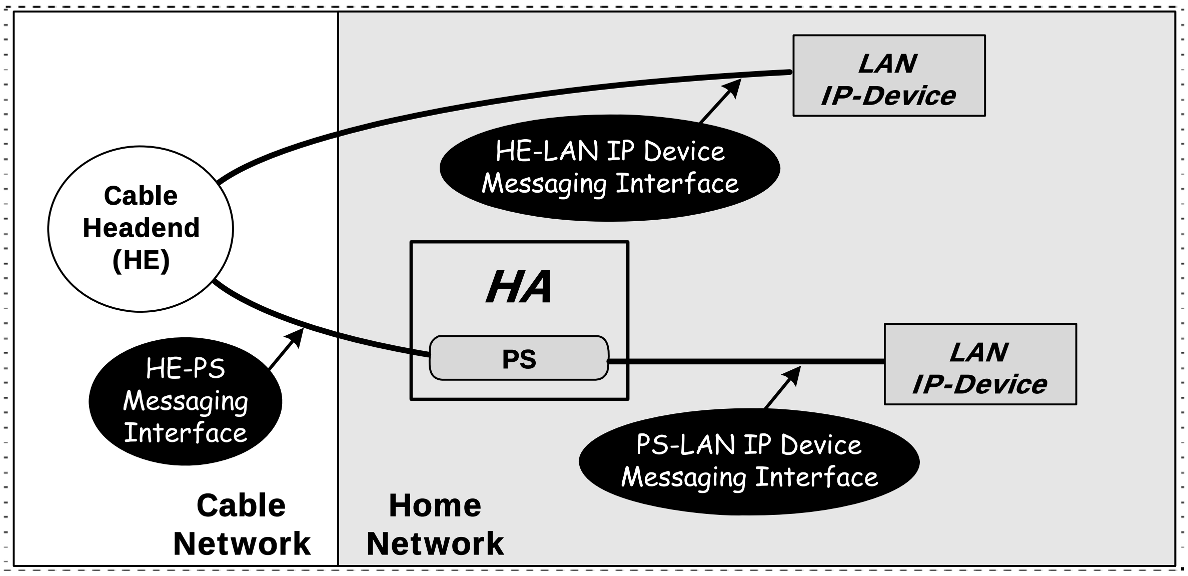
LAN IP-Device
HE-LAN IP Device Messaging Interface
Cable Headend (HE)
HA
PS
LAN IP-Device
HE-PS Messaging Interface
PS-LAN IP Device Messaging Interface
Cable Network
Home Network

Figure 5 Reference Interfaces

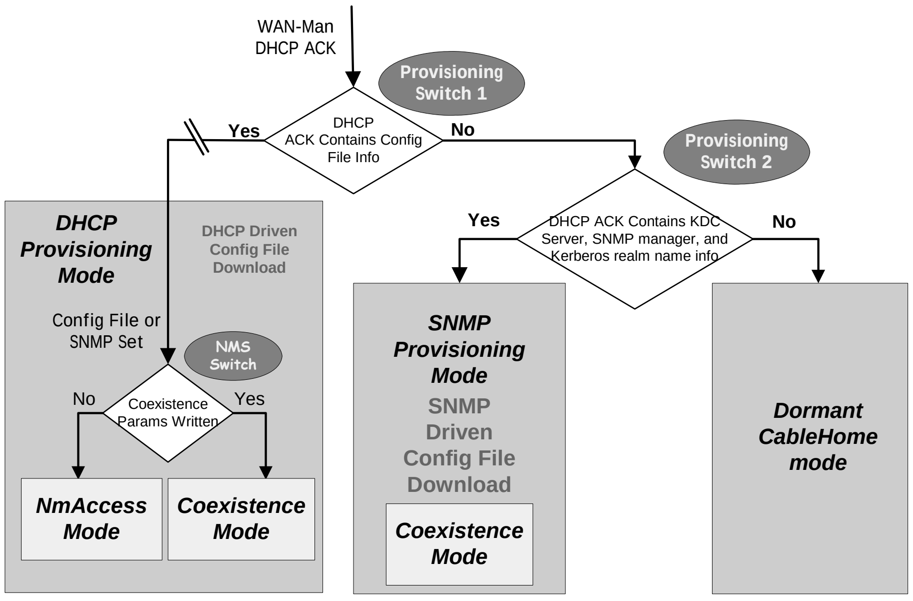

Figure 8 PS Operational Modes

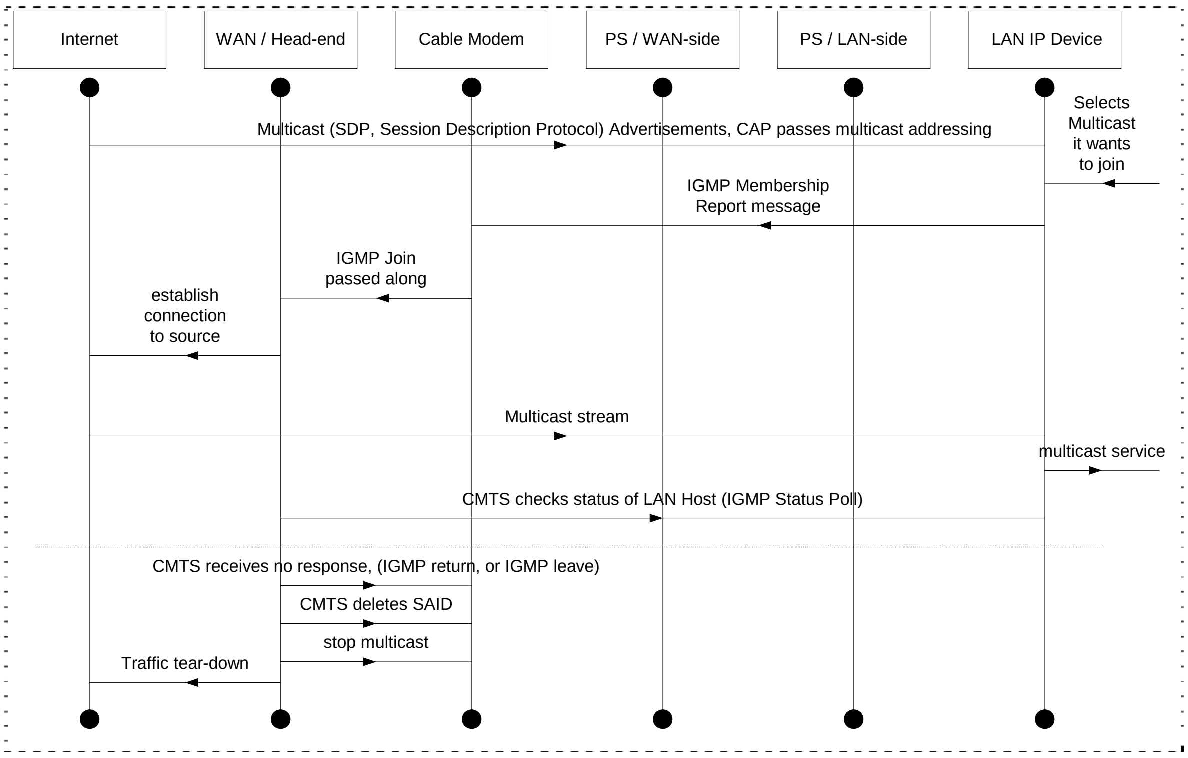
Internet
WAN / Head-end
Cable Modem
PS / WAN-side
PS / LAN-side
LAN IP Device
Selects Multicast it wants to join
Multicast (SDP, Session Description Protocol) Advertisements, CAP passes multicast addressing
IGMP Membership Report message
IGMP Join passed along
establish connection to source
Multicast stream
multicast service
CMTS checks status of LAN Host (IGMP Status Poll)
CMTS receives no response, (IGMP return, or IGMP leave)
CMTS deletes SAID
stop multicast
Traffic tear-down

Figure 18 Multicast via IGMP Sequence

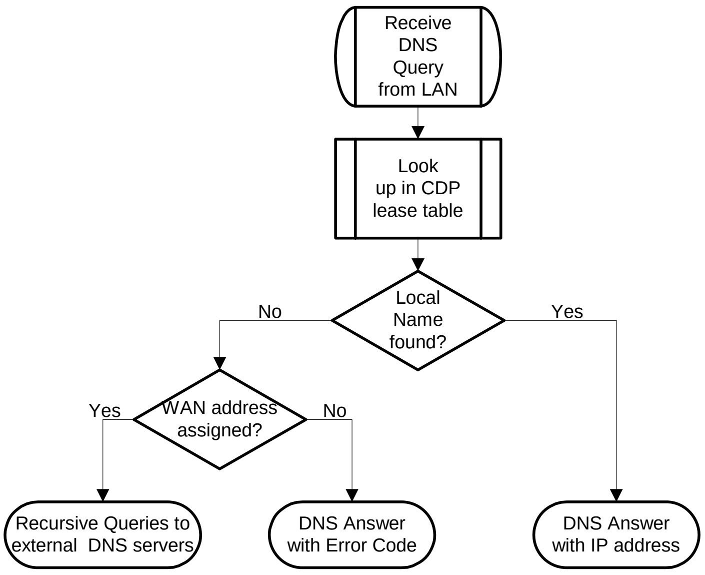
Receive DNS Query from LAN
Look up in CDP lease table
Local Name found?
No
Yes
WAN address assigned?
Yes
No
Recursive Queries to external DNS servers
DNS Answer with Error Code
DNS Answer with IP address

Figure 21 CNP Packet Processing

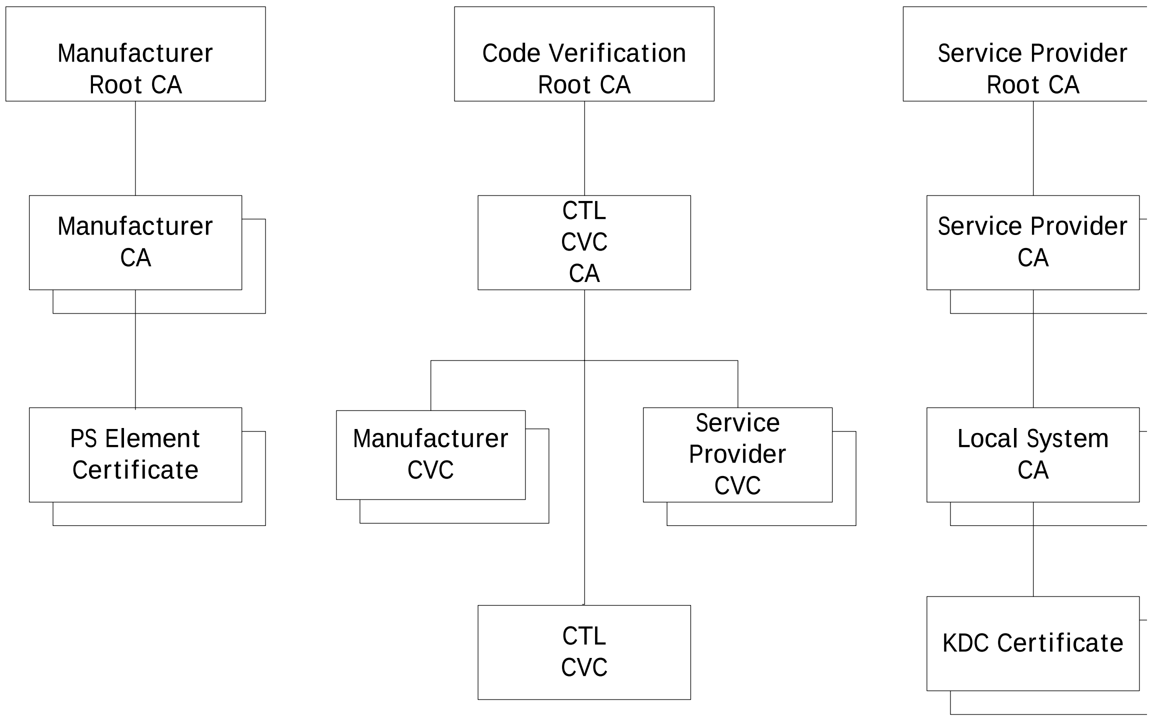

Figure 25 IPCable2Home Certificate Hierarchy

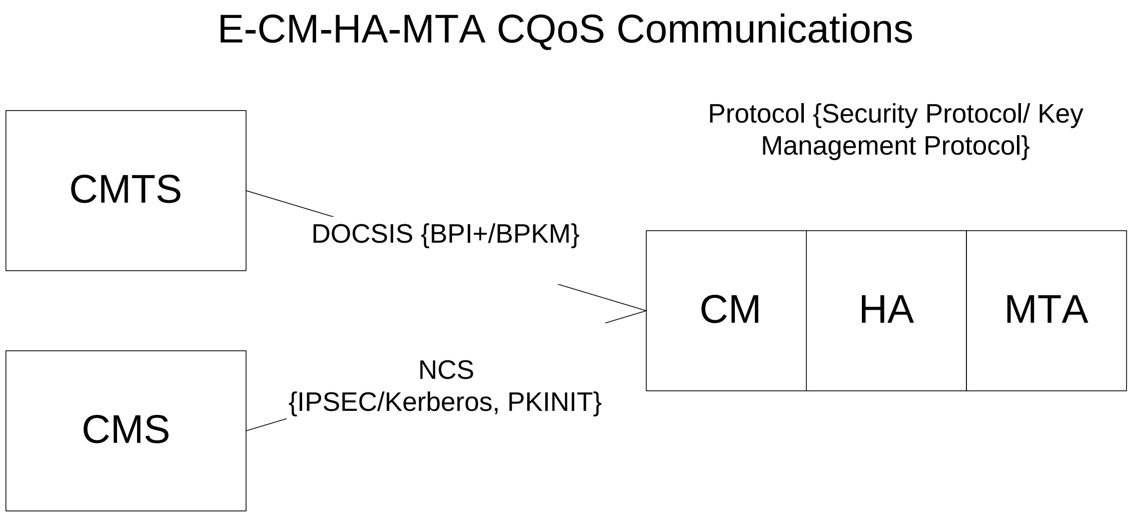
E-CM-HA-MTA CQoS Communications
Protocol {Security Protocol/ Key Management Protocol}
CMTS
DOCSIS {BPI+/BPKM}
CM
HA
MTA
NCS {IPSEC/Kerberos, PKINIT}
CMS

Figure 27 Secure CQoS Architecture to the MTA

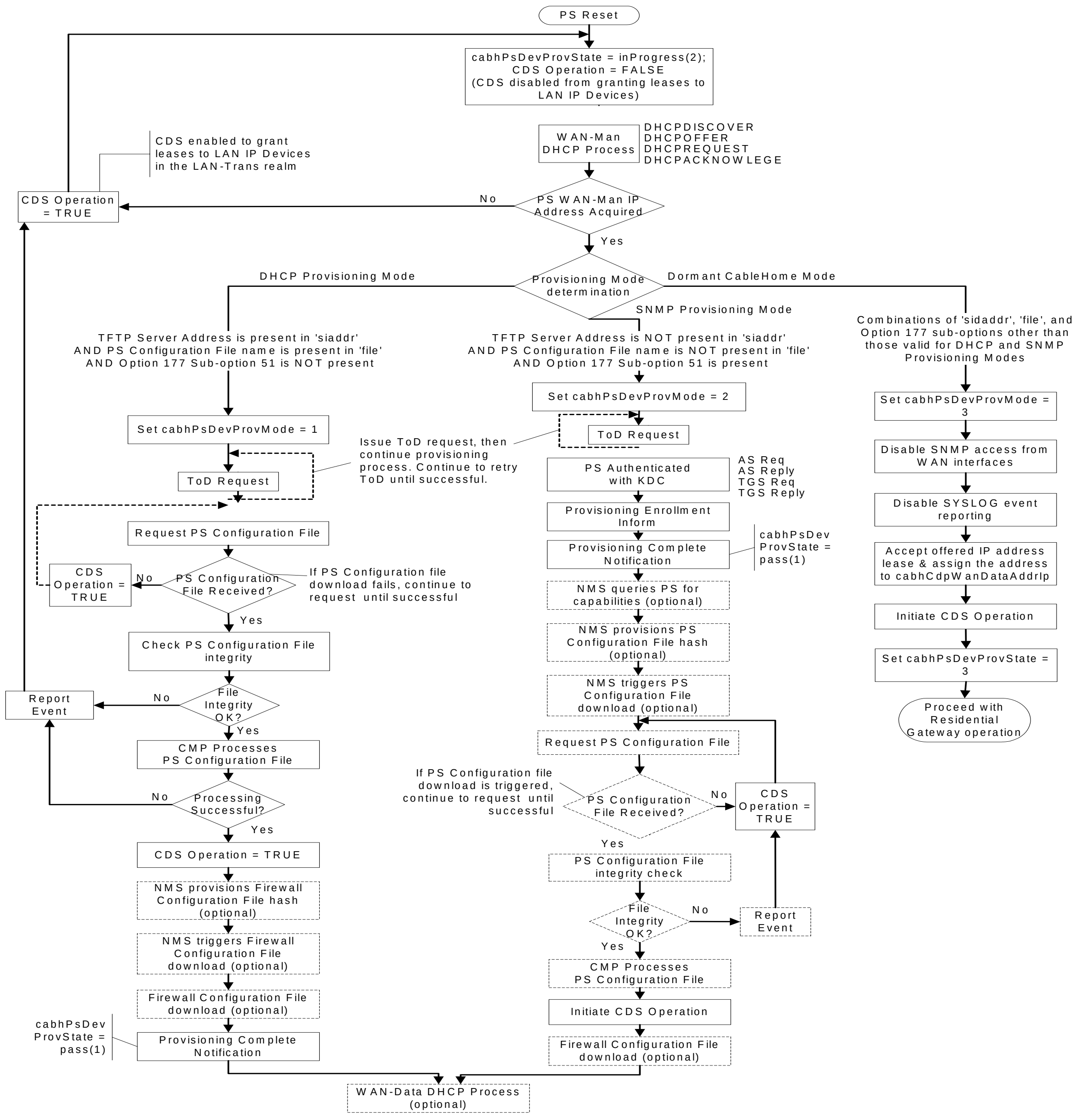
PS Reset
cabhPsDevProvState = inProgress(2);
CDS Operation = FALSE
(CDS disabled from granting leases to LAN IP Devices)
WAN-Man DHCP Process
DHCPDISCOVER
DHCPOFFER
DHCPREQUEST
DHCPACKNOWLEGE
CDS enabled to grant leases to LAN IP Devices in the LAN-Trans realm
CDS Operation = TRUE
No
PS WAN-Man IP Address Acquired
Yes
DHCP Provisioning Mode
Provisioning Mode determination
Dormant CableHome Mode
SNMP Provisioning Mode
TFTP Server Address is present in 'siaddr' AND PS Configuration File name is present in 'file' AND Option 177 Sub-option 51 is NOT present
TFTP Server Address is NOT present in 'siaddr' AND PS Configuration File name is NOT present in 'file' AND Option 177 Sub-option 51 is present
Combinations of 'sidaddr', 'file', and Option 177 sub-options other than those valid for DHCP and SNMP Provisioning Modes
Set cabhPsDevProvMode = 1
Set cabhPsDevProvMode = 2
Set cabhPsDevProvMode = 3
Issue ToD request, then continue provisioning process. Continue to retry ToD until successful.
ToD Request
ToD Request
PS Authenticated with KDC
AS Req
AS Reply
TGS Req
TGS Reply
Disable SNMP access from WAN interfaces
Disable SYSLOG event reporting
Request PS Configuration File
Provisioning Enrollment Inform
Provisioning Complete Notification
cabhPsDev ProvState = pass(1)
Accept offered IP address lease & assign the address to cabhCdpWanDataAddrIp
CDS Operation = TRUE
No
PS Configuration File Received?
If PS Configuration file download fails, continue to request until successful
NMS queries PS for capabilities (optional)
Initiate CDS Operation
Yes
Check PS Configuration File integrity
NMS provisions PS Configuration File hash (optional)
Set cabhPsDevProvState = 3
NMS triggers PS Configuration File download (optional)
Proceed with Residential Gateway operation
Report Event
No
File Integrity OK?
Yes
CMP Processes PS Configuration File
Request PS Configuration File
If PS Configuration file download is triggered, continue to request until successful
PS Configuration File Received?
No
CDS Operation = TRUE
No
Processing Successful?
Yes
Yes
CDS Operation = TRUE
PS Configuration File integrity check
NMS provisions Firewall Configuration File hash (optional)
File Integrity OK?
No
Report Event
Yes
NMS triggers Firewall Configuration File download (optional)
CMP Processes PS Configuration File
Firewall Configuration File download (optional)
Initiate CDS Operation
cabhPsDev ProvState = pass(1)
Provisioning Complete Notification
Firewall Configuration File download (optional)
WAN-Data DHCP Process (optional)

Figure 39 Provisioning Modes

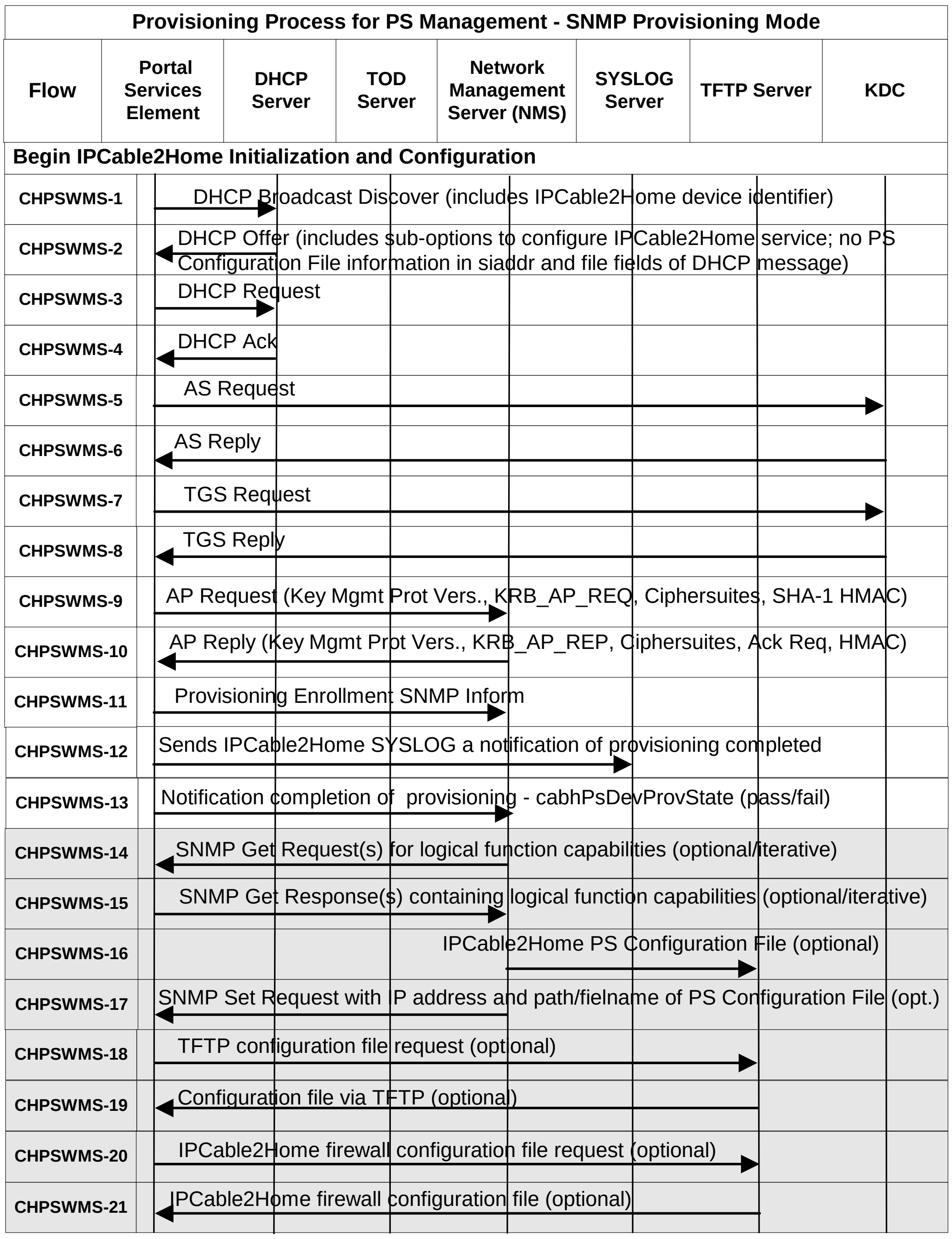
Provisioning Process for PS Management - SNMP Provisioning Mode
Flow
Portal Services Element
DHCP Server
TOD Server
Network Management Server (NMS)
SYSLOG Server
TFTP Server
KDC
Begin IPCable2Home Initialization and Configuration
CHPSWMS-1 DHCP Broadcast Discover (includes IPCable2Home device identifier)
CHPSWMS-2 DHCP Offer (includes sub-options to configure IPCable2Home service; no PS Configuration File information in siaddr and file fields of DHCP message)
CHPSWMS-3 DHCP Request
CHPSWMS-4 DHCP Ack
CHPSWMS-5 AS Request
CHPSWMS-6 AS Reply
CHPSWMS-7 TGS Request
CHPSWMS-8 TGS Reply
CHPSWMS-9 AP Request (Key Mgmt Prot Vers., KRB_AP_REQ, Ciphersuites, SHA-1 HMAC)
CHPSWMS-10 AP Reply (Key Mgmt Prot Vers., KRB_AP_REP, Ciphersuites, Ack Req, HMAC)
CHPSWMS-11 Provisioning Enrollment SNMP Inform
CHPSWMS-12 Sends IPCable2Home SYSLOG a notification of provisioning completed
CHPSWMS-13 Notification completion of provisioning - cabhPsDevProvState (pass/fail)
CHPSWMS-14 SNMP Get Request(s) for logical function capabilities (optional/iterative)
CHPSWMS-15 SNMP Get Response(s) containing logical function capabilities (optional/iterative)
CHPSWMS-16 IPCable2Home PS Configuration File (optional)
CHPSWMS-17 SNMP Set Request with IP address and path/fielname of PS Configuration File (opt.)
CHPSWMS-18 TFTP configuration file request (optional)
CHPSWMS-19 Configuration file via TFTP (optional)
CHPSWMS-20 IPCable2Home firewall configuration file request (optional)
CHPSWMS-21 IPCable2Home firewall configuration file (optional)

Figure 41 Provisioning Process for PS Management - SNMP Provisioning Mode

110. Changes in MIBs are very difficult to describe; hence it is easier to just copy the entire annex with changes highlited.

Annex E MIBS

E.1 Portal Service (PS) MIB

The ~~PS~~PSDev MIB MUST be implemented as defined below.

```
CABH-PS-DEV-MIB DEFINITIONS ::= BEGIN
IMPORTS
   MODULE-IDENTITY,
   OBJECT-TYPE,
   Integer32,
   NOTIFICATION-TYPE                                      FROM SNMPv2-SMI
   TruthValue,
   ~~DisplayString,~~


   PhysAddress,
   DateAndTime,
   TEXTUAL-CONVENTION                              FROM SNMPv2-TC
   SnmpAdminString                                     FROM SNMP-FRAMEWORK-MIB
   OBJECT-GROUP,
   MODULE-COMPLIANCE,
   NOTIFICATION-GROUP                                     FROM SNMPv2-CONF

   InetAddressType,
   InetAddress,
   InetAddressIPv4,
   InetAddressIPv6                                      FROM INET-ADDRESS-MIB

   docsDevSwCurrentVers,
   docsDevEvLevel,
   docsDevEvId,
   docsDevEvText,
   docsDevSwFilename,
   docsDevSwServer~~,~~                                              FROM DOCS-CABLE-
DEVICE-MIB -- RFC2669

   cabhCdpServerDhcpAddress,
   cabhCdpWanDataAddrClientId,
   cabhCdpLanTransThreshold                            FROM CABH-CDP-MIB

   clabProjCableHome                                      FROM CLAB-DEF-MIB;

--
=============================================================================
--
--         History:
--
--
--
=============================================================================


cabhPsDevMib MODULE-IDENTITY
    LAST-UPDATED    "~~0112190000Z" -- December 19, 2001~~0209200000Z"-- September 20, 2002
    ORGANIZATION    "~~Cable  NMP Group"~~CableLabs Broadband Access Department"
    CONTACT-INFO
            "Kevin Luehrs
            Postal: Cable Television Laboratories, Inc.
                               400 Centennial Parkway
                               Louisville, Colorado 80027-1266
                          U.S.A.
            Phone:  +1 303-661-9100
            Fax:    +1 303-661-9199
            E-mail: k.luehrs@cablelabs.com"
    DESCRIPTION
            "This MIB module supplies the basic management objects
             for the PS Device.  The PS device parameter describe
              general PS Device attributes and behavior characteristics.
              Most the PS Device MIB is need for configuration download.



    ::=  { clabProjCableHome 1 }

-- Textual conventions
             X509Certificate ::= TEXTUAL-CONVENTION
                          STATUS current
                          DESCRIPTION
                                      "An X509 digital certificate encoded as an ASN.1 DER object."
                          SYNTAX OCTET STRING (SIZE (0..4096))


~~--~~
~~--  assumes SNMPv3~~
~~--  load management is per DOCSIS 1.1 only~~
~~--~~


cabhPsDevMibObjects    OBJECT IDENTIFIER ::= { cabhPsDevMib 1 }
cabhPsDevBase               OBJECT IDENTIFIER ::= { cabhPsDevMibObjects 1 }
cabhPsDevProv              OBJECT IDENTIFIER ::= { cabhPsDevMibObjects 2 }

--
-- The following group describes the base objects in the PS.
-- These are device based parameters.
--

cabhPsDevDateTime OBJECT-TYPE
        SYNTAX      DateAndTime
        MAX-ACCESS  read-write
        STATUS      current
        DESCRIPTION
            "The date and time, with optional timezone
             information."
    ::= { cabhPsDevBase 1 }

cabhPsDevResetNow              OBJECT-TYPE
    SYNTAX                     TruthValue
    MAX-ACCESS                 read-write
    STATUS                     current
```

DESCRIPTION
"Setting this object to true(1) causes the stand-alone or embedded PS device to ~~reset.~~
reboot. Device code initializes as if starting from a power-on reset. The CMP
ensures that MIB object values persist as specified. Reading this object always returns
false(2).~~ When~~
~~cabhPsDevResetNow is set to true, the following actions~~
~~occur:~~
~~1. Clear all statistics in PS.~~
~~2. Clear trace logs.~~
~~3. Clear all security associations.~~
~~4. Initialize all configuration parameters~~
~~5. Delete all address translations~~
~~6. Delete all FQDN to IP mappings~~
~~7. Delete all stored ARP translations 8. The provisioning flow is started at~~
~~step PS - 1.~~"
::= { cabhPsDevBase 2 }

cabhPsDevSerialNumber OBJECT-TYPE
SYNTAX ~~DisplayString~~SnmpAdminString (SIZE (0..128))
MAX-ACCESS read-only
STATUS current
DESCRIPTION
"The manufacturer's serial number for this PS. This parameter
~~is manufacturer provided and is stored in non-volatile memory."~~
is manufacturer provided and is stored in non-volitile memory."
::= { cabhPsDevBase 3 }

cabhPsDevHardwareVersion OBJECT-TYPE
SYNTAX ~~DisplayString~~SnmpAdminString (SIZE (0..48))
MAX-ACCESS read-only
STATUS current
DESCRIPTION
"The manufacturer's hardware version for this PS. This
parameter is manufacturer provided and is stored in non-volatile
memory."
::= { cabhPsDevBase 4 }

~~cabhPsDevMacAddress~~cabhPsDevWanManMacAddress OBJECT-TYPE
SYNTAX PhysAddress
MAX-ACCESS read-only
STATUS current
DESCRIPTION
"The PS WAN-MAN MAC address.~~ Typically, ~~ This is the PS ~~WAN-MAN and PS ~~hardware
address
~~WAN-DATA addresses will be identical. The client identifiers~~
~~will not be the same so that each may be assigned a different~~
~~IP address."~~
~~::= { cabhPsDevBase 5 }~~
to be used by the CDC to uniquely identify the PS to the cable data network DHCP
server for the acquisition of an IP address to be used for
management messaging between the cable network NMS and the CMP"

::= { cabhPsDevBase 5 }

cabhPsDevWanDataMacAddress OBJECT-TYPE

```
        SYNTAX      PhysAddress
        MAX-ACCESS  read-only
        STATUS      current
        DESCRIPTION
                "The PS WAN-Data MAC address. The PS could have multiple WAN-Data
                Interfaces, which share the same hardware address.  The client
                identifiers will be unique so that each may be assigned
                a different, unique IP address."


   ::= { cabhPsDevBase 6 }

cabhPsDevTypeIdentifier    OBJECT-TYPE
        SYNTAX      ~~DisplayString~~SnmpAdminString
        MAX-ACCESS  read-only
        STATUS      current
        DESCRIPTION
                "This is a copy of the device type identifier used in the
                DHCP option 60 exchanged between the PS and the
                DHCP server."
   ~~::= { cabhPsDevBase 6 }~~

~~cabhPsDevResetDefaults    OBJECT-TYPE~~
   ~~SYNTAX      TruthValue~~
   ~~MAX-ACCESS  read-write~~
   ~~STATUS      current~~
   ~~DESCRIPTION~~
      ~~"Setting this object to True set all PS parameters to the~~
      ~~factory defaults"~~
        ::= { cabhPsDevBase 7 }

cabhPsDevSetToFactory    OBJECT-TYPE
        SYNTAX      TruthValue
        MAX-ACCESS  read-write
        STATUS      current
        DESCRIPTION
                "Setting this object to true(1) sets all PsDev MIB objects
                to the factory default values. Reading this object always
                returns false(2)."

   ::= { cabhPsDevBase 8 }

cabhPsDevWanManClientId    OBJECT-TYPE
        SYNTAX      OCTET STRING (SIZE (1..80))
        MAX-ACCESS  read-write
        STATUS      current
        DESCRIPTION
                "This is the client ID used for WAN-MAN DHCP requests.
                The default value is the 6 byte MAC address."
   ::= { cabhPsDevBase ~~8~~9 }

cabhPsDevTodSyncStatus OBJECT-TYPE
   SYNTAX      TruthValue
   MAX-ACCESS  read-only
   STATUS      current
   DESCRIPTION
```

```
            "This object indicates whether the PS was able to
            successfully synchronize with the Time of Day (ToD)
            Server in the cable network. The PS sets this object
            to true(1) if the PS successfully synchronizes its time
            with the ToD server. The PS sets this object to
            false(2) if the PS does not successfully synchronize
            with the ToD server"
    ~~REFERENCE~~
            ~~" "~~    DEFVAL { false }
    ~~::= { cabhPsDevBase 9 }~~

~~cabhPsDevProvMode OBJECT-TYPE~~
    ~~SYNTAX        INTEGER~~
    ~~{~~
        ~~dhcpmode (1),~~
        ~~snmpmode (2)~~
    ~~}~~
    ~~MAX-ACCESS  read-only~~
    ~~STATUS      current~~
    ~~DESCRIPTION~~
            ~~"This object indicates the provisioning mode in which~~
            ~~the PS is operating. If the PS receives PS Configuration~~
            ~~File information (server address and file name) in the DHCP~~
            ~~message issued by the DHCP server in the cable network,~~
            ~~the PS sets this object to DHCPmode(1). If the PS receives~~
            ~~DHCP Option 177 sub-option 51 AND does not receive PS~~
            ~~Configuration File information in the DHCP message the PS~~
            ~~receives from the DHCP server in the cable network, the PS~~
            ~~sets this object to SNMPmode(2). "~~
    ::= { cabhPsDevBase 10 }


~~cabhPsDevDwnldMode~~ cabhPsDevProvMode OBJECT-TYPE
            SYNTAX              INTEGER
            {
        ~~standard    (1),~~
        ~~enhanced    (2)~~
                    dhcpmode(1),
                    snmpmode(2)
            }
            MAX-ACCESS  read-only
            STATUS              current
            DESCRIPTION
                    "This ~~is~~object indicates the ~~download~~provisioning mode ~~that~~in which the ~~PS will used"~~
                    PS is operating. If the PS is operating in DHCP Provisioning
                    Mode, the PS
                    sets this object to dhcpmode(1). If the PS is operating in SNMP
                    Provisioning Mode, the PS sets this object to snmpmode(2)."
    ::= { cabhPsDevBase 11 }



--
--          The following group defines Provisioning Specific parameters
--
```

```
cabhPsDevProvisioningTimer OBJECT-TYPE
        SYNTAX      INTEGER (0..16383)
        UNITS           "minutes"
        MAX-ACCESS  read-write
        STATUS      current
        DESCRIPTION
                "This object enables the user to set the duration of the provisioning
                timeout timer. The value is in minutes. Setting the timer
                to 0 disables it. The default value for the timer is 5."
        DEFVAL {5}
        ::=  {cabhPsDevProv 1}

cabhPsDevProvConfigFile OBJECT-TYPE
   SYNTAX      ~~DisplayString~~SnmpAdminString (SIZE(1..128))
   MAX-ACCESS  read-write
   STATUS      current
   DESCRIPTION
        "The URL of the TFTP host for downloading provisioning
         and configuration parameters to this device. Returns NULL if the
         server address is unknown."
   ::= { cabhPsDevProv 2 }

cabhPsDevProvConfigHash OBJECT-TYPE
        SYNTAX OCTET STRING (SIZE(20))
        MAX-ACCESS read-write
        STATUS current
        DESCRIPTION
                "Hash of the contents of the config file, calculated and
                sent to the PS prior to sending the config file.  For the
                SHA-1 authentication algorithm the hash length is 160 bits."
        ::= { cabhPsDevProv 3 }

cabhPsDevProvConfigFileSize OBJECT-TYPE
        SYNTAX          Integer32
        UNITS           "bytes"
        MAX-ACCESS  read-only
        STATUS          current
        DESCRIPTION
                "Size of the configuration file."
        ::={ cabhPsDevProv 4 }

cabhPsDevProvConfigFileStatus OBJECT-TYPE
        SYNTAX                  INTEGER
        {
            idle      (1),
            busy    (2)
        }
        MAX-ACCESS  read-only
        STATUS                  current
        DESCRIPTION
                "This object indicates the current status
of the configuration file download process. It is
provided to indicate to the management entity
that the PS will reject PS Configuration File triggers
(set request to cabhPsDevProvConfigFile) when busy."
        ::={ cabhPsDevProv 5 }
```

```
cabhPsDevProvConfigTLVProcessed OBJECT-TYPE
          SYNTAX                    INTEGER (0..16383)
          MAX-ACCESS  read-only
          STATUS                    current
          DESCRIPTION
                    "Number of TLVs processed in config file."
          ::={ cabhPsDevProv ~~5 }~~6 }

cabhPsDevProvConfigTLVRejected OBJECT-TYPE
          SYNTAX                    INTEGER (0..16383)
          MAX-ACCESS  read-only
          STATUS                    current
          DESCRIPTION
                    "Number of TLVs rejected in config file."
          ::={ cabhPsDevProv ~~6 }~~7 }

cabhPsDevProvSolicitedKeyTimeout OBJECT-TYPE
  SYNTAX      Integer32 (15..600)
  UNITS                    "seconds"
  MAX-ACCESS  read-write
  STATUS      current
          DESCRIPTION
                    "This timeout applies only when the Provisioning Server initiated
                    key management (with a Wake Up message) for SNMPv3.  It is the
                    period during which the PS will save a number (inside the
                    sequence number field) from the sent out AP Request and wait for the
                    matching AP Reply from the Provisioning Server."
          DEFVAL { 120 }
          ::= { cabhPsDevProv ~~7 }~~8 }


cabhPsDevProvState    OBJECT-TYPE
   SYNTAX      INTEGER
   {
         pass                (1),
         inProgress          (2),
         fail                (3)
   }
   MAX-ACCESS  read-only
   STATUS      current
   DESCRIPTION
         "This object indicates the completion state of the
    initialization process. Pass or Fail states occur after
    completion of the initialization flow. InProgress occurs
    from PS initialization start to PS
    initialization end."
   ::= { cabhPsDevProv ~~8 }~~9 }

cabhPsDevProvAuthState    OBJECT-TYPE
   SYNTAX      INTEGER
   {
         accepted            (1),
         rejected            (2)
   }
   MAX-ACCESS  read-only
   STATUS      current
   DESCRIPTION
         "This object indicates the authentication state
         of the configuration file."
   ::= { cabhPsDevProv ~~9 }~~10 }

cabhPsDevProvCorrelationId OBJECT-TYPE
   SYNTAX      Integer32
   MAX-ACCESS  read-only
   STATUS      current
   DESCRIPTION
         " Random value generated by the PS for use in registration
           authorization.  It is for use only in the PS initialization
           messages and for PS configuration file download. This value
           appears in both cabhPsDevProvisioningStatus and
           cabhPsDevProvisioningEnrollmentReport informs to verify the
           instance of loading the configuration file."
   ::= { cabhPsDevProv ~~10 }~~11 }

cabhPsDevTimeServerAddrType OBJECT-TYPE
       SYNTAX      InetAddressType
       MAX-ACCESS  read-only
       STATUS      current
       DESCRIPTION
          "The IP address type of the Time server (RFC-868).  IP version 4
          is typically used."
   ::= { cabhPsDevProv ~~11 }~~12 }

cabhPsDevTimeServerAddr OBJECT-TYPE
       SYNTAX      InetAddress
       MAX-ACCESS  read-only
       STATUS      current
       DESCRIPTION
          "The IP address of the Time server (RFC-868). Returns
           0.0.0.0 if the time server IP address is unknown."
   ::= { cabhPsDevProv ~~12 }~~13 }

--
-- notification group is for future extension.
--

cabhPsNotification OBJECT IDENTIFIER ::= { cabhPsDevMib 2 0 }
cabhPsConformance  OBJECT IDENTIFIER ::= { cabhPsDevMib 3 }
cabhPsCompliances  OBJECT IDENTIFIER ::= { cabhPsConformance 1 }
cabhPsGroups       OBJECT IDENTIFIER ::= { cabhPsConformance 2 }

--
--    Notification Group
--

cabhPsDevInitTLVUnknownTrap     NOTIFICATION-TYPE
  OBJECTS   {
     docsDevEvLevel,
     docsDevEvId,
     docsDevEvText,
~~     cabhPsDevMacAddress~~
```

```
      cabhPsDevWanManMacAddress
   }
   STATUS  current
   DESCRIPTION
           "Event due to detection of unknown TLV during
           the TLV parsing process.
           The values of docsDevEvLevel, docsDevId, and docsDevEvText are from the
entry which logs
           this event in the docsDevEventTable.  The value of ~~cabhPsDevMacAddress~~
cabhPsDevWanManMacAddress indicates the ~~MAC~~
           Wan-Man MAC address of the PS.
           This part of the information is uniform across all PS Traps.
           "
   ::= { cabhPsNotification 1 }

cabhPsDevInitTrap  NOTIFICATION-TYPE
   OBJECTS {
                docsDevEvLevel,
                docsDevEvId,
                docsDevEvText,
      ~~cabhPsDevMacAddress,~~
      cabhPsDevWanManMacAddress,
                cabhPsDevProvConfigFile,
                cabhPsDevProvConfigTLVProcessed,
                cabhPsDevProvConfigTLVRejected
   }
   STATUS      current
   DESCRIPTION
           "This inform is issued to confirm the successful completion
           of the provisioning process."
   ::= { cabhPsNotification 2 }

cabhPsDevInitRetryTrap  NOTIFICATION-TYPE
   OBJECTS  {
                docsDevEvLevel,
                docsDevEvId,
                docsDevEvText,
      ~~cabhPsDevMacAddress~~
      cabhPsDevWanManMacAddress
   }
   STATUS      current
   DESCRIPTION
           "An event to report a failure happened during the initialization process
and detected
           in the PS.
           "
   ::= { cabhPsNotification 3 }

cabhPsDevDHCPFailTrap NOTIFICATION-TYPE
   OBJECTS {
       docsDevEvLevel,
       docsDevEvId,
       docsDevEvText,
       ~~cabhPsDevMacAddress,~~
       cabhPsDevWanManMacAddress,
       cabhCdpServerDhcpAddress
```

```
  }
  STATUS  current
  DESCRIPTION
        "An event to report the failure of a DHCP server.
        The value of  cabhCdpServerDhcpAddressis the IP address
        of the DHCP server.
        "
  ::= { cabhPsNotification 4 }


cabhPsDevSwUpgradeInitTrap NOTIFICATION-TYPE
  OBJECTS {
      docsDevEvLevel,
      docsDevEvId,
      docsDevEvText,
~~      cabhPsDevMacAddress,~~
      cabhPsDevWanManMacAddress,
      docsDevSwFilename,
      docsDevSwServer
  }
  STATUS  current
  DESCRIPTION
        "An event to report a software upgrade initiated
        event. The values of docsDevSwFilename, and
        docsDevSwServer indicate the software image name
        and the server IP address the image is from.
        "
  ::= { cabhPsNotification 5 }


cabhPsDevSwUpgradeFailTrap NOTIFICATION-TYPE
  OBJECTS {
      docsDevEvLevel,
      docsDevEvId,
      docsDevEvText,
~~      cabhPsDevMacAddress,~~
      cabhPsDevWanManMacAddress,
      docsDevSwFilename,
      docsDevSwServer
  }
  STATUS  current
  DESCRIPTION
        "An event to report the failure of a software upgrade
        attempt. The values of docsDevSwFilename, and
        docsDevSwServer indicate the software image name
        and the server IP address the image is from.
        "
  ::= { cabhPsNotification 6 }


cabhPsDevSwUpgradeSuccessTrap NOTIFICATION-TYPE
  OBJECTS {
      docsDevEvLevel,
      docsDevEvId,
      docsDevEvText,
~~      cabhPsDevMacAddress,~~
```

```
      cabhPsDevWanManMacAddress,
      docsDevSwFilename,
      docsDevSwServer
   }
   STATUS  current
   DESCRIPTION
          "An event to report the Software upgrade success event.
          The values of docsDevSwFilename, and
          docsDevSwServer indicate the software image name
          and the server IP address the image is from.
          "
   ::= { cabhPsNotification 7 }


cabhPsDevSwUpgradeCVCFailTrap NOTIFICATION-TYPE
   OBJECTS {
      docsDevEvLevel,
      docsDevEvId,
      docsDevEvText,
      ~~cabhPsDevMacAddress~~
      cabhPsDevWanManMacAddress
   }
   STATUS  current
   DESCRIPTION
          "An event to report the failure of the verification
          of code file happened during a secure software upgrade
          attempt.
          "
   ::= { cabhPsNotification 8 }


cabhPsDevTODFailTrap NOTIFICATION-TYPE
   OBJECTS {
      docsDevEvLevel,
      docsDevEvId,
      docsDevEvText,
      cabhPsDevTimeServerAddr ,
      cabhPsDevWanManMacAddress
   }
   STATUS  current
   DESCRIPTION
          "An event to report the failure of a time of day server.
          The value of cabhPsDevTimeServerAddr indicates the server IP
          address.
          "
   ::= { cabhPsNotification 9 }

cabhPsDevCdpWanDataIpTrap  NOTIFICATION-TYPE
   OBJECTS {
      docsDevEvLevel,
      docsDevEvId,
      docsDevEvText,
      cabhCdpWanDataAddrClientId ,
      cabhPsDevWanManMacAddress
   }
   STATUS      current
```

```
  DESCRIPTION
         "An event to report the failure of PS to obtain all needed WAN-Data Ip
Addresses.
         cabhCdpWanDataAddrClientId indicates the ClientId for which the failure ~~occurred.~~
occured.
         "
  ::= { cabhPsNotification 10 }


cabhPsDevCdpThresholdTrap  NOTIFICATION-TYPE
  OBJECTS {
      docsDevEvLevel,
      docsDevEvId,
      docsDevEvText,
~~      cabhPsDevMacAddress,~~
      cabhPsDevWanManMacAddress,
      cabhCdpLanTransThreshold
  }
  STATUS      current
  DESCRIPTION
         "An event to report that the ~~LAN~~Lan-Trans threshold has been exceeded."
  ::= { cabhPsNotification 11 }

cabhPsDevCspTrap  NOTIFICATION-TYPE
  OBJECTS {
      docsDevEvLevel,
      docsDevEvId,
      docsDevEvText,
~~      cabhPsDevMacAddress~~
      cabhPsDevWanManMacAddress
  }
  STATUS      current
  DESCRIPTION
         "To report an event with the Cable Security Portal."
  ::= { cabhPsNotification 12 }

cabhPsDevCapTrap  NOTIFICATION-TYPE
  OBJECTS {
      docsDevEvLevel,
      docsDevEvId,
      docsDevEvText,
~~      cabhPsDevMacAddress~~
      cabhPsDevWanManMacAddress
  }
  STATUS      current
  DESCRIPTION
         "To report an event with the Cable Address Portal."
  ::= { cabhPsNotification 13 }

cabhPsDevCtpTrap  NOTIFICATION-TYPE
  OBJECTS {
      docsDevEvLevel,
      docsDevEvId,
      docsDevEvText,
      cabhPsDevWanManMacAddress
  }
  STATUS      current
  DESCRIPTION
         "To report an event with the CableHome Test Portal."
  ::= { cabhPsNotification 14 }

cabhPsDevProvEnrollTrap  NOTIFICATION-TYPE
  OBJECTS {
     cabhPsDevHardwareVersion,
     docsDevSwCurrentVers,
     cabhPsDevTypeIdentifier,
     ~~cabhPsDevMacAddress,~~
     cabhPsDevWanManMacAddress,
     cabhPsDevProvCorrelationId
  }
  STATUS      current
  DESCRIPTION
         "This inform is issued to initiate the  CableHome
                process provisioning ."
  REFERENCE
            "Inform as defined in RFC 1902"
  ~~::= { cabhPsNotification 14 }~~
  ::= { cabhPsNotification 15 }

cabhPsDevCdpLanIpPoolTrap NOTIFICATION-TYPE
         OBJECTS {
         docsDevEvLevel, docsDevEvId, docsDevEvText, cabhPsDevWanManMacAddress,
         cabhCdpLanTransCurCount
         }
         STATUS current
         DESCRIPTION
"An event to report that the pool of IP addresses for LAN clients, as
defined by cabh CdpLanPoolStart and cabhCdpLanPoolEnd, is exhausted "

         ::= { cabhPsNotification 16}

-- compliance statements

cabhPsBasicCompliance MODULE-COMPLIANCE
   STATUS     current
   DESCRIPTION
          "The compliance statement for devices that implement
           PS feature."
   MODULE   --cabhPsMib




-- unconditionally mandatory groups

   MANDATORY-GROUPS {
          cabhPsGroup
   }

::= { cabhPsCompliances ~~3 }~~1}
```

```
cabhPsGroup OBJECT-GROUP
   OBJECTS {
       cabhPsDevDateTime,
       cabhPsDevResetNow,
       cabhPsDevSerialNumber,
       cabhPsDevHardwareVersion,
       ~~cabhPsDevMacAddress,~~
       cabhPsDevWanManMacAddress,
       cabhPsDevWanDataMacAddress,
       cabhPsDevTypeIdentifier,
              ~~cabhPsDevResetDefaults,~~
          cabhPsDevSetToFactory,
                    cabhPsDevWanManClientId,
              cabhPsDevTodSyncStatus,
       cabhPsDevProvMode,
       ~~cabhPsDevDwnldMode,~~


       cabhPsDevProvisioningTimer,
              cabhPsDevProvConfigFile,
              cabhPsDevProvConfigHash,
              cabhPsDevProvConfigFileSize,
              cabhPsDevProvConfigFileStatus,
          cabhPsDevProvConfigTLVProcessed,
              cabhPsDevProvConfigTLVRejected,
              cabhPsDevProvSolicitedKeyTimeout,
              cabhPsDevProvState,
              cabhPsDevProvAuthState,
       cabhPsDevProvCorrelationId,
       cabhPsDevTimeServerAddrType,
       cabhPsDevTimeServerAddr

   }
   STATUS    current
   DESCRIPTION
      "Group of objects for PS MIB."
   ::= { cabhPsGroups 1 }

cabhPsNotificationGroup                    NOTIFICATION-GROUP
         NOTIFICATIONS { cabhPsDevInitTLVUnknownTrap, cabhPsDevInitTrap,
cabhPsDevInitRetryTrap,
              cabhPsDevDHCPFailTrap, cabhPsDevSwUpgradeInitTrap,
cabhPsDevSwUpgradeFailTrap,
              cabhPsDevSwUpgradeSuccessTrap, cabhPsDevSwUpgradeCVCFailTrap,
cabhPsDevTODFailTrap,
              cabhPsDevCdpWanDataIpTrap, cabhPsDevCdpThresholdTrap,
cabhPsDevCspTrap,
              cabhPsDevCapTrap, cabhPsDevCtpTrap, cabhPsDevProvEnrollTrap }
         STATUS          current
         DESCRIPTION
                  "These notifications deal with change in status of
                  PS Device."
         ::= { cabhPsGroups 2 }
```

```
END
```

E.2 Cable Test Portal MIB

The CTP MIB MUST be implemented as defined below.

```
CABH-CTP-MIB DEFINITIONS ::= BEGIN
IMPORTS
  MODULE-IDENTITY,
  OBJECT-TYPE
                                                              FROM SNMPv2-SMI
  TruthValue,
  TEXTUAL-CONVENTION
                                                              FROM SNMPv2-TC
  OBJECT-GROUP,
  MODULE-COMPLIANCE
                FROM SNMPv2-CONF
  InetAddressType,
  InetAddress,
  InetAddressIPv4,
  InetAddressIPv6
                                         FROM INET-ADDRESS-MIB
  clabProjCableHome
                                                              FROM CLAB-DEF-MIB;

--
==========================================================================
--
--        History:
--
--
        ==========================================================================
=======
--        Date            Modified by            Reason
--
--
==========================================================================

cabhCtpMib MODULE-IDENTITY
   LAST-UPDATED ~~"0112190000Z" -- December 19, 2001~~ "0209200000Z" -- September 20, 2002
   ORGANIZATION    "~~Cable NMP Group"~~CableLabs Broadband Access Department"
   CONTACT-INFO
        "Kevin Luehrs
        Postal: Cable Television Laboratories, Inc.
                      400 Centennial Parkway
                      Louisville, Colorado 80027-1266
        U.S.A.
        Phone:  +1 303-661-9100
        Fax:    +1 303-661-9199
        E-mail: k.luehrs@cablelabs.com"
   DESCRIPTION
        "This MIB module defines the diagnostic controls
```

```
          offered by the Cable Test Portal (CTP).

          ~~Acknowledgements:~~

   ::=  { clabProjCableHome 5 }

-- Textual conventions



~~--~~
~~--   assumes SNMPv3~~
~~--   SW load management is per DOCSIS 1.1 only~~
~~--~~


cabhCtpObjects          OBJECT IDENTIFIER ::= { cabhCtpMib 1 }
cabhCtpBase             OBJECT IDENTIFIER ::= { cabhCtpObjects 1 }
cabhCtpConnSpeed                OBJECT IDENTIFIER ::= { cabhCtpObjects 2 }
cabhCtpPing                       OBJECT IDENTIFIER ::= { cabhCtpObjects 3 }

--
--         The following group describes the base objects in the Cable
--         Management Portal.
--

~~cabhCtpReset~~ cabhCtpSetToFactory OBJECT-TYPE
   SYNTAX               TruthValue
   MAX-ACCESS           read-write
   STATUS               current
   DESCRIPTION

"Setting this object to true(1) causes all ~~testing to be~~
          ~~terminated.~~  the tables in the CTP MIB to
be cleared, and all CTP MIB objects with default values set back to those
default values. Reading this object always returns false(2).
          ~~When cabhCtpReset is set to true, the following actions~~
          ~~occur:~~
          ~~1.        Terminate any diagnostic tests in progress.~~
                            ~~2. Clear all diagnostic statistics."~~ "
::= { cabhCtpBase 1 }

--
--         Parameter and results from Connection Speed Command
--

cabhCtpConnSrcIpType OBJECT-TYPE
         SYNTAX              InetAddressType
         MAX-ACCESS  read-write
  STATUS      current
  DESCRIPTION
          "The IP Address type used as the source address for the Connection
                            Speed Test."
         DEFVAL { ipv4 }
         ::= { cabhCtpConnSpeed 1 }
```

```
cabhCtpConnSrcIp       OBJECT-TYPE
          SYNTAX                 InetAddress
          MAX-ACCESS  read-write
  STATUS      current
  DESCRIPTION
         "The IP Address used as the source address for the Connection
                               Speed Test.~~ Typically the address with be the value in~~       The default
value is the value of cabhCdpServerRouter~~.  The default address is 192.168.0.1."~~
(192.168.0.1)."
          REFERENCE
                    " Specification Section ~~7.3.5.1"~~6.4.4"
          DEFVAL { 'c0a80001'h }-- 192.168.0.1
          ::= { cabhCtpConnSpeed 2 }

cabhCtpConnDestIpType          OBJECT-TYPE
          SYNTAX                 InetAddressType
          MAX-ACCESS  read-write
  STATUS                 current
  DESCRIPTION
~~         "The IP Address type used as the destination address for the Connection~~
~~                               Speed Test."~~
          "The IP Address Type for the CTP Connection Speed Tool destination
address."
          DEFVAL { ipv4 }
::= { cabhCtpConnSpeed 3 }



cabhCtpConnDestIp      OBJECT-TYPE
          SYNTAX                 InetAddress
          MAX-ACCESS  read-write
   STATUS      current
   DESCRIPTION
         "The IP Address used as the destination address for the Connection
                               Speed Test."
          ::= { cabhCtpConnSpeed 4 }

cabhCtpConnProto       OBJECT-TYPE
          SYNTAX                 INTEGER {
                                        udp                    (1),
                                        tcp                    (2)
                                        }
             MAX-ACCESS  read-write
   STATUS      current
   DESCRIPTION
         "The protocol used in the Connection Speed Test.  TCP
         testing is optional."
             DEFVAL { udp }
          ::= { cabhCtpConnSpeed 5 }


~~cabhCtpConnPort~~cabhCtpConnNumPkts           OBJECT-TYPE
             SYNTAX                 INTEGER (1..65535)
             MAX-ACCESS  read-write
   STATUS      current
```

```
    DESCRIPTION
            "The ~~port used for~~number of packets the CTP is to send when triggered to
 execute the Connection Speed ~~Test."~~Tool."
             DEFVAL {~~7}~~ 100 }
             ::= { cabhCtpConnSpeed 6 }

~~cabhCtpConnNumPkts  OBJECT-TYPE~~
~~             SYNTAX                    INTEGER (1..255)~~
~~             MAX-ACCESS  read-write~~
~~    STATUS      current~~
~~    DESCRIPTION~~
~~            "The number of packets to send."~~
~~             DEFVAL {1}~~
 ~~::= { cabhCtpConnSpeed 7 }~~

cabhCtpConnPktSize    OBJECT-TYPE
             SYNTAX                    INTEGER (64..1518)
             MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
            "The size of the test frames."
             REFERENCE
                          ""
             DEFVAL { 1518 }
             ::= { cabhCtpConnSpeed ~~8 }~~7 }

cabhCtpConnTimeOut   OBJECT-TYPE
             SYNTAX                    INTEGER (0..600000)                    -- Max 10 minutes
             UNITS              "milliseconds"
             MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
            "The timeout value for the response.  A value of zero indicates
                                  no time out and can be used for TCP only."
             DEFVAL {~~600000}~~30000}  -- 30 seconds
    ::= { cabhCtpConnSpeed ~~9 }~~8 }

cabhCtpConnControl    OBJECT-TYPE
             SYNTAX                    INTEGER {
             ~~notRun (1),~~                                          start~~ (2),~~(1),
                                                       abort~~ (3)~~(2)
~~                                                       }~~
             }
             MAX-ACCESS  read-write
    STATUS              current
    DESCRIPTION
            "The control for the Connection Speed ~~Test.  The value notRun~~Tool. Setting this object to
start(1)
causes the Connection Speed Tool to execute. Setting this object to abort(2)
            ~~is used to indicate never executed.~~ causes the Connection Speed Tool to stop running. This
parameter should           only be
set via SNMP."
            DEFVAL  {~~ notRun(1)~~abort }
             ::= { cabhCtpConnSpeed ~~10~~9 }
```

```
cabhCtpConnStatus      OBJECT-TYPE
        SYNTAX               INTEGER {
        notRun(1),
        running          (~~1~~2),
complete~~ (2),~~(3),
                         aborted          (~~3~~4),
timedOut(5)
                         }
        MAX-ACCESS  read-only
   STATUS               current
   DESCRIPTION
        "The ~~Status~~status of the ~~currently/last executed test~~Connection Speed Tool."
        DEFVAL   { ~~complete(2)~~notRun }
        ::= { cabhCtpConnSpeed ~~11~~10 }


cabhCtpConnPktsSent  OBJECT-TYPE
        SYNTAX               INTEGER (~~0..255)~~0..65535)
        MAX-ACCESS  read-only
   STATUS      current
   DESCRIPTION
        "The number of packets the CTP sent after it was triggered to
execute
the Connection Speed Tool."
        ::= { cabhCtpConnSpeed ~~12 }~~11 }

cabhCtpConnPktsRecv  OBJECT-TYPE
        SYNTAX               INTEGER (~~0..255)~~0..65535)
        MAX-ACCESS  read-only
   STATUS      current
   DESCRIPTION
        "The number ~~for packet~~of packets the CTP received after it executed the
        Connection Speed Tool."
        ::= { cabhCtpConnSpeed 12 }

cabhCtpConnRTT         OBJECT-TYPE
        SYNTAX         INTEGER (0..600000)
        UNITS          "millisec"
        MAX-ACCESS  read-only
   STATUS      current
   DESCRIPTION
        "The resulting round trip time for the set of
        packets sent to and received~~."~~ from the target LAN IP Device."
        ::= { cabhCtpConnSpeed 13 }

~~cabhCtpConnAvgRTT~~cabhCtpConnThroughput  OBJECT-TYPE
        SYNTAX               INTEGER (~~0..600000)~~0..65535)
~~        UNITS          "millisec"~~
        MAX-ACCESS  read-only
   STATUS      current
   DESCRIPTION
        "The ~~resulting ~~average ~~of round trip times for~~round-trip throughput measured in
~~        acknowledged packets."~~
        kilobits per second."
        ::= { cabhCtpConnSpeed 14 }
```

```
~~cabhCtpConnMaxRTT   OBJECT-TYPE~~
~~            SYNTAX                      INTEGER (0..600000)~~
~~            UNITS             "millisec"~~
~~            MAX-ACCESS  read-only~~
~~    STATUS       current~~
~~    DESCRIPTION~~
~~           "The resulting maximum of round trip times for~~
~~           acknowledged packets."~~
~~            ::= { cabhCtpConnSpeed 15 }~~

~~cabhCtpConnMinRTT    OBJECT-TYPE~~
~~            SYNTAX                      INTEGER (0..600000)~~
~~            UNITS             "millisec"~~
~~            MAX-ACCESS  read-only~~
~~    STATUS       current~~
~~    DESCRIPTION~~
~~           "The resulting minimum of round trip times for~~
~~           acknowledged packets."~~
~~            ::= { cabhCtpConnSpeed 16 }~~

~~cabhCtpConnNumIcmpError      OBJECT-TYPE~~
~~            SYNTAX                      INTEGER (0..255)~~
~~            MAX-ACCESS  read-only~~
~~    STATUS       current~~
~~    DESCRIPTION~~
~~           "Number of ICMP errors."~~
~~            ::= { cabhCtpConnSpeed 17 }~~

~~cabhCtpConnIcmpError OBJECT-TYPE~~
~~            SYNTAX                      INTEGER (0..255)~~
~~            MAX-ACCESS  read-only~~
~~    STATUS       current~~
~~    DESCRIPTION~~
~~           "The last ICMP error."~~
~~            ::= { cabhCtpConnSpeed 18 }~~

--
--          Parameters and Results for Ping Command
--

cabhCtpPingSrcIpType  OBJECT-TYPE
            SYNTAX                      InetAddressType
            MAX-ACCESS  read-write
   STATUS                 current
   DESCRIPTION
           "The IP Address Type ~~used as the~~for CTP Ping Tool source address~~ for the Ping      Test."~~."
DEFVAL { ipv4 }
            ::= { cabhCtpPing 1 }

cabhCtpPingSrcIp          OBJECT-TYPE
            SYNTAX                      InetAddress
            MAX-ACCESS  read-write
    STATUS       current
    DESCRIPTION
           "The IP Address used as the source address for the Ping
```

```
                                  Test.~~ Typically the address will be~~ The default value is the value of
                                  ~~PS WanMan IP address.  The address 192.168.0.x is used."~~
   CabhCdpServerRouter (192.168.0.1)."
    REFERENCE
             " Specification Section 6.4.4"
    DEFVAL { 'c0a80001'h }
             ::= { cabhCtpPing 2 }

cabhCtpPingDestIpType            OBJECT-TYPE
             SYNTAX                       InetAddressType
             MAX-ACCESS  read-write
    STATUS                   current
    DESCRIPTION
            "The ~~Destination~~ IP Address Type ~~used as~~for the CTP Ping Tool destination address~~ for
~~the Ping Test."~~."
DEFVAL { ipv4 }
             ::= { cabhCtpPing 3 }

cabhCtpPingDestIp       OBJECT-TYPE
             SYNTAX                       InetAddress
             MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
            "The Destination IP Address used as the destination address for
            the Ping Test."
             ::= { cabhCtpPing 4 }

~~cabhCtpPingProto          OBJECT-TYPE~~
~~             SYNTAX                       INTEGER {~~
~~                                                 icmp      (1),~~
~~                                                 }~~
~~             MAX-ACCESS  read-write~~
~~    STATUS      current~~
~~    DESCRIPTION~~
~~            "The protocol used to gather topology info."~~
~~             DEFVAL { icmp }~~
~~             ::= { cabhCtpPing 5 }~~

cabhCtpPingNumPkts    OBJECT-TYPE
             SYNTAX                       INTEGER (1..4)
             MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
            "The number of packets to send to each host."
             DEFVAL {1}
             ::= { cabhCtpPing ~~6 }~~5 }

cabhCtpPingPktSize      OBJECT-TYPE
             SYNTAX                       INTEGER (64..1518)
             MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
            "The size of the test frames."
             DEFVAL {64}
             ::= { cabhCtpPing ~~7 }~~6 }
```

```
cabhCtpPingTimeBetween        OBJECT-TYPE
          SYNTAX              INTEGER (0..600000)
          UNITS          "milliseconds"
          MAX-ACCESS  read-write
   STATUS      current
   DESCRIPTION
         "The time between sending one ping and the next."
          DEFVAL { 1000 }
   ::= { cabhCtpPing ~~8 }~~7 }

cabhCtpPingTimeOut            OBJECT-TYPE
          SYNTAX              INTEGER (~~0..600000~~1..600000)
          UNITS          "milliseconds"
          MAX-ACCESS  read-write
   STATUS            current
   DESCRIPTION
         "The time out for ping response ~~of sending~~(ICMP reply) for a single transmitted ping
message (ICMP request)."
          DEFVAL  { 5000 }          -- 5 seconds
   ::= { cabhCtpPing ~~9~~8 }



cabhCtpPingControl      OBJECT-TYPE
          SYNTAX              INTEGER {
          ~~notRun            (1),~~                                     start~~ (2),~~(1),
                                        abort              (~~3~~2)
                                        }
          MAX-ACCESS  read-write
   STATUS            current
   DESCRIPTION
         "The control for the Ping ~~Test.  The value notRun~~Tool. Setting this object to start(1) causes
the
         ~~is used to indicate never executed."~~
Ping Tool to execute. Setting this object to abort(2) causes the Ping Tool to
stop running. This parameter should only be set via SNMP."
        DEFVAL  {~~ notRun(1)~~abort }
          ::= { cabhCtpPing ~~10~~9 }


cabhCtpPingStatus       OBJECT-TYPE
          SYNTAX              INTEGER {
          notRun(1),
          running          (~~1~~2),
complete~~ (2),~~(3),
                                        aborted           (~~3~~4),
timedOut(5)
                                        }
          MAX-ACCESS  read-only
   STATUS            current
   DESCRIPTION
         "The ~~Status~~status of the ~~currently/last executed test~~Ping Tool."
DEFVAL  { notRun }
          ::= { cabhCtpPing ~~11~~10 }
```

```
cabhCtpPingNumSent   OBJECT-TYPE
            SYNTAX                    INTEGER (~~0..255~~0..4)
            MAX-ACCESS  read-only
    STATUS                    current
    DESCRIPTION
            "The number of ~~pings~~Pings sent~~.~~"
            ~~DEFVAL { complete(2) }~~
            ::= { cabhCtpPing ~~12~~11 }

cabhCtpPingNumRecv   OBJECT-TYPE
            SYNTAX                    INTEGER (0..255)
            MAX-ACCESS  read-only
    STATUS       current
    DESCRIPTION
            "The number of pings received."
            ::= { cabhCtpPing 12 }

cabhCtpPingAvgRTT      OBJECT-TYPE
            SYNTAX                    INTEGER (0..600000)
            UNITS                     "millisec"
            MAX-ACCESS  read-only
    STATUS       current
    DESCRIPTION
            "The resulting average of round trip times for acknowledged
packets."
            ::= { cabhCtpPing 13 }

cabhCtpPingMaxRTT      OBJECT-TYPE
            SYNTAX                    INTEGER (0..600000)
            UNITS                     "millisec"
            MAX-ACCESS  read-only
    STATUS       current
    DESCRIPTION
            "The resulting maximum of round trip times for acknowledged
packets."
            ::= { cabhCtpPing 14 }

cabhCtpPingMinRTT      OBJECT-TYPE
            SYNTAX                    INTEGER (0..600000)
            UNITS                     "millisec"
            MAX-ACCESS  read-only
    STATUS       current
    DESCRIPTION
            "The resulting minimum of round trip times for acknowledged
packets."
            ::= { cabhCtpPing 15 }

cabhCtpPingNumIcmpError          OBJECT-TYPE
            SYNTAX                    INTEGER (0..255)
            MAX-ACCESS  read-only
    STATUS       current
    DESCRIPTION
            "Number of ICMP errors."
            ::= { cabhCtpPing 16 }

cabhCtpPingIcmpError   OBJECT-TYPE
```

```
            SYNTAX                     INTEGER (0..255)
            MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
           "The last ICMP error."
            ::= { cabhCtpPing 17 }

--=========================================================================

--
-- notification group is for future extension.
--

cabhCtpNotification OBJECT IDENTIFIER ::= { cabhCtpMib 2 0 }
cabhCtpConformance  OBJECT IDENTIFIER ::= { cabhCtpMib 3 }
cabhCtpCompliances  OBJECT IDENTIFIER ::= { cabhCtpConformance 1 }
cabhCtpGroups       OBJECT IDENTIFIER ::= { cabhCtpConformance 2 }

--
--    Notification Group
--

-- compliance statements

cabhCtpBasicCompliance MODULE-COMPLIANCE
    STATUS     current
    DESCRIPTION
           "The compliance statement for devices that implement
            Portal Service feature."
    MODULE    --cabhCtpMib



-- unconditionally mandatory groups

    MANDATORY-GROUPS {
           cabhCtpGroup
    }

::= { cabhCtpCompliances 3 }


cabhCtpGroup OBJECT-GROUP
    OBJECTS {
~~      cabhCtpReset,~~

               cabhCtpSetToFactory,
               cabhCtpConnSrcIpType,
               cabhCtpConnSrcIp,
               cabhCtpConnDestIpType,
               cabhCtpConnDestIp,
               cabhCtpConnProto,
               ~~cabhCtpConnPort,~~        cabhCtpConnNumPkts,
               cabhCtpConnPktSize,
               cabhCtpConnTimeOut,
               cabhCtpConnControl,
               cabhCtpConnStatus,
               cabhCtpConnPktsSent,
               cabhCtpConnPktsRecv,
               ~~cabhCtpConnAvgRTT,~~
               ~~cabhCtpConnMinRTT,~~
               ~~cabhCtpConnMaxRTT,~~
               ~~cabhCtpConnNumIcmpError,~~
               ~~cabhCtpConnIcmpError,~~
               cabhCtpConnRTT,
               cabhCtpConnThroughput,

               cabhCtpPingSrcIpType,
               cabhCtpPingSrcIp,
               cabhCtpPingDestIpType,
               cabhCtpPingDestIp,
               ~~cabhCtpPingProto,~~       cabhCtpPingNumPkts,
               cabhCtpPingPktSize,
               cabhCtpPingTimeBetween,
               cabhCtpPingTimeOut,
               cabhCtpPingControl,
               cabhCtpPingStatus,
               cabhCtpPingNumSent,
               cabhCtpPingNumRecv,
               cabhCtpPingAvgRTT,
               cabhCtpPingMinRTT,
               cabhCtpPingMaxRTT,
               cabhCtpPingNumIcmpError,
               cabhCtpPingIcmpError
            }
    STATUS    current
    DESCRIPTION
        "Group of objects for ~~Cable~~ CTP MIB."
    ::= { cabhCtpGroups 1 }

END
```

E.3 Security MIB

The ~~Security~~SEC MIB MUST be implemented as defined below.

```
CABH-SEC-MIB DEFINITIONS ::= BEGIN
IMPORTS
    MODULE-IDENTITY,
             Unsigned32,
             BITS,
             OBJECT-TYPE                                                          FROM SNMPv2-SMI
             TruthValue,
             DisplayString,
             TimeStamp                                                            FROM SNMPv2-TC
             OBJECT-GROUP,
             MODULE-COMPLIANCE                        FROM SNMPv2-CONF
             InetAddressIPv4                     FROM INET-ADDRESS-MIB
             SnmpAdminString                     FROM SNMP-FRAMEWORK-MIB -- RFC2571
             X509Certificate             FROM DOCS-BPI2-MIB
```

```
        clabProjCableHome       FROM CLAB-DEF-MIB;

--=====================================================================
--
--      History:
--
--      Date            Modified by             Reason
--
--
--=====================================================================

cabhSecMib MODULE-IDENTITY
  LAST-UPDATED    "~~0112200000Z" -- December 20, 2001~~0209200000Z" --September 20, 2002
  ORGANIZATION    "~~Cable NMP Group"~~CableLabs Broadband Access Department"
  CONTACT-INFO
        "Kevin Luehrs
        Postal:   Cable Television Laboratories, Inc.
                  400 Centennial Parkway
                  Louisville, Colorado 80027-1266
                  U.S.A.
        Phone:  +1 303-661-9100
        Fax:    +1 303-661-9199
        E-mail: k.luehrs@cablelabs.com"
  DESCRIPTION
        "This MIB module supplies the basic management objects
        for the Security Portal Services.

~~        Acknowledgements:~~


  ::=  { clabProjCableHome 2 }

-- Textual conventions


~~--~~
~~-- assumes SNMPv3~~
~~-- SW load management is per DOCSIS 1.1 only~~
~~--~~

cabhSecFwObjects                OBJECT IDENTIFIER ::= { cabhSecMib 1 }
cabhSecFwBase           OBJECT IDENTIFIER ::= { cabhSecFwObjects 1 }
cabhSecFwLogCtl                 OBJECT IDENTIFIER ::= { cabhSecFwObjects 2 }
cabhSecCertObjects              OBJECT IDENTIFIER ::= { cabhSecMib 2 }
--
--      The following group describes the base objects in the Cable Home
--      Firewall.
--

cabhSecFwPolicyFileEnable OBJECT-TYPE
        SYNTAX      INTEGER {
                                    enable          (1),
                                    disable         (2)
                                    }
  MAX-ACCESS  read-write
  STATUS      current
```

```
   DESCRIPTION
           "This parameter indicates whether or not to enable the firewall
           functionality."
            DEFVAL {enable}
   ::= { cabhSecFwBase 1 }

cabhSecFwPolicyFileURL OBJECT-TYPE
   SYNTAX      DisplayString
   MAX-ACCESS  read-write
   STATUS      current
   DESCRIPTION
           "This object contains the name and IP address of the policy rule set
file ~~in  a~~ina TFTP URL format.  Once this object has been updated, it will
                                trigger the file download."

   ::= { cabhSecFwBase 2 }




cabhSecFwPolicyFileHash OBJECT-TYPE
                SYNTAX OCTET STRING (SIZE(20))
                MAX-ACCESS read-write
                STATUS current
                DESCRIPTION
                        "Hash of the contents of the rules set file, calculated                   and sent to the
PS prior to sending the rules set file.                   For the SHA-1 authentication
algorithm the ~~hash~~ length   of the hash is 160 bits. This hash value is
encoded in binary format."

            ::= { cabhSecFwBase 3 }


cabhSecFwPolicyFileOperStatus OBJECT-TYPE
            SYNTAX          INTEGER         {
                        inProgress(1),
                        completeFromProvisioning(2),
                        completeFromMgt(3),
                        failed(4)
            }
            MAX-ACCESS read-only
            STATUS current
            DESCRIPTION
      "InProgress(1) indicates that a TFTP download is underway,
       either as a result of a version mismatch at provisioning
       or as a result of a upgradeFromMgt request.
       CompleteFromProvisioning(2) indicates that the last
       software upgrade was a result of version mismatch at
       provisioning. CompleteFromMgt(3) indicates that the last
       software upgrade was a result of setting
       docsDevSwAdminStatus to upgradeFromMgt.
       Failed(4) indicates that the last attempted download
       failed, ordinarily due to TFTP timeout."

            ::= { cabhSecFwBase 4 }
```

```
cabhSecFwPolicyFileCurrentVersion OBJECT-TYPE
   SYNTAX      SnmpAdminString
   ~~--~~MAX-ACCESS  read-only
~~   -- Write access added to allow factory configuration~~
~~   MAX-ACCESS  read-write~~
   STATUS      current
   DESCRIPTION
         "The rule set version currently operating in the PS device.
          This object should be in the syntax used by the individual
          vendor to identify software versions.  Any PS element MUST
            return a string descriptive of the current rule set file load.
          If this is not applicable, this object MUST contain an empty
            string."
          ::= { cabhSecFwBase 5 }

--
--        Firewall log parameters
--



cabhSecFwEventType1Enable  OBJECT-TYPE
          SYNTAX          INTEGER          {
                    enable                         (1),       -- log event
                    disable                        (2)~~,~~      -- do not log event
                    }
          MAX-ACCESS read-write
          STATUS          current
          DESCRIPTION
~~ "Enables~~"This object enables or disables logging of type 1 firewall event
messages.~~"~~ Type 1 event
messages report attempts from both private and public clients to traverse
the
firewall that violate the Security Policy."

DEFVAL { disable }
          ::= { cabhSecFwLogCtl 1 }

cabhSecFwEventType2Enable  OBJECT-TYPE
          SYNTAX          INTEGER          {
                    enable                         (1),       -- log event
                    disable                        (2)~~,~~      -- do not log event
                    }
          MAX-ACCESS read-write
          STATUS current
          DESCRIPTION
~~ "Enables~~"This object enables or disables logging of type 2 firewall event
messages.~~"~~ Type 2 event messages report identified Denial of Service attack
attempts."

DEFVAL { disable }
          ::= { cabhSecFwLogCtl 2 }


cabhSecFwEventType3Enable  OBJECT-TYPE
```

```
            SYNTAX          INTEGER          {
                        enable                       (1),        -- log event
                        disable                      (2)~~,~~    -- do not log event
                        }
            MAX-ACCESS read-write
            STATUS current
            DESCRIPTION
                        "Enables or disables logging of type 3 firewall event messages.~~"~~ Type 3 event
messages report changes made to the following firewall management
parameters: cabhSecFwPolicyFileURL, cabhSecFwPolicyFileCurrentVersion,
cabhSecFwPolicyFileEnable"

DEFVAL { disable }
            ::= { cabhSecFwLogCtl 3 }

cabhSecFwEventAttackAlertThreshold  OBJECT-TYPE
            SYNTAX          INTEGER         (0..65535)
            MAX-ACCESS read-write
            STATUS current
            DESCRIPTION
                        "If the number of type 1 or 2 hacker attacks exceeds this
threshold
                        in the period define by cabhSecFwEventAttackAlertPeriod, a
firewall
                        message event MUST be logged with priority level 4."
DEFVAL { 65535 }
            ::= { cabhSecFwLogCtl 4 }


cabhSecFwEventAttackAlertPeriod          OBJECT-TYPE
            SYNTAX          INTEGER          (0..65535)
            MAX-ACCESS read-write
            STATUS current
            DESCRIPTION
                        "Indicates the period to be used (in ~~days~~hours) for the
                        cabhSecFwEventAttackAlertThreshold.~~"~~ This MIB variable should always keep
track of the last  x hours of events meaning that if the variable is set
to track events for 10 hours then when the 11th hour is reached, the 1st
hour of events is deleted from the tracking log. A default value is set
to zero, meaning zero time, so that this MIB variable will not track any
events unless configured."
DEFVAL {0}

            ::= { cabhSecFwLogCtl 5 }



cabhSecCertPsCert                    OBJECT-TYPE
            SYNTAX                  X509Certificate
            MAX-ACCESS  read-only
            STATUS                  current
            DESCRIPTION
~~ “~~"The X509 DER-encoded PS certificate.~~”~~"
            REFERENCE
~~ “Security”~~" Specification
 Section ~~x.x”~~11.3 Requirements (security requirements)"
```

```
          ::= { cabhSecCertObjects 1 }


--
-- notification group is for future extension.
--

cabhSecNotification OBJECT IDENTIFIER ::= { cabhSecMib 3 0 }
cabhSecConformance  OBJECT IDENTIFIER ::= { cabhSecMib 4 }
cabhSecCompliances  OBJECT IDENTIFIER ::= { cabhSecConformance 1 }
cabhSecGroups       OBJECT IDENTIFIER ::= { cabhSecConformance 2 }

--
--   Notification Group
--



-- compliance statements

cabhSecBasicCompliance MODULE-COMPLIANCE
   STATUS     current
   DESCRIPTION
         "The compliance statement for Cable Firewall feature."
   MODULE   --cabhSecMib





-- unconditionally mandatory groups

   MANDATORY-GROUPS {
         ~~cabhSecFwGroup~~
         cabhSecGroup
   }

::= { cabhSecCompliances 3 }


cabhSecGroup OBJECT-GROUP
   OBJECTS {
                    cabhSecFwPolicyFileEnable,
                    cabhSecFwPolicyFileURL,
                    cabhSecFwPolicyFileHash,
                    cabhSecFwPolicyFileOperStatus,
                    cabhSecFwPolicyFileCurrentVersion,

                    cabhSecFwEventType1Enable,
                    cabhSecFwEventType2Enable,
                    cabhSecFwEventType3Enable,
                    cabhSecFwEventAttackAlertThreshold,
                    cabhSecFwEventAttackAlertPeriod,
                    cabhSecCertPsCert
   }
```

```
    STATUS    current
    DESCRIPTION
            "Group of object in Cable Firewall MIB"
    ::= { cabhSecGroups 1 }

END
```

E.4 Definition ~~MIB~~

The Definition MIB MUST be implemented as defined below.

```
CLAB-DEF-MIB DEFINITIONS ::= BEGIN
IMPORTS
            MODULE-IDENTITY,
            X509Certificate           FROM DOCS-BPI2-MIB
            enterprises                                                  FROM SNMPv2-SMI;

cableLabs MODULE-IDENTITY
            LAST-UPDATED    "~~0201310000~~0209200000Z" -- ~~January 31,~~September 20, 2002
    ORGANIZATION    "CableLabs"
    CONTACT-INFO
           "Ralph Brown
           Postal: Cable Television Laboratories, Inc.
                           400 Centennial Parkway
                           Louisville, Colorado 80027-1266
                           U.S.A.
           Phone:  +1 303-661-9100
           Fax:    +1 303-661-9199
           E-mail: r.brown@cablelabs.com"
    DESCRIPTION
           "This MIB module supplies the basic management object categories for Cable
Labs."

    ::= { enterprises 4491 }

clabFunction              OBJECT IDENTIFIER ::= { cableLabs 1 }
clabFuncMib2              OBJECT IDENTIFIER ::= { clabFunction 1 }
clabFuncProprietary       OBJECT IDENTIFIER ::= { clabFunction 2 }
clabProject               OBJECT IDENTIFIER ::= { cableLabs 2 }
clabProjDocsis  OBJECT IDENTIFIER ::= { clabProject 1 }
clabProjPacketCable       OBJECT IDENTIFIER ::= { clabProject 2 }
clabProjOpenCable         OBJECT IDENTIFIER ::= { clabProject 3 }
clabProjCableHome         OBJECT IDENTIFIER ::= { clabProject 4 }
clabSecurity              OBJECT IDENTIFIER ::= { cableLabs 3}

clabSecCertObject OBJECT IDENTIFIER ::= { clabSecurity 1 }

clabSrvcPrvdrRootCACert         OBJECT-TYPE
    SYNTAX                    X509Certificate
    MAX-ACCESS                        read-only
    STATUS                    current
    DESCRIPTION
           "The X509 DER-encoded Service Provider Root CA Certificate."
```

```
  REFERENCE
  " Specification Section 11"
  ::= { clabSecCertObject 1 }

clabCVCRootCACert            OBJECT-TYPE
  SYNTAX             X509Certificate
  MAX-ACCESS                 read-only
  STATUS             current
  DESCRIPTION
        "The X509 DER-encoded CVC Root CA Certificate."
  REFERENCE
  " Specification Section 11 for Standalone PS Elements only"
  ::= { clabSecCertObject 2 }

clabCVCCACert                OBJECT-TYPE
  SYNTAX             X509Certificate
  MAX-ACCESS                 read-only
  STATUS             current
  DESCRIPTION
        "The X509 DER-encoded CableLabs CVC CA Certificate."
  REFERENCE
  " Specification Section 11 for Standalone PS Elements only"
  ::= { clabSecCertObject 3 }

clabMfgCVCCert               OBJECT-TYPE
  SYNTAX             X509Certificate
  MAX-ACCESS                 read-only
  STATUS             current
  DESCRIPTION
        "The X509 DER-encoded Manufacturer CVC Certificate."
  REFERENCE
  " Specification Section 11 for Standalone PS Elements only"
  ::= { clabSecCertObject 4 }

END
```

E.5 Cable DHCP Portal (CDP) MIB

The CDP MIB MUST be implemented as defined below.

```
CABH-CDP-MIB DEFINITIONS ::= BEGIN
IMPORTS
  MODULE-IDENTITY,
  OBJECT-TYPE,
       Integer32,
       Unsigned~~32~~32,
                                                   ~~FROM SNMPv2-SMI~~
TruthValue,
        TimeStamp,
       ~~DisplayString,~~   RowStatus,
  TEXTUAL-CONVENTION
                                             FROM SNMPv2-TC
```

```
    OBJECT-GROUP,
    MODULE-COMPLIANCE
                                                FROM SNMPv2-CONF

    InetAddressType,
    InetAddress,
    InetAddressIPv4,
    InetAddressIPv6
                                                         FROM INET-ADDRESS-
MIB
        SnmpAdminString            FROM SNMP-FRAMEWORK-MIB -- RFC2571
    clabProjCableHome
                                                FROM CLAB-DEF-MIB;

--================================================================
--
--      History:
--
--      Date           Modified by          Reason
--
~~--~~
--================================================================

cabhCdpMib MODULE-IDENTITY
    LAST-UPDATED    "~~0112190000Z" -- December 19, 2001~~0209200000Z" -- September 20, 2002
    ORGANIZATION    "~~Cable NMP Group"~~CableLabs Broadband Access Department"
    CONTACT-INFO
            "Kevin Luehrs
            Postal: Cable Television Laboratories, Inc.
                        400 Centennial Parkway
                      Louisville, Colorado 80027-1266
                        U.S.A.
            Phone:  +1 303-661-9100
            Fax:    +1 303-661-9199
            E-mail: k.luehrs@cablelabs.com"
    DESCRIPTION
             "This MIB module supplies the basic management objects
~~           for the CDP and the CAP portions of the PS database.~~

~~           Acknowledgements:~~
for the Cable DHCP Portal (CDP) portion of the PS database.



    ::=  { clabProjCableHome 4 }

-- Textual conventions
CabhCdpLanTransDhcpClientId ::= TEXTUAL-CONVENTION
        STATUS         current
        DESCRIPTION
                "LAN-Trans DHCP option61 information."
        SYNTAX OCTET STRING (SIZE (1..80))


~~--~~
~~--   assumes SNMPv3~~
~~--   SW load management is per DOCSIS 1.1 only~~
```

~~--~~

```
cabhCdpObjects                    OBJECT IDENTIFIER ::= { cabhCdpMib 1 }
cabhCdpBase                       OBJECT IDENTIFIER ::= { cabhCdpObjects 1 }
cabhCdpAddr                       OBJECT IDENTIFIER ::= { cabhCdpObjects 2 }
cabhCdpServer                     OBJECT IDENTIFIER ::= { cabhCdpObjects 3 }
--
--        The following group describes the base objects in the Cable
--        DHCP Portal.  The rest of this group deals addresses defined on
--        the LAN side.
--

cabhCdpSetToFactory   OBJECT-TYPE
           SYNTAX      TruthValue
   MAX-ACCESS  read-write
   STATUS      current
   DESCRIPTION
        "Setting this object to true(1) causes the DHCP default         options to be returned back to
factory defaults
        and all current mappings to use the factory ~~defaults and all        current mappings to use~~
~~the factory~~ default settings at            the next lease renewal time.  Reading this object
        always          returns false(2).  When ~~cabhCdpDhcpReset~~cabhCdpSetToFactory is set to
true,         the following actions occur:          1.          Reset all
        default CDS DHCP options to the factory            defaults.          2. The CDS will offer the
factory default DHCP options
                 at the next lease renewal time.

        The objects set to factory defaults are:
                          cabhCdpLanTransThreshold,
                 cabhCdpLanTransAction,
         cabhCdpWanDataIpAddrCount,
                 cabhCdpLanStartType,
                 cabhCdpLanPoolStart,
         cabhCdpLanPoolEndType,
                 cabhCdpLanPoolEnd,
         cabhCdpNetworkNumber,
                 cabhCdpServerSubnetMaskType,
                 cabhCdpServerSubnetMask,
                          cabhCdpServerTimeOffset,
         cabhCdpServerRouterType,
                 cabhCdpServerRouter,
                 cabhCdpServerDnsAddressType,
                 cabhCdpServerDnsAddress,
                 cabhCdpServerSyslogAddressType,
                 cabhCdpServerSyslogAddress,
                          cabhCdpServerDomainName,
                          cabhCdpServerTTL,
                          cabhCdpServerInterfaceMTU,
                          cabhCdpServerVendorSpecific,
                          cabhCdpServerLeaseTime,
                 cabhCdpServerDhcpAddressType,
                 cabhCdpServerDhcpAddress"
         REFERENCE
                 ""
   ::= { cabhCdpBase 1 }
```

```
cabhCdpLanTransCurCount OBJECT-TYPE
    SYNTAX      Unsigned32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "The current number of LAN-Trans IP addresses for
            Translated addresses (NAT and NAPT Interconnects).
            This is a count of ~~WAN~~LAN side addresses."
             REFERENCE
                    ""
    ::= { cabhCdpBase 2 }

cabhCdpLanTransThreshold OBJECT-TYPE
    SYNTAX      INTEGER (~~1..65533)~~0..65533)
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
             "The threshold number of LAN-Trans IP addresses             allocated or assigned above
which the PS generates an alarm             condition~~ MUST be generated. ~~. Whenever an attempt
is made to allocate ~~an~~
a LAN-~~TRANS~~Trans IP address when             cabhCdpLanTransCurCount is greater than or equal
to
cabhCdpLanTransThreshold, an event is generated. A value of 0 indicates that the
~~            For class C addresses 253 is used as default.  For~~
~~            class B addresses 65533 is used as a default.  In~~
~~            either case, this setting disables the feature."~~
~~             REFERENCE~~
~~                    ""~~
CDP sets the threshold at the highest number of addresses in the LAN address pool."

             DEFVAL { ~~65533 }~~0 }
    ::= { cabhCdpBase 3 }

cabhCdpLanTransAction OBJECT-TYPE
    SYNTAX      INTEGER {
             normal              (1),
                noAssignment            (2)
             }
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
            "The action taken when the CDS assigns a LAN-Trans address
            and the number of LAN-Trans addresses assigned
            (cabhCdpLanTransCurCount) is greater than the threshold
            (cabhCdpLanTransThreshold)  The actions are as follows:

                              normal -          assign a LAN-Trans IP address and treat the
                                                interconnection between the LAN and WAN as
                                                would normally occur if the threshold was not
                                                exceeded.


                              noAssignment - do not assign a LAN-Trans IP address and do
```

```
                                        not create an interconnection"
     REFERENCE
          ""
     DEFVAL { normal }
  ::= { cabhCdpBase 4 }

cabhCdpWanDataIpAddrCount OBJECT-TYPE
  SYNTAX      INTEGER ( 0..63 )
  MAX-ACCESS  read-write
  STATUS      current
  DESCRIPTION
       "This is the number of WAN-Data IP addresses that the CDC needs to acquire via DHCP."

     REFERENCE
          ""
     DEFVAL { 0 }
  ::= { cabhCdpBase 5 }

--
--       CDP Address Management Tables
--
--=====================================================================
--
--       cabhCdpLanAddrTable (CDP LAN Address Table)
--
--       The cabhCdpLanAddrTable contains the DHCP parameters
--       for each IP address served to the LAN-Trans realm.
--
--       This table contains a list of entries for the LAN side CDP parameters. These parameters can
be set
--       either by the CDP or by the cable operator through the CMP.
--
--=====================================================================

cabhCdpLanAddrTable OBJECT-TYPE
  SYNTAX      SEQUENCE OF CabhCdpLanAddrEntry
  MAX-ACCESS  not-accessible
  STATUS      current
  DESCRIPTION
       "This table is a list of LAN-Trans realm parameters. This
       list has one entry for each allocated LAN-Trans IP
       address."
  ::= {  cabhCdpAddr 1 }

cabhCdpLanAddrEntry OBJECT-TYPE
  SYNTAX      CabhCdpLanAddrEntry
  MAX-ACCESS  not-accessible
  STATUS      current
  DESCRIPTION
     "List of general parameter for CDP mappings."
  INDEX { cabhCdpLanAddrIpType, cabhCdpLanAddrIp }
  ::= { cabhCdpLanAddrTable 1 }

CabhCdpLanAddrEntry ::= SEQUENCE {
  cabhCdpLanAddrIpType                  InetAddressType,
     cabhCdpLanAddrIp                   InetAddress,
        cabhCdpLanAddrClientID                      CabhCdpLanTransDhcpClientId,
        ~~cabhCdpLanAddrCreateTime~~cabhCdpLanAddrLeaseCreateTime           TimeStamp,
        ~~cabhCdpLanAddrExpireTime~~cabhCdpLanAddrLeaseExpireTime TimeStamp,
        cabhCdpLanAddrMethod                        INTEGER,
        cabhCdpLanAddrHostName              ~~DisplayString,~~SnmpAdminString,
        cabhCdpLanAddrRowStatus             RowStatus
   }

cabhCdpLanAddrIpType OBJECT-TYPE
   SYNTAX      InetAddressType
   MAX-ACCESS  not-accessible
   STATUS      current
   DESCRIPTION
        "The address type assigned on the LAN side for the CDP Address       Table."
   DEFVAL  { ipv4 }
   ::= { cabhCdpLanAddrEntry 1 }



cabhCdpLanAddrIp OBJECT-TYPE
   SYNTAX      InetAddress
   MAX-ACCESS  not-accessible
   STATUS      current
   DESCRIPTION

"The address assigned on the LAN side for the CDP Address
    ~~Table."~~ Table. This parameter is entered by the CDP
when the CDS grants a lease to a LAN IP Device in the LAN-Trans realm and creates a row in this
table.
Alternatively, this parameter can be created by the NMS through the CMP, when the NMS creates a
new
DHCP address reservation by accessing the cabhCdpLanAddrRowStatus object with an index
comprised of
a new cabhCadpLanAddrIp and its Type."

   ::= { cabhCdpLanAddrEntry 2 }

cabhCdpLanAddrClientID OBJECT-TYPE
   SYNTAX      CabhCdpLanTransDhcpClientId
   MAX-ACCESS  read-~~only~~create
   STATUS      current
   DESCRIPTION
        "The client ID as indicated in Option 61 of the DHCP Discover.       There is a
        one-to-one relationship between the Client ID and the       assigned LAN address.
        This parameter is entered by the CDP when the CDS grants a lease to a LAN IP
        Device in the LANpTrans realm and creates a row in this table.  Alternatively,
        this parameter can be created by the NMS through the CMP, when the NMS creates
        a new DHCP address reservation by accessing the cabhCdpLanDataAddrRowStatus
        object with an index comprised of a new cabhCdpLanAddrIp and a new
        cabhCdpLanAddrClientID."

   ::= { cabhCdpLanAddrEntry 3 }

~~cabhCdpLanAddrCreateTime~~cabhCdpLanAddrLeaseCreateTime OBJECT-TYPE
   SYNTAX      TimeStamp
   MAX-ACCESS  read-only
```

```
    STATUS      current
    DESCRIPTION
       "The time the LAN side of the CDP LAN Table was created.
       This entry is only set the cabhCdpLanAddrTable
       entry is created and the entry does not already exist.  In
       other words, this value is not overwritten at lease renewal
       time."
    ::= { cabhCdpLanAddrEntry 4 }

~~cabhCdpLanAddrExpireTime~~cabhCdpLanAddrLeaseExpireTime OBJECT-TYPE
    SYNTAX      TimeStamp
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
       "This is the time that the LAN side lease expires.  When
       the lease expires this entry will be deleted from the table."
    ::= { cabhCdpLanAddrEntry 5 }

cabhCdpLanAddrMethod OBJECT-TYPE
    SYNTAX      INTEGER {
            cmp                        (1),
            cdp                        (2)
                                              }
            MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
       "The method that created this Address Entry.  cmp
       indicates that configuration through the CMP established this
       row (entry).  cdp indicates that a DHCP discover established
       this row (entry)."
    ::= { cabhCdpLanAddrEntry 6 }

cabhCdpLanAddrHostName OBJECT-TYPE
    SYNTAX      ~~DisplayString~~SnmpAdminString(SIZE(0..80))
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
       "This is the Host Name of the LAN IP address, based on DCHP option 12."
    ::= { cabhCdpLanAddrEntry 7 }


cabhCdpLanAddrRowStatus OBJECT-TYPE
    SYNTAX      RowStatus
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
       "The RowStatus interlock for creation and deletion."
    ::= { cabhCdpLanAddrEntry 8 }


--=====================================================================
--
--        cabhCdpWanDataAddrTable (CDP WAN-Data Address Table)
--
--        The cabhCdpWanDataAddrTable contains the configuration or DHCP parameters
--        for each IP address mapping per WAN-Data IP Address.
```

```
--
--=======================================================================

cabhCdpWanDataAddrTable OBJECT-TYPE
   SYNTAX      SEQUENCE OF CabhCdpWanDataAddrEntry
   MAX-ACCESS  not-accessible
   STATUS      current
   DESCRIPTION
         "This table contains WAN-Data address realm information."
   ::= {  cabhCdpAddr 2 }

cabhCdpWanDataAddrEntry OBJECT-TYPE
   SYNTAX      CabhCdpWanDataAddrEntry
   MAX-ACCESS  not-accessible
   STATUS      current
   DESCRIPTION
      "List of general parameter for CDP WAN-Data address realm."
   INDEX { cabhCdpWanDataAddrIndex }
   ::= { cabhCdpWanDataAddrTable 1 }

CabhCdpWanDataAddrEntry ::= SEQUENCE {
          cabhCdpWanDataAddrIndex                       INTEGER,
          cabhCdpWanDataAddrClientId                OCTET STRING,
          cabhCdpWanDataAddrIpType                    InetAddressType,
          cabhCdpWanDataAddrIp                                 InetAddress,
          cabhCdpWanDataAddrRenewalTime      Integer32,
          cabhCdpWanDataAddrRowStatus                   RowStatus
   }


cabhCdpWanDataAddrIndex OBJECT-TYPE
   SYNTAX      INTEGER (1..65535)
   MAX-ACCESS  not-accessible
   STATUS      current
   DESCRIPTION
     "Index into table."
   ::= { cabhCdpWanDataAddrEntry 1 }

cabhCdpWanDataAddrClientId OBJECT-TYPE
   SYNTAX OCTET STRING (SIZE (1..80))
   MAX-ACCESS  read-create
   STATUS      current
   DESCRIPTION
     "A unique WAN-Data ClientID used when ~~DHCPing~~attempting the acquire a WAN-Data IP
Address via DHCP."
   ::= { cabhCdpWanDataAddrEntry 2 }

cabhCdpWanDataAddrIpType OBJECT-TYPE
   SYNTAX      InetAddressType
   MAX-ACCESS  read-~~create~~only
   STATUS      current
   DESCRIPTION
     "The address type assigned on the WAN-Data side."
   DEFVAL { ipv4 }
   ::= { cabhCdpWanDataAddrEntry 3 }
```

```
cabhCdpWanDataAddrIp OBJECT-TYPE
   SYNTAX      InetAddress
   MAX-ACCESS  read-~~create~~only
   STATUS      current
   DESCRIPTION
     "The address assigned on the WAN-Data side."
   ::= { cabhCdpWanDataAddrEntry 4 }

cabhCdpWanDataAddrRenewalTime OBJECT-TYPE
   SYNTAX      Integer32
   MAX-ACCESS  read-~~create~~only
   STATUS      current
   DESCRIPTION
     "This is the time remaining before the lease expires.
     This is based on DHCP Option 51."
   ::= { cabhCdpWanDataAddrEntry 5 }

cabhCdpWanDataAddrRowStatus OBJECT-TYPE
   SYNTAX      RowStatus
   MAX-ACCESS  read-create
   STATUS      current
   DESCRIPTION
     "The RowStatus interlock for creation and deletion."
   ::= { cabhCdpWanDataAddrEntry 6 }

--=====================================================================
--
--         cabhCdpWanDataAddrServerTable (CDP WAN-Data DNS Server Table)
--
--         The cabhCdpWanDataAddrServerTable contains a table of referral DNS Servers.
--
--======================================================================

cabhCdpWanDataAddrServerTable OBJECT-TYPE
   SYNTAX      SEQUENCE OF CabhCdpWanDataAddrServerEntry
         MAX-ACCESS  not-accessible
   STATUS      current
   DESCRIPTION
        "This contains the IP addresses used for the WAN-Data DNS hosts
        obtained via the DHCP option 6 during the WAN-Data process."
   ::= {  cabhCdpAddr 3 }

cabhCdpWanDataAddrServerEntry OBJECT-TYPE
   SYNTAX      CabhCdpWanDataAddrServerEntry
   MAX-ACCESS  not-accessible
   STATUS      current
   DESCRIPTION
      "List of WAN-Data DNS Hosts."
   INDEX { cabhCdpWanDataAddrDnsIpType, cabhCdpWanDataAddrDnsIp }
   ::= { cabhCdpWanDataAddrServerTable 1 }

CabhCdpWanDataAddrServerEntry ::= SEQUENCE {
   cabhCdpWanDataAddrDnsIpType      InetAddressType,
   cabhCdpWanDataAddrDnsIp          InetAddress,
   cabhCdpWanDataAddrDnsRowStatus            RowStatus
   }
```

```
cabhCdpWanDataAddrDnsIpType OBJECT-TYPE
   SYNTAX      InetAddressType
            MAX-ACCESS  not-accessible
   STATUS      current
   DESCRIPTION
         "This parameter indicates the IP address type of a DNS server."
          DEFVAL  { ipv4 }
          ::= { cabhCdpWanDataAddrServerEntry 1 }


cabhCdpWanDataAddrDnsIp OBJECT-TYPE
   SYNTAX      InetAddress
            MAX-ACCESS  not-accessible
   STATUS      current
   DESCRIPTION
         "This parameter indicates the IP address of a DNS server."
          ::= { cabhCdpWanDataAddrServerEntry 2 }

cabhCdpWanDataAddrDnsRowStatus OBJECT-TYPE
   SYNTAX      RowStatus
   MAX-ACCESS  read-create
   STATUS      current
   DESCRIPTION
     "The RowStatus interlock for creation and deletion."
          ::= { cabhCdpWanDataAddrServerEntry 3 }


--
--        DHCP Server Side (CDS) Option Values for the LAN-Trans realm
--
cabhCdpLanPoolStartType OBJECT-TYPE
   SYNTAX      InetAddressType
   MAX-ACCESS  read-write
   STATUS      current
   DESCRIPTION
     "The Address type of the start of range LAN Trans IP Addresses."
          DEFVAL { ipv4 }
   ::= { cabhCdpServer 1 }

cabhCdpLanPoolStart OBJECT-TYPE
   SYNTAX      InetAddress
   MAX-ACCESS  read-write
   STATUS      current
   DESCRIPTION
     "The start of range LAN Trans IP Addresses."
          DEFVAL { 'c0a8000a'h }-- 192.168.0.10
                  -- 192.168.0.0 is the network number
                                                          -- 192.168.0.255 is broadcast
          -- address and 192.168.0.1
          -- is reserved for the router
   ::= { cabhCdpServer 2 }

cabhCdpLanPoolEndType OBJECT-TYPE
   SYNTAX      InetAddressType
```

```
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
      "The Address type of the end of range LAN Trans IP Addresses."
           DEFVAL { ipv4 }
    ::= { cabhCdpServer 3 }

cabhCdpLanPoolEnd OBJECT-TYPE
    SYNTAX      InetAddress
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
      "The end of range for LAN-Trans IP Addresses."
           DEFVAL { 'c0a800fe'h } -- 192.168.0.254
    ::= { cabhCdpServer 4 }

cabhCdpServerNetworkNumberType      OBJECT-TYPE
            SYNTAX              InetAddressType
MAX-ACCESS  read-write
STATUS              current
DESCRIPTION
  "The IP address type of the LAN-Trans network number."
DEFVAL  { ipv4 }
::= { cabhCdpServer 5 }

cabhCdpServerNetworkNumber          OBJECT-TYPE
SYNTAX              InetAddress
MAX-ACCESS  read-write
STATUS              current
DESCRIPTION
            "The LAN-Trans network number."
DEFVAL  { 'c0a80000'h }
::= { cabhCdpServer 6 }



cabhCdpServerSubnetMaskType OBJECT-TYPE
                SYNTAX      InetAddressType
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
           "Type of LAN-Trans Subnet Mask."
    DEFVAL { ipv4 }
    ::= { cabhCdpServer ~~5 }~~7 }

cabhCdpServerSubnetMask      OBJECT-TYPE
                SYNTAX      InetAddress
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
           "Option value 1 - Value of LAN-Trans Subnet Mask."
            DEFVAL { 'ffffff00'h }     -- 255.255.255.0
    ::= { cabhCdpServer ~~6 }~~8 }

cabhCdpServerTimeOffset        OBJECT-TYPE
                SYNTAX      Integer32 (-86400..86400)  -- 0 to 24 hours (in seconds)
```

```
          UNITS              "seconds"
   MAX-ACCESS  read-write
   STATUS      current
   DESCRIPTION
         "Option value 2 - Value of LAN-Trans Time Offset from
         Coordinated Universal Time (UTC)."
          DEFVAL { 0 }     -- UTC
   ::= { cabhCdpServer ~~7 }~~9 }

cabhCdpServerRouterType       OBJECT-TYPE
          SYNTAX      InetAddressType
   MAX-ACCESS  read-write
   STATUS      current
   DESCRIPTION
         "Type of Address, Router for the LAN-Trans
         address realm."
          DEFVAL { ipv4 }
   ::= { cabhCdpServer ~~8 }~~10 }

cabhCdpServerRouter   OBJECT-TYPE
          SYNTAX      InetAddress
   MAX-ACCESS  read-write
   STATUS      current
   DESCRIPTION
         "Option value 3 - Router for the LAN-Trans
         address realm."
          DEFVAL { 'c0a80001'h }-- 192.168.0.1
   ::= { cabhCdpServer ~~9 }~~11 }

cabhCdpServerDnsAddressType OBJECT-TYPE
   SYNTAX      InetAddressType
   MAX-ACCESS  read-write
   STATUS      current
   DESCRIPTION
     "The Type of IP Addresses of the LAN-Trans address realm
     DNS servers."
          DEFVAL { ipv4 }
   ::= { cabhCdpServer ~~10 }~~12 }

cabhCdpServerDnsAddress OBJECT-TYPE
   SYNTAX      InetAddress
   MAX-ACCESS  read-write
   STATUS      current
   DESCRIPTION
     "The IP Addresses of the LAN-Trans address realm
     DNS servers.  As a default there is only one DNS
     server and it is the address specified in Option
     Value 3 - cabhCdpServerRouter.  Only one address
     is specified."
          DEFVAL { 'c0a80001'h }-- 192.168.0.1
   ::= { cabhCdpServer ~~11 }~~13 }

cabhCdpServerSyslogAddressType OBJECT-TYPE
   SYNTAX      InetAddressType
   MAX-ACCESS  read-write
   STATUS      current
   DESCRIPTION
     "The Type of IP Address of the LAN-Trans SYSLOG servers."
          DEFVAL { ipv4 }
   ::= { cabhCdpServer ~~12 }~~14 }

cabhCdpServerSyslogAddress OBJECT-TYPE
   SYNTAX      InetAddress
   MAX-ACCESS  read-write
   STATUS      current
   DESCRIPTION
     "The IP Addresses of the LAN-Trans SYSLOG servers.
     As a default there are no SYSLOG Servers.
     The factory defaults contains the indication of
     no Syslog Server value equals (0.0.0.0)."
          DEFVAL { '00000000'h }-- 0.0.0.0
   ::= { cabhCdpServer ~~13 }~~15 }

cabhCdpServerDomainName    OBJECT-TYPE
             SYNTAX      ~~DisplayString~~SnmpAdminString(SIZE(0..128))
   MAX-ACCESS  read-write
   STATUS      current
   DESCRIPTION
         "Option value 15 - Domain name of LAN-Trans address realm."
          DEFVAL {""}
          ::= { cabhCdpServer ~~14 }~~16 }

cabhCdpServerTTL       OBJECT-TYPE
          SYNTAX      INTEGER (0..255)
          MAX-ACCESS  read-write
   STATUS      current
   DESCRIPTION
         "Option value 23 - LAN-Trans Time to Live."
          DEFVAL  { 64 }
          ::= { cabhCdpServer ~~15 }~~17 }

cabhCdpServerInterfaceMTU    OBJECT-TYPE
          SYNTAX      INTEGER (68..4096)
          MAX-ACCESS  read-write
   STATUS      current
   DESCRIPTION
         "Option value 26 - LAN-Trans Interface MTU."
          ~~DEFVAL  { 1500 }~~          ::= { cabhCdpServer ~~16 }~~18 }

cabhCdpServerVendorSpecific   OBJECT-TYPE
          SYNTAX      OCTET STRING (SIZE(0..255))
          MAX-ACCESS  read-write
   STATUS      current
   DESCRIPTION
         "Option value 43 - Vendor Specific Options."
          DEFVAL  { ''h }
          ::= { cabhCdpServer ~~17 }~~19 }

cabhCdpServerLeaseTime        OBJECT-TYPE
          SYNTAX      Unsigned32
          UNITS            "seconds"
          MAX-ACCESS  read-write
   STATUS      current
   DESCRIPTION
             "Option value 51 ~~- LAN-Trans default ~~-Lease Time for LAN IP Devices in the LAN-Trans
realm (seconds)."
            DEFVAL  { ~~60~~3600 }

            ::= { cabhCdpServer ~~18 }~~20 }

cabhCdpServerDhcpAddressType         OBJECT-TYPE
            SYNTAX      InetAddressType
            MAX-ACCESS  read-write
   STATUS      current
   DESCRIPTION
           "Option value 54 - Type of LAN-Trans DHCP server IP address."
            DEFVAL { ipv4 }
            ::= { cabhCdpServer ~~19 }~~21 }

cabhCdpServerDhcpAddress     OBJECT-TYPE
            SYNTAX      InetAddress
            MAX-ACCESS  read-write
   STATUS      current
   DESCRIPTION
           "Option value 54 - LAN-Trans DHCP server IP
           address. It defaults to the router address as
           specified in cabhCdpServerRouter.  Alternatively
           a vendor may want to separate CDS address from
           router address."
            DEFVAL { 'c0a80001'h }            --          192.168.0.1
            ::= { cabhCdpServer ~~20 }~~22 }


--
-- notification group is for future extension.
--

cabhCdpNotification OBJECT IDENTIFIER ::= { cabhCdpMib 2 0 }
cabhCdpConformance  OBJECT IDENTIFIER ::= { cabhCdpMib 3 }
cabhCdpCompliances  OBJECT IDENTIFIER ::= { cabhCdpConformance 1 }
cabhCdpGroups       OBJECT IDENTIFIER ::= { cabhCdpConformance 2 }

--
--   Notification Group
--


-- compliance statements

cabhCdpBasicCompliance MODULE-COMPLIANCE
   STATUS     current
   DESCRIPTION
           "The compliance statement for devices that implement
            MTA feature."
   MODULE   --cabhCdpMib
```

```
-- unconditionally mandatory groups

    MANDATORY-GROUPS {
            cabhCdpGroup
    }

::= { cabhCdpCompliances 3 }


cabhCdpGroup              OBJECT-GROUP

    OBJECTS {

             cabhCdpSetToFactory,
               cabhCdpLanTransCurCount,
             cabhCdpLanTransThreshold,
             cabhCdpLanTransAction,

~~               cabhCdpLanAddrIpType,~~
~~               cabhCdpLanAddrIp,~~
cabhCdpWanDataIpAddrCount,

               cabhCdpLanAddrClientID,
~~               cabhCdpLanAddrCreateTime,~~
~~               cabhCdpLanAddrExpireTime,~~
cabhCdpLanAddrLeaseCreateTime,
cabhCdpLanAddrLeaseExpireTime,
               cabhCdpLanAddrMethod,
               cabhCdpLanAddrHostName,
               cabhCdpLanAddrRowStatus,

~~               cabhCdpWanDataAddrIndex,~~

               cabhCdpWanDataAddrClientId,
~~               cabhCdpLanAddrIpType,~~
               cabhCdpWanDataAddrIp,
               cabhCdpWanDataAddrRenewalTime,
               cabhCdpWanDataAddrRowStatus,

~~               cabhCdpWanDataAddrDnsIpType,~~
~~               cabhCdpWanDataAddrDnsIp,~~

               cabhCdpWanDataAddrDnsRowStatus,


               cabhCdpLanPoolStartType,
               cabhCdpLanPoolStart,
               cabhCdpLanPoolEndType,
               cabhCdpLanPoolEnd,
cabhCdpServerNetworkNumberType,
cabhCdpServerNetworkNumber,
               cabhCdpServerSubnetMaskType,
               cabhCdpServerSubnetMask,
              cabhCdpServerTimeOffset,


              cabhCdpServerRouterType,
              cabhCdpServerRouterType,
              cabhCdpServerRouter,
              cabhCdpServerDnsAddressType,
              cabhCdpServerDnsAddress,
              cabhCdpServerSyslogAddressType,
              cabhCdpServerSyslogAddress,
              cabhCdpServerDomainName,
              cabhCdpServerTTL,
              cabhCdpServerInterfaceMTU,
              cabhCdpServerVendorSpecific,
              cabhCdpServerLeaseTime,
              cabhCdpServerDhcpAddressType,
              cabhCdpServerDhcpAddress
                   }
   STATUS    current
   DESCRIPTION
      "Group of objects for Cable ~~CDB~~CDP MIB."
   ::= { cabhCdpGroups 1 }



END
```

E.6 Cable Address Portal

The CAP MIB MUST be implemented as defined below.

```
CABH-CAP-MIB DEFINITIONS ::= BEGIN
IMPORTS
   MODULE-IDENTITY,
   OBJECT-TYPE,
          Unsigned32
                                                  FROM SNMPv2-SMI
          TimeStamp,
          TruthValue,
          RowStatus,
          PhysAddress
                                       FROM SNMPv2-TC
   OBJECT-GROUP,
   MODULE-COMPLIANCE                                                        FROM
SNMPv2-CONF
   InetAddressType,
   InetAddress,
   InetAddressIPv4,
   InetAddressIPv6                                                          FROM
INET-ADDRESS-MIB

   clabProjCableHome
                                         FROM CLAB-DEF-MIB;

--===================================================================
--
```

```
--         History:
--
--         Date              Modified by       Reason
--
--
--=====================================================================

cabhCapMib MODULE-IDENTITY
    LAST-UPDATED    "~~0112190000Z" -- December 19, 2001~~0209200000Z" --September 20, 2002
    ORGANIZATION    "~~Cable NMP Group"~~CableLabs Broadband Access Department"
    CONTACT-INFO
            "Kevin Luehrs
            Postal: Cable Television Laboratories, Inc.
                            400 Centennial Parkway
                            Louisville, Colorado 80027-1266
                            U.S.A.
            Phone:  +1 303-661-9100
            Fax:    +1 303-661-9199
            E-mail: k.luehrs@cablelabs.com"
    DESCRIPTION
            "This MIB module supplies the basic management objects for the Cable
            ~~for the CDP and the CAP portions of the PS database.~~

        ~~Acknowledgements:~~
        ~~"~~
           Addressing Portal (CAP) portion of the PS database.



    ::=  { clabProjCableHome 3 }

-- Textual conventions

CabhCapPacketMode ::= TEXTUAL-CONVENTION
           STATUS          current
           DESCRIPTION
                    "The data type established when
                    a binding/mapping is established."
           SYNTAX          INTEGER {
                                napt            (1),      -- NAT with port translation
                                nat             (2),      -- Basic NAT
                                passthrough     (3)~~,~~      -- Pass Through External Address
                                }

~~--~~
~~--   assumes SNMPv3~~
~~--   SW load management is per DOCSIS 1.1 only~~
~~--~~


cabhCapObjects      OBJECT IDENTIFIER ::= { cabhCapMib 1 }
cabhCapBase         OBJECT IDENTIFIER ::= { cabhCapObjects 1 }
cabhCapMap          OBJECT IDENTIFIER ::= { cabhCapObjects 2 }


--=================================================================
--
--         General CAP Parameters
--
--================================================================

cabhCapTcpTimeWait OBJECT-TYPE
   SYNTAX      Unsigned32
~~          UNITS~~
   UNITS "seconds~~"~~
~~   MAX-ACCESS  read-write~~"
   MAX-ACCESS read-write
   STATUS      current
   DESCRIPTION
         "~~The~~This object is the maximum inactivity time to wait before assuming TCP
         session is terminated.~~"~~ It has no relation to the TCP session TIME_WAIT
~~          REFERENCE~~
~~                ""~~
         state referred to in [RFC793]"
            DEFVAL { ~~240 }             -- 4 minutes~~300 }
   ::= { cabhCapBase 1 }


cabhCapUdpTimeWait OBJECT-TYPE
          SYNTAX      Unsigned32
           UNITS    "seconds"
          MAX-ACCESS  read-write
          STATUS      current
          DESCRIPTION
         "The ~~maximum~~inactivity time to wait before ~~assuming UDP~~destroying
~~         session is terminated."~~
~~          REFERENCE~~
~~                ""~~
         CAP mappings for UDP."
          DEFVAL { 300 }  -- 5 minutes
          ~~DEFVAL { 86400 }         -- 1 day~~    ::= { cabhCapBase 2 }

cabhCapIcmpTimeWait OBJECT-TYPE
          SYNTAX      Unsigned32
           UNITS           "seconds"
          MAX-ACCESS  read-write
          STATUS      current
          DESCRIPTION
         "The ~~maximum~~inactivty time to wait before ~~assuming Icmp~~destroying
~~         session is terminated."~~
~~          REFERENCE~~
~~                ""~~
         CAP mappings for ICMP."
          DEFVAL { 300 }  --  5 minutes
          ~~DEFVAL { 86400 }         -- 1 day~~    ::= { cabhCapBase 3 }


cabhCapPrimaryMode OBJECT-TYPE
   SYNTAX       CabhCapPacketMode
   MAX-ACCESS  read-write
   STATUS      current
   DESCRIPTION
```

```
        "The Primary Packet Handling Mode to be used."
           DEFVAL { napt }
   ::= { cabhCapBase 4 }

cabhCapSetToFactory OBJECT-TYPE
           SYNTAX      TruthValue
   MAX-ACCESS  read-write
   STATUS      current
   DESCRIPTION
        "Setting this object to true(1) causes the all the tables in the CAP
        to be cleared, and all CAP objects with defaults to be reset back to
        their default values.

           The objects to set to factory default values when this object is set to 'true' are
           listed below:
           cabhCapTcpTimeWait,
           cabhCapUdpTimeWait,
           cabhCapIcmpTimeWait,
           cabhCapPrimaryMode,
           cabhCapMappingWanAddrType,
           cabhCapMappingWanPort,
           cabhCapMappingLanAddrType,
           cabhCapMappingLanPort
           "
   ::= { cabhCapBase 5 }


--=====================================================================
--
--        cabhCapMappingTable (CAP Mapping Table)
--
--        The cabhCapMappingTable contains the mappings for all CAP mappings.
--
--======================================================================

cabhCapMappingTable OBJECT-TYPE
   SYNTAX      SEQUENCE OF CabhCapMappingEntry
   MAX-ACCESS  not-accessible
   STATUS      current
   DESCRIPTION
        "This table contains IP address mapping for all CAP mappings."
   ::= {  cabhCapMap 1 }

cabhCapMappingEntry OBJECT-TYPE
   SYNTAX      CabhCapMappingEntry
   MAX-ACCESS  not-accessible
   STATUS      current
   DESCRIPTION
      "List of CAP IP mappings."
```
~~INDEX { cabhCapMappingWanAddrType, cabhCapMappingWanAddr, cabhCapMappingWanPort,~~
~~cabhCapMappingLanAddrType, cabhCapMappingLanAddr, cabhCapMappingLanPort}~~
```
   INDEX { cabhCapMappingIndex }
   ::= { cabhCapMappingTable 1 }

        CabhCapMappingEntry ::= SEQUENCE {
        cabhCapMappingWanAddrType                    InetAddressType,
        cabhCapMappingIndex                 INTEGER,
        cabhCapMappingWanAddr                       InetAddress,
        cabhCapMappingWanPort                       INTEGER,
        cabhCapMappingLanAddrType           InetAddressType,
        cabhCapMappingLanAddr                       InetAddress,
        cabhCapMappingLanPort                       INTEGER,
        ~~cabhCapMappingMode        CabhCapPacketMode,~~    cabhCapMappingMethod
                INTEGER,
        cabhCapMappingProtocol                      INTEGER,
        cabhCapMappingRowStatus             RowStatus
        }


cabhCapMappingIndex    OBJECT-TYPE
  SYNTAX          INTEGER   (1..65535)
  MAX-ACCESS  not-accessible
  STATUS      current
  DESCRIPTION
    "The Index into the CAP Mapping Table."
  ::= { cabhCapMappingEntry 1 }

cabhCapMappingWanAddrType OBJECT-TYPE
  SYNTAX      InetAddressType
  MAX-ACCESS  ~~not-accessible~~read-create
  STATUS      current
  DESCRIPTION
    "The IP address type assigned on the WAN side.  IP version
      4       is typically used."
  DEFVAL { ipv4 }
  ::= { cabhCapMappingEntry ~~1 }~~2 }

cabhCapMappingWanAddr OBJECT-TYPE
  SYNTAX      InetAddress
  MAX-ACCESS  ~~not-accessible~~read-create
  STATUS      current
  DESCRIPTION
    "The IP address assigned on the WAN side.  IP version 4
    is typically used."
  ::= { cabhCapMappingEntry ~~2 }~~3 }

cabhCapMappingWanPort OBJECT-TYPE
  SYNTAX      INTEGER (~~1..65535)~~0..65535)
  MAX-ACCESS  ~~not-accessible~~read-create
  STATUS      current
  DESCRIPTION
    "The TCP/UDP port number on the WAN side."
    DEFVAL { 0 }
  ::= { cabhCapMappingEntry ~~3 }~~4 }

cabhCapMappingLanAddrType OBJECT-TYPE
  SYNTAX      InetAddressType
  MAX-ACCESS  ~~not-accessible~~read-create
  STATUS      current
  DESCRIPTION
    "The IP address type assigned on the LAN side.  IP version
     4       is typically used."
  DEFVAL { ipv4 }
  ::= { cabhCapMappingEntry ~~4 }~~5 }

cabhCapMappingLanAddr OBJECT-TYPE
  SYNTAX      InetAddress
  MAX-ACCESS  ~~not-accessible~~read-create
  STATUS      current
  DESCRIPTION
    "The IP address assigned on the LAN side.  IP version 4
    is typically used."
  ::= { cabhCapMappingEntry ~~5 }~~6 }

cabhCapMappingLanPort OBJECT-TYPE
  SYNTAX      INTEGER (~~1..65535)~~0..65535)
  MAX-ACCESS  ~~not-accessible~~read-create
  STATUS      current
  DESCRIPTION
    "The TCP/UDP port number on the LAN side."
  ~~    ::= { cabhCapMappingEntry 6 }~~

~~cabhCapMappingMode OBJECT-TYPE~~
  ~~    SYNTAX      CabhCapPacketMode~~
  ~~    MAX-ACCESS  read-only~~
  ~~    STATUS      current~~
  ~~    DESCRIPTION~~
    ~~       "The type of packet handling mode for this mapping.  Note, this information could~~
    ~~       be gleaned from the IP address and Port information for this mapping"~~
  DEFVAL { 0 }
  ::= { cabhCapMappingEntry 7 }


cabhCapMappingMethod OBJECT-TYPE
  SYNTAX      INTEGER {
                                            static   (1),
               dynamic             (2)~~,~~
                                            }
  MAX-ACCESS  read-only
  STATUS      current
  DESCRIPTION
    "Indicates how this mapping was created.  Static means that it was provisioned,
and dynamic
    means that it was handled by the PS itself."
  ::= { cabhCapMappingEntry 8 }

cabhCapMappingProtocol OBJECT-TYPE
  SYNTAX      INTEGER {
               other              (1),       -- not specified
               icmp               (2),
               udp                (3),
               tcp                (4)
                                  }
  MAX-ACCESS  read-~~only~~create
  STATUS      current
  DESCRIPTION
    "The protocol for this mapping."
  ::= { cabhCapMappingEntry 9 }
```

```
cabhCapMappingRowStatus OBJECT-TYPE
    SYNTAX      RowStatus
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
"The RowStatus interlock for the creation and deletion of a cabhCapMappingTable entry.
Changing the value of the IP address or port number columns of the CAP Mapping Table
may have an effect on active traffic, so the CMP will prevent modification of this
table's columns when the cabhCapMappingRowStatus object is in the active state."
    ::={ cabhCapMappingEntry 10 }


--=====================================================================
--
--          cabhCapPassthroughTable (CAP Passthrough Table)
--
--          The cabhCapPassthroughTable contains the MAC Addresses for all LAN-IP Devices
--              which will be configured as passthrough.
--
--=====================================================================

cabhCapPassthroughTable OBJECT-TYPE
    SYNTAX      SEQUENCE OF CabhCapPassthroughEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "This table contains MAC addresses for LAN-IP Devices which are configured
                as passthrough mode."
    ::= {  cabhCapMap 2 }

cabhCapPassthroughEntry         OBJECT-TYPE
    SYNTAX       CabhCapPassthroughEntry
    MAX-ACCESS           not-accessible
    STATUS       current
    DESCRIPTION
            "List of ~~MAC~~hardware addresses ~~for~~of LAN~~-~~ IP Devices which are configured ~~as~~for
                    passthrough mode."
~~    INDEX {cabhCapPassthroughMACAddr }~~
    INDEX {cabhCapPassthroughIndex}
    ::= { cabhCapPassthroughTable 1 }


CabhCapPassthroughEntry ::= SEQUENCE {
~~            cabhCapPassthroughMACAddr               PhysAddress,~~
            cabhCapPassthroughIndex                 INTEGER,
            cabhCapPassthroughMacAddr   PhysAddress,
            cabhCapPassthroughRowStatus                         RowStatus
            }

cabhCapPassthroughIndex                 OBJECT-TYPE
    SYNTAX       INTEGER (1..65535)
    MAX-ACCESS           not-accessible
    STATUS       current
    DESCRIPTION
            "The index into the CAP Passthrough Table."
    ::= { cabhCapPassthroughEntry 1 }

~~cabhCapPassthroughMACAddr~~ cabhCapPassthroughMacAddr  OBJECT-TYPE
    SYNTAX       PhysAddress
    MAX-ACCESS          ~~not-accessible~~read-create
    STATUS       current
    DESCRIPTION
                        "~~MAC Address~~Hardware address of the LAN-IP Device to be configured as
passthrough
             mode."
    ::={cabhCapPassthroughEntry 2}

cabhCapPassthroughRowStatus            OBJECT-TYPE
    SYNTAX       RowStatus
    MAX-ACCESS          read-create
    STATUS              current
    DESCRIPTION
            "The RowStatus interlock for the creation and deletion of a
             cabhCapPassthroughTable entry.
             There are no restrictions on setting the read-create column of this table
             (i.e., cabhCapPassthroughMacAddr ) when the status of
                  cabhCapPassthroughRowStatus is active."
    ::= {~~cabhCapPassthroughEntry 1 }~~

~~cabhCapPassthroughRowStatus OBJECT-TYPE~~
~~    SYNTAX      RowStatus~~
~~    MAX-ACCESS  read-create~~
~~    STATUS      current~~
~~    DESCRIPTION~~
~~       "The RowStatus interlock for creation and deletion~~
~~       of cabhCapPassthroughTable entry."    ::= { cabhCapPassthroughEntry 2~~
cabhCapPassthroughEntry 3 }

--
-- notification group is for future extension.
--

cabhCapNotification      OBJECT IDENTIFIER ::= { cabhCapMib 2 0 }
cabhCapConformance   OBJECT IDENTIFIER ::= { cabhCapMib 3 }
cabhCapCompliances    OBJECT IDENTIFIER ::= { cabhCapConformance 1 }
cabhCapGroups                     OBJECT IDENTIFIER ::= { cabhCapConformance 2 }

--
--    Notification Group
--


-- compliance statements

cabhCapBasicCompliance MODULE-COMPLIANCE
    STATUS      current
    DESCRIPTION
            "The compliance statement for devices that implement
             MTA feature."
    MODULE    --cabhCapMib
```

```
-- unconditionally mandatory groups

   MANDATORY-GROUPS {
         cabhCapGroup
   }

::= { cabhCapCompliances ~~3 }~~1 }


cabhCapGroup OBJECT-GROUP
   OBJECTS {
                    cabhCapTcpTimeWait,
                    cabhCapUdpTimeWait,
                    cabhCapIcmpTimeWait,
                    cabhCapPrimaryMode,

~~--~~
                    cabhCapMappingWanAddrType,
~~--~~
                    cabhCapMappingWanAddr,
~~--~~
                    cabhCapMappingWanPort,
~~--~~
                    cabhCapMappingLanAddrType,
~~--~~
                    cabhCapMappingLanAddr,
~~--~~
                    cabhCapMappingLanPort,
~~                  cabhCapMappingMode,~~
                    cabhCapMappingMethod,
                    cabhCapMappingProtocol,

                    cabhCapMappingRowStatus,
~~--~~              cabhCapPassthroughMacAddr,
                    cabhCapPassthroughRowStatus
                    }
   STATUS    current
   DESCRIPTION
      "Group of objects for ~~CDB~~CAP MIB."
   ::= { cabhCapGroups 1 }


END
```

Legend:	
Insertion	
~~Deletion~~	
~~Moved from~~	
Moved to	
Style change	
Format change	
~~Moved deletion~~	
Inserted cell	🟦
Deleted cell	🟥
Moved cell	🟪
Split/Merged cell	🟨
Padding cell	⬜