International Telecommunication Union

ITU-T

TELECOMMUNICATION
STANDARDIZATION SECTOR
OF ITU

J.192

(11/2005)

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

Cable modems

A residential gateway to support the delivery of cable data services

ITU-T Recommendation J.192

ITU-T Recommendation J.192

A residential gateway to support the delivery of cable data services

Summary

This Recommendation describes a Residential Gateway by providing a set of IP-based features that may be added to a Cable Modem or incorporated into a standalone device. This will enable cable operators to provide an additional set of enhanced home network-based services to their customers including support for Quality of Service (QoS), device and service discovery, enhanced security, firewall management, home network focused management and provisioning features, managed network address translation, improved addressing and packet handling and LAN device diagnostics. This Recommendation is based upon the architectural frameworks defined in ITU-T Rec. J.190.

This Recommendation represents an enhancement to ITU-T Rec. J.191, retaining a majority of J.191 functionality as a foundation, and building upon this base to provide additional advanced features. A key design goal for equipment conforming to this Recommendation is interoperability with equipment conforming to ITU-T Rec. J.191. For example, common MIBs are used for the foundational functionality. As a result, a J.192-based headend may manage a mixed J.191 and J.192 deployment.

Source

ITU-T Recommendation J.192 was approved on 29 November 2005 by ITU-T Study Group 9 (2005-2008) under the ITU-T Recommendation A.8 procedure.

FOREWORD

The International Telecommunication Union (ITU) is the United Nations specialized agency in the field of telecommunications. The ITU Telecommunication Standardization Sector (ITU-T) is a permanent organ of ITU. ITU-T is responsible for studying technical, operating and tariff questions and issuing Recommendations on them with a view to standardizing telecommunications on a worldwide basis.

The World Telecommunication Standardization Assembly (WTSA), which meets every four years, establishes the topics for study by the ITU-T study groups which, in turn, produce Recommendations on these topics.

The approval of ITU-T Recommendations is covered by the procedure laid down in WTSA Resolution 1.

In some areas of information technology which fall within ITU-T's purview, the necessary standards are prepared on a collaborative basis with ISO and IEC.

NOTE

In this Recommendation, the expression "Administration" is used for conciseness to indicate both a telecommunication administration and a recognized operating agency.

Compliance with this Recommendation is voluntary. However, the Recommendation may contain certain mandatory provisions (to ensure e.g. interoperability or applicability) and compliance with the Recommendation is achieved when all of these mandatory provisions are met. The words "shall" or some other obligatory language such as "must" and the negative equivalents are used to express requirements. The use of such words does not suggest that compliance with the Recommendation is required of any party.

INTELLECTUAL PROPERTY RIGHTS

ITU draws attention to the possibility that the practice or implementation of this Recommendation may involve the use of a claimed Intellectual Property Right. ITU takes no position concerning the evidence, validity or applicability of claimed Intellectual Property Rights, whether asserted by ITU members or others outside of the Recommendation development process.

As of the date of approval of this Recommendation, ITU had not received notice of intellectual property, protected by patents, which may be required to implement this Recommendation. However, implementors are cautioned that this may not represent the latest information and are therefore strongly urged to consult the TSB patent database.

CONTENTS

ITU-T Recommendation J.192

A residential gateway to support the delivery of cable data services

1 Scope

This Recommendation describes a Residential Gateway by providing a set of IP-based features that may be added to a Cable Modem or incorporated into a standalone device. This will enable cable operators to provide an additional set of enhanced home network-based services to their customers including support for Quality of Service (QoS), device and service discovery, enhanced security, firewall management, home network focused management and provisioning features, managed network address translation, improved addressing and packet handling and LAN device diagnostics. This Recommendation is based upon the architectural frameworks defined in ITU-T Rec. J.190.

This Recommendation represents an enhancement to ITU-T Rec. J.191, retaining a majority of J.191 functionality as a foundation, and building upon this base to provide additional advanced features. A key design goal for equipment conforming to this Recommendation is interoperability with equipment conforming to ITU-T Rec. J.191. For example, common MIBs are used for the foundational functionality. As a result, a J.192-based headend may manage a mixed J.191 and J.192 deployment.

The key functionality that this Recommendation defines in addition to that defined by ITU-T Rec. J.191 includes:

- Device and service discovery for applications and services on the LAN;
- NAT support for IPSec VPN clients and home based servers;
- Standardized firewall configuration language and reporting;
- Standardized baseline firewall functionality;
- Simple parental control;
- Quality of Service for the LAN, managed at the Residential Gateway.

Non-normative text referring to UPnP functionality is contained within this Recommendation as example implementations of home networking QoS and Management, and has been enclosed in brackets and marked as follows: "{informative text: … }". All text included within these brackets is non-normative.

2 References

2.1 References (normative)

The following ITU-T Recommendations and other references contain provisions which, through reference in this text, constitute provisions of this Recommendation. At the time of publication, the editions indicated were valid. All Recommendations and other references are subject to revision; users of this Recommendation are therefore encouraged to investigate the possibility of applying the most recent edition of the Recommendations and other references listed below. A list of the currently valid ITU-T Recommendations is regularly published. The reference to a document within this Recommendation does not give it, as a stand-alone document, the status of a Recommendation.

– ITU-T Recommendation J.112 Annex B (2004), *Data-over-cable service interface specifications: Radio-frequency interface specification*.

– ITU-T Recommendation J.125 (2004), *Link privacy for cable modem implementations*.

– ITU-T Recommendation J.126 (2004), *Embedded Cable Modem device specification.*

– ITU-T Recommendation J.161 (2001), *Audio codec requirements for the provision of bidirectional audio service over cable television networks using cable modems*.

– ITU-T Recommendation J.162 (2005), *Network call signalling protocol for the delivery of time-critical services over cable television networks using cable modems*.

– ITU-T Recommendation J.163 (2005), *Dynamic quality of service for the provision of real-time services over cable television networks using cable modems*.

– ITU-T Recommendation J.164 (2005), *Event message requirements for the support of real-time services over cable television networks using cable modems*.

– ITU-T Recommendation J.167 (2005), *Media terminal adapter (MTA) device provisioning requirements for the delivery of real-time services over cable television networks using cable modems*.

– ITU-T Recommendation J.170 (2005), *IPCablecom security specification*.

– ITU-T Recommendation J.175 (2005), *Audio server protocol*.

– ITU-T Recommendation J.178 (2005), *IPCablecom CMS to CMS signalling*.

– ITU-T Recommendation J.191 (2004), *IP feature package to enhance cable modems*.

– ITU-T Recommendation X.690 (2002) | ISO/IEC 8825-1:2002, *Information technology – ASN.1 encoding rules: Specification of Basic Encoding Rules (BER), Canonical Encoding Rules (CER) and Distinguished Encoding Rules (DER)*.

– ITU-T Recommendation X.509 (2005) | ISO/IEC 9594-8:2005, *Information technology – Open Systems Interconnection – The Directory: Public-key and attribute certificate frameworks.*

– ANSI/SCTE 22-1-2002, *DOCSIS 1.0, Radio Frequency Interface*.

– ANSI/SCTE 23-3-2005, *DOCSIS 1.1 Part 3: Operations Support System Interface*.

– FIPS 140-2 (2001), *Security Requirements for Cryptographic Modules*, Department of Commerce, NIST.

– FIPS 180-1 (1995), *Secure Hash Algorithm*, Department of Commerce, NIST.

– IANAifType MIB Definitions, http://www.iana.org/assignments/ianaiftype-mib

– IEEE 802.11-1999-MIB-D6.2, *IEEE 802.11 Management Information Base.*

– IEEE 802.11A-1999, *IEEE Standard for Telecommunications and Information Exchange Between Systems – LAN/MAN Specific Requirements – Part 11: Wireless Medium Access Control (MAC) and physical layer (PHY) specifications: High Speed Physical Layer in the 5 GHz Band, Annex D.*

– IEEE 802.11B/Cor1-2001, *Information Technology – Telecommunications and Information Exchange Between Systems – Local and Metropolitan Networks – Specific Requirements – Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications, Amendment 2: Higher Speed Physical Layer (PHY) Extension in the 2.4 GHz band, Corrigendum 1, Annex D.*

– IEEE 802.11D, *IEEE Standard for IT. Telecommunications and information exchange between systems – LAN/MAN Specific Requirements – Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications, Amendment 3: Specification for operation in additional regulatory domains, Annex D.*

– IEEE 802.11G-2003, *IEEE Standard for IT. Telecommunications and information exchange between systems – LAN/MAN Specific Requirements – Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications, Amendment 4: Further Higher Data Rate Extension in the 2.4 GHz Band, Annex D.*

– ISO/IEC 8802-2 (ANSI/IEEE Std 802.2):1998, *Information technology – Telecommunications and information exchange between systems – Local and metropolitan area networks – Specific requirements – Part 2: Logical link control.*

– ISO/IEC 10038 (ANSI/IEEE Std 802.1D):1993, *Information technology – Telecommunications and information exchange between systems – Local area networks – Media access control (MAC) bridges.*

– IETF RFC 347 (1972), *Echo Process*.

– IETF RFC 768 (1980), *User Datagram Protocol (UDP)*.

– IETF RFC 791 (MIL STD 1777) (1981), *DARPA Internet Program, Protocol Specification. Internet Protocol.*

– IETF RFC 792 (1981), *DARPA Internet Program, Protocol Specification. Internet Control Message Protocol (ICMP)*.

– IETF RFC 793 (1981), *DARPA Internet Program, Protocol Specification. Transmission Control Protocol*.

– IETF RFC 868 (1983), *Time Protocol*.

– IETF RFC 919 (1984), *Broadcasting Internet Datagrams*.

– IETF RFC 922 (1984), *Broadcasting Internet datagrams in the presence of subnets*.

– IETF RFC 1034 (1987), *Domain Names – Concepts and Facilities*.

– IETF RFC 1035 (1987), *Domain Names – Implementation and Specification*.

– IETF RFC 1122 (1989), *Requirements for Internet Hosts – Communication Layers*.

– IETF RFC 1123 (1989), *Requirements for Internet Hosts – Application and Support*.

– IETF RFC 1157 (1990), *A Simple Network Management Protocol (SNMP)*.

– IETF RFC 1213 (1991), *Management Information Base for Network Management of TCP/IP-based Internets MIB-II.*

– IETF RFC 1350 (1992), *The TFTP Protocol (Revision 2)*.

– IETF RFC 1510 (1993), *The Kerberos Network Authentication Service (V5)*.

– IETF RFC 1812 (1995), *Requirements for IP Version 4 Routers*.

– IETF RFC 1889 (1996), *RTP: A Transport Protocol for Real-Time Applications*.

– IETF RFC 1901 (1996), *Introduction to Community-based SNMPv2*.

– IETF RFC 2011 (1996), *SNMPv2 Management Information Base for the Internet Protocol using SMIv2*.

– IETF RFC 2013 (1996), *SNMPv2 Management Information Base for the User Datagram Protocol using SMIv2*.

– IETF RFC 2104 (1997), *HMAC: Keyed-Hashing for Message Authentication*.

– IETF RFC 2131 (1997), *Dynamic Host Configuration Protocol*.

– IETF RFC 2132 (1997), *DHCP Options and BOOTP Vendor Extensions*.

– IETF RFC 2236 (1997), *Internet Group Management Protocol, Version 2*.

– IETF RFC 2246 (1999), *The TLS Protocol Version 1.0*.

– IETF RFC 2315 (1998), *PKCS #7*, *Cryptographic Message Syntax*, Version 1.5.

– IETF RFC 2349 (1998), *TFTP Timeout Interval and Transfer Size Options*.

– IETF RFC 2401 (1998), *Security Architecture for the Internet Protocol*.

– IETF RFC 2402 (1998), *IP Authentication Header*.

– IETF RFC 2406 (1998), *IP Encapsulating Security Payload (ESP).*

– IETF RFC 2409 (1998), *The Internet Key Exchange (IKE)*.

– IETF RFC 2474 (1998), *Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers*.

– IETF RFC 2578 (1999), *Structure of Management Information Version 2 (SMIv2).*

– IETF RFC 2579 (1999), *Textual Conventions for SMIv2*.

– IETF RFC 2580 (1999), *Conformance Statements for SMIv2*.

– IETF RFC 2616 (1999), *Hypertext Transfer Protocol – HTTP/1.1*.

– IETF RFC 2663 (1999), *IP Network Address Translator (NAT) Terminology and Considerations*.

– IETF RFC 2669 (1999), *DOCSIS Cable Device MIB – Cable Device Management Information Base for DOCSIS compliant Cable Modems and Cable Modem Termination Systems*.

– IETF RFC 2670 (1999), *Radio Frequency (RF) Interface Management Information Base for MCNS/DOCSIS compliant RF interfaces*.

– IETF RFC 2786 (2000), *Diffie-Hellman USM Key Management Information Base and Textual Convention*.

– IETF RFC 2863 (2000), *The Interfaces Group MIB.*

– IETF RFC 3022 (2001), *Traditional IP Network Address Translator (Traditional NAT).*

– IETF RFC 3046 (2001), *DHCP Relay Agent Information Option*.

– IETF RFC 3280 (2002), *Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile*.

– IETF RFC 3291 (2002), *Textual Conventions for Internet Network Addresses*.

– IETF RFC 3396 (2002), *Encoding Long Options in the Dynamic Host Configuration Protocol (DHCPv4).*

– IETF RFC 3410 (2002), *Introduction and Applicability Statements for Internet-Standard Management Framework*.

– IETF RFC 3411 (2002), *An Architecture for Describing Simple Network Management Protocol (SNMP) Management Frameworks*.

– IETF RFC 3412 (2002), *Message Processing and Dispatching for the Simple Network Management Protocol (SNMP).*

– IETF RFC 3413 (2002), *Simple Network Management Protocol (SNMP) Applications.*

– IETF RFC 3414 (2002), *User-based Security Model (USM) for version 3 of the Simple Network Management Protocol (SNMPv3).*

– IETF RFC 3415 (2002), *View-based Access Control Model (VACM) for the Simple Network Management Protocol (SNMP).*

– IETF RFC 3416 (2002), *Version 2 of the Protocol Operations for the Simple Network Management Protocol (SNMP)*.

– IETF RFC 3417 (2002), *Transport Mappings for the Simple Network Management Protocol (SNMP)*.

– IETF RFC 3418 (2002), *Management Information Base (MIB) for the Simple Network Management Protocol (SNMP)*.

– IETF RFC 3584 (2003), *Coexistence between Version 1, Version 2, and Version 3 of the Internet-standard Network Management Framework*.

– W3C Working Draft, World Wide Web Consortium (W3C), *Simple Object Access Protocol (SOAP) Version 1.2*, December 19, 2002, http://www.w3.org/2000/xp/Group/#drafts

– W3C Working Draft, World Wide Web Consortium (W3C) *XML Protocol (XMLP) Requirements*, June 26, 2002, http://www.w3.org/TR/2002/WD-xmlp-reqs-20020626

2.2 References (informative)

– IANA Port Numbers, http://www.iana.org/assignments/port-numbers

– IETF RFC 2644 (1999), *Changing the Default for Directed Broadcasts in Routers*.

– IETF RFC 3164 (2001), *The BSD Syslog Protocol*.

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

– IETF RFC 3435 (2003), *Media Gateway Control Protocol (MGCP) Version 1.0*.

– [draft-ietf-ipcdn-bpiplus-mib-05]

DOCSIS Baseline Privacy Plus MIB – Management Information Base for DOCSIS Cable Modems and Cable Modem Termination Systems for Baseline Privacy Plus, IETF Internet Draft, http://www.watersprings.org/pub/id/draft-ietf-ipcdn-bpiplus-mib-05.txt

– Federal Information Processing Standards Publications (FIPS PUB) 186 (1994), *Digital Signature Standard (DSS)*.

– Fenner W., *et al., IGMP-based Multicast Forwarding ("IGMP Proxying")*, IETF Internet Draft, http://www.ietf.org/internet-drafts/draft-ietf-magma-igmp-proxy-01.txt

– RSA Laboratories (1999), *PKCS #1, v2.0: RSA Cryptography Standard.*

– SCTE 22-3-2002, *DOCSIS 1.0 Part 3: Operations Support System Interface*.

– UDA 1.0 UPnP™ Device Architecture, Version 1.0, 08 June, 2000 http://www.upnp.org/download/UPnPDA10_20000613.htm

– UQA UPnP™ QoS Architecture 1.0, 10 March, 2005, http://www.upnp.org

– UQD UPnP™ QosDevice 1.0 Service Definition Document, 10 March, 2005.

– UQM UPnP™ QosManager 1.0 Service Definition Document, 10 March, 2005.

– UQPH UPnP™ QosPolicyHolder 1.0 Service Definition Document, 10 March, 2005.

– UIGD, InternetGatewayDevice:1, Device Template Version 1.01 for Universal Plug and Play Version 1.0, 12 November, 2001, http://www.upnp.org

– UWIC WANIPConnection:1 Service Template Version 1.01 For UPnP™ Version 1.0, 12 November, 2001, http://www.upnp.org

3 Definitions

This Recommendation defines the following terms:

3.1 IPCable2Home security portal (CSP): A functional element that provides security management and translation functions between the HFC and Home network.

3.2 embedded PS: A Portal Services element that does not use a standalone interface to connect to a CM.

3.3 home access (HA) device: A grouping of logical elements used to achieve HFC access for IPCable2Home network(s), referred to as a Residential Gateway in this Recommendation.

3.4 home client (HC) device: A group of logical elements used to provide functionality to client applications, referred to as an IPCable2Home Host in this Recommendation.

3.5 LAN IP device: 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.

3.6 portal services (PS): A functional element that provides management and translation functions between the HFC and Home network.

3.7 standalone PS: A Portal Services element that connects to the CM using only a standalone interface.

4 Abbreviations and conventions

4.1 Abbreviations

This Recommendation uses the following abbreviations:

A/V	Audio/Video
ALG	Application Layer Gateway
APP	Application
ASP	Application Specific Proxy
BP	Boundary Point
BPSC	Bulk Portal Services Configuration
CA	Certification Authority
CAP	IPCable2Home Address Portal
CAT	IPCable2Home Address Translation
CDC	IPCable2Home DHCP Client
CDP	IPCable2Home DHCP Portal
CDS	IPCable2Home DHCP Server
CH	IPCable2Home Host
CM	Cable Modem
CMP	IPCable2Home Management Portal
CMS	Call Management Server
CMTS	Cable Modem Termination System
C-NAPT	IPCable2Home Network Address and Port Translation
C-NAT	IPCable2Home Network Address Translation

CNP	IPCable2Home Name Portal
CPU	Central Processing Unit
CQoS	IPCable2Home Quality of Service
CQP	IPCable2Home QoS Portal
CRG	IPCable2Home Residential Gateway
CRL	Certificate Revocation List
CSP	IPCable2Home Security Portal
CTL	Certification Testing Laboratory
CTP	IPCable2Home Test Portal
CVC	Code Verification Certificate
CVS	Code Verification Signature
CxP	IPCable2Home Portal Services Sub-function
DER	Distinguished Encoding Rules
DHCP	Dynamic Host Configuration Protocol
DNS	Domain Name Server
DOCSIS	Data-Over-Cable Service Interface Specification
DoS	Denial of Service
DQoS	Dynamic Quality-of-Service (PacketCable)
E-MTA	Embedded Multimedia Terminal Adapter
FTP	File Transfer Protocol
FW	Firewall
GMT	Greenwich Mean Time
HA	Home Access
HE	Headend
HEX	Hexadecimal
HFC	Hybrid Fibre Coax
ICMP	Internet Control Message Protocol
IETF	Internet Engineering Task Force
IGMP	Internet Group Management Protocol
IP	Internet Protocol
IPCDN	IP over Cable Data Network – a working group of the IETF
IPF	Inbound Packet Filter
IPSec	Internet Protocol Security
KDC	Key Distribution Centre
LAN	Local Area Network
LAN-Pass	Pass-through Local Area Network address
LAN-Trans	Translated Local Area Network address

MAC	Media Access Control
MBP	Management Boundary Point
MCF	Management Client Function
MGCP	Media Gateway Control Protocol
MIB	Management Information Base
MPLS	Multiprotocol Label Switching
MSF	Management Server Function
MTA	Multimedia Terminal Adapter
NAPT	Network Address and Portal Translation
NAT	Network Address Translation
NCS	Network-based Call Signalling
NMS	Network Management System
NS	Authoritative Name Server
OID	Object Identifier
OPF	Outbound Packet Filter
OSI	Open Systems Interconnection
OSS	Operations Support System
PDU	Protocol Data Unit
PF	Packet Filter
PING	Packet Inter-Network Grouper
PKI	Public Key Infrastructure
PKINIT	Public-Key Cryptography for Initial Authentication
PS	Portal Services
PS WAN-Data	IPCable2Home Portal Services element WAN data interface
PS WAN-Man	IPCable2Home Portal Services element WAN management interface
QBP	Quality of Service Boundary Point
QCC	Quality of Service Characteristics Client
QCS	Quality of Service Characteristics Server
QFM	Quality of Service Forwarding & Media Access
QoS	Quality of Service
RAM	Random Access Memory
RDN	Relative Distinguished Name
RFC	Request for Comments
RG	Residential Gateway
ROM	Read-Only Memory
RSA	Rivest, Shamir, Adleman
RSVP	Resource ReSerVation Protocol

RTCP	Real-Time Control Protocol
RTP	Real-Time Transport Protocol
SDP	Session Description Protocol
SHA-1	Secure Hash Algorithm 1
S-MTA	Standalone Multimedia Terminal Adapter
SNMP	Simple Network Management Protocol
SoA	Start of Authority
SPF	Stateful Packet Filtering
SYSLOG	System Log
TCP	Transmission Control Protocol
TFTP	Trivial File Transfer Protocol
TLS	Transport Layer Security
TLV	Type-Length-Value
ToD	Time of Day
UDP	User Datagram Protocol
URL	Uniform Resource Locator
USFS	Upstream Selective Forwarding Switch
USM	User Security Model
UTC	Coordinated Universal Time
VACM	View-based Access Control Model
VoIP	Voice over Internet Protocol
WAN	Wide Area Network
WAN-Data	Wide Area Network Data Address Realm
WAN-Man	Wide Area Network Management Address Realm

4.2 Conventions

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

"MUST"	This word or the adjective "REQUIRED" means that the item is an absolute requirement of this Recommendation.
"MUST NOT"	This phrase means that the item is an absolute prohibition of this Recommendation.
"SHOULD"	This word or the adjective "RECOMMENDED" means that there may exist valid reasons in particular circumstances to ignore this item, but the full implications should be understood and the case carefully weighed before choosing a different course.
"SHOULD NOT"	This phrase means that there may exist valid reasons in particular circumstances when the listed behaviour is acceptable or even useful, but the full implications should be understood and the case carefully weighed before implementing any behaviour described with this label.

"MAY" This word or the adjective "OPTIONAL" means that this item is truly optional. One vendor may choose to include the item because a particular marketplace requires it or because it enhances the product, for example; another vendor may omit the same item.

5 Reference architecture

The goal of IPCable2Home is to enable the delivery of new cable-based services to devices within the home, complementing the CableModem and IPCablecom infrastructures and enabling the delivery of these services. Specifically, IPCable2Home provides an infrastructure, by specifying a home networking environment, over which IPCablecom and other related application services can be delivered, managed and supported.

This Recommendation facilitates the development of an interoperable Residential Gateway (CRG). The goal is the creation of a cable operator-configurable Residential Gateway centric environment that will interact meaningfully with IP-based home devices (LAN IP Devices). This brings cable operator driven management, provisioning, QoS, and Security to the Residential Gateway. In addition, discovery messaging, prioritized QoS, and simple remote diagnostics for home devices are specified. {informative text: Quality of Service messaging required for policy distribution to applications running on UPnP QoS compliant hosts is also specified.} A summary of the capabilities provided by this Recommendation follows:

Management, Discovery and Provisioning

- Remote management and configuration of the Residential Gateway device.
- Simple Residential Gateway diagnostics proxy for IP-based home devices.
- Hands off provisioning for Residential Gateway devices.
- Discovery of IP-based home devices and associated applications.
- Management of the Residential Gateway from the LAN.

Addressing and Packet Handling

- One-to-many addressing translation for home devices.
- One-to-one addressing translation for home devices.
- Non-translated addressing for home devices (for translated address phobic applications).
- HFC traffic protection from in-home device intra-communications.
- Home addressing support during HFC outage.
- Simple DNS server in the Residential Gateway.
- NAT support for IPSec VPN clients.
- NAT support for IP-based servers in the home using address translation.
- Client configuration of NAT.

Quality of Service (QoS)

- Residential Gateway device transparent bridging functionality for IPCablecom QoS messaging from/to IPCablecom compliant applications.
- Ability to assign traffic priorities (differentiated media access) to specific applications.
- Ability to prioritize queuing in the Residential Gateway device in conjunction with the packet handling functionality.
- {informative text: Provide UPnP QosManager and PolicyHolder Services for UPnP Hosts.}

Security

- Residential Gateway device authentication.
- Secure management messages between the cable data network and the Residential Gateway.
- Secure download of configuration and software files.
- Optional configuration file security.
- Remote Residential Gateway firewall management.
- Standardized firewall configuration and reporting.
- Simple parental control.

The remainder of this clause examines the IPCable2Home Reference Architecture from six perspectives:

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

5.1 Logical reference architecture

As shown in Figure 5-1, this clause introduces the concepts of the IPCable2Home logical elements, and the IPCable2Home devices.

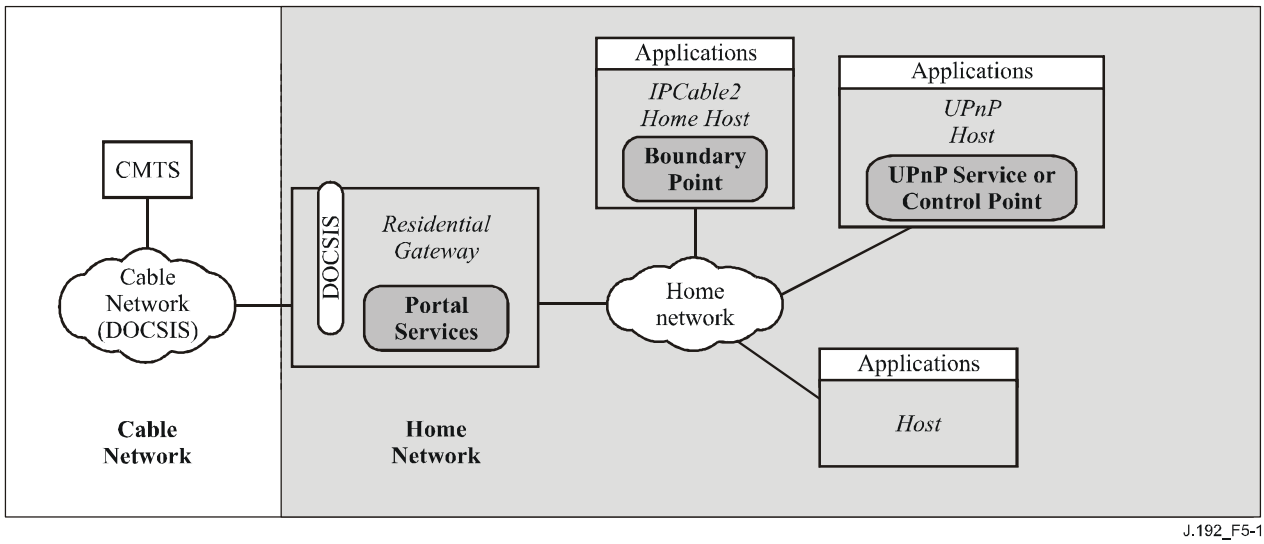


Figure 5-1/J.192 – IPCable2Home key logical concepts

5.1.1 IPCable2Home devices

The IPCable2Home architecture identifies devices in order to lend tangible context to the logical elements described in 5.1.2. Device definitions provide an informative way of depicting home network topology as well as logical elements located within the home network, but are not considered definitive or restrictive. IPCable2Home devices include the Residential Gateway and the IPCable2Home Host.

The Residential Gateway device (HA in ITU-T Rec. J.190) represents the physical location of the Portal Services (PS) logical element, which is described in 5.1.2.1. The Residential Gateway has a single WAN interface, a single PS logical element, and may have one or more LAN interfaces.

The term LAN IP Device is used to refer to any LAN Host that implements an IPv4 stack, including a DHCP client. A LAN IP Device that implements IPCable2Home functionality is referred to as a *IPCable2Home Host device* (HC in ITU-T Rec. J.190). {informative text: A LAN IP Device that implements UPnP functionality is referred to as a UPnP Host device. A LAN IP Device without IPCable2Home nor UPnP functionality is referred to as a Host.}

The IPCable2Home Host device represents the physical location of the Boundary Point (BP). The BP, defined in 5.1.2.3, enables IPCable2Home Hosts to interact with IPCable2Home Residential Gateways. The IPCable2Home Host has only one LAN interface.

{informative text: The UPnP Host device represents the physical location of UPnP Services or Control Point functionality. The UPnP Host interacts with CableHome Residential Gateway using UPnP Discovery and UPnP QoS messaging to communicate its device attributes and establish QoS on the home LAN.}

IPCable2Home assumes a home networking topology with only one DOCSIS cable modem (CM) and one IPCable2Home Residential Gateway on the home LAN. It is assumed that the DOCSIS CM is the only direct connection to the HFC. Ideally, the IPCable2Home Residential Gateway will be directly connected to the CM with no other devices attached between the CM and IPCable2Home Residential Gateway in order for the IPCable2Home Residential Gateway to provide the specified protection to the home network. All LAN Hosts are connected to the LAN behind the IPCable2Home Residential Gateway.

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 specified messages. IPCable2Home logical elements operate at the IP 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 discover, manage, and deliver services over IPCable2Home networks. IPCable2Home defines a logical entity specific to each IPCable2Home Device: The PS logical entity encapsulates IPCable2Home functionality defined for Residential Gateways and the BP logical entity encapsulates functionality defined for IPCable2Home Hosts (see 5.1.1 for a description of the IPCable2Home devices).

5.1.2.1 Portal Services (PS)

The Portal Services is a logical element of a Residential Gateway device that provides in-premise and aggregated security, management, provisioning, addressing and QoS services. The term "portal" is used to indicate services that interface the WAN to the LAN. This clause describes features of the Portal Services logical element.

5.1.2.1.1 Standalone PS and PS with embedded Cable Modem

The two primary components possible within a Residential Gateway, the Cable Modem (CM) and the Portal Services (PS) element, may use shared or independent hardware and software resources. It is this resource sharing between the CM and PS that distinguishes the Standalone PS from an Embedded PS.

A Standalone PS MUST NOT share hardware or software components with a CM. The separation of the CM from the standalone PS MUST appear to the PS as a simple disconnection of its WAN – i.e., the PS will continue fully functional as if it had the WAN disconnected. Otherwise, the PS will be considered Embedded. Given these definitions, 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 the Standalone and Embedded cases, and respond to unique management addresses. In the Embedded case, the CM and PS share hardware or software components, but from the management perspective, they are separate entities.

Figure 5-2 illustrates 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.

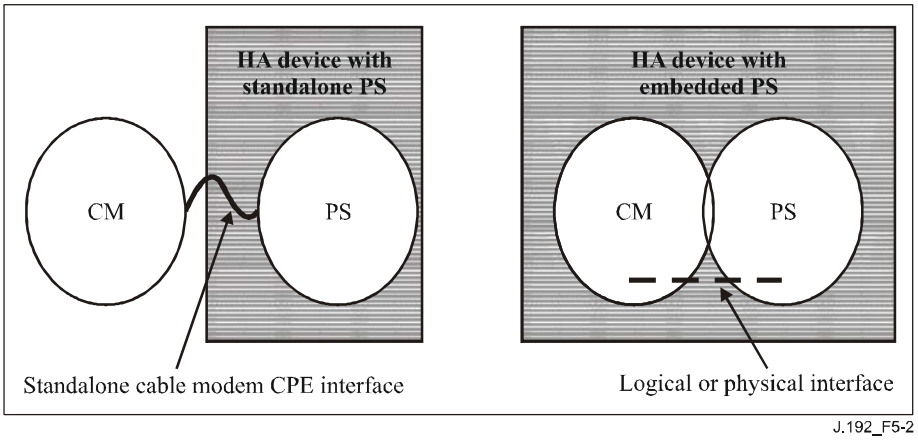


Figure 5-2/J.192 – Standalone PS and PS with embedded CM

5.1.2.2 {informative text: UPnP services and control points

A UPnP Host encapsulates logical UPnP functionality such as UPnP Services or Control Points. While the PS is a logical entity specified by CableHome, the PS interacts with logical elements on the LAN that are not specified by CableHome. The PS provides limited services to all LAN IP Devices, and provides additional services to interact with UPnP Services and UPnP Control Points. The CableHome Architecture uses UPnP compliant messaging between the PS and UPnP Services & Control Points on the LAN. In the UPnP architecture [UDA 1.0], control messages are initiated from UPnP Control Point and responded to by UPnP Services.}

5.1.2.3 Boundary Point (BP)

A Boundary Point (BP) is a logical element that encapsulates all of the IPCable2Home functionality defined for an IPCable2Home Host on the home LAN.

5.1.3 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 5-3.)

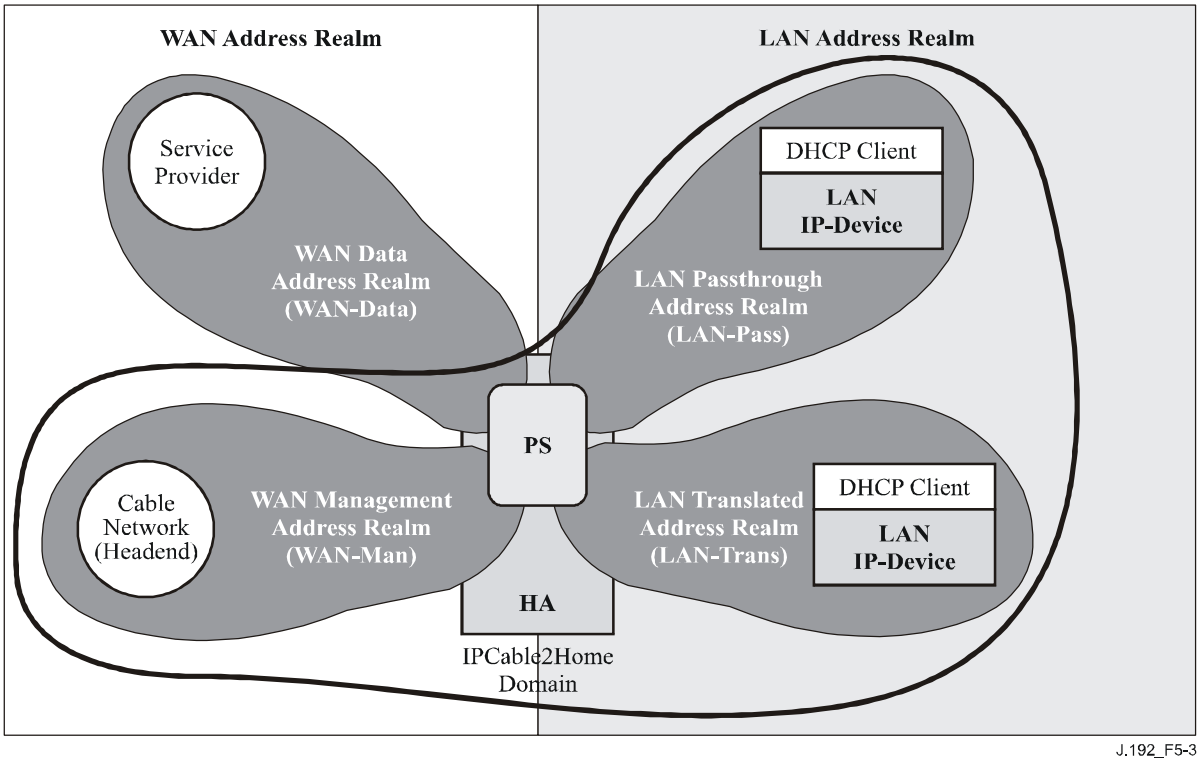


Figure 5-3/J.192 – IPCable2Home Address Realms

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 network, 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 be assigned translated addresses from the LAN Translated Address Realm (LAN-Trans).

Physical LAN interfaces in the PS are assigned an index in accordance with the Interfaces Group MIB [RFC 2863] as described in 6.3.3.1.4.8, Interfaces Group MIB. A virtual LAN interface aggregating all physical LAN interfaces is also defined for the PS in 6.3.3.1.4.8. Similarly, a virtual LAN interface aggregating only the physical wireless LAN interfaces is defined for the PS in 6.3.3.1.4.8. The LAN-side IP address defined for the PS is "bound" to the "all" physical LAN interfaces virtual interface. PS DHCP and domain name server functions, and the PS router function, are applications implemented in the PS addressed using the LAN-side IP address bound to the "all" physical LAN interfaces virtual LAN interface.

5.2 IPCable2Home functional reference model

IPCable2Home Functions are IP-based services defined in this Recommendation, to be implemented by the PS or the cable operator's data network, and support the delivery of cable-based services. IPCable2Home functions are defined for each of the major specification areas: Provisioning, Management, Security and Quality of Service.

Sub-elements represent groupings of related functionality within the PS. The PS logical element can contain any number of sub-elements, and sub-elements may themselves contain sub-groupings of functions (i.e., sub-elements within sub-elements).

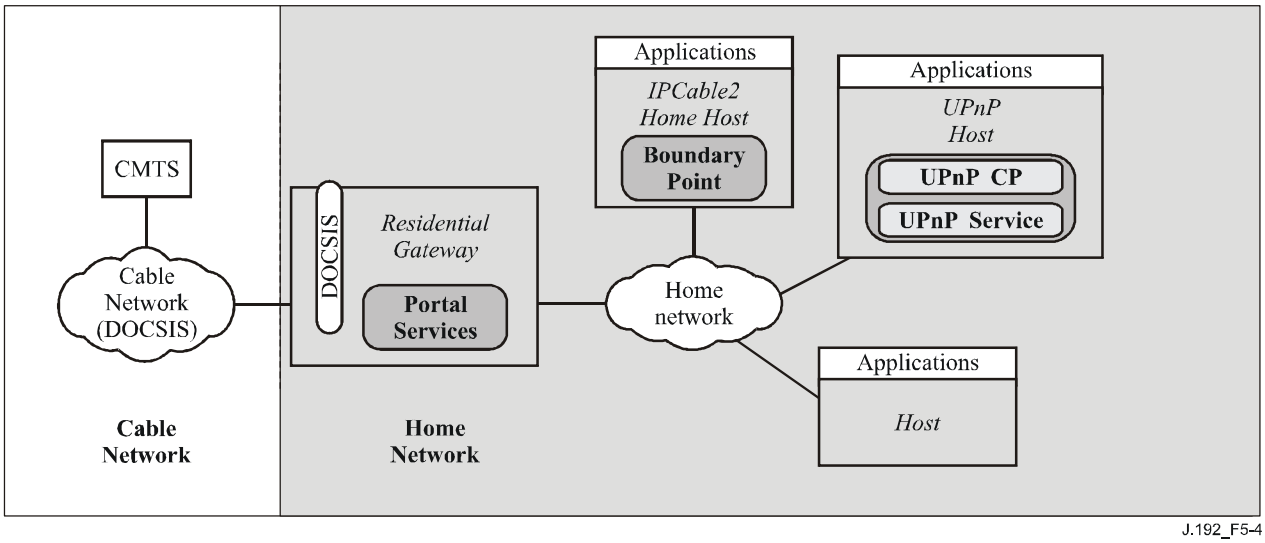


Figure 5-4/J.192 – IPCable2Home sub-elements

The PS contains a number of sub-elements, which are introduced below.

5.2.1 {informative text: Relationship between CableHome and UPnP

The IPCable2Home Architecture uses UPnP compliant messaging to manage and interact with UPnP Host Devices. Thus, functionality of some PS sub-elements is described in terms of UPnP Control Point and UPnP Services. For example, the PS includes functionality such as the UPnP IGD WANIpConnection Service [UWIC] to allow configuration of the IPCable2Home Address Translation (CAT) functionality from UPnP Hosts. Similarly, the PS uses UPnP Control Point functionality for discovery of UPnP Devices & Services on the home network. (See Figure 5-5.)

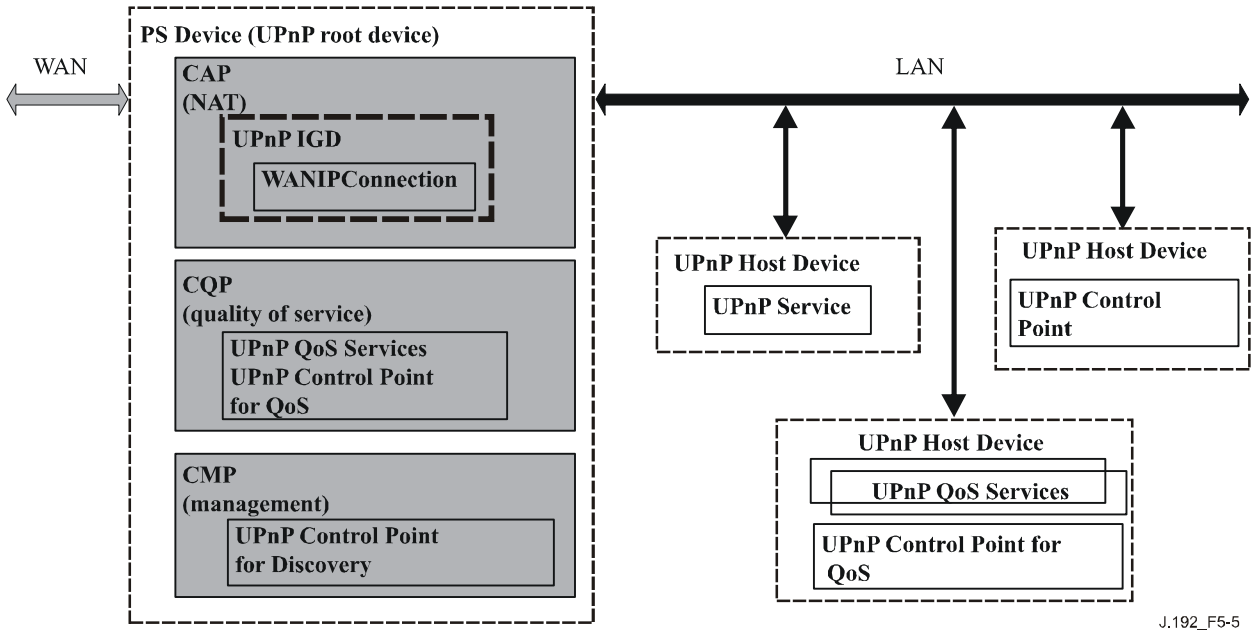


Figure 5-5/J.192 – Hierarchy of UPnP devices and services in the IPCable2Home PS}

5.2.2 IPCable2Home management and provisioning functions

To support the requirements during the provisioning and management of Hosts within the home, IPCable2Home uses management and provisioning functions that reside in the cable data network, and defines functions for the PS. Cable network-based management and provisioning functions include a number of services used by IPCable2Home-defined management and provisioning processes. Portal Services management and provisioning functions are located within the Residential Gateway and include server-like, client-like, and other types of functionality. Examples of CableNetwork and PS functions are introduced in Tables 5-1 and 5-2 and are illustrated in Figure 5-6.

Table 5-1/J.192 – Cable network management functions

Functions	Description
Cable Network DHCP Server	The DHCP server is a cable network component that provides address information for the WAN-Man and WAN-Data address realms to the PS
Cable Network Management Servers	The IPCable2Home management messaging, download, event notification servers including protocols such as SNMP, SYSLOG, and TFTP [RFC 2349]
Cable Network Time of Day Server	The time of day (ToD) server provides clients with the current time of day.

Table 5-2/J.192 – PS management and provisioning functions

Management portal functions	Description
IPCable2Home Address Portal (CAP)	Within the PS, the CAP interconnects the WAN and LAN address realms for data traffic. (See CAT/Passthrough) {informative text: It also provides UPnP IGD WANIpConnection Service interface for UPnP Control Points to configure the IPCable2Home Address Translation (CAT) table.}
IPCable2Home Address Translation (CAT)	A sub-function of the CAP, a CAT translates public IP network addresses on the WAN-Data side of the CAP to private IP network 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.
IPCable2Home Management Portal (CMP)	The function that provides an interface between the Operator and the PS-database. {informative text: It also discovers various UPnP Hosts and UPnP services offered by them using UPnP discovery process.}
IPCable2Home 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.
IPCable2Home Naming Portal (CNP)	The CNP provides a simple DNS service for LAN IP Devices requiring naming services.
IPCable2Home Test Portal (CTP)	The CTP provides a remote means to initiate pings and loopbacks within the LAN.
HTTP Server	HTTP is the transport protocol used to convey SOAP messaging on the LAN. The PS contains an HTTP server which serves data upon BP requests.
XML and SOAP Parsers	SOAP and XML are used for messaging on the LAN. The PS contains parsers for both.

To communicate with the PS Management functions listed above, the following functions (see Table 5-3) are expected in the LAN IP Devices but are not required by this Recommendation.

Table 5-3/J.192 – LAN IP device management and provisioning expected functions

Management client functions	Description
LAN IP Device DHCP Client	The IPCable2Home DHCP client function is a in-home component used during the LAN IP Device provisioning process to dynamically request IP addresses.
{informative text: UPnP Control Point or Services}	{informative text: UPnP is the protocol used to convey Management and Discovery messaging on the LAN.}

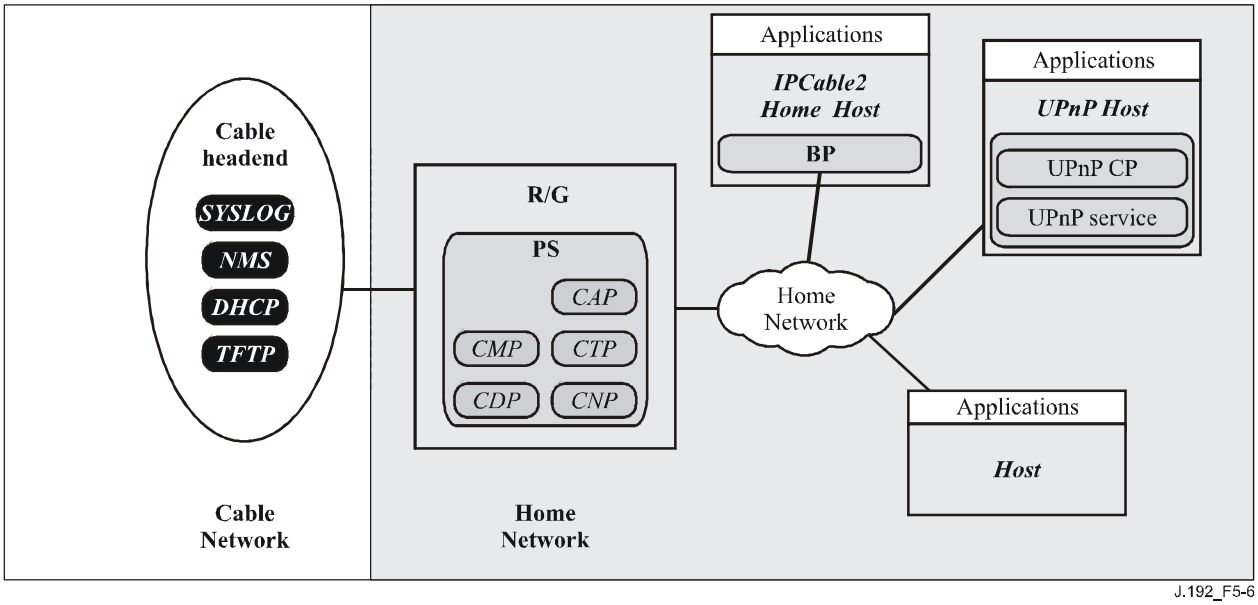


Figure 5-6/J.192 – IPCable2Home management elements

5.2.3 IPCable2Home security functions

To support the IPCable2Home security requirements (see 11.2.1), IPCable2Home uses security functions that reside in the cable data network and defines functions for the PS. Cable network-based security functions include servers used for key distribution, encryption and authentication. Portal Services security functions are located within the Residential Gateway which includes client functions and other types of functions. Examples of cable network-based and PS security functions are introduced in Tables 5-4 and 5-5 and are illustrated in Figure 5-7.

Table 5-4/J.192 – Portal Services security functions

Functions	Description
IPCable2Home Security Portal (CSP)	The CSP communicates with Headend security servers, and includes functions that provide client side participation in the authentication, key exchange and certificate management processes. Other security functions include management message security, participation in secure download processes, and remote firewall management.
Firewall (FW)	The Firewall provides functionality that protects the home network from malicious attack.

Table 5-5/J.192 – Cable network security function

Functions	Description
Key Distribution Centre (KDC) Servers	The key distribution centre (KDC) servers provide security services to the CSP and include functions that participate in the authentication and key exchange processes.

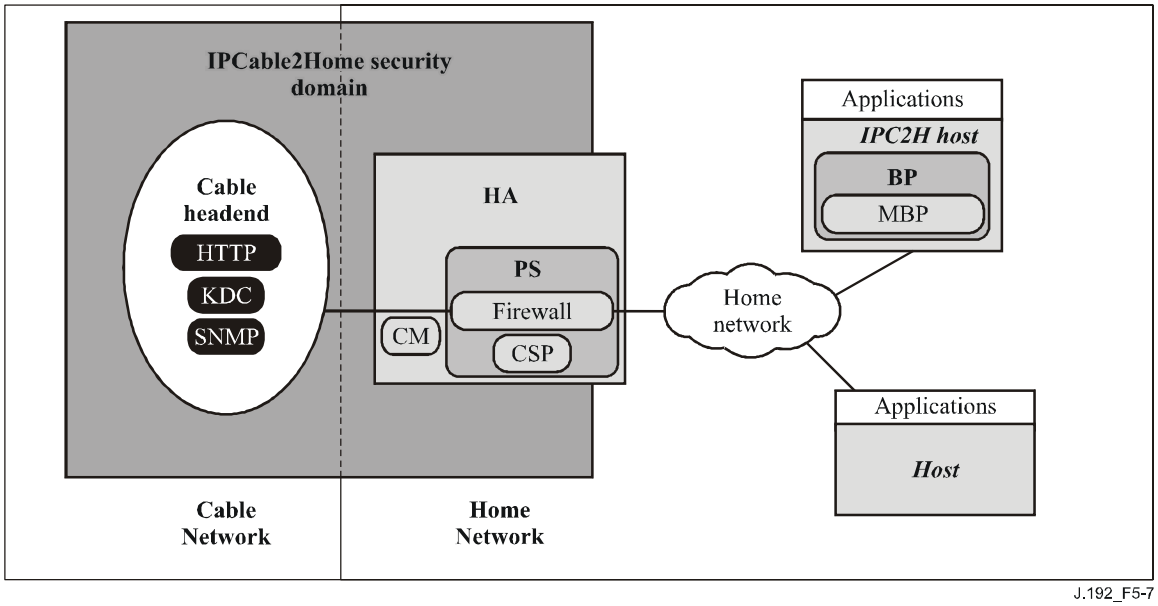


Figure 5-7/J.192 – IPCable2Home Security Elements

5.2.4 IPCable2Home QoS functions

To support the Quality of Service requirements (see 10.2.1), IPCable2Home defines functions for the PS. Portal Services QoS functions are located within the Residential Gateway and include a server function and other types of functions. Examples of PS QoS functions are introduced in Tables 5-6 and 5-7 and are illustrated in Figure 5-8.

Table 5-6/J.192 – Portal Services QoS functions

Functions	Description
QoS Policy Server (QPS)	This functionality maintains a repository of QoS Policy for various devices and applications within the home network and communicates QoS Policy {informative text: When requested by a UPnP QosManager Entity.}
QoS Forwarding and Media access (QFM)	QFM is responsible for prioritized queuing and packet forwarding as well as for prioritized shared media access in the PS.
{informative text: UPnP QoS Device Service Interface (QD)}	1) This service interface triggers the CableHome PS to negotiate access network QoS using the CH-PCMM Interface for the traffic streams that span access network and home network. 2) The interface receives traffic classifier requests {informative text: from UPnP QosManager Entities and places them in the PS database.}These classifiers are used by the QFM functionality for packet classification.
QoS Manager Service (QM)	{informative text: A UPnP QoS Manager Service in the PS. Ensures that there is at least one UPnP QosManager Service for UPnP Control Points to request QoS on the home LAN.}
QoS functionality of PS Control Point	This entity acts as a Control Point for different{informative text: UPnP} QoS Services on the home LAN. The QoS Discovery logic of this control point is responsible for collecting QoS related information from various {informative text: UPnP} QoS Services on the home LAN and storing it in the PS database. The entity also captures {informative text: UPnP} QoS announcements, events and issuing actions as needed for various {informative text: UPnP} QoS Services.

To communicate with the PS Management functions listed above, the following functions (see Table 5-7) are expected in the LAN IP Devices but are not required by this Recommendation.

Table 5-7/J.192 – BP QoS function

Functions	Description
{informative text: UPnP QoS Control Point or Services.}	{informative text: UPnP QoS is the protocol used to convey QoS messaging on the LAN.}

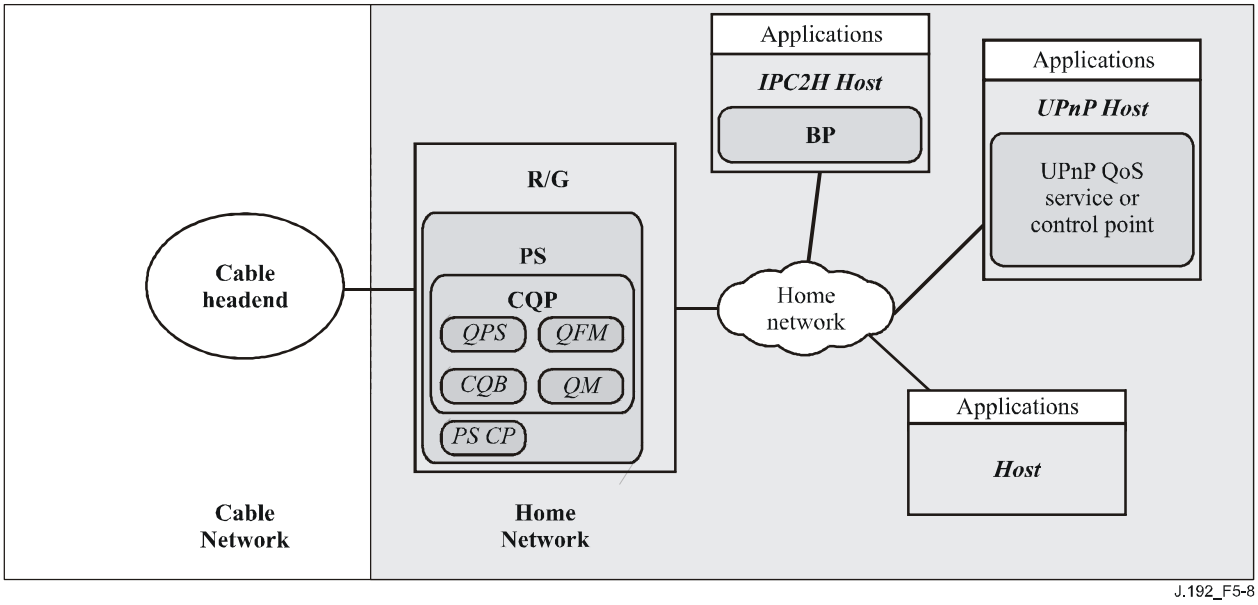


Figure 5-8/J.192 – IPCable2Home QoS elements

5.3 IPCable2Home messaging interface model

Communication between the functions in the cable data network, Residential Gateway, and LAN IP Devices occur on messaging interfaces identified and labelled in Figure 5-9. The types of messaging interfaces are differentiated by the elements that are involved in the communication.

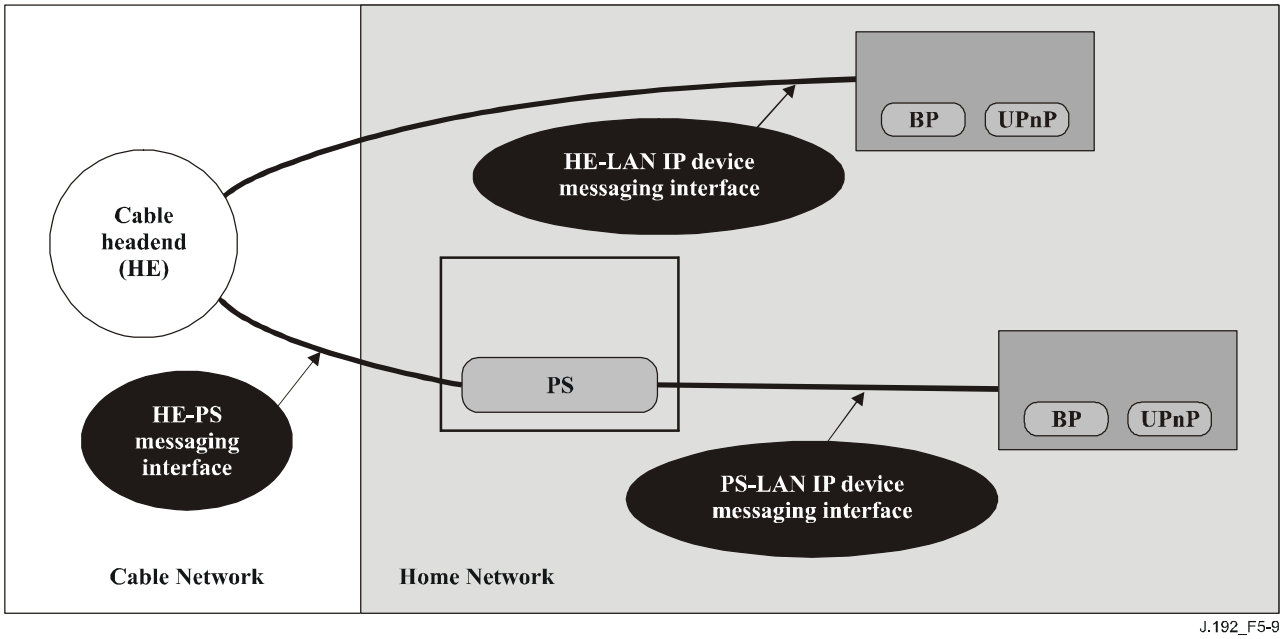


Figure 5-9/J.192 – IPCable2Home reference interfaces

Table 5-8 identifies interfaces for which IPCable2Home specifies messaging.

Table 5-8/J.192 – Valid interface paths for each functionality

Functionality	Protocol	Interface		
		HE-PS	HE-LAN IP device	PS-LAN IP device
Name service	DNS	Unspecified	Unspecified	This Recommendation
Software Download	TFTP	This Recommendation	Unspecified	Unspecified
Address Acquisition	DHCP	This Recommendation	Unspecified	This Recommendation
Management (single) (bulk)	 SNMP TFTP or HTTP	 This Recommendation This Recommendation	 Unspecified Unspecified	 Unspecified Unspecified
Event Notification	SNMP SYSLOG	This Recommendation This Recommendation	Unspecified	Unspecified
QoS	IPCablecom QoS Protocols SNMP {informative text: UPnP QoS}	Unspecified This Recommendation Unspecified	IPCablecom Unspecified Unspecified	Unspecified Unspecified This Recommendation
Security (key distribution)	Kerberos	This Recommendation	Unspecified	Unspecified
Security (authentication)	Kerberos or TLS	This Recommendation	Unspecified	Unspecified
Ping	ICMP	This Recommendation	Unspecified	This Recommendation
Loopback/Echo	UDP/TCP	Unspecified	Unspecified	This Recommendation

Table 5-8/J.192 – Valid interface paths for each functionality

Functionality	Protocol	Interface		
		HE-PS	HE-LAN IP device	PS-LAN IP device
Application Discovery	SNMP SOAP/XML	This Recommendation	Unspecified	 This Recommendation
Device Discovery	SNMP {informative text: UPnP Discovery/ SSDP}	This Recommendation Unspecified	Unspecified Unspecified	Unspecified This Recommendation

5.4 IPCable2Home information reference model

The operation of the management model is based upon a store of information maintained in the PS by the various sub-elements of the PS (CAP, CDP, CMP, etc.). These sub-elements need a means of interacting via information exchange, and the PS Database is a conceptual entity that represents a store for this information. The 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 IPCable2Home elements.

Figure 5-10 shows the relationship between the database and the PS functions. Table 5-9 describes the typical information associated with each of these functions. Figure 5-11 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.

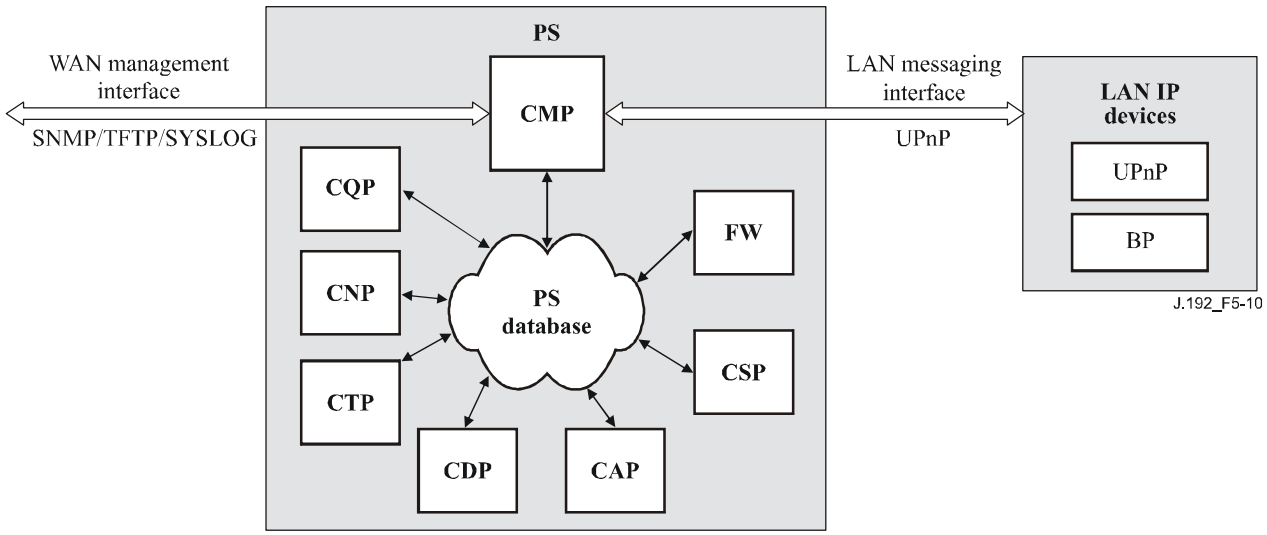


Figure 5-10/J.192 – PS function and database relationship

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

Table 5-9/J.192 – Typical 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 PS functions. Information about {informative text: UPnP Host devices and services collected through UPnP Discovery Messaging.}
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 behaviour of the Firewall (ruleset), firewall events and logging.
Event information	Information associated with the local log for all general events, traps, etc.
CQP Information	QoS Policy received from cable operator and QoS information received from the {informative text: UPnP} QoS host devices and Control Points {informative text: via UPnP QoS Messaging.}

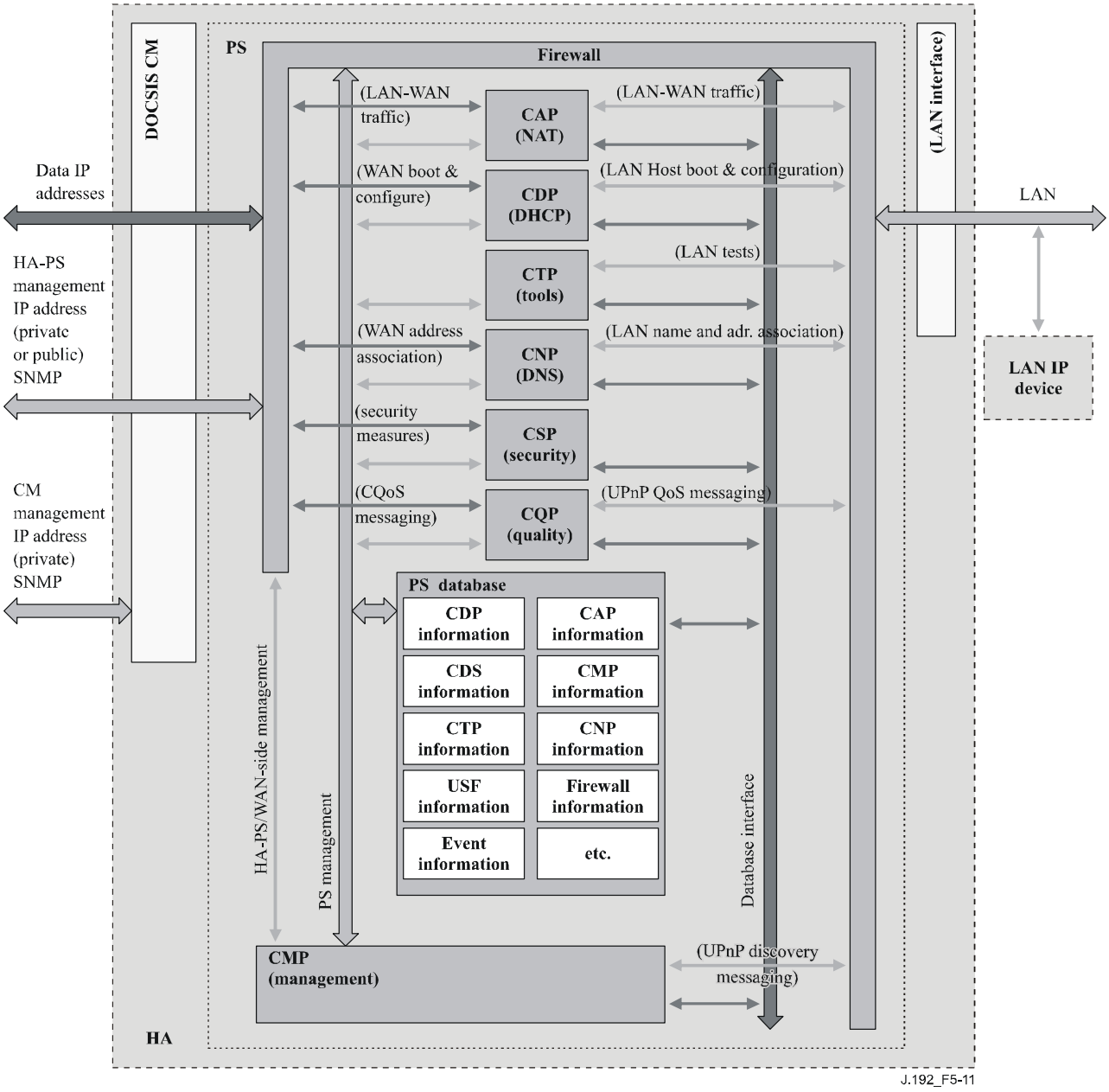


Figure 5-11/J.192 – PS database detailed example implementation

The PS is primarily managed from the WAN via the CMP, and to a large degree this involves access to the information in the PS Database. Management is used for initialization and provisioning of the PS functions, and remote diagnostics or status of the LAN. 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 Server. 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 CDP contains the address functions to act as the DHCP server in the LAN-Trans realm and implements 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 provides PS authentication capabilities as well as key exchange activities.

The CQP is part of a system that enables IPCable2Home QoS. The CQP provides IPCable2Home traffic priorities as well as differentiated media access functions.

5.5 IPCable2Home operational models

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 CableModem (ITU-T Rec. J.112 or J.122) only provisioning infrastructure, as well as within a CableModem plus IPCablecom provisioning infrastructure. The CableModem plus IPCablecom provisioning IPCable2Home infrastructure builds upon the CableModem infrastructures to enable additional services, and incorporates a number of capabilities that are similar to those within an IPCablecom provisioning system.

The PS can be configured to be fully provisioned and managed by the cable operator, or it can be configured to operate as an unmanaged and unprovisioned (except for an IP address lease and configuration provided via DHCP) residential gateway. Additionally, an embedded PS can be disabled to allow the device in which it is implemented to be deployed as a cable modem only.

In its fully provisioned and managed mode, the PS receives its operational mode information in the DHCP messages [RFC 2131], [RFC 2132] when it boots up and requests a network address using DHCP. Depending upon the information passed in the DHCP messages, the PS can be configured to operate in one of two provisioning modes:

- The DHCP Provisioning Mode.
- The SNMP Provisioning Mode.

If the PS is not configured to operate in either DHCP Provisioning Mode or SNMP Provisioning Mode, it assumes that the back office support is not currently available, and will default to operate in Dormant CableHome Mode. In Dormant CableHome Mode, the Residential Gateway will be fully operational from the user perspective, but it will not be operator configured or managed. If a PS is embedded in a device with an [eDOCSIS]-compliant cable modem, the embedded PS can be directly configured through the cable modem to operate in Dormant CableHome Mode. The cable operator can also "turn off", or disable, the PS functionality in a device with an embedded cable modem and embedded PS, directly through the cable modem.

When the PS is configured to operate in DHCP Provisioning Mode, it can be configured to begin a Transport Layer Security (TLS) session over HTTP in order to provide secure download of PS and Firewall configuration files.

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

- NmAccess Mode.
- SNMPv3 Coexistence Mode.

When the PS is configured to operate in SNMP Provisioning Mode, it operates in SNMPv3 Coexistence Network Management Mode only.

Figure 5-12 illustrates the various PS operational modes along with the associated triggers for each. See 7.3.3.2.4, CDC Requirements, for a full description of provision mode determination.

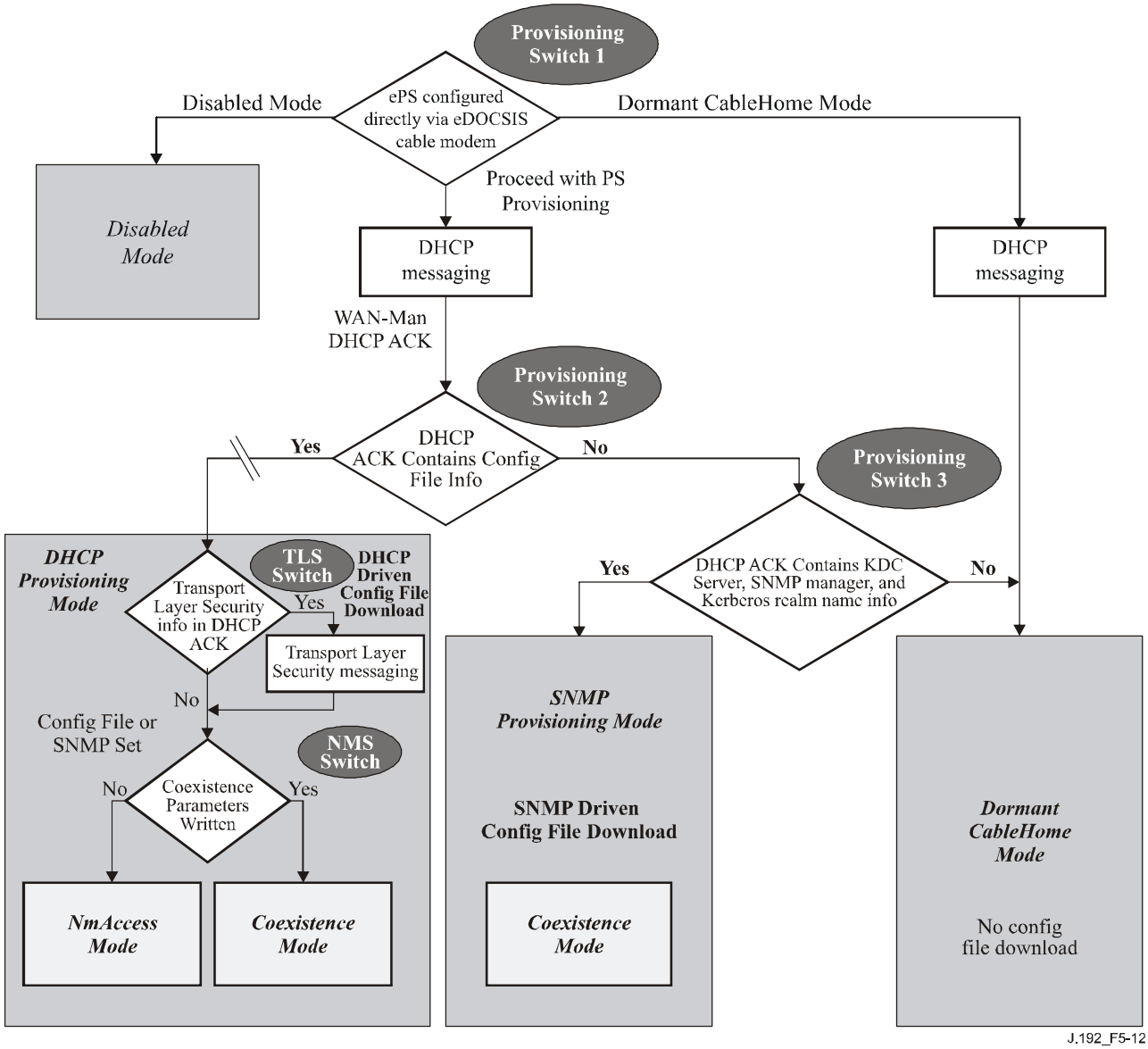


Figure 5-12/J.192 – PS operational modes

Table 5-10 describes the infrastructures within which each PS mode is intended to operate.

Table 5-10/J.192 – PS infrastructures

Mode	Capability directly effected	Intended infrastructure
SNMP Provisioning Mode	Configuration file download.	The CableModem plus IPCablecom provisioning Infrastructure
DHCP Provisioning Mode	Configuration file download.	CableModem infrastructures with IPCable2Home support
DHCP Provisioning Mode: with TLS/HTTP	Secure configuration file download	CableModem infrastructures with IPCable2Home and TLS support
DHCP Provisioning Mode: NmAccess Network Management Mode	SNMP version used between NMS and PS	J.112 (1998) Infrastructure (SNMPv1/v2) with IPCable2Home support
DHCP Provisioning Mode: SNMP Coexistence Network Management Mode	SNMP version used between NMS and PS	J.112 & J.122, and the CableModem plus IPCablecom provisioning Infrastructures (SNMPv3) with IPCable2Home support
Dormant IPCable2Home Mode	Configuration and Management	No IPCable2Home support
Disabled Mode	Configuration, management, address translation, traffic forwarding, and firewall	DOCSIS 1.0, 1.1, and 2.0 infrastructures with IPCable2Home support. Supports deployment of a device with embedded cable modem and embedded PS as a cable modem only.

5.6 Physical interfaces on the Residential Gateway

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, to the cable network via the cable modem acting as a transparent bridge for a PS with an embedded cable modem, and other WAN Networking Interfaces, intended for WAN connection, in the Standalone PS case.
- LAN Networking Interfaces for connection to LAN IP Devices and IPCable2Home hosts.
- Hardware Test Interfaces, such as JTAG and other proprietary approaches, which are part of the silicon and do not 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 labelled 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. (PS functions are defined by this Recommendation.) 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 premises data network functions) that could require configuration by a user. A PS MAY provide the ability to configure non-IPCable2Home functions. The PS SHOULD implement a User Interface to enable user configuration of non-IPCable2Home functions and CableHome functions. The User Interface is permitted to provide user access to CableHome-defined management functions (i.e., to IPCable2Home-defined MIB objects), but is required to adhere to cable operator-configured access rules. Refer to 6.3.3.1.4.2.2.

6 Management tools

6.1 Introduction/Overview

The IPCable2Home Management Tools provide the cable operator with functionality to monitor and configure the Portal Services (PS) element, {informative text: to discover UPnP Host Devices and the UPnP Services they offer,} to remotely check connectivity between the PS and LAN IP Devices, and to report on status and exception events in the PS. This clause describes and specifies requirements for these capabilities.

Differences between Management Tools defined in ITU-T Rec. J.191 and those defined in this Recommendation are listed below:

- This Recommendation adds the requirement for the PS to support SNMP management from any LAN Interface.
- {informative text: This Recommendation adds the requirement for the PS to support UPnP Discovery messaging to enable cable operators to discover various UPnP devices and their capabilities on the home LAN.}
- This Recommendation adds the following MIB objects to the PS:
  - objects needed to support prioritized Quality of Service on the LAN;
  - objects supporting enhanced firewall functionality;
  - {informative text: objects enabling the cable operator to view attributes and capabilities of UPnP Host devices on the LAN.}

6.1.1 Goals

The goals for the IPCable2Home Management Tools include:

- {informative text: Provide a means for the cable operator to discover UPnP Host Devices.}
- Provide cable operators with visibility to LAN IP Devices.
- {informative text: Provide cable operators with visibility to applications and services offered by UPnP Host devices.}
- Define a minimum set of remote diagnostic tools that will allow the cable operator to verify connectivity between the Portal Services element and any LAN IP Device.
- Provide cable operators with access, via the MIBs, to internal data in the PS element and enable the cable operator to monitor IPCable2Home-specified parameters and to configure or reconfigure IPCable2Home-specified capabilities as necessary.
- Provide a means for reporting exceptions and other events in the form of SNMP traps, messages to a local log, or messages to a system log (SYSLOG) in the cable network.

6.1.2 Assumptions

The assumptions for the IPCable2Home network management environment include the following:

- IPCable2Home-compliant devices implement the Internet Protocol (IPv4) suite of protocols.
- {informative text: UPnP Host Devices implement UPnP Device and Service discovery protocols as specified by UPnP Device Architecture 1.0 [UDA 1.0].}
- SNMP is used for the exchange of management messages between the cable network NMS and the PS in the IPCable2Home Residential Gateway device. SNMP provides visibility for the NMS to interfaces on the PS, via access to internal PS data, through required MIBs.
- Any of SNMPv1/v2c/v3 can be used as a management protocol between the NMS and the IPCable2Home Portal Services element.
- LAN IP Devices implement a DHCP client.
- The IPCable2Home Residential Gateway and LAN IP Devices support ICMP.
- The PING utility supplies functionality sufficient to provide the cable operator with the desired information about connectivity between the PS element and LAN IP Devices.

6.2 Management architecture

6.2.1 System design guidelines

The Management Tools system design guidelines are listed in Table 6-1. This list provides guidance for the development of the IPCable2Home management tools specifications.

Table 6-1/J.192 – Management tools system design guidelines

Reference	Guidelines
Mgmt 1	The PS will implement SNMPv1/v2c/v3 protocols to provide access to internal Portal Services data.
Mgmt 2	The PS will be capable of issuing an ICMP Request (Ping) command to any LAN IP Device specified by the cable operator and store results in the PS Database. Remote Ping test results will be accessible through CTP MIB objects.
Mgmt 3	The PS will be capable of executing a Connection Speed Test with a specified LAN IP Device specified by the cable operator and store results in the PS Database. Remote Connection Speed test results will be accessible through CTP MIB objects.
Mgmt 4	The PS element will be capable of reporting events.
Mgmt 5	{informative text: The PS element will be capable of communicating with UPnP Host devices in the LAN-Pass and LAN-Trans realms for the exchange of device attributes, QoS priorities, and UPnP Host services and application information.}
Mgmt 6	In the event that the PS loses connectivity with the cable data network and its applications, the Discovery function and LAN Messaging function will continue to operate.

6.2.2 Management tools system description

As shown in Figure 6-1, IPCable2Home Management Tools architecture consists of the following components:

1) the IPCable2Home Management Portal (CMP);
2) the IPCable2Home Test Portal (CTP);
3) a Management Information Base (MIB);
4) an SNMP Network Management System (NMS) that is part of the cable network.

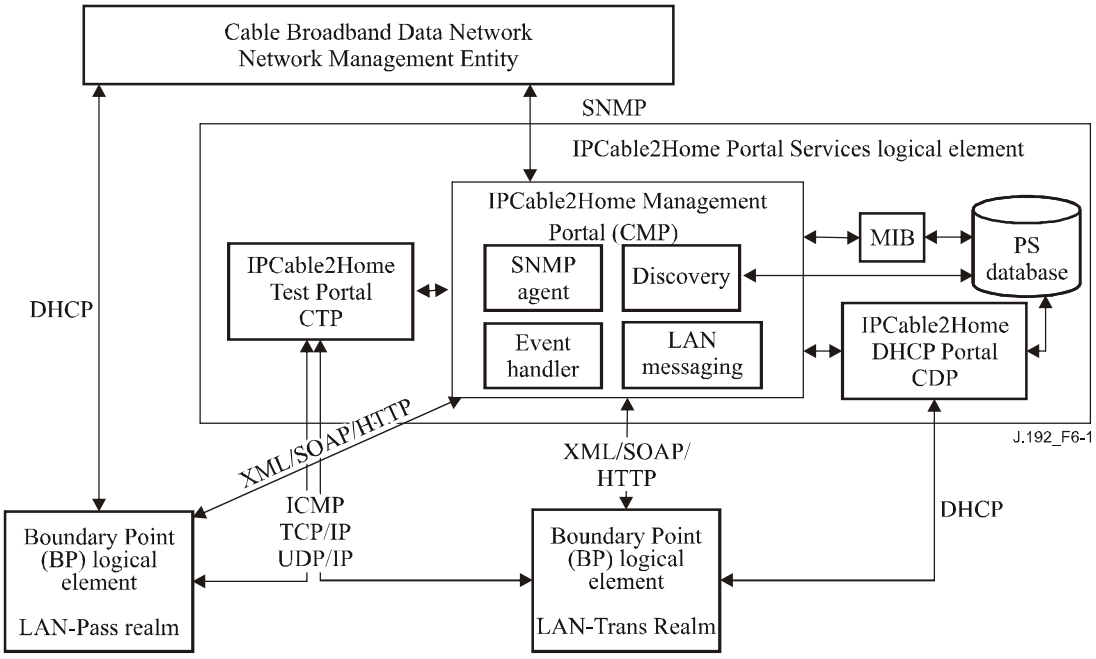


Figure 6-1/J.192 – IPCable2Home management architecture

The cable data network NMS monitors and configures the PS by accessing the PS Database through MIBs specified in 6.3.3.1.4.7. {informative text: The cable operator accesses UPnP Host Device and IPCable2Home Residential Gateway attributes through the PSDev MIB [see E.4] and through the QoS MIB [see E.7], and configures IPCable2Home Host devices with QoS policy (in the form of QoS priorities) using the PS as a proxy. The IPCable2Home PS implements a UPnP Control Point (PS CP) to obtain discovery information from UPnP Host Devices.}

The NMS can also directly communicate with LAN IP Devices in the IPCable2Home LAN-Pass realm.

The IPCable2Home DHCP Portal, described in the Provisioning Tools clause (clause 7), plays a role in basic LAN IP Device discovery. Through DHCP communication between LAN IP Devices and the CDP, the LAN IP Device provides its hardware address and may provide configuration information to the CMP through DHCP Option codes. The CMP will use the information to populate CDP MIB LAN Address Table (cabhCdpLanAddrTable) objects.

The CMP and CTP functional elements reside within the PS. The PS logical element may be co-resident with an embedded cable modem or standalone, without embedded cable modem functionality, as described in 5.1.2.1.1.

The CM and PS are separate and independent management entities. In the case of a PS with an embedded cable modem, no data sharing between CM and PS is implied, with the following exceptions:

1) the software image download is controlled via the cable modem's MIB;

2) the MIB for SNMP [RFC 3418], the SNMP Group of MIB-2 (mib-2 11) [RFC 1213], the IP Group and the ICMP Group of the SNMPv2 MIB for IP [RFC 2011], and the SNMPv2 MIB for UDP [RFC 2013] are allowed to be shared between the PS and CM.

In a PS with an embedded cable modem, the cable modem's docsDevSoftware objects are accessed to set up, initiate and monitor the download of a single combined software image. This process is described in 11.8, Secure Software Download for the PS.

Because of this management independence, the CM and PS 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. IPCable2Home does not preclude the use of a single SNMP agent for a PS with an embedded CM.

The Portal Services element supports SNMPv1, SNMPv2c and SNMPv3 protocols. Clause 5.5 introduces the provisioning modes supported by an IPCable2Home Portal Services element, and clause 7 provides additional detail about these modes. The provisioning mode in which the PS is operating partially determines which version of SNMP the PS uses. Additional detail is provided in 6.3.3.

6.3 PS logical element – IPCable2Home Management Portal (CMP)

The IPCable2Home Management Portal (CMP) is a sub-element of the PS logical element. It serves as the hub of management control of the PS and for discovery of devices present on the LAN.

The CMP aggregates and interconnects management information in the WAN-Man and LAN-Trans realms, since they are not directly accessible to each other.

6.3.1 CMP goals

The goals for the IPCable2Home Management Portal include:

- Enable the NMS to remotely view and update IPCable2Home Address Portal (CAP) configuration information.
- Enable the NMS to remotely view and update Firewall configuration information.
- Enable remote testing of connectivity between the IPCable2Home Residential Gateway and LAN IP Devices in the LAN-Trans realm, via the IPCable2Home Test Portal (CTP).
- Enable remote configuration of LAN IP Device Addressing parameters.
- Enable viewing of LAN IP Device information obtained via the IPCable2Home DHCP Portal (CDP).
- {informative text: Provide cable operator access to the attributes of UPnP Host devices and UPnP Services implemented by them, acquired through the UPnP discovery process.}
- Enable viewing of the results of LAN IP Device performance monitoring done by the IPCable2Home Test Portal (CTP).
- Enable the NMS to access other PS configuration parameters.
- Facilitate security by providing access to security parameters, and the use of SNMPv1/v2c/v3 in the appropriate network management mode.
- Provide the capability to disable LAN segments.

6.3.2 CMP design guidelines

The CMP design guidelines are listed in Table 6-2. This list provides guidance for the specification of CMP functionality.

Table 6-2/J.192 – CMP system design guidelines

Reference	Guidelines
CMP 1	Interfaces will support the management and diagnosis features and functions required to support cable-based services provisioned across the home network.
CMP 2	Loss of connection between broadband service provider(s) and the home network will not disable or degrade the operation of internal home networking functions.
CMP 3	The home network will recover from a power outage, and devices connected to the home network must return to the operational state they were in prior to the outage.
CMP 4	Home network devices will be easy to install and configure for operation, much like a home appliance.
CMP 5	{informative text:The PS and UPnP Host will support UPnP Discovery protocol for acquiring UPnP Host Device attributes and UPnP Services implemented by them.}
CMP 6	{informative text:The PS will provide to the cable operator, upon request, information about IPCable2Home Host Device attributes and UPnP Services implemented by them.}
CMP 7	{informative text:UPnP Discovery protocol message exchange within the home LAN will not noticeably degrade performance of the home LAN.}
CMP 8	{informative text:UPnP Discovery messaging will not propagate onto the WAN.}

6.3.3 CMP system description

The CMP is responsible for the following important IPCable2Home capabilities:

- Enable management of the Portal Services functions from the cable operator's data network Network Management System (NMS) by providing access to the PS Database and its state variables through IPCable2Home-specified Management Information Base (MIB) objects.
- Enable visibility to the PS Database for the subscriber through IPCable2Home-specified MIB objects.
- {informative text:Enable the manager to remotely discover devices connected to the home LAN and the UPnP Services running on them.}
- Process and log event messages.

The CMP is comprised of the following three functions to support the management and discovery responsibilities listed above. These functions are also shown in Figure 6-1:

1) *SNMP agent function*

   The SNMP agent function receives and processes SNMP messages from the WAN Interface through the WAN-Man IP address and from the LAN Interface, through the PS Server Router IP address. It provides access to MIB objects for the purpose of monitoring and/or configuring PS and LAN IP Device functionality.

2) *WAN Event handling function*

   The CMP reports events to cable operator's data network on the WAN according to the settings of the docsDevEvent table settings. The list of supported events appears in Annex B.

3) *Discovery function*

{informative text:The CMP, through its UPnP discovery functionality, acquires information about each UPnP Host device and the UPnP Services on it. The CMP stores this information in the PS database and makes it available to an SNMP management entity, via the PSDev MIB [see E.4] and QoS MIB [see E.7].}

These functions are described in 6.3.3.1 – 6.3.3.3.

6.3.3.1 CMP SNMP agent function

6.3.3.1.1 SNMP agent function goals

Goals of the SNMP Agent function of the CMP are listed below:

- Receive and process SNMP messages received through PS WAN-Man and PS Server Router (LAN) Interfaces.
- Provide SNMP manager access to the PS Database through IPCable2Home-specified MIBs.
- Enforce PS Database access rules defined by the docsDevNmAccessTable and VACM views.
- Support authentication and encryption/decryption processes for SNMP defined by IETF RFCs.
- Adhere to SNMP implementation rules and guidelines defined by IETF RFCs.

6.3.3.1.2 SNMP agent function system design guidelines

The system design guidelines listed in Table 6-3 have guided development of SNMP Agent Function requirements.

Table 6-3/J.192 – System design guidelines

Reference	Guidelines
SNMP Agent 1	The PS will provide remote access to manageable parameters in the PS database via specified MIBs.
SNMP Agent 2	The PS will implement an SNMP agent compatible with existing cable data network management systems.
SNMP Agent 3	The PS will support access control methods enabling the cable operator to configure control of PS Database access.

6.3.3.1.3 SNMP agent function system description

The CMP SNMP Agent function serves as the hub of Management control for WAN side management accesses and it gathers information for, and interconnects management of, WAN Management and LAN network elements. It also supports management messaging, via SNMP through any LAN Interface.

The CMP works in any of three network management modes:

- SNMP Provisioning Mode/SNMPv3 Coexistence Management Mode.
- DHCP Provisioning Mode/NmAccessTable Management Mode.
- DHCP Provisioning Mode/SNMPv3 Coexistence Management Mode.

SNMP Provisioning Mode/SNMP Coexistence Management Mode

As described in 5.5, when in SNMP 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 3414] and View-based Access Control Model

(VACM) [RFC 3415] are supported to allow the cable operator to implement management policy for access to specified MIBs.

DHCP Provisioning Mode/NmAccessTable Management Mode

As described in 5.5, when in DHCP Provisioning Mode, the PS defaults to operate in NmAccessTable mode. In NmAccessTable mode, management access is controlled by the NmAccessTable of the DOCSIS Device MIB [RFC 2669] and the SNMPv1/v2c protocols are supported.

The PS MUST apply the following rules in order to determine whether to permit SNMP access from a given source IP address (SrcIpAddr):

> If (docsDevNmAccessIp == "255.255.255.255"), permit the access from any SrcIpAddr.
>
> If ((docsDevNmAccessIp AND docsDevNmAccessIpMask) == (SrcIpAddr AND docsDevNmAccessIpMask)), permit the access from SrcIpAddr

The PS MUST assign 0.0.0.0 as the default value of docsDevNmAccessIpMask.

Table 6-4 illustrates the rules for docsDevNmAccessIp and docsDevNmAccessIpMask described above by providing sample MIB values and the access granted for each combination.

Table 6-4/J.192 – Example: MIB access granted for various values for docsDevNmAccessIp and docsDevNmAccessIpMask

docsDevNmAccessIp	docsDevNmAccessIpMask	Access
255.255.255.255	Any IP Address Mask	Any NMS
Any IP Address	0.0.0.0	Any NMS
Any IP Address except 255.255.255.255	255.255.255.255	Single NMS
0.0.0.0	255.255.255.255	No NMS

DHCP Provisioning Mode/SNMPv3 Coexistence Management Mode

When the PS is operating in DHCP Provisioning Mode, the cable operator can populate the Coexistence Table via SNMP set-request messages or via PS configuration file, thereby configuring the PS to operate in SNMPv3 Coexistence Management Mode. For a PS configured to operate in SNMPv3 Coexistence Mode, management access is controlled as described in [RFC 3584], the SNMPv1/v2c/v3 protocols are supported, USM and VACM are supported, and SNMPv3 keying material is distributed using [RFC 2786] and TLVs in the PS Configuration File.

Table 6-5 contains definitions for terms that are specific to the CMP.

Table 6-5/J.192 – Definition of terms

Management-control	Read or write access to a set of parameters that control or monitor the behaviour of the PS.
PS Database	A set of parameters that controls or monitors the behaviour 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 (section 2.1 of [RFC 3414]), 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 3415]).

Table 6-5/J.192 – Definition of terms

Ultimate Authorization	The single authority that establishes, modifies, or deletes User IDs, authentication keys, encryption keys, and access rights to the PS Database. 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.

Examples of the types of information that can be read or manipulated via IPCable2Home Management-control include the firewall policy settings, NMS-configured NAT mappings, remote diagnostic tool initiation and results access, PS status, discovered device and applications information, and LAN address range configuration. As will be illustrated later, the various management messaging interfaces may have access rights to different sets of parameters. A compliant PS supports access to the PS database through the MIB hierarchy from both the WAN and LAN using SNMP. Compliant IPCable2Home Host devices can also exchange messages with the Residential Gateway using XML-formatted data transported, via HTTP. Figure 6-2 indicates management messaging interfaces:

- NMS – CMP: Management message exchange between the cable network NMS and the CMP.
- CMP – IPCable2Home Host/LAN-Trans: Message exchange between the CMP and IPCable2Home Hosts in the LAN-Trans realm.
- CMP – IPCable2Home Host/LAN-Pass: Message exchange between the CMP and IPCable2Home Hosts in the LAN-Pass realm.
- NMS – LAN IP Device: Management message exchange between the cable network NMS and LAN IP Devices in the LAN-Pass realm. This management messaging is outside the scope of this Recommendation.

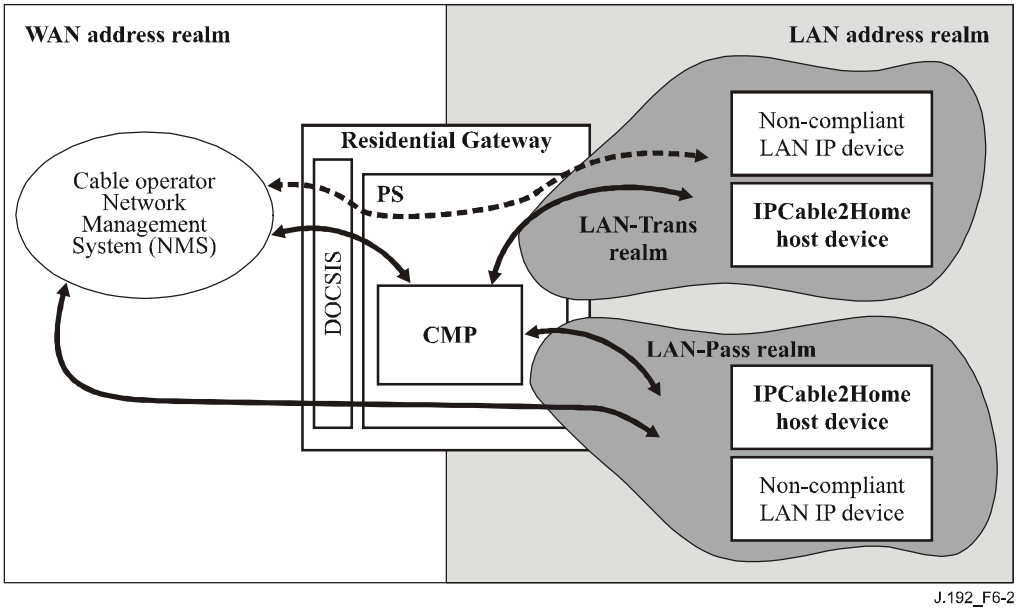


Figure 6-2/J.192 – IPCable2Home management message interfaces

The CMP is primarily a WAN (NMS) accessed and WAN controlled entity, but also supports access from the PS LAN interface (Server Router address – usually the default gateway for LAN IP Devices in the LAN-Trans realm). Additionally, the CMP may be called upon to inform the cable network NMS of events or transfer system log files as required. An example of a CMP implementation is illustrated in Figure 6-3, to convey concepts for CMP functionality.

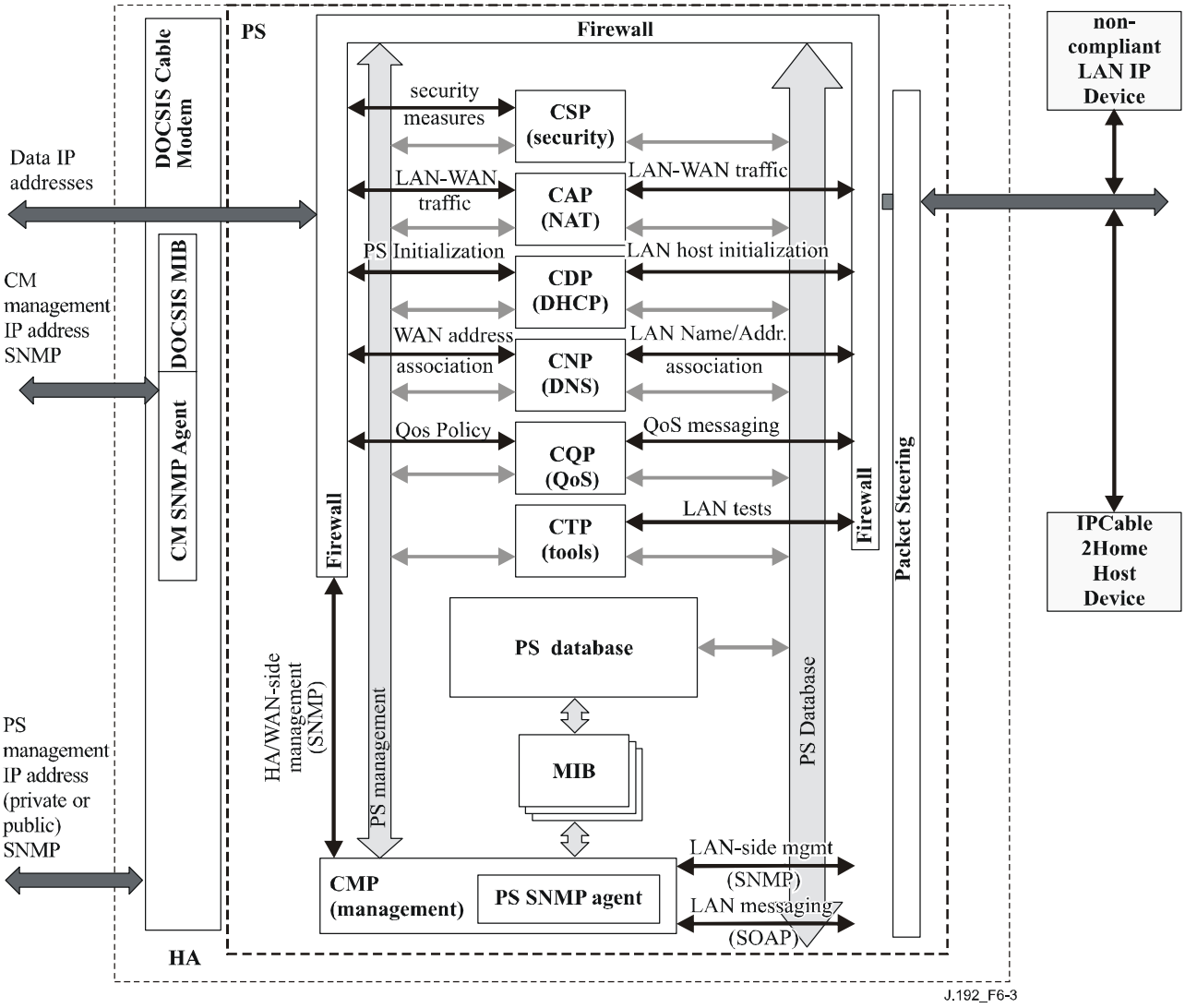


Figure 6-3/J.192 – PS block diagram

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.

The CMP SNMP agent function has the task of mapping the Object ID (OID) and the instance of the OID for all the leaves within the functional blocks in the PS, such as the CAP or local storage such as the PS Database.

A cable data network NMS operator may access or "manage" IPCable2Home Hosts in one of two ways. The cable operator can directly access IPCable2Home Hosts using passthrough addressing between the cable network and the LAN device element (BP) to be managed. The cable operator can also access BP Device profile attributes through the PSDev MIB in the PS and a list of BP applications and their priorities through the QoS MIB in the PS. The cable operator accesses these MIBs via SNMP set-request or SNMP get-request messages issued to the PS WAN-Man IP address and the PS, acting as a management proxy, accesses a BP using SOAP/HTTP. The cable operator can provision QoS Policy, in the form of QoS priorities for IPCable2Home Host applications, in the PS via SNMP.

6.3.3.1.4 SNMP agent function requirements

The PS MUST implement an SNMP agent compliant with IETF RFCs as indicated in 6.3.3.1.4.1, SNMP Protocol Requirements.

When operating in DHCP Provisioning Mode or SNMP Provisioning Mode (cabhPsDevProvMode = dhcpmode(1) or snmpmode(2)), the SNMP agent in the PS MUST only receive and process SNMP messages from the WAN that are addressed to its WAN-Man IP address.

When operating in DHCP Provisioning Mode or SNMP Provisioning Mode (cabhPsDevProvMode = dhcpmode(1) or snmpmode(2)), while in NAPT or NAT Primary Packet-handling Mode (cabhCapPrimaryMode = napt(1) or nat(2)), the SNMP agent in the PS MUST only receive and process SNMP messages from the LAN that are addressed to its LAN side CDP Server Router address (cabhCdpServerRouter).

When operating in DHCP Provisioning Mode or SNMP Provisioning Mode (cabhPsDevProvMode = dhcpmode(1) or snmpmode(2)) , while in Passthrough Primary Packet-handling Mode (cabhCapPrimaryMode = passthrough(3)), the SNMP agent in the PS MUST only receive and process SNMP messages from the LAN that are addressed to its LAN side Well Known PS LAN IP Address (192.168.0.1).

When operating in Dormant CableHome Mode (cabhPsDevProvMode = dormantCHmode(3)) and esafePsCableHomeModeStatus = dormantCHMode(3) [ITU-T Rec. J.126], the SNMP agent in the PS MUST ignore all SNMP messages from the WAN and MUST only receive and process SNMP messages from the LAN that are addressed to its LAN side CDP Server Router address (cabhCdpServerRouter).

The PS MUST ignore SNMP messages received through any LAN interface addressed to the PS WAN-Man IP address.

In the case of a PS co-resident with an embedded cable modem, i.e., an embedded PS, the PS and cable modem MUST respond to different and independent management IP addresses.

The PS MUST implement ICMP Echo and Echo Reply Message types (Type 8 and Type 0 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 PS MUST default to using SNMPv1/v2c for management messaging with the NMS and follow rules for NmAccess mode and Coexistence Mode, described in 6.3.3.1.4.2.1, Network Management Modes for a PS Operating in DHCP Provisioning Mode.

If the PS is operating in SNMP Provisioning Mode (indicated by a value of '2' in MIB object cabhPsDevProvMode), the PS MUST use SNMPv3 for management messaging with the NMS, following rules described in 6.3.3.1.4.3, Network Management Mode for a PS Operating in SNMP Provisioning Mode.

When the PS is operating in SNMP Coexistence Mode, the default Ultimate Authorization setting MUST be WAN Administrator (CHAdministrator).

The PS MUST include – in the following specified order – the hardware version, vendor name, boot ROM image version, software version, and model number in the sysDescr object (from [RFC 3418]). The format of the specific information contained in the sysDescr MUST be as shown in Table 6-6:

Table 6-6/J.192 – Format of sysDescr fields

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>

The sysDescr MUST be composed of a list of five Type/Value pairs enclosed in double angle brackets. The separation between the Type and Value is ": " – a colon and a blank space. For instance, a sysDescr of a PS of vendor X, hardware version 5.2, Boot ROM version 1.4, software version 2.2, and model number X would appear as follows:

anytext<<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 to determine what software and firewall policy versions the PS is capable of loading. If any fields of the sysDescr object are not applicable, the PS MUST report "NONE" as the value. For example, a PS with no BOOTR will report "BOOTR: NONE".

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.

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 sysObjectID object of the MIB-2 System group [RFC 3418] MUST be implemented and MUST be persistent across device reset and power cycles.

The sysUpTime object of the MIB-2 System group [RFC 3418] MUST be implemented. SysUpTime is the amount of time that has elapsed since the system reset.

The sysContact object of the MIB-2 System group [RFC 3418] MUST be implemented and MUST be persistent across device reset and power cycles. SysContact returns the name of the user or system administrator if known.

The sysLocation object of the MIB-2 System group [RFC 3418] MUST be implemented and MUST be persistent across device reset and power cycles.

The sysServices object of the MIB-2 System group [RFC 3418] MUST be implemented and MUST be persistent across device reset and power cycles.

The sysName object of the MIB-2 System group [RFC 3418] MUST be implemented and MUST be persistent across device reset and power cycles. Querying sysName returns the system name.

The Interfaces Group MIB [RFC 2863] MUST be implemented in accordance with Annex A and requirements in 6.3.3.1.4.8.

The MIB-2 SNMP group [RFC 3418] MUST be implemented.

The snmpSetSerialNo object of the snmpSet group [RFC 3418] MUST be implemented. SnmpSetSerialNo is an advisory lock used to allow several cooperating SNMPv2 entities, all acting in a manager role, to coordinate their use of the SNMPv2 set operation.

The PS MUST count LAN-to-WAN and WAN-to-LAN octets as defined by cabhPsDevLanIpTrafficTable [see E.4], according to the value of cabhPsDevLanIpTrafficEnabled [see E.4].

When the PS element MIB objects are set to their factory defaults using the cabhCapSetToFactory, cabhCdpSetToFactory, cabhCtpSetToFactory, or cabhPsDevSetToFactory MIB objects the corresponding PS functionality MUST use these factory default settings for operation without having to re-provision the PS element.

6.3.3.1.4.1 SNMP protocol requirements

The PS MUST adhere to or implement, as appropriate, the following IETF RFCs:

- "A Simple Network Management Protocol" [RFC 1157].

  NOTE 1 – This RFC has been declared "historic" by [RFC 3410]. The PS is required to support SNMPv1.
- "Introduction to Community-based SNMPv2" [RFC 1901].

  NOTE 2 – This RFC has been declared "historic" by [RFC 3410]. The PS is required to support SNMPv2c.
- "Introduction and Applicability Statements for Internet Standard Management Framework" [RFC 3410].
- "An Architecture for Describing Simple Network Management Protocol (SNMP) Management Frameworks" [RFC 3411].
- "Message Processing and Dispatching for SNMP" [RFC 3412].
- "Simple Network Management Applications" [RFC 3413].
- "User-based Security Model (USM) for version 3 of the Simple Network Management Protocol" [RFC 3414].
- "View-based Access Control Model (VACM) for the Simple Network Management Protocol" [RFC 3415].
- "Version 2 of the Protocol Operations for the Simple Network Management Protocol (SNMP)" [RFC 3416].
- "Transport Mappings for the Simple Network Management Protocol" [RFC 3417].
- "Management Information Base (MIB) for the Simple Network Management Protocol (SNMP)" [RFC 3418].
- "Coexistence between Version 1, Version 2, and Version 3 of the Internet-Standard Network Management Framework" [RFC 3584].

In support of SMIv2, the PS MUST implement the following IETF RFCs:

- "Structure of Management Information Version 2 (SMIv2)" [RFC 2578].
- "Textual Conventions for SMIv2" [RFC 2579].
- "Conformance Statements for SMIv2" [RFC 2580].

6.3.3.1.4.2 Network management mode requirements

Clause 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. Clauses 7.3.3.1 and 7.3.3.2 provide additional detail about PS operation in each of the two provisioning modes, in addition to Dormant CableHome Mode of operation.

This clause describes rules for the network management modes the PS is required to support. Clause 6.3.3.1.4.2.1 describes network management modes for a PS operating in DHCP Provisioning Mode. Clause 6.3.3.1.4.3 describes network management modes for a PS operating in SNMP Provisioning Mode.

The PS can operate in SNMPv3 Coexistence network management mode, regardless of whether it is configured to operate in DHCP Provisioning Mode or SNMP Provisioning Mode. It defaults to operation in SNMPv3 Coexistence mode when operating in SNMP Provisioning Mode. When operating in DHCP Provisioning Mode, the PS defaults to operating in NmAccessTable network management mode, but can be configured to operate in SNMPv3 Coexistence Mode.

Control of access to the MIBs implemented by the PS depends upon the network management mode in which the PS is configured to operate. When the PS is configured to operate in NmAccessTable network management mode, MIB access is controlled by writing to the docsDevNmAccessTable [RFC 2669]. When operating in SNMPv3 Coexistence Mode, access to the MIBs is controlled by the SNMPv3 tables ([RFC 3584], [RFC 3413], [RFC 3414] and [RFC 3415]). The SNMPv3 tables can be configured by the NMS through SNMP Set commands, or via the PS Configuration File. Clause 6.3.3.1.4.6, Mapping TLV Fields Into Created SNMPv3 Table Rows, describes how PS Configuration File configuration parameters are mapped into these SNMPv3 tables.

6.3.3.1.4.2.1 Network management modes for a PS operating in DHCP provisioning mode

The PS MUST support SNMPv1, SNMPv2c, and SNMPv3 and SNMP Coexistence as described by [RFC 3411] through [RFC 3415] and [RFC 3584]. The PS MUST also support NmAccessTable mode as defined by [RFC 2669]. Support for the network management modes for a PS operating in DHCP Provisioning Mode is subject to the guidelines described in 6.3.3.1.4.2.2, 6.3.3.1.4.3 and 6.3.3.1.4.4.

6.3.3.1.4.2.2 Basic operation for a PS operating in DHCP provisioning mode

Initial operation of the PS configured for DHCP Provisioning Mode can be thought of as having three steps: 1) behaviour 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) behaviour 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:
   - All SNMP packets are dropped.
   - None of the SNMPv3 MIBs (Community MIB, TARGET-MIB, VACM-MIB, USM-MIB, NOTIFICATION-MIB) are accessible to the SNMP manager in the NMS.
   - None of the elements in the SNMP-USM-DH-OBJECTS-MIB is accessible 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.
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 docsDevNmAccessTable 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 TLV 38), 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.

3) After completion of the provisioning process described in 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 SNMPv1/v2c

- The PS MUST process SNMPv1/v2c packets and drop SNMPv3 packets.
- docsDevNmAccessTable controls access and trap destinations as described in [RFC 2669]. The PS operating in NmAccess Network Management Mode MUST enforce the management access policy, as defined by the NmAccessTable, for any access to the IPCable2Home-specified MIB objects, regardless of the interface (such as a vendor-specific graphical user interface (GUI)) 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 docsDevNmAccess 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 { DisableSNMPv2trap }
     ::={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 11.4.4.1.3 and 11.4.4.1.4. 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 3411], [RFC 3412], [RFC 3413], [RFC 3414], [RFC 3415] and [RFC 3584].
- 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 IPCable2Home-specified MIB objects, regardless of the interface (such as a vendor-specific graphical user interface (GUI)) or access protocol used.
- Community MIB controls the translation of SNMPv1/v2c packet community string into security name which selects 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.

6.3.3.1.4.3 Network management mode for a PS operating in SNMP provisioning mode

If the PS is operating in SNMP Provisioning Mode following DHCP ACK (as indicated by a value '2' (SNMPmode) for cabhPsDevProvMode), it operates in SNMPv3 Coexistence Mode using SNMPv3 by default for exchanging management messages with the NMS, and uses Kerberos for exchanging key material with the KDC, following rules described in this clause. Just as when the PS is operating in DHCP Provisioning Mode and has been configured for SNMPv3 Coexistence network management mode, when the PS is operating in SNMP Provisioning Mode and SNMPv3 Coexistence network management mode, it is required to ignore attempts to configure the docsDevNmAccessTable.

6.3.3.1.4.4 Management views

The management controls defined for IPCable2Home are in the CMP function of the PS. Settings, based on management mode, define the access rights that are granted to a User for access to the Portal Services database, through IPCable2Home-specified MIBs, via SNMP from the PS WAN-Man or LAN Server Router interfaces. A single User is defined by this Recommendation.

The concept of Management Views was introduced with SNMPv3 and is defined in [RFC 3410] through [RFC 3415] and [RFC 3584]. It is a method for specifying what user(s) is/are allowed to access which MIB object(s).

Figure 6-4 illustrates some possible management Views for the PS. A WAN Administrator View (CHAdministrator view) and a WAN Administrator User (CHAdministrator user) are defined by this Recommendation. 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 (CHAdministrator), following rules defined in [RFC 3414] and [RFC 3415].

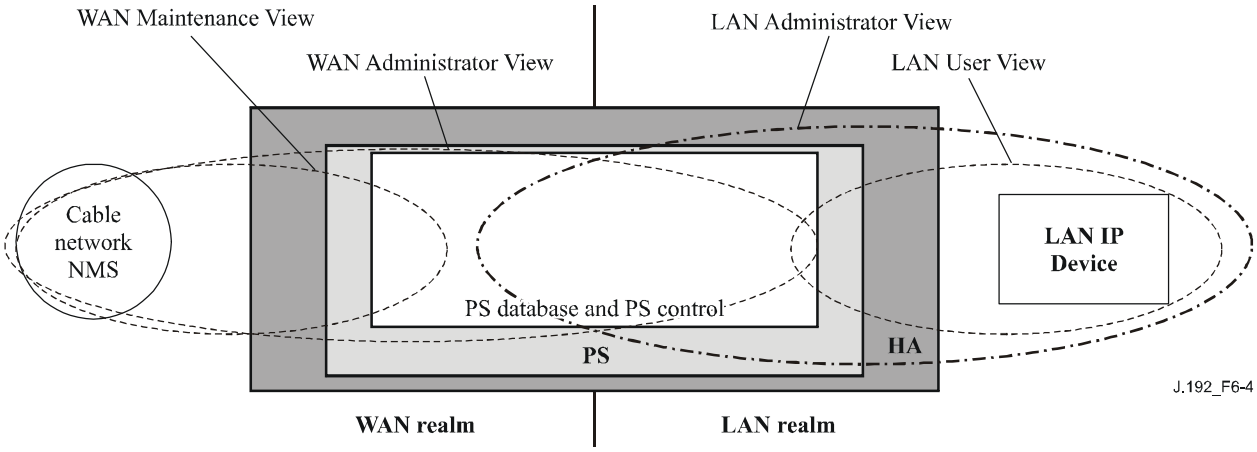


Figure 6-4/J.192 – Management views

Managed parameters defined by IPCable2Home are stored in the PS Database. As shown in Figure 6-4, there is a concept of Access Views into the PS Database and PS Control, which allows simultaneous management from both the LAN and WAN by defining Management Views into the PS Database and PS Control. The Views are a mechanism to provide privacy and security, and the policy can be set separately by the CHAdministrator User.

The Ultimate Authorization (CHAdministrator User) has its own User ID and keys, and has the following responsibilities:

- Responsible for setting up all access Views on both the LAN and WAN management interface.
- Responsible for creating and managing all User profiles including user IDs, Keys, and PS database access privileges.
- Responsible for setting policy for both LAN and WAN side access.

Descriptions for how View-based Access Control Model and User-based Security Model work are provided in [RFC 3414] and [RFC 3415].

The CHAdministrator View provides full read and write access to all MIBs specified by IPCable2Home.

Management View requirements are specified in 6.3.3.1.4.5.

6.3.3.1.4.4.1 WAN-Access Control

IPCable2Home defines two methods for controlling access to manageable parameters via IPCable2Home-defined MIBs. The docsDevNmAccessTable [RFC 2669] defines management access when the PS is operating in NmAccess Network Management Mode (refer to 6.3.3.1.4.2.2). When the PS is operating in SNMPv3 Coexistence Network Management Mode (see 6.3.3.1.4.2.2), per User Security Model (USM) [RFC 3414] and View-based Access Control Model (VACM) [RFC 3415], Table settings are used to control access to IPCable2Home-specified MIB objects, regardless of the interface (such as a graphical user interface) through which the request arrives. VACM defines a set of services that can be used for checking access rights. VACM Groups define the rights to access the CMP.

As defined in RFC 3415 clause 2.4, a "MIB View" is a specific set of managed object types that can be defined, and this concept is used in IPCable2Home to support WAN Management of the PS. The CHAdministrator User access and View are specified in 11.4.4.1.3 and 6.3.3.1.4.5. An example sequence of PS Database access from the WAN interface is provided in 12.3.1.

6.3.3.1.4.4.2 Security

Security of management messages is provided by SNMPv3. Refer to clause 11 for a detailed description of how SNMPv3 is used. The CMP may use SNMPv3 to counter threats identified in Annex C.

To protect against replay attacks, a time of day clock is utilized to provide timestamps for messaging. Management messaging security requirements are specified in 11.4.

6.3.3.1.4.5 View-based Access Control Model (VACM) requirements

To provide controlled access to management information and the creation of distinct management realms for a PS operating in SNMPv3 Coexistence Mode, View-based Access Control Model (VACM) MUST be employed as defined by [RFC 3415].

The WAN Administrator View MUST be implemented in a compliant 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.

The User specification for the WAN Administrator View MUST be implemented as follows:

vacmSecurityModel	3 (USM)
vacmSecurityName	'CHAdministrator'
vacmGroupName	'CHAdministrator'
vacmSecurityToGroupStorageType	permanent
vacmSecurityToGroupStatus	active

The Group specification for the CHAdministrator View MUST be implemented as follows:

CHAdministrator Group	
vacmGroupName	'CHAdministrator'
vacmAccessContextPrefix	''
vacmAccessSecurityModel	3 (USM)
vacmAccessSecurityLevel	AuthPriv
vacmAccessContextMatch	exact
vacmAccessReadViewName	'CHAdministratorView'
vacmAccessWriteViewName	'CHAdministratorView'
vacmAccessNotifyViewName	'CHAdministratorView'
vacmAccessStorageType	permanent
vacmAccessStatus	active

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

CHAdministratorView subtree 1.3.6.1 (Entire MIB)

6.3.3.1.4.6 Mapping TLV fields into created SNMPv3 table rows

This clause details how the *SNMP Notification Receiver* Configuration File Element (TLV Type 38) is mapped into SNMPv3 functional tables. Refer to 7.4.4.1.10, SNMP Notification Receiver, for a description of configuration parameter TLV Type 38. Details of how the encryption keys are exchanged for SNMPv3 operation are provided in 11.4.4.2.2.

Upon receiving one Type 38 configuration file element, the PS MUST make MIB table entries following the procedure described in Tables 6-7, snmpNotifyTable, through 6-16, vacmSecurityToGroupTable, using values passed in the TLV as described below. The MIB tables

the PS is required to populate when it receives a Type 38 configuration file element are listed below for convenience:

- snmpNotifyTable;
- snmpTargetAddrTable;
- snmpTargetAddrExtTable;
- snmpTargetParamsTable;
- snmpNotifyFilterProfileTable;
- snmpNotifyFilterTable;
- snmpCommunityTable;
- usmUserTable;
- vacmSecurityToGroupTable;
- vacmAccessTable;
- vacmViewTreeFamilyTable.

A PS configuration file is allowed to contain TLV MIB elements (Type 28) that make entries to any of the 11 tables listed above.

The tables in this clause show how the fields from the PS Configuration file TLV element (the tags in angle brackets <>) are placed into the SNMPv3 tables.

The correspondence between TLV fields and table tags <TAG> is shown below:

PS<IP Address> TLV 38.1

<Port> TLV 38.2

<Trap type> TLV 38.3

<Timeout> TLV 38.4

<Retries> TLV 38.5

<Filter OID> TLV 38.6

<Security Name> TLV 38.7

These tables are shown in the order that the agent will search down through them when a notification is generated in order to determine who to send the notification to and how to fill out the contents of the notification packet.

snmpNotifyTable

Create two rows with fixed values, if one or more TLV elements are present.

Table 6-7/J.192 – snmpNotifyTable [RFC 3413]

SNMP-NOTIFICATION-MIB	First Row	Second Row
Column Name (* = Part of Index)	Column Value	Column Value
* snmpNotifyName	"@PSconfig_inform"	"@PSconfig_trap"
snmpNotifyTag	"@PSconfig_inform"	"@PSconfig_trap"
snmpNotifyType	inform(2)	trap(1)
snmpNotifyStorageType	volatile	volatile
snmpNotifyRowStatus	active(1)	active(1)

snmpTargetAddrTable

Create one row for each TLV element in the PS configuration file.

Table 6-8/J.192 – snmpTargetAddrTable [RFC 3413]

SNMP-TARGET-MIB	New Row
Column Name (* = Part of Index)	Column Value
* snmpTargetAddrName	"@PSconfig_n", where n ranges from 0 to m – 1, and m is the number of notification receiver TLV elements in the PS configuration file
snmpTargetAddrTDomain	snmpUDPDomain – snmpDomains
snmpTargetAddrTAddress (IP Address and UDP Port of the Notification Receiver)	OCTET STRING (6) Octets 1-4: <IP Address> Octets 5-6: <Port>
snmpTargetAddrTimeout	<Timeout> from the TLV
snmpTargetAddrRetryCount	<Retries> from the TLV
snmpTargetAddrTagList	If <Trap type> == 1, 2, or 4 "@PSconfig_trap" Else If <Trap type> = 3 or 5 "@PSconfig_inform"
snmpTargetAddrParams	"@PSconfig_n" (same as snmpTargetAddrName value)
snmpTargetAddrStorageType	volatile
snmpTargetAddrRowStatus	active(1)

snmpTargetAddrExtTable

Create one row for each TLV element in the PS configuration file.

Table 6-9/J.192 – snmpTargetAddrExtTable [RFC 3584]

SNMP-COMMUNITY MIB	New Row
Column Name (* = part of index)	Column Value
* snmpTargetAddrName	"@PSconfig_n", where n ranges from 0 to m – 1, and m is the number of notification receiver TLV elements in the PS configuration file
snmpTargetAddrTMask	<zero length octet string>
snmpTargetAddrMMS	0

snmpTargetParamsTable

Create one row for each TLV element in the config file. If <Trap type> is 1, 2, or 3, or if the <Security Name> Field is zero-length, create the table as follows:

Table 6-10/J.192 – snmpTargetParamsTable for <Trap Type> 1, 2, or 3 [RFC 3413]

SNMP-TARGET-MIB	New Row
Column Name (* = part of index)	Column Value
* snmpTargetParamsName	"@PSconfig_n", where n ranges from 0 to m – 1, and m is the number of notification receiver TLV elements in the PS configuration file
snmpTargetParamsMPModel SYNTAX: SnmpMessageProcessingModel	If <Trap type> = 1 SNMPv1(0) Else if <Trap type> = 2 or 3 SNMPv2c(1) Else if <Trap type> = 4 or 5 SNMPv3(3)
snmpTargetParamsSecurityModel SYNTAX: SnmpSecurityModel	If <Trap type> = 1 SNMPv1(1) Else if <Trap type> = 2 or 3 SNMPv2c(2) Else if <Trap type> = 4 or 5 USM(3) NOTE – The mapping of SNMP protocol types to value here are different from snmpTargetParamsMPModel
snmpTargetParamsSecurityName	"@PSconfig"
snmpTargetParamsSecurityLevel	noAuthNoPriv
snmpTargetParamsStorageType	volatile
snmpTargetParamsRowStatus	active(1)

If <Trap type> is 4 or 5, and the <Security Name> field is non-zero length, create the table as follows:

Table 6-11/J.192 – snmpTargetParamsTable for <Trap Type> 4 or 5 [RFC 3413]

SNMP-TARGET-MIB	New Row
Column Name (* = part of index)	Column Value
* snmpTargetParamsName	"@PSconfig_n", where n ranges from 0 to m – 1, and m is the number of notification receiver TLV elements in the PS configuration file
snmpTargetParamsMPModel SYNTAX: SnmpMessageProcessingModel	If <Trap type> = 1 SNMPv1(0) Else if <Trap type> = 2 or 3 SNMPv2c(1) Else if <Trap type> = 4 or 5 SNMPv3(3)

Table 6-11/J.192 – snmpTargetParamsTable for <Trap Type> 4 or 5 [RFC 3413]

snmpTargetParamsSecurityModel SYNTAX: SnmpSecurityModel	If <Trap type> = 1 SNMPv1(1) Else if <Trap type> = 2 or 3 SNMPv2c(2) Else if <Trap type> = 4 or 5 USM(3) NOTE – The mapping of SNMP protocol types to value here are different from snmpTargetParamsMPModel
snmpTargetParamsSecurityName	<Security Name>
snmpTargetParamsSecurityLevel	The security level of <Security Name>
snmpTargetParamsStorageType	volatile
snmpTargetParamsRowStatus	active(1)

snmpNotifyFilterProfileTable

Create one row for each TLV that has a non-zero <Filter Length>.

Table 6-12/J.192 – snmpNotifyFilterProfileTable [RFC 3413]

SNMP-NOTIFICATION-MIB	New Row
Column Name (* = Part of Index)	Column Value
*snmpTargetParamsName	"@PSconfig_n", where n ranges from 0 to m – 1 and m is the number of notification receiver TLV elements in the PS configuration file.
snmpNotifyFilterProfileName	"@PSconfig_n", where n ranges from 0 to m – 1 and m is the number of notification receiver TLV elements in the PS configuration file.
snmpNotifyFilterProfileStorType	volatile
snmpNotifyFilterProfileRowStatus	active(1)

snmpNotifyFilterTable

Create one row for each TLV that has a non-zero <Filter Length>.

Table 6-13/J.192 – snmpNotifyFilterTable [RFC 3413]

SNMP-NOTIFICATION-MIB	New Row
Column Name (* = Part of Index)	Column Value
* snmpNotifyFilterProfileName	"@PSconfig_n" , where n ranges from 0 to m – 1 and m is the number of notification receiver TLV elements in the PS configuration file.
* snmpNotifyFilterSubtree	<Filter OID> from the TLV
snmpNotifyFilterMask	<Zero Length Octet String>
snmpNotifyFilterType	included(1)
snmpNotifyFilterStorageType	volatile
snmpNotifyFilterRowStatus	active(1)

snmpCommunityTable

Create one row with fixed values if one or more TLVs are present. This causes SNMPv1 and v2c Notifications to contain the community string in snmpCommunityName.

Table 6-14/J.192 – snmpCommunityTable [RFC 3584]

SNMP-COMMUNITY-MIB	First Row
Column Name (* = Part of Index)	Column Value
* snmpCommunityIndex	"@PSconfig"
snmpCommunityName	"public"
snmpCommunitySecurityName	"@PSconfig"
snmpCommunityContextEngineID	<The PS engineID>
snmpCommunityContextName	<Zero length octet string>
snmpCommunityTransportTag	<Zero length octet string>
snmpCommunityStorageType	volatile
snmpCommunityStatus	active(1)

usmUserTable

Create one row with fixed values, if one or more TLVs are present. Other rows are created, one each time the engine ID of a trap receiver is discovered. This specifies the user name on the remote notification receivers to send notifications to.

One row in the usmUserTable is created. Then when the engine ID of each notification receiver is discovered, the agent copies this row into a new row and replaces the 0x00 in the usmUserEngineID column with the newly discovered value.

Table 6-15/J.192 – usmUserTable [RFC 3414]

SNMP-USER-BASED-SM-MIB	First Row
Column Name (* = Part of Index)	Column Value
* usmUserEngineID	0
* usmUserName	"@PSconfig" When other rows are created, this is replaced with the <Security Name> field from the TLVelement.
usmUserSecurityName	"@PSconfig" When other rows are created, this is replaced with the <Security Name> field from the TLVelement.
usmUserCloneFrom	<don't care> – cannot clone this row
usmUserAuthProtocol	None. When other rows are created, this is replaced with None or MD5, depending upon the security level of the v3 User.
usmUserAuthKeyChange	<don't care> – write only
usmUserOwnAuthKeyChange	<don't care> – write only
usmUserPrivProtocol	None. When other rows are created, this is replaced with None or DES, depending on the security level of the v3 User.
usmUserPrivKeyChange	<don't care> – write only
usmUserOwnPrivKeyChange	<don't care> – write only
usmUserPublic	<zero length string>

Table 6-15/J.192 – usmUserTable [RFC 3414]

usmUserStorageType	volatile
usmUserStatus	active(1)

vacmSecurityToGroupTable

Create three rows with fixed values, if one or more TLVs are present.

These are the three rows with fixed values, which are used for the TLV entries with <Trap Type> set to 1, 2, or 3 or with a zero length <Security Name>.

Table 6-16/J.192 – vacmSecurityToGroupTable [RFC 3415]

SNMP-VIEW-BASED-ACM-MIB	First Row	Second Row	Third Row
Column Name (* = Part of Index)	Column Value	Column Value	Column Value
* vacmSecurityModel	SNMPv1(1)	SNMPv2c(2)	USM(3)
* vacmSecurityName	"@PSconfig"	"@PSconfig"	"@PSconfig"
vacmGroupName	"@PSconfigv1"	"@PSconfigv2"	"@PSconfigUSM"
vacmSecurityToGroupStorageType	volatile	volatile	volatile
vacmSecurityToGroupStatus	active(1)	active(1)	active(1)

6.3.3.1.4.7 IPCable2Home MIB requirements

The PS MUST implement each MIB object listed in Annex A. If the Persistent column for a MIB object listed in Annex A contains the value Yes, the PS MUST retain the value of the object across a PS power cycle or re-boot, making the same value available for access by an SNMP manager immediately after provisioning complete (cabhPsDevProvState = pass(1)), following a reboot that was available for access by that SNMP manager immediately before reboot.

Required MIB objects are from the following MIB documents:

- Interfaces Group MIB [RFC 2863].
- DOCSIS Cable Device MIB [RFC 2669].
- CableLabs Definition MIB [see E.6].
- IPCable2Home PSDev MIB [see E.4].
- IPCable2Home CAP MIB [see E.1].
- IPCable2Home CDP MIB [see E.2].
- IPCable2Home CTP MIB [see E.3].
- IPCable2Home Security MIB [see E.5].
- IPCable2Home QoS MIB [see E.7].
- [draft-ietf-ipcdn-bpiplus-mib-05].
- IP MIB (SNMPv2) [RFC 2011].
- UDP MIB (SNMPv2) [RFC 2013].
- Diffie-Hellman USM Key [RFC 2786].
- INET Address MIB [RFC 3291].
- DOCS IF MIB [RFC 2670].
- IANA ifType MIB [IANAType].
- IEEE 802.11 MIB [802dot11MIB].

In an IPCable2Home Residential Gateway or any other device with an embedded PS and embedded cable modem, the cable modem management entity and PS management entity (CMP) MUST respond to different and independent management IP addresses. ITU-T Rec. J.112 and this Recommendation specify some of the same MIB objects but if a J.112-compliant cable modem and an 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 a PS with an embedded cable modem, software download of the single image of the combined cable modem software and Portal Services software is controlled by the cable modem. The following docsDevSoftware group objects [RFC 2669] MUST NOT be implemented for a PS with an embedded cable modem, i.e., these objects MUST only be accessible through the cable modem management IP address in a PS with an embedded CM:

- docsDevSwServer.
- docsDevSwFilename.
- docsDevSwAdminStatus.
- docsDevSwOperStatus.

The docsDevSoftware Group of objects MUST be implemented in a Standalone PS. Modification of the docsDevSoftware objects (as specified in 11.8.4) by the cable operator for the purpose of downloading the standalone PS software image MUST result in proper secure software download operation.

In a PS with an embedded cable modem, cable modem MIB objects MUST only be visible and accessible when the manager accesses them through the cable modem management IP address, and MUST NOT be visible or accessible via any PS 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.

In a PS with an embedded cable modem, 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) or through the PS LAN Server Router 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.

The general MIB hierarchy is illustrated in Figure 6-5. Specific OIDs required for individual MIBs are listed in Annex A.

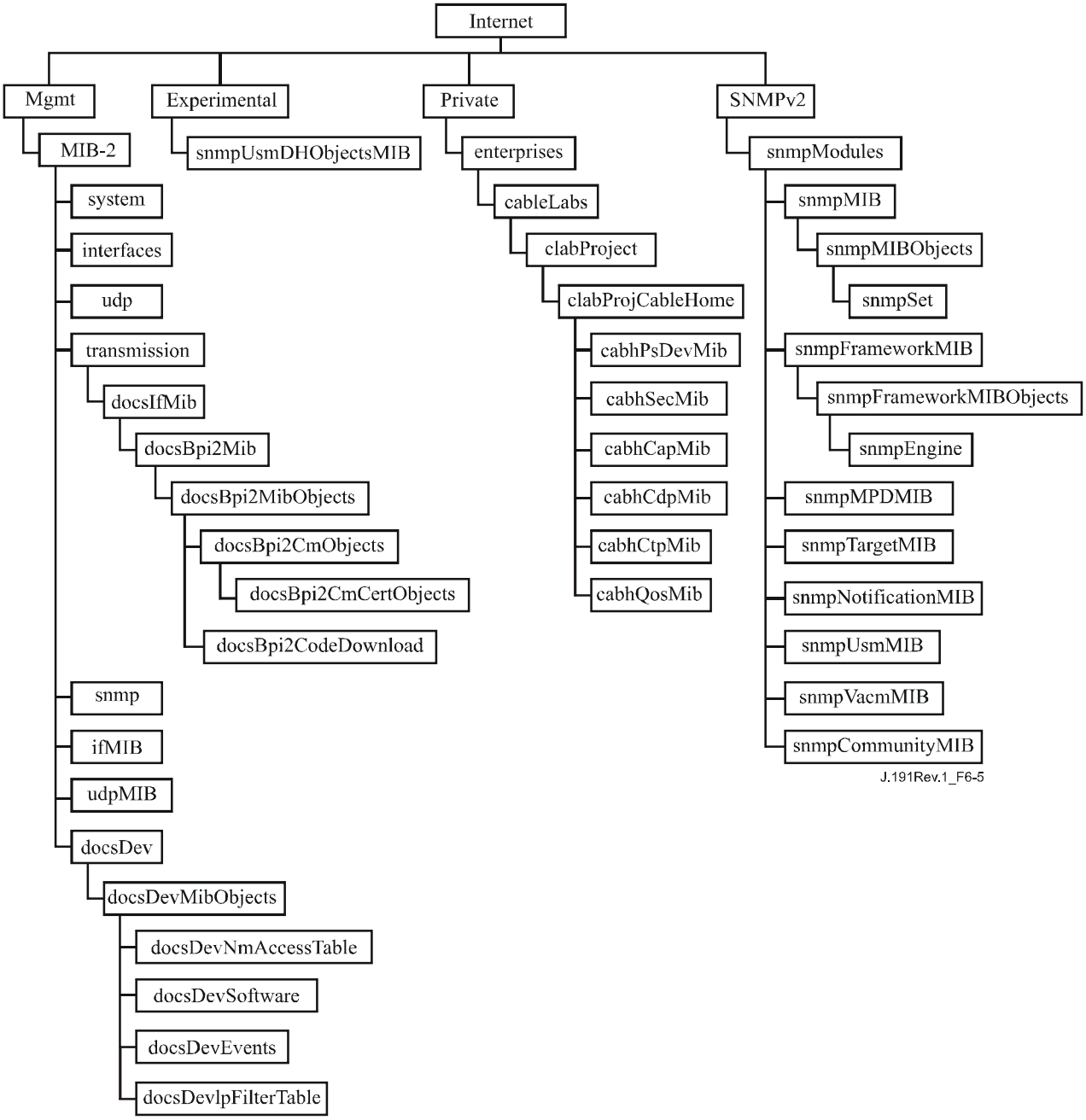


Figure 6-5/J.192 – IPCable2Home MIB hierarchy

6.3.3.1.4.8 Interfaces Group MIB

The Interfaces Group MIB [RFC 2863] provides a powerful tool to allow cable operators to understand the state of and see statistics for all of the physical interfaces on the Portal Service element. A *physical interface* is one for which a connector is exposed on the exterior of the device enclosure, and for which the object *ifConnectorPresent* is true. In order to enable the intelligent use of this MIB, an interface numbering scheme is essential. Therefore PS elements need to comply to the following requirements:

The PS MUST implement an instance of ifEntry for the WAN-Data interface of the PS element, even if that interface is internal – as exists in the case of an Embedded PS utilizing an integrated chip design.

The PS MUST implement an instance of ifEntry for each physical LAN interface of the PS element.

The PS MUST implement an instance of ifEntry for an "Aggregated LAN Interfaces" interface, which is identified by the ifIndex value 255.

The PS MUST implement an instance of ifEntry for an "Aggregated Wireless LAN Interfaces" (virtual) interface, representing the superset of all physical wireless LAN interfaces implemented on the product, and identified by the ifIndex value 254.

The PS ifTable interfaces MUST be numbered as shown in Table 6-17.

Table 6-17/J.192 – Numbering interfaces in the ifTable

Interface	Description
1	WAN-Man Interface
2	WAN-Data Interface
2 + n	Each LAN Interface
254	Aggregated Wireless LAN Interface
255	Aggregated LAN Interface

If a given interface's ifAdminStatus = down, that interface MUST NOT accept or forward any traffic. The ifAdminStatus object corresponding to ifIndex value 255 MUST provide administrative control over all LAN interfaces and MUST be implemented as read-write.

The PS MUST assign the value other(1) to ifTable [RFC 2863] ifType entries corresponding to ifIndex 255. The PS MUST assign the value other(1) to ifTable ifType entries corresponding to ifIndex 254. An embedded PS element MUST assign the value other(1) to ifTable ifType entries corresponding to ifIndex values 1 and 2. A standalone PS element MUST assign the appropriate IANAifType [IANAType] value to the ifTable ifType value corresponding ifIndex values 1 and 2.

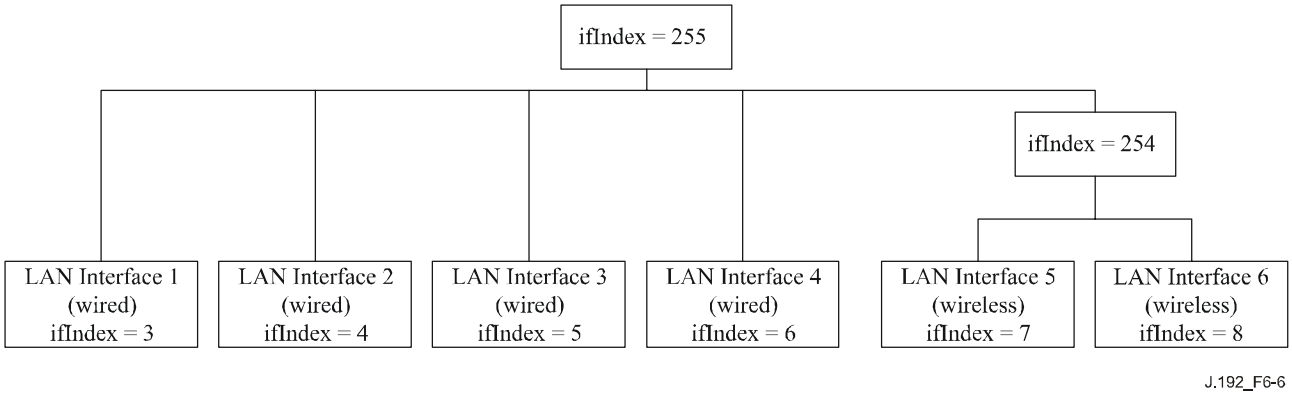
The ifTable ifPhysAddress value corresponding to ifIndex 255 MUST be a zero length octet string.

The ifTable ifPhysAddress value corresponding to ifIndex 254 MUST be a zero length octet string.

The ifTable counters of WAN interfaces of ifIndex values 1 and 2 MUST be shared between the two interfaces. The PS MAY implement ifTable counters for ifIndex values 254 and 255.

The PS interfaces MUST be implemented according to the layer and sub-layer definitions described in section 3.1 of [RFC 2863], with all PS interfaces in the range (3-254) implemented as sub-layers of interface 255, and all PS wireless LAN physical interfaces implemented as sub-layers of interface 254.

The Interface Stack (ifStack) group of [RFC 2863] MUST be implemented to identify relationships among the higher-layer "Aggregated LAN Interfaces" interface and the lower-layer (wired) LAN sub-interfaces to identify the relationship between the higher-layer "Aggregated LAN Interfaces" interface and the lower-layer "Aggregated Wireless LAN Interfaces" interface, and to identify the relationship between the "Aggregated Wireless LAN Interfaces" interface and the lower-layer wireless LAN sub-interfaces. Figure 6-6 illustrates the use of the ifStack group for a PS with four wired and two wireless LAN interfaces.



Implementation of ifStack for this example:

ifStackHigherLayer	0	1	0	2	0	255	255	255	255	255	254	254	3-8
ifStackLowerLayer	1	0	2	0	255	3	4	5	6	254	7	8	0

Figure 6-6/J.192 – ifStack Implementation Example

6.3.3.1.4.9 IEEE 802.11 Wireless LAN MIB

If the PS implements IEEE 802.11 wireless LAN functionality, then the PS MUST support the wireless 802.11 MIB and the implementation applicable extensions among [802.11A-1999], [802.11B/Cor1-2001], [802.11D], [802.11G-2003] as specified in Annex A.

Table 6-18/J.192 – IEEE 802.11 MIB module requirements

MIB structure	Notes
dot11StationConfigEntry	Required
dot11WEPDefaultKeysTable	Required
dot11PrivacyEntry	Required
dot11OperationEntry	Required
dot11PhyTxPowerEntry	Required
dot11PhyDSSSEntry	Required for 802.11B and 802.11G
dot11PhyOFDMTable	Required if 802.11A is supported
cabhPsDev802dot11BaseTable	Required
dot11AuthenticationAlgorithmsEntry	Optional
dot11MultiDomainCapabilityEntry	Optional
dot11PhyOperationEntry	Optional
dot11RegDomainsSupportEntry	Optional
dot11SupportedDataRatesTxEntry	Optional
dot11SupportedDataRatesRxEntry	Optional
dot11PhyHRDSSSTable	Optional, only if 802.11B is supported
dot11PhyERPTable	Optional, only if 802.11G is supported

The following subclauses detail IPCable2Home specific requirements for the MIB requirements listed in Table 6-18.

6.3.3.1.4.9.1 Specific 802.11 MIB Requirements

802.11 MIB objects not listed in Annex A are considered OPTIONAL, and MAY not need to be instantiated in the required tables.

If OPTIONAL 802.11 MIB Objects defined herein are instantiated, this indicates that the protocol primitives associated to the PHY or MAC management layer are implemented, therefore values MUST be reported accordingly, specifically:

– Counters and statistics MUST increment based on their operations.

– Dynamic MIB objects with read-only SYNTAX MUST report accurate values.

– Configurable MIB objects with read-write SYNTAX MAY be implemented as read-only.

6.3.3.1.4.9.1.1 Requirements for dot11StationConfigEntry

MIB objects dot11BeaconPeriod, dot11DTIMPeriod and dot11AssociationResponseTimeOut MAY be implemented as read-only.

The MIB object dot11OperationalRateSet, which MAY have read-only access, MUST correspond to values in the context of cabhPsDev802dot11PhyOperMode. If the MIB object dot11OperationalRateSet implements read-write access , an SNMP set operation to a data rate not in the context of cabhPsDev802dot11PhyOperMode MUST report an SNMP error 'wrongValue'.

Support of Multi-domain features [802.11D] varies among regulatory domain, in particular for the regulatorydomain 0x00 (FCC) (US) the support of the MIB OBJECTS dot11MultiDomainCapabilityImplemented, dot11MultiDomainCapabilityEnabled and dot11CountryString is OPTIONAL and if implemented, dot11MultiDomainCapabilityImplemented MUST be 'false'.

6.3.3.1.4.9.1.2 Requirements for dot11WEPDefaultKeysTable

The PS MUST implement the MIB object dot11WEPDefaultKeyValue SYNTAX clause as "OCTET STRING (0|5|13)" which means 40 bits and 104 bit keys are supported.

PS MAY support only one entry for this table in which case dot11WEPDefaultKeyID MAY have read-only access or either MUST be restricted to value 0, to avoid undetected misconfigurations.

6.3.3.1.4.9.1.3 Requirements for dot11OperationEntry

The MIB objects dot11RTSThreshold, and dot11FragmentationThreshold MAY have read-only access.

6.3.3.1.4.9.1.4 Requirements for dot11PhyTxPowerEntry

PS SHOULD implement the values for MIB objects in dot11PhyTxPowerEntry in mW equivalent to a value in the range of percentages of the maximum power output of the device for the configured operational mode in cabhPsDev802dot11PhyOperMode as described in Table 6-19. This configuration simplifies the power transmit value setting of MIB object dot11CurrentTxPowerLevel.

If PS conforms to the configuration values in Table 6-19, it MUST support the eighth instances (1..8) of the MIB object dot11NumberSupportedPowerLevels which permits to the implementation to support 8 or 4 power levels. As an example, Table 6-19 rightmost column indicates the values of MIB objects in dot11PhyTxPowerEntry for the UNII lower band defined in Table 89 of [802.11A-1999]. In particular, the odd levels (1, 3, 5, 7) represent deterministic references at 100%, 75%, 50% and 25% of maximum device power; while even levels (2, 4, 6, 8) have power values in the corresponding ranges of the corresponding upper and lower odd numbers (1-3, 3-5, 5-7, 7-).

Table 6-19/J.192 – Recommended power levels

Power level	Value in mW Relative % of PS maximum power level	Value in mW % Upper limit	Value in mW % Lower limit	Band 5.15-5.25 GHz 40 (mW)
Dot11TxPowerLevel1	100%	100%	100%	40
Dot11TxPowerLevel2	100%-75%	100%	75%	40-30
Dot11TxPowerLevel3	75%	75%	75%	30
Dot11TxPowerLevel4	75%-50%	75%	50%	30-20
Dot11TxPowerLevel5	50%	50%	50%	20
Dot11TxPowerLevel6	50%-25%	50%	25%	20-10
Dot11TxPowerLevel7	25%	25%	25%	10
Dot11TxPowerLevel8	25%-12%	25%	12%	10-5

6.3.3.1.4.9.1.5 dot11PhyDSSSEntry

PS MUST support the MIB object dot11CurrentChannel for implementations compliant with 802.11B/Cor1-2001 or 802.11G-2003 PHY modes.

6.3.3.1.4.9.1.6 Requirements for dot11PhyOFDMEntry

PS MUST implement dot11PhyOFDMEntry if operating in a compliant 802.11A PHY mode.

6.3.3.1.4.9.2 IPCable2Home configuration extensions for 802.11 MIB Requirements

If the PS implements IEEE 802.11 wireless LAN functionality, the PS MUST implement the MIB objects under OBJECT IDENTIFIER cabhPsDevPs802dot11.

6.3.3.1.4.9.2.1 Requirements for cabhPsDev802dot11BaseEntry

The MIB object cabhPsDev802dot11BaseAdvertiseSSID MAY be implemented as read only.

6.3.3.1.4.10 ipNetToMediaTable Requirements

The ipNetToMediaTable [RFC 2011] maps IP addresses to physical addresses, and its use is straightforward if each IP address is associated to one physical interface and if each physical interface is associated to one physical address. The PS, however, implements different IP addresses that may apply to several physical interfaces, and associates the physical WAN interface to two hardware addresses. The PS also implements different Primary Packet-handling Modes, which also has an affect on the ipNetToMediaTable. The PS MUST list in the ipNetToMediaTable each of the IP addresses that are part of its active configuration, creating one entry per distinct IP value and abiding by Table 6-20 for NAPT and NAT Primary Packet-handling Modes (including Mixed Mode), and abiding by Table 6-21 for Passthrough Primary Packet-handling Mode.

Table 6-20/J.192 – PS static entries in the ipNetToMediaTable for NAPT, NAT, & mixed modes

ipNetToMediaAddress	ipNetToMediaPhys Address	ipNetTo MediaIfIndex	ipNetTo MediaType
WAN-Man IP Address	WAN-Man hardware address	1	static(4)
1st WAN-Data IP Address	WAN-Data hardware address	2	static(4)
2nd WAN-Data IP Address	WAN-Data hardware address	2	static(4)
Nth WAN-Data IP Address	WAN-Data hardware address	2	static(4)
CDP Server Router IP address	Zero length octet string	255	static(4)
Well Known PS LAN IP Address (if different from ServerRouter IP)	Zero length octet string	255	static(4)

Table 6-21/J.192 – PS static entries in the ipNetToMediaTable for passthrough mode

ipNetToMediaAddress	ipNetToMediaPhys Address	ipNetToMediaIf Index	ipNetToMediaType
WAN-Man IP Address	WAN-Man hardware address	1	static(4)
Well Known PS LAN IP Address	Zero length octet string	255	static(4)

The PS element MUST dynamically learn the IP and hardware addresses of OSI Layer 3 devices of each of its active physical LAN interfaces and each of its active WAN interfaces. IP and hardware addresses learned by the PS element, along with the appropriate PS ifIndex numbers and ipNetToMediaType information, MUST be accessible to the NMS system (through the CMP) via the [RFC 2011] ipNetToMediaTable. All dynamically learned entries in the ipNetToMediaTable MUST have an ipNetToMediaType value of dynamic(3).

A row entry for the PS's CM MUST NOT appear in the PS's ipNetToMediaTable since the CM acts as a transparent bridge from the perspective of the PS.

As a result of completing the PS provisioning process, the PS MUST create a row entry in its ipNetToMediaTable representing the next hop router for the WAN-Man interface, with an ifIndex value of 1, ipNetToMediaPhysAddress & ipNetToMediaNetAddress values specific to that router, and an ipNetToMediaType value of dynamic(3). If the PS has an active WAN-Data interface, the PS MUST create a row entry in its ipNetToMediaTable representing the next hop router for the WAN-Data interface, with an ifIndex value of 2, ipNetToMediaPhysAddress & ipNetToMediaNetAddress values specific to that router, and an ipNetToMediaType value of dynamic(3).

The PS element MUST delete entries from its ipNetToMediaTable that have an ipNetToMediaType value of dynamic(3) when an implementation-specific inactivity timeout expires.

6.3.3.1.5 PS user interface control

The CMP supports configuration of a User Interface (UI), if one is implemented, through a set of objects defined in the PSDev MIB (see E.4). These objects allow the cable operator to select which UI is presented to the user when the user points its web browser to the PS Server Router IP address. The MIB supports selection between a local manufacturer UI, a local cable operator UI, a network server-based UI, and allows the UI to be disabled. The MIB also enables the cable operator to configure a UI login and password for the subscriber. Refer to the IPCable2Home User Interface Miscellaneous MIB in Annex E.

6.3.3.2 CMP event reporting function

The CMP is required to support the handling and reporting of events generated by the PS, for the WAN Domain. Event messages defined by IPCable2Home for the PS element can be reported via SNMP Trap to the cable operator's notification receiver, via a System Log message sent to the cable operator's system log, or via a log local to the PS and accessible through specified MIB objects. Events defined for the PS are listed in Annex B, Format and Content for Event, SYSLOG, and SNMP Trap. These are the same processes defined in DOCSIS specifications for event reporting in cable modems.

IPCable2Home Host devices are not required to support event messaging. Therefore, LAN Domain event messaging is not defined by this Recommendation.

Event reporting for the WAN domain

IPCable2Home uses the [RFC 2669] event reporting and control mechanisms for events generated in the PS (CMP). [RFC 2669] defines a standard format for reporting event information, regardless of the message type, including a local event log table in which certain entries will persist across reboot of the PS. Note that events may be generated by any part of a PS, but the CMP logs and/or reports the event either locally or to a Syslog or Trap server.

6.3.3.2.1 Event reporting function goals

The goals of the CMP Event Reporting function are listed below:

- enable the transfer of unsolicited messages from the PS to the NMS across the WAN in the form of SNMP Traps and SYSLOG messages;
- enable the logging of status and exception information in the PS Database (local log);
- enable access to local log status and exception information via MIB objects;
- maintain compatibility with event reporting as defined in DOCSIS specifications.

6.3.3.2.2 Event reporting function system design guidelines

The system design guidelines listed in Table 6-22 have guided specification of the CMP Event Reporting Function.

Table 6-22/J.192 – CMP event reporting function system design guidelines

Reference	Guidelines
EvRep 1	The PS will support the reporting of status and exception information as SNMP Notifications, SYSLOG messages, and volatile and non-volatile local log messages.
EvRep 2	The PS will support configurable event throttles and limits.
EvRep 3	The PS will support configurable event priorities.

6.3.3.2.3 Event reporting function system description

Event reporting is a means for an element to report on status or an error condition in an unsolicited message. IPCable2Home supports four types of event reporting:

1) SNMP notify or trap;
2) SYSLOG messaging;
3) Non-volatile local log;
4) Volatile local log.

The use of the DOCSIS Device MIB [RFC 2669] to configure the PS for where to send SNMP traps (notifications) and SYSLOG messages and for event inhibiting and throttling values is required. Event notification by the PS is fully configurable. This Recommendation defines where the PS is to report events assigned a particular priority (see Table 6-23) and the DOCSIS Device MIB allows the priority of each event to be configured. The DOCSIS Device MIB also maintains statistics for the occurrence of each event. The Event Table (docsDevEventTable) in the DOCSIS Device MIB includes an entry for each unique event reported by the PS, a count for the number of occurrences for each unique event entry, and the time at which the last entry was made for each event entry.

IPCable2Home defines the procedure for re-indexing the Event Table in the event that the PS is re-initialized such that volatile local log entries are lost. When volatile local log entries are lost, the PS is required to re-index the Event Table such that the remaining (volatile) local log entries are sequentially indexed.

6.3.3.2.4 Event reporting function requirements

PS requirements for CMP Event Reporting Function are specified in 6.3.3.2.4.1-6.3.3.2.4.9.

6.3.3.2.4.1 Event notification

The PS MUST generate asynchronous events that indicate important events and situations as specified (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 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.

The PS MUST support the following event notification mechanisms:

- local event logging where certain entries in the local log can be identified to persist across a reboot of the PS;
- SNMP Trap and Inform;
- SYSLOG.

The PS MUST implement the docsDevEvControlTable from [RFC 2669] to control reporting of events. The following bit values for the [RFC 2669] object docsDevEvReporting MUST be supported by the PS:

- local-nonvolatile(0);
- traps(1);
- syslog(2);
- local-volatile(3).

SNMP SET request messages to the [RFC 2669] object docsDevEvReporting using the following values MUST result in a 'Wrong Value' error for SNMP PDUs:

- 0x20 = SYSLOG only;
- 0x40 = trap only;
- 0x60 = (trap + SYSLOG) only.

An event reported by Trap, SYSLOG, or Inform MUST also generate a local log entry, whether volatile or non-volatile according to Table 6-23, and as described in 6.3.3.2.4.2.

6.3.3.2.4.2 Local event logging

The PS MUST maintain a single local-log event table that contains events stored as both local-volatile events and local-nonvolatile events. Events stored as local-nonvolatile events MUST persist across reboots of the PS. The local-log event-table MUST be organized as a cyclic buffer with a

minimum of ten entries. The single local-log event-table MUST be accessible through the docsDevEventTable as defined in [RFC 2669].

Event descriptions MUST NOT be longer than 255 bytes, which is the maximum defined for SnmpAdminString.

The EventId is a 32-bit unsigned integer. EventIds ranging from 0 to ((2^31) – 1) are reserved. The EventId MUST be converted from the error codes defined in Annex B. The EventIds ranging from 2^31 to ((2^32) – 1) MUST be used as vendor-specific EventIds using the following format:

- Bit 31 set to indicate vendor-specific event;
- Bits 30-16 contain bottom 15 bits of vendor's SNMP enterprise number;
- Bits 15-0 used by vendor to number their events.

The [RFC 2669] object docsDevEvIndex provides for relative ordering of events in the log. The tagging of local log events as local-volatile and local-nonvolatile necessitates a method for synchronizing docsDevEvIndex values between the two types of events after a PS reboot. After a PS reboot, to synchronize the docsDevEvIndex values for volatile and non-volatile events, the following procedures MUST be used:

- The values of docsDevEvIndex for local log events tagged as local-nonvolatile MUST be renumbered beginning with 1.
- The local log MUST then be initialized with the events tagged as local-nonvolatile in the same order as they had been immediately prior to the reboot.
- Subsequent events recorded in the local log, whether tagged as local-volatile or local-nonvolatile, MUST use incrementing values of docsDevEvIndex.

A reset of the local log initiated through an SNMP SET of RFC 2669 object docsDevEvControl MUST clear all events from the local log, including log events tagged as both local-volatile and local-nonvolatile.

6.3.3.2.4.3 SNMP Trap and Inform

The PS MUST support the SNMP Trap PDU as described in RFC 3411. The PS MUST support the SNMP INFORM PDU as described in RFC 3411. Inform is a variation of Trap and requires the receiving host to acknowledge the arrival of an InformRequest-PDU with an InformResponse-PDU.

When a standard SNMP Trap is enabled in the PS, it MUST send notifications for any event in that category whose priority is either "error" or "notice".

The PS MAY support vendor-specific events. If supported, vendor-specific PS events reportable via SNMP Trap MUST be described in a private MIB that is distributed with the PS. When defining a vendor-specific SNMP Trap, the OBJECTS statement of the private Trap definition SHOULD contain at least the objects explained below:

- EvLevel;
- EvIdText;
- Event Threshold (if any for the Trap);
- IfPhysAddress (the physical address associated with the WAN-Man IP address of the PS).

More objects can be contained in the OBJECTS statement as desired.

6.3.3.2.4.4 SYSLOG

SYSLOG messages issued by the PS MUST be in the following format:

<level>PortalServicesElement[vendor]: <eventId> text

Where:

Level – ASCII presentation of the event priority, enclosed in angle brackets, which is constructed as the bitwise of the default Facility (128) and event priority (0-7). The resulted level has the range between 128 and 135.

vendor – Vendor name for the vendor-specific SYSLOG messages or "CABLEHOME" for the standard IPCable2Home messages.

EventId – ASCII presentation of the INTEGER number in decimal format, enclosed in angle brackets, that uniquely identifies the type of event. This EventID MUST be the same number that is stored in docsDevEvId object in docsDevEventTable. For the standard IPCable2Home events, this number is converted from the error code using the following rules:

- The number is an eight-digit decimal number.
- The first two digits (left most) are the ASCII code (decimal) for the letter in the Error code.
- The next four digits are filled by 2 or 3 digits between the letter and the dot in the Error code with zero filling in the zap in the left side.
- The last two digits are filled by the number after the dot in the Error code with zero filling in the zap in the left.

For example, event D04.2 is converted into 68000402, and Event I114.1 is converted into 73011401.

Please note that this notion only uses a small portion of available number space reserved for IPCable2Home (0 to 2^31 – 1). The first letter of an error code is always in upper case.

text – For the standard messages, this string MUST have the textual description as defined in Annex B.

The example of the syslog event for the event D04.2: "Time of the day received in invalid format":

<132>Portal ServicesElement[CABLEHOME]: <68000402> Time of the day received in invalid format.

The number 68000402 in the given example is the number assigned to this particular event.

6.3.3.2.4.5 Format of events

The IPCable2Home Management Event messages MAY contain any of the following information:

- Event Counter – Indicator of event sequence.
- Event Time – Time of occurrence.
- Event Priority – Severity of condition. [RFC 2669] defines eight levels of severity. The default event severity can be changed to a different value for each given event via the SNMP interface.
- Event Enterprise Number – This number identifies the event as either a standard event or a vendor-defined event.
- Event ID – Identifies the exact event when combined with the Event Enterprise Number. Vendors define their own Event IDs. IPCable2Home standard management events are defined in Annex B. Each management event described in the annex is assigned an Event ID.
- Event Text – Describes the event in human readable form.
- 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.

The exact format of this information for Traps and Informs is defined in Annex B. The format for SYSLOG messages is defined in the requirements portion of this subclause.

6.3.3.2.4.6 Event priorities

RFC 2669 defines 8 different priority levels and the corresponding reporting mechanism for each level. The standard events specified in this Recommendation utilize these priority levels.

– **Emergency event (priority 1)**

  Reserved for vendor-specific 'fatal' hardware or software errors that prevent normal system operation and cause the reporting system to reboot. Each vendor may define its own set of emergency events. Examples of such events could be 'no memory buffers available', 'memory test failure', etc.

– **Alert event (priority 2)**

  A serious failure which causes the reporting system to reboot but the reboot is not caused by either hardware or software malfunctioning. After recovering from the event, the system MUST send the cold/warm start notification.

– **Critical event (priority 3)**

  A serious failure that prevents the device from transmitting data but could be recovered without rebooting the system. After recovering from a Critical event, the PS MUST send the Link Up notification. Examples of such events could be PS Configuration File problems or the inability to get an IP address through DHCP.

– **Error event (priority 4)**

  A failure that could interrupt the normal data flow but does not cause device to reboot. Error events can be reported in real time by using either the Trap or SYSLOG mechanism.

– **Warning event (priority 5)**

  A failure that could interrupt the normal data flow. SYSLOG and Trap reporting are enabled by default for this level.

– **Notice event (priority 6)**

  An event of importance that is not a failure and could be reported in real time by using either the Trap or SYSLOG mechanism. Examples of the NOTICE events are 'Cold Start', 'Warm Start', 'Link Up' and 'SW upgrade successful'.

– **Informational event (priority 7)**

  An event of importance that is not a failure, but which could be helpful for tracing the normal operation of the device.

– **Debug event (priority 8)**

  Reserved for vendor-specific non-critical events.

The priority associated with standard events MUST NOT be changed.

Table 6-23 shows the default notification types for the various event priorities. The PS MUST implement the default notification types as defined in Table 6-23, Default Notification Types for PS Event Priorities, for the eight event priorities. For example, the default notification type for Emergency and Alert events is to place them in the local-log as non-volatile entries.

Table 6-23/J.192 – Default notification types for PS event priorities

Event priority	Local-non-volatile (bit 0)	SNMP Trap (bit 1)	SYSLOG (bit 2)	Local-volatile (bit 3)	Note
1 Emergency	Yes	No	No	No	Vendor Specific
2 Alert	Yes	No	No	No	This Recommendation
3 Critical	Yes	No	No	No	This Recommendation
4 Error	Yes	Yes	Yes	No	This Recommendation
5 Warning	Yes	Yes	Yes	No	This Recommendation
6 Notice	No	Yes	Yes	Yes	This Recommendation
7 Informational	No	No	No	No	This Recommendation and Vendor Specific
8 Debug	No	No	No	No	Vendor Specific

The PS MUST support the ability to be configured to generate all notification types for each event priority level listed in Table 6-23.

6.3.3.2.4.7 Standard events

The PS MUST send the following generic SNMP Traps, as defined in [RFC 3418] and [RFC 2863]:

- coldStart [RFC 3418];
- linkUp [RFC 2863];
- linkDown [RFC 2863];
- SNMP authentication-Failure [RFC 3418].

The PS MUST be capable of generating event notifications based on standard events listed in Annex B.

6.3.3.2.4.8 Event throttling and limiting

The PS MUST support SNMP Trap/Inform and SYSLOG throttling and limiting as described in [RFC 2669].

The PS MUST consider events identical if their EventIds are identical.

[RFC 2669] specifies four throttling states:

- unconstrained(1) causes Traps and SYSLOG messages to be transmitted without regard to the threshold settings.
- maintainBelowThreshold(2) causes Trap transmission and SYSLOG messages to be suppressed if the number of Traps would otherwise exceed the threshold.
- stopAtThreshold(3) causes Trap transmission to cease at the threshold, and not resume until directed to do so.
- inhibited(4) causes all Trap transmission and SYSLOG messages to be suppressed.

A single event MUST be treated as a single event for threshold counting, that is, an event causing both a Trap and a SYSLOG message is still treated as a single event.

6.3.3.2.4.9 Secure software download event reporting

Table B.1 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 a PS with an embedded cable modem is controlled and managed by the DOCSIS cable modem. Clause 11.8, Secure Software Download for the PS, defines requirements for secure software download for the two classes of Portal Services elements. The embedded PS, as defined in 5.1.2.1, 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.

{informative text:

6.3.3.3 CMP discovery function

6.3.3.3.1 Discovery function goals

The goals for the CMP Discovery function are listed below:

- Provide cable operators with visibility to UPnP Host devices and IPCable2Home Residential Gateway device attributes.
- Provide cable operators with visibility to UPnP Services on UPnP Host devices.

Assumptions

The assumptions for the CMP discovery capability include the following:

- IPCable2Home Host devices, UPnP Host devices, and IPCable2Home Residential Gateway devices implement the Internet Protocol (IPv4) suite of protocols.
- UPnP Host devices implement a UPnP Device for discovery, description, and control as specified in UPnP Device Architecture.
- UPnP Host devices optionally implement UPnP QoS Services.

6.3.3.3.1.1 Discovery function system design guidelines

The system design guidelines listed in Table 6-24 have provided guidance in the development of the CMP Discovery function specification.

Table 6-24/J.192 – PS discovery system design guidelines

Reference	Guidelines
Discovery 1	PS will implement UPnP Device Discovery functionality consistent with UPnP Device Architecture 1.0.
Discovery 2	PS will provide the cable operator upon request information about UPnP devices on the home LAN.
Discovery 3	PS will implement UPnP Control Point functionality consistent with UPnP Device Architecture 1.0 for the discovery, description, and control of UPnP devices and services.
Discovery 4	PS will implement specified UPnP device & service hierarchy.
Discovery 5	UPnP discovery protocol messaging will not propagate onto the WAN.

6.3.3.3.2 Discovery function system description

The purpose of the CMP Discovery Function is to provide the cable operator with information about the UPnP Host devices and UPnP Services available on a subscriber's LAN.

The Discovery function of the PS provides a central repository of information about UPnP devices and UPnP services available on the subscriber's LAN. The PS implements a UPnP control point (PS CP) that allows it to discover all the UPnP device and service instances on the home network. In addition, the Discovery function may request additional details about specific UPnP devices and services in the form of UPnP Device and Service Description Documents.

6.3.3.3.3 Discovery function requirements

1) When cabhPsDevUpnpCommand MIB is set to discoveryInfo and cabhPsDevUpnpCommandUpdate is set to true, the PS MUST invoke UPnP discovery via M-Search with the search target (ST) of upnp:rootdevice and the maximum wait (MX) value of less than or equal to 3 seconds as specified in the UPnP Device Architecture (UDA1.0).
2) The PS MUST populate the PS Database with the device description information that is accessible via cabhPsDevUpnpInfoTable MIB Table. While updating the PS database with device discovery information received in response to M-Search, the PS MUST filter the information based on cabhPsDevUpnpCommandIp MIB.
   - If cabhPsDevUpnpCommandIp is set to 255.255.255.255, the PS MUST populate the PS database with device discovery information from all the root UPnP devices on the home network, and MUST also include its own UPnP device discovery information.
   - If cabhPsDevUpnpCommandIp is set to 192.168.0.1, the PS MUST populate the PS database with its own UPnP Device discovery information.
3) All URIs exposed by the PS for UPnP communications MUST use IP addresses and MUST NOT use host names.
4) If PS wants to change a device description or a service description file, then it MUST first leave the UPnP network by sending an ssdp:byebye message and then re-join the UPnP network with the new XML files using an ssdp:alive message.
5) PS MUST send its all discovery advertisements with the HTTP LOCATION header of 192.168.0.1.
6) The deviceType of the root device in the PS MUST be urn:schemas-cablelabs-com:device:CableHomePSDevice:1.
7) PS MUST advertise IGD Device 1.0 as an embedded device of the CableHome PS root Device.
8) PS MUST advertise all UPnP QoS Services as services of the CableHome PS root Device.
9) The PS MUST advertise the following hierarchy as part of its UPnP device description document:
   - CableHomePSDevice
     - IGD Device 1.0
       - IGD WAN Device 1.0
         - IGD WANConnection Device 1.0
           - IGD WANIPConnection Service 1.0
     - QoS Manager Service 1.0
     - QoS Policy Holder Service 1.0
     - QoS Device Service 1.0 (Optional)

(See Appendix I for an example of this hierarchy.)

10) In order to provide unique naming for all the UPnP Devices in the CableHome PS Root Device Description, it is suggested to use the following format when constructing the unique device names for all UPnP devices in the CableHome PS.

- CableHomePSDevice unique device name SHOULD be formatted as CableHomePSDevice-1_0-00aabbccddee, where 00aabbccddee corresponds to the PS WAN-MAN MAC address.
- InternetGatewayDevice unique device name SHOULD be formatted as InternetGatewayDevice-1_0-00aabbccddee, where 00aabbccddee corresponds to the PS WAN-MAN MAC address.
- WANConnectionDevice unique device name SHOULD be formatted as WANConnectionDevice-1_0-00aabbccddee, where 00aabbccddee corresponds to the PS WAN-MAN MAC address.

11) When a UPnP Service implemented by the PS is disabled, the PS MUST behave as follows:

- The PS MUST multicast SSDP:byebye message for that service.
- The PS MUST NOT advertise the disabled service in the future SSDP advertisements.
- The PS MUST NOT return the disabled service in the root device description.
- The PS MUST NOT respond to a M-SEARCH for the disabled service.
- The PS MUST NOT respond to an action for the disabled service.

}

6.4 PS logical element IPCable2Home Test Portal (CTP)

6.4.1 CTP goals

The goals for the IPCable2Home Test Portal include:

- Enable LAN IP Device and IPCable2Home Host fault diagnostics.
- Enable visibility to LAN IP Devices and IPCable2Home Hosts, as well as access to the number and types of LAN IP Devices and IPCable2Home Host.
- Enable LAN IP Device and IPCable2Home Host performance monitoring.

6.4.2 CTP design guidelines

The Test Portal system design guidelines are listed in Table 6-25. A number of these guidelines are common with the CMP design guidelines. This list has provided guidance for the specification of CTP functionality.

Table 6-25/J.192 – CTP system design guidelines

Reference	Guidelines
CTP 1	The need exists for interfaces to support the management and diagnosis features and functions required to support cable-based services provisioned across the home network.
CTP 2	Local and remote monitoring capabilities are needed that can monitor home network operation and help the consumer and cable operator identify problem areas.
CTP 3	The cable network NMS requires a method to gather identification information about each IP device connected to the home network.
CTP 4	The cable network NMS requires a method to detect whether a connected device is in an operable state.

6.4.3 CTP system description

The CTP (IPCable2Home Test Portal) contains the "remote tools" with which the NMS can collect further LAN device information. Tests must be run remotely, since getting past a network address translation (NAT) function in a router can be a challenge. For example, a WAN-to-LAN ping will not pass through a PS, unless the CAP has been preconfigured to pass this traffic. The CTP is a local proxy used to interpret and execute the remote fault/diagnostic class of SNMP messages it receives from the NMS operator. These LAN IP Device and IPCable2Home Host tests are defined based on problems likely to be encountered for IPCable2Home 1.1 type of home networks: connectivity and throughput diagnostics.

These functions are termed the CTP Connection Speed Tool and CTP Remote Ping Tool. The Connection Speed and Remote Ping Tools enable the cable operator's customer support centre and network operations centre to learn more about the connection between the PS element and LAN IP Devices and IPCable2Home Hosts in the home.

6.4.3.1 CTP connection speed tool function

6.4.3.1.1 Connection speed tool function goals

The goal of the Connection Speed Tool function is to enable the IPCable2Home system manager to remotely acquire metrics about the performance of the home LAN between the PS and a specific LAN IP Device or IPCable2Home Host.

6.4.3.1.2 Connection speed tool system design guidelines

Design guidelines listed in Table 6-25, CTP System Design Guidelines, were used to guide specification of the Connection Speed Tool function.

6.4.3.1.3 Connection speed tool function system description

The Connection Speed Tool function is used to get a rough measure of the throughput performance across the link between the PS and a LAN IP Device or IPCable2Home Host. It sends a burst of packets between the PS and the LAN IP Device or IPCable2Home Host under test, and the round trip 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 [see E.3].

The Connection Speed function relies on the LAN IP Devices and IPCable2Home Hosts 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 (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 and IPCable2Home Hosts directly from the NMS console if desired. This test feature works on LAN IP Devices and IPCable2Home Hosts in either the LAN-Trans or LAN-Pass address realms that implement the Echo Service function as described in [RFC 347].

The CTP Testable Requirements clause below lists the parameters and responses for the Connection Speed Tool. Clause 12.2.1.1 details the operation of the Connection Speed Tool.

6.4.3.1.4 Connection speed tool function requirements

The PS MUST implement the Connection Speed Tool, and MUST comply with the default values and value ranges defined for the Connection Speed Tool-specific objects of the CTP MIB [see E.3].

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

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

The PS MUST NOT generate packets out any WAN Interface when running the Connection Speed Tool function.

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

- reset the timer;
- set cabhCtpConnStatus = running(2);
- 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;
- 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 Connection Speed Tool test times out before the last bit is received from the target LAN IP Device or IPCable2Home Host, 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 and IPCable2Home Host or before the Connection Speed Tool test times out, the PS MUST set cabhCtpConnStatus = aborted(4) and report the appropriate event (refer to Annex B – CTP Events).

When the Connection Speed Tool function is executing, the PS MUST determine the average round-trip throughput between the PS and the LAN IP Device or IPCable2Home Host whose address is passed in cabhCtpConnDestIp (the target LAN IP Device) in kbit/s, round the number to the nearest whole integer, and store the result in cabhCtpConnThroughput.

The PS 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 PS 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 PS 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 PS 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 PS 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 PS 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 PS 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 PS 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).

6.4.3.2 CTP ping tool function

6.4.3.2.1 Ping tool function goals

The goal of the Ping Tool function is to enable the system manager to remotely test or verify connectivity between the PS and a specific LAN IP Device.

6.4.3.2.2 Ping tool function system design guidelines

Design guidelines listed in Table 6-25, CTP System Design Guidelines, were used to guide specification of the Ping Tool function.

6.4.3.2.3 Ping tool function system description

The Ping Tool function is called to test connectivity between the PS and individual LAN IP Devices or IPCable2Home Host devices. Results of multiple executions of the Ping Tool test can be assembled by the NMS to create a network scan of the LAN IP Devices or IPCable2Home Host devices. The DHCP table of the CDP has a list of historical devices, but only the devices that employ DHCP. Ping may capture a current state including non-DHCP clients. To keep the PS simple, it is expected that the NMS increments the address and stores the results in the NMS tool to perform a scan of a LAN subnet.

The PING Tool is initiated by a series of SNMP set-request messages issued by the cable network NMS console to the PS management address.

Clause 12.2.1.2 details the operation of the Ping Tool.

6.4.3.2.4 Ping tool function requirements

The CTP Ping Tool MUST be implemented using the Internet Control Message Protocol (ICMP) "Echo" facility. The CTP will issue an ICMP Echo Request and the LAN IP Device is expected to return an ICMP Echo Reply.

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

The PS MUST implement the CTP Ping Tool, and MUST comply with the default values and value ranges defined for the Ping Tool-specific objects of the CTP MIB [see E.3].

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

- 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. A timeout for each ping (ICMP Echo Request/Response pair) is the value of cabhCtpPingTimeOut;
- 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.

If the CTP receives all Ping replies before their individual timeout timer expires, the PS 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 PS MUST set cabhCtpPingStatus = aborted(4) and report the appropriate event (refer to Annex B – CTP Events).

If a timeout timer expires for at least one of the pings, before its reply is received from the target LAN IP Device, the PS MUST set cabhCtpPingStatus = timedOut(5) and report the appropriate event (refer to Annex B – CTP Events).

When the CTP Ping Tool function is initiated, the PS MUST determine the average round-trip time between the PS and the LAN IP Device or IPCable2Home Host 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 Ping Tool function is initiated, the PS 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 PS MUST increment the value of cabhCtpPingNumIcmpError and log the error in cabhCtpPingIcmpError. The last ICMP error that occurs will overwrite the previous one written.

The PS 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 ICMP Echo Request packet transmitted by the CTP Ping Tool, to the time when the last bit of the corresponding ICMP Echo Reply packet is received.

The PS MUST allow the Ping Tool destination IP address (cabhCtpPingDestIp) to be set to any valid IPv4 address of any LAN IP Device or IPCable2Home Host device accessible through any LAN Interface of the PS running the CTP Ping Tool.

The PS MUST NOT generate packets out any WAN Interface when executing the Ping Tool function.

The PS 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 PS 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).

7 Provisioning tools

7.1 Introduction/Overview

The Portal Services element and LAN IP Devices must be properly initialized and configured in order to exchange meaningful information with one another and with elements connected to the cable network and the Internet. IPCable2Home provisioning tools provide the means for this initialization and configuration to occur seamlessly and with minimum user intervention. They also enable cable operators to add value to high-speed data service subscribers by defining processes through which the cable operator can facilitate and customize PS and LAN IP Device initialization and configuration. The three provisioning tools defined to accomplish this task are listed below:

- DHCP Portal (CDP) function in the Portal Services element;
- Bulk Portal Services Configuration (BPSC) tool;
- Time of Day Client in the Portal Services element.

7.1.1 Goals

Goals of the Provisioning Tools are listed below:

- Enable the PS to acquire a network address on its WAN interface to be used for management of the PS.
- Enable the PS to acquire one or more network addresses on its WAN interface to be used for the exchange of traffic between LAN IP Devices and the Internet or between IPCable2Home Host devices and the Internet.
- Enable the PS to request and acquire configuration parameters in a configuration file.
- Enable the PS to acquire current time of day from time of day services in the cable operator's data network.
- Enable the PS to assign network address leases to LAN IP Devices and IPCable2Home Host devices.
- Enable the PS to assign configuration parameters to LAN IP Devices and IPCable2Home Host devices.

7.1.2 Assumptions

The Provisioning Tools operating assumptions are listed below:

- LAN IP Devices and IPCable2Home Host devices implement a DHCP client as defined by [RFC 2131].
- The cable network provisioning system implements a DHCP server as defined by [RFC 2131].
- If the cable network provisioning system's DHCP server supports DHCP option 61 (client identifier option), the WAN-Man and all WAN-Data IP interfaces can share a common MAC address.
- LAN IP Devices and IPCable2Home Host devices may support various DHCP Options and BOOTP Vendor Extensions, allowed by [RFC 2132].

- Bulk PS configuration will be accomplished via the download of a PS Configuration File containing one or more parameters, using Trivial File Transfer Protocol (TFTP) [RFC 1350] or Hypertext Transfer Protocol (HTTP) [RFC 2616] with Transport Layer Security (TLS) [RFC 2246].
- The Headend DHCP server will provide a DHCP option, to the WAN-Management interface, which points to a Time of Day server, operating within the Headend network.

7.2 Provisioning architecture

7.2.1 Provisioning modes

Three provisioning modes are supported. They are referred to as DHCP Provisioning Mode (DHCP Mode), SNMP Provisioning Mode (SNMP Mode), and Dormant CableHome Mode. The three provisioning modes are compared in Table 7-1.

Table 7-1/J.192 – Provisioning modes

	DHCP mode	**SNMP mode**	**Dormant CableHome mode**
DHCP Fields and Option Codes	Receives configuration file information in 'siaddr' and 'file' fields. Receives no option 122	Receives no configuration file information. Receives valid values for option 122 sub-options 3, 6, and 10.	Receives no configuration file information and no option 122, or receives an invalid combination of configuration file information and option 122 sub-options.
PS Configuration File Trigger	Triggered by presence of TFTP server information in DHCP message	Triggered by NMS via SNMP message	PS receives no configuration file
PS Configuration File Requirement	PS Configuration File download is required	PS Configuration File download is not required	PS configuration file is not required

Specified behaviour of the Provisioning Tools is dependent upon the Provisioning Mode in which the PS operates.

Clause 13, Provisioning Processes, describes the sequence of events for DHCP and SNMP Provisioning Modes.

7.2.2 Provisioning architecture description

The provisioning architecture is illustrated in Figure 7-1. Portal Services elements will interact with server functions in the cable network over the HFC interface, or with IPCable2Home Host Devices to satisfy the system design guidelines listed in 7.3.2.

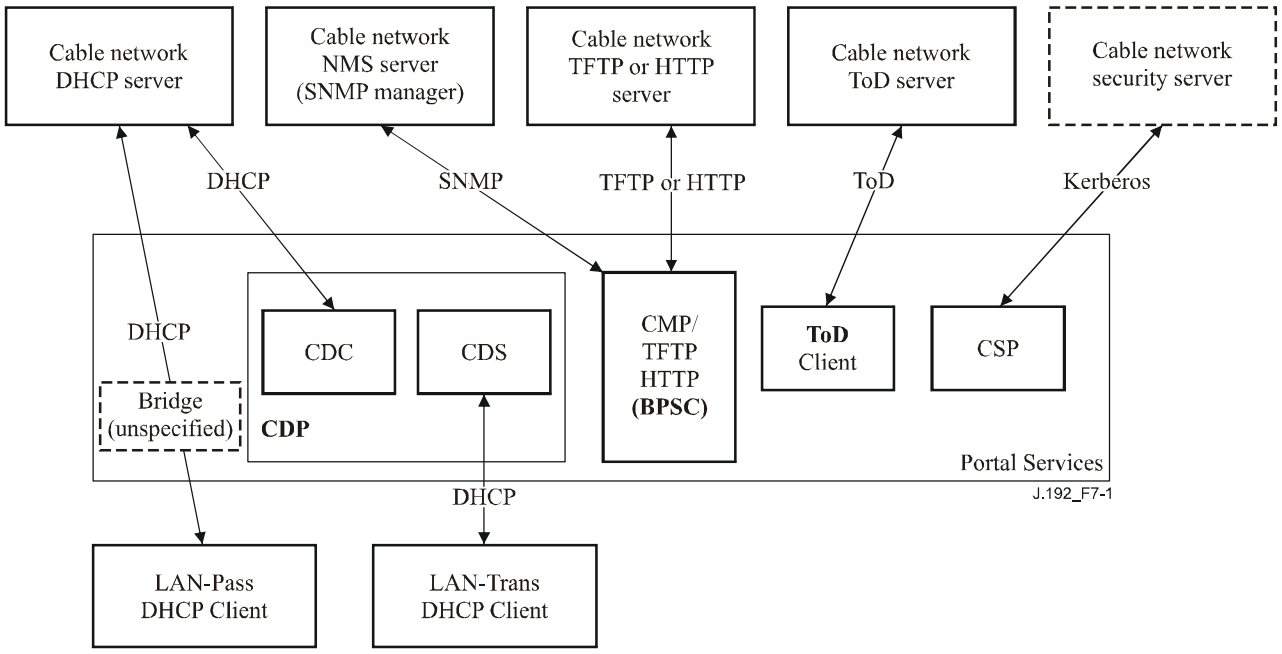


Figure 7-1/J.192 – Provisioning architecture

7.3 PS logical element – DHCP Portal (CDP)

The IPCable2Home DHCP Portal (CDP) is a logical sub-element of the PS logical element. The CDP has two primary roles: acquisition of network address leases for the PS and assignment of network address leases to LAN IP Devices and IPCable2Home Host devices in the LAN, and is one of the three provisioning tools introduced in 7.1. This clause describes the Goals, System Design Guidelines, System Description and Requirements pertaining to the CDP.

7.3.1 CDP goals

The goals of the CDP include the following:

- Enable client functions in the PS to communicate with corresponding server functions in the cable data network.
- Provide the PS with initial configuration parameters, giving it the ability to further configure itself.

7.3.2 CDP system design guidelines

The following design guidelines (Table 7-2) drive the capabilities defined for the CDP:

Table 7-2/J.192 – CDP system design guidelines

Number	Guidelines
CDP 1	Addressing mechanisms will be operator controlled, and will provide operator knowledge of and accessibility to IPCable2Home network elements and LAN IP Devices.
CDP 2	Address acquisition and management processes will not require human intervention (assuming that a user/household account has already been established).
CDP 3	Address acquisition and management will be scalable to support the expected increase in the number of LAN IP devices.
CDP 4	It is preferable for LAN IP Device addresses to remain the same after events such as a power cycle or Internet Service Provider switch.
CDP 5	Provide a mechanism by which the number of LAN IP Devices in the LAN-Trans realm can be monitored and controlled.
CDP 6	In-home communication will continue to work as provisioned during periods of Headend address server outage. Addressing support will be provided for newly added LAN IP Devices and address expirations during remote address server outages.
CDP 7	IP addresses will be conserved when possible (both globally routable addresses and private cable network management addresses).

7.3.3 IPCable2Home DHCP Portal system description

The IPCable2Home DHCP Portal (CDP) is the logical entity that is responsible for addressing activities. The CDP address request and address allocation responsibilities within the IPCable2Home environment include:

- IP address assignment, IP address maintenance, and the delivery of configuration parameters (via DHCP) to LAN IP Devices in the LAN-Trans Address Realm.
- Acquisition of a WAN-Man and zero or more WAN-Data IP addresses and associated DHCP configuration parameters for the Portal Services (PS) element.
- Provide information to the IPCable2Home Name Portal (CNP) in support of LAN IP Device host name services.

The PS 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 deception, the PS does not allow either of the two hardware addresses to be modified.

The Portal Services element requires an IP Address on the home LAN for its role on the LAN as a router (see clause 8, Packet Handling and Address Translation), DHCP Server (CDS), and DNS Server (see clause 9, Name Resolution). The PS listens on a single LAN-side IP address for each of these functionalities. The PS needs to communicate the IP address for each of these server functionalities to the LAN IP Devices in the DHCP OFFER and ACK option fields. In order to uniquely identify these option values, each of these server addresses are identified by different MIB objects in the PS which are listed below and in Table 7-2.

Router (default gateway) Address	cabhCdpServerRouter	Option 3
Domain Name Server (DNS) Address	cabhCdpServerDnsAddress	Option 6
Dynamic Host Configuration Server (DHCP) (CDS) Address	cabhCdpServerDhcpAddress	Option 54

The default value of cabhCdpServerRouter is 192.168.0.1. However, the NMS can set cabhCdpServerRouter to a different value.

The value of cabhCdpServerDhcpAddress is always the same as the value of cabhCdpServerRouter and the NMS cannot change its value directly.

The default value of cabhCdpServerDnsAddress is equal to the value of the cabhCdpServerRouter. However, the NMS can change it to a different value (e.g., DNS server in the cable operator's date network) so that a LAN IP Device can direct its DNS queries to a server other than the PS DNS server.

Thus, the PS always listens on the IP address assigned to cabhCdpSeverRouter for its LAN side router, Name Server (DNS), and DHCP server functionality.

As shown in Figure 7-2, the CDP capabilities are embodied by two functional elements residing within the CDP:

- IPCable2Home DHCP Server (CDS).
- IPCable2Home DHCP Client (CDC).

Figure 7-2 also illustrates interaction between the CDP components and the address realms introduced in clause 5. The CDC exchanges DHCP messages with the DHCP server in the cable network (WAN Management address realm) to acquire an IP address and DHCP options for the PS, for management purposes. The CDC could also exchange DHCP messages with the DHCP server in the cable network (WAN Data address realm) to acquire zero (0), or more IP address(es) on behalf of LAN IP Devices in the LAN-Trans realm. The CDS exchanges DHCP messages with LAN IP Devices in the LAN-Trans realm, and assigns private IP addresses, grants leases to, and could provide DHCP options to DHCP clients within those LAN IP Devices.

LAN IP Devices in the LAN-Pass realm receive their IP addresses, leases, and DHCP options directly from the DHCP server in the cable network. The CDP bridges DHCP messages between the DHCP server in the cable network, and LAN IP Devices in the LAN-Pass realm.

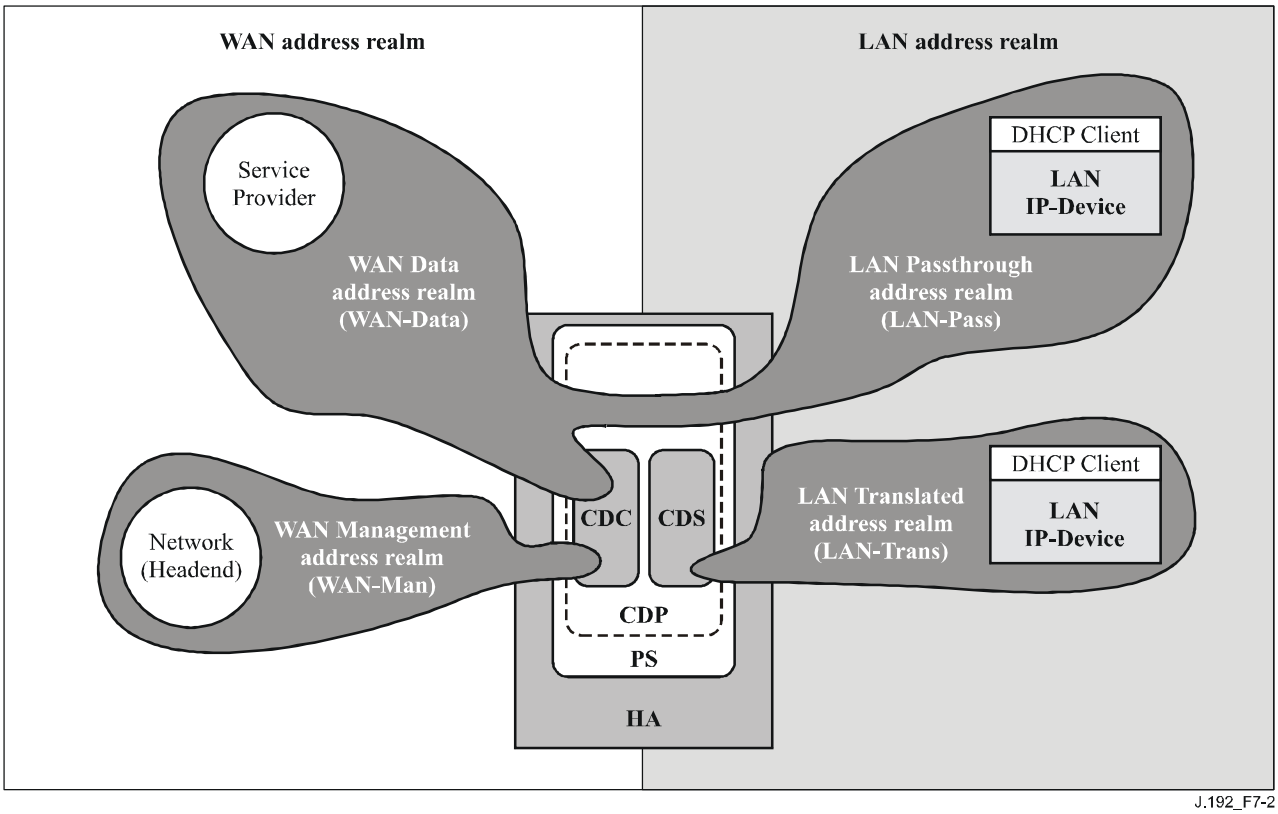


Figure 7-2/J.192 – CDP functions

An embedded PS can be configured to operate in any of four operating modes, as described in 5.5, based on the value of the eDOCSIS [eDOCSIS1] eSAFE MIB object esafePsCableHomeModeControl (ePS only) and on the values of DHCP header fields and options, in the DHCP ACK message received from the cable operator's DHCP server. If the value of esafePsCableHomeModeControl is set to provSystem(2), the embedded PS is required to attempt to acquire a PS WAN-Man IP address lease and act upon the DHCP header values and options. If the value of esafePsCableHomeModeControl is dormantCHMode(3), the embedded PS is required to attempt to acquire an IP address lease to use for its Network Address and Port Translation (NAPT) function (see clause 8, Packet Handling and Address Translation) and to disregard the DHCP header field and option values that would otherwise configure it to operate in a mode other than Dormant CableHome Mode. See 7.3.3.2.4 for a complete description of Dormant CableHome Mode. If esafePsCableHomeModeControl is set to disabled(1), the embedded PS does not attempt to provision, and operates in Disabled Mode, as described in 7.3.3.2.4.

A standalone PS is always required to attempt to acquire a PS WAN-Man IP address lease and is configured to operate in any of three operating modes, based on the values of DHCP header fields and options. A standalone PS cannot be configured to operate in Disabled Mode. See 7.3.3.2.4 for details.

7.3.3.1 DHCP Server (CDS) sub-element

The CDS is a sub-element of the CDP logical element of the PS, and is the function responsible for allocating network address leases to LAN IP Devices in the LAN-Trans realm. It is also responsible for providing LAN IP Devices with configuration information via DHCP Option codes, as specified in [RFC 2132]. The CDS is required to perform this function whether or not the PS has an active WAN connection.

7.3.3.1.1 CDS function goals

Goals for the CDS Function include the following:

- Allocate network address leases to LAN IP Devices in the LAN-Trans realm according to CDP MIB settings and according to [RFC 2131].
- Allocate configuration information according to [RFC 2132].
- Satisfy goals for operation in the absence of a WAN connection by allocating LAN-Trans IP address leases and providing configuration information to LAN IP Devices upon request as long as the PS is operational, whether or not the PS has an active WAN connection.
- Do not allocate IP address leases and do not provide configuration information to LAN IP Devices for which the PS has been configured to treat as existing in the LAN-Pass realm.

7.3.3.1.2 CDS function system design guidelines

The design guidelines listed in Table 7-3 have guided development of the CDS Function specifications.

Table 7-3/J.192 – IPCable2Home DHCP Server (CDS) function system design guidelines

Number	Guidelines
CDS 1	Provide a means by which LAN IP Devices can acquire network address leases and configuration information for the LAN-Trans realm.
CDS 2	The mechanism for allocating LAN-Trans IP addresses and configuration information will operate whether the PS has a WAN connection to the cable operator's data network or not.
CDS 3	The mechanism for allocating LAN-Trans IP address leases and configuration information will not allocate IP address leases or provide configuration information for LAN IP Devices in the LAN-Pass realm.

7.3.3.1.3 CDS function system description

The CDS is a standard DHCP server as defined in [RFC 2132], and responsibilities include:

- The CDS assigns addresses to and delivers DHCP configuration parameters to LAN IP Devices receiving an address in the LAN-Trans address realm. The CDS learns DHCP options from the NMS system and provides these DHCP options to LAN IP Devices. If DHCP options have not been provided by the NMS system (for example when the PS boots during a cable outage), the CDS relies on built-in default values (DefVals) for required options.
- The CDS is able to provide DHCP addressing services to LAN IP Devices, independent of the WAN connectivity state.
- The number of addresses supplied by the CDS to LAN IP Devices is controllable by the NMS system. The behaviour 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 00:00.0 (midnight) GMT, January 1, 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.
- During the PS Boot process, the CDS remains inactive until activated by the PS.
- 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.

LAN IP Devices may receive addresses that reside in the LAN-Pass realm. As shown in Figure 7-2, LAN-Pass address requests are served by the WAN addressing infrastructure, not the PS. LAN-Pass addressing processes will occur when the PS is configured to operate in Passthrough Mode or Mixed Bridging/Routing Mode (see 8.3.4.3, Passthrough Requirements, for more details). In these cases, DHCP interactions will take place directly between LAN IP Devices and cable data network servers, and this Recommendation does not specify the process.

Throughout this Recommendation, the terms **Dynamic Allocation** and **Manual Allocation** are used as defined in [RFC 2132]. The **CDS Provisioned DHCP Options**, cabhCdpServer objects in the CDP MIB, are 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, persist after a PS power cycle and the NMS system can establish, read, write and delete these objects. CDS Provisioned DHCP Options, cabhCdpServer objects, are retained during periods of cable outage and these objects are offered to LAN IP devices assigned a LAN-Trans address during periods of cable outage. The CDS persistent storage of DHCP options is consistent with [RFC 2132], section 2.1. The default values of CDS Provisioned DHCP Options, cabhCdpServer objects, are defined (Table 7-4) and the NMS can reset the CDS Provisioned DHCP Options, cabhCdpServer objects, and cabhCdpLanAddrTable to their default values, by writing to the cabhCdpSetToFactory MIB object.

The CDS Address Threshold (cabhCdpLanTrans) objects contain the event control parameters used by the CDS to signal the CMP to generate a notification to the Headend management system, when the number of LAN-Trans addresses assigned by the CDS exceeds the preset threshold.

The Address Count (cabhCdpLanTransCurCount) object is a value indicating the number of LAN-Trans addresses assigned by the CDS that have active DHCP leases.

The Address Threshold (cabhCdpLanTransThreshold) object is a value indicating when a notification is generated to the Headend management system. The notification is generated when the CDS assigns an address to the LAN IP Device that causes the Address Count (cabhCdpLanTransCurCount) to exceed the Address Threshold (cabhCdpLanTransThreshold).

The Threshold Exceeded Action (cabhCdpLanTransAction) is the action taken by the CDS while the Address Count (cabhCdpLanTransCurCount) exceeds the Address Threshold (cabhCdpLanTransThreshold). If the Threshold Exceeded Action (cabhCdpLanTransAction) allows address assignments after the count is exceeded, the notification is generated each time an address is assigned. The defined actions are:

a) assign a LAN-Trans address as normal; and

b) do not assign an address to the next requesting LAN IP Device.

The Address Count (cabhCdpLanTransCurCount) continues to be updated during periods of cable outage.

The CDS MIB also contains the Address Pool Start (cabhCdpLanPoolStart) and Address Pool End (cabhCdpLanPoolEnd) parameters. These parameters indicate the range of addresses in the LAN-Trans realm that can be assigned by the CDS to LAN IP Devices.

The CDP LAN Address Table (cabhCdpLanAddrTable) contains the list of parameters associated with addresses allocated to LAN IP Devices with LAN-Trans addresses. These parameters include:

- The Client Identifiers, [RFC 2132], section 9.14 (cabhCdpLanAddrClientID).
- The LAN IP address assigned to the client (cabhCdpLanAddrIp).
- An indication that the address was allocated either manually (via the CMP) or dynamically (via the CDP) (cabhCdpLanAddrMethod).

The CDS stores LAN IP Device identifying information in the cabhCdpLanAddrClientID MIB object. The CDS uses the value passed in the chaddr field of the DHCP REQUEST message sent by the LAN IP Device for this purpose.

The CDS creates a CDP Table (cabhCdpLanAddrTable) entry when it allocates an IP address to a LAN IP Device. The CDS can create CDP Table (cabhCdpLanAddrTable) entries during periods of cable outage.

The CDP Table (cabhCdpLanAddrTable) maintains a DHCP lease time for each LAN IP Device.

NMS-provisioned CDP Table (cabhCdpLanAddrTable) entries are retained during periods of cable outage and persist across a PS power-cycle.

7.3.3.1.4 CDS function requirements

The PS MUST comply with the Server requirements of [RFC 2131], section 4.3.

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

PS Manual IP address allocation MUST be supported using CDP MIBs cabhCdpLanAddrTable entries created via the NMS system or PS Configuration file.

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

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

Upon PS reset or reboot, the PS MUST NOT exchange DHCP messages with LAN IP Devices until the CDS is activated by the PS.

The PS MUST activate the CDS, i.e., the PS 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, Part 1):

- 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 Centre (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 PS MUST assign a unique, available IP address from the range of addresses 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.

If the value of cabhCdpLanTransThreshold is 0, the PS 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 PS MUST maintain the Address Count parameter (cabhCdpLanTransCurCount) indicating the number of active LAN-Trans address leases granted to LAN IP devices.

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

The PS 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), the PS MUST generate a notification in accordance with the event reporting mechanism defined in 6.3.3.2, CMP Event Reporting Function, and Annex B. While the Address Count parameter (cabhCdpLanTransCurCount) exceeds the Address Threshold parameter (cabhCdpLanTransThreshold), the PS 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.

If cabhCdpLanTransCurCount equals or exceeds cabhCdpLanTransThreshold and a LAN IP Device requests and additional IP address lease, the PS MUST take specific action as indicated by the Threshold Exceeded Action (cabhCdpLanTransAction) provisioned parameter.

The PS MUST assign IP addresses and deliver DHCP configuration parameters listed in Table 7-4 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 PS 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), the PS MUST notify the event in accordance with Annex B and the event reporting mechanism defined in 6.3.3.2, CMP Event Reporting Function.

The PS MUST store the value passed in the chaddr field of the DHCP REQUEST message sent by the LAN IP Device when an active lease is created for the LAN IP Device.

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

The CDS function of the PS MUST support the DHCP options indicated as mandatory in the CDS Protocol Support column of Table 7-4 CDS DHCP Options.

The CDS MUST include in DHCP OFFER and DHCP ACK messages it sends to its DHCP clients, the DHCP option code 43 sub-option 101 containing the string "CableHome1.1LAN-Trans" (with no spaces and without the quotation marks) as the sub-option information, only in response to DHCP DISCOVER and DHCP REQUEST messages that include DHCP option code 60 containing the string value "CableHome1.1BP" (with no spaces and without the quotation marks).

The CDS MUST NOT include DHCP option code 43 sub-option 101 in the DHCP OFFER and DHCP ACK messages it sends to any DHCP client that did not provide the string value "*CableHome1.1BP*" in DHCP option code 60, in its DHCP DISCOVER and DHCP REQUEST messages.

The CDS function of the PS MUST support offering the default values indicated in the CDS Factory Defaults column of Table 7-4 CDS DHCP Options, if the DHCP option has not been provisioned with other values.

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 function of the PS MUST be disabled.

The CDS function of the PS MUST NOT respond to DHCP messages that are received through any WAN Interface, nor originate DHCP messages from any WAN Interface.

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

The CDS MUST NOT offer a lease for IP address 192.168.0.1, i.e., the CDS MUST NOT transmit a DHCP offer or DHCP Ack message with the value 192.168.0.1 in the yiaddr field.

A distinction is made when the PS is in the provisioning state cabhPsDevProvState equal to inProgress(2). In this state, when the PS provides a DHCP lease to the LAN CPE client(s), it MUST set option 51, IP Address Lease Time, equal to 60 in place of 3600 as specified in Table 7-4.

Table 7-4/J.192 – CDS DHCP options

Option number	Option function	CDS protocol support (M)andatory or (O)ptional	CDS factory defaults	MIB object name
0	Pad	M	N/A	N/A
255	End	M	N/A	N/A
1	Subnet Mask	M	255.255.255.0	cabhCdpServerSubnetMask
2	Time Offset	M	0	cabhCdpServerTimeOffset
3	Router Option	M	192.168.0.1	cabhCdpServerRouter
6	Domain Name Server	M	192.168.0.1	cabhCdpServerDnsAddress
7	Log Server	M	0.0.0.0	cabhCdpServerSyslogAddress
12	Host Name	M	N/A	N/A
15	Domain Name	M	Null String	cabhCdpServerDomainName
23	Default Time-to-live	M	64	cabhCdpServerTTL
26	Interface MTU	M	N/A	cabhCdpServerInterfaceMTU
43	Vendor-specific information	M	Vendor Selected	cabhCdpServerVendorSpecific
43.101	Vendor-specific information sub-option 101	M (Note)	String: "CableHome 1.1 LAN-Trans" (with no spaces)	N/A
50	Requested IP Address	M	N/A	N/A
51	IP Address Lease Time	M	3600 seconds	cabhCdpServerLeaseTime
54	Server identifier	M	192.168.0.1	cabhCdpServerDhcpAddress
55	Parameter Request List	M	N/A	N/A
60	Vendor-class identifier	M	N/A	N/A
61	Client-identifier	O	N/A	N/A

Table 7-4/J.192 – CDS DHCP options

NOTE – The CDS is required to include DHCP option code 43 sub-option 101 containing the string CableHome1.1LAN-Trans, with no spaces, in the DHCP OFFER and DHCP ACK messages it sends to IPCable2Home compliant LAN IP Devices only. IPCable2Home compliance of LAN IP Devices is indicated by the presence of the string CableHome1.1BP in the DHCP DISCOVER and DHCP REQUEST messages.

7.3.3.2 CDP DHCP Client (CDC) function

7.3.3.2.1 CDC function goals

The goals of the CDP CDC Function include the following:

- acquire an IP address lease for the PS IP stack, used for management messaging and file transfer between the cable operator's network servers and the PS;
- acquire configuration information from the cable operator's network DHCP server;
- determine the Provisioning Mode in which the PS is to operate;
- acquire one or more IP address lease(s) for mapping to LAN IP Devices in the LAN-Trans realm.

7.3.3.2.2 CDC function system design guidelines

The guidelines listed in Table 7-5 were used to guide specification of the CDC function:

Table 7-5/J.192 – IPCable2Home DHCP Client (CDC) function system design guidelines

Number	Guidelines
CDC 1	Provide a means by which the PS can acquire a network address lease and configuration information for its WAN-Man interface.
CDC 2	Provide a means by which the PS can acquire one or more network address leases and configuration information for its WAN-Data interface.
CDC 3	The mechanism for allocating LAN-Trans IP address leases and configuration information will not allocate IP address leases or provide configuration information for LAN IP Devices in the LAN-Pass realm.

7.3.3.2.3 CDC function system description

The CDC is a standard DHCP client as defined in [RFC 2131] and responsibilities include:

- The CDC makes requests to Headend DHCP servers for the acquisition of addresses in the WAN-Man and may make requests to Headend DHCP servers for the acquisition of addresses in the WAN-Data address realms. The CDC also understands and acts upon a number of DHCP configuration parameters.
- The CDC makes a determination about which Provisioning Mode the PS is to operate in, based on information received in the DHCP ACKNOWLEDGE message from its DHCP server.
- The CDC supports acquisition of one WAN-Man IP address and zero or more WAN-Data IP addresses.
- The CDC supports the Vendor Class Identifier Option (DHCP option 60), the Vendor Specific Information option (DHCP option 43), and the Client Identifier Option (DHCP option 61).

- In the default case, the CDC will acquire a single IP address for simultaneous use by the WAN-Man and WAN-Data IP interfaces. In order to minimize changes needed to existing Headend DHCP servers, the use of a Client Identifier (DHCP option 61) by the CDC is not required in this default case.
- The CDC in a standalone PS issues a DHCP lease renewal request when the PS detects that the WAN link between itself and the cable modem is lost then re-established. The standalone PS can use the event of the WAN Link between itself and the Cable Modem going down and regaining synchronization as a signal for the Cable Modem having gone through a reset. Since, when it is reset, the Cable Modem loses any ARP entry forwarding information for the MAC addresses of the standalone PS it had before the reset, there needs to be a mechanism for Cable Modem to re-learn all the standalone PS MAC addresses that are receiving frames from RF via the Cable Modem. The standalone PS will now use this link loss and regain information to initiate a DHCP renew for all the IP address leases that it currently has acquired via DHCP and that are still valid. This DHCP renew message will enable the Cable Modem to learn the MAC addresses of the standalone PS and re-establish the MAC forwarding it needs to perform for frames on the RF that are destined to the standalone PS device.

The CDP supports various DHCP Options and BOOTP Vendor Extensions, allowed by [RFC 2132].

The CDC determines the provisioning mode in which the PS is to operate based upon information received from the DHCP server in the DHCP ACK message, as introduced in 5.5, IPCable2Home Operational Models. The provisioning mode of an embedded PS can also be configured through an eDOCSIS compliant cable modem, by setting the value of the MIB object esafePsCableHomeModeControl [ITU-T Rec. J.126]. If esafePsCableHomeModeControl is set to provSystem(2), the ePS is required to initiate the provisioning process, and the CDC determines the provisioning mode. If esafePsCableHomeModeControl is set to disabledMode(1) or dormantCHMode(3), the CDC does not participate in the determination of the provisioning or operating mode.

DHCP provisioning mode of operation

The PS operates in DHCP provisioning mode if it receives a valid file name for the PS Configuration File in the *file* field and a valid IP address in the *siaddr* field of the DHCPACK message, and *does not* receive DHCP option 122 sub-options 3, 6 or 10.

Behaviour of the PS when operating in DHCP Provisioning Mode is summarized below:

- requires a PS configuration file to be downloaded from a cable network file server;
- defaults to using SNMPv1 and SNMPv2c for management messaging;
- defaults to using the docsDevNmAccessTable of the DOCSIS Device MIB [RFC 2669] to control access to the PS Database via specified MIBs;
- can be configured to use Transport Layer Security (TLS) [RFC 2246] to authenticate and encrypt the PS Configuration File (see 11.9 PS, Configuration File Security in DHCP Provisioning Mode);
- can be configured to operate in SNMPv3 Coexistence Mode, using Diffie-Hellman key management [RFC 2786], (see 6.3.3.1.4.2.2).

SNMP provisioning mode of operation

The PS operates in SNMP provisioning mode if it receives DHCP option 122 with sub-option fields 3, 6 and 10, *does not* receive a valid file name in the *file* field and *does not* receive a valid IP address in the *siaddr* field of the DHCPACK message.

Behaviour of the PS when operating in SNMP Provisioning Mode is summarized below:

- The PS is not required to download a PS configuration file from the cable network file server. The PS can be triggered to download a PS configuration file at any time but will operate using factory default parameters without downloading a PS configuration file.
- A configurable timer (cabhPsDevProvisioningTimer) controls the length of time the PS will wait to be triggered to download a PS configuration file after it completes authentication with the KDC and issues a Provisioning Enrollment SNMP inform. The value of cabhPsDevProvState will change from inProgress(2) to pass(1) after the amount of time specified by the value of cabhPsDevProvisioningTimer has elapsed if the PS is not triggered to download a configuration file. If the PS is triggered by the SNMP manager to download a PS configuration file before the time specified by the provisioning timer has elapsed, the value of cabhPsDevProvState will not change to pass(1) until the PS successfully downloads and completes the processing of the PS configuration file.
- The PS defaults to operating in SNMPv3 Coexistence Mode with SNMPv1 and SNMPv2 support *not* enabled (see 11.4, Secure Management Messaging to the PS).
- The PS defaults to using the User-based Security Model of SNMPv3 [RFC 3414] and View-based Access Control Model of SNMPv3 [RFC 3415] to control access to the PS Database via specified MIBs (see 11.4).
- The PS uses Kerberos message exchanges with a Key Distribution Centre server whose IP address is provided to the PS in DHCP option 177 sub-option 51, and AP listener to authenticate SNMPv3 messages (see 11.4.4.2, Security Algorithms for SNMPv3 in SNMP Provisioning Mode).
- The PS can be configured to receive and process SNMPv1 and SNMPv2c messages as well as SNMPv3 messages.

Dormant CableHome mode

The PS operates in Dormant CableHome Mode if it receives neither the combination of *file* field, *siaddr* field, or DHCP option code 122 sub-options to configure it for DHCP Provisioning Mode, nor the combination of these fields and sub-options to configure it for SNMP Provisioning Mode. An embedded PS can also be configured to operate in Dormant CableHome Mode through the eDOCSIS [ITU-T Rec. J.126] compliant embedded cable modem's MIB. If the value of eDOCSIS eSAFE MIB object esafePsCableHomeModeControl is dormantCHMode(3), the embedded PS is required to operate in Dormant CableHome Mode, regardless of the values of the DHCP file and siaddr fields or DHCP option 122 in DHCP messages received from the cable operator's DHCP server.

When the PS is operating in Dormant CableHome Mode, its behaviour is required to be as described in 7.3.3.2.4, including the following. This mode of operation is designed to enable the PS to operate and perform residential gateway functions when connected to a cable data network that does not yet support IPCable2Home provisioning and management systems:

- reject any SNMP messages received through any WAN interface;
- disable the TFTP client function;
- disable SYSLOG event reporting;
- terminate the provisioning timer;
- enable CNP, CAP, USFS, and CDS functionality.

The PS is required to include certain DHCP option fields in DHCP DISCOVER and DHCP REQUEST messages it issues to cable network DHCP servers. The Vendor Class Identifier Option (DHCP option 60) defines a CableLabs device class. For this Recommendation, the Vendor Class Identifier Option will contain the string "CableHome1.1", to identify a compliant Portal Services (PS) logical element, whenever the CDC requests a WAN-Man or WAN-Data address.

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. DHCP option 43 and its sub-options are defined in 7.3.3.2.4.

Details of the requirements for supporting DHCP options 60 and 43 are in Tables 7-6 and 7-7. Details related to other optional and mandatory DHCP options are provided in Table 7-8.

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.

7.3.3.2.3.1 DHCP client option 61

The PS element can have one or more WAN IP addresses associated with a one or more link layer (e.g., MAC) interfaces. Therefore, the CDC cannot rely solely on a MAC address as a unique client identifier value.

This Recommendation allows for the use of the Client Identifier Option (DHCP option 61), [RFC 2132] section 9.14, to uniquely identify the logical WAN interface associated with a particular IP address.

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).

7.3.3.2.3.2 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:

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 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. These two Interfaces share a single, common IP address. This 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 Address Mode. In WAN Address Mode 1, the value of cabhCdpWanDataIpAddrCount is zero.

WAN Address Mode 2

The PS Element 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 will have one WAN-Man and one or more WAN-Data IP Interface(s). All WAN-Data IP addresses will share a common hardware address that is unique from the WAN-Man hardware address. The two or more Interfaces (one WAN-Man and one or more WAN-Data) each has its own, unshared IP address. The CDP is configured by the cable operator to operate in WAN Address Mode 2 by writing a non-zero 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 might need software modification to include support for Client IDs (DHCP option 61) so that it can assign multiple IP addresses to the single WAN-Data hardware address.

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 non-zero 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 ($n \leq 0xFF$)

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.

7.3.3.2.4 CDC requirements

The PS MUST implement a DHCP client function in accordance with the Client requirements of [RFC 2131].

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.

The PS MUST broadcast DHCP DISCOVER in accordance with client requirements of [RFC 2131] and attempt to acquire a PS WAN-Man IP address lease during the PS boot process.

The PS MUST set cabhPsDevProvState to inProgress (2) when the PS broadcasts the DHCP DISCOVER message the first time following device reboot or PS reset. The PS ignores DHCP header fields and options used to determine Provisioning Mode and is not required to set cabhPsDevProvState to inProgress(2) when renewing its IP address lease via DHCP.

As a result of the process of renewing its IP address lease, the PS sets the Provisioning State object (cabhPsDevProvState) to the value pass(1) or to the value fail(2). When it renews its WAN-Man or WAN-Data IP address lease(s), the PS MUST update its system time and related MIB objects (cabhPsDevDateTime) based on the value of DHCP option 2 (Time Offset) of the DHCP ACK message if the value of cabhCdpTimeOffsetSelection is useDhcpOption2(1), OR based on the value of cabhCdpSnmpSetTimeOffset if the value of cabhCdpTimeOffsetSelection is useSnmpSetOffset(2), AND with a one-hour Daylight Saving Time adjustment during Daylight Saving Time if the value of cabhCdpDaylightSavingTimeEnable is enabled(1). When it renews its WAN-Man or WAN-Data IP address lease(s), the PS MUST update its lease information, including updating the values of cabhCdpWanDataAddrLeaseCreateTime and cabhCdpWanDataAddrLeaseExpireTime as appropriate, based on the value of DHCP option 51 (IP Address Lease Time). When it renews its WAN-Man or WAN-Data IP address lease(s), the PS MUST ignore DHCP option 122 sub-options 3, 6, and 10, and DHCP header file and siaddr fields.

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

The PS 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.

The standalone PS MUST initiate a DHCP renew for all the IP address leases that it currently has acquired via DHCP and that are still valid, whenever it learns there has been a link loss and regain on the WAN link between the standalone PS and the Cable Modem. This DHCP renew message

will enable the Cable Modem to learn the MAC addresses of the standalone PS and re-establish the MAC forwarding it needs to perform for frames on the RF that are destined to the standalone PS device.

It is assumed that in case of most embedded PS devices, when the Cable Modem goes through a reset, the embedded PS device will also go through a reset so as to avoid any inconsistent state between the two devices. The standalone PS MUST initiate a DHCP renew for all the IP address leases that it currently has acquired via DHCP and that are still valid, whenever it learns there has been a link loss and regain on the WAN link between the standalone PS and the Cable Modem.

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

If a non-zero 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.

The PS MUST implement the Vendor-Specific Information Option (DHCP option 43) as specified in Tables 7-7 and 7-8. Details of DHCP option 43 and its sub-options are further defined below. The definitions of DHCP option 43 sub-options MUST conform to requirements imposed by [RFC 2132].

The option begins with a type octet with the value of number 43, followed by a length octet. The length octet is followed by the number of octets of data equal to the value of the length octet. The value of the length octet does not include the two octets specifying the tag and length.

DHCP option 43 is a compound option. The content of option 43 is composed of one or more sub-options. Supported DHCP option 43 sub-options are: 1, 2, 3, 4, 5, 6, 11, 12, 13 and 14. A sub-option begins with a tag octet containing the sub-option code, followed by a length octet which indicates the total number of octets of data. The value of the length octet does not include itself or the tag octet. The length octet is followed by "length" octets of sub-option data.

The encoding of each option 43 sub-option is defined below. See Tables 7-7 and 7-8 for the intended purpose of each sub-option.

The PS MUST encode DHCP option 43 sub-option 1 by the number of octets equal to the value of the length octet of this sub-option, with each octet codifying a requested sub-option.

The PS MUST encode each of the DHCP option 43 sub-options 2, 3, 4, 5, 6, 12, 13 and 14 as a character string consisting of characters from the NVT ASCII character set, with no terminating NULL.

A standalone PS MUST send DHCP option 43 sub-option 2 containing the character string "SPS" (without the quotation marks).

An embedded PS MUST send DHCP option 43 sub-option 2 containing the character string "EPS" (without the quotation marks).

A standalone PS MUST send DHCP option 43 sub-option 3 containing the character string "SPS" (without the quotation marks).

An embedded PS MUST send DHCP option 43 sub-option 3 containing a colon-separated list of all device types in the complete device, including at a minimum the colon-separated character string "ECM:EPS" (without the quotation marks).

If the PS is requesting a PS WAN-Man IP address lease, it MUST send DHCP option 43 sub-option 11 containing the value 0x01, encoded as a binary number, in its DHCP DISCOVER and DHCP REQUEST messages.

If the PS is requesting a PS WAN-Data IP address lease, it MUST send DHCP option 43 sub-option 11 containing the value 0x02, encoded as a binary number, in its DHCP DISCOVER and DHCP REQUEST messages.

Table 7-6 summarizes how the PS is required to set the values for DHCP option 43, sub-option 11, for the WAN interfaces of the PS.

Table 7-6/J.192 – 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 length limit of sub-option 4, 5, 6, 12, 13 and 14 is each 255 octets. Thus the total length of option 43 could exceed 255 octets. If the total number of octets in all DHCP option 43 sub-options exceeds 255 octets, the PS MUST follow RFC 3396 to split the option into multiple smaller options.

The PS MUST implement the Vendor Class Identifier Option (DHCP option 60) as specified in Tables 7-7 and 7-8.

In the case of an Embedded PS with cable modem, the cable modem and PS element each send separate DHCP requests. Table 7-7 describes how the PS 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 7-7/J.192 – DHCP options for embedded PS WAN-Man and WAN-Data Address Requests

DHCP Request options	Value	Description
Embedded Portal Services DHCP Request for WAN Management Address		
CPE option 60	"CableHome1.1"	
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., "1.0.2"	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 cabhSec2FirewallPolicyCurrentVersion
Embedded Portal Services DHCP Request for WAN-Data Address		
CPE option 60	"CableHome1.1"	
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 7-8 describes to what the PS MUST set the contents of options 60 and 43, when the PS is a standalone device.

Table 7-8/J.192 – DHCP options for standalone PS WAN-Man and WAN-Data Address Requests

DHCP Request options	Value	Description
Standalone Portal Services DHCP Request for WAN Management Address		
CPE option 60	"CableHome1.1"	
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"	Standalone PS
CPE option 43 sub-option 3	"SPS"	List of Embedded devices (Standalone PS only)
CPE option 43 sub-option 4	e.g., "123456"	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., "1.0.2"	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 revision from docsDevSwCurrentVers
CPE option 43 sub-option 14	e.g., "1.2.3..."	Firewall Policy File Version from cabhSec2FirewallPolicyCurrent Version
Standalone Portal Services DHCP Request for WAN-Data Address		
CPE option 60	"CableHome1.1"	
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"	Standalone 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"	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

For a detailed description of the contents of the PS sysDescr object, see 6.3.3.1.4, SNMP Agent Function Requirements.

The PS MUST support the DHCP Options indicated as mandatory in the CDC Protocol Support column in Table 7-9. Table 7-9 lists the DHCP Options that are mandatory and optional for the CDC to support.

Table 7-9/J.192 – 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
122	Sub-option 3 – Service Provider's SNMP Manager Address	M
122	Sub-option 6 – Kerberos Realm Name of the Provisioning Realm	M
122	Sub-option 10 – Kerberos Server IP address	M

The PS MUST include DHCP options listed as mandatory in Table 7-10 in DHCP DISCOVER and DHCP REQUEST messages sent to the cable network DHCP server.

Table 7-10/J.192 – CDC DHCP options in DISCOVER and REQUEST messages

Option number	Option function	CDC protocol inclusion (M)andatory
255	End	M
43	Vendor-specific information	M
50	Requested IP Address	M(DHCP REQUEST only)
55	Parameter Request List	M
60	Vendor-class identifier	M
61	Client-identifier	M

The PS MUST request DHCP options listed as mandatory in Table 7-11, within the DHCP option 55 (Parameter Request List) [RFC 2132] sent in the DHCP DISCOVER and DHCP REQUEST messages.

Table 7-11/J.192 – CDC DHCP options requested within option 55

Option number	Option function	CDC protocol inclusion (M)andatory
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
15	Domain Name	M
23	Default Time-to-live	M
26	Interface MTU	M
51	IP address Lease Time	M
54	Server Identifier	M
122	CableLabs Client Configuration Option	M

The following list identifies the actions the PS is required to take if any of the DHCP options requested in DHCP option 55 (listed in Table 7-11) are absent from the DHCP OFFER message it receives from the DHCP server.

1) If DHCP option 1 (Subnet Mask) is not present in the DHCP lease offer message the PS received from the DHCP server, the reply is incomplete and the PS MUST generate Event ID 68000301 AND restart the provisioning process beginning with broadcasting a DHCP DISCOVER message through its WAN interface.

2) If DHCP option 2 (Time Offset) is not present in the DHCP lease offer message the PS received from the DHCP server, the lease offer is not considered incomplete and the PS is permitted to accept the lease.

3) If DHCP option 3 (Router) is not present in the DHCP lease offer message the PS received from the DHCP server, the lease offer is not considered incomplete and the PS is permitted to accept the lease.

4) If DHCP option 4 (Time Server) is not present in the DHCP lease offer message the PS received from the DHCP server, the reply is incomplete and the PS MUST generate Event ID 68000301 AND restart the provisioning process beginning with broadcasting a DHCP DISCOVER message through its WAN interface.

5) If DHCP option 6 (Domain Name Server) is not present in the DHCP lease offer message the PS received from the DHCP server, the reply is incomplete and the PS MUST generate Event ID 68000301 AND restart the provisioning process beginning with broadcasting a DHCP DISCOVER message through its WAN interface.

6) If DHCP option 7 (Log Server) is not present in the DHCP lease offer message the PS received from the DHCP server, the lease offer is not considered incomplete and the PS is permitted to accept the lease.

7) If DHCP option 15 (Domain Name) is not present in the DHCP lease offer message the PS received from the DHCP server, the lease offer is not considered incomplete and the PS is permitted to accept the lease.

8) If DHCP option 23 (Default IP Time-to-Live) is not present in the DHCP lease offer message the PS received from the DHCP server, the lease offer is not considered incomplete and the PS is permitted to accept the lease.

9) If DHCP option 26 (Interface MTU) is not present in the DHCP lease offer message the PS received from the DHCP server, the lease offer is not considered incomplete and the PS is permitted to accept the lease.

10) If DHCP option 51 (IP Address Lease Time) is not present in the DHCP lease offer message the PS received from the DHCP server, the reply is incomplete and the PS MUST generate Event ID 68000301 AND restart the provisioning process beginning with broadcasting a DHCP DISCOVER message through its WAN interface.

11) If DHCP option 54 (Server Identifier) is not present in the DHCP lease offer message the PS received from the DHCP server, the lease offer is not considered incomplete and the PS is permitted to accept the lease.

12) If DHCP option 122 (CableLabs Client Configuration Option) is not present in the DHCP lease offer message the PS received from the DHCP server, the lease offer is not considered incomplete and the PS is permitted to accept the lease.

If the PS does not receive a complete lease offer after exhausting the maximum number of retry attempts, the PS MUST operate in Dormant Mode, as described in 7.3.3.2.4.

The PS MUST support a Service Provider's SNMP Entity Manager Address (DHCP option 122 sub-option 3) configured as an IPv4 address. The format of DHCP option 122 sub-option 3 is described in [RFC 3495].

The PS MUST support a Kerberos Realm Name (DHCP option 122 sub-option 6). A Kerberos realm name is required by the PS to permit a DNS lookup for the address of the service provider's Key Distribution Centre (KDC) entity. The format of DHCP option 122 sub-option 6 is described in [RFC 3495].

The PS MUST support a Key Distribution Centre (KDC) server IP address (DHCP option 122 sub-option 10). The KDC server IP address sub-option informs the PS of the network address of one or more Key Distribution Centre servers.

The encoding of the KDC Server Address sub-option is described in [RFC 3634].

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:

– "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 whenever there is no regular WAN-Data IP address available at the PS. 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 to the cable modem's diagnostic IP address (192.168.100.1). The PS MUST default to the "Fallback" WAN-Data IP address immediately after power up and whenever current WAN-Data IP address leases expire such that no WAN-Data IP address is left active, in order to provide continuous access to the CM diagnostic capabilities. The PS MUST NOT use the "Fallback" WAN-Data IP address 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 PS 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 PS MUST always insert its WAN hardware address into the Client ID (DHCP option 61) field in the DHCP Discover message.

If during its attempt to acquire a lease for the PS WAN-Man IP address the CDC receives no DHCP OFFER, the PS MUST log Event ID 68000100 in the local log and re-broadcast a DHCP DISCOVER message (i.e., restart the provisioning sequence in the event of this failure condition) – repeating the DHCP lease acquisition attempt up to 5 times. If on its fifth attempt to acquire a PS WAN-Man IP address lease the CDC receives no DHCP OFFER, the PS MUST use the "Fallback" WAN IP address, netmask, and default gateway as described above and continue to attempt to acquire a valid WAN-Man IP address by broadcasting DHCP DISCOVER out its WAN interface every 10 seconds until a valid DHCP lease is granted for the WAN-Man IP address.

When a PS operating in WAN Address Mode 2 (as described in 7.3.3.2) 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 PS MUST include the Client Identifier option (cabhCdpWanDataAddrClientId) in the DHCP Discover message. To enable these unique 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 7.3.3.2) the PS 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 PS MUST continue to retransmit the broadcast DHCP DISCOVER message implementing a randomized exponential backoff algorithm, consistent with that described in [RFC 2131], until it acquires a valid PS WAN-Man IP and/or PS WAN-Data IP address lease, as needed.

If the PS (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 7.5.4, before attempting to download the PS Configuration File.

If the PS (CDC) 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), so that the CDS can serve DHCP requests from LAN IP Devices in the LAN-Trans realm.

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

When the WAN-Man DHCP lease expires, the PS MUST clear all row entries from the cabhCdpWanDnsServerTable.

Provisioning PS operating modes

This clause defines PS requirements for operation in modes introduced in 5.5.

If a PS is embedded as an eSAFE with an eDOCSIS [eDOCSIS1] compliant cable modem and the eDOCSIS esafePsCableHomeModeControl MIB object is set to provSystem(2), then the embedded PS MUST attempt to acquire a PS WAN-Man IP address lease, and adhere to requirements for DHCP Provisioning Mode, SNMP Provisioning Mode, or Dormant CableHome Mode as described below. Additional requirements for embedded PS operation based on the value of esafePsCableHomeModeControl are defined below.

DHCP provisioning mode

If during the process of acquiring a lease for the PS WAN-Man IP address the CDC receives, in the DHCP ACK message [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 122 sub-option 3, sub-option 6, or sub-option 10 (valid combination 1), the PS MUST set cabhPsDevProvMode to dhcpmode(1) and attempt to synchronize time of day with the ToD server as described in 7.5.4, Time of Day Client Function Requirements. Depending upon the value of the siaddr and file fields of the DHCP message header, the PS might be required to attempt to download a configuration file. Refer to 7.4.4.2, BPSC Triggering Requirements.

SNMP provisioning mode

If during the process of acquiring 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 122 with a valid IP address (SNMP Manager'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 10, 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 snmpmode(2) 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 11.3.4, Authentication Infrastructure Requirements. The PS operating in SNMP Provisioning Mode could also be configured to attempt to download a configuration file. Refer to 7.4.4.2, BPSC Triggering Requirements.

Dormant CableHome mode

If during the process of acquiring a lease for the PS WAN-Man IP address the CDC receives, in the DHCP ACK from the DHCP server in the cable network, any combination of DHCP option 122 sub-options 3, 6, and 10, '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 event ID 68000301 (see Table B.1, Defined Events for IPCable2Home) AND do the following on the assumption that it is connected via cable modem to a cable data network that does not support CableHome provisioning (Dormant CableHome 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 be operating without a WAN-Man IP address, which is different from any of the WAN Address Modes described in 7.3.3.2.3.2.
- Terminate the provisioning timer.

- Set the value of cabhPsDevProvMode to dormantCHmode(3).
- Set the value of cabhPsDevProvState to fail(3).
- Enable the CDS.
- Enable the CAP and USFS functionality.
- Enable the CNP.
- Enable the firewall.
- Operate with parameters that have been provisioned in the past, including those values of persistent MIB objects. The PS operating in Dormant CableHome Mode MUST NOT reset its MIB objects to factory default settings.

An embedded PS can also be configured to operate in Dormant CableHome Mode through the eDOCSIS eSAFE MIB object esafePsCableHomeModeControl [eDOCSIS1]. If the embedded eDOCSIS compliant cable modem's esafePsCableHomeModeControl object is set to dormantCHMode(3), the ePS is required to attempt to acquire an IP address lease and operate in Dormant CableHome Mode as described above, irrespective of the values of the DHCP siaddr and file fields or of the presence or absence of DHCP option 122 and its sub-options, in the DHCP OFFER and DHCP ACK messages. See Table 7-12.

Disabled mode

When an embedded PS is configured to operate in Disabled Mode through the eDOCSIS eSAFE MIB (esafePsCableHomeModeControl = disabledMode(1)) [eDOCSIS1], then the embedded PS MUST do the following:

- release its WAN-Man IP address lease and any WAN-Data IP address leases if it has either, or stop attempting to acquire a WAN-Man or WAN-Data IP address lease by silently discarding DHCP OFFER, DHCP ACK, or other DHCP messages;
- act as a transparent bridge as defined in [ISO/IEC 10038], between the WAN-Data realm and LAN-Pass realm, including transparently bridging all frame types that the DOCSIS specifications [DOCSIS1], [DOCSIS9] require a cable modem to pass;
- NOT perform any C-NAT or C-NAPT transparent routing functions;
- disable the DHCP server (CDS), HTTP server, SNMP agent, and DNS (CNP) functionality;
- disable the firewall;
- drop all packets addressed to the PS Server Router address (cabhCdpServerRouter) or to the PS "well-known" LAN IP address 192.168.0.1; and
- give precedence to Upstream Selective Forwarding Switch (USFS) processing, over LAN-to-WAN bridging decisions.

The only way to configure a PS to operate in Disabled Mode is to set the value of an eDOCSIS cable modem's eSAFE MIB object esafePsCableHomeModeControl to disabled(1). Disabled Mode does not apply to a standalone PS since it is not embedded as an eSAFE with an eDOCSIS cable modem.

The embedded PS can be taken out of Disabled Mode by setting the value of the embedded cable modem's esafePsCableHomeModeControl eSAFE MIB object to provSystem(2) or to dormantCHMode(3) [eDOCSIS1].

When the PS is administratively changed from (taken out of) Disabled Mode, the PS MUST set all CableHome MIB objects to their factory default values.

Table 7-12 defines actions the embedded PS is required to take when the eDOCSIS eSAFE MIB object esafePsCableHomeModeControl [eDOCSIS1] is set, for each PS operating mode. The ePS MUST take the action listed in Table 7-12 if it is operating in the mode listed in the first column,

'Current ePS Mode', and the eDOCSIS eSAFE MIB object esafePsCableHomeModeControl is set as indicated in the second column, 'esafePsCableHomeModeControl'. Note that the Current ePS Mode listed in the first column of Table 7-12 is instrumented in the eDOCSIS MIB object esafePsCableHomeModeStatus.

Table 7-12/J.192 – Required ePS Actions for esafePsCableHomeModeControl settings

Current ePS Mode	esafePsCableHomeModeControl	Required ePS actions
Disabled Mode	disabledMode(1)	No action required
Dormant CableHome Mode	disabledMode(1)	Take actions listed in the Disabled Mode clause above
CableHome Mode	disabledMode(1)	Take actions listed in the Disabled Mode clause above
Disabled Mode	provSystem(2)	Restart the provisioning process beginning with broadcasting DHCP DISCOVER through the WAN interface
Dormant CableHome Mode	provSystem(2)	Restart the provisioning process beginning with broadcasting DHCP DISCOVER through the WAN interface
CableHome Mode	provSystem(2)	Restart the provisioning process beginning with broadcasting DHCP DISCOVER through the WAN interface
Disabled Mode	dormantCHMode(3)	– Restart the provisioning process beginning with broadcasting DHCP DISCOVER through the WAN interface – Do not include DHCP option 43 sub-options 2-14 in DHCP DISCOVER / REQUEST messages – Do not include 'CableHome1.0' or 'CableHome1.1' as DHCP option 60 value – Do not include DHCP option 122 in DHCP option 55 parameter request list – Ignore DHCP file and siaddr header fields in DHCP OFFER and DHCP ACK messages: do not download a configuration file and do not operate in DHCP Provisioning Mode or SNMP Provisioning Mode, regardless of the values of DHCP file and siaddr fields and DHCP options. – Take actions listed in the Dormant CableHome Mode clause above
Dormant CableHome Mode	dormantCHMode(3)	No action required

Table 7-12/J.192 – Required ePS Actions for esafePsCableHomeModeControl settings

Current ePS Mode	esafePsCableHomeModeControl	Required ePS actions
CableHome Mode	dormantCHMode(3)	– Restart the provisioning process beginning with broadcasting DHCP DISCOVER through the WAN interface – Do not include DHCP option 43 sub-options 2-14 in DHCP DISCOVER/REQUEST messages – Do not include 'CableHome1.0' or 'CableHome1.1' as DHCP option 60 value – Do not include DHCP option 122 in DHCP option 55 parameter request list – Ignore DHCP file and siaddr header fields in DHCP OFFER and DHCP ACK messages: do not download a configuration file and do not operate in DHCP Provisioning Mode or SNMP Provisioning Mode, regardless of the values of DHCP file and siaddr fields and DHCP options. – Take actions listed in the Dormant CableHome Mode clause above

7.4 PS function – Bulk Portal Services Configuration (BPSC)

7.4.1 Bulk Portal Services Configuration function goals

The primary goals of the BPSC function are to request, receive, and process PS and firewall configuration parameters.

7.4.2 Bulk Portal Services Configuration function system design guidelines

The guideline identified in Table 7-13 has guided specification of capabilities for the Bulk PS Configuration function:

Table 7-13/J.192 – Bulk Portal Services system design guidelines

Number	Guidelines
BPSC 1	Provide a mechanism by which the PS can download and process PS and Firewall Configuration Files.

7.4.3 Bulk Portal Services Configuration function system description

Bulk Portal Services configuration is typically carried out during the provisioning of the PS element, via the processing of configuration settings contained within a configuration file. However, this process may be initiated at any time. Within this clause, the term "configuration file" is used to mean either the PS Configuration File or the Firewall Configuration File. Specific requirements for either type of configuration file will be labelled with the appropriate file label, i.e., PS Configuration File or Firewall Configuration File. The Bulk PS Configuration tool consists of the following components:

- The format of the Configuration File;
- Modes of triggering the download process;
- Means of authenticating the file;
- Means of reporting back the status of the configuration file download and other considerations.

Bulk PS Configuration (BPSC) is a tool that Operators can use to change PS and Firewall configuration settings in bulk, via a Configuration File. Typically, the Configuration File will contain many settings, since the primary usefulness afforded by Configuration Files use is the ability to change a number of configuration settings with minimal cable operator intervention. However, it is expected that the Firewall Configuration File will only be used for firewall-specific settings.

The Bulk PS Configuration process can behave the same as successive SNMP sets executed by an operator manually. The Configuration File is a tool meant to make operators more productive and to make large configuration changes less error prone.

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.

7.4.4 Bulk Portal Services Configuration function requirements

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 7.3.3.2. The PS is not required to download a Firewall Configuration File in either DHCP or SNMP Provisioning Mode.

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

7.4.4.1 Configuration file format requirements

PS or firewall configuration data MUST be contained in a file, which is downloaded via TFTP or HTTPS. The Configuration File MUST consist of a number of configuration settings (1 per parameter), each of the form "Type Length Value (TLV)". Definitions of these terms are provided in Table 7-14.

Table 7-14/J.192 – TLV definitions

Type	A single-octet identifier which defines the parameter
Length	A two-octet field specifying the length of the Value field (not including Type and Length fields)
Value	A set of octets Length long containing the specific value for the parameter

The configuration settings MUST follow each other directly in the file, which is a stream of octets (no record markers). The 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 7.4.4.1.1 for a definition of the pad. Configuration settings are divided into three types:

- Configuration settings which are required to be present;
- Additional or optional IPCable2Home-specified configuration settings which MAY be present;
- Vendor-specific configuration settings.

A PS Configuration File MAY contain many different parameters, but the only parameters that MUST be included in the PS configuration file are the PS Message Integrity Check (MIC) (Type 53) and the End of Data Marker (Type 255). A Firewall Configuration File MAY contain many different Type 28 TLV parameters for configuring the firewall, but the only parameter that MUST be included in the Firewall Configuration File is the End of Data Marker (Type 255). If the Firewall Configuration File contains a PS Message Integrity Check (MIC) (Type 53), the PS MUST ignore it.

To allow uniform management of the PS, the PS MUST support a Configuration File that is up to 64 K-bytes long.

Each Portal Services element MUST support configuration parameter Types 0, 9, 10, 21, 28, 32, 33, 34, 38, 43, 53 and 255, which are described in this clause. Each TLV parameter in the Firewall Configuration File describes a firewall attribute. Since the IPCable2Home firewall is configured via access to the IPCable2Home Security MIB (see 11.6.4, Firewall Requirements), a Firewall Configuration File typically includes TLV Type 28 configuration settings, which contain SNMP MIB objects. Vendor-specific firewall configuration information is permitted to be passed to the PS in the Firewall Configuration File using the vendor-specific configuration setting Type 43 (TLV-43). If the configuration file does not contain the required attributes, the PS MUST reject the file.

The size of the value in the Length field for any configuration parameter included in an IPCable2Home configuration file MUST be 2 octets.

The Length value for each Type described in the TLV descriptions in this clause is the actual length in octets of the Value field.

7.4.4.1.1 Pad configuration setting

This has no Length or Value fields and is only used following the end of data marker to pad the file to an integral number of 32-bit words.

Type	Length	Value
0	---	---

7.4.4.1.2 Software upgrade filename

The filename of the software upgrade file for the 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

7.4.4.1.3 SNMP Write-Access Control

This object makes it possible to disable SNMP "Set" access to individual MIB objects. Each instance of this object controls access to all of the writeable MIB objects whose Object ID (OID) prefix matches. This object may be repeated to disable access to any number of MIB objects.

Type	Length	Value
10	n	OID prefix plus control flag

Where n is the size of the ASN.1 Basic Encoding Rules [ITU-T Rec. X.690 | ISO/IEC 8825-1] encoding of the OID prefix plus one byte for the control flag.

The control flag may take values:

0 – allow write-access

1 – disallow write-access

Any OID prefix may be used. The Null OID 0.0 may be used to control access to all MIB objects. (The OID 1.3.6.1 will have the same effect.)

When multiple instances of this object are present and overlap, the longest (most specific) prefix has precedence.

Thus, one example might be:

someTable disallow write-access

someTable.1.3 allow write-access

This example disallows access to all objects in someTable except for someTable.1.3.

7.4.4.1.4 Software upgrade TFTP server

The IP address of the TFTP server, on which the software upgrade file for the IPCable2Home device resides.

Type	Length	Value
21	4	ip1, ip2, ip3, ip4

7.4.4.1.5 First-phase SNMP MIB object with extended length

This object allows SNMP MIB objects to be Set via the TFTP-Registration process prior to SNMP Sets done with TLV-28. The intent of this TLV is to include only those SNMP Sets that have to occur prior to other SNMP Sets to ensure correct operation, such as SetToFactory objects (e.g., cabhPsDevSetToFactory) that clear persistent MIB objects. Non-priority SNMP Sets are expected to be included in TLV-28.

The value of this parameter is an SNMP variable binding (VarBind) as defined in [RFC 3416]. The VarBind is encoded in ASN.1 Basic Encoding Rules, just as it would be if part of an SNMP Set Request PDU.

Type	Length	Value
27	Variable	variable binding

The PS MUST treat the variable binding, in a Type 27 TLV, as if it were part of an SNMP Set Request with the following caveats:

- It MUST treat the request as fully authorized (it cannot refuse the request for lack of privilege).
- SNMP Write-Control provisions do not apply.
- No SNMP response is generated by the PS.
- This object MAY be repeated with different VarBinds to "Set" a number of MIB objects. All SNMP Sets in a Configuration File that occur within Type 27 TLVs MUST be treated as if simultaneous. Each VarBind MUST be limited to 65 535 bytes.
- This object MUST be processed before any Type 28 TLV present in the Configuration File are processed.

7.4.4.1.6 SNMP MIB object with extended length

This object allows arbitrary SNMP MIB objects to be Set via the TFTP-Registration process, where the value is an SNMP variable binding (VarBind) as defined in [RFC 3416]. The VarBind is encoded in ASN.1 Basic Encoding Rules, just as it would be if part of an SNMP Set request.

Type	Length	Value
28	Variable	variable binding

The PS MUST treat the variable binding, in a Type 28 TLV, as if it were part of an SNMP Set Request with the following caveats:

- It MUST treat the request as fully authorized (it cannot refuse the request for lack of privilege).
- SNMP Write-Control provisions (see previous clause) do not apply.
- No SNMP response is generated by the PS.
- This object MAY be repeated with different VarBinds to "Set" a number of MIB objects. All SNMP Sets in a Configuration File that occur within Type 28 TLVs MUST be treated as if simultaneous. Each VarBind MUST be limited to 65 535 bytes.

7.4.4.1.7 Manufacturer Code Verification Certificate

The Manufacturer's Code Verification Certificate (M-CVC) for Secure Software Downloading. Refer to 11.8.4.4.2, Network Initialization.

Type	Length	Value
32	Variable	Manufacturer CVC (DER-encoded ASN.1)

7.4.4.1.8 Co-signer Code Verification Certificate

The Co-signer's Code Verification Certificate (C-CVC) for Secure Software Downloading. Refer to 11.8.4.4.2, Network Initialization.

Type	Length	Value
33	Variable	Co-signer CVC (DER-Encoded ASN.1.)

7.4.4.1.9 SNMPv3 Kickstart Value

(see C.1.2.8 DOCSIS 1.1 RFI Specification SP-RFIv1.1-I09-020830.)

Compliant Portal Services elements MUST understand the following TLV and its sub-elements and be able to kickstart SNMPv3 access to the PS regardless of whether the PS is operating in NmAccess Mode or Coexistence Mode (see 6.3.3, CMP System Description, and 6.3.3.1.4.2, Network Management Mode Requirements).

Type	Length	Value
34	n	Composite

Up to 5 of these objects may be included in the configuration file. Each results in an additional row being added to the usmDHKickstartTable and the usmUserTable and results in an agent public number being generated for those rows.

7.4.4.1.9.1 SNMPv3 Kickstart Security Name

Type	Length	Value
34.1	2-16	UTF8 Encoded security name

For the ASCII character set, the UTF8 and the ASCII encodings are identical. Normally, this will be specified as one of the IPCable2Home built-in USM users, e.g., "CHAdministrator".

The security name is NOT zero terminated. This is reported in the usmDHKickStartTable as usmDHKickStartSecurityName and in the usmUserTable as usmUserName and usmUserSecurityName.

7.4.4.1.9.2 SNMPv3 Kickstart Manager Public Number

Type	Length	Value
34.2	n	Manager's Diffie-Hellman public number expressed as an octet string.

This number is the Diffie-Hellman public number derived from a privately (by the manager or operator) generated random number and transformed according to [RFC 2786]. This is reported in the usmDHKickStartTable as usmKickstartMgrPublic. When combined with the object reported in the same row as usmKickstartMyPublic, it can be used to derive the keys in the related row in the usmUserTable.

7.4.4.1.10 SNMP Notification Receiver

Type	Length	Value
38	n	Composite

This PS Configuration File element specifies a Network Management Station that will receive notifications from the PS when it is in Coexistence network management mode. This TLV (38) consists of several sub-TLVs inside the TLV configuration file element. Up to 10 of these elements may be included in the PS Configuration File. Clause 6.3.3.1.4.6, Mapping TLV Fields Into Created SNMPv3 Table Rows, provides details about how this configuration file element is mapped into SNMPv3 functional tables.

All multi-byte fields of this sub-TLV MUST be placed in the network byte order.

7.4.4.1.10.1 Sub-TLV 38.1 – IP Address of trap receiver

IPv4 address of the trap receiver, in binary.

Type	Length	Value
38.1	4	IP address

7.4.4.1.10.2 Sub-TLV 38.2 – UDP Port number of the trap receiver

UDP Port number of the trap receiver, in binary.

Type	Length	Value
38.2	2	UDP Port

If this sub-TLV is not present in a configuration file, the default value 162 is used.

7.4.4.1.10.3 Sub-TLV 38.3 – Type of trap sent by the PS (Note 2)

Trap type.

Type	Length	Value
38.3	2	Trap type

The PS MUST support the following trap type values:

1 = SNMPv1 trap in an SNMPv1 packet

2 = SNMPv2c trap in an SNMPv2c packet

3 = SNMPinform in an SNMPv2c packet

4 = SNMPv2c trap in an SNMPv3 packet

5 = SNMPinform in an SNMPv3 packet

7.4.4.1.10.4 Sub-TLV 38.4 – Timeout

Timeout, in milliseconds, used for sending SNMP inform messages.

Type	Length	Value
38.4	2	0-65 535

7.4.4.1.10.5 Sub-TLV 38.5 – Retries

Number of retries when sending an inform, after sending the inform the first time.

Type	Length	Value
38.5	2	0-65 535

7.4.4.1.10.6 Sub-TLV 38.6 – Notification Filtering Parameters

Type	Length	Value
38.6	n	Filter OID

Where n is the size of the ASN.1-encoded Filter Object Identifier.

Filter OID is an ASN.1-formatted Object Identifier of the snmpTrapOID value that identifies the notifications to be sent to the notification receiver. This notification and all below it will be sent.

If this Sub-TLV is not present, the notification receiver will receive all notifications generated by the SNMP agent.

7.4.4.1.10.7 Sub-TLV 38.7 – Security Name to use when sending SNMPv3 Notification

Type	Length	Value
38.7	2-16	UTF8-encoded security name

This sub-TLV is not required for Trap type = 1, 2, or 3. The PS MUST ignore sub-TLV 38.7 if the trap type in sub-TLV 38.3 is 1, 2, or 3. If sub-TLV 38.7 is not supplied for a Trap type of 4 or 5, the PS MUST send the SNMPv3 Notification in the noAuthNoPriv security level using the security name "@PSconfig". (Note 2)

SecurityName

The SNMPv3 Security Name to use when sending an SNMPv3 Notification. Only used if Trap Type is set to 4 or 5. This name MUST be a name specified in a Config File TLV Type 34 as part of the DH Kickstart procedure. The notifications MUST be sent using the Authentication and Privacy Keys calculated by the PS during the DH Kickstart procedure.

Notes to 7.4.4.1.10.x:

NOTE 1 – Upon receiving one of these TLV elements, the PS MUST make entries to the following tables in order to cause the desired trap transmission: snmpNotifyTable, snmpTargetAddrTable, snmpTargetParamsTable, snmpNotifyFilterProfileTable, snmpNotifyFilterTable, snmpCommunityTable, usmUserTable, vacmSecurityToGroupTable, vacmAccessTable, and vacmViewTreeFamilyTable

NOTE 2 – Trap Type: The community String for traps in SNMPv1 and v2 packets MUST be "public". The Security Name in traps and informs in SNMPv3 packets where no security name has been specified MUST be "@PSconfig" and in that case the security level MUST be NoAuthNoPriv.

NOTE 3 – Filter OID: SNMPv3 allows the specification of which Trap OIDs are to be sent to a trap receiver. The filter OID in the config element specifies the OID of the root of a trap filter sub-tree. All Traps with a Trap OID contained in this trap filter sub-tree MUST be sent to the trap receiver.

NOTE 4 – The PS Configuration File is permitted to also contain TLV MIB elements (TLV-28) that make entries to any of the 10 tables listed in Note 1. The PS MUST ignore TLV MIB elements that use index columns that start with the characters "@PSconfig".

7.4.4.1.11 Vendor-specific information

If vendor-specific information is provided to the PS, it MUST be encoded in the vendor-specific information field (VSIF) (code 43) using the Vendor ID field to specify which TLV tuples apply to which vendors' products. A properly-formed VSIF has a single Vendor ID Sub-TLV (code 43.1) as the first sub-TLV. The PS MUST reject the PS Configuration File if any VSIF (Type 43) TLV is not correctly formed.

A PS configuration file can have multiple VSIFs with either different or the same Vendor ID Sub-TLVs present. The PS will process only those VSIFs that have a Vendor ID Sub-TLV matching Vendor ID and will ignore the VSIFs that have Vendor ID Sub-TLVs which do not match. Vendor-specific sub-types are allowed to be added after Type 43.1.

Type	Length	Value
43	N	vendor-specific settings

Sub-TLV 43.1 – Vendor ID type

Vendor identification specified by the three-byte Organization Unique Identifier of the PS vendor.

Type	Length	Value
43.1	3	v1, v2, v3

7.4.4.1.12 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 NIST, FIPS PUB 180-1: Secure Hash Standard, April 1995. This TLV is only used in the configuration file immediately before the end of data marker.

7.4.4.1.13 End-of-Data Marker

This is a special marker for end of data. It has no Length or Value fields.

Type	Length	Value
255	---	---

7.4.4.2 BPSC triggering requirements

Transfer of the configuration file, from the TFTP server or HTTPS server in the cable data network to the PS, is initiated by an event referred to as a trigger. Requirements for triggering the transfer of a PS Configuration File or Firewall Configuration File from the TFTP server or HTTPS server to the PS follow.

The mode of triggering the PS Configuration File download is dependent upon the Provisioning Mode in which the PS is operating. The CMP MUST read the value of cabhPsDevProvMode (see 7.3.3.2.4) prior to initiating any PS Configuration File download. The method of triggering for the Firewall Configuration File download is not dependent upon the Provisioning Mode.

7.4.4.2.1 PS configuration file download trigger for DHCP provisioning mode

If the PS receives the TFTP or HTTPS server address in the 'siaddr' field and the PS Configuration File name in the 'file' field of the DHCP ACK, AND the value of cabhPsDevProvState = inProgress(2), the PS MUST combine the server address and PS Configuration File name to form a URL-encoded value and write that value into PSDev MIB object cabhPsDevProvConfigFile. The PS MUST use the following format for the URL-encoded value for the TFTP server IP address and PS Configuration File name:

tftp://IPv4_address_of_the_TFTP_server/full_path_ to_the_PS_Configuration_File/PS_Configuration_File_name

The PS MUST use the following format for the URL-encoded value for the HTTPS server IP address and PS Configuration File name:

https://IPv4_address_of_the_HTTPS_server/full_path_ to_the_PS_Configuration_File/PS_Configuration_File_name

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 or HTTPS server IP address) and name in the DHCP message issued to the PS (CDC) by the DHCP server in the cable network. Refer to 7.3.3.2.4, CDC Requirements.

If the PS is operating in DHCP Provisioning Mode (as indicated by the value of cabhPsDevProvMode), after the PS (CDC) receives a DHCP ACK from the DHCP server in the cable network, and the IP address in the 'siaddr' field does not match the first IP address in DHCP option 72, AND the value of cabhPsDevProvState = inProgress(2), then the PS MUST issue a TFTP Get request to the server identified in the DHCP message 'siaddr' field to download the configuration file.

If the PS is operating in DHCP Provisioning Mode (as indicated by the value of cabhPsDevProvMode), after the PS (CDC) receives a DHCP ACK from the DHCP server in the cable network, and the IP address in the 'siaddr' field matches the first IP address in DHCP option 72, and the cabhPsDevTodSyncStatus MIB object has a value of '1' (ToD access succeeded), then the PS MUST establish a TLS session as defined in clause 11, and issue a HTTP Get request to the server identified in the DHCP message 'siaddr' field, to download the configuration file.

If the PS is operating in DHCP Provisioning Mode (as indicated by the value of cabhPsDevProvMode), after the PS (CDC) receives a DHCP ACK from the DHCP server in the cable network, and the IP address in the 'siaddr' field matches the first IP address in DHCP option 72, and the cabhPsDevTodSyncStatus MIB object has a value of '2' (ToD access failed), the PS MUST wait until the cabhPsDevTodSyncStatus MIB object has a value of '1' (ToD access succeeded), before establishing a TLS session as defined in clause 11, and issuing an HTTP Get request to the server identified in the DHCP message 'siaddr' field, to download the configuration file.

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.

7.4.4.2.2 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 SNMPv3 set-up process (refer to 11.4, Secure Management Messaging to the PS, for details about the SNMP set-up 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 the cabhPsDevTodSyncStatus MIB object has a value of '2' (ToD access failed).

If the PS is operating in SNMP Provisioning Mode and the time elapsed since it sent the Provisioning Enrollment inform described in Table 13-3, following KDC authentication, is equal to the value of cabhPsDevProvisioningTimer AND the PS has not been triggered to download a PS Configuration File, then the PS MUST set the value of cabhPsDevProvState to pass(1) and continue the provisioning process for SNMP Provisioning Mode as described in 13.4. If the PS is operating in SNMP Provisioning Mode and is triggered to download a PS Configuration File when the amount of time elapsed since it issued the Provisioning Enrollment inform is less than the value of cabhPsDevProvisioningTimer, then the PS MUST NOT set the value of cabhPsDevProvState to pass(1) until it successfully downloads and processes the specified PS Configuration File. If the PS is operating in SNMP Provisioning Mode and was triggered to download a PS Configuration File, the PS MUST set the value of cabhPsDevProvState to pass(1) when it successfully completes the download and processing of the PS Configuration File.

Once the PS, operating in SNMP Provisioning Mode (as indicated by the value of cabhPsDevProvMode), 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.

The PS MUST attempt to download and process the configuration file whose name and address are specified in cabhPsDevProvConfigFile when it receives an SNMP Set command for the cabhPsDevProvConfigFile object, if the following conditions are true:

- the PS is operating in SNMP Provisioning Mode;
- the cabhPsDevTodSyncStatus MIB object has a value of '1' (ToD access succeeded); 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.

7.4.4.2.3 Firewall configuration file trigger

The Firewall Configuration File download is triggered when the value used to SET the cabhSec2FwPolicyFileURL MIB object, by either the PS Configuration File or by a SNMP SET command, is different than the value of the cabhSec2FwPolicySuccessfulFileURL MIB. If the value used to SET the cabhSec2FwPolicyFileURL MIB object, by either the PS Configuration File or by a SNMP SET command, is the same as the value of the cabhSec2FwPolicySuccessfulFileURL MIB, the Firewall Configuration File download MUST NOT be triggered.

When a download has been triggered, the PS MUST use the prefix of the cabhSecFwPolicyFileURL MIB object value to determine whether to use TFTP (tftp://) or a TLS session (https://) as defined in clause 11 for Firewall Configuration File download.

7.4.4.2.4 Post-trigger operation

Once triggered, the PS MUST use an [RFC 1350] and [RFC 2348] compliant TFTP or [RFC 2616] HTTP client to download the configuration files.

A signalling 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 signalling mechanism.

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

Once triggered to download a configuration file, the PS element MUST continue to attempt to download the specified configuration file from the specified location until the configuration file is successfully downloaded and the hash successfully computed as described in 7.4.4.3, Configuration File Check and SNMP Provisioning Mode Authentication Requirements. The PS MUST use an adaptive timeout for TFTP and HTTPS based on binary exponential backoff as described below, if the first attempt is not successful, until the PS successfully receives the requested file from the server in the cable data network:

- Each retry is 2^n second(s) following the previous attempt, where the PS Configuration File Retry Counter or the Firewall Configuration File Retry Counter, n = [1, 2, 3, 4 or 5].
- n = 1 for the first retry, then is incremented by one for each subsequent attempt until n = 5.
- If the PS does not successfully acquire the requested PS Configuration File following the attempt with n = 5, n is to be reset to 1 and the PS is to restart the WAN-Man IP address acquisition process via DHCP.
- If the PS does not successfully acquire the requested Firewall Configuration File following the attempt with n = 5, n is to be reset to 1 and the PS is to continue normal operation, i.e., the PS is not to restart the WAN-Man IP address acquisition process.

The PS MUST exchange TFTP and HTTPS messages only through the PS WAN-Man Interface. The PS MUST reject any configuration file not received through the PS WAN-Man Interface.

When the download of the configuration file is complete and the configuration file is properly authenticated as described in 7.4.4.3, Configuration File Check and SNMP Provisioning Mode Authentication Requirements, the PS MUST process the TLVs contained within the file as defined below. See 7.4.4.4, Configuration File Processing and Status Reporting Requirements, for specifics of error handling and event generation while processing the configuration file.

The PS MUST use parameters extracted from the configuration file to set the managed objects in the PS database. This process is functionally equivalent to an SNMP SET operation, but it does not rely on the user or view-based access permissions. The PS MUST unconditionally update managed objects in the PS database corresponding to recognized OIDs.

The PS MUST translate Configuration File TLV-27 elements into a single SNMP PDU containing (n) MIB OID/instance and value components (SNMP varbinds) and translate TLV-28 elements into a single SNMP PDU containing (n) MIB OID/instance and value components (SNMP varbinds). In accordance with [RFC 3416], the single TLV-27 Configuration File-generated SNMP PDU will be treated "as if simultaneous", the single TLV-28 Configuration File generated SNMP PDU will be treated "as if simultaneous" and the PS MUST behave consistently, regardless of the order in which TLV-27 or TLV-28 elements appear in the Configuration File or SNMP PDUs. The single configuration file-generated SNMP PDU requirement is consistent with SNMP PDU packet behaviours received from an SNMP manager: SNMP PDU varbind order does not matter, and there is no defined MAX SNMP PDU limit. Once a single SNMP PDU is constructed, the PS processes

the SNMP PDU and determines the PS configuration acceptance/rejection based on the rules for configuration file processing, described in 7.4.4.4, Configuration File Processing and Status Reporting Requirements. In processing the SNMP PDU, the PS MUST support CreateAndGo for row creation.

The PS MUST update the size of the PS Configuration file in the MIB object cabhPsDevProvConfigFileSize.

The PS MUST update 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) from a PS Configuration File, in the MIB objects cabhPsDevProvConfigTLVProcessed and cabhPsDevConfigTLVRejected, respectively[1]. Configuration parameter Types 255 (End-of-Data Marker), 53 (PS MIC), 0 (Pad Configuration Setting), and Type and Length field pairs that encompass sub-TLVs do not specify values in Value fields intended to change PS configuration and thus MUST NOT be counted in the values of cabhPsDevProvConfigTLVProcessed and cabhPsDevConfigTLVRejected.

7.4.4.3 Configuration file check and SNMP provisioning mode authentication requirements

The algorithm used to authenticate the configuration file depends upon the provisioning mode in which the PS is operating (see 5.5, IPCable2Home Operational Models). The PS supports two provisioning modes: DHCP Provisioning Mode and SNMP Provisioning mode. Two methods of configuration file authentication are supported for DHCP Provisioning Mode, depending upon the information received in the 'siaddr' field of the DHCP ACK message.

The following subclauses describe the security algorithms and requirements needed to check the configuration file Hash based on the provisioning mode of the PS element. The PS element MUST support both security algorithms specified in 7.4.4.3.1, PS Configuration File Check for DHCP Provisioning Mode, and 7.4.4.3.2, PS Configuration File Authentication Algorithm for SNMP Provisioning Mode.

7.4.4.3.1 PS configuration file check for DHCP provisioning mode

When operating the DHCP Provisioning Mode, the PS will use a hash-based check of the configuration file, or it will authenticate the message in which the file is transferred, depending upon the configuration of the cable operator's provisioning system.

The PS MUST conduct the hash-based configuration file check described below:

1) When the configuration file Generator of the Provisioning System creates a new PS Configuration File or modifies an existing file, the Config File Generator will create a SHA-1 hash of the 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 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 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.

---

[1] Per these definitions, a TLV that does not successfully configure the PS is counted twice, once by each of cabhPsDevProvConfigTLVProcessed and cabhPsDevProvConfigTLVRejected. A TLV that successfully configures the PS is counted only by cabhPsDevProvConfigTLVProcessed.

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

7.4.4.3.2 PS configuration file authentication algorithm for SNMP provisioning mode

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

1) When the Config File Generator of the Provisioning System creates a new PS Configuration File or modifies an existing file, the 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. The PS updates its cabhPsDevProvConfigHash MIB object with the new value.
3) The NMS sends the Name and location of the PS Configuration File via SNMP SET. The PS updates its cabhPsDevProvConfigFile MIB object 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 contents of the PS Configuration File excluding the TLV 53 if it exists, the end of data marker and any padding that follows. If the computed hash and the value of the cabhPsDevProvConfigHash MIB object are the same, the PS Configuration File integrity is verified and the configuration file MUST be processed; otherwise, the file MUST be rejected.

7.4.4.3.3 Firewall configuration file check

The PS is required to use the Firewall Configuration File check on the Firewall Configuration File as described in this clause if the file is provided in SNMP Provisioning Mode or DHCP Provisioning Mode without the use of HTTPS/TLS as defined in 11.9, PS Configuration File Security in DHCP Provisioning Mode.

If the Firewall Configuration File was downloaded without the use of HTTP/TLS, the PS MUST follow the procedure defined in steps 1 through 5 below to check the integrity of the Firewall Configuration File:

1) The Firewall Configuration 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 cabhSec2FwPolicyFileHash MIB object via a Type 28 TLV in the PS Configuration File;
   b) sends an SNMP Set command to update the cabhSec2FwPolicyHash MIB object.
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 cabhSec2FwPolicyFileURL MIB object via a Type 28 TLV in the PS Configuration File;
   b) sends an SNMP Set command to update the cabhSec2FwPolicyURL MIB object.
4) If the cabhSecFwPolicyFileOperStatus is not inProgress(1) and the value used to SET the cabhSec2FwPolicyFileURL MIB object is different than the value of the

cabhSec2FwPolicySuccessfulFileURL MIB, then the PS element MUST immediately download the named file from the configured server.

5) The PS MUST compute a SHA-1 hash over the entire contents of the Firewall Configuration File and compare the computed hash to the hash represented by the value of the cabhSec2FwPolicyFileHash MIB object. If the computed hash and the value of the cabhSec2FwPolicyFileHash MIB object are the same, the integrity of the Firewall Configuration File is verified and the PS MUST use Firewall Configuration File to configure the firewall, otherwise the PS MUST reject the file.

7.4.4.4 Configuration file processing and status reporting requirements

The PS MUST report configuration file download status and error conditions using the Event Reporting process described in 6.3.3.2, CMP Event Reporting Function.

Table 7-15 identifies success and failure modes that might be encountered with PS Configuration File download and processing, and the action that the PS MUST take when it detects these modes.

Table 7-15/J.192 – Configuration file processing conditions

Condition	Action
TFTP failed – Get Request sent, no response received	Report an event (Event ID 68000500) and retry TFTP.
HTTPS failed – Get Request sent, no response received or failed to connect with HTTPS server.	Report an event (Event ID 68002000) and retry HTTPS.
TFTP failed – configuration file not found	Report an event (Event ID 68000600) and retry TFTP.
HTTPS failed – configuration file download attempt failed and maximum number of retries not exceeded.	Report an event (Event ID 68003000) and retry HTTPS.
TFTP failed – out of order packets	Report an event (Event ID 68000700) and retry TFTP.
TFTP download failed – configuration file download attempt failed and maximum allowable number of retries have been done.	Report an event (Event ID 68000900) and reset.
HTTPS failed – configuration file download attempt failed and maximum allowable number of retries have been done.	Report an event (Event ID 68003100) and reset.
Configuration file download successful	Report an event (Event ID 68001000 download was done using TFTP (TLS was not used) or Event ID 68003200 if download was done using HTTPS/TLS) and begin configuration file check or authentication.
Configuration file fails authentication check	Report an event (Event ID 68000800) and reset. Do not attempt to process the file.
Configuration File is too large	Report an event (Event ID 73040102) and reset. Do not attempt to process the file.
No End Of Data marker	Report an event (Event ID 7340102) and reset. Do not attempt to process the file.

Table 7-15/J.192 – Configuration file processing conditions

Condition	Action
Duplicate TLV-27 or TLV-28 OID	Report an event (Event ID 73040102), reject the configuration file, and reset. Preserve all object values that existed before the attempt to process this bad configuration file. The PS is not required to restore MIB objects to the values they were assigned before the attempt to process the configuration file if the single SNMP PDU created from TLV-27 parameters has been set. Refer to the Post-trigger Operation clause.
Duplicate TLV-9, TLV-21, TLV-32, TLV-33 or duplicate Sub-TLV in a single TLV-34, TLV-38, TLV-43.	Report an event (73040102), reject the configuration file, and reset. Preserve all objects that existed before the attempt to process this bad configuration file.
Recognized Type but bad Value or valid TLV-27 or TLV-28 OID but bad MIB value	Report an event (Event ID 73040102), reject the configuration file, and reset. Preserve all object values that existed before the attempt to process this bad configuration file. The PS is not required to restore MIB objects to the values they were assigned before the attempt to process the configuration file if the single SNMP PDU created from TLV-27 parameters has been set. Refer to the Post-trigger Operation clause.
An unrecognized SNMP OID is encountered	Disregard the subject TLV and report an event (Event ID 73040100). Continue to process the file.
Type field is not valid for PS	Disregard the subject TLV and report an event (Event ID 73040101). Continue to process the file.

Refer to Annex B for a list of events, including those listed in Table 7-15, and information about how events are reported.

7.4.4.4.1 Unsuccessful configuration file download attempt – TFTP or HTTPS retries permitted

If the PS Configuration File Retry Counter is less than 5 and the TFTP or HTTPS Get Request times out, the PS Configuration File is not found on the server, or the TFTP or HTTPS Get failed due to out of order packets, the PS MUST initiate operation of the CDS and CNP functions, report the appropriate event, and retry the attempt to download the PS Configuration File, in accordance with the retry algorithm described in 7.4.4.2.4, Post-trigger Operation.

If the Firewall Configuration File Retry Counter is less than 5 and the TFTP or HTTP Get Request times out, the Firewall Configuration File is not found on the server, or the TFTP or HTTP Get failed due to out of order packets, the PS MUST continue normal operations, report the appropriate event, and retry the attempt to download the Firewall Configuration File, in accordance with the retry algorithm described in 7.4.4.2.4, Post-trigger Operation.

7.4.4.4.2 Unsuccessful configuration file download attempt – TFTP or HTTPS retries exhausted

If the PS Configuration File Retry Counter is equal to 5 and the PS has not successfully downloaded the PS Configuration File, the PS MUST report the event identified in Table 7-15, Configuration File Processing Conditions, for indicating failure of the PS Configuration File download process and release its PS WAN-Man IP address in accordance with [RFC 2131], and restart the WAN-Man IP address acquisition process via DHCP.

If the Firewall Configuration File Retry Counter is equal to 5 and the PS has not successfully downloaded the PS Configuration File, the PS MUST report the event identified in Table 7-15, Configuration File Processing Conditions, for indicating failure of the Firewall Configuration File download process and continue normal operations. If the Firewall Configuration File is not successfully downloaded, the PS MUST function as it did prior to the failed Firewall Configuration File download attempt.

7.4.4.4.3 Successful PS configuration file download

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

If the PS successfully downloads the PS Configuration File, the PS MUST reset the PS Configuration File Retry Counter to zero and report the event identified for 'Failure Mode' TFTP Download Successful in Table 7-15, Configuration File Processing Conditions.

7.4.4.4.4 Unsuccessful PS configuration file download

If the PS Configuration File fails the Configuration File Check as specified in 7.4.4.3, Configuration File Check and SNMP Provisioning Mode Authentication Requirements, or in 11.9, PS Configuration File Security in DHCP Provisioning Mode, the PS MUST stop the provisioning process, reject the PS Configuration File, report the appropriate event, and restart the WAN-Man IP acquisition process via DHCP.

If the PS Configuration File contains no End-of-Data TLV (TLV-255), no PS MIC TLV (TLV-53), or is too large to process, the PS MUST stop the provisioning process, reject the PS Configuration File, report the appropriate event, and restart the WAN-Man IP address acquisition process via DHCP.

If the PS Configuration File contains duplicate TLV-27 or TLV-28 elements (duplicate means two or more SNMP MIB objects have an identical object identifier (OID)), the PS MUST stop the provisioning process, reject the PS Configuration File, report the appropriate event, and restart the WAN-Man IP address acquisition process via DHCP.

If the PS Configuration File contains a recognized Type field but bad Value field or a valid TLV-27 or TLV-28 OID with a bad MIB value, the PS MUST stop the provisioning process, reject the PS Configuration File, report the appropriate event, and restart the WAN-Man IP address acquisition process via DHCP.

If the PS Configuration File contains an unrecognized Type field or a TLV-27 or TLV-28 element with an unrecognized OID, the PS MUST ignore that TLV, report the appropriate event, and continue processing the PS Configuration File.

If the PS completes the processing of the single SNMP PDU created from the TLV-27 parameter then discovers duplicate TLV-28 elements, or TLV-28 elements with bad Value, the PS is not required to restore the MIB objects changed by the TLV-27 parameter back to their previous values, before rejecting the configuration file, reporting the event, and resetting the PS.

7.4.4.4.5 Successful firewall configuration file download

Successful download of the Firewall Configuration File is defined as complete and correct reception of the file by the PS element within the TFTP or HTTPS timeout period and error-free file validation as defined by the integrity check procedure described in 7.4.4.3, Configuration File Check and SNMP Provisioning Mode Authentication Requirements. After the PS successfully downloads the Firewall Configuration File, the PS MUST update the cabhSec2FwPolicySuccessfulFileURL MIB with the same value as the cabhSec2FwPolicyFileURL MIB.

If the PS successfully downloads the Firewall Configuration File, the PS MUST reset the Firewall Configuration File Retry Counter to zero and report Event ID 80013500 (see Table B.1, Defined Events for IPCable2Home). After the PS successfully downloads and processes the Firewall Configuration File, the firewall MUST function as configured by the downloaded file.

7.4.4.4.6 Unsuccessful firewall configuration file download

If the Firewall Configuration File fails the Configuration File Check as specified in 7.4.4.3, Configuration File Check and SNMP Provisioning Mode Authentication Requirements, the PS MUST continue normal operations, reject the Firewall Configuration File and report the appropriate event identified in Table B.1, Defined Events for IPCable2Home.

If the Firewall Configuration File contains duplicate TLV-27 or TLV-28 elements (duplicate means two or more SNMP MIB objects have an identical object identifier (OID)), the PS MUST continue normal operations, reject the Firewall Configuration File and report the appropriate event identified in Table B.1, Defined Events for IPCable2Home.

If the Firewall Configuration File contains a recognized Type field but bad Value field or valid TLV-27 or TLV-28 OID with a bad MIB value, the PS MUST continue normal operations, reject the Firewall Configuration File and report the appropriate event identified in Table B.1, Defined Events for IPCable2Home.

If the Firewall Configuration File contains an unrecognized Type field or a TLV-28 element with an unrecognized OID, the PS MUST ignore that TLV, report the appropriate event identified in Table B.1, Defined Events for IPCable2Home, and continue processing the Firewall Configuration File.

If the download of the Firewall Configuration File fails for any reason, the firewall MUST function as configured prior to the failed download attempt.

7.5 PS function – Time of Day client

7.5.1 Time of Day client function goals

The goal of the Time of Day client function of the PS is to acquire the current time of day from the Time of Day server in the cable operator's network.

7.5.2 Time of Day client function system design guidelines

The guideline identified in Table 7-16 has guided specification of the capabilities defined for the PS Time of Day Client function:

Table 7-16/J.192 – Time of Day client system design guidelines

Number	Guidelines
ToD 1	Provide a mechanism by which the PS can achieve time synchronization with the Headend network

7.5.3 Time of Day client function system description

The Portal Services element makes use of an [RFC 868] compliant Time of Day client, in order to achieve time synchronization with a time server on in the cable operator's data network. Time synchronization is essential for PS security functions as well as event messaging.

When the CDC DHCP client requests an IP Address – from the cable operator's data network DHCP server – for the WAN-Man interface, the DHCP client will receive the IP address of the cable operator's data network time server within DHCP option 4.

Once the WAN-Man IP stack begins use of the IP address it received from the DHCP server in the cable operator's data network, the PS will send an [RFC 868] time query to the time server. If the time server responds with a valid response, the PS uses the UTC time acquired from the time server and a time offset to establish current time of day. The time offset provides adjustment from UTC for the time zone in which the PS resides.

One source of time offset information is the DHCP server, which can include the information in DHCP option 2 (Time Offset option) of the DHCP OFFER and DHCP ACK messages. Alternatively, the cable operator can provision the PS with a time offset by writing to the CDP MIB (see E.2) object cabhCdpSnmpSetTimeOffset. Another CDP MIB object, cabhCdpTimeOffsetSelection, enables the cable operator to configure the PS to use either the time offset provided in DHCP option 2 or the time offset from cabhCdpSnmpSetTimeOffset. Another potential time offset is the adjustment for Daylight Saving Time for areas that observe it. The CDP MIB object cabhCdpDaylightSavingTimeEnable allows the cable operator to configure the PS to add 1 hour to the time offset to adjust for Daylight Saving Time. Refer to (E.2) for details.

Once it acquires the UTC time from the time server and the appropriate time offset value (determined by the value of cabhCdpTimeOffsetSelection), the PS will combine them, add the Daylight Saving Time adjustment when applicable (if cabhCdpDaylightSavingTimeEnable is set to enabled(1)), update the value as the current time in the IPCable2Home MIB object cabhPsDevDateTime, and will begin using this time of day for event message time stamps and security functions.

7.5.4 Time of Day client function requirements

The Portal Services element MUST implement a Time of Day Client.

The Portal Services Time of Day Client MUST comply with the Time of Day Protocol [RFC 868] and make use of the UDP Protocol only.

Upon reset, before the PS synchronizes with a Time of Day server, the Portal Services Element MUST initialize its time to 00:00.0 (midnight) GMT, January 1, 1970.

If the PS receives DHCP option 4 (Time Server Option) in the DHCP ACK, the PS MUST save the IP address of the Time Server from which the PS accepted a response as the value of cabhPsDevTimeServerAddr.

An Embedded PS 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 or overwriting the system time originally initialized to epoch time (00:00.0 (midnight) GMT, January 1, 1970).

If the value of cabhPsDevTodSyncStatus is true(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 its WAN-Man Interface.

The PS MUST use the value of cabhPsDevDateTime for any functions requiring time of day, and which need only be accurate to the nearest second.

The IPCable2Home Time of Day acquisition process is comprised of two phases: the Initial Time of Day Synchronization Attempt (Initial Attempt) phase and the Time of Day Synchronization Retry (Retry) phase. If the PS is successful synchronizing Time of Day with the Time of Day server during the Initial Attempt phase, it does not initiate the Retry phase. The PS is required to enter the Initial Attempt phase and attempt synchronization with a Time of Day server upon receipt of a DHCP ACK message, if the value of cabhPsDevTodSyncStatus is false(2). Clause 7.5.4.1 describes the required Initial Attempt behaviour for the PS. Clause 7.5.4.2 describes the required behaviour for the PS if it is required to initiate the ToD Retry phase.

7.5.4.1 Initial Time of Day synchronization attempt requirements

If the PS is operating in DHCP Provisioning Mode or SNMP Provisioning Mode (cabhPsDevProvMode = dhcpmode(1) or snmpmode(2)), the PS MUST attempt to synchronize with a Time of Day server whose address was passed to the PS in DHCP option 4 of the DHCP ACK message, in accordance with [RFC 868]. A PS operating in Dormant CableHome Mode is not required to attempt to synchronize with a Time of Day server.

If the PS is not successful in synchronizing with a Time of Day (ToD) server on its first attempt, the PS MUST attempt to synchronize with the next ToD server in the order listed in DHCP option 4, until it successfully synchronizes with a server, OR until it makes an unsuccessful attempt with each listed ToD server. The PS MUST report the appropriate event (see Table B.1) for each unsuccessful attempt at synchronizing with a Time of Day server. A Time of Day server synchronization attempt is a single attempted access as described in [RFC 868], initiated by the PS, of port 37 on a Time of Day server. An unsuccessful attempt to synchronize with a Time of Day server is one which results in the PS NOT receiving valid time information from the Time of Day server, or, if an attempt to resolve the Time of Day server's network (IP) address results in the PS NOT acquiring the server's network address.

If the PS successfully synchronizes with a Time of Day server, the PS MUST do the following:

- set the value of cabhPsDevTodSyncStatus to true(1);
- set the value of cabhCdpServerTimeOffset with the value of DHCP option 2 (Time Offset) from the DHCP ACK message if the value of cabhCdpTimeOffsetSelection is useDhcpOption2(1), OR with the value of cabhCdpSnmpSetTimeOffset if the value of cabhCdpTimeOffsetSelection is useSnmpSetOffset(2);
- set the value of cabhPsDevDateTime equal to the UTC time acquired from the time server, plus the time offset from DHCP Option 2 in the DHCP ACK message OR from the value of cabhCdpSnmpSetTimeOffset according to the value of cabhCdpTimeOffsetSelection, plus 1 hour for Daylight Saving Time adjustment during Daylight Saving Time if the value of cabhCdpDaylightSavingTimeEnable is enabled(1);
- set the value of cabhPsDevTimeServerAddr with the IP address of the Time of Day server with which the PS synchronized its time;
- if the PS CDS function has current LAN IP address leases, update cabhCdpLanAddrCreateTime with the value of cabhPsDevDateTime and set the value of cabhCdpLanAddrExpire time equal to cabhCdpLanAddrCreateTime, plus the value of cabhCdpServerLeaseTime, for each active lease;
- continue with the Provisioning Process as defined in clause 13.

If an embedded PS operating in DHCP Provisioning Mode is not successful in synchronizing with any of the Time of Day servers listed in DHCP option 4 of the DHCP ACK message, after attempting to do so once with each listed ToD server, the embedded PS MUST attempt to acquire system time from the cable modem. The embedded PS operating in SNMP Provisioning Mode is not required to attempt to acquire system time from the cable modem.

If the embedded PS operating in DHCP Provisioning Mode is not successful in synchronizing with any Time of Day servers on its first attempt with each AND is successful acquiring system time from the cable modem, the embedded PS MUST do the following:

- set the value of cabhPsDevTodSyncStatus to false(2);
- set the value of cabhPsDevDateTime to the cable modem's system time;

- if the embedded PS CDS function has current LAN IP address leases, update cabhCdpLanAddrCreateTime with the value of cabhPsDevDateTime (cable modem's time) and set the value of cabhCdpLanAddrExpire time equal to cabhCdpLanAddrCreateTime, plus the value of cabhCdpServerLeaseTime, for each active lease;
- initiate the Time of Day Synchronization Retry process defined in 7.5.4.2 AND continue with the Provisioning Process defined in clause 13.

An embedded PS operating in DHCP Provisioning Mode that is not successful in synchronizing with any Time of Day server on its first attempt with each and is not successful in acquiring system time from the cable modem MUST do the following:

- set the value of cabhPsDevTodSyncStatus to false(2);
- set the value of cabhPsDevDateTime to epoch time (00:00.0 (midnight) GMT, January 1, 1970);
- if its CDS function has current LAN IP address leases, update cabhCdpLanAddrCreateTime with the value of cabhPsDevDateTime (epoch time) and set the value of cabhCdpLanAddrExpire time equal to cabhCdpLanAddrCreateTime plus the value of cabhCdpServerLeaseTime, for each active lease;
- initiate the Time of Day Synchronization Retry process defined in 7.5.4.2 AND continue with the Provisioning Process defined in clause 13;
- report the appropriate event (see Table B.1) for each unsuccessful attempt to synchronize with the Time of Day server.

A standalone PS operating in DHCP Provisioning Mode that is not successful in synchronizing with any Time of Day server on its first attempt with each MUST do the following:

- set the value of cabhPsDevTodSyncStatus to false(2);
- set the value of cabhPsDevDateTime to epoch time (00:00.0 (midnight) GMT, January 1, 1970);
- if its CDS function has current LAN IP address leases, update cabhCdpLanAddrCreateTime with the value of cabhPsDevDateTime (epoch time), and set the value of cabhCdpLanAddrExpire time equal to cabhCdpLanAddrCreateTime, plus the value of cabhCdpServerLeaseTime, for each active lease;
- initiate the Time of Day Synchronization Retry process defined in 7.5.4.2 AND continue with the Provisioning Process defined in clause 13;
- report the appropriate event (see Table B.1) for each unsuccessful attempt to synchronize with the Time of Day server.

A PS operating in SNMP Provisioning Mode that is not successful in synchronizing with any Time of Day server listed in DHCP option 4 of the DHCP ACK message on its first attempt with each MUST initiate the Time of Day Synchronization Retry process defined in 7.5.4.2. A PS operating in SNMP Provisioning Mode that is not successful in synchronizing with any Time of Day server MUST NOT continue with the Provisioning Process defined in clause 13. The requirement for the PS to report an event for each unsuccessful Time of Day server synchronization attempt applies to the PS operating in SNMP provisioning mode.

7.5.4.2 Time of Day synchronization retry requirements

If a PS operating in DHCP Provisioning Mode is not successful in synchronizing with any Time of Day server listed in option 4 of the DHCP ACK message AND cabhPsDevTodSyncStatus = false(2), the PS MUST continue to attempt to synchronize with the Time of Day servers listed in option 4 of the DHCP ACK message until it is successful and report the appropriate event (see Table B.1) for each unsuccessful attempt.

While the value of cabhPsDevTodSyncStatus = false(2), a PS operating in SNMP Provisioning Mode MUST continue to attempt to synchronize with each of the Time of Day servers listed in option 4 of the DHCP ACK message, for a total of six attempts (initial attempt plus five retries) and report the appropriate event (see Table B.1) for each unsuccessful attempt.

The PS Time of Day client MUST NOT exceed more than 3 ToD requests per Time of Day Server in any 5-minute period. At a minimum, a PS attempting to synchronize with a ToD server MUST issue at least 1 ToD request per 5-minute period.

A PS operating in SNMP Provisioning Mode that is not successful in synchronizing with any Time of Day server after attempting six times with each ToD server listed in option 4 of the DHCP ACK message MUST do the following:

- set cabhPsDevTodSyncStatus = false(2);
- log Event ID 68000403 (refer to Annex B, Table B.1) according to the configured Priority for the event and following the procedure defined in 6.3.3.2, CMP Event Reporting Function;
- restart the provisioning process beginning with issuing DHCP DISCOVER;
- report the appropriate event (see Table B.1) for each unsuccessful attempt to synchronize with the Time of Day server.

7.6 BP function – DHCP client

7.6.1 BP DHCP client function goals

The goal of the BP DHCP client function is to acquire an IP address lease and configuration parameters for the BP from the system DHCP server.

7.6.2 BP DHCP client function system design guidelines

The guideline listed in Table 7-17 has guided specification of the BP DHCP Client function:

Table 7-17/J.192 – BP DHCP client function system design guidelines

Number	Guidelines
BP DHC 1	Provide a means by which the BP can acquire a network address lease and configuration information.

7.6.3 BP DHCP client function system description

The DHCP Client function of the BP is responsible for acquiring an IP address lease from a system DHCP server. The server could be the CDS Function of the CDP sub-element of the PS or it could be a DHCP server in the cable operator's data network, depending upon how the PS packet handling mode is configured. The BP DHCP Client function also acquires configuration information passed in DHCP Option fields from the system DHCP server.

7.6.4 BP DHCP client function requirements

The BP MUST implement a DHCP client function in accordance with the Client requirements of [RFC 2131].

Upon reset, the BP MUST issue a DHCP DISCOVER broadcast message to acquire an IP address lease.

The BP MUST support the DHCP Options and sub-options indicated as mandatory (M) in Table 7-18.

The BP MUST include the following DHCP option codes, in each DHCP DISCOVER and DHCP REQUEST message it sends:

- DHCP option code 55 Parameter Request List;
- DHCP option code 60 Vendor Class Identifier, with the string "CableHome1.1BP" (with no spaces and without quotation marks;
- DHCP option code 255 End.

Table 7-18/J.192 – BP DHCP client required DHCP options

Option number	Option function	Support (M)andatory or (O)ptional	Factory default value
0	Pad	–	N/A
255	End	M	N/A
1	Subnet Mask	M	N/A
2	Time Offset	O	0
3	Router Option	M	N/A
6	Domain Name Server	M	N/A
7	Log Server	M	N/A
12	Host Name	O	N/A
15	Domain Name	M	Null String
23	Default Time-to-live	M	N/A
26	Interface MTU	M	N/A
43	Vendor Specific Information	M	Vendor Selected
50	Requested IP Address	M	null value or vendor selected
51	IP Address Lease Time	M	N/A
54	Server Identifier	M	N/A
55	Parameter Request List	M	N/A
60	Vendor Class Identifier	M	"CableHome1.1BP"
61	Client-identifier	O	N/A

8 Packet handling and address translation

8.1 Introduction/Overview

8.1.1 Goals

The key goals which drive the packet handling capabilities include:

- Provide cable friendly address translation functionality, enabling cable operator visibility and manageability of home devices while preserving cable-based sourced-based routing architectures.
- Prevent unnecessary traffic on the cable and home network.
- Conservation of globally routable public IP addresses as well as cable network private management addresses.
- Facilitate in-home IP traffic routing by assigning network addresses to LAN IP Devices such that they reside on the same logical subnetwork.

8.1.2 Assumptions

- It is assumed that when cable operator provisioning servers provide multiple globally routable IP addresses to customer devices in a home, these addresses will not necessarily reside on the same subnet.
- Changing Internet service providers is assumed to occur relatively infrequently, occurring at a rate similar to a household changing its primary long distance carrier.

8.2 Architecture

This clause describes the key concepts behind the IPCable2Home packet handling and address translation functionality.

8.3 PS logical element – IPCable2Home Address Portal (CAP)

The IPCable2Home Address Portal (CAP) is a logical sub-element of the Portal Services logical element. Its functions are to route traffic between the LAN and the WAN, route LAN-to-LAN traffic, and to perform address and port translation functions.

8.3.1 CAP goals

The goals of the CAP are listed below and in 8.1.1:

- Route IP packets between LAN IP Devices, and between LAN IP Devices and the Portal Services' default gateway on the WAN.
- Provide Network and Port Address Translation (NAPT) capability for mapping between a single global IP address on the PS WAN Interface and one or more private IP addresses in the LAN.
- Provide Network Address Translation (NAT) capability for 1-to-1 mapping between global IP addresses on the PS WAN Interface and private IP addresses on the LAN.
- Keep traffic between LAN IP Devices on the LAN and do not permit it to traverse the WAN.

8.3.2 CAP system design guidelines

The system design guidelines listed in Table 8-1 have guided specification of the IPCable2Home Address Portal functionality.

Table 8-1/J.192 – CAP system design guidelines

Number	Guidelines
CAP 1	Addressing mechanisms will be operator controlled, and will provide operator knowledge of and accessibility to IPCable2Home devices.
CAP 2	Addressing will do nothing that will compromise current cable network routing architectures (for example source based routing, MPLS).
CAP 3	Traffic management mechanisms will insulate the cable network from traffic generated by in house peer-to-peer communications.
CAP 4	IP Addresses will be conserved when possible (both globally routable addresses and private cable network management addresses).
CAP 5	{informative text:The CAP will allow UPnP LAN devices to configure network address translation mappings, but to the extent that this does not conflict with the operator's policies.}

8.3.3 CAP system description

Address translation and packet handling functionality is provided by the functional entity known as the IPCable2Home Addressing Portal (CAP). The CAP encompasses the following address translation and packet forwarding elements:

- IPCable2Home Address Translation (CAT);
- IPCable2Home Passthrough Function;
- Upstream Selective Forwarding Switch (USFS).

As shown in Figure 8-1, the CAT function provides a mechanism to interconnect the WAN-Data address realm and LAN-Trans address realm (via address translation), while Passthrough provides a mechanism to interconnect the WAN-Data address realm and the LAN-Pass address realm (via bridging). The CAT function is compliant with Traditional Network Address Translation (NAT) [RFC 3022] section 2. As with Traditional NAT, there are two variations of CAT, referred to as IPCable2Home Network Address Translation (C-NAT) Transparent Routing and IPCable2Home Network Address and Port Translation (C-NAPT) Transparent Routing. C-NAT Transparent Routing is the IPCable2Home compliant version of Basic NAT [RFC 3022] section 2.1 and C-NAPT Transparent Routing is the IPCable2Home compliant version of NAPT [RFC 3022] section 2.2.

Per [RFC 3022], C-NAT transparent routing is "a method by which IP addresses are mapped from one group to another, transparent to end users," and C-NAPT transparent routing "is a method by which many network addresses and their TCP/UDP (Transmission Control Protocol/User Datagram Protocol) ports are translated into a single network address and its TCP/UDP ports." Also, per [RFC 3022], the purpose of C-NAT and C-NAPT functionality is to "provide a mechanism to connect a realm with private addresses to an external realm with globally unique registered addresses."

The IPCable2Home Passthrough function is an IPCable2Home specified bridging process that interconnects the WAN-Data Address Realm and the LAN-Pass Address Realm without address translation.

The Upstream Selective Forwarding Switch (USFS) defines a function within the CAP with the capability of confining home networking traffic to the home network, even when home networking devices generating this traffic reside on different logical IP subnets. Specifically, this function forwards traffic sourced from an IP address in one of the LAN Address realms, destined to IP addresses in one of the LAN Address realms, directly to its destination. This direct forwarding functionality prevents the traffic from traversing the HFC network, and interconnects the LAN-Trans and LAN-Pass Address Realms.

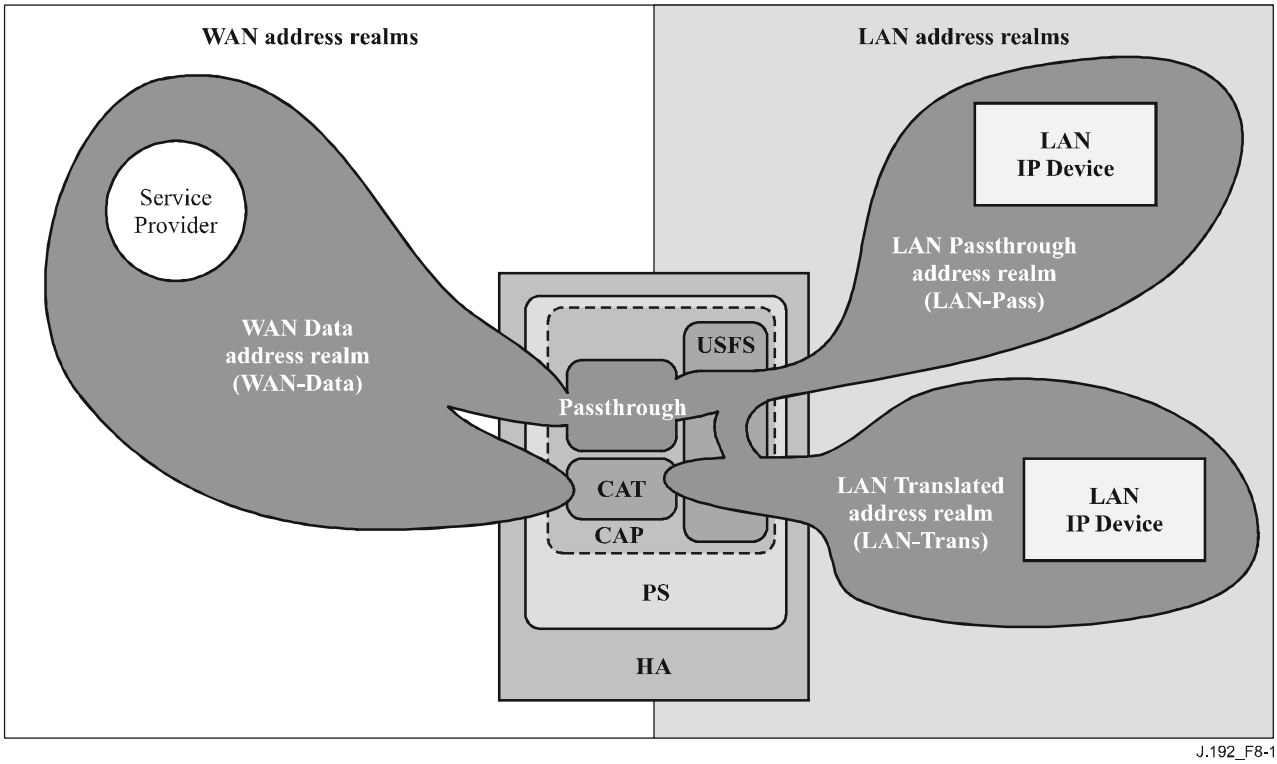


Figure 8-1/J.192 – IPCable2Home Address Portal (CAP) functions

Throughout this Recommendation, the terms Address Binding, Address Unbinding, Address Translation, and Session are used as defined in [RFC 2663]. In addition, IPCable2Home defines the term Mapping as the information required to perform C-NAT Transparent Routing and C-NAPT Transparent Routing.

In particular, a C-NAT Mapping is defined as a tuple of the form (WAN-Data IP address, LAN-Trans IP address) providing a one-to-one mapping between WAN-Data addresses and LAN-Trans addresses. Similarly, a C-NAPT Mapping is defined as a tuple of the form (WAN-Data IP address and TCP/UDP port, LAN-Trans IP address and TCP/UDP port) providing a one-to-many mapping between a single WAN-Data address and multiple LAN-Trans addresses. For ICMP traffic (such as ping), an ICMP identifier is used in place of the TCP/UDP port number.

LAN-to-WAN traffic is defined as packets sourced by LAN IP Devices destined to devices on the WAN side of the PS. WAN-to-LAN traffic is defined packets sourced by WAN hosts destined to LAN IP devices. LAN-to-LAN traffic is defined as packets sourced by LAN IP Devices destined to LAN IP Devices on the same or different subnet.

8.3.3.1 Packet handling modes

The Portal Services element is configurable, via the cabhCapPrimaryMode MIB object, to operate in one of three Primary Packet-handling Modes when handling LAN-to-WAN and WAN-to-LAN traffic: Passthrough Mode, C-NAT Transparent Routing Mode, and C-NAPT Transparent Routing Mode. Further, the C-NAT or C-NAPT primary modes may also operate in a Mixed Mode described below.

In Passthrough mode, the CAP acts as a transparent bridge [ISO/IEC 10038] between the WAN-Data realm and LAN-Pass realm. In Passthrough mode, forwarding decisions are made primarily at OSI Layer 2 (data link layer). In this mode, the CAP does not perform any C-NAT or C-NAPT Transparent Routing functions. The PS bridging traffic for LAN-Pass IP devices is required to pass all OSI Layer 2 frames that a DOCSIS compliant cable modem is required to pass, including SNAP [ISO/IEC 8802-2] and DIX Ethernet Version 2.0 frames.

The CAP supports OSI Layer 3 (network layer) forwarding in both the C-NAT Transparent Routing Mode and the C-NAPT Transparent Routing Mode, described below.

In C-NAT Mode, the PS element (CDC) acquires one or more IP addresses used for WAN-Data traffic during the PS boot process. After acquisition, via DHCP, these IP addresses are used as the WAN-Data IP address portion of Dynamically created C-NAT Mapping tuples. These WAN IP addresses make up a pool of addresses available for Dynamically created C-NAT Mappings. If an available IP address exists in the WAN-Data IP address pool, the CAP creates a Dynamic C-NAT Mapping when it first sees LAN-to-WAN IP traffic that does 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 IPCable2Home 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. When the PS is operating in NAT primary packet handing mode (cabhCapPrimaryMode = nat(2)), the CAP will enter the value 0 into the WAN and LAN port number entries of the CAP Mapping Table. The CAP will also enter the value 0 into the WAN and LAN port number entries of the CAP Mapping Table for provisioned static port forwarding entries of the CAP Mapping Table when the PS is operating in NAPT primary packet handling mode (cabhCapPrimaryMode = napt(1)). For the case of a static port forwarding entry provisioned in the CAP Mapping Table for a PS operating in NAPT primary packet handling mode, the 0-value port number entry will serve two purposes:

1) indicate to the CAP that the port numbers are not to be translated, i.e., that the ports are "wildcarded"; and

2) indicate to anyone reading the CAP Mapping Table that this static port mapping is effectively a C-NAT mapping, thereby providing a distinction between static port forwarding entries (C-NAT mappings) (port number 0) and C-NAPT Mappings (non-zero port number).

   Refer to 8.3.3.2, Static Port Forwarding with Port Wild Cards, for more information about static port forwarding operation of the CAP.

Dynamic C-NAT Mappings for UDP traffic are destroyed when an inactivity timeout period, cabhCapUdpTimeWait, expires. Dynamic C-NAT Mappings for TCP traffic are destroyed when an inactivity timeout period, cabhCapTcpTimeWait, expires or a TCP session terminates. Dynamic C-NAT Mappings for ICMP traffic are destroyed when an inactivity timeout period, cabhCapIcmpTimeWait, expires. In addition, Static C-NAT Mappings may be created or destroyed when the NMS system writes to or deletes from the cabhCapMappingTable MIB table.

In C-NAPT Mode (the factory default mode for the system) the PS element (CDC) acquires one IP address, used for WAN-Data traffic. After acquisition, via DHCP, this IP address is used as the WAN-Data IP address portion of Dynamically created C-NAPT Mapping tuples. If the WAN-Data IP address has been acquired, Dynamic C-NAPT Mappings are created when the CAP first sees LAN-to-WAN IP traffic that does not have an existing Mapping. If the WAN-Data IP address has

not been acquired (i.e., does not have an active DHCP lease), the Dynamic C-NAPT Mapping can not be created, and this traffic is dropped, and a standard event is generated (see Annex B).

Dynamic C-NAPT Mappings for UDP traffic are destroyed when an inactivity timeout period, cabhCapUdpTimeWait, expires. Dynamic C-NAPT Mappings for TCP traffic are destroyed when an inactivity timeout period, cabhCapTcpTimeWait, expires or a TCP session terminates. Dynamic C-NAPT Mappings for ICMP traffic are destroyed when an inactivity timeout period, cabhCapIcmpTimeWait, expires. In addition, Static C-NAPT Mappings may be created or destroyed when the NMS system writes to or deletes from the cabhCapMappingTable MIB table.

Figure 8-2 shows a typical Dynamic C-NAPT Mapping process with a TCP packet. In this example, the PS is configured to operate in NAPT mode and already has obtained a WAN IP address, and the LAN IP Device has already obtained an IP in the LAN-Trans realm.

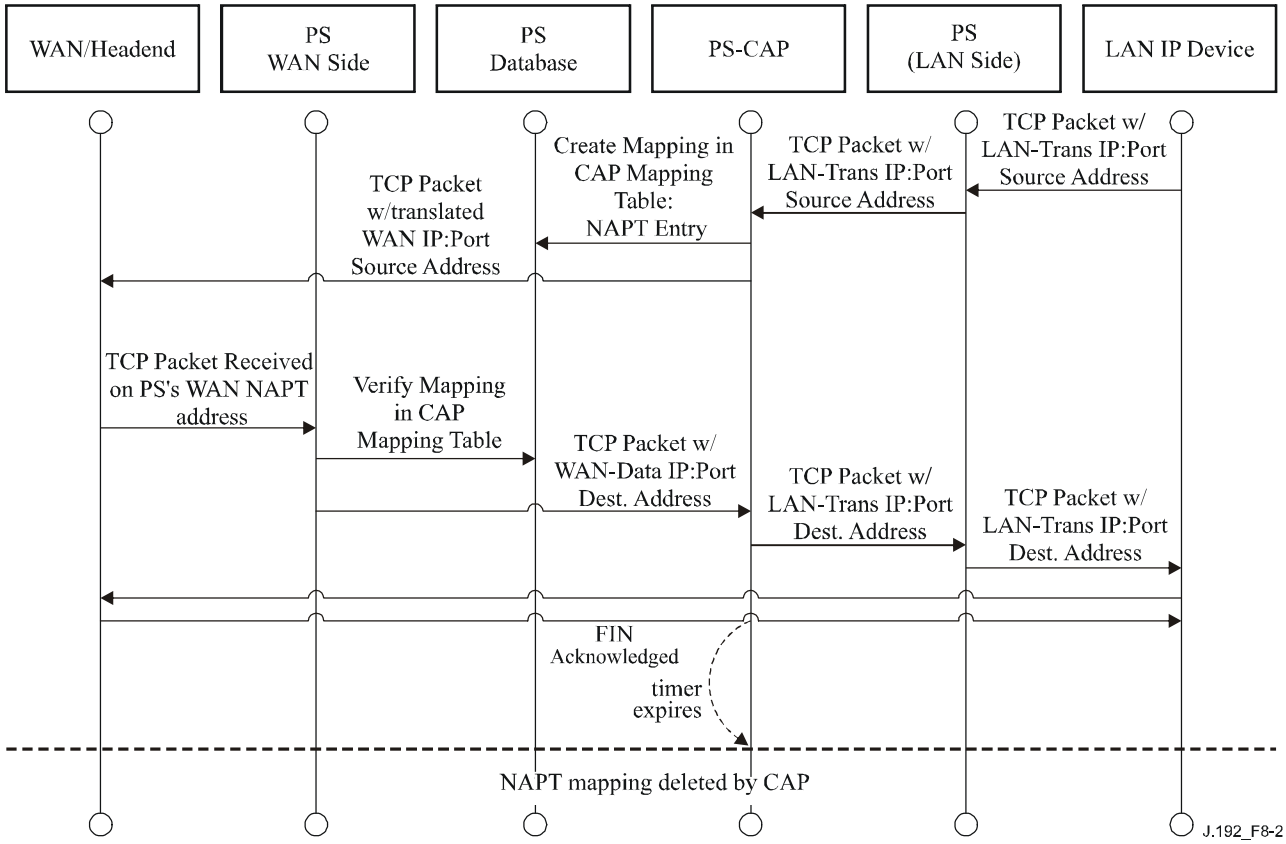


Figure 8-2/J.192 – PS configuration (CAP mapping table – NAPT) sequence diagram

It is also possible for the PS to operate in a Mixed Bridging/Routing Mode. In this case, the NMS sets the primary mode to C-NAT or C-NAPT Transparent Routing, and the NMS writes one or more MAC addresses belonging to LAN IP Devices, whose traffic is to be bridged, into the Passthrough Table (cabhCapPassthroughTable). In this Mixed Mode, the PS examines MAC addresses of received frames to determine whether to transparently bridge the frame or to perform any C-NAT or C-NAPT Transparent Routing functions at the IP layer. In the case of LAN-to-WAN traffic, the PS examines the source MAC address, and if that MAC address exists in the cabhCapPassthroughTable, the frame is transparently bridged to the WAN-Data interface. In the case of WAN-to-LAN traffic, the PS examines the destination MAC address, and if that MAC address exists in the cabhCapPassthroughTable, the frame is transparently bridged to the appropriate LAN interface. If the MAC address does not exist in the cabhCapPassthroughTable, the packet is processed by higher 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 [RFC 919], [RFC 922], [RFC 1812] and [RFC 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 (see 8.3.3.4) is applied in each of the three primary packet-handling modes, and regardless of whether or not Mixed mode is in use. USFS forwarding decisions will take precedence over other forwarding decisions that could potentially forward traffic from the LAN to the WAN.

8.3.3.2 CAP DMZ functionality (Static port forwarding with port wild cards)

When the PS is provisioned to operate in C-NAPT primary packet handling mode and a C-NAPT mapping is statically created with both WAN and LAN port numbers set to zero (i.e., when a DMZ entry has been created), then the CAP will handle inbound traffic in a special way. The CAP will forward all WAN-to-LAN traffic not associated with an existing C-NAPT session or an existing C-NAPT static mapping to the LAN IP address (DMZ IP address) specified in this special type of C-NAPT mapping (DMZ entry).

The CAP will process packets as follows:

1) Check all incoming WAN-to-LAN packets to see if they are associated with an existing session specified by a C-NAPT dynamic mapping. If this is the case, then the packet is translated as specified and is forwarded.
2) If not, then the CAP checks to see if there is a static C-NAPT mapping associated with the packet. If this is the case, then the packet is translated as specified and is forwarded.
3) If not, then the CAP checks to see if there is a static C-NAPT mapping for this WAN IP address with the port number set to 0. If this is the case, then the CAP translates the IP address to the LAN IP Address specified in this special C-NAPT static mapping. Note that C-NAPT does not translate the port in this case. After the address translation the packet is forwarded.

NOTE – If none of the above is true, the packet is dropped.

When a DMZ entry is created in the CAP for a LAN IP address that is dynamically assigned by the PS (CDS), the PS is required to create an IP address lease reservation for that address. This ensures that the IP address of the LAN device that is setup for the DMZ functionality does not change upon lease renewal. The PS can look up the DMZ IP address in the cabhCdpLanAddrTable. If a corresponding entry exists in this table with the value of cabhCdpLanAddrMethod equal to either dynamicActive(4) or dynamicInactive(3), then the PS is required to replace that row entry with one that represents an IP address lease reservation, that is, one with the value of cabhCdpLanAddrMethod equal to either psReservationActive(6) or psReservationInactive(5), respectively. If there is no entry corresponding to the DMZ IP address in the cabhCdpLanAddrTable, then the PS is not required to create an IP address lease reservation for that IP address. In this case, it is possible that the DMZ IP address is statically assigned to the LAN IP Device.

When a DMZ entry is removed from the cabhCapMappingTable for a LAN IP address (DMZ host), the PS is required to remove the corresponding IP address lease reservation that it had internally created (identified by cabhCdpLanAddrMethod=psReservationActive(6)) from the cabhCdpLanAddrTable as long as either the docsDevFilterIpTable or the cabhSec2FwLocalFilterIpTable does not have a corresponding firewall filter rule entry that requires it. (See 11.6.4.3.3, Factory Default Ruleset.)

8.3.3.3 Virtual Private Network (VPN) support in the CAP

The PS is required to implement a *VPN Passthrough* feature that allows IPSec [RFC 2401]-based VPN clients to exchange keys using Internet Key Exchange protocol [RFC 2409]. A single VPN client in the home at a time is supported, and that client is assumed to satisfy the following conditions:

- the LAN IP Device is in the LAN-Trans realm, i.e., it has a LAN-Trans IP address;
- the LAN IP Device uses IPSec as the VPN protocol;
- the LAN IP Device uses Internet Key Exchange to dynamically exchange encryption keys with the VPN server.

This Recommendation does not limit the number of VPN clients in the LAN-Pass realm (i.e., LAN IP Devices whose MAC address is in the PS Passthrough Table) that can simultaneously access VPN servers outside the home.

For the VPN client to operate properly a firewall policy file must be active in the PS that opens the proper ports for incoming (WAN-to-LAN) traffic, most notably port 500, for IKE traffic.

When keys are dynamically exchanged using IKE [RFC 2406] prior to initiation of an IPSec session, the CAP will translate network addresses as usual and will additionally associate port 500 as an inbound port for the private (LAN-Trans) IP address of the device that initiated the VPN connection. This will ensure that incoming IKE messages will be properly forwarded to the VPN client. IPSec sessions are defined in the CAP by the port used for inbound and outbound traffic, the port used for key exchange, the VPN server address and the VPN client address.

Even though the firewall has opened port 500, incoming traffic on port 500 will only be forwarded by the CAP after an IPSec session has been initiated by a client in the LAN-Trans address realm.

If a second VPN client in the home attempts to initiate an IPSec session with a different VPN server, the CAP will shift the ports used on the WAN-Data IP address for traffic and key exchange and translate these ports to the standard ports on the VPN Client IP address in the LAN-Trans realm. Additional VPN clients can be supported as well. However, the CAP does not support more than one VPN client in the home connecting to the same VPN server.

IPSec has three modes that can be used for VPNs. The PS is required to support Encapsulating Security Payload Tunneling mode [RFC 2406]. Support for Encapsulating Security Payload Transport mode [RFC 2406] and IP Authentication Header mode [RFC 2402] are not required.

8.3.3.4 Upstream Selective Forwarding Switch overview

In some cases, a LAN IP Device in the LAN-Pass address realm will reside on a different logical IP subnet than other LAN IP Devices connected to the same PS element. It is important to prevent the traffic between these LAN IP Devices from traversing the HFC network. Preventing this unwanted HFC traffic is the function that is provided by the Upstream Selective Forwarding Switch (USFS).

Specifically, the USFS routes traffic – that is sourced from within the home network and is destined to the home network – directly to its destination. LAN IP Device sourced traffic whose destination IP address is outside the LAN address realm is passed unaltered to the CAP bridging/routing functionality.

The USFS functionality makes use of the IP Address Translation Table (as defined in [RFC 2011]) within the PS element. This table, the [RFC 2011] ipNetToMediaTable, contains a list of MAC Addresses, their corresponding IP Addresses, and PS Interface Index numbers of the physical interfaces that these addresses are associated with. The USFS will refer to this table in order to make decisions about directing the flow of LAN-to-WAN traffic. In order to populate the ipNetToMediaTable, the PS learns IP and MAC addresses and their associations. For every associated physical interface, the PS learns all of the LAN-Trans and LAN-Pass IP addresses along

with their associated MAC bindings, and this learning can occur via a variety of methods. Vendor specific IP/MAC address learning methods may include: ARP snooping, traffic monitoring, and consulting CDP entries. Entries are purged from the ipNetToMediaTable after a reasonable inactivity timeout period has expired.

The USFS inspects all IP traffic received on PS LAN interfaces. If the destination IP address is found (via the ipNetToMediaTable) to reside on a PS LAN interface, the original frame's data-link destination address is changed from that of the default gateway address to that of the destination LAN IP Device, and the traffic is forwarded to the QoS Forwarding and Media Access (QFM) functionality (see 10.2, QoS Architecture) in the PS to be forwarded out on the proper PS LAN interface according to the packet priority. If a match to the destination IP address is not found in the ipNetToMediaTable, the packet is passed, in its original form, to the C-NAT/C-NAPT transparent routing function or the Passthrough bridging function (depending on the active packet handling mode).

8.3.3.5 MAC Access Control list

In order to help reduce or eliminate the potential for theft of service and other unauthorized access to the subscriber's LAN resources, IPCable2Home supports an Access Control list. This is a list of the hardware addresses for those LAN IP Devices for which the PS will forward traffic. The list is implemented as a MIB Table (cabhPsDevAccessControlTable) and consists of a list of physical addresses. Administrative control of the Access Control Table is provided by a scalar MIB object, cabhPsDevAccessControlEnable. Access control is enabled by interface type. An interface type is enabled for Access Control by setting the corresponding bit of the cabhPsDevAccessControlEnable object. When the bit corresponding to an interface type is set (1), the PS will forward traffic to or from any LAN IP Device through that interface type whose physical or hardware address is an element of the Access Control Table, but will not forward traffic through that interface type to or from a LAN IP Device whose physical address is not an element of the Access Control Table. When the bit corresponding to an interface type is not set, the PS will not use the Access Control Table when making a determination about whether to forward traffic to or from devices through that interface type. Refer to the description of the Access Control Table in E.4.

8.3.3.6 Multicast

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 WAN-originated IGMP traffic to the LAN to allow the advertisements to reach LAN IP Devices. A LAN IP Device will determine which multicast it wishes to join and will send a multicast "join" message. The multicast source will then be able to pass data to the LAN IP Device. When the multicast service is no longer desired, the LAN IP Device can either ignore the service and the stream will time out, or the LAN IP Device can send an IGMP "leave" message to the chain to tear down the streaming traffic. Figure 8-3 provides a detailed example of IGMP and Multicast processes passing through a PS.

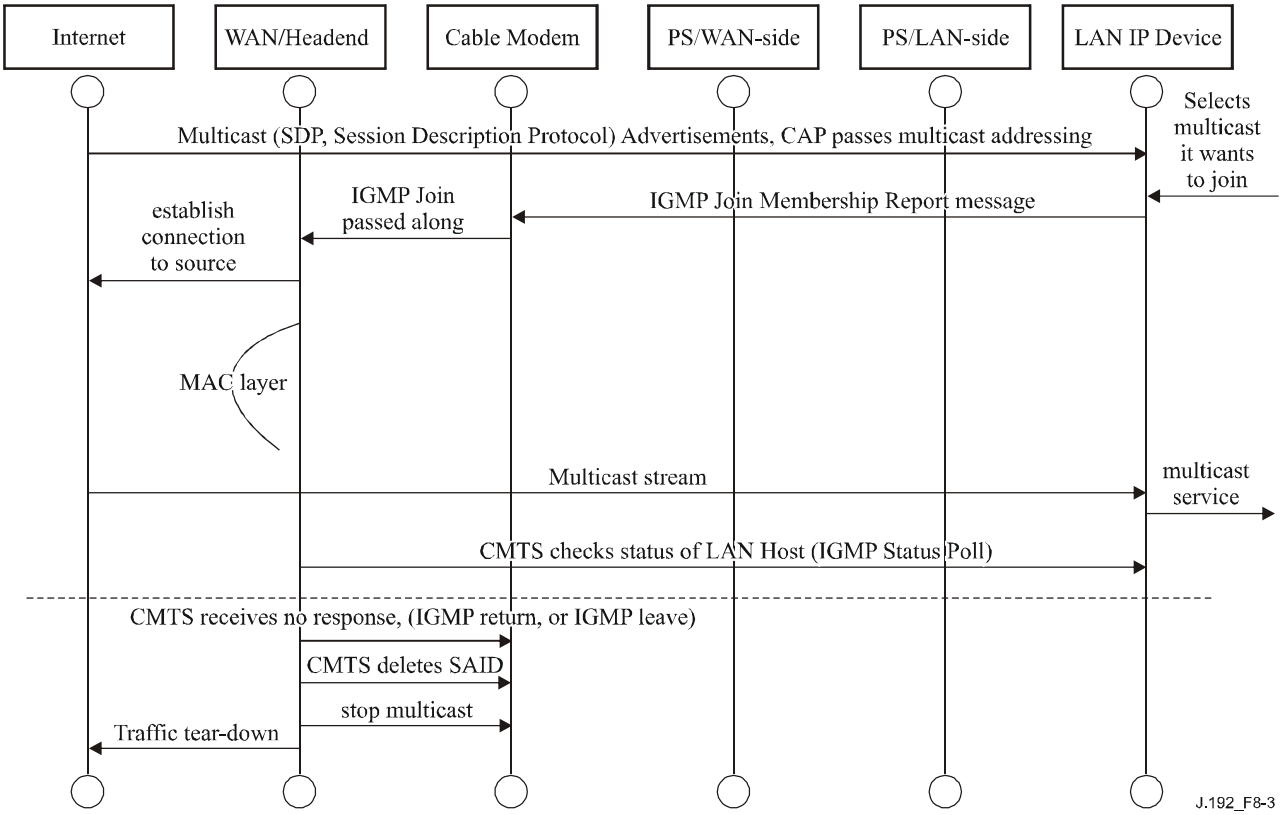


Figure 8-3/J.192 – Multicast via IGMP sequence

{informative text:

8.3.3.7 C-NAPT configuration using UPnP WANIPConnection service

The PS implements the UPnP WANIPConnection service (UWIC) to enable UPnP compliant LAN applications to configure port mappings at the CAP.

The PS declares the UPnP WANIPConnection service in the UPnP InternetGatewayDevice device description. The PS advertises the WANIPConnection service only when it operates in NAPT mode.

8.3.3.7.1 Relationship between WANIPConnection variables and cabhCapMappingTable objects

The PS lists all network address translation mappings in the cabhCapMappingTable. These include mappings created by management through SNMP, by the PS dynamically, and by LAN devices through UPnP. Several objects in the cabhCapMappingTable have a matching variable defined in the UPnP InternetGatewayDevice service specification [UIGD]. The relationships between cabhCapMappingTable objects and WANIPConnection service variables are shown in Table 8-2.

Table 8-2/J.192 – Related cabhCapMappingTable objects and WANIPConnection service variables

cabhCapMappingTable	WANIPConnection
cabhCapMappingWanAddr	ExternalIpAddress
cabhCapMappingWanPort	ExternalPort
cabhCapMappingLanAddr	InternalClient
cabhCapMappingLanPort	InternalPort
cabhCapMappingProtocol	PortMappingProtocol
cabhCapMappingCreateTime	N/A
cabhCapMappingLastUpdate	N/A
cabhCapMappingDuration	PortMappingLeaseDuration
cabhCapMappingRowDescription	PortMappingDescription
cabhCapMappingNumPorts	N/A
cabhCapMappingMethod	N/A
cabhCapMappingRemoteHost	RemoteHost
CabhCapMappingEnable	PortMappingEnabled
N/A Not Available	

The following clause describes how the PS uses these variables and objects in the context of the WANIPConnection service actions.

8.3.3.7.2 Analysis of actions

The PS enables the WANIPConnection service only when it operates in NAPT mode. In all the other modes (i.e., NAT, Passthrough or Disabled mode) the PS disables the WANIPConnection service.

The PS supports the following WANIPConnection service actions in order to allow UPnP devices to create, modify, delete, and read port mappings: GetNATRSIPStatus, AddPortMapping, DeletePortMapping, GetGenericPortMappingEntry, GetSpecificPortMappingEntry and GetExternalIPAddress.

The PS implements the GetNATRSIPStatus action (per UWIC) to inform a requesting UPnP Control Points whether NAPT is enabled or not. The response of the PS to this action will depend on whether the requesting device is in the LAN-Trans or LAN-Pass domain. For devices in the LAN-Trans domain the PS will respond that NAPT is enabled, and for devices in the LAN-Pass domain that NAPT is disabled.

UPnP control points can create new mappings at the PS by invoking the AddPortMapping action. The PS shows mappings created through this action in the cabhCapMappingTable with a cabhCapMappingMethod value of UPnP (3). The PS creates a new mapping when the action's ExternalPort and PortMappingProtocol variables do not match a port and protocol currently in use, as defined in UPnP. The PS also creates a new mapping if the action's variables match an existing mapping's External Port, Port Mapping Protocol, and Internal Client but do not match the mapping's Remote Host.

Control points can modify NAPT mappings at the PS by using the AddPortMapping action. As defined in UPnP, the action specifies an already existing mapping when RemoteHost, ExternalPort, PortMappingProtocol and InternalClient match the mapping. The PS permits control points to modify mappings created through UPnP only. The PS does not permit control points to modify mappings that were created at the PS through SNMP. The PS, upon reception of an

AddPortMapping that matches a current entry which was created through SNMP, will not modify the entry but will return an OK in response to the action.

Control points can invoke the DeletePortMapping action to eliminate a mapping at the PS. When this action is executed, the PS will delete a mapping if the mapping was created through UPnP. Otherwise, the PS will not delete the mapping and will return an error.

When control points invoke the GetGenericPortMappingEntry action, the PS will return mappings that the PS created dynamically and those that were created at the PS through SNMP or UPnP. More specifically, the PS will return all mappings in the cabhCapMappingTable with a cabhCapMappingProtocol value of UDP(3) or TCP(4). For mappings that the PS created dynamically or that were created at the PS through SNMP, the PS returns for RemoteHost an empty string and for PortMappingEnabled a value of 1 (True).

Upon reception of a GetSpecificPortMappingEntry action the PS checks the entries of the cabhCapMappingTable and returns the entry, if any, that matches the action's input parameters RemoteHost, ExternalPort and PortMappingProtocol.

In response to a GetExternalIPAddress action, the PS returns the current WAN-Data IP address.

}

8.3.3.8 IPCable2Home packet handling examples

This clause provides an informative look at processing involved for packet handling. Figure 8-4 shows an example of possible packet processing steps for LAN-to-WAN uni-cast traffic, and Figure 8-5 shows an example of possible packet processing steps for WAN-to-LAN uni-cast traffic.

NOTE – These examples are informative only and do not imply any requirements on implementation.

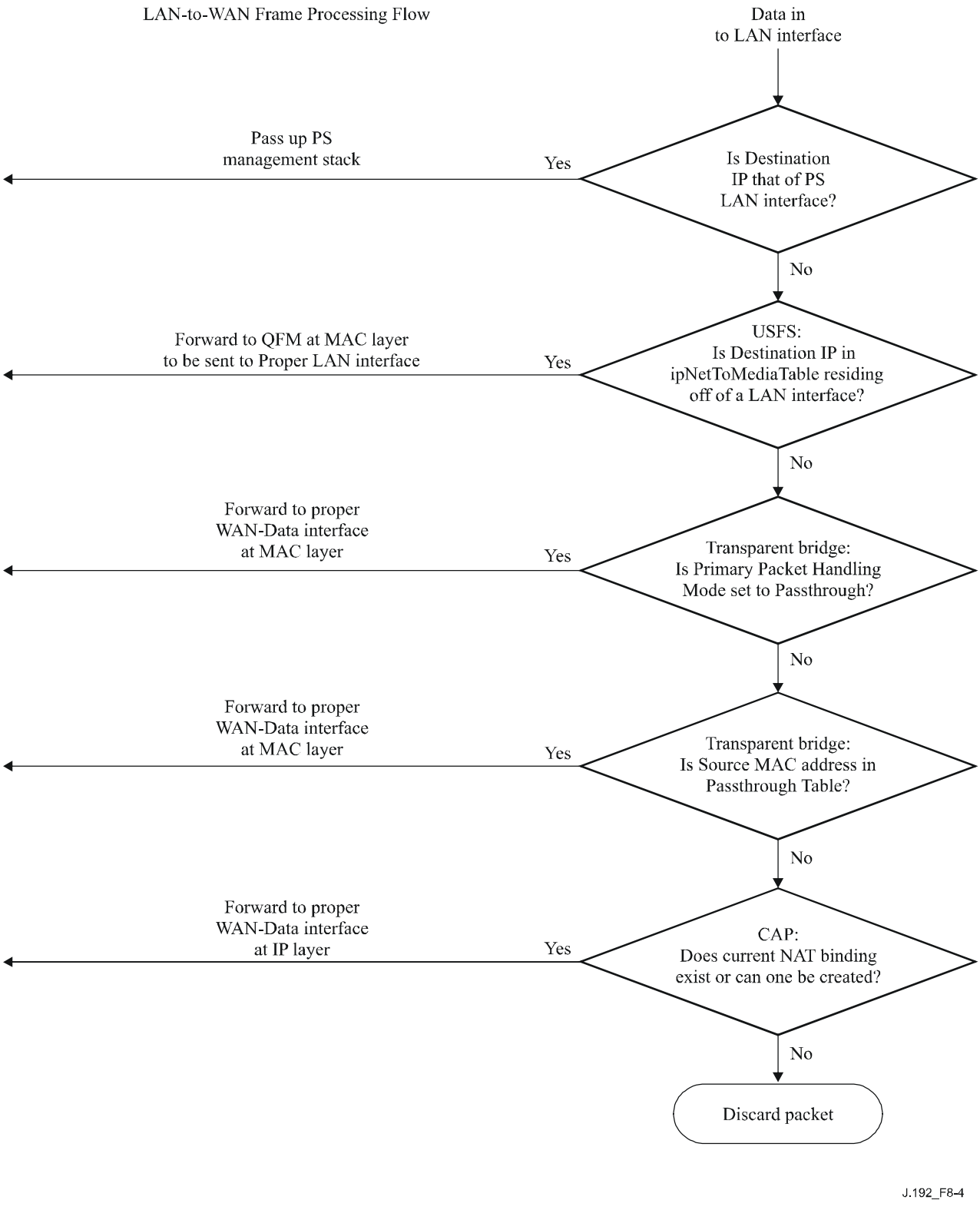


Figure 8-4/J.192 – LAN-to-WAN packet processing example

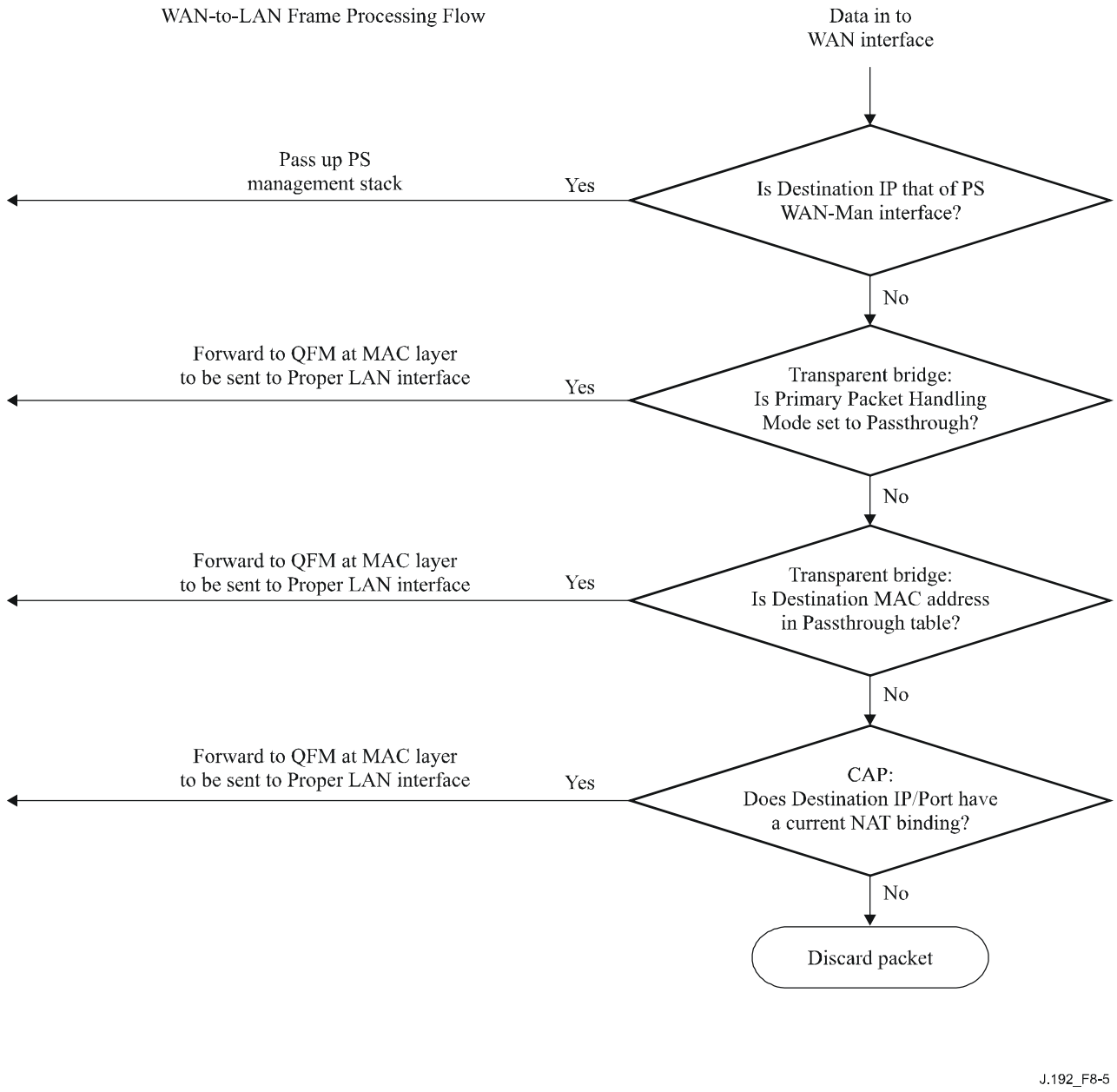


Figure 8-5/J.192 – WAN-to-LAN packet processing example

8.3.4 CAP requirements

8.3.4.1 General requirements

All logical IP interfaces on the Portal Services element MUST be compliant with [RFC 1122] and [RFC 1123], sections 3 and 4, to enable standard communication with Internet Hosts.

The PS 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].

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.

8.3.4.2 Packet handling requirements

The PS MUST support Passthrough Mode, C-NAT Transparent Routing Mode, and C-NAPT Transparent Routing Mode, and the PS MUST support the selection of this Primary Packet-handling Mode, via the cabhCapPrimaryMode MIB object.

If the Primary Packet-handling Mode, cabhCapPrimaryMode, is set to C-NAT, the PS MUST make certain there exists an available Headend supplied IP address in the WAN-Data IP Address Pool (with a current DHCP lease) before attempting to use this IP address as part of a C-NAT Mapping. If the CAP is unable to create a C-NAT Mapping, due to WAN-Data IP Address Pool depletion, it MUST generate a standard event (as defined in Annex B).

The PS 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 PS MUST enter zero for cabhCapMappingLanPort and cabhCapMappingWanPort for that row.

The PS 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 PS 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 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.

8.3.4.3 Passthrough requirements

When the CAP's Primary Packet-handling Mode, cabhCapPrimaryMode, is set to Passthrough mode, the PS MUST act as a transparent bridge, as defined in [ISO/IEC 10038], between the WAN-Data realm and LAN-Pass realm, and MUST NOT perform any C-NAT or C-NAPT Transparent Routing functions. A PS acting as a transparent bridge for LAN-Pass devices (cabhCapPrimaryMode = passthrough(3) or cabhCapPrimaryMode = napt(1) with entries in the cabhCapPassthroughTable) MUST transparently bridge all frame types that the DOCSIS specifications require a cable modem to pass. Even when the Primary Packet-handling Mode is set to Passthrough, USFS processing MUST take precedence over LAN-to-WAN bridging decisions.

8.3.4.4 C-NAT and C-NAPT transparent routing requirements

When the Primary Packet-handling Mode (cabhCapPrimaryMode) is set to C-NAT, the PS MUST support C-NAT address translation processes in accordance with the basic NAT requirements defined in [RFC 3022].

When the Primary Packet-handling Mode (cabhCapPrimaryMode) is set to C-NAPT, the PS MUST support C-NAPT address translation processes in accordance with the basic NAPT requirements defined in [RFC 3022].

Regardless of the Primary Packet-handling Mode, the PS MUST support the creation and deletion of Static C-NAT and C-NAPT Mappings, by enabling the NMS system to read, create, and delete (via the CMP) Static CAP Mapping (cabhCapMappingTable) entries.

NMS created Static C-NAT and C-NAPT Mappings MUST persist across PS reboots.

The PS MUST support the creation of Dynamic C-NAT and C-NAPT Mappings, initiated by LAN-to-WAN TCP, UDP, or ICMP traffic. The PS MUST enable the NMS system to read (via the CMP) Dynamic CAP Mapping (cabhCapMappingTable) entries.

The PS MUST support the deletion of Dynamic C-NAT and C-NAPT Mappings if a given Mapping is associated with a TCP session and that TCP session terminates or the TCP inactivity timeout, cabhCapTcpTimeWait, for that Mapping elapses.

The PS MUST support the deletion of Dynamic C-NAT and C-NAPT Mappings if a given Mapping is associated with a UDP session and the UDP inactivity timeout, cabhCapUdpTimeWait, for that Mapping elapses.

The PS MUST support the deletion of Dynamic C-NAT and C-NAPT Mappings if a given Mapping is associated with an ICMP session and the ICMP inactivity timeout, cabhCapIcmpTimeWait, for that Mapping elapses.

Dynamic C-NAT and C-NAPT Mappings MUST NOT persist across PS reboots.

A correspondence, or mapping, is created in the CAP Mapping Table (cabhCapMappingTable) between a LAN-Trans realm LAN IP device's private IP address and a port number (private pair: cabhCapMappingLanAddr and cabhCapMappingLanPort) and a public IP address and a port number (public pair: cabhCapMappingWanAddr and cabhCapMappingWanPort), corresponding to a session established by the LAN IP Device. The private pair – public pair combination is referred to as a translation binding, or as a mapping. The translation binding is represented as an entry in the CAP Mapping Table, and is created automatically by the PS when the LAN IP Device in the LAN-Trans realm sends traffic destined to a device on the WAN through the PS acting as the LAN IP Device's default gateway, created automatically by the PS under conditions defined for {informative text:UPnPWANIPConnection service} support in 8.3.3.7, or by direct configuration of the CAP Mapping Table through a PS configuration file or SNMP set-request messages. C-NAT and C-NAPT IP address mappings (private and public pairs and translation bindings) MUST be consistent and not change for a LAN IP device once the mappings are created and until they are destroyed.

Each private pair in a CAP Mapping Table entry MUST have the same corresponding public pair each time it appears in the CAP Mapping Table, regardless of the Remote Host IP address (cabhCapMappingRemoteHostAddr) value for that entry. That is, a private pair is required to always be associated with the same public pair. This restriction prohibits implementation of a Symmetric NAT as described in [RFC 3489].

If a mapping exists in the CAP Mapping Table for a LAN IP Device in the LAN-Trans realm, with a particular value for cabhCapMappingRemoteHostAddr, and traffic arrives at the PS LAN interface from that same LAN IP Device with the same source IP address and same source port number but destined for a different Remote Host IP address, then the PS MUST create a new entry in the CAP Mapping Table, with the same private pair (cabhCapMappingLanAddr and cabhCapMappingLanPort) and public pair (cabhCapMappingWanAddr and cabhCapMappingWanPort) and with a cabhCapMappingRemoteHostAddr entry with the value of the new destination IP address. That is, for the same private pair the PS is required to use the same public pair for the binding. The PS is therefore required to create a new, unique entry in the CAP

Mapping Table with a different Remote Host IP address value, but with the same private-public translation binding as the previous entry. The result could be that the private-public binding (private pair and public pair in a CAP Mapping Table entry) appears multiple times in the CAP Mapping Table, but with a different value of Remote Host IP address for each entry.

8.3.4.5 Virtual Private Network support requirements

When the CAP is operating in C-NAT or C-NAPT Primary Packet-handling mode (as indicated by the value of cabhCapPrimaryMode), the PS MUST recognize IPSec sessions initiated by VPN clients in the LAN-Trans realm, create appropriate mappings in the CAP Mapping Table, and map port 500 for inbound (WAN-to-LAN) traffic to the LAN-Trans IP address bound to the LAN IP Device that initiated the session.

When the CAP is operating in C-NAT or C-NAPT Primary Packet-handling mode (as indicated by the value of cabhCapPrimaryMode) and it recognizes an IPSec session when another one has already been mapped in the CAP Mapping Table to a different VPN server, the PS MAY create mappings for the new session, e.g., by port shifting.

If inbound traffic on port 500 is received by the CAP and there is no active IPSec VPN session, then the packets received through port 500 MUST be discarded.

The PS MUST support IPSec sessions using Encapsulating Security Payload Tunneling mode, [RFC 2406].

8.3.4.6 CAP DMZ functionality requirements

When the Primary Packet-handling Mode (cabhCapPrimaryMode) is set to C-NAPT and there is a C-NAPT static mapping with the WAN port number and the LAN port number set to 0 (i.e., when a DMZ entry has been created in the CAP), then the PS MUST translate the IP addresses specified in the mapping (DMZ entry) for packets that are not associated with an existing dynamic or static C-NAPT mapping.

When a DMZ entry is created in the CAP Mapping Table for a LAN IP address that is dynamically assigned by the PS (CDS), the PS MUST create an IP address lease reservation for that address. The PS MUST determine if the DMZ IP address is dynamically assigned by the CDS, e.g., by looking it up in the cabhCdpLanAddrTable. If a corresponding entry exists in this table with the value of cabhCdpLanAddrMethod equal to either dynamicActive(4) or dynamicInactive(3), then the PS MUST replace that entry with one that represents a lease reservation for that IP address in the table, that is, one whose value of cabhCdpLanAddrMethod is set to either psReservationActive(6) or psReservationInactive(5), respectively. If there is no entry in the CAP Mapping Table corresponding to the DMZ IP address in the cabhCdpLanAddrTable, then the PS MUST NOT create a lease reservation for that IP address.

When a DMZ entry is removed from the cabhCapMappingTable for a LAN IP address (DMZ host), the PS MUST remove the corresponding IP address lease reservation that it had internally created (identified by cabhCdpLanAddrMethod=psReservationActive(6)) from the cabhCdpLanAddrTable as long as either the docsDevFilterIpTable or the cabhSec2FwLocalFilterIpTable does not have a corresponding firewall filter rule entry that requires it.

8.3.4.7 Mixed bridging/Routing mode requirements

The PS MUST support Mixed Bridging/Routing Mode as described in 8.3, where the CAP Primary Packet-handling Mode, cabhCapPrimaryMode, is set to C-NAT or C-NAPT Transparent Routing and where the CAP will also transparently bridge traffic for particular MAC addresses. If the CAP Primary Packet-handling Mode, cabhCapPrimaryMode, is set to C-NAT or C-NAPT Transparent Routing and the NMS has written a MAC address, belonging to a LAN IP Device, into the cabhCapPassthroughTable, the PS MUST transparently bridge LAN-to-WAN traffic sourced by this MAC address and WAN-to-LAN traffic destined for this MAC address.

When in Mixed Bridging/Routing Mode, as described in 8.3, the USFS function MUST be applied to all LAN originated traffic received.

8.3.4.8 USFS requirements

Upstream Selective Forwarding Switch (USFS) functionality MUST be applied to packet processing, regardless of the CAP's packet-handling mode (Passthrough, C-NAT, C-NAPT, or mixed Bridging/Routing).

The USFS function MUST inspect all IP traffic originating on PS LAN interfaces, to determine if the destination IP address of a packet is that of a device residing on a PS LAN interface. If the destination IP address in a packet inspected by the USFS is that of a LAN IP Device residing off of a PS LAN interface, the USFS function MUST replace the MAC Layer Destination address, within the packet's Layer 2 header, with the MAC address of that destination LAN IP Device and forward the frame to the QoS Forwarding and Media Access (QFM) entity (see 10.3.1) in the PS to be forwarded out on the proper physical LAN interface according to the packet priority.

The USFS MUST NOT forward any packet destined for a LAN IP Device out any WAN Interface.

{informative text:

8.3.4.9 C-NAPT configuration using UPnP WANIPConnection service requirements

The PS MUST implement the UPnP InternetGatewayDevice WANIPConnection service as defined in UWIC.

The PS MUST enable the WANIPConnection service only when it operates in NAPT mode (cabhCapPrimaryMode = napt(1)) AND when cabhCapUpnpPortForwardingEnable = true(1). The PS MUST NOT enable the WANIPConnection service when it operates in NAT, Passthrough, or Disabled modes, or when cabhCapUpnpPortForwardingEnable = false(2). Whenever the WANIPConnection service is disabled, the PS MUST eliminate all mappings created through the WANIPConnection service.

When the PS is configured in NAPT mode, it MUST support the following WANIPConnection actions (UWIC) in order to allow UPnP devices to create, modify, delete, and read portmappings: GetNATRSIPStatus, AddPortMapping, DeletePortMapping, GetGenericPortMappingEntry, GetSpecificPortMappingEntry and GetExternalIPAddress.

If the PS is configured in NAPT mode and it receives a GetNATRSIPStatus UWIC request from a device in the LAN-Trans domain, it MUST respond that NAT is enabled. If the PS is configured in NAPT mode and it receives a GetNATRSIPStatus request from device in the LAN-Pass domain, it MUST respond that NAT is disabled.

The PS MUST list mappings created through the AddPortMapping action in the cabhCapMappingTable with a cabhCapMappingMethod value of UPnP(3). The PS MUST create a new mapping when the AddPortMapping action's ExternalPort and PortMappingProtocol variables do not match a port and protocol currently in use in another mapping. The PS MUST create a new mapping if the action's variables match an existing mapping's External Port, Port Mapping Protocol, and Internal Client but do not match the mapping's Remote Host.

The PS MUST allow control points to modify, through the AddPortMapping action (UWIC), mappings which have a cabhCapMappingMethod value of UPnP(3). The PS is expected to understand that the AddPortMapping action refers to an already existing mapping when the action variables RemoteHost, ExternalPort, PortMappingProtocol and InternalClient match the mapping, and the PS MUST NOT modify mappings with a cabhCapMappingMethod value of static (1) in response to the AddPortMapping action. If the AddPortMapping action specifies an existing mapping with a cabhCapMappingMethod value of static(1), however, the PS MUST return an OK in response to the action.

When the DeletePortMapping action is invoked, the PS MUST delete a mapping that matches the request if the mapping has a cabhCapMappingMethod of UPnP(3). If the mapping to be deleted has a cabhCapMappingMethod value of static(1) or dynamic(2), the PS MUST ignore the DeletePortMapping action and return the UPnP error code 501 (Action Failed).

When a control point invokes the GetGenericPortMappingEntry action, the PS MUST return all mappings in the cabhCapMappingTable with a cabhCapMappingProtocol value of UDP(3) or TCP(4).

Upon reception of a GetSpecificPortMappingEntry action, the PS MUST check the entries of the cabhCapMappingTable and return the entry, if any, that matches the action's input parameters RemoteHost, ExternalPort and PortMappingProtocol.

In response to a GetExternalIPAddress action, the PS MUST return the current WAN-Data IP address.

}

9 Name resolution

9.1 Introduction/Overview

9.1.1 Goals

The goals of name resolution include:

- Provide Domain Name Service from a server in the PS to DNS clients within LAN IP Devices, even during cable connection outages.
- Enable subscribers to refer to local devices via intuitive device names rather than by IP address.
- Via recursive queries to remote DNS servers, provide answers to LAN DNS clients when queried for resolution of non-local hostnames.
- Provide easy DNS service recovery upon re-establishment of cable connectivity after an outage.

9.1.2 Assumptions

The operating assumptions for the naming services include:

- The DNS server in the PS element is the only DNS server authoritative for LAN IP Devices in the LAN-Trans realm.
- The PS element will not provide DNS service to LAN IP Devices in the LAN-Pass realm.
- If the PS element makes use of multiple WAN-Data addresses, the WAN DNS Server information obtained during the most recent WAN-Data address acquisition process (DHCP) will be used.

9.2 Architecture

9.2.1 System design guidelines

See Table 9-1.

Table 9-1/J.192 – Name resolution system design guidelines

Reference	Guidelines
Name Rsln 1	Provide Domain Name Service from a server in the PS to DNS clients within LAN IP Devices, for name resolution of LAN IP Devices (independent of the state of the WAN connection).
Name Rsln 2	Provide DNS answers, via recursive queries beginning with a cable network DNS server, for DNS clients within LAN IP Devices, for resolution of non-local hostnames.

9.2.2 System description

This clause provides an overview of the IPCable2Home name resolution services within the PS element.

9.2.2.1 Name resolution functional overview

The IPCable2Home Naming Portal (CNP) is a service running in the PS that provides a simple DNS server for LAN IP Devices in the LAN-Trans address realm. However, CNP functionality for LAN-Trans address realm is bypassed if the cabhCdpServerDnsAddress MIB is set to the value other than cabhCdpServerRouter. The CNP is not used by LAN IP Devices in the LAN-Pass address realm because they will be directly served by DNS servers external to the home.

Typically, LAN IP Devices in the LAN-Trans realm are configured by the CDP to use the CNP as their Domain Name Server. The CNP service in the LAN-Trans realm does not depend on the state of the WAN connection. The CNP performs the following tasks:

- Resolves hostnames for LAN IP Devices, returning their corresponding IP addresses.
- Provide DNS answers, via recursive queries beginning with a DNS server in the cable network, 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.

Making the CNP the primary DNS server on the 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.

9.2.2.2 Name resolution operation

When queried to resolve a hostname, the CNP function of the PS performs the lookup process shown in Figure 9-1. The CNP responds to initial standard DNS queries [RFC 1035], directed to cabhCdpServerDnsAddress, for all name lookups. It is the responsibility of the CNP to make recursive queries to external DNS servers – beginning with the first cabhCdpWanDnsServerIp entry in the CDP's cabhCdpWanDnsServerTable – 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 performs recursive queries to external DNS servers (of which the initial one is learned by the CDC via DHCP options).

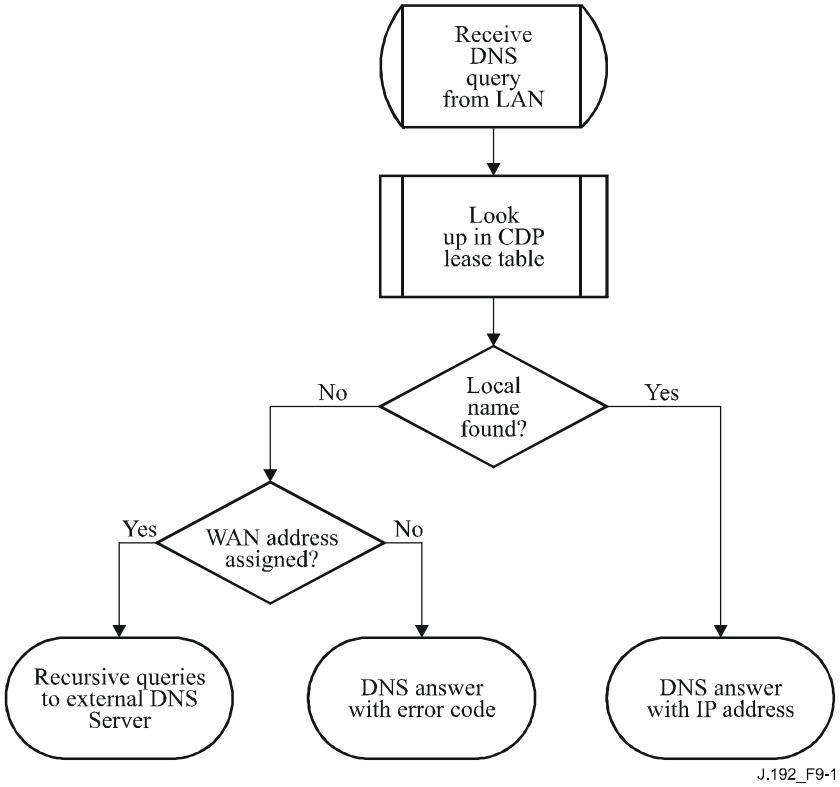


Figure 9-1/J.192 – CNP packet processing

A standard DNS query specifies a target domain name (QNAME), query type (QTYPE), and query class (QCLASS), and asks for Resource Records that match. The CNP will respond to the DNS queries with QCLASS = IN, and QTYPE = A, NS, SOA or PTR as defined in [RFC 1035]. Support for zone transfers and DNS over TCP is not required.

Since the CNP is an authoritative DNS server inside the LAN-Trans realm, it will provide Start of Authority (SOA) and Authoritative Name Server (NS) records on request. An example of the SOA record fields (see section 3.3.13 of [RFC 1035]) follows:

Table 9-2/J.192 – SOA record fields

[RFC 1035] RDATA field	IPCable2Home CDP MIB Object
MNAME	cabhCdpServerDomainName
RNAME	Not specified
SERIAL	Not specified
REFRESH	Not specified
RETRY	Not specified
EXPIRE	Not specified
MINIMUM	Not specified

The MNAME field is the domain name of the LAN-trans address realm. The CNP uses the value stored in cabhCdpServerDomainName as the LAN-trans address realm domain name.

The RNAME field is the mailbox of the responsible person for the domain. If the PS maintains an E-mail address for an administrator, this information could be specified in this field.

The SERIAL field is an unsigned 32-bit number, used to identify the version of the zone information. But since this Recommendation does not specify zone transfers, value of this field is not specified.

9.3 Name resolution requirements

The CNP MUST comply with the standard DNS message format and support standard DNS queries, as described in [RFC 1034] and [RFC 1035].

The CNP is a stateless server that MUST be able to receive queries and send replies in UDP packets [RFC 768].

The CNP MUST support recursive mode, as defined in [RFC 1034].

The CNP answers name queries, beginning with local information within the PS, and its response messages MUST contain an answer or an error.

The CNP MUST only respond to DNS queries addressed to the IP address represented by the value of the cabCdpServerRouter MIB object (i.e., the PS's LAN side IP address).

The CNP MUST NOT respond to any DNS queries addressed to the PS WAN-Man or 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 existence of any cabhCdpWanDnsServerIp entries in the CDP MIB cabhCdpWanDnsServerTable, 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.

If the queried host name cannot be resolved by the CNP from local data, and the CDP's cabhCdpWanDnsServerTable is populated with at least one cabhCdpWanDnsServerIp entry, the CNP function of the PS MUST attempt to resolve the hostname query via recursive queries to external DNS servers, starting with queries to the DNS server, represented by the first cabhCdpWanDnsServerIp entry in the cabhCdpWanDnsServerTable. In this query, the CNP MUST use the WAN-DATA IP Address as the source IP address for the DNS Query Message. It is assumed that the Operator will have provided a Public IP Address as the WAN-DATA-IP to the CNP, if that is not the case then the Operator will provide for routing between a private WAN-DATA IP Address and the DNS server operated by the Operator at the headend.

If the host name cannot be resolved by the CNP from local data and no cabhCdpWanDnsServerIp entries exist in the cabhCdpWanDnsServerTable, the CNP function of the PS MUST respond to the host name resolution query with the appropriate error specified by [RFC 1035].

The CNP MUST respond to DNS queries of type QCLASS = IN, and QTYPE = A, NS, SOA or PTR. SRV [RFC 2782], and ENUM [RFC 3761].

The CNP responses to DNS queries MUST comply with section 3.3 of [RFC 1035], with Authoritative Answer bit set to '1' in the Header Section (see section 4.1.1 of [RFC 1035]).

Since the CNP is an authoritative DNS server inside the LAN-Trans realm, it MUST provide Start of Authority (SOA) and Authoritative Name Server (NS) records on request. The SOA record fields (see section 3.3.13 of [RFC 1035]) MUST contain an entry for the MNAME field that is equal to the value of the CDP's cabhCdpServerDomainName MIB object.

If cabhCdpServerDomainName is not set, the CNP MUST still provide DNS service to LAN IP Devices.

9.3.1 ENUM, NAPTR and SRV type records in DNS

9.3.1.1 ENUM queries and response at the CNP [RFC 3761]

This clause describes the CNP behaviour when it receives DNS queries of type ENUM from LAN CPE clients.

The CNP MUST be able to accept queries of type ENUM from the LAN CPE devices and propagate those towards the MSO using the cabhCdpWanDnsServerIp Address, when that MIB is populated and the MSO's DNS server is reachable.

The CNP MUST attempt to query the MSO's DNS Server represented in cabhCdpWanDnsServerIp, querying for NAPTR records corresponding to the DNS query of type ENUM from the LAN CPE device, as specified in [RFC 3761].

If the CNP gets NAPTR records for the ENUM query, the CNP MUST forward those records "as is", to the LAN CPE clients.

The CNP does not have a need to implement the Dynamic Delegation Discovery System (DDDS), as specified in [RFCs 3401-3404]. It is sufficient for the CNP to be able to propagate DNS queries of type ENUM towards the MSO's DNS servers represented in cabhCdpWanDnsServerIp. It is assumed that if an operator is providing services to their CableHome subscribers which require resolving DNS queries of type ENUM, then the operator will make available the capability of resolving those queries at the DNS server(s) represented in the cabhCdpWanDnsServerIp MIB.

9.3.1.2 SRV queries and response at the CNP [RFC 2782]

This clause describes the CNP behaviour when it receives DNS queries of type SRV from LAN CPE clients.

The CNP MUST be able to accept queries of type SRV from the LAN CPE devices and propagate those towards the operator using the cabhCdpWanDnsServerIp Address, when that MIB is populated and the operator's DNS server is reachable.

The CNP MUST attempt to query the operator's DNS Server represented in cabhCdpWanDnsServerIp, querying for SRV records corresponding to the DNS query from the LAN CPE device, as specified in [RFC 3761].

If the CNP gets SRV records for the query, the CNP MUST forward those records "as is", to the LAN CPE clients.

It is sufficient for the CNP to be able to propagate DNS queries of type SRV towards the operator's DNS servers represented in cabhCdpWanDnsServerIp. It is also noted that if an MSO is providing services to their CableHome subscribers which require resolving DNS queries of type SRV, then the operator will make available the capability of resolving those queries at the DNS server(s) represented in the cabhCdpWanDnsServerIp MIB.

10 Quality of Service

10.1 Introduction

This clause describes the IPCable2Home environment for enabling home networking applications running on the devices connected to home network to utilize QoS features supported by LAN protocol. This environment provides a management mechanism that prioritizes data flows to support real-time application traffic, such as VoIP, A/V streaming, and video gaming, by using prioritized media access and queuing. IPCable2Home QoS is complementary to the IPCablecom & J.112 QoS mechanisms, which allow QoS traffic management over the HFC network. {informative text:IPCable2Home uses Universal Plug and Play (UPnPTM) QoS messaging on the LAN interface(s).}

This Recommendation defines the necessary PS element and sub-element QoS requirements that enable applications to establish different levels of QoS within the home network and for operators and users to communicate the desired priority treatment to the cable operator-enabled {informative text:as well as UPnP-enabled applications} on the home network.

10.1.1 Goals

The goals for IPCable2HomeQoS include:

{informative text:

- Enable home networking applications to establish prioritized data transmission among UPnP Hosts as well as between the UPnP Hosts and the Residential Gateway using UPnP compliant messaging.
- Enable home networking applications to establish prioritized data transmission among Hosts as well as between the Hosts and the IPCable2Home Residential Gateway using UPnP compliant messaging.

}

10.1.2 Assumptions

The following assumptions were made for IPCable2Home QoS:

{informative text:

- Applications that benefit from QoS can be running either on a CableLabs Host devices or in UPnP QoS compliant devices.}
- IPCable2Home Host applications could include IPCablecom services.

{informative text:

NOTE – Any LAN host device that would like to receive QoS for operator services will have to comply with the UPnP QoS 1.0 Specification and the device's operating system and network stack will need to have appropriate QoS capabilities.}

10.2 QoS architecture

The IPCable2Home quality-of-service (CQoS) architecture is composed of IPCable2Home residential gateway functional element (PS and sub-elements within the PS). Developers of IPCable2Home residential gateway implement one or more of these elements depending on the desired feature set of these products. The basic CQoS elements are presented in 10.2.3.

10.2.1 System design guidelines

The overall IPCable2Home QoS system design guidelines are listed in Table 10-1 below.

Table 10-1/J.192 – IPCable2Home QoS system design guidelines

Number	Guidelines
QoS 1	QoS Media Access: IPCable2Home will define a layer 3 management function that controls transmission access using priorities on shared media for the PS logical element. It will provide prioritized media access to various devices and applications on the home network.
QoS 2	QoS Forwarding: The PS will support a queuing mechanism that prioritizes packets that are received from multiple interfaces (LAN or WAN) and are to be retransmitted /forwarded through LAN interfaces.
QoS 3	{informative text:QoS Policy Management: IPCable2Home will specify a signalling and management mechanism for communication of QoS policies between the PS and BPs desiring QoS as well as between the PS and UPnP QoS compliant devices within a home network. This mechanism will be aggregated and managed in the PS.}
QoS 4	Outline appropriate PS capabilities to facilitate integration with IPCablecom Multimedia to achieve end-to-end QoS in the future.

{informative text:

10.2.2 Relationship with UPnP QoS

The CQoS Architecture uses UPnP QoS compliant messaging between the QoS capable elements. In the UPnP architecture, control messages are initiated from UPnP Control Point elements and responded to by UPnP Service elements. Thus, the elements or sub-element of the PS entities are described in terms of the UPnP QoS Control Point and UPnP QoS Service elements (UQA).

The UPnP architecture is also characterized as a distributed architecture in that there may be multiple instantiations of a particular service in the HN that may be used interchangeably. While the CQoS Architecture describes certain UPnP QoS Services that are contained within the PS, there may be other devices within the HN that also implement the same UPnP QoS Services. When PS descriptions or requirements are written using UPnP QoS element terminology, the interaction may be occurring with non-PS devices. For example, in a description of a Control Point interacting with a QoS Manager Service, a Control Point may be interacting with the QoS Manager functionality in a third-party device instead of interacting with the QoS Manager Service in the PS.}

10.2.3 IPCable2Home QoS system description

The CQoS Architecture is composed of the following entities:

- Portal Services element (PS);
- IPCable2Home Quality-of-Service Portal sub-element (CQP);
- {informative text:UPnP Hosts with QoS capabilities.}

The cable data network equipment (headend) manages the IPCable2Home QoS functions.

10.2.3.1 CQP sub-element

The PS element includes an IPCable2Home Quality of Service Portal (CQP) sub-element. The CQP acts as a CQoS portal for QoS capable UPnP Hosts. Its primary function is to enable priorities based QoS for the devices within the home network. It performs priorities based queuing/forwarding and media access for the traffic originating from the PS as well as for the traffic transiting through the PS. It is also responsible for communication of QoS Policies{informative text:to UPnP QoS Manager(s) [UQM] within the home when operating as the sole UPnP QoS Policy Holder Service [UQPH].}

10.2.3.2 QoS functionality in CQP

CQP sub-elements consist of the following functionalities:

- **IPCable2Home QoS Broker (CQB)**: Responsible for setting up QoS on the home network as well as access network. This entity also configures QoS in the IPCable2Home PS. CQB consists of following functionalities:
  - **QoS prioritized Forwarding and Media Access (QFM)**: specifies prioritized queuing and packet forwarding as well as prioritized shared media access in the PS.
  - **IPCable2Home-IPCablecom Multimedia (CH-PCMM) Interface**: This is an interface to request access network QoS conforming to the IPCablecom Multimedia architecture [ITU-T Rec. J.179] from the IPCable2Home PS. The CH-PCMM Interface can also receive requests for establishing HN QoS from PCMM entities in the WAN. Upon such a request, the CH QoS Broker Entity may use CH QoS Control Point to set up HN QoS. Specific requirements for this interface are for future study.
  - {informative text:**UPnP QoS Device Service Interface (QD)**: The CQB may include a UPnP QoS Device Service (UQD) Interface for the following purposes:
    1) Through this interface the PS receives traffic classifier requests from UPnP QoS Manager Entities (UQM) on the LAN and places them in the PS database. These classifiers are used by the QFM functionality for packet classification.
    2) Using this service interface, the PS can trigger a request for access network QoS for the traffic streams that span access network and home network using the CH-PCMM Interface to be defined in the future.}

{informative text:

- **QoS Policy Server (QPS)**: This functionality is responsible for maintaining a repository of QoS policy for various devices and applications within the home network and for communication of the QoS policy when requested by a UPnP QoS Manager Entity [UQM] on the LAN. The QPS utilizes UPnP QoS Policy Holder Service [UQPH] interface to communicate QoS policies on the home LAN. QPS has WAN side MIB interface for operators to manage and retrieve QoS Policies.
- **QoS Manager Service (QM)**: The CQP is required to implement the UPnP QoS Manager Service (UQM). This requirement ensures that there is at least one UPnP QoS Manager Service for UPnP Control Points to request QoS on the home LAN.
- **QoS functionality of PS Control Point (QCP)**: This entity acts as a Control Point for different UPnP QoS Services on the home LAN. It is responsible for capturing UPnP QoS announcements, events and issuing actions as needed for various UPnP QoS Services on the LAN.
  1) The QoS Discovery logic of this control point is responsible for collecting QoS related information from various UPnP QoS Services on the home LAN and storing it in the PS database. Cable Operators access this PS database information on the WAN through the SNMP MIB interface.
  2) The IPCable2Home QoS Broker logic of this control point is responsible for establishing QoS on the home LAN using UPnP QoS messaging under the direction of IPCable2Home QoS Broker. This logic is also responsible for requesting access network QoS using CH-PCMM interface.

}

{informative text:

10.2.3.3 QoS functionality in UPnP hosts

QoS Functionality in the UPnP Host consists of one or more of the following UPnP QoS Services:

- UPnP QoS Manager Service (UQM).
- UPnP QoS Device Service (UQD).
- UPnP QoS Policy Holder Service (UQPH).

The UPnP Host can also implement a UPnP Control Point that requests QoS on the home LAN.

}

10.2.3.4 Physical device classes & CQoS functional elements and messaging interfaces

An example of the relationship between the IPCable2Home Devices and the CQoS functional elements is presented in Figure 10-1. Figure 10-2 represents the messaging interfaces between various functional elements of the CQoS Architecture and does not imply any specific implementation. {informative text:Dashed arrows indicate potentially internal communication when UPnP QoS Entities within the PS interact.}

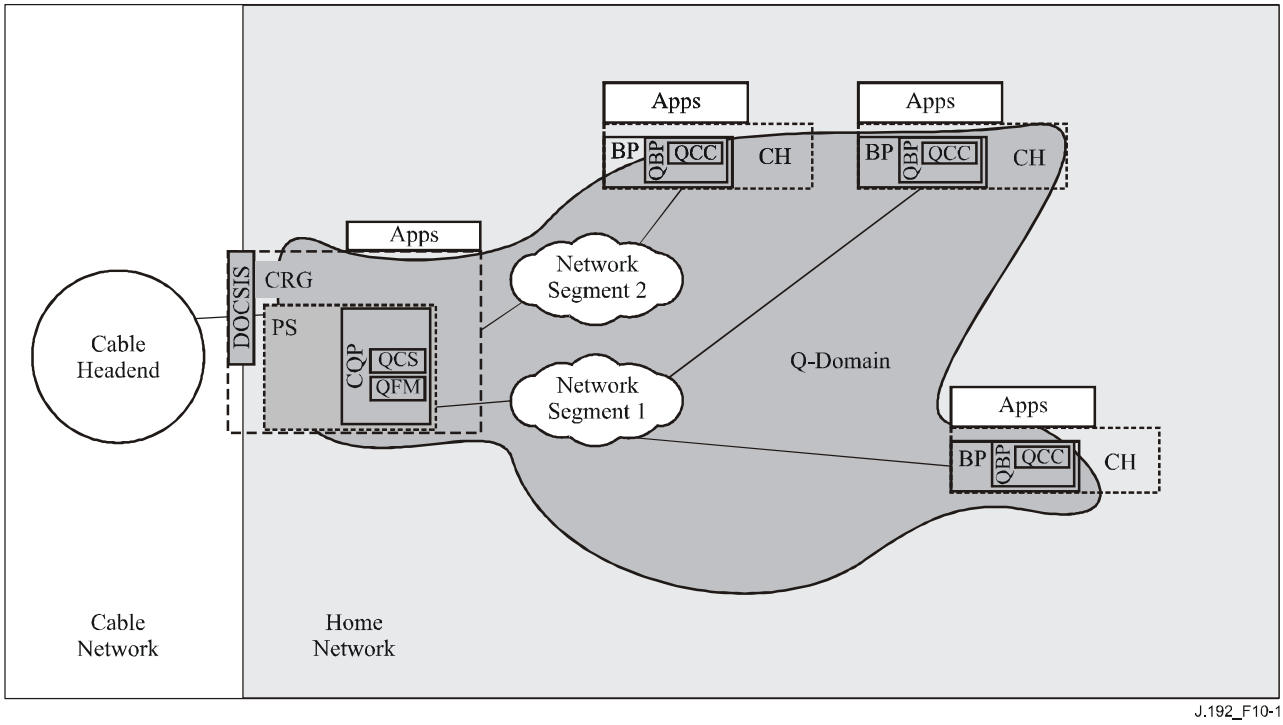


Figure 10-1/J.192 – Example of CQoS functional elements

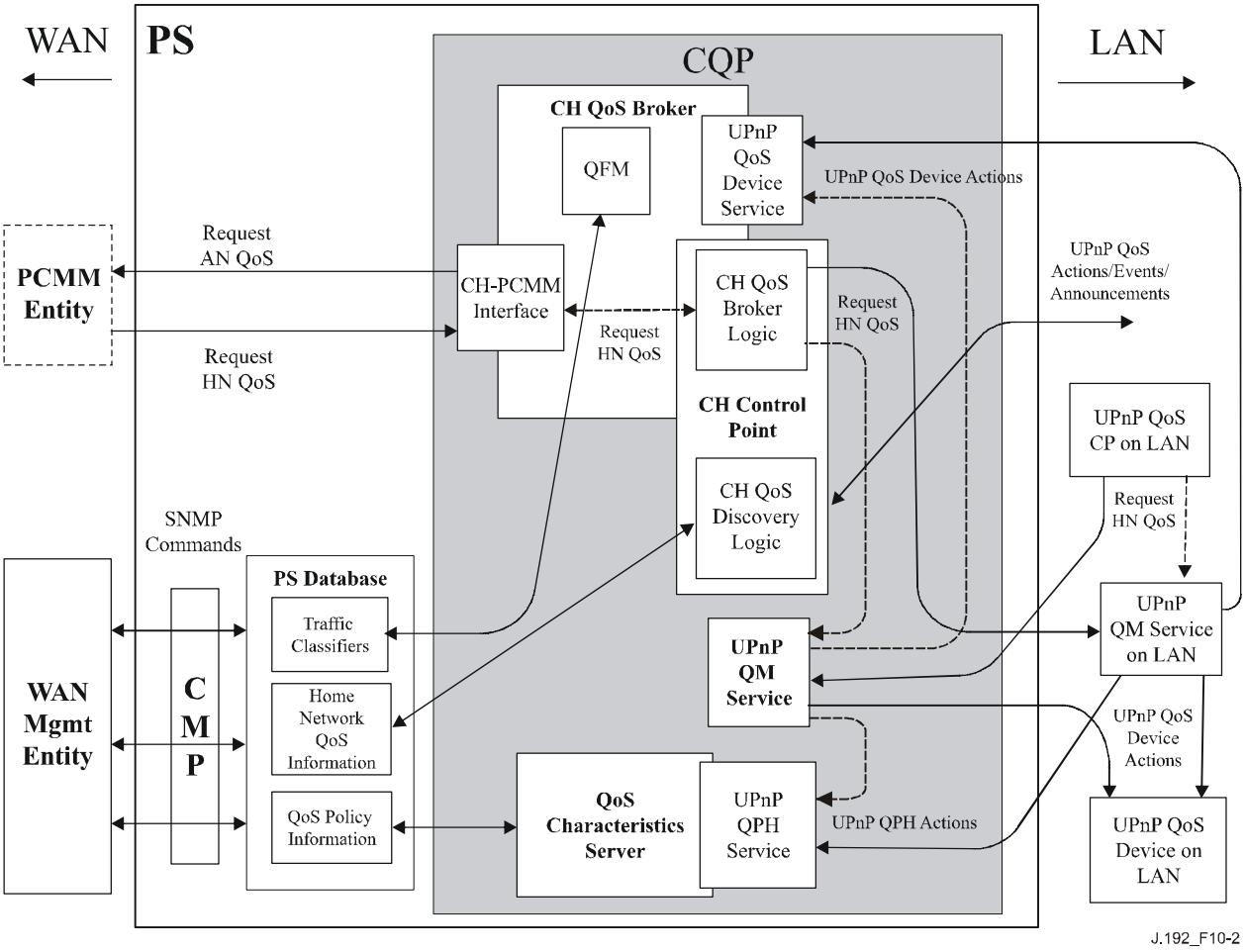


Figure 10-2/J.192 – CableHome QoS architecture messaging interfaces

10.2.3.5 IPCable2Home priorities and their mappings

10.2.3.5.1 IPCable2Home priorities

This Recommendation defines three different QoS priorities. They are:

- {informative text:
  UPnP Traffic Importance Number.}
- IPCable2Home Queuing Priorities.
- IPCable2Home Media Access Priorities.

{informative text:

10.2.3.5.1.1 UPnP TrafficImportanceNumber (TIN)

UPnP TrafficImportanceNumber (TIN per UQPH) is not linear but follows the same numbering scheme as ISO/IEC 10038, Annex G, packet priority values. This numbering scheme is outlined in Table 10-2 below. Cable operators assign UPnP TrafficImportanceNumber value to a traffic stream in the QoS Policy Table (cabhQos2PolicyTable) (per E.7) stored in the PS database.

Table 10-2/J.192 – Numbering scheme for UPnP TrafficImportanceNumber

UPnP Traffic Importance Number
7 (Highest)
6
5
4
3
0 (Best Effort/Legacy)
2
1 (Lowest)

NOTE – UPnP TrafficImportanceNumber values of 1 and 2 indicate lower priority values than 0. (0 is typically assigned for legacy best effort traffic).

}

10.2.3.5.1.2 IPCable2Home queuing priorities

In the PS, packets may arrive from multiple interfaces and be destined for a single interface. Each interface may implement a queueing function. In order to provide prioritized QoS for traffic within the home passing through the PS, this Recommendation specifies prioritized queueing functionality per interface in the PS. For this purpose, an individual queue within an interface is designated with a certain queuing priority. This is defined as IPCable2Home Queuing Priority. This IPCable2Home queuing priority may be identified for each packet to be transmitted on each PS interface so that the packet can be placed in an appropriate queue. This queuing priority is derived from the

{informative text:

UPnP TrafficImportanceNumber assigned to a traffic stream, using the number of queues supported by an interface on the PS. This mapping is performed as specified in ISO/IEC 10038, Annex G.}

10.2.3.5.1.3 IPCable2Home media access priorities

This Recommendation defines a prioritized QoS media access system in which traffic over a shared media may be prioritized based on the assigned packet priority. Thus, a shared media technology may support prioritized QoS such that a packet with higher priority is given preferential access to the shared media, versus a packet with lower priority. Various shared media technologies support varying number of media access priorities (e.g., Wi-Fi WMM supports four media access priorities, HomePNA support eight media access priorities, HomePlug supports four). {informative text:IPCable2Home Media Access Priority for the packet is derived from its UPnP TrafficImportanceNumber based on the number of media access priorities supported by the interface's layer-2 shared media technology. This mapping is performed as specified in ISO/IEC 10038, Annex G.} IPCable2Home Media Access Priority values are logical levels that represent a level of preference that a packet obtains for media access.

10.3 PS logical sub-element CQP

The CQP contains the CQB, QM, QCP and QPS functionalities as shown in Figure 10-1. The CQB functionality is described in 10.3.1. The QPS functionality is described in 10.3.2. The QM functionality is described in 10.3.3. The QCP Functionality is described in 10.3.4.

10.3.1 IPCable2Home QoS Broker (CQB)

CQB consists of QoS Forwarding and Media Access (QFM) functionality and optionally a QoS Device Service Interface.

10.3.1.1 QoS Forwarding and Media Access (QFM)

The Quality of Service Forwarding and Media access functionality (QFM) in the PS is responsible for prioritized forwarding and media access for the packets going through the PS onto the home LAN. This clause provides description of the QFM functionality in the PS and specifies associated PS requirements.

10.3.1.1.1 QoS Forwarding and Media Access goals

The goals for the QoS Forwarding and Media Access functionality include:

- To order packets arriving from multiple PS interfaces and forward them to a destination LAN interface according to their priorities and LAN interfaces' queuing capabilities.
- To provide prioritized access to the shared media during the packet transmission on the LAN interfaces based on the packet priority and prioritized media access capabilities of LAN interfaces.

10.3.1.1.2 QoS Forwarding and Media Access design guidelines

See Table 10-3.

Table 10-3/J.192 – QFM system design guidelines

Number	Guidelines
QFM.1	The QFM may operate on packets to and from the LAN-Trans and LAN-Pass address realms.
QFM.2	The QFM may determine the packet priority using the packet classification information available in the PS database.
QFM.3	The QFM may order incoming packets to exit through LAN interfaces according to their priorities.
QFM.4	The QFM should be able to work with different number of queues per interface.
QFM.5	{informative text:The QFM maps UPnP TrafficImportanceNumber of the packet to IPCable2Home Queuing priority according to the defined mapping.}
QFM.6	The QFM may provide prioritized access to the shared media on each LAN interface according to the packet priority and prioritized media access capabilities of LAN interface.
QFM.7	{informative text:The QFM maps UPnP TrafficImportanceNumber of the packet to IPCable2Home Media Access priority according to the defined mapping.}
QFM.8	The QFM should be able to operate with interfaces that support different numbers of priorities for media access.

10.3.1.1.3 QoS Forwarding and Media Access design assumptions

- Each PS LAN interface may support less than eight queues.
- Maximum number of queues supported a PS LAN interface is eight.
- Each PS LAN networking technology may support less than eight media access priorities.
- Maximum number of media access priorities supported by a PS LAN networking technology is eight.

10.3.1.1.4 QoS Forwarding and Media Access system description

The QFM provides the PS a mechanism to order and transmit packets on an outgoing LAN interface according to assigned priorities. It is through the assignment of priorities to packets and the action of the QFM that packets passing through the PS onto the home LAN are provided prioritized access to the shared LAN media. QFM is responsible for the following three functions:

1) {informative text:Classification process to identify the UPnP Traffic Importance Number (TIN) of the packet.}
2) Prioritized queuing.
3) Prioritized media access.

{informative text:

10.3.1.1.4.1 Classification of the packet to identify UPnP TrafficImportanceNumber

Packets passing through the PS and destined to a LAN interface can be originated either from the WAN (WAN-LAN downstream traffic) or from another LAN interface on the PS (LAN-LAN in-home traffic). For LAN-LAN in-home traffic, the QFM can optionally perform packet classification. However, for WAN-LAN downstream traffic, the QFM is required to perform packet classification to determine the appropriate UPnP TrafficImportanceNumber (TIN) if the outgoing LAN interface supports multiple queues or multiple priorities.

In order to perform packet classification, the PS examines the packet to identify a UPnP TrafficImportanceNumber for the packet. The PS examines the source IP, source port, destination IP, destination port, and Protocol type of the packet and tries to find a first match in the classifiers tables stored in the PS database that is represented by cabhQos2TrafficClassTable MIB (Annex A). If PS finds a matching entry, then it uses the value represented by cabhQos2TrafficClassImpNum MIB for that entry as a UPnP TrafficImportanceNumber that packet. If no matching entry is found, then the PS uses a UPnP TrafficImportanceNumber value of 0 for the packet. The PS uses this UPnP TrafficImportanceNumber value to determine the packet's IPCable2Home Queuing Priority and IPCable2Home Media Access Priority.

10.3.1.1.4.2 Prioritized queuing

The number of queues supported by an interface on the PS, to which the packet is destined, may not be the same as the eight levels of UPnP TrafficImportanceNumber. Hence the PS maps UPnP TrafficImportanceNumber value of the packet to a IPCable2Home Queuing Priority value as defined in [802.1D] Annex G. Then the PS places the packet in an appropriate queue of the destination interface that corresponds to this mapped IPCable2Home Queuing Priority value.

For each outgoing interface, the QFM polls all of the queues on that interface according to their priorities to extract packets out to be transmitted on the shared media. Every time the QFM is to extract a packet from the queues for a particular PS interface, it always starts its polling with the highest priority queue first. If the highest priority queue has no packets to be sent, the QFM polls the next highest priority queue of the remaining queues in the hierarchy until it finds a packet to be sent in one of the queues. Packets are extracted from each queue in the order they arrive. Thus, the queuing scheme used by the QFM may be described as First in, First Out with Priorities, and Highest Priority Queue First.

10.3.1.1.4.3 Prioritized media access

Once the QFM extracts a packet from the set of queues of an interface, the packet needs to be transmitted on the shared LAN media with an appropriate priority. Hence, the QFM maps the UPnP TrafficImportanceNumber value of the packet to the IPCable2Home Media Access Priority value as defined in [802.1D] Annex G. This value determines the level of preference the packet should use for accessing the shared media. Therefore, vendors need to insure that relative media access

preferences, as required by IPCable2Home Media Access Priority values, are maintained when transmitting packets over the shared LAN media.

}

10.3.1.1.4.4 IPCablecom applications support

Since the goal of QoS is to provide QoS over home network only, this Recommendation does not give special consideration for access network QoS. However, this Recommendation retains the support for home networking applications to establish prioritized data sessions between the CMTS and IPCable2Home Residential Gateway device, using IPCablecom compliant messaging, as specified by ITU-T Rec. J.191. Hence, the necessary requirements to support this functionality in the PS are included in the QoS specifications, as it is from ITU-T Rec. J.191.

The PS acts as a transparent bridge and forwards IPCablecom QoS messaging between the CMTS and IPCablecom applications. Application data is associated to a DOCSIS service flow according to a classifier that is created in the CM interface, based on the information included in the IPCablecom messages (such as RSVP PATH).

Since the PS requirement is to forward IPCablecom QoS messaging, there is no dependency on the NMS to support this function. Therefore, this CQP function remains the same for both DHCP Provisioning Mode and SNMP Provisioning Mode (see 5.5).

IPCable2Home QoS messaging over the HFC or access network is defined by ITU-T Recs J.161 and J.163. As such, the IPCable2Home QoS policy management and admission control functions for access network QoS are also defined by ITU-T Recs J.161 and J.163.

10.3.1.1.5 QoS Forwarding and Media Access requirements

10.3.1.1.5.1 Packet classification requirements

The PS MAY perform packet classification for LAN-LAN traffic.

For WAN-to-LAN traffic, if an outgoing LAN interface implements multiple queues or multiple priorities, then the PS MUST perform packet classification.

In order to perform packet classification, the PS MUST take the following actions:

1) The PS MUST examine the source IP, destination IP, source port, destination port and protocol type values of the packet and find a first matching entry (i.e., lowest index number) in the PS classifier table (cabhQos2TrafficClassTable) stored in the PS database (Annex A).
2) {informative text:The PS MUST use the cabhQos2TrafficClassImpNum MIB value of the matching entry as a UPnP Traffic Importance Number for that packet.
3) If no matching entry is found in the classifier table, then the PS MUST assign the UPnP Traffic Importance Number of 0 to the packet.

10.3.1.1.5.2 Prioritized queuing requirements

}

{informative text:

The PS MUST store the number of queues implemented by each of its interface in the PS database that can be accessed via a cabhQos2PsIfAttribIfNumQueues MIB [see E.7].

The PS MUST map the UPnP TrafficImportanceNumber value of the packet identified during the classification process to IPCable2Home Queuing Priority value as specified in [802.1D] Annex G using the number of queues cabhQos2PsIfAttribIfNumQueues [see E.7] implemented by an interface on which the packet is to be transmitted. The PS MUST queue the packet appropriately on the destination interface according to this mapped IPCable2Home Queuing Priority value.

For each LAN interface, the PS MUST poll various queues on that interface according to their priorities to extract packets out to be transmitted on the shared media. Every time the PS is to extract a packet from the various queues for a particular interface, the PS MUST always start its polling with the highest priority queue first. If the highest priority queue has no packets to be sent, the PS MUST poll the next highest priority queue of the remaining queues in the hierarchy, until it finds the next available highest priority packet to be sent. The PS MUST always extract packets from each queue in the order they arrive.

10.3.1.1.5.3 Prioritized media access requirements

The PS MUST store the number of native layer-2 media access priorities supported by each of its interface in the PS database that can be accessible via a MIB cabhQos2PsIfAttribIfNumPriorities [see E.7].

After the packet is extracted from the queues of a particular interface, the PS MUST map UPnP TrafficImportanceNumber of the packet to IPCable2Home Media Access Priority, as defined in [802.1D] Annex G, using the number of media access priorities supported (cabhQos2PsIfAttribIfNumPriorities) by that interface. The PS MUST transmit the packet through the shared media technology such that its relative preferential access to the media, as required by IPCable2Home Media Access Priority value, is maintained.

}

10.3.1.1.5.4 IPCablecom applications support requirements

The PS MUST act as a transparent bridge and forward IPCablecom [ITU-T Rec. J.161], [ITU-T Rec. J.163] QoS messaging between the CMTS and IPCablecom applications. Application data is associated to a CM service flow according to a classifier that is created in the CM interface, based on the information included in the IPCablecom messages (such as RSVP PATH).

Since the PS requirement for IPCable2Home is to just forward IPCablecom QoS messaging, there is no dependency on the NMS to support this function. Therefore, this CQP function remains the same for both DHCP Provisioning Mode and SNMP Provisioning Mode (see 5.5).

{informative text:

10.3.1.2 UPnP QoS Device Service interface (QDS)

10.3.1.2.1 UPnP QoS Device Service interface goals

- To provide an interface for a UPnP QoS Manager Service to set up home network QoS on the LAN interfaces of the CableHome PS.
- To provide an interface on a CableHome PS to detect the need of access network QoS for a session that spans access network and home network.

10.3.1.2.2 UPnP QoS Device Service interface design guidelines

See Table 10-4.

Table 10-4/J.192 – UPnP QoS Device Service design guidelines

Number	Guidelines
QDS.1	The QDS provides an interface for a UPnP QoS Management Entity to set up QoS on the CableHome PS.
QDS.2	The QDS sets up packet classifiers in the CableHome PS.
QDS.3	The QDS detects the need of an access network QoS negotiation for a session.

10.3.1.2.3 UPnP QoS Device Service interface assumptions

Control Points on the home LAN may utilize the QoS Management Entity on the home LAN (UQM) that is not part of the CableHome PS to set up QoS.

10.3.1.2.4 UPnP QoS Device Service interface description

CableHome PS may optionally implement UPnP QoS Device Service Interface (UQD) on the LAN side. If implemented, UPnP QoS Device Service is advertised as a part of the CableHome PS root device. UPnP QoS Device Service provides an interface so that the UPnP QoS Management Entity on the LAN can configure the CableHome PS with appropriate QoS settings. When a UPnP QoS Management Entity invokes SetUpTrafficQoS action of the QDS on the CableHome PS, the PS sets up packet classifiers using the information in the UPnP TrafficDescriptor that is passed as an input argument to the action. These classifiers are used by QFM for packet classification. If a source or a destination IP address in the UPnP TrafficDescriptor resides on the WAN, then CableHome PS detects that access network QoS is needed for this session as well.

10.3.1.2.5 UPnP QoS Device Service Interface requirements

The PS MAY implement UPnP QoS Device Service (UQD).

If PS implements UPnP QoS Device Service, then it is required to adhere to the following requirements.

The PS MUST advertise UPnP QoS Device Service as an embedded service of CableHome PS root device.

The PS MUST be able to process GetQoSCapabilities action of UPnP QoS Device Service. Upon receipt of this action, the PS MUST return the attributes of its LAN-side interfaces ONLY represented by the ifTable MIB in the QoSDeviceCapabilities output argument of the action, as specified in Table 10-5.

Table 10-5/J.192 – ifTable MIB values and UPnP QoSDeviceCapabilities state variable

IfTable MIB Values	UPnP QoSDeviceCapabilitties state variable XML values
IfIndex	InterfaceId
IfPhysAddress	MacAddress
IfType	IanaTechnologyType
IfSpeed	MaxPhyRate

The PS MUST be able to process GetQosState action of UPnP QoS Device Service. Upon receipt of this action, the PS MUST return all the packet classifiers represented by cabhQos2TrafficClassTable MIB, in the ListOfTrafficDescriptors XML output argument and the total number of packet classifiers in the NumberOfTrafficDescriptors output argument. The PS MUST also return QoSStateId as an output argument.

The PS MUST support the SetupTrafficQos action of UPnP QoS Device Service. Upon receipt of this action, the PS MUST use SetupTrafficDescriptor input argument to establish packet classifier in the PS database that is accessible via cabhQos2TrafficClassTable MIB in the PS.

The PS MUST support the receipt of ReleaseTrafficQos action of UPnP QoS Device Service. Upon receipt of this action, the PS MUST remove a packet classifier entry stored in its database identified by the ReleaseTrafficHandle input argument of the action.

}

{informative text:

10.3.2 PS QoS Policy Server (QPS)

The QoS Policy Server (QPS) in the PS acts as a repository of home network QoS Policies established by a cable operator as well as a home user. The QPS has a WAN side SNMP interface for cable operators to manage the QoS Policies. On the LAN-side, the QPS has UPnP QoS Policy Holder Service (UQPH) interface to communicate QoS policies upon request from a UPnP QoS Manager Entity. The QPS may also have an unspecified LAN interface that enables home users to manage QoS policies on their own. This clause provides the description of the QPS functionality and associated PS requirements.

10.3.2.1 QoS Policy Server goals

- To establish a set of criteria by which applications and services can request and use QoS policies for traffic within the home network.
- To enable both cable operator as well as user configuration of QoS policies in the IPCable2Home PS.
- To provide a mechanism for the cable data network to communicate the desired QoS policies to the PS and then to UPnP QoS Compliant Devices via any QoS Manager within the home network.

10.3.2.2 QoS Policy Server design guidelines

See Table 10-6.

Table 10-6/J.192 – QPS design guidelines

Number	Guidelines
QPS.1	QPS will be provided QoS Policy information from the Network Management Server (NMS) in the Headend as well as from a home user on the LAN side.
QPS.2	QoS Policy information supplied to the QPS may be updated by the headend as well by the home (HomeUser). Updated information is only obtained when any QoSManager makes a request.
QPS.3	QoS Policy information supplied to the QPS from the headend may contain policy rules that cannot be updated by the home user. QPS will use UPnP messaging.
QPS.4	QPS will use a defined messaging content interface (MIB) for providing information of various applications in the home LAN to Network Management Server (NMS) in the headend.

10.3.2.3 QoS Policy Server assumptions

IPCable2Home uses UPnP QoS Messaging to exchange QoS Policy information between the PS and UPnP compliant QoS Entities UPnP Hosts can have more than one defined service or application.

10.3.2.4 QoS Policy Server system description

The QPS maintains a database of traffic QoS policies in the PS. The QPS can receive policy rule information from the headend via the initial configuration file download of the PS, or through a MIB interface in the CMP. The QPS can also receive policy rule information from the home user via a LAN interface (such as an HTTP server) not specified within CableHome 1.1 which must also be represented in the QPS traffic policy database. The QPS will communicate the policy information to any UPnP QoS Manager (UQM) when requested to be used for prioritized media access by applications or services on the LAN.

10.3.2.4.1 WAN information exchange

From the WAN side, the cable operator headend provides to the PS traffic QoS policies in a PS configuration file, or by using a SNMP MIB interface. The NMS, at the headend, can read and update (change/modify/delete) these traffic QoS policies in the PS database using a SNMP MIB interface.

10.3.2.4.1.1 IPCable2Home QoS Policy information from the WAN to the PS

The headend provides the PS with a list of ordered traffic QoS policies that a cable operator wishes applications and services to use. This information is supplied to the PS through a configuration file at the time of PS initialization, or via SNMP SET commands from the headend. The PS stores this information in the PS database that is accessible via a MIB table cabhQos2PolicyTable.

PS can also receive requests from the NMS to update (add/modify/delete) traffic QoS policies for applications and services in its policy table using SNMP. In response to these requests, the PS updates (add/modify/delete) the policy information in the PS database that can be accessed via cabhQos2PolicyTable MIB using SNMP interface. See Figure 10-3.

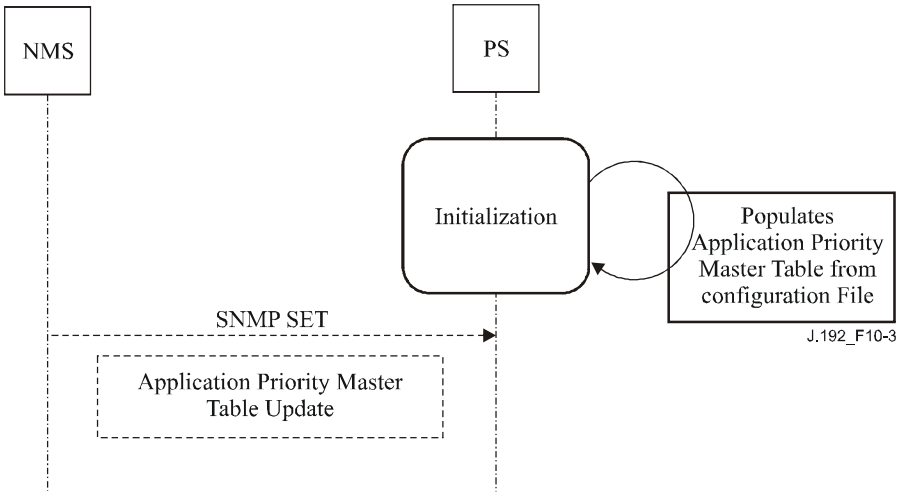


Figure 10-3/J.192 – WAN information exchange and processing at the PS

10.3.2.4.2 UPnP QoS Policy Holder service interface (QPH) for LAN

On the LAN side, the UPnP QoS Policy Holder service interfaces directly to any UPnP QoS Manager for the retrieval of a traffic policy for a specific traffic descriptor. In response to the UPnP QoS Manager request for a traffic policy, the QPH service will return the TrafficImportanceNumber, UserImportanceNumber, and AdmissionPolicyEnabled values for the corresponding traffic descriptor. If a match is not found, the QPS returns 0 for the TrafficImportanceNumber and 0 for the UserImportanceNumber.

10.3.2.5 QoS Policy Server requirements

10.3.2.5.1 WAN information exchange requirements

The PS MUST store a list of traffic policies, provided by a cable operator, in the PS database, that is accessible via the cabhQos2PolicyTable SNMP MIB interface. The PS MUST support updates (add/modify/delete) to this table cabhQos2PolicyTable through a configuration file at the time of PS initialization, or via SNMP SET commands from the headend.

10.3.2.5.2 UPnP QPH service requirements

The PS MUST support at least (32) 'operatorOnly' row entries in the cabhQos2PolicyTable.

The PS MUST support at least (32) 'homeUser' row entries in the cabhQos2PolicyTable.

The PS MUST support at least (32) 'operatorForHomeUser' row entries in the cabhQos2PolicyTable.

The PS must support the 'GetTrafficPolicy' action of UPnP QoS Policy Holder service (UQPH).

In response to an UPnP QPH service GetTrafficPolicy action, the PS MUST perform the following functions.

1) The PS MUST find the first match for a QoS Policy Rule in the cabhQos2PolicyTable. The PS MUST process the rules based on the SNMP table index and cabhQos2PolicyRuleOrder MIB. That is, the PS MUST process all the cabhQos2PolicyTable:'operatorOnly' row entries first, followed by 'homeUser' entries, and then the 'operatorForHomeUser' entries. In addition, within these individual "owner" entries, the PS processes entries with the lowest numerical value of cabhQos2PolicyRuleOrder.
2) After the PS finds a first matching row entry in cabhQos2PolicyTable, the PS MUST return the value of cabhQos2PolicyTraffImpNum MIB [see E.7] as a "trafficImportanceNumber" output argument of GetTrafficPolicy action.
3) The PS MUST also return the value of cabhQos2PolicyAdmissionPolicyEnable MIB object as admissionPolicyEnable output argument of the GetTrafficPolicy action.
4) The PS MUST also return the value of cabhQos2PolicyUserImportanceNumber MIB object as "userImportanceNumber" output argument of GetTrafficPolicy action.
5) If there is no matching entry found in the policy table, then the PS MUST return 0 for the "trafficImportanceNumber" as well as for "userImportanceNumber" output arguments. In this case, the PS MUST return the value of cabhQos2PolicyAdmissionPolicyEnable MIB object as admissionPolicyEnable output argument of the GetTrafficPolicy action. In addition, the PS MUST create a policy rule entry of type "UPnP" in the cabhQos2PolicyTable MIB with UPnP traffic descriptor sent in the GetTrafficPolicy action. The PS MUST set the cabhQosPolicyTraffImpNum for such an entry to 0. The user or the operator can change such entries and, in that case, the PS MUST change the entry to either "homeUser" or "operatorForHomeUser", respectively. The PS MUST NOT change the entries of type "upnp" to "operatorOnly".}

{informative text:

10.3.3 UPnP QoS Manager Service (QM)

10.3.3.1 UPnP QoS Manager Service goals

To ensure that there is at least one UPnP QoS Manager Service for UPnP compliant Control Points to request QoS on the home LAN.

10.3.3.2 UPnP QoS Manager Service design guidelines

The CQP implements the complete UPnP QoS Manager functionality per UQM.

10.3.3.3 UPnP QoS Manager Service assumptions

There may be other UPnP QoS Managers within the home LAN.

10.3.3.4 UPnP QoS Manager Service system description

The PS implements UPnP QoS Manager Service exactly as defined in UQM. The UPnP QoS Manager Service in the PS does not have an interface or controllability from the WAN since it is autonomous.

10.3.3.5 UPnP QoS Manager Service requirements

The PS MUST implement the complete UPnP QoS Manager functionality per UQM.

The PS MUST announce the QoS Manager Service as a part of the CableHome PS root device.

The PS MUST support the RequestTrafficQos action of the UPnP QoS Manager Service.

When a UPnP Control Point invokes the QM:RequestTrafficQos action of the PS, the PS MUST perform the following:

1) As per UQM, if the PS finds multiple UPnP QoS Policy Holder Service instances, the PS MUST use the UPnP QoS Manager default policy table and the PS MUST return the UPnP TrafficImportanceNumber based on that default table.
2) If the PS finds only the UPnP QoS Policy Holder Service resident in the PS, then the PS MUST use the UPnP TrafficImportanceNumber, UserImportanceNumber and AdmissionPolicyEnable values that are extracted from cabhQosPolicyTable MIB when invoking the QD:SetUpTrafficQos action on various UPnP QoS Device Service instances on the LAN.
3) If the PS QoS Policy Holder Service is disabled and the PS finds a sole UPnP QoS Policy Holder external to the PS, the PS MUST call the QPH:GetTrafficPolicy action to that QoS Policy Holder Service and the PS MUST use UPnP TrafficImportanceNumber, UserImportanceNumber and AdmissionPolicyEnable values that are returned.
4) The PS MAY call QD:GetPathInformation action on QoS Device Service (UQD) instances on the LAN.
5) To distribute the UPnP TrafficImportanceNumber, the PS MUST call QD:SetUpTrafficQos action on the QoS Device Service instance on the source and MAY call QD:SetUpTrafficQos action on other QoS Device Service instances on the LAN.

The PS MUST support the UpdateTrafficQos action of the UPnP QoS Manager Service.

When a UPnP Control Point invokes the QM:UpdateTrafficQos action of the PS, the PS MUST perform the following:

1) The PS MUST first invoke the QD:ReleaseTrafficQos action on the source QoS Device Service instance and MAY invoke the QD:ReleaseTrafficQos action on other QoS Device Service instances on the LAN.
2) The PS then MUST invoke QD:SetUpTrafficQos on the source QoS Device Service instance and the PS MAY invoke QD:SetUpTrafficQos on the QoS Device Service instances on the path with the updated Traffic descriptor provided by the Control Point in the QM:UpdateTrafficQos action.

The PS MUST support the ReleaseTrafficQos action of the UPnP QoS Manager Service.

When a UPnP Control Point invokes QM:ReleaseTrafficQos action on the PS, the PS MUST invoke QD:ReleaseTrafficQos action on the source QoS Device Service instance and MAY call it on other QoS Device Service instances on the LAN.

The PS MUST support the BrowseAllTrafficDescriptors action of the UPnP QoS Manager Service.

When a UPnP Control Point invokes the QM:BrowseAllTrafficDescriptors action on the PS, the PS MUST call the GetQosState action on all known QoS Device Service instances and return the information to the Control Point.

10.3.4 QoS functionality of PS Control Point (QCP)

The QoS functionality of PS Control Point (QCP) acts as the Control Point for all the UPnP QoS service instances in the home network. It performs the functions associated with a Control Point in terms of device and service discovery, description and control. Another important function of PS

QoS Control Point is to implement the logic to set up Access Network QoS in response to Home Network QoS requests via UPnP QoS; and to set up Home Network QoS in response to Access Network QoS requests via CH-PCMM interface.

10.3.4.1 PS Control Point QoS functionality goals

- To enable cable operators to collect information about various QoS capable UPnP devices and services on the home LAN.
- To enable CableHome PS to request home network QoS using UPnP QoS Messaging.

10.3.4.2 PS Control Point QoS functionality system design guidelines

See Table 10-7.

Table 10-7/J.192 – PS Control Point QoS functionality system design guidelines

Number	Guidelines
QPSCP.1	PS Control Point will implement the control logic consistent with UPnP QoS.
QPSCP.2	PS Control Point may implement the intelligence to request LAN QoS (via UPnP QoS) in response to Access Network QoS request via PCMM interface.

10.3.4.3 PS Control Point QoS functionality assumptions

Home LAN consists of UPnP Host Devices that implement QoS functionality consistent with UPnP-QoS.

10.3.4.4 PS Control Point QoS functionality system description

QoS Functionality of PS Control Point consists of the following components:

- QoS Discovery Logic
- CableHome Qos Broker Logic

10.3.4.4.1 QoS discovery logic

QoS Discovery Logic is responsible for discovery and description of all UPnP QoS entities on the home LAN.

PS Control Point acts as a UPnP Control Point responsible for discovery, description, control, etc. as per the UPnP 1.0 Device Architecture (UDA 1.0). The QoS functionality of the PS Control Point specifically is responsible for executing UPnP QoS Action Calls as necessary on the UPnP QoS service instances on the home network.

CableHome defines cabhPsDevUpnpCommandUpdate MIB to request the PS to update its cabhPsDevUpnpInfoTable as per constraints specified in cabhPsDevUpnpInfoIp and cabhPsDevUpnpCommand.

If cabhPsDevUpnpCommand MIB is set to qosDeviceCapabilities and cabhPsDevUpnpCommandUpdate is set to true, the Qos functionality of PS CP invokes QD:GetQosCapabilities action on IP address specified in cabhPsDevUpnpInfoIp. The PS stores the device capability information returned by the QoS Device in the PS database and it is accessible via the cabhPsDevUpnpInfoTable MIB.

If cabhPsDevUpnpCommand MIB is set to qosDeviceState and cabhPsDevUpnpCommandUpdate is set to true, the Qos functionality of PS CP invokes QD:GetQosState() action on IP address specified in cabhPsDevUpnpCommandIp. The PS stores the device capability information returned by the QoS Device in the PS database, and it is accessible via the cabhPsDevUpnpInfoTable MIB.

10.3.4.4.2 CableHome QoS Broker logic

CableHome QoS Broker logic of the PS control point is responsible for setting up QoS on the home network in response to a request from CH-PCMM interface.

As explained in 10.2, QoS Architecture, the CableHome QoS Broker entity may implement CH-PCMM interface on the WAN-side to negotiate access network QoS for the streams that span home network and access network. The PS may receive a request for home network QoS from the CH-PCMM Interface. The PS control point is responsible for setting up QoS over on the home network in response to such a request from the CH-PCMM interface. It should be noted that the actual requirements for CH-PCMM interface and setting up QoS on the home network in response to requests from PCMM interface will be defined in the future CableHome specifications.

10.3.4.5 PS Control Point QoS functionality requirements

1) When cabhPsDevUpnpCommand MIB [see E.4] is set to qosDeviceCapabilities and cabhPsDevUpnpCommandUpdate [see E.4] is set to true, the PS MUST invoke QD:GetQosCapabilities() action on IP address specified in cabhPsDevUpnpInfoIp [see E.4].
2) The PS MUST store the device capability information returned by the QoS Device in response to QD:GetQosCapabilities() action in the PS database, which can be accessed using cabhPsDevUpnpInfoTable MIB [see E.4].
3) When cabhPsUpnpCommand MIB is set to qosDeviceState and cabhPsDevUpnpCommandUpdate is set to true, the PS MUST invoke QD:GetQosState() action on IP address specified in cabhPsDevUpnpInfoIp Address.

The PS MUST store the device capability information returned by the QoS Device Service in response to the QD:GetQosState() action in the PS database, which can be accessed using cabhPsDevUpnpInfoTable MIB.}

11 Security

11.1 Introduction/Overview

This clause defines the security interfaces, protocols and functional requirements needed to secure the PS and its operations.

The delivery of reliable multimedia IP services to client devices on a home network requires a secure residential gateway along with the security mechanisms to protect these services from illegal access, monitoring and disruption. The purpose of any security technology is to protect value, including revenue-based services. Threats to a revenue stream exist when a user of the network perceives the value, expends effort and money, and invents a technique to get around making the necessary payments (see Annex C). Some network users will go to extreme lengths to steal when value is perceived. The addition of security technology to protect value has an associated cost; the more money expended, the greater the security (security effectiveness is thus basic economics).

The security architecture focuses on securing the LAN from network attacks as well as securing communications between the PS and the Headend servers. The PS functionality can provide the foundation for other applications and services served by the cable operator to the home LAN. Security can exist for these applications independent of the IPCable2Home security architecture. IPCablecom specifies interfaces for a multimedia applications and has its own security architecture. For all references to IPCablecom security, please refer to [ITU-T Rec. J.170].

11.1.1 Goals

The goals for the security model include:

- Employ a cost-effective security technology to force any user with the intent to steal or disrupt network services to spend an unreasonable amount of money or time.
- Secure the IPCable2Home network used to offer high value cable-based services so that it is at least as secure as the CableModem and IPCablecom technologies on the hybrid fibre-coax (HFC) network.
- Where possible, align security mechanisms with the CableModem and IPCablecom security Recommendations.
- From the LAN, it is the intent of security architecture to assist the operator with a secure identity to make it hard for the average subscriber to gain unauthorized access to the HFC network and cable-based services.

11.1.2 Assumptions

The assumptions for the security environment include:

- It is assumed the Embedded PS has a J.112 or J.122 cable modem.
- The home network includes less security for low value services.
- Backoffice configurations are not specified and IPCable2Home assumes minimal configurations by the cable operator to operate in the specified modes.

11.2 Security architecture

The security architecture is based on the architecture as defined in Reference Architecture, clause 5. The architecture defines a Portal Services (PS) element, which includes Management, Provisioning, Security and QoS functions.

The architecture also includes the following set of Headend elements: Cable Modem Termination System (CMTS), Dynamic Host Configuration Protocol (DHCP) [RFC 2131] server, Network Management System, Trivial File Transfer Protocol (TFTP) server in the cable network, TFTP client in the PS, Hypertext Transfer Protocol (HTTP) server in the cable network, HTTP client in the PS, Transport Layer Security (TLS) [RFC 2246] server in the cable network, TLS client in the PS, and a Key Distribution Centre (KDC) server in the cable network.

The security architecture focuses on securing the LAN from network attack, as well as securing communications between the PS and the Headend servers.

11.2.1 System design guidelines

The security design requirements are listed below in Table 11-1. This list has provided guidance for the development of the security architecture.

Table 11-1/J.192 – Security system design guidelines

Reference	Guidelines
SEC1	Include the design necessary to communicate the authentication credentials for elements.
SEC2	Authentication credentials for PS and critical back office servers will be provided. The credentials will define specific usage and ensure a source of trust.
SEC3	Network management messages between the cable Headend and PS can be authenticated and optionally encrypted to protect against unauthorized monitoring and control.

Table 11-1/J.192 – Security system design guidelines

Reference	Guidelines
SEC4	The firewall will accept configuration files in a standard language and format. (Note)
SEC5	The cable operator will have the ability to remotely manage compliant firewall products through configuration file or SNMP commands.
SEC6	The firewall will include a default set of rules for an expected minimum set of functionality.
SEC7	Provide the necessary support for IPCablecom through the firewall.
SEC8	A minimum set of requirements will be placed on the firewall filtering capabilities for packet, port, IP addresses and time of day.
SEC9	A detailed firewall event logging interface will allow the cable operator to monitor and review firewall activity as configured.
SEC10	The firewall will support commonly used applications in specific scenarios.
SEC11	The firewall will protect the LAN and WAN from common network attacks.
SEC12	The management of the events and rulesets for the firewall will be defined in detail via the Security MIB.
SEC13	The cable operator will have the ability to securely download software images to the PS element.
SEC14	The cable operator will have the ability to authenticate and optionally encrypt the transport of configuration files for the PS or firewall.
NOTE – The Firewall Configuration File Requirements are defined in 7.4, PS Function – Bulk Portal Services Configuration (BPSC).	

This clause limits the scope of the specified security architecture to meet these primary system security requirements. However, it is acknowledged that in some cases additional security is desired and can be added by the cable operator, as needed. The concerns of individual cable operators or manufacturers can result in additional security protections. This Recommendation does not restrict the use of further protections, as long as they do not conflict with the intent and guidelines of this specification.

11.2.2 System description

The security architecture includes the following security elements:

- Security-Domain.
- Portal Services function (PS).
- Cable Security Portal function (CSP).
- Firewall (FW).
- Key Distribution Centre (KDC).
- HTTPS Server with TLS.

The architecture defines the PS Element within the residential gateway. Security exists only in a few of the specified interfaces, as the System Design Guidelines require. Figure 11-1 illustrates the relationship between the various elements which contain security.

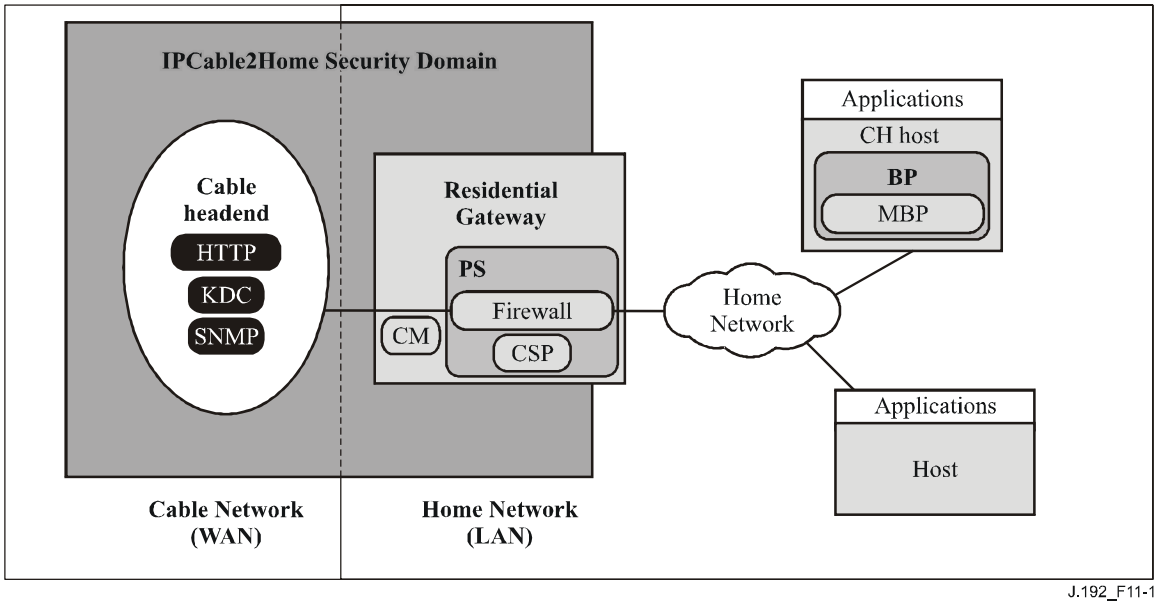


Figure 11-1/J.192 – IPCable2Home security elements

11.2.2.1 Security Domain

The Security Domain is defined in Figure 11-1 and encompasses the PS element in the residential gateway and the illustrated Headend servers, with specified security. The Security Domain defines the boundary of the sphere of direct influence where security functionality is extended to the residential gateway from the cable network's Headend. The PS element is wholly within the Security Domain, with the exception of the LAN side USFS functionality. The CSP and Firewall act as the boundary elements between the Security-Domain and the non-secure domain.

11.2.2.2 PS related security sub-elements

The PS includes the following security elements:

- Cable Security Portal (CSP).
- Firewall (FW).

The CSP acts as a security portal for other PS sub-elements such as negotiating the SNMPv3 keys either through Diffie-Helman or Kerberos, as required. The CSP ensures there is security for SNMPv3 between the NMS and the PS, when turned on by the cable operator. The CSP provides the ability to validate and verify digital certificates for the purposes of authentication and encryption. The CSP initiates, manages, and closes a TLS session for secure downloading of the PS configuration file and firewall configuration file, if instructed by the cable operator during the DHCP exchange.

The PS firewall functionality provides protection to the user, as well as the HFC network, from unwanted traffic coming from the WAN, LAN, or PS address realms. Such traffic can include deliberate attacks on the in-home network, as well as traffic limiting for parental control applications. The security requirements include specific rules for remote management by the cable operator.

11.2.2.3 Key Distribution Centre (KDC) server

The Key Distribution Centre (KDC) server is required if the cable operator deploys IPCable2Home with SNMP provisioning mode. If a KDC server is available in the Headend, it will be used to provide mutual authentication and key distribution services with the use of the Kerberos protocol. If available, the KDC will communicate with the CSP function to establish these services.

11.3 PS device authentication infrastructure

This clause describes authentication for the PS device and its communication to the KDC and the HTTPS server.

11.3.1 Device authentication infrastructure goals

It is important to establish the secure identity of the PS element to assist with the following goals:

- Reduce the possibility of device and software cloning, as well as theft of service. The gateways are in a widely distributed environment where the consumer has in-home physical access to the gateway. Providing a secure identity reduces risk of tampering with the gateway hardware device.
- Establish the source of trust. The PKI provides an established source of trust which is rooted within the Manufacturer base.

11.3.2 Authentication infrastructure system design guidelines

See Table 11-2.

Table 11-2/J.192 – Authentication infrastructure system design guidelines

Reference	Guidelines
SEC1	Include the design necessary to communicate the authentication credentials for IPCable2Home elements.
SEC2	Authentication credentials for CPE and critical backoffice servers will be provided. The credentials will define specific usage and ensure a source of trust.

11.3.3 Authentication infrastructure system description

For security purposes, it is important to know with whom you are communicating prior to exchanging any meaningful information. Authentication provides a secure identity. There are three parts to authentication: the identity credential, the checking of the identity credential for validity, and the common means to securely communicate the identity information. An industry standard identification credential, X.509 certificates, in conjunction with [RFC 3280] for certificate use, and Kerberos, which is a common communications protocol for mutual authentication is specified. X.509 certificates are exchanged between the PS Element and the KDC during the Kerberos PKINIT exchange, which is wrapped in the AS Request and AS Reply messages. The PS Element Certificate provides the identity of the associated PS Element by cryptographically binding the PS Element WAN-Man MAC address to a public key certificate. Each side validates the information in the certificate and verifies the certificate chain back to the root for each chain. Once the trust has been established, the information for the SNMPv3 keys is sent from the KDC to the PS Element. This authentication clause describes the use of Kerberos and X.509 certificates.

11.3.4 Authentication infrastructure requirements

11.3.4.1 Element authentication via Kerberos

Authentication is specified when a KDC that supports IPCable2Home is available in the Headend. If a KDC is available, it is recommended that the cable operator provision the PS Element in SNMP Provisioning Mode (as described in 5.5), to take advantage of the specified mutual authentication protocol with the use of Kerberos, using the PKINIT extension. Kerberos provide a protocol to secure mutual authentication in order to provide keying material and communication establishment only between authenticated parties on the IPCable2Home network. Because this authentication model has been specified by another ITU project, i.e., IPCablecom, IPCable2Home references the IPCablecom model when appropriate.

Various Kerberos MIB objects are required by IPCablecom. Some MIB objects to cover the Kerberos functionality needed by IPCable2Home have been defined. These MIB objects are defined in the Security MIB and described in the MIB Object clauses.

Communication between the KDC and PS is initiated by the PS immediately after the DHCP options are processed during provisioning, if the DHCP options require the PS to initiate communication to the KDC. The DHCP options specified in 7.3.3.2.4 require option 122, sub-option 10, which contains the value for the KDC's IP Address to be included with the other DHCP options, and MUST be used by the PS to establish communication between the PS and KDC. Even though IPCablecom requires a DNS name as part of the DHCP options, DNS is not required for IPCable2Home and, therefore, the IP address of the KDC is required for the PS to be able to find the appropriate KDC.

11.3.4.1.1 Kerberos/PKINIT

When the PS Element is provisioned in SNMP Provisioning Mode, the use of Kerberos with the PKINIT public key extension for authenticating IPCable2Home elements and supporting key management requirements is specified. IPCable2Home elements (clients) authenticate themselves to the KDC with the PKINIT protocol. Once authenticated to the KDC, clients will receive a Kerberos ticket for authenticating themselves to a particular server.

In SNMP provisioning mode, the PS Element, the NMS (i.e., SNMP Manager) and KDC MUST follow the specification for Kerberos/PKINIT, as defined in 6.4 and 6.5 of ITU-T Rec. J.170, unless otherwise noted in this Recommendation. The IPCable2Home KDC is equivalent to or the same as the IPCablecom 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. This SNMP functionality in general is 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 and 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 key management, as well as for device authentication. The certificates used in PKINIT for IPCable2Home are specified in the PKI clause (see 11.3.4.2). Where IPCablecom specifies an MTA device certificate, IPCable2Home provides a certificate for the PS Element (PS Element Certificate), and implementations of PS Elements MUST include the PS Element Certificate.

The following clauses for Kerberos functionality from [ITU-T Rec. J.170] do not apply to IPCable2Home:

- Clause 6.4.2.1.3, Pre-authenticator for provisioning server location.
- Clause 6.4.5, Kerberos server locations and naming conventions.
- Clause 6.4.6, MTA principal names.
- Clause 6.4.7, Mapping of MTA MAC address to MTA FQDN.
- Clause 6.4.9, Service key versioning.
- Clause 6.5.2.1, Rekey messages.
- Clause 6.5.3, Kerberized IPSec.

11.3.4.1.2 IPCable2Home specific authentication variables

The model IPCablecom specifies some specific variable names for Kerberos in the IPCablecom Network Architecture. In order for IPCable2Home to use the IPCablecom model, the following variable names MUST be changed:

- Replace pktcKdcToMtaMaxClockSkew as defined in the IPCablecom Security Spec, with KdcToClientMaxClockSkew.
- Replace pktcSrvrToMtaMaxClockSkew as defined in the IPCablecom Security Spec, with SrvrToClientMaxClockSkew.
- Replace mtaprovsrvr as defined in the IPCablecom Security Specification, with provsrvr.

IPCable2Home Kerberos implementations MUST ignore the Object Identifier (OID) field portion, which reads clabProjIPCablecom (2), within the AppSpecificTypedData, within the KRB-ERROR messages.

11.3.4.1.3 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 Annex B/J.170.

The KDC conventions listed in 6.4.5.2/J.170 are considered informative with the expectation that the KDC will perform the necessary functions in the backoffice 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 "PS" (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.

A NMS Element principal name MUST be of type NT-SRV-HST with exactly two components, where the first component MUST be the string "provsrvr" (not including the quotes), and the second component MUST be the service provider's SNMP entity address:

provsrvr/<SNMP entity address>

The <SNMP manager address> MUST be the service provider's SNMP manager IP address (CDC DHCP option 122, sub-option 3) in dotted notation enclosed in square brackets (e.g., [12.34.56.78]).

11.3.4.2 Public Key Infrastructure (PKI)

Public key certificates, which comply with the X.509 specification and the IETF [RFC 3280] are used.

11.3.4.2.1 Generic certificate requirements

This clause describes what is commonly referred to as the generic structure, since all certificates share these requirements. All certificates specified in this clause MUST include the following information:

- **Certificate Version** – The Version of the certificates MUST be [ITU-T Rec. X.509], v3, and noted as v2 in the actual certificate. All certificates MUST comply with [RFC 3280], except where the non-compliance with the RFC is explicitly stated in this clause. Any

non-compliance request by this Recommendation for content does not imply non-compliance for format. Any specific non-compliance request for format will be explicitly described.

- **Public Key Type** – RSA Public Keys are used throughout the certificate hierarchies described in 11.3.4.2.2. The subjectPublicKeyInfo.algorithm OID used MUST be 1.2.840.113549.1.1.1 (rsaEncryption). The public exponent for all RSA keys MUST be F_4 – 65537.
- **Extensions** – The extensions (subjectKeyIdentifier, authorityKeyIdentifier, KeyUsage, and BasicContraints) MUST follow [RFC 3280]. All other certificate extensions, if included, MUST be marked as non-critical. The encoding tags are [c:critical, n:non-critical; m:mandatory, o:optional] and are identified in the table for each certificate.
- **subjectKeyIdentifier** – The subjectKeyIdentifier extension included in all certificates as required by [RFC 3280] (e.g., all certificates except the device and ancillary certificates) MUST include the keyIdentifier value composed of the 160-bit SHA-1 hash of the value of the BIT STRING subjectPublicKey (excluding the tag, length and number of unused bits from the ASN.1 encoding) (see [RFC 3280]).
- **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.
- **KeyUsage** – The keyUsage extension MUST be used for all Certification Authority (CA) certificates and Code Verification Certificates (CVCs). For CA certificates, the keyUsage extension MUST be marked as critical with a value of keyCertSign and cRLSign. For CVC certificates, the keyUsage extension MUST be marked as critical with a value of digitalSignature and keyEncipherment. The end-entity certificates MAY use the keyUsage extension as listed in [RFC 3280].
- **BasicConstraints** – The basicConstraints extension MUST be used for all CA and CVC certificates and MUST be marked as critical. The values for each certificate for basicConstraints MUST be marked as specified in the certificate description Tables 11-3 through 11-14.
- **Signature Algorithm** – The signature mechanism used MUST be SHA-1 [FIPS 186] with RSA Encryption. The specific OID is 1.2.840.113549.1.1.5.
- **SubjectName and IssuerName** – If a string cannot be encoded as a PrintableString, it MUST be encoded as a UTF8String (tag [UNIVERSAL 12]).

When encoding an X.500 Name:

- Each RelativeDistinguishedName (RDN) MUST contain only a single element in the set of X.500 attributes.
- The order of the RDNs in an X.500 name MUST be the same as the order in which they are presented in this Recommendation.
- **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.

11.3.4.2.2 Certificate hierarchies

There are three distinct certificate hierarchies used:

1) Manufacturer Chain is used to identify authorized manufacturers;

2) Code Verification Chain is used to identify compliant software images;

3) Service Provider Chain is used to identify devices on the Service Provider's network for mutual authentication to the subscriber's devices.

The certificate hierarchies described in this Recommendation can apply to all related projects needing certificates. Each project can adopt this hierarchy as there is an opportunity to move to a more generic, shared certificate structure. Also, each project can 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 project. There can be differences in the end-entity certificates required for each project. However, in the cases where end-entity certificates overlap, one end-entity certificate could be used for several services within the cable infrastructure. For example, IPCablecom requires a KDC for the service provider and 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 IPCable2Home. In this case, the IPCable2Home KDC is equivalent to the IPCablecom MSO KDC (IPCablecom specifies the use of several KDCs).

In Figure 11-2, the term Certification Authority is abbreviated as CA and Code Verification Certificate is abbreviated as CVC.

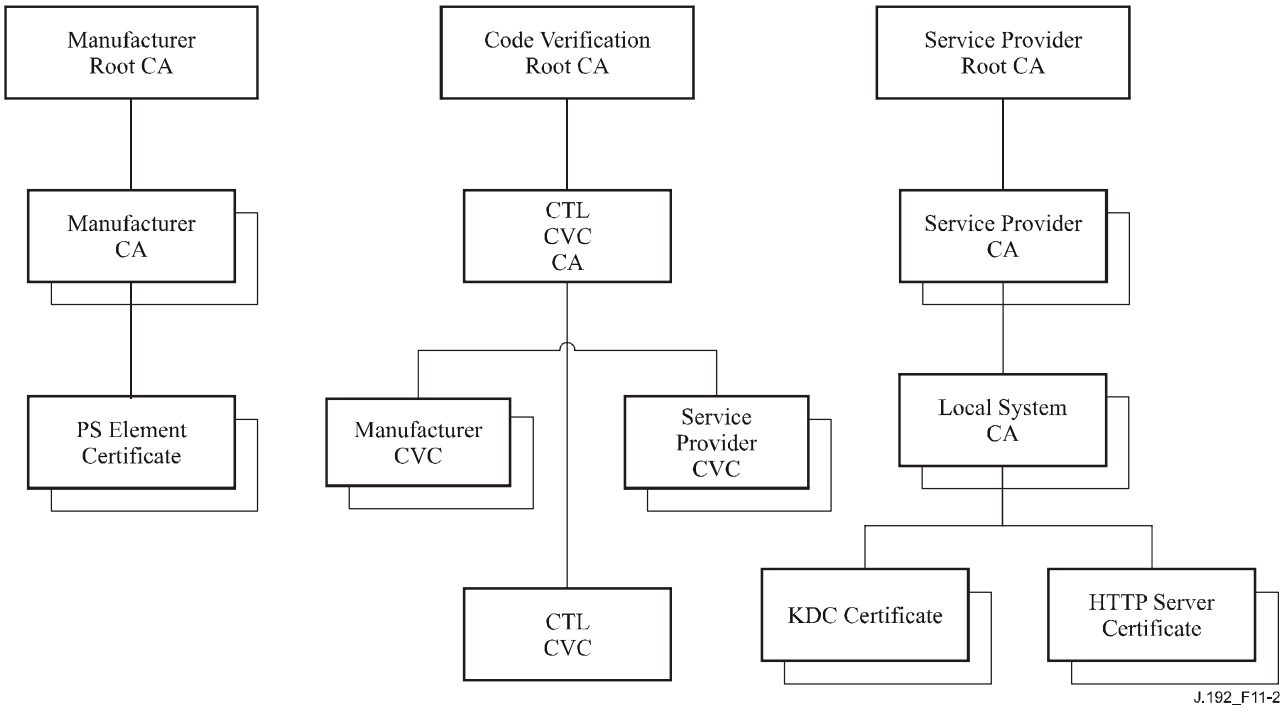


Figure 11-2/J.192 – IPCable2Home certificate hierarchy

11.3.4.2.2.1 Manufacturer certificate hierarchy

The Manufacturer certificate hierarchy, or Manufacturer chain, is rooted at a Manufacturer Root CA, which is used to issue Manufacturer Certification Authority (CA) certificates for a set of authorized manufacturers. Manufacturers request PS Element Certificates from a first-tier CA (e.g., a Manufacturer CA or the Hosted Manufacturer CA). This chain is used for authentication of devices in the home.

The information contained in the following tables are the specific values for the required fields according to [RFC 3280]. These specific values for the Manufacturer Certificate hierarchy MUST be followed according to Tables 11-3, 11-4, 11-5 and 11-6. If a required field is not specifically listed in the tables, then the guidelines in [RFC 3280] MUST be followed. The generic extensions MUST also be included as specified in PKI, see 11.3.4.2.

Manufacturer Root CA Certificate

The Manufacturer Root CA Certificate (see Table 11-3) MUST be verified as part of the certificate chain containing the Manufacturer Root CA Certificate, Manufacturer CA Certificate and the PS Element Certificate.

Table 11-3/J.192 – Manufacturer Root CA Certificate

Subject Name Form	C=<country> O=<Company Name> CN=[Company Name] Manufacturer Root CA
Intended Usage	This certificate is used to issue Manufacturer CA Certificates.
Signed By	Self-Signed
Validity Period	20 to 30 years
Modulus Length	2048
Extensions	KeyUsage [c,m] (keyCertSign, cRL Sign), subjectKeyIdentifier [n,m], basicConstraints [c,m](cA=true)

Manufacturer CA Certificate

If the manufacturer is issued a CA Certificate and it is used to issue the PS Element Certificate it MUST be verified as part of a certificate chain containing the Manufacturer Root CA Certificate, the Manufacturer CA Certificate, and the PS Element Certificate.

The state/province, city, and manufacturer's facility are optional attributes. A manufacturer MAY have more than one manufacturer's CA certificate. If a manufacturer is using more than one manufacturer CA certificate, the PS element MUST have access to the appropriate certificate as verified by matching the issuer name in the PS Element Certificate with the subject name in the Manufacturer CA Certificate. The authorityKeyIdentifier of the PS Element Certificate MUST be matched to the subjectKeyIdentifier of the manufacturer certificate as described in [RFC 3280].

Table 11-4/J.192 – Manufacturer CA Certificate

Subject Name Form	C=<country> O=<CompanyName> [ST=<state/province>] [L=<city>] OU= <organization unit> [OU=<Manufacturer's Facility>] CN=<CompanyName> Mfg CA
Intended Usage	This certificate is issued to each Manufacturer by a certificate authority Manufacturer Root CA and can be provided to each PS Element either at manufacture time, or during a field code update. This certificate appears as a read-only parameter in the PS element. This certificate issues PS Element Certificates. This certificate, along with the Manufacturer Root CA Certificate and the PS Element Certificate, is used to authenticate the PS element identity. The optional listing for manufacturer's facility can be the facility name and/or facility location.
Signed by	Hierarchy's Manufacturer Root CA
Validity Period	20 years
Modulus Length	2048
Extensions	keyUsage[c,m](keyCertSign, cRLSign), subjectKeyIdentifier [n,m], authorityKeyIdentifier [n,m], basicConstraints[c,m](cA=true, pathLenConstraint=0)

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

Hosted Manufacturer CA Certificate

When the Hosted Manufacturer CA Certificate is used to issue the PS Element Certificate, it MUST be verified as part of a certificate chain containing the Manufacturer Root CA Certificate, the Hosted Manufacturer CA Certificate, and the PS Element Certificate.

The state/province, city, and manufacturer's facility are optional attributes. The authorityKeyIdentifier of the PS Element Certificate MUST be matched to the subjectKeyIdentifier of the Hosted Manufacturer CA Certificate as described in [RFC 3280].

Table 11-5/J.192 – Hosted Manufacturer CA Certificate

Subject Name Form	C=<country> O=<CompanyName> [ST=<state/province>] [L=<city>] OU=<CA Identifier>] CN=<CompanyName> Mfg CA
Intended Usage	This certificate is issued by the Manufacturer Root CA and can be provided to each PS Element either at manufacture time, or during a field code update. This certificate appears as a read-only parameter in the PS element. This certificate issues PS Element Certificates. This certificate, along with the Manufacturer Root CA Certificate and the PS Element Certificate, is used to authenticate the PS element identity.
Signed by	Manufacturer Root CA
Validity Period	20 years
Modulus Length	2048
Extensions	keyUsage[c,m](keyCertSign, cRLSign), subjectKeyIdentifier [n,m], authorityKeyIdentifier [n,m], basicConstraints[c,m](cA=true, pathLenConstraint=0)

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

PS Element Certificate

The PS Element Certificate MUST be verified as part of a certificate chain containing the Manufacturer Root CA Certificate, Manufacturer CA Certificate or Hosted Manufacturer CA Certificate, and the PS Element Certificate.

The state/province, city, product name and manufacturer's facility are optional attributes.

The PS Element WAN-Man MAC address MUST be expressed as six pairs of hexadecimal digits separated by colons, e.g., "00:60:21:A5:0A:23". The Alpha HEX characters (A-F) MUST be expressed as uppercase letters.

A PS Element Certificate is permanently installed and not renewable or replaceable. Therefore, the PS Element Certificate has a validity period greater than the expected operational lifetime of the specific device.

Table 11-6/J.192 – PS Element Certificate

Subject Name Form	C=<country> O=<Company Name> [ST=<state/province>] [L=<city>] OU=<organization unit> [OU=<Product Name>] [OU=<Manufacturer's Facility>] CN=<WAN-Man MAC Address>
Intended Usage	This certificate is issued by the Manufacturer CA and installed in the factory. The NMS server cannot update this certificate. This certificate appears as a read-only parameter in the PS Element. This certificate is used to authenticate the PS element identity.
Signed By	Manufacturer CA
Validity Period	20+ years
Modulus Length	1024, 1536, 2048
Extensions	keyUsage[c,m](digitalSignature, keyEncipherment), authorityKeyIdentifier [n,m]

11.3.4.2.2.2 Code Verification Certificate hierarchy

The Code Verification Certificate (CVC) hierarchy, or code verification chain, is rooted at a Code Verification Root CA, which issues the Code Verification CA certificate. The Code Verification CA is used to issue CVCs to a set of authorized manufacturers and service providers. Code Verification CA also issues the CVC. This chain is specifically used to authenticate software downloads. The IPCable2Home PKI allows for Manufacturer CVCs, a CVC and Service Provider CVCs.

CableLabs will be responsible for registering names of authorized CVC subscribers. It is the responsibility of the CableLabs Code Verification 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:

- The organization name used to identify itself as a code co-signer agent in a CVC MUST be assigned by CableLabs.
- The name MUST be a printable string of eight hexadecimal digits that uniquely distinguishes a code-signing agent from all others.
- Each hexadecimal digit in the name MUST be chosen from the character set 0-9 (0x30-0x39) or A-F (0x41-0x46).
- The string consisting of eight 0-digits is not allowed and MUST NOT be used in a CVC.

The information contained in the following tables are the specific values for the required fields according to [RFC 3280]. These specific values for the Code Verification Certificate hierarchy MUST be followed according to Tables 11-7, 11-8, 11-9, 11-10 and 11-11 below. If a required field is not specifically listed in the tables, the guidelines in [RFC 3280] MUST be followed. The generic extensions MUST also be included as specified in PKI, see 11.3.4.2.

Code Verification Root CA Certificate

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

Table 11-7/J.192 – Code Verification Root CA Certificate

Subject Name Form	C=<country> O=<Company Name> CN= [Company Name] CVC Root CA
Intended Usage	This certificate is used to sign Code Verification CA Certificates
Signed By	Self-signed
Validity Period	20 to 30 years
Modulus Length	2048
Extensions	KeyUsage [c,m] (keyCertSign, cRL Sign), subjectKeyIdentifier [n,m], basicConstraints [c,m](cA=true)

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. A StandAlone PS MUST only support one CVC CA at a time.

Table 11-8/J.192 – Code Verification CA Certificate

Subject Name Form	C=<country> O=<Company Name><Eight(8) character hexadecimal value> CN= [Company Name] CVC CA
Intended Usage	This certificate is issued to the certificate authority by the Code Verification Root CA. This certificate issues Code Verification Certificates.
Signed By	Hierarchy's Code Verification Root CA
Validity Period	20 years
Modulus Length	2048
Extensions	KeyUsage [c,m] (keyCertSign, cRL Sign), subjectKeyIdentifier [n,m], authorityKeyIdentifier [n,m], basicConstraints [c,m](cA=true, pathLenConstraint=0)

Manufacturer Code Verification Certificate

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

Table 11-9/J.192 – Manufacturer Code Verification Certificate

Subject Name Form	C=<country> O=<CompanyName> [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	up to 10 years
Modulus Length	1024, 1536, 2048
Extensions	keyUsage[c,m](digitalSignature, keyEncipherment), extendedKeyUsage [c,m] (id-kp-codeSigning), authorityKeyIdentifier [n,m]

Code Verification Certificate

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

Table 11-10/J.192 – Code Verification Certificate

Subject Name Form	C=<country> O=<Eight(8) character hexadecimal value> CN= <Company Name>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 CA
Validity Period	up to 10 years
Modulus Length	1024, 1536, 2048
Extensions	keyUsage[c,m](digitalSignature, keyEncipherment), extendedKeyUsage [c,m] (id-kp-codeSigning), authorityKeyIdentifier [n,m]

Service Provider Code Verification Certificate

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

Table 11-11/J.192 – Service Provider Code Verification Certificate

Subject Name Form	C=<country> O=<Eight(8) character hexadecimal value> [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 CA
Validity Period	up to 10 years
Modulus Length	1024, 1536, 2048
Extensions	keyUsage[c,m](digitalSignature, keyEncipherment), extendedKeyUsage [c,m] (id-kp-codeSigning), authorityKeyIdentifier [n,m]

11.3.4.2.2.3 Service Provider certificate hierarchy

The Service Provider certificate hierarchy, or Service Provider chain, is rooted at a Service Provider Root CA, which is used to issue certificates for a set of authorized Service Providers. The Service Provider CA can be used to issue optional Local System CA Certificates or ancillary certificates. If the Service Provider CA does not issue the ancillary certificates, the Local System CA will. The ancillary certificates are the end entity certificates on the cable operator's network.

The information contained in the following tables are the specific values for the required fields according to [RFC 3280]. These specific values for the Service Provider Certificate hierarchy MUST be followed according to Tables 11-12 through 11-16 below. If a required field is not specifically listed in the tables, the guidelines in [RFC 3280] MUST be followed. The generic extensions for IPCable2Home MUST also be included as specified in PKI, see 11.3.4.2.

Service Provider Root CA Certificate

This certificate MUST be verified as part of the certificate chain containing the Service Provider Root CA Certificate, Service Provider CA Certificate, optional Local System CA Certificate, and Ancillary Certificates.

Table 11-12/J.192 – Service Provider Root CA Certificate

Subject Name Form	C=<country> O=<Company Name> CN=<Company Name> Service Provider Root CA
Intended Usage	This certificate is used to issue Service Provider CA Certificates.
Signed By	Self-signed
Validity Period	20 to 30 years
Modulus Length	2048
Extensions	KeyUsage [c,m] (keyCertSign, cRL Sign), subjectKeyIdentifier [n,m], basicConstraints [c,m](cA=true)

Service Provider CA Certificate

The Service Provider CA certificate MUST be verified as part of the certificate chain containing the Service Provider Root CA Certificate, Service Provider CA Certificate, optional Local System CA Certificate, and Ancillary Certificates.

Table 11-13/J.192 – Service Provider CA Certificate

Subject Name Form	C=<country> O=<CompanyName> CN=<CompanyName> Service Provider CA
Intended Usage	The Service Provider Root CA issues this certificate to each Service Provider. In order to make it easy to update this certificate, each network element is configured with the OrganizationName attribute of the Service Provider CA Certificate SubjectName. This is the only attribute in the certificate that must remain constant. This certificate appears as a read-write parameter in the MIB object that identifies the OrganizationName attribute for the IPCable2Home Kerberos realm. The IPCable2Home element does not accept Service Provider certificates that do not match this value of the OrganizationName attribute in the SubjectName. If the Headend contains a KDC that supports IPCable2Home, then the PS element needs to perform the first PKINIT exchange with the KDC right after a reboot, at which time its MIB tables are not yet configured. At that time, the IPCable2Home Kerberos client MUST accept any Service Provider OrganizationalName attribute, but it MUST later check that the value added into the MIB object for this realm is the same as the one in the initial PKINIT reply. This CA issues Local System CA certificates or ancillary certificates.
Signed By	Service Provider Root CA
Validity Period	20 years
Modulus Length	2048
Extensions	keyUsage[c,m](keyCertSign, cRLSign), subjectKeyIdentifier [n,m], authorityKeyIdentifier [n,m], basicConstraints[c,m](cA=true, pathLenConstraint=1)

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

Local System CA Certificate

This certificate is optional for the service provider. If this certificate exists, it MUST be verified as part of the certificate chain containing the Service Provider Root CA Certificate, Service Provider CA Certificate, optional Local System CA Certificate, and Ancillary Certificates.

Table 11-14/J.192 – Local System CA Certificate

Subject Name Form	C=<country> O=<CompanyName> OU=<Local System Name> CN=<CompanyName> Local System CA
Intended Usage	This certificate is optional and, if it exists, is issued by the Service Provider CA. This CA issues ancillary certificates. Network servers are allowed to move freely between regional CAs of the same service provider.
Signed By	Service Provider CA
Validity Period	20 years
Modulus Length	1024, 1536, 2048
Extensions	keyUsage[c,m](keyCertSign, cRLSign), subjectKeyIdentifier [n,m], authorityKeyIdentifier [n,m], basicConstraints[c,m](cA=true, pathLenConstraint=0)

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

KDC Certificate

This certificate MUST be verified as part of the certificate chain containing the Service Provider Root CA Certificate, Service Provider CA Certificate, optional Local System CA Certificate, and Ancillary Certificates (e.g., the KDC Certificates).

The KDC Certificate MUST include the Kerberos PKINIT subjectAltName as specified in 8.2.3.4.1/J.170, Key Distribution Centre Certificate.

Table 11-15/J.192 – KDC Certificate

Subject Name Form	C=<country> O=<Company Name> [OU=<Local System Name>] OU= <Company Name> Key Distribution Centre 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[c,o](digitalSignature) authorityKeyIdentifier[n,m](keyIdentifier=<subjectKeyIdentifier value from CA certificate>) subjectAltName[n,m] (Annex C/J.170)

HTTPS server Server Certificate

This certificate MUST be verified as part of the certificate chain containing the Service Provider Root CA Certificate, Service Provider CA Certificate, optional Local System CA Certificate, and Ancillary Certificates (e.g., the KDC Certificates).

Table 11-16/J.192 – HTTPS Server Certificate

Subject Name Form	C=<country> O=<Company Name> [OU=<Local System Name>] OU= <Company Name> HTTPS 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 HTTPS server to the HTTP clients for the TLS session during provisioning. This certificate is passed to the PS element inside the TLS Server Certificate message.
Signed By	Service Provider CA or the Local System CA
Validity Period	20 years
Modulus Length	1024, 1536, 2048
Extensions	keyUsage[c,m](digitalSignature, keyEncipherment, dataEncipherment), extendedKeyUsage[n,m] (id-kp-serverAuth), authorityKeyIdentifier [n,m]

11.3.4.2.3 Certificate validation

IPCable2Home certificate validation involves validation of a linked chain of certificates from the end entity certificates up to the valid Root. For example, the signature on the PS Element Certificate is verified with the Manufacturer CA Certificate, and then the signature on the Manufacturer CA Certificate is verified with the Manufacturer Root CA Certificate. The Manufacturer Root CA Certificate is self-signed, and is received from a trusted source in a secure way. The public key present in the Manufacturer Root CA Certificate is used to validate the signature on the same certificate.

The exact rules for certificate chain validation MUST fully comply with [RFC 3280], where they are referred to as "Certificate Path Validation". In general, X.509 certificates support a liberal set of rules for determining if the issuer name of a certificate matches the subject name of another. The rules are such that two name fields MAY be declared to match, even though a binary comparison of the two name fields does not indicate a match. [RFC 3280] recommends that certificate authorities restrict the encoding of name fields, so that an implementation can declare a match or mismatch, using simple binary comparison. IPCable2Home security follows this recommendation. Accordingly, the DER-encoded tbsCertificate.issuer field of an IPCable2Home certificate MUST be an exact match to the DER-encoded tbsCertificate.subject field of its issuer certificate. An implementation MAY compare an issuer name to a subject name by performing a binary comparison of the DER-encoded tbsCertificate.issuer and tbsCertificate.subject fields.

The validation of validity periods for nesting is not checked and intentionally not enforced, which is compliant with current standards. At the time of issuance, the validity start date for any end-entity certificate MUST be the same as or later than the start date of the issuing CA certificate validity period. After a CA certificate is renewed, the start dates of end-entity certificates MAY be earlier than the start date of the issuing CA certificate. The validity end date for end entity certificates MAY be before, the same as, or after the validity end date for the issuing CA, as specified in the IPCable2Home Certificate tables.

11.3.4.2.3.1 Validation for the manufacturer chain and root verification

The KDC will validate the linked chain of manufacturer certificates. Usually the first certificate in the chain is not explicitly included in the certificate chain that is sent over the wire. In the cases where the Manufacturer Root CA Certificate is explicitly included over the wire, it MUST already be known to the verifying party ahead of time to verify this certificate. The Manufacturer Root CA Certificate sent over the wire MUST NOT contain any changes to the certificate, with the possible exception of the certificate serial number, validity period, and the value of the signature. If changes other than the certificate serial number, validity period, and the value of the signature, exist in the Manufacturer Root CA certificate that was passed over the wire in comparison to the known Manufacturer Root CA Certificate, the KDC making the comparison MUST fail the certificate verification.

11.3.4.2.3.2 Validation for the code verification chain and root verification

A backoffice server can check the validity of the Code Verification Chain prior to beginning the software download process. For details, see the secure software download, clause 11.8.

11.3.4.2.3.3 Validation for the service provider chain and root verification

The PS Element MUST validate the linked chain of Service Provider certificates. Usually the first certificate in the chain is not explicitly included in the certificate chain that is sent over the wire. In the cases where the Service Provider Root CA Certificate is explicitly included over the wire, it MUST already be known to the verifying party ahead of time to verify this certificate. The Service Provider Root CA Certificate MUST NOT contain any changes to the certificate, with the possible exception of the certificate serial number, validity period, and the value of the signature. If changes other than the certificate serial number, validity period, and the value of the signature, exist in the

Service Provider Root CA Certificate that was passed over the wire in comparison to the known Service Provider Root CA Certificate, the PS element making the comparison MUST fail the certificate verification.

11.3.4.2.4 Certificate revocation

Certificate revocation is out of scope of this Recommendation.

11.4 Secure management messaging to the PS

The security algorithm used to initialize SNMP management messaging depends upon the provisioning mode of the PS element (see 5.5). In IPCable2Home, there are three provisioning modes: DHCP Provisioning Mode, SNMP Provisioning mode and Dormant mode. DHCP Provisioning Mode has additional sub-modes that identify whether it is configured for NmAccess Mode or Coexistence Mode. SNMP Provisioning Mode requires SNMPv3 for management messaging.

The following clauses describe the security algorithms and requirements needed to initialize SNMP management messaging, based on the provisioning mode of the PS element. The PS element MUST support the SNMPv3 security algorithms specified in 11.4.4.1.2 and 11.4.4.2.

11.4.1 Goals of secure management messaging

Securing management messages include the following goals:

- Provide options to encrypt network management messages to the PS.
- Provide options to authenticate network management messages to the PS.
- If possible, provide security on management messaging that will not require additional protocols to be implemented.
- Provide guidelines and minimum requirements for the encryption and authentication algorithms.

11.4.2 Secure management messaging system design guidelines

Reference	Guidelines
SEC3	Network management messages between the cable Headend and PS can be authenticated and optionally encrypted to protect against unauthorized monitoring and control.

11.4.3 Secure management messaging system description

The use of SNMP of management to the PS from the cable operators network. SNMP has been adopted into cable industry products for several years. The cable operator backoffice can support SNMPv1, v2 or v3. The PS is required to support management messaging for all three versions of SNMP. There is no security, *per se*, built into SNMPv1 or v2. SNMPv3 provides basic authentication and encryption algorithms defined in [RFC 3410] [RFC 3415] and [RFC 3584] and IPCable2Home specifies the use of the RFC defined security. SNMPv3 does not specify how the keys are set up to start the encryption and authentication process, and therefore, some details to generate and establish key exchange are specified. The details are listed within the next clause.

11.4.4 Secure management messaging requirements

11.4.4.1 Security algorithms for SNMP in DHCP Provisioning Mode

In DHCP Provisioning Mode, the PS element can be configured for NmAccess Mode or Coexistence Mode. In Coexistence Mode, the PS element can be configured for SNMPv1, SNMPv2, and/or SNMPv3 management messaging.

11.4.4.1.1 NmAccess Mode

If the PS Element is provisioned in DHCP Provisioning Mode with NmAccess Mode, the SNMP-based network management within the PS Element does not use SNMPv3 and therefore does not need to initialize SNMPv3 security functions. Initialization of the SNMPv1/v2 management link is defined in 6.3.3.1.

11.4.4.1.2 Coexistence Mode

If the PS Element is provisioned in DHCP Provisioning Mode with Coexistence Mode and the management messaging protocol is determined to be SNMPv3 (see 6.3.3.1), then the PS Element MUST use SNMPv3 security specified by [RFC 3414]. The PS MUST support SNMPv3 authentication and SNMPv3 privacy. The cable operator is strongly encouraged to turn on SNMPv3 authentication at all times. The use of SNMPv3 privacy is recommended if the cable operator can handle the additional load for encryption.

In order to establish SNMPv3 keys in DHCP provisioning mode, all IPCable2Home SNMP interfaces MUST utilize the SNMPv3 initialization and key changes procedure as defined in 2.2 of the DOCSIS 1.1 Operations Support Systems Interface specification, [ANSI/SCTE 23-3 2005] (replace "CM" wording with "PS element" and replace "DOCSIS 1.1 compliant" wording with "IPCable2Home compliant").

To support SNMPv3 initialization and key changes in DHCP provisioning mode, the PS element MUST also be capable of receiving TLVs of Type 34, 34.1, and 34.2, as defined in B.C.1.2.8 of the DOCSIS 1.1 Radio Frequency Interface specification, [J.112 Annex B] and implement the key-change mechanism specified in [RFC 2786] which includes the usmDHKickstartTable MIB object.

11.4.4.1.3 SNMPv3 Key Initialization

For each of up to 5 different security names, the Ultimate Authorization (CHAdministrator) generates a pair of numbers. First, the CHAdministrator generates a random number Rm.

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

$$z = g \text{ ^ } Rm \text{ MOD } p$$

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

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

TLV type 34.1 (SNMPv3 Kickstart Security Name) = CHAdministrator

TLV type 34.2 (SNMPv3 Kickstart Public Number) = z

The PS MUST support the VACM entries defined in 6.3.3.1.4.5. 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 = "CHAdministrator"

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

usmUserCloneFrom: ZeroDotZero

usmUserAuthProtocol: usmHMACMD5AuthProtocol [RFC 2104]

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

After the 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).

$$C = g \text{ ^ } xa \text{ MOD } p$$

where g is from the set of Diffie-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 = "CHAdministrator"

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, "CHAdministrator" 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) [RFC 2104]

authentication key ← PBKDF2(salt = 0x98dfb5ac,
iterationCount = 500,
keyLength = 16 (usmHMACMD5AuthProtocol) [RFC 2104],
prf = id-hmacWithSHA1) [RFC 2104]

At this point, the 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.

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 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.

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:

dhKickstartGroup

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 the PS's usmDHKickstartMypublic number associated with the securityName for which the manager wants to derive authentication and privacy keys. 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 securityName that the manager is going to use to communicate with the PS.

11.4.4.1.4 Diffie-Hellman key changes

The PS MUST support the key-change mechanism specified in the above clause as well as [RFC 2786].

11.4.4.2 Security algorithms for SNMPv3 in SNMP Provisioning Mode

If the PS Element is provisioned in SNMP Provisioning Mode, the SNMP-based network management within the PS Element MUST run over SNMPv3 with security specified by [RFC 3414]. The PS MUST support SNMPv3 authentication and SNMPv3 privacy. The cable operator is strongly encouraged to turn on SNMPv3 authentication at all times. The use of SNMPv3 privacy is recommended if the cable operator can handle the additional load for encryption. In order to establish SNMPv3 keys in SNMP provisioning mode, the PS MUST utilize Kerberized SNMPv3 key management as specified in 11.4.4.2.1.

11.4.4.2.1 Kerberized SNMPv3

The Kerberized key management profile specific for SNMPv3 MUST be followed as defined in 6.5.4/J.170.

11.4.4.2.2 SNMPv3 encryption algorithms

The encryption Transform Identifiers for Kerberized SNMPv3 key management MUST be followed as defined in 6.3.1/J.170.

11.4.4.2.3 SNMPv3 authentication algorithms

The authentication algorithms for Kerberized SNMPv3 key management MUST be followed as defined in 6.3.2/J.170.

11.4.4.2.4 SNMPv3 engine IDs

Because the SNMP Manager and Client MUST verify that the SNMPv3 Engine ID in the AP Request and AP Reply messages are based on the appropriate NMS principal name in the ticket [J.170], the following rules are used in generating SNMPv3 Engine IDs:

Rule 1: The SNMPv3 Engine ID MUST follow the format defined in [RFC 3411], i.e., the first bit is set to 1 (one) and the appropriate value is used for the first four bytes [RFC 3411].

Rule 2: The fifth byte MUST be the value 4 (four) to indicate that the following bytes, up to 27, are to be considered as text and are defined as follows:

- The characters of the NMS principal name MUST be used for the engine ID bytes starting on the 6th byte.
- The sequence of bytes, indicating the NMS principal name, MUST be followed by one byte and is considered as an 8-bit Hex value. Each unique value identifies a particular SNMP engine in the device (element or NMS server). The value 0 (zero) MUST not be used.
- The text string that starts on the 6th byte MUST be terminated with a Null character.

NOTE – Other formats are possible by following the approach in [RFC 3411]. The above selection, though, is intended to reduce implementation complexity that would be required if all of the approaches in [RFC 3411] were allowed.

11.4.4.2.5 Populating the usmUserTable

SNMPv3 security settings for the cable operator "CHAdministrator" user are defined in 6.3.3.1.4.5, View-based Access Control Model (VACM) Requirements. The CHAdministrator is the ultimate authority for management of the Portal Services element. Other users can also be defined. In this clause, a USM user is defined specifically for the provisioning process. In particular, it is defined to enable a notification receiver to be specified for the cabhPsDevProvEnrollTrap and cabhPsDevInitTrap, which the PS is required to send during the provisioning process (see Table 13-1, Flow Descriptions for PS WAN-Man Provisioning Process for DHCP Provisioning Mode, step CHPSWMD-11; Table 13-3, Flow Descriptions for PS WAN-Man Provisioning Process for SNMP Provisioning Mode, step CHPSWMS-11 and step CHPSWMS-13; and clause 13.4.3, Provisioning Enrollment/Provisioning Complete Informs).

The msgSecurityParameters in SNMPv3 messages carry a msgUserName field that specifies the user on whose behalf the message is being exchanged and with whose security information the fields msgAuthenticationParameters and msgPrivacyParameters are produced. For the SNMP engine of an IPCable2Home element to process these messages, the necessary information is required to be entered in the usmUserTable [RFC 3414] for the element engine.

The usmUserTable MUST be populated with the following information in the PS Element right after the AP Reply message is received:

- usmUserEngineID: the local SNMP engine ID as defined in 11.4.4.2.4, SNMPv3 Engine IDs
- usmUserName: CHAdministratorxx:xx:xx:xx:xx:xx, where xx:xx:xx:xx:xx:xx is the device's WAN-Man hardware address
- usmUserSecurityName: CHAdministratorxx:xx:xx:xx:xx:xx, where xx:xx:xx:xx:xx:xx is the device's WAN-Man hardware address
- usmUserCloneFrom: 0.0
- usmUserAuthProtocol: indicates the authentication protocol selected for the user, from the AP Reply message
- usmUserAuthKeyChange: default value ""
- usmUserOwnAuthKeyChange: default value ""
- usmUserPrivProtocol: indicates the encryption protocol selected for the user, from the AP Reply message
- usmUserPrivKeyChange: default value ""
- usmUserOwnPrivKeyChange: default value ""
- usmUserPublic: default value ""
- usmUserStorageType: permanent
- usmUserStatus: active

New SNMPv3 users MAY be created with standard SNMPv3 cloning, as defined in [RFC 3414].

The VACM Security To Group Table [RFC 3415] MUST be populated with the following information in the PS right after the AP Reply message is received:

- vacmSecurityModel: 3(usm)
- vacmSecurityName: CHAdministratorxx:xx:xx:xx:xx:xx
- vacmGroupName: CHAdministratorSNMP
- vacmSecurityToGroupStatus: active

The VACM Access Table [RFC 3415] MUST be populated with the following information, linked to the vacmSecurityToGroupTable defined above, in the PS right after the AP Reply message is received:

- vacmAccessContentPrefix: ""
- vacmAccessSecurityModel: 3(usm)
- vacmAccessSecurityLevel: AuthNoPriv
- vacmAccessContextMatch: exact(1)
- vacmAccessReadViewName: CHAdministratorView
- vacmAccessWriteViewName: CHAdministratorView
- vacmAccessNotifyViewName: CHAdministratorNotifyView
- vacmAccessStorageType: permanent
- vacmAccessStatus: active

Seven rows of the VACM View Tree [RFC 3415] MUST be populated with the following information in the PS right after the AP Reply message is received:

- vacmViewTreeFamilyName: CHAdministratorNotifyView
- vacmViewTreeFamilySubtree: cabhPsDevProvEnrollTrap
- vacmViewTreeFamilyType: included
- vacmViewTreeFamilyMask: ""
- vacmViewTreeFamilyName: CHAdministratorNotifyView
- vacmViewTreeFamilySubtree: cabhPsDevBase
- vacmViewTreeFamilyType: included
- vacmViewTreeFamilyMask: ""
- vacmViewTreeFamilyName: CHAdministratorNotifyView
- vacmViewTreeFamilySubtree: docsDevSoftware
- vacmViewTreeFamilyType: included
- vacmViewTreeFamilyMask: ""
- vacmViewTreeFamilyName: CHAdministratorNotifyView
- vacmViewTreeFamilySubtree: cabhPsDevInitTrap
- vacmViewTreeFamilyType: included
- vacmViewTreeFamily Mask: ""
- vacmViewTreeFamilyName: CHAdministratorNotifyView
- vacmViewTreeFamilySubtree: cabhPsDevBase
- vacmViewTreeFamilyType: included
- vacmViewTreeFamily Mask: ""
- vacmViewTreeFamilyName: CHAdministratorNotifyView
- vacmViewTreeFamilySubtree: docsDevEventTable
- vacmViewTreeFamilyType: included
- vacmViewTreeFamily Mask: ""
- vacmViewTreeFamilyName: CHAdministratorNotifyView
- vacmViewTreeFamilySubtree: cabhPsDevProv
- vacmViewTreeFamilyType: included
- vacmViewTreeFamily Mask: ""

11.5 CQoS in the PS

CQoS is a transparent bridge for IPCablecom and LAN-to-LAN QoS. The IPCablecom DQoS messages between the MTA and the CMTS, CMS, or CM are secured by the IPCablecom Security Specification. It is not within the scope of IPCable2Home to add security for IPCablecom messages. IPCable2Home in-home, LAN-to-LAN QoS messaging is not secured since the threat for attacks within the home are seen as extremely low. It is not within the scope of IPCable2Home to add security for IPCablecom messages. Since there is no security requirement for the PS element to secure CQoS messages originated on the WAN, there is no dependency on the backoffice servers to support this function.

11.6 Firewall in the PS

Security issues have been a major concern for companies relying on networks for decades, and now there is increasing awareness of security and privacy issues for home users with the "always on" CM. Because the average subscriber might lack some technical knowledge or even, if not, lacks the time to keep their home computers in top-notch secure operation, a firewall has become a necessary first line of defense in protecting the unsecured computers and other LAN IP devices in the home.

11.6.1 Goals and assumptions of IPCable2Home firewall

Goals

- Provide the cable operator with a standard and interoperable configuration for the firewall.
- Provide the cable operator with a minimum set of required functionality for the firewall.
- Enable monitoring of the firewall events using the event messaging mechanism.
- Protect the home network and LAN IP devices on that network from unwanted WAN-to-LAN traffic.
- Protect the HFC from unwanted LAN-to-WAN traffic.
- Protect the PS from attacks and traffic deemed as unwanted by the cable operator.
- Ensure the firewall will process packets at sufficient rates so packet filtering does not introduce a performance bottleneck, regardless of the complexity or size of the ruleset.
- Ensure support of identified applications through the firewall for specified scenarios.
- Provide the cable operator the ability to monitor and change rules used by the firewall.
- Ensure that the proper security configurations (e.g., filtering rules and policies) exist on the firewall system.
- Identify the types of attacks the firewall will log and specify the log so the operator can view the log as needed.
- Support IPCablecom through the firewall.
- Notify the administrator of defined important events in real time.
- Provide a factory default ruleset to ensure consistent baseline resets of the firewall.

Assumptions

- The firewall treats all packets destined to or coming from the LAN according to the current policy regardless of address mode, CAT or Passthrough (e.g., address mode has no affect on the firewall operations).
- The firewall begins operation immediately after the provisioning complete message, regardless of provisioning mode.
- SNMP, in particular SNMP messaging directed at the IPCable2Home Management Portal (CMP), can be used to configure IPCable2Home firewall rulesets. Thus, the ruleset is represented, externally as a collection of MIB objects.
- MIB objects control the logging actions taken by the firewall.
- The firewall will apply filtering rules and policies conjunction with checking the translated addresses known to the CAT in the PS.

11.6.2 Firewall system design guidelines

Firewall system design guidelines listed in Table 11-17 have guided IPCable2Home firewall specifications.

Table 11-17/J.192 – IPCable2Home security system design guidelines

Reference	Guidelines
SEC4	The firewall will accept configuration files in a standard language and format. (Note)
SEC5	The cable operator will have the ability to remotely manage compliant firewall products through configuration file or SNMP commands.
SEC6	The compliant firewall will include a default set of rules for an expected minimum set of functionality.
SEC7	Provide the necessary support for IPCablecom through the firewall.
SEC8	A minimum set of requirements will be placed on the firewall filtering capabilities for packet, port, IP addresses, ToD, etc.
SEC9	A detailed firewall event logging interface will allow the cable operator to monitor and review firewall activity as configured.
SEC10	The firewall will support commonly used applications in specific scenarios.
SEC11	The firewall will protect the LAN and WAN from common network attacks.
SEC12	The management of the events and rulesets for the firewall will be defined in detail via the Security MIB.
NOTE – The Firewall Configuration File Requirements are defined in 7.4, PS Function – Bulk Portal Services Configuration (BPSC).	

11.6.3 Firewall system description

Typically, firewalls are built using a combination of the following components: Packet Filter (PF), Stateful Packet Filtering (SPF), Application Layer Gateway (ALG), and Application Specific Proxy (ASP). A packet-filtering module is probably the most common firewall component because it determines which packet streams are blocked and which are allowed to cross the firewall. Each individual packet decision is based on static configuration information (the ruleset) configured into the firewall's filtering mechanisms (policy) so that the packet will be allowed or denied, based on the inspection of packet header fields: source and destination IP addresses, source and destination protocol port numbers, protocol type, etc. Depending on the desired level of security, a great number of filters might need to be configured on a firewall. The cable operator will need to balance the ruleset complexity with customer needs. This Recommendation attempts to specify a rich set of configuration filters, managed via MIB objects, so the various types of services (protocols and applications) can be individually configured, if needed.

A stateful packet filtering (SPF) module uses accumulated state information from packets that belong to the same connection when making packet-dropping decisions. The SPF differentiates between different protocols and handles each protocol's connection correctly. The SPF module stores and utilizes information found in the packet's network layer and transport layer headers.

An Application Layer Gateway (ALG) is a component that knows how to extract information required for connection tracking from the packet's application layer. As some protocols incorporate connection control information at the application layer, the SPF will incorporate ALGs to perform the connection tracking. The specific ALG (e.g., FTP-ALG, IPSec-ALG) is required for handling each such protocol needed to support IPCable2Home. For example, the FTP protocol in active mode incorporates the TCP port number that will be used later on for the data transfer. Therefore, it is required to use an FTP ALG to track the state of all FTP connections. See Annex D for more information on ALG requirements.

An Application Specific Proxy (ASP) firewall filters, based on the application layer protocol unique features, or messages specifically for the client-server protocols. There are security benefits in the use of ASPs. For one, it is possible to add access control lists to protocols, requiring users or systems to provide some level of authentication before access is granted. In addition to being protocol specific, an ASP understands the protocol and can be configured to block only subsections of the protocol. The ASP allows the operation of NAT-unfriendly applications when the Portal Service is operating in either of its two transparent routing modes: C-NAT or C-NAPT. For example, an FTP ASP can be configured to block the traffic from unauthenticated users, while granting authenticated users selective access to the "put" and "get" commands, depending on which directions these commands are issued.

The particular combination of packet filer, SPF AGLs and ASPs on a given firewall product, constitutes a trade off between performance and the security level. Typically, being a network layer mechanism, packet filters tend to yield better performance than ALGs/ASPs that are application layer mechanisms. A compromise solution becoming increasingly popular consists of the use of stateful packet filtering (SPF), where state information accumulated from packets that belong to the same connection is kept and used in making packet-dropping decision.

SPFs and ASPs both include filtering against the security policy to achieve the desired level of security for a site. However, while the security policy determines the allowed services and the way in which they are used across the firewall, the security policy does not spell out the specific configuration for the firewall. The ruleset is expressed in human readable form, then interpreted by the firewall, and implemented into the filtering policy in the internal language of the firewall. The filters inspect each packet and determines which packets the firewall forwards and which it rejects.

The following (Figure 11-3) is a high-level diagram of the firewall and the roles of the various firewall components referenced by this Recommendation.

NOTE – This diagram does not indicate any specific technical architecture or implementation. It is only for logical reference.

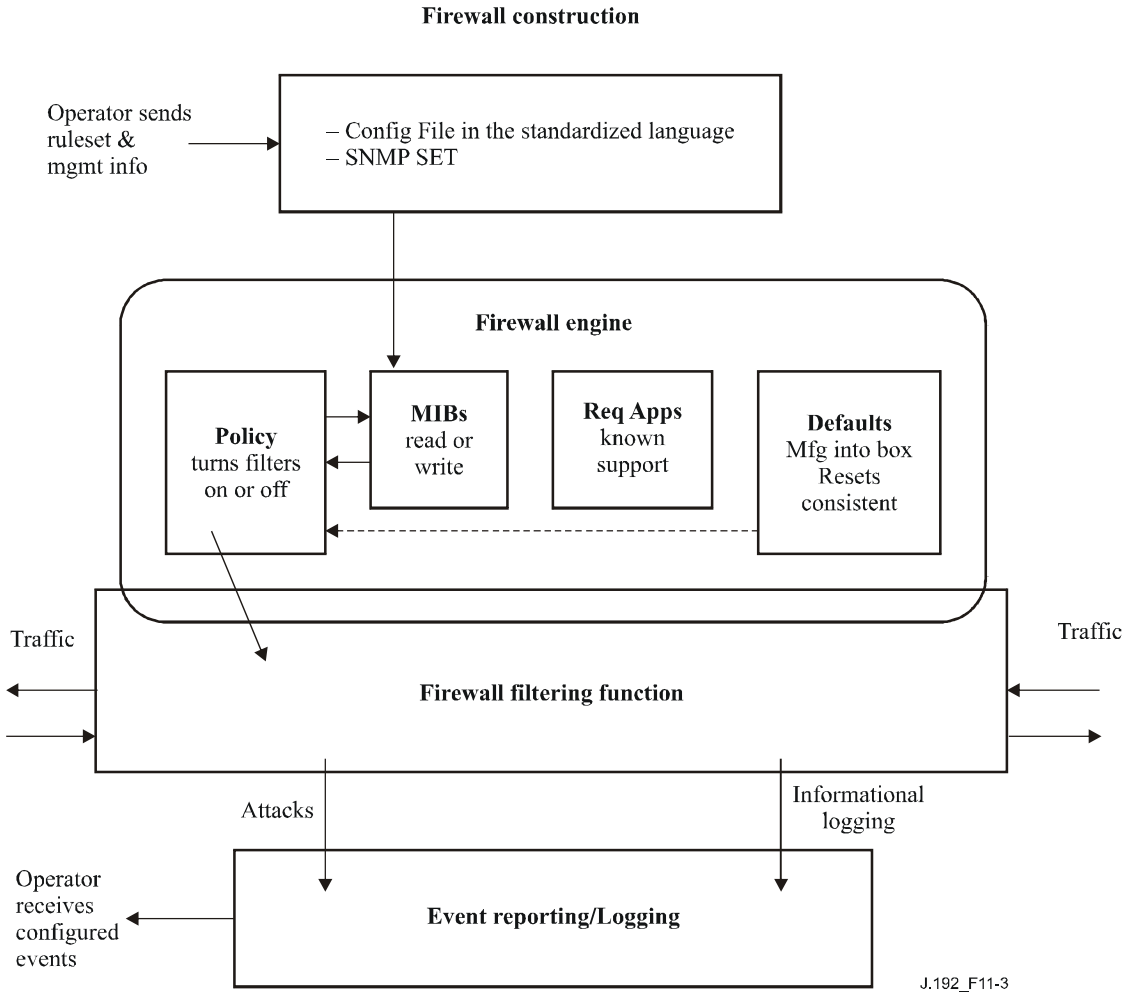


Figure 11-3/J.192 – Firewall logical reference

11.6.4 Firewall requirements

11.6.4.1 Configuration file language for the firewall

A cable operator chosen ruleset can be configured into the firewall via a PS configuration file or firewall configuration file download. Within this clause, the term configuration file is used to mean either the PS configuration file or firewall configuration file. The language and format for the configuration file containing the ruleset applicable to a particular firewall product is defined.

The PS MUST be able to receive and interpret a firewall configuration file constructed, using TLVs formatted as described in 7.4.4.1, Configuration File Format Requirements. Inside the firewall, the compiler translates the policy language into a vender specific internal format. TLV type 28 MUST be used for all the firewall MIB objects. The language of the PS configuration file and the firewall configuration file is the same. The requirements for firewall configuration file processing are defined in clause 7.

11.6.4.2 Firewall configuration

The PS supports, but the operator is not required to utilize, remote management of firewall functions. The firewall in the PS MUST accept rulesets configured in bulk, via the specified PS Firewall Configuration Files or configured individually via SNMP SET commands. The PS MUST NOT activate the firewall while the value of cabhPsDevProvState = inProgress(2). When a configuration file is used to configure rulesets, upon completion of configuration file download and

processing, i.e., cabhPsDevProvState = pass(1), the firewall rules from the configuration file MUST be immediately applied and available for use in the PS without reboot of the PS.

If the PS cannot process the configuration file for any reason, i.e., if cabhPsDevProvState = fail(3), the PS MUST use the existing firewall filter table rules indicated by the cabhSec2FwPolicySelection object.

11.6.4.3 Firewall policy and rulesets

The firewall policy instructs the firewall to filter traffic based on particular rules. The policy accepts the rulesets to be applied by the filtering function since the filtering function alone has no meaning, as it is only a set of capabilities. The firewall filtering capabilities, combined with the firewall policy, provide firewall protection for the LAN. The firewall filters actively inspect each packet or connection with the policy to apply the two allowed actions: allow or deny.

IPCable2Home defines three components as inputs into the firewall policy depending upon the configuration:

- General Behaviour Rules – The expected behaviour for either allowing or denying traffic flows. These rules always apply unless there is an exception written into either the IPCable2Home Factory Default Ruleset or Configured Ruleset.
- IPCable2Home Factory Default Ruleset – The firewall filtering factory default rules used as exceptions to the General Behaviour Rules. These rules may also be used in conjunction with the Configured Ruleset.
- Configured Ruleset – The configured rules used as exceptions to the General Behaviour Rules. These rules may also be used in conjunction with the IPCable2Home Factory Default Ruleset.

The General Behaviour Rules, IPCable2Home Factory Default Ruleset, and the Configured Ruleset apply to session initiation traffic and not to response traffic.

The PS may receive traffic for the IPCablecom MTA. Therefore, it is appropriate to take a brief look at the support needed for the MTA. Support for IPCablecom, described in 11.6.4.4, consists of the IPCable2Home Factory Default Ruleset plus the needed protocols to enable IPCablecom messaging to traverse the firewall. Annex D also notes which ports must be opened for the MTA. Support for IPCablecom enables provisioning, management, and services through the firewall.

11.6.4.3.1 Firewall policy and address realms

The concept of IP addressing realms are defined in this Recommendation for WAN and LAN IP addresses. Although the PS is considered part of the LAN, packets originating from or destined to the PS are not referred to as LAN traffic for the purpose of firewall filtering. Instead, the specific PS IP address is called out. Packets originating from or destined to the PS are indicated by the use of the WAN-Man IP address, PS server router IP address or to the fixed 192.168.0.1 IP address (which can be, but may not necessary be, the same as the PS server router IP address). Accordingly, the firewall will distinguish traffic to and from the PS in the Factory Default Ruleset and Configured Ruleset. Firewall behaviour is independent of Addressing Realms defined in 5.1.3. Firewall rules are not effected by Primary Packet Handling Mode or WAN Address Modes.

11.6.4.3.2 General firewall behaviour

The firewall in the PS is required to filter traffic based on the specified General Behaviour Rules. These rules are specified to provide a baseline level of filtering behaviour by the firewall in the PS. The General Behaviour applies unless an exception is defined in the Default or Configured Ruleset. The states defined for the General Behaviour Rules are either to allow or deny traffic. With the General Behaviour Rules in place, the cable operator can expect the PS to always behave in a standardized way with regard to filtering traffic. The PS MUST apply the General Behaviour Rules for firewall filtering as specified in Table 11-18 to a packet unless the firewall is configured to use

another rule written into the Factory Default Ruleset (cabhSec2FwFactoryDefaultFilterTable, see E.5) or the Configured Ruleset (docsDevFilterIpTable).

The firewall filters at the LAN side using the LAN side Packet Filters (LPF) and at the WAN side using the WAN side Packet Filter (WPF). The default behaviour is specified in terms of LPFs and WPFs. Please refer to 11.6.4.6.1 for a detailed description of the firewall architecture.

Table 11-18/J.192 – Firewall general behaviour rules

Source device	Destination IP address	General behaviour rule, WPF	General behaviour rule, LPF
Any WAN Device	PS WAN-Man IP Address	Deny All	N/A
	PS WAN-Data IP Address	Deny All	N/A
	Any LAN IP Address (passthrough mode)	Deny All	Allow All
PS Device: WAN-Man OR WAN-Data IP Address	Any WAN IP Address	Allow All	N/A
	Any LAN IP Address	N/A	Deny All
PS Device: Server Router IP Address OR 192.168.0.1	Any WAN IP Address	Deny All	N/A
	Any LAN IP Address	N/A	Allow All
Any LAN Device	PS Server Router IP Address OR 192.168.0.1	N/A	Allow All
	PS WAN-Man OR WAN-Data IP Address	N/A	Deny All
	Any WAN IP Address	Allow All	Allow All
N/A: Not Applicable. The traffic instance does not go through the interface.			

11.6.4.3.3 Factory default ruleset

The Factory Default Ruleset defines a set of filtering rules to be applied when the Default Ruleset option of the cabhSec2FwPolicySelection object is selected. The IPCable2Home Factory Default Ruleset MUST be hard-coded into the PS at the time of manufacture. The PS MUST use the IPCable2Home Factory Default Ruleset when the cabhSec2FwPolicySelection object is set to factoryDefault(1), or factoryDefaultAndConfiguredRuleset (3). Table 11-19 specifies the Factory Default Ruleset. Both LAN address realms, LAN-Trans and the LAN-Pass, are treated the same for the Factory Default Ruleset and are labeled LAN IP Address. The firewall MUST be able to look up addresses in the CAT mapping table to apply policy based on the real Host device IP Address. The table bases information on session initiation, not on allowed traffic. Therefore, the Firewall Factory Default Ruleset MUST be implemented for session initiation and not for traffic returning in response to an allowed session. Traffic returning at the request of the initiator is understood as state information for a session and the firewall will check the session state after checking the policies to ensure a packet is not denied that is part of a current session.

Table 11-19/J.192 – Firewall factory default policy

Source device	Destination IP address	General behaviour rule, WPF	General behaviour rule, LPF	Exception filtering protocol list (Rule number)
Any WAN IP Device	PS WAN-Man IP Address	Deny All	N/A	Allow ICMP (1) Allow SNMP (2,3)
	PS WAN-Data IP Address	Deny All	N/A	Allow ICMP (15)
	PS Server Router IP Address OR 192.168.0.1	Deny All	N/A	None
	Any LAN IP Address (passthrough mode)	Deny All	Allow All	Allow ICMP (4)
PS Device: WAN-Man OR WAN-Data IP Address	Any WAN IP Address	Allow All	N/A	None
	Any LAN IP Address	N/A	Deny All	Allow ICMP (5,16)
PS Device: Server Router IP Address OR 192.168.0.1	Any WAN IP Address	Deny All	N/A	None
	Any LAN IP Address	N/A	Allow All	None
Any LAN Device	PS Server Router IP Address OR 192.168.0.1	N/A	Allow All	None
	PS WAN-Man OR WAN-Data IP Address	N/A	Deny All	Allow ICMP (6,17)
	Any WAN IP Address	Allow All	Allow All	Deny Syslog (13,14)

The IPCable2Home Firewall Factory Default Ruleset listed in Table 11-20 MUST be implemented in the cabhSec2FwFactoryDefaultFilterTable MIB object. The column headers correspond to the defined MIB objects in the IPCable2Home Security MIB but since the object names are rather long, only the varying part of the object name is used in the table below. The rules that include the PS WAN-Data IP address are listed starting with Table Index 15, since the cable operator can optionally provision one or more WAN-Data IP addresses in the PS. This table will be correctly populated at the time the PS completes provisioning dependent upon how the cable operator has configured the IP addresses.

Table 11-20/J.192 – Firewall factory default ruleset

Table Index	Control	IfIndex	Direction	Saddr	Smask	Daddr	Dmask	Protocol	SourcePortLow	SourcePortHigh	DestPortLow	DestPortHigh	Continue
1	Allow	1	1	WAN IP (0.0.0.0)	(0.0.0.0)	PS WAN-Man	(255.255.255.255)	1	0	65535	0	65535	true
2	Allow	1	1	WAN IP (0.0.0.0)	(0.0.0.0)	PS WAN-Man	(255.255.255.255)	6	0	65535	161	161	true
3	Allow	1	1	WAN IP (0.0.0.0)	(0.0.0.0)	PS WAN-Man	(255.255.255.255)	17	0	65535	161	161	true
4	Allow	1	1	WAN IP (0.0.0.0)	(0.0.0.0)	LAN IP (0.0.0.0)	(0.0.0.0)	1	0	65535	0	65535	true
5	Allow	255	2	PS WAN-Man	(255.255.255.255)	LAN IP (0.0.0.0)	(0.0.0.0)	1	0	65535	0	65535	true
6	Allow	255	1	LAN IP (0.0.0.0)	(0.0.0.0)	PS WAN-Man	(255.255.255.255)	1	0	65535	0	65535	true
7	Deny	255	1	LAN IP (0.0.0.0)	(0.0.0.0)	WAN IP (0.0.0.0)	(0.0.0.0)	6	0	65535	88	88	true
8	Deny	255	1	LAN IP (0.0.0.0)	(0.0.0.0)	WAN IP (0.0.0.0)	(0.0.0.0)	17	0	65535	88	88	true
9	Deny	255	1	LAN IP (0.0.0.0)	(0.0.0.0)	WAN IP (0.0.0.0)	(0.0.0.0)	6	0	65535	749	749	true
10	Deny	255	1	LAN IP (0.0.0.0)	(0.0.0.0)	WAN IP (0.0.0.0)	(0.0.0.0)	17	0	65535	749	749	true
11	Deny	255	1	LAN IP (0.0.0.0)	(0.0.0.0)	WAN IP (0.0.0.0)	(0.0.0.0)	6	0	65535	1293	1293	true
12	Deny	255	1	LAN IP (0.0.0.0)	(0.0.0.0)	WAN IP (0.0.0.0)	(0.0.0.0)	17	0	65535	1293	1293	true
13	Deny	255	1	LAN IP (0.0.0.0)	(0.0.0.0)	WAN IP (0.0.0.0)	(0.0.0.0)	6	0	65535	514	514	true
14	Deny	255	1	LAN IP (0.0.0.0)	(0.0.0.0)	WAN IP (0.0.0.0)	(0.0.0.0)	17	0	65535	514	514	true
15	Allow	1	1	WAN IP (0.0.0.0)	(0.0.0.0)	PS WAN-Data	(255.255.255.255)	1	0	65535	0	65535	true
16	Allow	255	2	PS WAN-Data	(255.255.255.255)	LAN IP (0.0.0.0)	(0.0.0.0)	1	0	65535	0	65535	true
17	Allow	255	1	LAN IP (0.0.0.0)	(0.0.0.0)	PS WAN-Data	(255.255.255.255)	1	0	65535	0	65535	true

The cable operator can configure the PS with any firewall ruleset via configuration file or SNMP Set. When a cable operator sends rules to the PS, these rules are known as the Configured Ruleset. The PS MUST store the Configured Rules in the docsDevFilterIpTable [RFC 2669] and any scheduling information for particular rules in the MIB objects defined by IPCable2Home in the cabhSec2FwFilterScheduleTable (see E.5). The Configured Ruleset is only active for firewall filtering if the firewall is enabled, and the policy selection is set to configuredRuleset(2) or factoryDefaultAndConfiguredRuleset(3). The Configured Ruleset can be cleared from the docsDevFilterIpTable by setting the value of cabhSec2FwClearPreviousRuleset to true(1).

11.6.4.3.4 Configured ruleset

The IPCable2Home firewall filtering policy can be configured by creating filtering rules in the docsDevFilterIpTable and/or the cabhSec2FwLocalFilterIpTable (see 11.6.4.9.3). These filtering rules are considered the Configured Ruleset. The filtering rules defined in the Configured Ruleset are used as exceptions to the General Behaviour Rules. The Configured Ruleset may also be used in conjunction with the Factory Default Ruleset. When this is done, the filtering rules defined in the Configured Ruleset are used as exceptions to both the General Behaviour Rules and the Factory Default Ruleset.

Both the docsDevFilterIpTable and the cabhSec2FwLocalFilterIpTable have similar filter rule configuration capabilities. Having both filter tables eases Configured Ruleset management. For example, the docsDevFilterIpTable can be used to define generic filtering rules that apply across multiple devices and the cabhSec2FwLocalFilterIpTable can be used to define local or customer specific filtering rules that apply only to that device. The Operator may also allow the customer to configure their own filtering rules in the cabhSec2FwLocalFilterIpTable.

The cabhSec2FwPolicySelection MIB object allows the Operator to select which filtering ruleset(s) is active (see 11.6.4.9.1). If the docsDevFilterIpTable and the cabhSec2FwLocalFilterIpTable are both active, it is possible that a filtering rule may exist in each table that is in conflict. For example, one filter rule indicates an allow action and the other indicates a deny action for the same matched packet. For conflict resolution between these two tables, the cabhSec2FwPolicyConfiguredRulesetPriority MIB object is used to determine filter rule priority. For filter rule conflicts that may exist between the Configured Ruleset, the Factory Default Ruleset, and the General Behaviour rules, the PS MUST always give the Configured Ruleset priority.

When a firewall filter rule entry is created in the docsDevFilterIpTable or the cabhSec2FwLocalFilterIpTable that applies to a single LAN IP address that is dynamically assigned by the PS (CDS), the PS MUST create an IP address lease reservation for that address. This ensures that the IP address of the LAN device that the firewall filter rule entry applies to does not change upon lease renewal. A firewall filter rule entry that applies to a single source or destination IP address has the mask value set to 255.255.255.255.

The PS MUST determine if the single source and/or destination IP address of the firewall filter rule entry is dynamically assigned by the CDS, e.g., by looking it up in the cabhCdpLanAddrTable. If a corresponding entry exists in this table with the value of cabhCdpLanAddrMethod equal to either dynamicActive(4) or dynamicInactive(3), then the PS MUST replace that entry with one that represents a lease reservation for that IP address in the table, that is, one whose value of cabhCdpLanAddrMethod is set to either psReservationActive(6) or psReservationInactive(5), respectively. If a corresponding entry does not exist in the cabhCdpLanAddrTable, then the PS MUST NOT create a lease reservation for that IP address. In this case, it is possible that the IP address is statically assigned to the LAN IP Device.

When a firewall filter rule entry is removed from the docsDevFilterIpTable or the cabhSec2FwLocalFilterIpTable that applies to a single LAN IP address that is dynamically assigned by the PS (CDS), the PS MUST remove the corresponding IP address lease reservation that was internally created (identified by cabhCdpLanAddrMethod=psReservationActive(6)) from the cabhCdpLanAddrTable as long as there is not a corresponding DMZ entry in the cabhCapMappingTable that requires it.

11.6.4.4 IPCablecom support

If the cable operator deploys IPCablecom, the firewall may need to pass traffic to and from the MTA, depending upon network and device configuration. If operating a IPCablecom network, the protocols defined by the IPCablecom set of Recommendations MUST NOT be broken by the firewall. The cable operator may need to configure the firewall for any additional rules to ensure IPCablecom will be enabled through the firewall. Table 11-21 is a list of specifications which have unique port requirements for communication with the MTA. However, it is not a comprehensive list of all the IPCablecom specifications.

Table 11-21/J.192 – Relevant IPCablecom 1.x specifications for IPCable2Home firewall

Description	Specification
Audio/Video Codecs Specification	[ITU-T Rec. J.161]
Dynamic Quality of Service Specification	[ITU-T Rec. J.163]
Network-Based Call Signalling Protocol Specification	[ITU-T Rec. J.162]
MTA Device Provisioning Specification	[ITU-T Rec. J.167]
Security Specification	[ITU-T Rec. J.170]
Management Event Mechanism Specification	[ITU-T Rec. J.164]
Audio Server Protocol Specification	[ITU-T Rec. J.175]
Call Management Server Signalling Specification	[ITU-T Rec. J.178]

The list of the required IPCablecom protocols to the MTA have been taken from the indicated specifications. The IANA assigned port numbers to open the ports needed by the IPCablecom specified protocols through the firewall are listed in Annex D, Applications Through CAT and the Firewall. The IPCablecom defined protocols include the following:

- Provisioning SNMPv3, DHCP, DNS, TFTP, SYSLOG
- Media Stream RTP, RTCP
- QoS RSVP
- Security Kerberos, IPSec
- Network Call Signalling MGCP, SDP

NOTE – SDP does not require any specific port.

{informative text:

11.6.4.5 DMZ and UPnP WanIPConnection service support

DMZ and UPnP WANIPConnection (UWIC) CAP mappings allow unsolicited WAN-originated traffic to traverse the PS CAP (NAPT) function. To support this, the PS needs to create firewall filter rules that correspond with DMZ and UWIC CAP mapping table entries that allow this traffic to pass.

When a UPnP compliant LAN application configures a port mapping in the cabhCapMappingTable, the cabhCapMappingMethod MIB object will have a value of UPnP(3) (see 8.3.4.9). For each entry in the cabhCapMappingTable with a cabhCapMappingMethod MIB object value of UPnP(3), the PS MUST create a corresponding firewall filter rule that allows unsolicited WAN-to-LAN traffic to pass. When an entry in the cabhCapMappingTable with a cabhCapMappingMethod MIB object value of UPnP(3) is removed, the PS MUST remove the corresponding firewall filter rule.

A DMZ entry in the cabhCapMappingTable has the WAN and LAN port values set to zero (see 8.3.3.2). For each DMZ entry in the cabhCapMappingTable, the PS MUST create a corresponding firewall filter rule that allows unsolicited WAN-to-LAN traffic to pass. When a DMZ entry in the cabhCapMappingTable is removed, the PS MUST remove the corresponding firewall filter rule.

}

11.6.4.6 Firewall filtering

This clause specifies requirements for the firewall's packet filtering component. The specified packet filter examines individual packets and determines whether to allow or deny their passage across the firewall. More specifically, the packet filter inspects packet header fields and makes per-packet decisions based upon the contents of those fields and configured ruleset.

11.6.4.6.1 Minimum set of filtering capability

For the purpose of IPCable2Home, a simple NAT or packet filter is not sufficient. In order to provide a flexible and secure solution, the firewall MUST implement an Application Specific Proxy (ASP) or a Stateful Packet Filtering (SPF) firewall. Additionally, specific requirements for these filtering techniques are needed in order to provide a sufficient level of testable, reliable, and interoperable products for the cable industry. The firewall's ASP/SPF component controls traffic flows associated with application-layer protocols that cannot be effectively and transparently controlled through static filtering. The filtering mechanisms will examine applications that are dynamically established over IP, TCP, UDP, or ICMP sessions. Port, IP address, and scheduling activity is managed as related to a "session" within the firewall. Also, the application specific proxy allows the operation of NAT-unfriendly applications when the Portal Service is operating in either of its two transparent routing modes: C-NAT or C-NAPT.

Regardless of the type of firewall that is implemented, the PS firewall MUST be session aware and able to track information on an IP address pair (source and destination) in conjunction with the current policy for the specified IP address. A session consists of a pairing of IP addresses on a per request basis. This request includes matching the request with the allowed policy for that session which consists of IP address, application port and curfew.

The firewall's packet filter architecture specifies separate WAN side and LAN side packet filters at the PS. The WAN side packet filter examines packets at the WAN interface that originate from the WAN domain, the LAN domain, or the PS internal components. The LAN side packet filter examines packets at the Aggregated LAN Interfaces that originate from the LAN domain, the WAN domain, or the PS internal components. Separate rules can be applied to WAN and LAN side packet filters.

The components of the PS are located in relationship to the Firewall as shown in Figure 11-4. Packets received by the PS from either the WAN or LAN domains are filtered at the firewall before they reach any of the PS non-firewall components CDP, CNP, CSP, CQP, CMP, and the CAP with the exception of the USFS. In the same way, packets to be transmitted by the PS to either the WAN or LAN domains will pass through the non-firewall components before reaching the WPF or LPF.

WPF and LPF also take action over packets originated at the PS internal components. These packets are filtered by the WPF and LPF before they are forwarded to either the WAN or LAN domains, respectively.

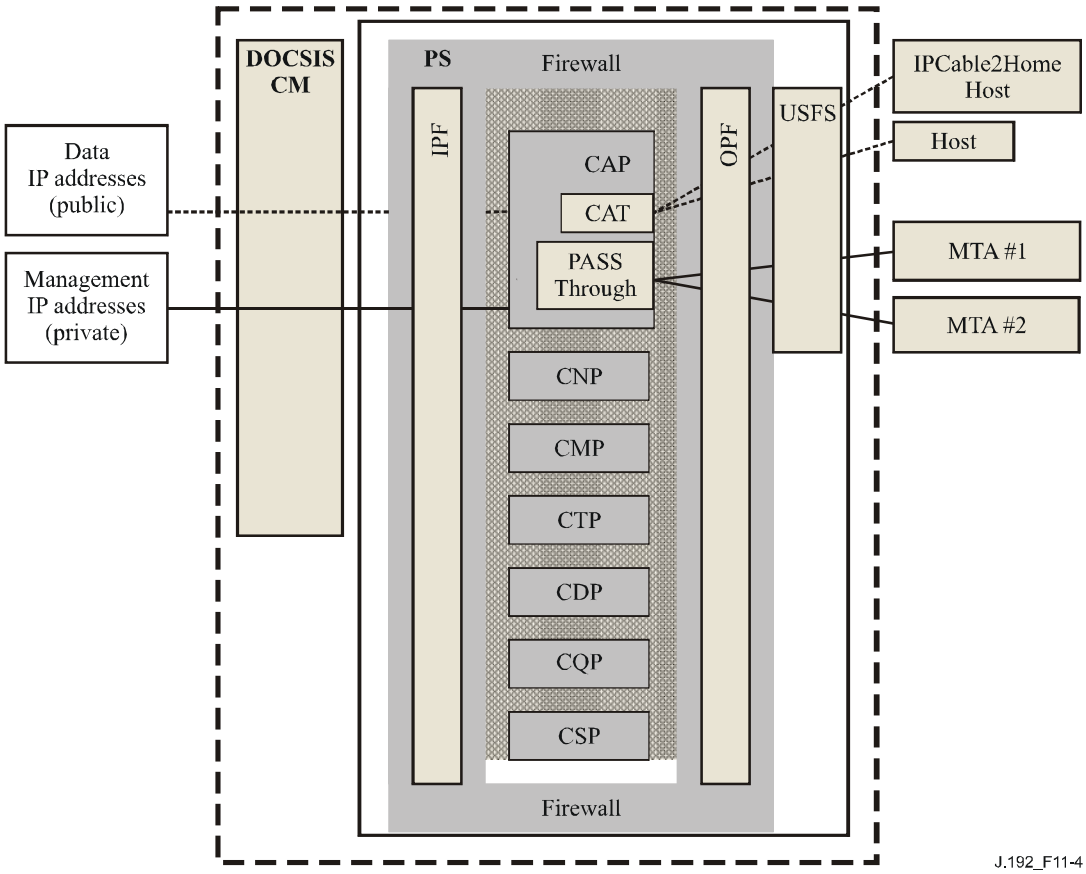


Figure 11-4/J.192 – Firewall functionality inside the PS

The following filtering definitions are used:

- ALLOW means "let the packet through".
- DENY means to "drop the packet".

The firewall's WPF and LPF MUST exhibit the following behaviour:

- The firewall MUST filter traffic based on the IPCable2Home defined policy as listed in 11.6.4.3, Firewall Policy and ruleset, in cases where there is not an explicit rule to follow when checking a packet.
- The firewall MUST deny replayed packets from either the LAN or the WAN.
- The firewall MUST create a "state" for all allowed packets initiating a session. A packet will either be accepted because there is a static rule to allow packets of that criteria, or there is a state that implies that the packet will be allowed through as a result of an ongoing allowed session.
- The firewall SHOULD NOT allow TCP outbound traffic prior to establishing a TCP session (i.e., prior to completing a 3-way TCP handshake).
- Packets with one of the following IP Options: LSRR (Loose-Source-Route), SSRR (Strict-Source-Route), RR (Record-Route) MUST be denied.

There are many types of network attacks that the firewall can filter. Many methods and tools are used to attack various devices on a network. The list is very long and changes faster than any current published document can claim. This Recommendation calls out some of the well-known attacks for general security consideration. The firewall SHOULD protect against port or network scanning launched from LAN or WAN. The firewall SHOULD protect against floods of packets and malformed packets. The firewall SHOULD protect against the following list of denial of

service attacks: "Ping of Death", "Teardrop", "Bonk", "Nestea", "SYN Flood", "LAND Attack", "IP Spoofing", "Smurf Attack", "WinNuke", and any high-frequency messaging originated by LAN IP Devices, such as BP_Init or DHCP DISCOVER messages.

11.6.4.6.2 Filter criteria

The default is to deny traffic initiated from WAN IP addresses, the PS WAN-Man IP address, or the PS Server Router IP address. Therefore, the rulesets are built to allow particular traffic for these addresses. The default is to allow traffic from LAN IP addresses unless explicitly set to deny; therefore, the rulesets are built to deny particular traffic to these addresses. This clause does not specify all the expected filtering capabilities, but lists a minimum set of criteria which is expanded by specified MIB objects. Inbound and outbound packet filters MUST examine traffic to see if a rule will allow the traffic based on the following filtering criteria:

- IP source address;
- IP destination address;
- IP ("next level") protocol; e.g., TCP, UDP, ICMP, IPSec AH, IPSec ESP;
- TCP or UDP source and destination ports;
- Start-of-connection information for TCP packets (i.e., absence of ACK bit) for session tracking;
- Sequence number tracking for sessions.

The above packet data is used as criteria for matching incoming packets to a specific rule, and hence, arriving at a specific filtering decision (allow/deny). The firewall MUST check the IP source and destination address to see if any rule applies to that address. If the ruleset currently prohibits forwarding traffic to or from an IP address, the firewall MUST deny the packet, unless it needs to be passed due to state.

NOTE – Filtering against the current policy includes more requirements for filtering that must be applied; however, those requirements are not considered a part of the built-in filtering criteria.

11.6.4.6.3 Filtering architecture

The firewall packet filter will be able to filter traffic providing distinct filtering for WAN, LAN or PS originated traffic. This firewall MUST:

- Filter packets at the WAN interface that originate from either side of the interface. Filtering rules at the WAN side of the PS are identified by the IfIndex value of 1 (one) and apply to all traffic to and from the WAN.
- Filter packets at the PS Aggregated LAN Interfaces interface, ifIndex value 255, that originate at either side of the interface.
- Filter packets originating from within the PS going either to the LAN or WAN.
- Apply filters only as currently enabled.
- Apply packet filtering before performing any ASP/SPF processing.
- Apply packet filtering to the packets the PS receives before passing such packets to any of the PS non-firewall components. However, since the firewall is not required to be able to filter LAN to LAN traffic, LAN side originated traffic received by the PS encounters the USFS before encountering the LPF.

The WPF MUST exhibit the following general behaviour:

- Filter as defined in 11.6.4.3.
- Deny all packets that enter the PS from the WAN and that have source addresses that belong to LAN-Pass or LAN-Trans address realms.
- Deny all packets with broadcast or multicast source addresses.

The LPF MUST exhibit the following general behaviour:

- Filter as defined in 11.6.4.3.
- Reject all packets with broadcast or multicast source addresses.

11.6.4.7 Firewall event reporting

The information coming out of the firewall is critical for routine management and monitoring, as well as providing the appropriate events for specified attacks. The events generated by the firewall can be used for intrusion detection, DoS attacks, and any failures or logs related to the firewall system. The analysis of the logs can be quite cumbersome if there are large amounts of data to sort through. Also, if there are too many events sent to the cable operator, it could tie up bandwidth, since there can be many firewalls sending events to the NMS located in the cable operator backoffice. The cable operator will need to decide which items they wish to turn on to monitor the firewall and how often they would like to receive events. Turning on event reporting is separate from turning on the ruleset for the firewall filtering criteria. When the firewall event enable MIB objects have been set to enable the firewall to track defined event types, the firewall will log and send specified event messages as defined in this clause and Annex B.

Each of the specified events can be turned on or off by the cable operator through setting a SNMP MIB object through a configuration file, or a SNMP set. It is recommended that SNMPv3 be used to secure SNMP messages containing firewall information.

11.6.4.7.1 Firewall events

Firewall events allow a cable operator to remotely assess the level of hacker activity and modifications to the firewall on specific PS elements. Event generation is based on management changes to the ruleset, events detected by the firewall as enabled by the ruleset, or TFTP/HTTP events based on downloading. The TFTP/HTTP events for firewall download MUST be sent, as defined by Annex B.

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

TYPE 1: Type 1 MUST log all attempts from both LAN and WAN clients to traverse the firewall that violate the Security Policy when this type is turned on via the cabhSec2FwEventEnable MIB object. This logs all connection attempts that are dropped due to policy violation. An attack is defined as packets (meaning each packet is counted as an attack), that attempt to traverse the firewall and violate the current policy. If enabled, and the threshold is reached, the PS MUST immediately send event 80010201.

TYPE 2: Type 2 MUST log identified Denial of Service attack attempts when this type is turned on, via the cabhSec2FwEventEnable MIB object. A type 2 attack is defined as any attempt that is considered to be disrupting any service, like the flood of duplicate packets (meaning 10 packets are counted as one attempt), or malformed packets or unpermitted connection attempts from the same host, for a multiple number of times. If enabled, and the threshold is reached, the PS MUST immediately send event 80010202.

TYPE 3: Type 3 MUST log all changes made to the cabhSec2FwPolicyFileURL or cabhSec2FwPolicyFileCurrentVersion or cabhSec2FwEnable MIB objects when this type is turned on, via the cabhSec2FwEventEnable MIB object. Tracking the changes to the firewall configuration provides valuable feedback to the cable operator for debugging purposes. If enabled and the threshold is reached, the PS MUST immediately send event 80010203.

TYPE 4: Type 4 MUST log all failed attempts to modify cabhSec2FwPolicyFileURL and cabhSec2FwEnable MIB objects when this type is turned on, via the cabhSec2FwEventEnable MIB. If enabled and the threshold is reached, the PS MUST immediately send event 80010204.

TYPE 5: Type 5 MUST log allowed inbound packets from the WAN when this type is turned on, via the cabhSec2FwEventEnable MIB object. This enables the cable operator to monitor traffic in a

scenario where there are signs of detection intrusion or DoS attacks from the WAN side. If enabled and the threshold is reached, the PS MUST immediately send event 80010205.

TYPE 6: Type 6 MUST log allowed outbound packets from the LAN when this type is turned on, via the cabhSec2FwEventEnable MIB object. This enables the cable operator to monitor traffic in a scenario where there are signs of attacks coming from a home LAN across the WAN. If enabled and the threshold is reached, the PS MUST immediately send event 80010206.

The event types for IPCable2Home are defined for monitoring purposes only. It is up to the individual cable operator to evaluate and execute any necessary response to issues detected and reported by the firewall.

11.6.4.7.2 Firewall logs

The firewall log information MUST be recorded in the PS for each enabled log type, as specified in 11.6.4.7.1. For each enabled event type, the PS MUST log the specified information in the cabhSec2FwLogTable whenever the event count reaches the specified threshold within the recording interval. The event count, threshold, and interval are defined for each event type in the cabhSec2FwEventControlTable (cabhSec2FwEventCount, cabhSec2FwEventThreshold and cabhSec2FwEventInterval in 11.6.4.9.2). Any event logged in the cabhSec2FwLogTable MUST also be logged in the docsDevEventTable, provided the additional throttling constraints for the docsDevEventTable specified in 6.3.3.2.4.8 are met.

If the log table is full, the PS MUST remove the oldest entry and add the new one. If the cabhSec2FwEventThreshold is not set to zero, the cabhSec2FwEventEnable is enabled, and the cabhSec2FwEventInterval is not set to zero, the PS MUST continue to log events of the enabled type. Once the cabhSec2FwEventLogReset is set to 1 to clear the log, and the cabhSec2FwEventEnable is enabled, the cabhSec2FwEventCount MUST start counting from zero.

The PS, at a minimum, MUST support the logging of 40 entries in the Firewall Log Table (cabhSec2FwLogTable). If an event type is enabled, the PS MUST log information required by the event type at a minimum rate of 1 event per every 5 seconds, even while under attack. It is expected that the PS will not consume the majority of its computing resources on logging and when attacks occur, the PS SHOULD be able to pass traffic at a normal rate and otherwise function normally.

Logging can pose different problems if not properly done. Logging all events and packets can make the log complex, lengthy, and difficult to understand. It is difficult to sort through a lot of information to look for one item in particular. If logging is limited to only a few types of events, it will not provide enough information to the cable operator to debug intrusions or detect attacks. Note that logs can be sniffed if they are not encrypted. A hacker can use log information to gain insight into the various services running on the PS or LAN Host devices.

IPCable2Home requires a particular set of information to be logged for each type of event that is enabled. The log function MUST log packets of each type according to the rules for that type of event. The requirement for Date and Time assumes that the Date and Time will be as accurate as the last update of the PS clock during the provisioning sequence.

The cabhSec2FwLogTable for the event types 1, 2, 5, and 6 MUST record the following information for each occurrence unless otherwise specified:

- Event Number – MUST be recorded as defined in Annex B, one time, at the start of the log.
- Event Priority – MUST be recorded as defined in Annex B, one time, at the start of the log.
- Date and Time – When the event occurred:
  - MUST consist of the four-digit year, month, and day;
  - MUST consist of the hour, minute, and second.

- Protocol – The protocol indicated in the IP header field (1 = ICMP; 2 = IGMP; 6 = TCP; 17 = UDP).
- Source IP Address.
- Destination IP Address.
- Source Port (TCP and UDP).
- Destination Port (TCP and UDP).
- Message Type (ICMP) – [RFC 2474] defines ICMP and when the firewall blocks an ICMP packet the log MUST display a number indicating what type of ICMP message it was. 0 – Echo Reply, 3 – Destination Unreachable, 4 – Source Quench, 5 – Redirect, 8 – Echo Request, 9 – Router Advertisement, 10 – Router Solicitation, 11 – Time Exceeded, 12 – Parameter Problem, 13 – Timestamp Request, 14 – Timestamp Reply, 15 – Information Request, 16 – Information Reply, 17 – Address Mask Request, 18 – Address Mask Reply.
- Replay Count – If the data being recorded is a replay attack, the firewall SHOULD NOT record each occurrence of the attack. However, the firewall SHOULD record the number of occurrences up to the threshold value set for the specific type.
- Matching Filter Table Name (when applicable) – When the event is generated due to a packet match of a filter rule table entry, the filter table name (docsDevFilterIpTable, cabhSec2FwFactoryDefaultFilterTable, or cabhSec2FwLocalFilterIpTable) MUST be provided.
- Matching Filter Table Index (when applicable) – When the event is generated due to a packet match of a filter rule table entry, the filter table index MUST be provided.
- Matching Filter Description (when applicable) – When the event is generated due to a packet match of a filter rule table entry, the filter description MUST be provided.

The cabhSec2FwLogTable for the event type 3 MUST record the following information for each occurrence unless otherwise specified:

- Event Number – MUST be recorded as defined in Annex B, one time, at the start of the log.
- Event Priority – MUST be recorded as defined in Annex B, one time, at the start of the log.
- Date and Time – When the event occurred:
  - MUST consist of the four-digit year, month, and day;
  - MUST consist of the hour, minute, and second.
- Source IP Address.
- MIB object changed.

The cabhSec2FwLogTable for the event type 4 MUST record the following information for each occurrence unless otherwise specified:

- Event Number – MUST be recorded as defined in Annex B, one time, at the start of the log.
- Event Priority – MUST be recorded as defined in Annex B, one time, at the start of the log.
- Date and Time – When the event occurred:
  - MUST consist of the four-digit year, month, and day;
  - MUST consist of the hour, minute, and second.
- Source IP Address.
- MIB object attempted to be changed.

11.6.4.8 Applications through the firewall

As part of the minimum set of capabilities, the firewall MUST be capable of allowing specified applications, as defined by Annex D, to traverse the PS to reach its intended destination. The firewall applies the current ruleset to the policy to ensure the correct openings are created to support specific traffic between the LAN and WAN, as well as to and from the PS itself. The PS MUST NOT limit the number of sessions or connections to be supported simultaneously, unless otherwise specified in Annex D, Applications through the CAT and Firewall.

The firewall policy is applied to the traffic as it attempts to traverse the firewall. The packets are first processed in the firewall prior to being sent to the PS for further processing, or to the destination on the WAN or LAN. The policy is applied to source and destination IP addresses, ports, and time of day. Annex D lists the requirements and provides more detail.

11.6.4.9 Firewall MIB objects

The firewall MIB objects consist of a three general groupings:

1) a set to manage the firewall configuration;

2) a set to monitor and log events; and

3) a set to manage the rulesets themselves.

The requirements for the firewall MIB objects MUST be used in conjunction with the Security MIB document [see E.5].

11.6.4.9.1 Firewall ruleset management MIB objects

The following firewall management objects MUST be implemented in the PS:

cabhSec2FwPolicyFileURL – Contains the name of the policy ruleset file and the IP address of the TFTP or HTTPS server containing the policy ruleset file, in a TFTP or HTTPS URL format. A policy ruleset file download is triggered when the value used to SET this MIB is different than the value in the cabhSec2FwPolicySuccessfulFileURL MIB. Refer to 7.4.4.2.3, Firewall Configuration File Trigger.

If the download of the Firewall Configuration File is not successful, the PS MUST NOT update the cabhSec2FwPolicySuccessfulFileURL MIB with the same value as the cabhSec2FwPolicyFileURL MIB. In any case, the cabhSec2FwPolicyFileURL MIB object MUST contain the value SET by either the PS Configuration File or by a SNMP SET command. When the PS is reset, the cabhSec2FwPolicyFileURL MIB object MUST be populated with its default value.

CabhSec2FwPolicySuccessfulFileURL – Contains the name of the policy ruleset file and the IP address of the TFTP server that contained the policy ruleset file, in a TFTP or HTTPS URL format, which was used to trigger the last successful download. If a successful download has not yet occurred, this MIB should have a Null value.

cabhSec2FwPolicyFileHash – Defines the SHA-1 digest for the corresponding ruleset file.

cabhSec2FwPolicyFileOperStatus – Contains the operational status of the firewall configuration file download and it MUST contain the following three states:

- inProgress(1) – Indicates that a firewall configuration file download is under way.
- complete(2) – Indicates that the firewall configuration file has downloaded successfully.
- failed(3) – Indicates that the last attempted download of the firewall configuration file failed.

cabhSec2FwPolicyFileCurrentVersion – A label set by the cable operator that can be used to track various versions of configured rulesets. When this label is set via SNMP or configuration file, its value is changed and the configured firewall filter rules are changed. However, since firewall filter rules can be modified using SNMP after the initial configuration by the Policy File (firewall

configuration file), the value of this label (i.e., the version of the current policy file) may not accurately reflect the firewall configuration currently in effect. This object MUST contain the string "null", if it has never been configured.

cabhSec2FwEnable – Allows for activation and deactivation of firewall. If this object is set to disabled, the firewall MUST be completely turned off. If this object is set to enable, the firewall MUST be activated immediately without rebooting the PS.

cabhSec2FwClearPreviousRuleset – Allows the Operator to clear the filter rule entries in the docsDevFilterIpTable.

cabhSec2FwPolicySelection – Allows for selection of the filtering policy as defined by the following options:

- factoryDefault (1) – Indicates the firewall is using the factory default settings defined in 11.6.4.3.3. If the cabhSec2FwPolicySelection MIB object is set to factoryDefault (1), then the firewall filters against the Factory Default Ruleset in the cabhSec2FwFactoryDefaultFilterTable.
- configuredRulesetBoth (2) – Indicates the firewall is using the Configured Ruleset defined by both the docsDevFilterIpTable and the cabhSec2FwLocalFilterIpTable. If the cabhSec2FwPolicySelection MIB object is set to configuredRulesetBoth (2), then the firewall filters against the filter rules defined in both the docsDevFilterIpTable and the cabhSec2FwLocalFilterIpTable.
- factoryDefaultAndConfiguredRulesetBoth (3) – Indicates the firewall is using the Factory Default Ruleset and the Configured Ruleset defined by both the docsDevFilterIpTable and the cabhSec2FwLocalFilterIpTable. If the cabhSec2FwPolicySelection MIB object is set to factoryDefaultAndConfiguredRulesetBoth (3) the PS MUST filter against the IPCable2Home specified Factory Default Ruleset in the cabhSec2FwFactoryDefaultFilterTable and the filter rules defined in both the docsDevFilterIpTable and the cabhSec2FwLocalFilterIpTable.
- configuredRulesetDocsDevFilterIpTable (4) – Indicates the firewall is using the Configured Ruleset defined by the docsDevFilterIpTable. If the cabhSec2FwPolicySelection MIB object is set to configuredRulesetDocsDevFilterIpTable (4), then the firewall filters against the Configured Ruleset in the docsDevFilterIpTable.
- configuredRulesetCabhSec2FwLocalFilterIpTable (5) – Indicates the firewall is using the Configured Ruleset defined by the cabhSec2FwLocalFilterIpTable. If the cabhSec2FwPolicySelection MIB object is set to configuredRulesetCabhSec2FwLocalFilterIpTable (5), then the firewall filters against the Configured Ruleset in the cabhSec2FwLocalFilterIpTable.
- factoryDefaultAndConfiguredRulesetDocsDevFilterIpTable (6) – Indicates the firewall is using the Factory Default Ruleset and the Configured Ruleset defined by the DocsDevFilterIpTable. If the cabhSec2FwPolicySelection MIB object is set to factoryDefaultAndConfiguredRulesetDocsDevFilterIpTable (6), the PS MUST filter against the IPCable2Home specified Factory Default Ruleset in the cabhSec2FwFactoryDefaultFilterTable and the Configured Ruleset in the docsDevFilterIpTable.
- factoryDefaultAndConfiguredRulesetCabhSec2FwLocalFilterIpTable (7) – Indicates the firewall is using the Factory Default Ruleset and the Configured Ruleset defined by the cabhSec2FwLocalFilterIpTable. If the cabhSec2FwPolicySelection MIB object is set to factoryDefaultAndConfiguredRulesetCabhSec2FwLocalFilterIpTable (7), the PS MUST filter against the IPCable2Home specified Factory Default Ruleset in the cabhSec2FwFactoryDefaultFilterTable and the Configured Ruleset in the cabhSec2FwLocalFilterIpTable.

cabhSec2FwEventSetToFactory – Allows the operator to clear all the events currently set in the event table. The PS MUST immediately clear the cabhSec2FwEventControlTable if this object is set to true.

cabhSec2FwEventLastSetToFactory – This object reports the last time the event table was cleared.

cabhSec2FwConfiguredRulesetPriority – Defines which Configured Ruleset filter rule has priority when a conflict exists between a filter rule in the docsDevFilterIpTable and a filter rule in the cabhSec2FwLocalFilterIpTable as indicated by the following options:

- docsDevFilterIpTable (1) – Indicates that filter rules in the docsDevFilterIpTable have priority over any conflicting filters that may exist in the cabhSec2FwLocalFilterIpTable.
- cabhSec2FwLocalFilterIpTable (2) – Indicates that filter rules in the cabhSec2FwLocalFilterIpTable have priority over any conflicting filters that may exist in the docsDevFilterIpTable.

cabhSec2FwClearLocalRuleset – Allows the Operator to clear the filter rule entries in the cabhSec2FwLocalFilterIpTable.

11.6.4.9.2 MIB objects for firewall events

The following firewall event objects MUST be implemented in the PS, as defined in the Security MIB and are included in the cabhSec2FwEventControlTable:

cabhSec2FwEventType – Assigns the event type for the table to track. Event types are defined in 11.6.4.7.1.

cabhSec2FwEventEnable – Enables or disables counting and logging of firewall events by type as assigned in cabhSec2FwEventType. Logging requirements are defined in the log data clause (see 11.6.4.7.2). This is an on/off switch only. If the enable value changes, the PS MUST immediately send the appropriate event (8001010x). If this value is enabled, the firewall MUST log occurrences in the cabhSec2FwLog. The firewall MUST NOT count, send events, or collect log data for attacks when cabhSec2FwEventEnable is disabled. Default = false.

cabhSec2FwEventThreshold – Number of attacks to count before sending the appropriate event by type as assigned in cabhSec2FwEventType. If the value is set to zero, the firewall MUST NOT count, send events, or collect log data for this type. Default = 0.

cabhSec2FwEventInterval – Indicates the time interval in hours to count and log occurrences of a firewall event type as assigned in cabhSec2FwEventType. This time interval applies as long as the cabhSec2FwEventThreshold object is not reached. If the cabhSec2FwEventInterval MIB object has a value of zero, there is no interval assigned and the PS MUST NOT count, send, or log events. Default = 0.

cabhSec2FwEventCount – Indicates the current count of attacks, up to the cabhSec2FwEventThreshold value by type as assigned by cabhSec2FwEventType. The firewall MUST start counting attacks from zero each time the cabhSec2FwEventEnable MIB object is enabled, or the cabhSec2FwEventInterval is over, or the cabhSec2FwEventCount equals the cabhSec2FwEventThreshold value. If the number of attacks counted in the cabhSec2FwEventCount equals the threshold set in the cabhSec2FwEventThreshold, prior to the end of the time interval defined by the cabhSec2FwEventInterval object, the PS MUST immediately send the appropriate event (8001020x). Default = 0.

cabhSec2FwEventLogReset – Setting this object to true clears the log table for the specified event type. Reading this object always returns false. Default = false.

cabhSec2FwEventLogLastReset – This object reports the last time the log was cleared.

11.6.4.9.3 Firewall policy configured ruleset MIB objects

The firewall policy MIB objects provide a way for the cable operator to configure rules that will be used by the firewall to filter traffic. The cable operator can create any configured ruleset needed to filter traffic passing through the firewall on the PS. The firewall filtering policy Configured Ruleset MIB objects are based on the minimum set of filtering requirements. The firewall's filtering capability is similar to the filters defined in the cable industry CM MIB objects, specified in [RFC 2669]. Therefore, IPCable2Home had adopted some of the filtering objects already defined in [RFC 2669], and add some firewall specific MIB objects within the Security MIB.

Within [RFC 2669], the docsDevFilterIpTable provides the basic filtering properties. The docsDevFilterIpTable contains a sequence, docsDevFilterIpEntry, of MIB object. Each row in the table describes rules associated with IP addresses which is then compared to IP packets traversing the firewall. The template includes source and destination IP addresses (and their associated masks), upper level protocol (e.g., TCP, UDP), as well as the source and destination port ranges. The cabhSec2FwLocalFilterIpTable is similar to the docsDevFilterIpTable and can also be used to define filtering properties. Both the docsDevFilterIpTable and the cabhSec2FwLocalFilterIpTable are the heart of policy implementation for the Configured Ruleset. It is in these MIB tables that the Configured Ruleset policy is defined and constructed.

IPCable2Home defines a docsDevFilterIPTable extension, cabhSec2FwFilterScheduleTable that provides filter attributes for start time, end time and day of week to the corresponding filter entries in the docsDevFilterIPTable. These attributes also exist in the cabhSec2FwLocalFilterIpTable and allow a rule or filter to be enforced via the day of week (Sunday, Monday, Tuesday, Wednesday, Thursday, Friday, or Saturday), during a start and end time. These filter time settings can be used to support parental control applications. For example, a parent may request that communications be denied between the WAN and the child's computer for Monday through Friday, 9 p.m. to 7 a.m. and on Saturday and Sunday, 10 p.m. to 8 a.m. The cabhSec2FwFilterScheduleTable also provides a description attribute that can be used for placing comments/notes that help identify what the filter entry is being used for. Filter rule entries in the docsDevFilterIpTable MIB object MUST always be applied if their associated cabhSec2FwFilterScheduleTable MIB objects have the following values:

- cabhSec2FwFilterScheduleStartTime = 0;
- cabhSec2FwFilterScheduleEndTime = 2359; and
- cabhSec2FwFilterScheduleDOW = 0xFE.

Filter rule entries in the cabhSec2FwLocalFilterIpTable MIB object MUST always be applied if their MIB objects have the following values:

- cabhSec2FwLocalFilterStartTime = 0;
- cabhSec2FwLocalFilterEndTime = 2359; and
- cabhSec2FwLocalFilterDOW = 0xFE.

The combination of filters defined in [RFC 2669], and in the Security MIB, allow for any rules to be created based on any combination of source IP address, destination IP address, source port, destination port, time of day, and day of week.

If there is not a match when the PS is comparing each inbound or outbound packet to the rules in the docsDevFilterIpTable, cabhSec2FwLocalFilterIpTable, or the cabhSec2FwFactoryDefaultFilterTable, then the PS MUST apply the General Behaviour Rules and the minimum set of firewall capabilities and architecture, as defined in 11.6.4.3.1 and 11.6.4.3.3. The docsDevFilterIpDefault flag defined in [RFC 2669] MUST be ignored.

The following MIB objects MUST be implemented from [RFC 2669] to create the IPCable2Home version of the docsDevFilterIpTable. Unless otherwise noted in this clause, the functionality is as specified in [RFC 2669]:

- docsDevFilterIpTable >>DocsDevFilterIpEntry
  - **docsDevFilterIpIndex**
    - Consistent with [RFC 2669], the filter with the lowest index is always applied, meaning the filter is checked, then the PS MUST continue checking filters and apply the filter with the highest index in the case of conflicts.
  - **docsDevFilterIpStatus**
  - **docsDevFilterIpControl**
    - The PS MUST ignore the setting (3) for policy; IPCable2Home does not use the policy table.
  - **docsDevFilterIpIfIndex**
    - This object MUST use a default value of 255 (Aggregated LAN Interfaces).
    - The PS MUST support the values 1 (one), for filters at the PS WAN side, and 255 (the 'Aggregated LAN Interface' interface) for filters at the PS LAN side.
  - **docsDevFilterIpDirection**
    - For IPCable2Home, this value represents direction in relationship to the assigned docsDevFilterIpIfIndex in this particular rule, meaning that the PS MUST represent traffic direction (inbound, outbound, or both), relative to the indicated ifIndex. Vendor assigned ifIndex values MUST follow the same rule for application of direction. For example, IPCable2Home assigns the number 255 to the aggregated LAN interface. In this case, the PS will see inbound traffic on ifIndex 255, as all traffic coming from the LAN going to or traversing through the PS and outbound traffic on ifIndex 255, as all traffic going to the LAN coming from or traversing through the PS.
  - **docsDevFilterIpBroadcast**
    - It is expected that this will always be the default value of false. Therefore, the rule will apply to all traffic.
  - **docsDevFilterIpSaddr**
  - **docsDevFilterIpSmask**
  - **docsDevFilterIpDaddr**
  - **docsDevFilterIpDmask**
  - **docsDevFilterIpProtocol**
  - **docsDevFilterIpSourcePortLow**
  - **docsDevFilterIpSourcePortHigh**
  - **docsDevFilterIpDestPortLow**
  - **docsDevFilterIpDestPortHigh**
  - **docsDevFilterIpMatches**
    - Since filter rules are applied to session initiation traffic, this object at a minimum MUST count the number of times this filter is matched when a new session is attempted.
  - **docsDevFilterIpTos**
    - This object can be ignored, its function is not required.
  - **docsDevFilterIpTosMask**
    - This object can be ignored, its function is not required.
  - **docsDevFilterIpContinue**

- This object MUST always be set to true so the PS will continue checking filters until all the filters have been checked. Unlike RFC 2669, this object MUST NOT trigger a discard until all the filters have been checked and there are no later filters which require the packet to be accepted.

– **docsDevFilterIpPolicyId**

- This object can be ignored, its function is not required.

Additionally, the firewall MUST support the following MIB objects as specified in the Security MIB document:

- **cabhSec2FwFilterScheduleStartTime**
- **cabhSec2FwFilterScheduleEndTime**
- **cabhSec2FwFilterScheduleDOW**
- **cabhSec2FwFilterScheduleDescr**
- **cabhSec2FwLocalFilterIpTable**

11.6.4.9.4 Firewall factory default ruleset MIB objects

The IPCable2Home Firewall Factory Default Ruleset MIB objects provide a way for the cable operator to view the IPCable2Home Factory Default Rules, which are exceptions to the general rules, or General Firewall Behaviour defined in Tables 11-18 and 11-19. For more information on the Default Ruleset MIB objects used for filtering, please refer to the Security MIB for description of the cabhSec2FwFactoryDefaultFilterTable and its entries.

11.7 Additional security MIB objects in the PS

The firewall MIB objects are described in the firewall clause (see 11.6). This clause describes the remaining security MIB objects required. The security MIB objects are defined in more detail and MUST be supported as defined in Annex A.

11.7.1 Secure software download MIB objects

Secure software download follows the design as created by Annex B/J.112, and as such, the MIB objects can be reused in the PS just as the CM uses them. The PKI structure for IPCable2Home is defined separately and therefore some of the certificate MIBs MUST be used as defined by IPCable2Home, not by the J.112 MIBs, as currently written in [draft-ietf-ipcdn-bpiplus-mib-05].

The Standalone PS MUST support the following MIB objects as defined in the CL-SP-MIB-CLABDEF-I03-030411 [see E.6]:

- **clabCVCRootCACert** – Code Verification Root CA used for CVC validation.
- **clabCVCCACert** – Code Verification CA used for CVC validation.
- **clabMfgCACert** – Manufacturer CA Certificate used to store the Mfg CA Cert.

The Standalone PS MUST support the following software download MIB objects defined in [draft-ietf-ipcdn-bpiplus-mib-05]:

- **docsBpi2CodeDownloadGroup** – Collection of objects that provide authenticated software download support. The docsBpi2CodeDownloadGroup includes:
  - **docsBpi2CodeDownloadStatusCode** – Indicates the result of the latest configuration file CVC verification, SNMP CVC verification, or code file verification.
  - **docsBpi2CodeDownloadStatusString** – Additional information to the status code.
  - **docsBpi2CodeMfgOrgName** – The device manufacturer's organizationName.

– **docsBpi2CodeMfgCodeAccessStart** – The device manufacturer's current codeAccessStart value referenced to Greenwich Mean Time (GMT).
– **docsBpi2CodeMfgCvcAccessStart** – The device manufacturer's current cvcAccessStart value referenced to Greenwich Mean Time (GMT).
– **docsBpi2CodeCoSignerOrgName** – The Co-Signer's organizationName.
– **docsBpi2CodeCoSignerCodeAccessStart** – The co-signer's current codeAccessStart value referenced to Greenwich Mean Time (GMT).
– **docsBpi2CodeCoSignerCvcAccessStart** – The co-signer's current cvcAccessStart value referenced to Greenwich Mean Time (GMT).
– **docsBpi2CodeCvcUpdate** – Triggers the device to verify the CVC and update the cvcAccessStart value.

11.7.2 Security configuration file MIB objects

The PS MUST support the following configuration file download MIB object as defined in the Security MIB:

cabhPsDevProvConfigHash – SHA-1 [FIPS 186] hash of the entire content of the configuration file, taken as a byte string.

11.7.3 Security service provider MIB objects

The PS MUST support the following service provider authentication MIB object as defined in the Security MIB:

clabSrvcPrvdrRootCACert – The Service Provider Root CA used to validate certificates of devices on the service provider's network.

11.7.4 PS certificate MIB objects

The PS MUST support the following PS Certificate MIB object as defined in the Security MIB:

cabhSecCertPsCert – The X.509 DER-encoded PS certificate used to provide secure identity of the PS.

11.7.5 Kerberos MIB objects

The need of Kerberos within IPCable2Home is a subset of the functionality required by IPCablecom. The following MIB objects are required for IPCable2Home and the PS MUST support these MIB objects, as defined in the Security MIB:

- **cabhSecKerbPKINITGracePeriod** – The number of minutes prior to current ticket expiration for the PS to initiate a request with the KDC for a new ticket.
- **cabhSecKerbTGSGracePeriod** – The number of minutes prior to current ticket expiration for the PS to initiate a request with the KDC for a new ticket.
- **cabhSecKerbUnsolicitedKeyMaxTimeout** – The maximum timeout value for the AP Req/Rep exchange.
- **cabhSecKerbUnsolicitedKeyMaxRetries** – The maximum number of retries the PS is allowed to attempt AP Req/Rep negotiation.

11.8 Secure software download for the PS

11.8.1 Goals of secure software download

Secure Software Download goals include the following:

- The cable operator can securely load code into the PS as needed.
- The cable operator can manage secure downloads with various configuration policies.
- The security of the download will provide integrity, authentication, and if possible, encryption.
- The PS will only download images appropriate for the device.

11.8.2 Secure software download design guidelines

See Table 11-22.

Table 11-22/J.192 – IPCable2Home security system design guidelines

Reference	Guidelines
SEC13	The cable operator will have the ability to securely download software images to the PS element.

11.8.3 Secure software download system description

Secure software download ensures that only a software image can be downloaded to the PS if the image is created by the same manufacturer. It also ensures that the image has not been modified since the manufacturer signed the code image. The image can also be signed by a Certification Testing Laboratory, as a co-signer, to guarantee that the image has been certified. For additional security on the download process, the cable operator can optionally sign any image as a co-signer to ensure that only images will be loaded into the PS that the cable operator has approved. The control mechanism for secure software download is to insert the code verification certificates (CVCs) into the configuration file which match the CVCs on the code image to be downloaded. After the PS has received CVC(s) in the configuration file, the PS is enabled to download the new code image when triggered via configuration file, or SNMP SET.

11.8.4 Secure software download requirements

A Standalone PS Element MUST be capable of remotely downloading a software image over the network. As described in 6.3.3.2.4.9, secure software download to an Embedded PS is controlled by the cable modem. The new software image would allow the cable operator to improve performance, accommodate new functions and features, correct design deficiencies, and to allow a migration path for IPCable2Home devices as the IPCable2Home 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:

- The mechanism used for software download MUST be TFTP file transfer.
- The software download MUST be initiated in one of two ways:
  1) An SNMP SET request issued by the NMS to the docsDevSwAdminStatus;
  2) via the PS element's configuration file.

  If the Software Upgrade File Name in the configuration file does not match the current software image of the device, the PS element MUST request the specified file via TFTP from the Software Server.

- The PS element MUST verify that the downloaded software image is appropriate for itself. If the downloaded software image is appropriate, the PS element MUST write the new software image to non-volatile storage. Once the file transfer is completed successfully, the device MUST restart itself with the new code image.
- If the PS element is unable to complete the file transfer for any reason, the PS element MUST remain capable of accepting new software downloads (without operator or user interaction), even if power or connectivity is interrupted between attempts.
- The PS element MUST log software download failures and can report failures asynchronously to the network manager.
- Where software has been upgraded to meet a new version of this Recommendation, then it is critical that the software MUST work with the previous version in order to allow a gradual transition of units on the network.
- The PS element MUST authenticate the downloaded software image.
- The PS element MUST verify that the downloaded code has not been altered from the original form in which it was provided by the trusted source.
- The software download process MUST provide a cable operator with mechanisms to upgrade or downgrade the code version of the IPCable2Home elements.
- The software download process MUST provide options for a cable operator to dictate their own download policies.
- The code file manufacturer MUST apply a Code Verification Signature (CVS) over the code image and any other authenticated attributes as defined in this Recommendation for the PKCS#7 structure digital signature to the code file; the private key used to apply the signature MUST be bound to a public key certificate that chains up to the CVC root. The manufacturer's signature authenticates the source and integrity of the code file.
- A Co-Signer (cable operator or CTL) MAY countersign the code file in addition to the manufacturer's signature.
- The PS element MUST be able to process a PKCS#7 digital signature and an X.509 certificate as defined in 11.8.4.1.1 and 11.3.4.1.1, respectively.
- The PS element MUST be able to update the CVC Root CA Certificate stored in the device after the Certificate has been validated if contained in a Code File as a TLV.
- The PS element MUST be able to replace the Manufacturer CA Certificate(s) stored in the device after the Certificate has been validated if contained in a Code File as a TLV.
- The PS element MUST be able to update the CVC CA Certificate stored in the device after the Certificate has been validated if contained in a Code File as a TLV.
- The PS element MUST be able to update the Service Provider Root CA Certificate stored in the device after the Certificate has been validated if contained in a Code File as a TLV.

The optional downloading of the Service Provider Root CA Certificate, CVC Root CA Certificate, CVC CA Certificate, and/or the Manufacturer CA Certificate, as a part of the Code File, are clearly separated from the code image and the other parameters in the code download file. It is possible to change the 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 since the Service Provider Root CA Certificate, CVC Root CA Certificate, CVC CA Certificate, and/or the Manufacturer CA Certificate, or SignedData parameters, are appended to the code image.

An IPCable2Home Complaint Residential Gateway device MAY include a cable modem and the PS Element, as separate entities or embedded as defined in the architecture clause (see clause 5).

- 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 standalone entities, the software download for the IPCable2Home elements MUST be performed by the PS Element, as described below in this specification.

11.8.4.1 Code download file structure for secure software download

For secure software download, the code download file is a file built using a [RFC 2315] compliant structure that has been defined in a specific format for use with PS Elements. The code file MUST comply with [RFC 2315] and MUST be DER encoded. The code file MUST match the structure shown in Table 11-23.

When certificates are downloaded as a part of the Code File, the certificates MAY be contained in the fields as specified in Table 11-23, and separated from the actual code image contained in the CodeImage field.

Table 11-23/J.192 – Code file structure

Code file	Description
PKCS #7 Digital Signature {	
ContentInfo	
ContentType	SignedData
SignedData ()	EXPLICT signed-data content value: includes CVS and X.509 compliant CVSs
} *end [RFC 2315] 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 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*	

11.8.4.1.1 Signed data

The code download file will contain the information in a [RFC 2315] Signed Data content type as shown in Table 11-24. Though maintaining compliance to [RFC 2315], the structure used has been restricted in format to ease the processing performed by the PS to validate the signature. The [RFC 2315] Signed Data MUST be DER encoded and exactly match the structure shown below, except for any change in order required to DER encode (e.g., the ordering of SET OF attributes). The PS element SHOULD reject the [RFC 2315] signature if the [RFC 2315] Signed Data does not match the DER encoded structure.

Table 11-24/J.192 – PKCS#7 signed data

PKCS #7 field	Description
Signed Data {	
version	1
digestAlgorithms	SHA-1
contentInfo	
contentType	data (SignedContent is concatenated at the end of the PKCS#7 structure)
certificates {	
mfgCVC	(REQUIRED for all code files) (Note 1)
co-signerCVC	(OPTIONAL; required for co-signatures) (Note 2)
} *end certificates*	
signerInfos{	
MfgSignerInfo {	(REQUIRED for all code files)
version	1
issuerAndSerialNumber	
issuer	
countryName	US
organizationName	CableLabs
commonName	CableLabs CVC Root CA
serialNumber	<Mfg CVC serial number>
digestAlgorithm	SHA-1
authenticatedAttributes	
contentType	data (contentType of signedContent)
signingTime	UTCTime (GMT), YYMMDDhhmmssZ
messageDigest	(digest on the content together with the signer's authenticated attributes as defined in [PKCS#7])
digestEncryptionAlgorithm	rsaEncryption
encryptedDigest	
} *end mfg signer info*	
CoSignerInfo {	(OPTIONAL; required for co-signatures)
version	1
issuerAndSerialNumber	
issuer	
countryName	US
organizationName	CableLabs
commonName	CableLabs CVC CA
serialNumber	<Co-Signer CVC serial number>
digestAlgorithm	SHA-1
authenticatedAttributes	
contentType	data (contentType of signedContent)
signingTime	UTCTime (GMT), YYMMDDhhmmssZ
messageDigest	(digest on the content together with the signer's authenticated attributes as defined in [PKCS#7])
digestEncryptionAlgorithm	rsaEncryption
encryptedDigest	
} *end co-signer info*	
} *end signer infos*	
} *end signed data*	

NOTE 1 – Manufacturer CVC MUST comply with the format specified in Table 11-9.

NOTE 2 – Co-signer CVC MUST comply with the format specified in Table 11-10 or Table 11-11 depending on the type of co-signer, which can be CTL or Service Provider correspondingly.

11.8.4.1.2 Signed content

The signed content field of the code file contains the code image and the download parameters field, which possibly contains the following additional optional items:

- Service Provider Root CA Certificate.
- Certification Testing Laboratory (CTL) CVC Root CA Certificate.
- CTL CVC CA Certificate.
- Manufacturer CA Certificate.

The final code image is in a format compatible with the destination PS element. In support of the [RFC 2315] signature requirements, the code content is typed as data; i.e., a simple octet string. The format of the final code image is not specified here and will be defined by each manufacturer according to their requirements.

Each manufacturer SHOULD build their code with additional mechanisms that verify an upgrade code image is compatible with the destination PS element.

If included in the signed content field, a certificate is intended to replace the certificate currently stored in the PS element. If the code download and installation is successful, the PS element MUST replace its currently stored certificate with the new certificate received in the signed content field after the certificate has been validated. This new certificate will be used for subsequent verification.

11.8.4.1.3 Code signing keys

The [RFC 2315] digital signature uses the RSA Encryption Algorithm [PKCS #1] with SHA-1 [FIPS 186]. The PS element MUST be able to verify code file signatures. The public exponent is F_4 (65537 decimal).

11.8.4.1.4 Manufacturer CA Certificate

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

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

11.8.4.1.5 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 Rec. 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)

11.8.4.1.6 CVC Root CA Certificate

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

11.8.4.1.7 CVC CA Certificate

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

11.8.4.2 CVC format for secure software download

For secure software download, the format used for the CVC is 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 comply with Tables 11-9, 11-10 and 11-11 depending on type of CVC. The PS element SHOULD reject the CVC if it does not match with the corresponding table.

11.8.4.2.1 Certificate revocation

This Recommendation does not require or define the use of certificate revocation lists (CRLs). The PS element is not required to support CRLs. Operators can define and use CRLs to help manage code files provided to them by manufacturers. However, there is a method for revoking certificates based on the validity start date of the certificate. This method requires that an updated CVC be delivered to the PS element with an updated validity start time. Once the CVC is successfully validated, the X.509 validity start time will update the PS element's current value of cvcAccessStart.

11.8.4.3 Code file access controls

For secure software download, special control values are included in the code file for the PS element to check before it will validate a code image. The conditions placed on the values of these control parameters MUST be satisfied before the PS element will validate the CVC or the CVS, and accepts the code image.

11.8.4.3.1 Subject organization names

The PS element will recognize up to two names, at any one time, that it considers a trusted code-signing agent in the subject field of a code file CVC:

- The device manufacturer: The manufacturer name in the manufacturer's CVC subject field MUST exactly match the manufacturer name stored in the PS element's non-volatile memory by the manufacturer. A manufacturer CVC MUST always be included in the code file.
- A co-signing agent: It is permitted that another trusted organization co-sign code files destined to the device. In most cases, this is the cable operator controlling the current operating domain of the device. The organization name of the co-signer is communicated to the PS element via a co-signer's CVC in the configuration file when initializing the PS element's code verification process. The co-signer's organization name in the co-signer's CVC subject field MUST exactly match the co-signer's organization name previously received in the co-signer's initialization CVC and stored by the PS element.

The PS element MAY compare organization names using a binary comparison.

11.8.4.3.2 Time-varying controls

To mitigate the possibility of a PS element receiving a previous code file via a replay attack, the code files include a signing-time value in the PKCS #7 structure that can be used to indicate the time the code image was signed. The PS element MUST keep two UTC time values associated with each code-signing agent. One set MUST always be stored and maintained for the device's manufacturer. Additionally, if the code file is co-signed, the PS element MUST also store and maintain a separate set of time values for the co-signer.

These values are used to control code file access to the PS element by individually controlling the validity of the CVS and the CVC:

- codeAccessStart: a 12-byte UTC time value referenced to Greenwich Mean Time (GMT).
- cvcAccessStart: a 12-byte UTC time value referenced to GMT.

UTCTime values in the CVC MUST be expressed as GMT and MUST include seconds. That is, they MUST be expressed in the following form: YYMMDDhhmmssZ. The year field (YY) MUST be interpreted as follows:

- Where YY is greater than or equal to 50, the year shall be interpreted as 19YY.
- Where YY is less than 50, the year shall be interpreted as 20YY.

These values will always be referenced to Greenwich Mean Time, so the final ASCII character (Z) can be removed when stored by the PS element as codeAccessStart and cvcAccessStart.

The PS element MUST maintain each of these time values in a format that contains equivalent time information and accuracy to the 12-character UTC format (i.e., YYMMDDhhmmss). The PS element MUST accurately compare these stored values with UTC time values delivered to the PS element in a CVC. These requirements are discussed later in this Recommendation.

The values of codeAccessStart and cvcAccessStart corresponding to the PS Element's manufacturer MUST NOT decrease. The value of codeAccessStart and cvcAccessStart, corresponding to the co-signer, MUST NOT decrease as long as the co-signer does not change and the PS element maintains that co-signer's time-varying control values.

11.8.4.4 Code upgrade initialization

11.8.4.4.1 Manufacturer initialization

It is the responsibility of the manufacturer to correctly install the initial code version in the PS Element.

In support of secure software download, values for the Manufacturer's time-varying controls MUST be loaded into the PS Element's non-volatile memory:

- PS Element manufacturer's organizationName.
- Manufacturer's time-varying control values:
  - codeAccessStart initialization value;
  - cvcAccessStart initialization value.

The organization name of the PS Element manufacturer MUST always be present in the device. The PS Element manufacturer's organizationName MAY be stored in the device's code image. The manufacturer name used for code upgrade is not necessarily the same name used in the Manufacturer CA certificate.

The time-varying control values, codeAccessStart, and cvcAccessStart, MUST be initialized to a UTCTime compatible with the validity start time of the manufacturer's latest CVC. These time-varying values will be updated periodically under normal operation via manufacturer's CVCs that are received and verified by the PS element.

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.

11.8.4.4.2 Network initialization

In support of code verification, the PS configuration file is used as an authenticated means in which to initialize the code verification process. In the PS element configuration file, the PS element receives configuration settings relevant to code upgrade verification.

The configuration file SHOULD always include the most up-to-date CVC applicable for the destination PS element. When the configuration file is used to initiate a code upgrade, it MUST include a Code Verification Certificate (CVC) to initialize the PS element for accepting code files according to this Recommendation. Regardless of whether a code upgrade is required, a CVC in the configuration file MUST be processed by the PS element. A configuration file MAY contain:

- No CVC – The PS element MUST NOT accept a code file.
- A Manufacturer's CVC only – The PS element MUST verify that the manufacturer's CVC chains up to the CVC Root before accepting a code file. When the PS element's configuration file only contains a valid Manufacturer's CVC, the device will only require a manufacturer signature on the code files. In this case, the PS element MUST NOT accept code files that have been co-signed.
- A Co-Signer's (cable operator or CTL) CVC only – The PS element MUST verify the Co-Signer CV chains up to the CVC Root before accepting a code file. When the PS element's configuration file contains a valid co-signer's CVC, it is used to initialize the device with a co-signer. Once validated, the name of the CVC's subject organizationName will become the code co-signer assigned to the PS element. In order for a PS element to subsequently accept a code image, the co-signer, in addition to the device manufacturer, MUST have signed the code file.
- Both a Manufacturer's CVC and a Co-Signer's CVC – The PS element MUST verify that both CVCs chain up to the CVC Root before accepting a code file.

Before the PS element will enable its ability to upgrade code files on the network, it MUST receive a valid CVC in a configuration file. In addition, when the PS element's configuration file does not contain a valid CVC, which means that its ability to upgrade code files has been disabled, the PS element MUST reject any information in a CVC subsequently delivered via docsBpi2CodeCvcUpdate SNMP MIB object.

The organization name of the PS Element manufacturer and the manufacturer's time-varying control values MUST always be present in the PS element. If the PS element is initialized to accept code co-signed by an additional code-signer, the name of the organization and their corresponding time-varying control values MUST be stored and maintained while operational. Space MUST be allocated in the PS element's memory for the following co-signer's control values:

- co-signing agent's organizationName.
- co-signer's time-varying control values:
  - cvcAccessStart;
  - codeAccessStart.

The manufacturer's set of these values MUST be stored in the PS element's non-volatile memory and not lost when the device's main power source is removed or during a reboot.

When a co-signer is assigned to the PS element, the co-signer's set of CVC values MUST be stored in the PS element's memory. The PS element MAY retain these values in non-volatile memory that will not be lost when the device's main power source is removed or during a reboot. However, when assigning a PS element a co-signer, the CVC is always in the configuration file. Therefore, the PS element will always receive the co-signer's control values during the initialization phase and is not required to store the co-signer's time-varying control values when main power is lost or during a reboot process.

11.8.4.4.3 CVC processing

To expedite the delivery of an updated CVC without requiring the PS to process a code upgrade, the CVC MAY be delivered in the configuration file or an SNMP Set message. The format of the CVC is the same whether it is in a code file, configuration file, or SNMP message.

11.8.4.4.3.1 Processing the configuration file CVC

When a CVC is included in the configuration file, the PS element MUST verify the CVC before accepting any of the code upgrade settings it contains. At receipt of the CVC in the configuration file, the PS element MUST perform the following validation and procedural steps. If any of the following verification checks fail, the PS element MUST immediately halt the CVC verification process and log the error if applicable. If the PS Configuration File does not include a CVC that validates properly, the PS element MUST NOT download the upgrade code files whether triggered by the PS Configuration File, or via docsDevSwAdminStatus SNMP MIB object. In addition, if the PS Configuration File does not include a CVC that validates properly (manufacturer or co-signer CVC), the PS element is not required to process CVCs subsequently delivered, via docsBpi2CodeCvcUpdate SNMP MIB object, and MUST NOT accept information from these CVCs (i.e., the PS element MUST ignore any SNMP Set requests made to docsBpi2CodeCvcUpdate SNMP MIB object).

At receipt of the CVC in a configuration file, the PS element MUST:

1) Verify that the CVC complies with the structure and values as required in 11.3.4.2.
2) Check the CVC subject organization name:
    a) If the CVC is a Manufacturer's CVC (Type 32) then:
        i) If the organizationName is identical to the device's manufacturer name, then this is the manufacturer's CVC. In this case, the PS element MUST verify that the manufacturer's CVC validity start time is greater-than or equal-to the manufacturer's cvcAccessStart value currently held in the PS element.
        ii) If the organizationName is not identical to the device's manufacturer name, then this CVC MUST be rejected and the error logged.
    b) If the CVC is a Co-signer's CVC (Type 33) then:
        i) If the organizationName is identical to the PS element's current code co-signer, then this is the current co-signer's CVC, and the PS element MUST verify that the validity start time is greater-than or equal-to the co-signer's cvcAccessStart value currently held in the PS element.
        ii) If the organizationName is not identical to the current code co-signer name, then after the CVC has been validated (and registration is complete), this subject organization name will become the PS element's new code co-signer. The PS element MUST NOT accept a code file unless it has been signed by the manufacturer, and co-signed by this code co-signer.

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.

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.

11.8.4.4.3.2 Processing the SNMP CVC

The PS element MUST process SNMP delivered CVCs when enabled to upgrade code files. Otherwise, all CVCs delivered via SNMP MUST be rejected. When validating the CVC delivered via SNMP, the PS element MUST perform the following validation and procedural steps.

NOTE – If any of the following verification checks fails, the PS element MUST immediately halt the CVC verification process, log the error if applicable, and remove all remnants of the process to that step.

The PS element MUST:

1) Verify that the CVC complies with the structure and values as required in 11.3.4.2.2.2.

2) Check the CVC subject organization name:

   a) If the organizationName is identical to the device's manufacturer name, then this is the manufacturer's CVC. In this case, the PS element MUST verify that the manufacturer's CVC validity start time is greater than the manufacturer's cvcAccessStart value currently held in the PS element.

   b) If the organizationName is identical to the PS element's current code co-signer, then this is a current co-signer's CVC and the validity start time MUST be greater than the co-signer's cvcAccessStart value currently held in the PS element.

   c) If the organizationName is not identical to device's manufacturer or current co-signer's name, then the PS element MUST immediately reject this CVC.

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.

11.8.4.5 Code signing requirements

11.8.4.5.1 Certification Authority (CA) requirements

Code Verification Certificates (CVCs) are signed and issued by the Certification Testing Laboratory (CTL) CVC CA. The CVC MUST be exactly as specified in 11.3.4.2.2.2 depending on type of CVC.

In any alternate format, all the information MUST be maintained and the original format MUST be reproduced; e.g., as a 32-bit non-zero integer, with an integer value of 0 representing the absence of a code-signer.

11.8.4.5.2 Manufacturer CVC requirements

To sign their code files, the manufacturer MUST obtain a valid CVC from the CTL CVC CA. All manufacturer code images provided to a cable operator for remote upgrade of a device MUST be signed according to the requirements defined in this Recommendation. When signing a code file, a manufacturer MAY choose not to update the [RFC 2315] signingTime value in the manufacturer's signing information. This Recommendation requires that the [RFC 2315] signingTime value be equal to or greater than the CVC's validity start time. If the manufacturer uses a signingTime equal to the CVC's validity start time when signing a series of code files, those code files can be used and re-used. This allows a cable operator to use the code file to upgrade or downgrade the code version for that manufacturer's devices. These code files will be valid until a new CVC is generated and received by the PS element.

11.8.4.5.3 Cable operator requirements

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

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

When signing a code file, the cable operator MUST co-sign the file content according to the PKCS #7 signature standard, and include their cable operator CVC, as defined in 11.8.4.1.1. IPCable2Home does not require a cable operator to co-sign code files. However, when the cable operator follows all the rules defined in this Recommendation for preparing a code file, the PS element MUST accept it.

11.8.4.6 Triggering process

Code downloads, regardless of the provisioning mode, can be initiated during the provisioning and registration process via a configuration-file-initiated download, or, during normal operation, using an SNMP-initiated download command. The PS element MUST support both methods.

NOTE – Prior to triggering a secure software download, appropriate CVC information MUST be included in the configuration file. If the operator decides to use the SNMP-initiated download as a method to trigger a secure software download, it is recommended that CVC information always be present in the configuration file so that a PS element will always have the CVC information initialized when needed. If the operator decides to use the configuration-file-initiated download as a method to trigger secure software download, CVC information is needed to be present in the configuration file at the time the device is rebooted to get the configuration file that will trigger the upgrade.

11.8.4.6.1 SNMP-initiated software download

From a network management station:

- Set docsDevSwServer to the address of the TFTP server for software upgrades.
- Set docsDevSwFilename to the file pathname of the software upgrade image.
- Set docsDevSwAdminStatus to Upgrade-from-mgt. docsDevSwAdminStatus MUST persist across reset/reboots until overwritten from an SNMP manager, or via the PS element configuration file.

The default state of docsDevSwAdminStatus MUST be allowProvisioningUpgrade{2} until it is overwritten by ignoreProvisioningUpgrade{3}, following a successful SNMP-initiated software upgrade, or otherwise altered by the management station. docsDevSwOperStatus MUST persist across resets to report the outcome of the last software upgrade attempt.

If a PS element suffers a loss of power or resets during SNMP-initiated upgrade, the PS element MUST resume the upgrade without requiring manual intervention, and when the PS element resumes the upgrade process:

- docsDevSwAdminStatus MUST be Upgrade-from-mgt{1}.
- docsDevSwFilename MUST be the filename of the software image to be upgraded.
- docsDevSwServer MUST be the address of the TFTP server containing the software upgrade image to be upgraded.
- docsDevSwOperStatus MUST be inProgress{1}.
- docsDevSwCurrentVers MUST be the current version of software that is operating on the device.

In case the PS element reaches the maximum number of retries (max retries = 3) resulting from multiple losses of power or resets during an SNMP-initiated upgrade, the PS element's status MUST adhere to the following requirements after it is registered:

- docsDevSwAdminStatus MUST be allowProvisioningUpgrade{2}.
- docsDevSwFilename MUST be the filename of the software that failed the upgrade process.
- docsDevSwServer MUST be the address of the TFTP server containing the software that failed the upgrade process.
- docsDevSwOperStatus MUST be other{5}.
- docsDevSwCurrentVer MUST be the current version of software that is operating on the device.

If a PS element exhausts the required number of TFTP retries by issuing a total of 16 consecutive retries, the PS element MUST fall back to the last known working image, proceed to an operational state, and adhere to the following requirements:

- docsDevSwAdminStatus MUST be allowProvisioningUpgrade{2}.
- docsDevSwFilename MUST be the filename of the software that failed the upgrade process.
- docsDevSwServer MUST be the address of the TFTP server containing the software that failed the upgrade process.
- docsDevSwOperStatus MUST be failed{4}.
- docsDevSwCurrentVer MUST be the current version of software that is operating on the device.

After the PS element has completed the SNMP-initiated secure software upgrade, the PS element MUST reboot and become operational with the correct software image. When the device is operational, it MUST adhere to the following requirements:

- set its docsDevSwAdminStatus to ignoreProvisioningUpgrade{3}.
- set its docsDevOperStatus to completeFromMgt{3}.
- reboot.

The PS element MUST properly use ignoreProvisioningUpgrade status to ignore the software upgrade value that can be included in the PS element configuration file. The PS MUST become operational with the correct software image and it MUST adhere to the following requirements:

- docsDevSwAdminStatus MUST be ignoreProvisioningUpgrade{3}.
- docsDevSwFilename MAY be the filename of the software currently operating on the PS element.

- docsDevSwServer MAY be the address of the TFTP server containing the software that is currently operating on the PS element.
- docsDevSwOperStatus MUST be completeFromMgt{3}.
- docsDevSwCurrentVer MUST be the current version of the software that is operating on the PS element.

In the case where the PS element successfully downloads (or detects during download) an image that is not intended for the device, the:

- DocsDevSwAdminStatus MUST be allowProvisioningUpgrade{2}.
- DocsDevSwFilename MUST be the filename of the software that failed the upgrade.
- DocsDevSwServer MUST be the address of the TFTP server containing the software that failed the upgrade process.
- DocsDevSwOperStatus MUST be other{5}.
- DocsDevSwCurrentVer MUST be the current version of software that is operating on the device.

In the case where the PS element determines that the download image is damaged or corrupted, the PS element MUST reject the newly downloaded image. The PS element MAY re-attempt to download if the MAX number of TFTP sequence retries has not been reached. If the PS element chooses not to retry and the MAX number of TFTP sequence retries has not been reached, the PS element MUST fall back to the last known working image and proceed to an operational state, generate an appropriate event notification, as specified in 11.8.4.8, and adhere to the following requirements:

- DocsDevSwAdminStatus MUST be allowProvisioningUpgrade{2}.
- DocsDevSwFilename MUST be the filename of the software that failed the upgrade.
- DocsDevSwServer MUST be the address of the TFTP server containing the software that failed the upgrade process.
- DocsDevSwOperStatus MUST be other{5}.
- DocsDevSwCurrentVer MUST be the current version of software that is operating on the device.

In the case where the PS element determines that the image is damaged or corrupted, the PS element MUST reject the newly downloaded image. The PS element MAY re-attempt to download the new image if the MAX number of TFTP sequence retries has not been reached. On the 16th consecutive failed software download attempt, the PS element MUST fall back to the last known working image and proceed to an operational state. In this case, the PS element is required to send two notifications; one to notify that the MAX TFTP retry limit has been reached, and another to notify that the image is damaged. Immediately after the PS element reaches the operational state, the PS element MUST adhere to the following requirements:

- DocsDevSwAdminStatus MUST be allowProvisioningUpgrade{2}.
- DocsDevSwFilename MUST be the filename of the software that failed the upgrade.
- DocsDevSwServer MUST be the address of the TFTP server containing the software that failed the upgrade process.
- DocsDevSwOperStatus MUST be other{5}.
- DocsDevSwCurrentVer MUST be the current version of software that is operating on the device.

11.8.4.6.2 Configuration-file-initiated software download

The configuration-file-initiated software download is initiated in a standalone PS by including the Software Upgrade File Name parameter (TLV-9) AND the Software Upgrade TFTP Server parameter (TLV-21) in its PS Configuration File. An embedded PS MUST ignore TLV-9 and TLV-21 if they are present in its PS Configuration File, since the software upgrade of an embedded PS is controlled by the cable modem. If the Software Upgrade File Name parameter (TLV-9) with a valid value AND the Software Upgrade TFTP Server parameter (TLV-21) with a valid value are present in the standalone PS element's PS Configuration File, AND if the value of the Software Upgrade File Name parameter does not match the current software image file name, i.e., the value of docsDevSwFilename, the PS element MUST request the specified file, via TFTP, from the server whose address was provided in the Software Upgrade TFTP Server parameter.

If a standalone PS receives a PS Configuration File in which both the Software Upgrade File Name parameter (TLV-9) and a TLV-28 setting the docsDevSwFilename object are present, AND the values of TLV-9 and docsDevSwFilename are different, the PS MUST reject the PS Configuration File, report Event ID 73040102 (Invalid TLV Format/contents), preserve all object values that existed before the attempt to process this bad configuration file, and reset.

NOTE – The Software Server IP Address is a separate parameter. If present, the PS element MUST attempt to download the specified file from this server. If not present, the PS element MUST attempt to download the specified file from the configuration file server.

In a case where the PS element reaches the maximum number of retries (max retries = 3) resulting from multiple loss of powers, or resets during a configuration-file-initiated upgrade, the PS element's status MUST adhere to the following requirements, after it is registered:

- docsDevSwAdminStatus MUST be allowProvisioningUpgrade{2}.
- docsDevSwFilename MUST be the filename of the software that failed the upgrade process.
- docsDevSwServer MUST be the address of the TFTP server containing the software that failed the upgrade process.
- docsDevSwOperStatus MUST be other{5}.
- docsDevSwCurrentVer MUST be the current version of software that is operating on the device.

If a PS element exhausts the required number of TFTP retries by issuing a total of 16 consecutive retries, the PS element MUST fall back to the last known working image, proceed to an operational state, and adhere to the following requirements:

- docsDevSwAdminStatus MUST be allowProvisioningUpgrade{2}.
- docsDevSwFilename MUST be the filename of the software that failed the upgrade process.
- docsDevSwServer MUST be the address of the TFTP server containing the software that failed the upgrade process.
- docsDevSwOperStatus MUST be failed{4}.
- docsDevSwCurrentVer MUST be the current version of software that is operating on the device.

After the PS element has completed the configuration-file-initiated secure software upgrade, the PS element MUST reboot and become operational with the correct software image. After the PS element is registered the:

- docsDevSwAdminStatus MUST be allowProvisioningUpgrade{2}.
- docsDevSwFilename MAY be the filename of the software currently operating on the device.

- docsDevSwServer MAY be the address of the TFTP server containing the software that is currently operating on the device.
- docsDevSwOperStatus MUST be completeFromProvisioning{2}.
- docsDevSwCurrentVer MUST be the current version of the software that is operating on the device.

11.8.4.7 Code verification

For secure software download, the PS element MUST perform the verification checks presented in this clause. If any of the verification checks fails, or if any portion of the code file is rejected due to invalid formatting, the PS element MUST immediately halt the download process, log the error if applicable, remove all remnants of the process to that step, and continue to operate with its existing code.

The following verification checks can be made in any order, as long as all of the applicable checks presented in this clause are made:

1) The PS element MUST validate the manufacturer's signature information by verifying that the [RFC 2315] signingTime value is:
   a) equal to or greater than the manufacturer's codeAccessStart value currently held in the PS element;
   b) equal to or greater than the manufacturer's CVC validity start time;
   c) less than or equal to the manufacturer's CVC validity end time.
2) The PS element MUST validate the manufacturer's CVC by verifying that the:
   a) CVC is exactly the same as it is specified in Table 11-8.
   b) CVC subject organizationName is identical to the manufacturer name currently stored in the PS element's memory.
   c) CVC validity start time is equal to or greater than the manufacturer's cvcAccessStart value currently held in the PS element.
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.
4) The PS element MUST verify the manufacturer's code file signature:
   a) The PS element MUST perform a new SHA-1 hash over the SignedContent. If the value of the messageDigest does not match the new hash, the PS element MUST consider the signature on the code file as invalid.
   b) If the signature does not verify, all components of the code file (including the code image), and any values derived from the verification process MUST be rejected and SHOULD be immediately discarded.
5) If the manufacturer signature verifies, and a co-signing agent signature is required:
   a) The PS element MUST validate the co-signer's signature information by verifying that the:
      i) co-signer's signature information is included in the code file.
      ii) [RFC 2315] signingTime value is equal to or greater than the corresponding codeAccessStart value currently held in the PS element.
      iii) [RFC 2315] signingTime value is equal to or greater than the corresponding CVC validity start time.

iv) [RFC 2315] signingTime value is less than or equal to the corresponding CVC validity end time.

b) The PS element MUST validate the co-signer's CVC, by verifying that the:

i) CVC subject organizationName is identical to the co-signer's organization name currently stored in the PS element's memory.

ii) CVC is exactly the same as it is specified in Table 11-9 or Table 11-10 depending on the type of co-signer (CTL or Service Provider).

iii) CVC validity start time is equal to or greater than the cvcAccessStart value currently held in the PS element for the corresponding subject organizationName.

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 CTL CVC 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.

d) The PS element MUST verify the co-signer's code file signature.

e) The PS element MUST perform a new SHA-1 hash, over the SignedContent. If the value of the messageDigest does not match the new hash, the PS element MUST consider the signature on the code file as invalid.

f) If the signature does not verify, all components of the code file (including the code image), and any values derived from the verification process MUST be rejected and SHOULD be immediately discarded.

6) If the manufacturer's, and optionally, the co-signer's signature has verified, the code image can be trusted and installation can proceed. Before installing the code image, all other components of the code file and any values derived from the verification process, except the [RFC 2315] signingTime values and the CVC validity start values, SHOULD be immediately discarded.

7) If the code installation is unsuccessful, the PS element MUST reject the [RFC 2315] signingTime values and CVC validity start values it just received in the code file.

8) When the code installation is successful, the PS element MUST update the manufacturer's time-varying controls with the values from the manufacturer's signature information and CVC:

a) Update the current value of codeAccessStart with the [RFC 2315] signingTime value.

b) Update the current value cvcAccessStart with the CVC validity start value.

9) When the code installation is successful, and if the code file was co-signed, the PS element MUST update the co-signer's time-varying controls with the values from the co-signer's signature information and CVC:

a) Update the current value of codeAccessStart with the [RFC 2315] signingTime value.

b) Update the current value of cvcAccessStart with the CVC validity start value.

11.8.4.8 Error codes

Error codes are defined to reflect the failure states possible during the secure software download code verification process.

1) Improper code file controls:

a) CVC subject organizationName for manufacturer does not match the PS element's manufacturer name.

b) CVC subject organizationName for code co-signing agent does not match the PS element's current code co-signing agent.

c) The manufacturer's [RFC 2315] signingTime value is less than the codeAccessStart value currently held in the PS element.

d) The manufacturer's [RFC 2315] validity start time value is less than the cvcAccessStart value currently held in the PS element.

e) The manufacturer's CVC validity start time is less than the cvcAccessStart value currently held in the PS element.

f) The manufacturer's [RFC 2315] signingTime value is less than the CVC validity start time.

g) Missing or improper extended key-usage extension in the manufacturer CVC.

h) The co-signer's [RFC 2315] signingTime value is less than the codeAccessStart value currently held in the PS element.

i) The co-signer's [RFC 2315] validity start time value is less than the cvcAccessStart value currently held in the PS element.

j) The co-signer's CVC validity start time is less than the cvcAccessStart value currently held in the PS element.

k) The co-signer's [RFC 2315] signingTime value is less than the CVC validity start time.

l) Missing or improper extended key-usage extension in the co-signer's CVC.

2) Code file manufacturer CVC validation failure.

3) Code file manufacturer CVS validation failure.

4) Code file co-signer CVC validation failure.

5) Code file co-signer CVS validation failure.

6) Improper Configuration File CVC format (e.g., Missing or improper key usage attribute).

7) Configuration File CVC validation failure.

8) Improper SNMP CVC format:

a) CVC subject organizationName for manufacturer does not match the device's manufacturer name.

b) CVC subject organizationName for code co-signing agent does not match the PS element's current code co-signing agent.

c) The CVC validity start time is less than or equal to the corresponding subject's cvcAccessStart value currently held in the PS element.

d) Missing or improper key usage attribute.

9) SNMP CVC validation failure.

11.8.4.9 Software downgrade

The Software Downgrade defines the process of removing the upgraded version of the software image download, thus reverting the Cable Home Device to the exact previous state.

When the PS element receives a code file with a signing-time that is later than the signing-time it has in its memory, the device MUST update its internal memory with the received value.

Because the PS element will not accept code files with an earlier signing-time than this internally stored value, to upgrade a device with a new code file without denying access to past code files, the signer (e.g., the Manufacturer, the cable operator, CTL) can choose not to update the signing-time. In this manner, multiple code files with the same code signing-time allows an operator to freely downgrade a device's code image to a past version (that is, until the CVC is updated). This has a number of advantages for the cable operator, but these advantages will be weighed against the possibilities of a code file replay attack.

Another approach would be to sign the previous code file with a signing-time that is equal to or greater than the signing-time of the last upgrade.

11.9 PS configuration file security in DHCP provisioning mode

11.9.1 Configuration file security infrastructure goals

The goals for securing the configuration file include:

- Provide an authenticated tunnel between the PS client device and HTTPS server to ensure that configuration files are secured from the cable operator to the PS. An integrity check is automatically included when a message is authenticated.
- Encryption of configuration files while in transport to reduce the possibility of eavesdropping on firewall and PS configuration.
- Reduce the risk of an unauthorized configuration file download to the PS by an unauthorized source.

11.9.2 Configuration file security system design guidelines

See Table 11-25.

Table 11-25/J.192 – Security system design guidelines

Reference	Guidelines
SEC14	The cable operator will have the ability to authenticate and optionally encrypt the transport of configuration files for the PS or firewall.

11.9.3 Configuration file security system description

In DHCP provisioning mode, the cable operator can choose to turn on security for the configuration file download. Within this clause, the term configuration file refers to the PS configuration file, or the firewall configuration file. Security is provided by establishing a TLS session between the PS and the HTTPS server. IPCable2Home requires the PS to understand this security option and to use TLS within the provisioning sequence to provide a secure session between the HTTPS Server and the PS, for the purposes of downloading the PS configuration file, and the firewall configuration file, in a secure manner. TLS provides authentication and encryption for the session, as configured by the cable operator. The session is torn down prior to sending the Syslog and/or NMS notification provisioning completed message. The configuration file download trigger, management, and contents remain as industry standards when TLS is layered under the HTTPS protocol. IPCable2Home specifies the requirements for an [RFC 2246] complaint TLS session. The TLS options are tightened to create a minimum set of interoperable behaviour for the PS. The provisioning flow with HTTP/TLS is described in detail in clause 13.

TLS provides an encrypted and authenticated transport tunnel for any application above TLS in the ISO stack. The HTTP protocol itself is not affected by the TLS layering. The italicized and underlined layers in the stack are encrypted for a standard TLS data packet. The HTTP protocol, which normally sits on TCP, sits directly on TLS. See Table 11-26.

Table 11-26/J.192 – TLS encryption

Configuration file data (Payload)
HTTP
TLS
TCP
IP
MAC
PHY

11.9.4 Configuration file security requirements

The PS MUST implement the Transport Layer Security (TLS) protocol as defined by [RFC 2246], TLS Protocol Version 1.0, with the exceptions as listed in this Recommendation. The exceptions within this Recommendation are intended to simplify the requirements necessary for implementation and testing purposes. Some of the exceptions place a minimum set requirements that already align with other technology used within the cable industry. The requirements placed will ensure the PS shall provide a consistent level of performance for the cable operators. This clause also helps remove any ambiguity and define processes which are not defined in the RFCs, but is needed for IPCable2Home purposes. This is especially true in the case of failure handling.

NOTE – The compression algorithm feature of TLS will not be used.

TLS version 1.0 (SSL3, TLSv1) MUST be supported. Earlier versions of TLS MUST NOT be supported by the PS. The PS MUST reject messages from the server if it attempts to use previous TLS versions.

11.9.4.1 Triggering TLS

To trigger a secure configuration file download in DHCP provisioning mode, the DHCP Ack will contain the IP address of the HTTPS server in the siaddr field. The DHCP Ack will also contain option 72 with the IP address of the HTTPS server. If the IP address in the siaddr field and the first IP address in option 72 match, the PS MUST establish a TLS session with the HTTPS server at the IP address listed in the ack, prior to requesting the configuration file. The PS MUST download the configuration file using HTTP/TLS, if the first IP address in TLV option 72 matches that IP address in siaddr, of the DHCP Ack message. If the PS does not receive a match in the DHCP ack, the PS MUST NOT initiate a TLS session, the requirements in this clause are not applicable, and the PS client MUST use DHCP provisioning mode with the specified TFTP download process. The provisioning flow diagram and description table are specified in clause 13. If option 66 is included as well as option 72, and the IP address in option 72 matches the IP address in the siaddr field, the PS MUST initiate a TLS session to the HTTPS server and MUST NOT initiate download from the TFTP server listed in option 66.

If the PS receives, the necessary information to initiate firewall configuration file, download via HTTPS, as specified in clause 7, then the PS will need to determine if it needs to continue or set up a TLS session with an HTTPS server.

11.9.4.2 TLS session prerequisites

Prior to establishing a TLS session, the PS client MUST synchronize its clock with the TOD server. Details are specified in clause 13.

Additionally, the PS client MUST establish the TCP/IP connection to the HTTPS server prior to sending the TLS ClientHello. Once the configuration file download is complete, the PS MUST close the TCP/IP connection. The PS client MUST use TCP port #443, specified by IANA standards, to connect to the HTTP/TLS server. In case the PS is not able to successfully establish a

TCP/IP connection, it MUST log event 68002000 and then restart provisioning following the retry procedure as defined in 7.4.4.2.4, Post-trigger Operation, to handle retries.

11.9.4.3 TLS messages

Unless otherwise noted, all the messages are [RFC 2246] compliant.

11.9.4.3.1 ClientHello

The PS client MUST send a ClientHello to the HTTP/TLS server to initiate the TLS Handshake sequence. After the initial ClientHello message has been sent to the HTTP/TLS server, if the TLS session is not established after 5 attempts, with 30 seconds allowed for each attempt, the PS MUST fail the session and send event 68002100.

11.9.4.3.2 PS processing of the server messages

The PS MUST be able to process the server messages, as defined in [RFC 2246], with the following exceptions:

- HelloRequest: The PS MUST ignore HelloRequest messages from a server. This protects the PS from answering rogue requests from HTTPS servers. The HTTP/TLS process can only be initiated if the appropriate DHCP options are configured by the cable operator. This assumes DHCP is trusted, even though it is not secured by IPCable2Home.
- ServerCertificate: The HTTPS server is expected to send its device certificate to the PS within the ServerCertificate message. In addition to the [RFC 2246] requirements for this message, the PS client MUST validate and verify the HTTPS server certificate. If the HTTPS server certificate authentication fails, the TLS session is considered a failure and the PS MUST send event 68002200 with the [RFC 2246] defined error code.

11.9.4.3.3 ClientCertificate

When requested by the HTTPS server, the PS MUST send its PS Element certificate and the issuing manufacturer CA certificate within the Client Certificate message. It is expected the HTTPS server will validate and verify the PS client certificates prior to proceeding with the handshake. If the PS certificates are not successfully authenticated by the server, the PS MUST treat the received alert message as a fatal alert and send event 68002200 with the appropriate error code from [RFC 2246] and then restart provisioning following the retry procedure as defined in 7.4.4.2.4, Post-trigger Operation.

11.9.4.4 TLS ciphersuites and compression

Within the ClientHello message, the requested ciphersuite MUST be listed. The required ciphersuite support is a subset of [RFC 2246] to align with the technology already used within the cable industry. The cable operator will need to select the appropriate encryption and authentication algorithm on the HTTPS server to communicate to the PS that meets the security model for that operator. The ciphersuites required in this Recommendation are a subset of those available and the PS can support additional ciphersuites.

The following cryptographic algorithms MUST be supported by the PS.

- TLS_RSA_WITH_NULL_MD5
- TLS_RSA_WITH_NULL_SHA
- TLS_RSA_WITH_DES_CBC_SHA
- TLS_RSA_WITH_3DES_EDE_CBC_SHA

The compression feature of the TLS protocol is not required. Therefore, the PS client MUST use compressionMethod.null, as the compression type.

11.9.4.5 TLS session tear-down

If the PS is required to download a separate firewall configuration file immediately after the PS configuration file is downloaded, and the firewall configuration file will be downloaded from the same HTTPS server as the PS configuration file was downloaded from, the TLS session is expected to remain active. The PS MUST ensure the TLS and corresponding TCP/IP session is closed with each HTTPS server after:

- The PS configuration file is downloaded, if, and only if, there is no firewall configuration file to be downloaded from the same HTTPS server, immediately after the PS configuration file is processed;
- The firewall configuration file is downloaded and processed.

11.9.4.6 TLS Events

[RFC 2246] defines an alert protocol to handle closure and errors for TLS. The TLS alerts and errors MUST be supported and used as defined in [RFC 2246], except the decompression_failure (30) alert will not be used, since compression is not supported. All TLS alerts MUST be recorded by the PS using event 68002200 with the appropriate error code defined in [RFC 2246] inserted in the Event Text <P1> field. The certificate related errors MUST be treated as critical, since the PS relies on server authentication.

If the PS client has not received a message from the HTTP/TLS server in response to any TLS message sent after 5 attempts, with 30 seconds allowed for each attempt, the TLS connection is considered a failure, and the PS MUST send event 68002100.

11.9.4.7 HTTP download and events

The HTTP transfer MUST only be initiated after the TLS handshake has been completed. The PS MUST communicate to the HTTP/TLS server using standard HTTP, as defined by [RFC 2616]. The PS client MUST initiate an HTTP version 1.1 request to the server for the PS configuration file, or the firewall configuration file. The PS configuration filename used in the HTTP "GET Request" MUST be the same filename the PS received in the DHCP ack. The firewall configuration filename used in the HTTP "GET Request" MUST be the same filename the PS received in the PS configuration filename, or via SNMP set.

The PS client MUST handle all status messages according to [RFC 2616]. If the PS client receives an HTTP status message indicating that the HTTP download cannot be completed, the PS MUST fail the session, send event 68003000 with the appropriate error code from [RFC 2616], and then restart provisioning following the retry procedure as defined in 7.4.4.2.4, Post-trigger Operation, to handle retries.

NOTE – Once the configuration file is downloaded successfully, the PS MUST send event 68003200.

11.10 Physical security

The PS is required to maintain, in its non-volatile memory, keys and other crypto-variables related to network security. The PS MUST deter unauthorized physical access to this cryptographic material.

The level of physical protection of keying material required for the PS is specified in terms of the security levels defined in the FIPS PUBS 140-2, Security Requirements for Cryptographic Modules. In particular, the PS MUST meet FIPS PUBS 140-2 Security Level 1 requirements.

FIPS PUBS 140-2 Security Level 1 requires minimal physical protection through the use of production-grade enclosures and recommended software practices.

11.11 Cryptographic algorithms

11.11.1 SHA-1

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

12 Management processes

12.1 Introduction/Overview

This clause provides examples of processes associated with the use of the tools described in clause 6 (Management Tools), and additional processes that facilitate other required management functions defined in this Recommendation. PS Database access and other PS operations of the IPCable2Home Management Portal (CMP) are described in clause 6. Typical MIB access rules are provided in 6.3.3.1.4.2.

Management-related and other descriptive processes are provided for the following scenarios:

- Management Tool Processes:
  - CTP Operation:
    - Connection Speed Tool;
    - Ping Tool.
- PS Operation:
  - PS Database Access;
  - Reconfiguration:
    - PS Software Download;
    - PS Configuration File Download.
- MIB Access:
  - VACM Configuration;
  - Management Event Messaging Configuration:
    - CMP Event Notification Operation;
    - CMP Event Throttling and Limiting Operation.

12.1.1 Goals

This clause is primarily composed of informative text, intended to aid in understanding, and does not contain requirements. The examples describe how the Management Tools are used to accomplish typical management functions. Sequence charts of additional management-related processes (i.e., those not defined in clause 6) are also provided, including management processes or process steps associated with the use of required Management Tools. All processes shown involve interaction of the PS element with Headend systems.

12.2 Management tool processes

Management Tool Processes are those associated with the required Management Tools defined in clause 6.

12.2.1 CTP operation

The IPcable2Home Test Portal (CTP) provides Connection Speed Tool and Ping Tool capabilities, described in 6.4.3.1 and 6.4.3.2, respectively.

12.2.1.1 Remote connection speed test

The Remote Connection Speed Test can be useful in validating performance levels, identifying possible configuration errors, and determining other performance-oriented characteristics:

1) The Network Management System (NMS) starts the test by initializing the test parameters and setting the Begin Test flag, via SNMP SET Request.
2) The CMP SNMP Agent updates the PS Database with the test parameters and notifies the CTP to begin the test.
3) The CTP queries the PS database for the test parameters.
4) The CTP issues a burst of UDP packets to port 7 of the specified LAN IP Device. Port 7 is reserved for the echo service.
5) The target LAN IP Device simply echoes the UDP packet payload back to the CTP.
6) Once all of the packets have been received, or the test timeout period has expired, the CTP updates the PS Database with the results of the test and sets the Test Complete flag.
7) The NMS verifies that the command is complete by checking Status = complete.
8) The NMS requests the test results via SNMP GET Request.
9) The CMP SNMP agent queries the PS database for the test results and reports them in the SNMP GET Response. If the test has not completed, the test data will indicate the test is still running. The NMS must repeat the SNMP GET Request until the test results indicate the test has completed.

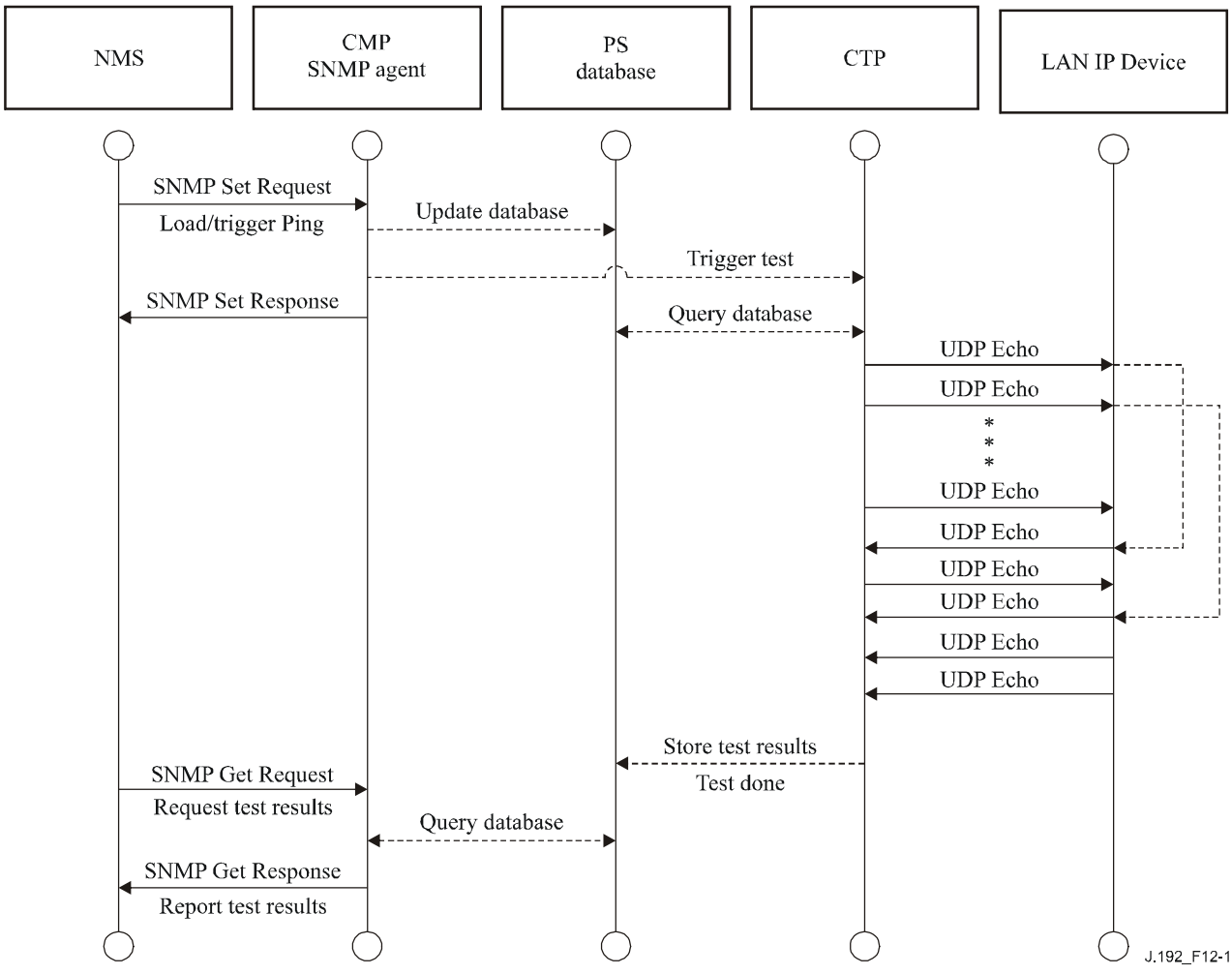


Figure 12-1/J.192 – Connection speed tool process sequence diagram

12.2.1.2 Ping tool process

The Ping Tool can be useful in validating connectivity state, performance levels, and identifying possible configuration errors.

1) The NMS starts the test by initializing the test parameters and setting the Begin Test flag, via SNMP SET Request.
2) The CMP SNMP Agent updates the PS Database with the test parameters and notifies the CTP to begin the test.
3) The CTP queries the PS database for the test parameters.
4) The CTP issues an ICMP Echo Request packet to the specified LAN IP Device.
5) The target LAN IP Device responds with an ICMP Echo Response.
6) The CTP updates the PS Database with the results of the test and sets the Test Complete flag.
7) The NMS verifies that the command is complete by checking Status = complete.
8) The NMS requests the test results via SNMP GET Request.
9) The CMP SNMP agent queries the PS database for the test results and reports them in the SNMP GET Response. If the test has not completed, the test data will indicate the test is still running. The NMS must repeat the SNMP GET Request until the test results indicate the test has completed.

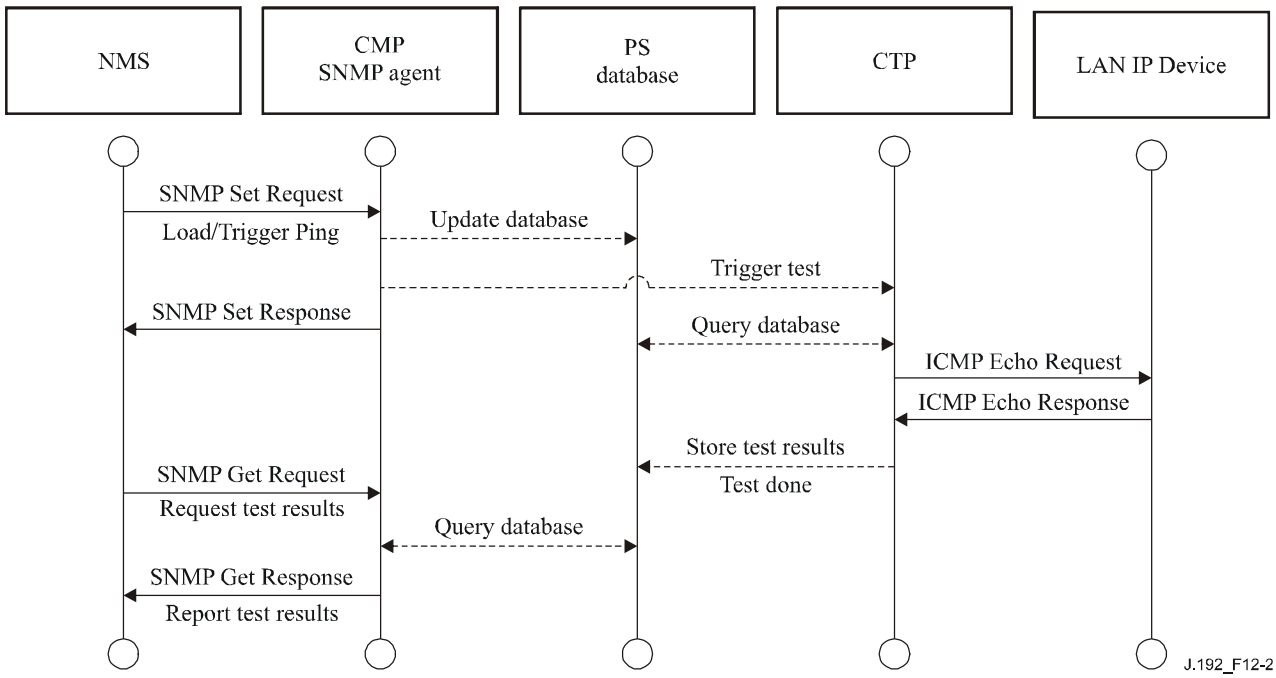


Figure 12-2/J.192 – Ping tool process sequence diagram

12.3 PS operation

The IPCable2Home Management Portal (CMP) provides access to the PS Database via the PS WAN-Man interface, as described in clause 6. The message sequence for a typical PS Database access operation from the PS WAN-Man interface is described below.

12.3.1 PS database access

Configuration and management parameters stored in the PS Database are accessed by the NMS via SNMP MIBs. Parameters are retrieved using SNMP GetRequest, GetNextRequest, and GetBulk messages issued by the NMS with the PS WAN-Man address as the destination address. Parameters can be modified and actions (such as the Connection Speed and Ping tools), executed by the NMS issuing SNMP Set-Request messages with the appropriate parameters, to the PS WAN-Man address.

Figure 12-3 describes the management message sequences for a typical PS Database access from the PS WAN-Man interface. The following message sequences assumes that a secure SNMPv3 link has been established:

1) The NMS reads data from the PS database using the SNMP "GET Request". The request lists the specific objects the NMS wants from the database.
2) The CMP SNMP Agent queries the PS Database for the specified parameters.
3) The CMP SNMP reports the data to the NMS with the SNMP "GET Response".

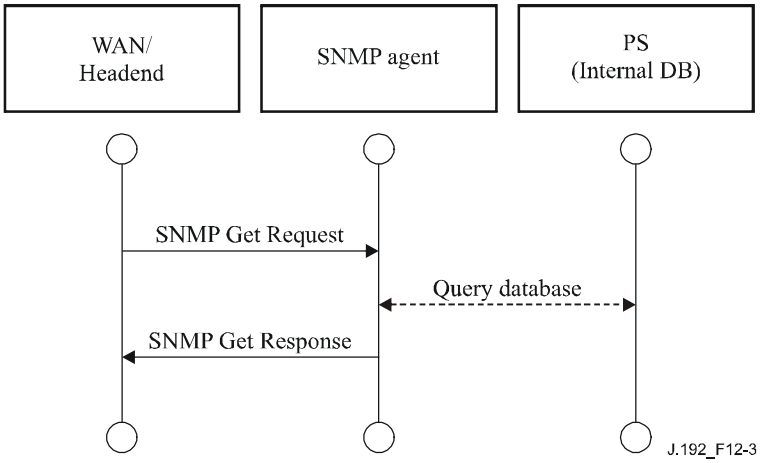


Figure 12-3/J.192 – PS database access from the PS WAN-Man interface sequence diagram

12.3.2 Reconfiguration

12.3.2.1 PS software download

Figure 12-4 illustrates a software/firmware download process for a PS in SNMP Provisioning Mode, which is triggered by the NMS. The PS is instructed where to obtain the new software code file. Once download of the code file is complete, the PS will test the image for any corruption that may have occurred during the download. Authentication is performed to verify that the code file can be trusted. Following this step, a system reboot is performed.

Following the reboot, the PS resumes operation on the new software image. The PS may need to be reconfigured after the software upgrade, and the WAN interfaces may need to be provisioned again (not shown). If the PS does not accept the new software image, it will revert back to the prior (backup) software version and report the results to the NMS.

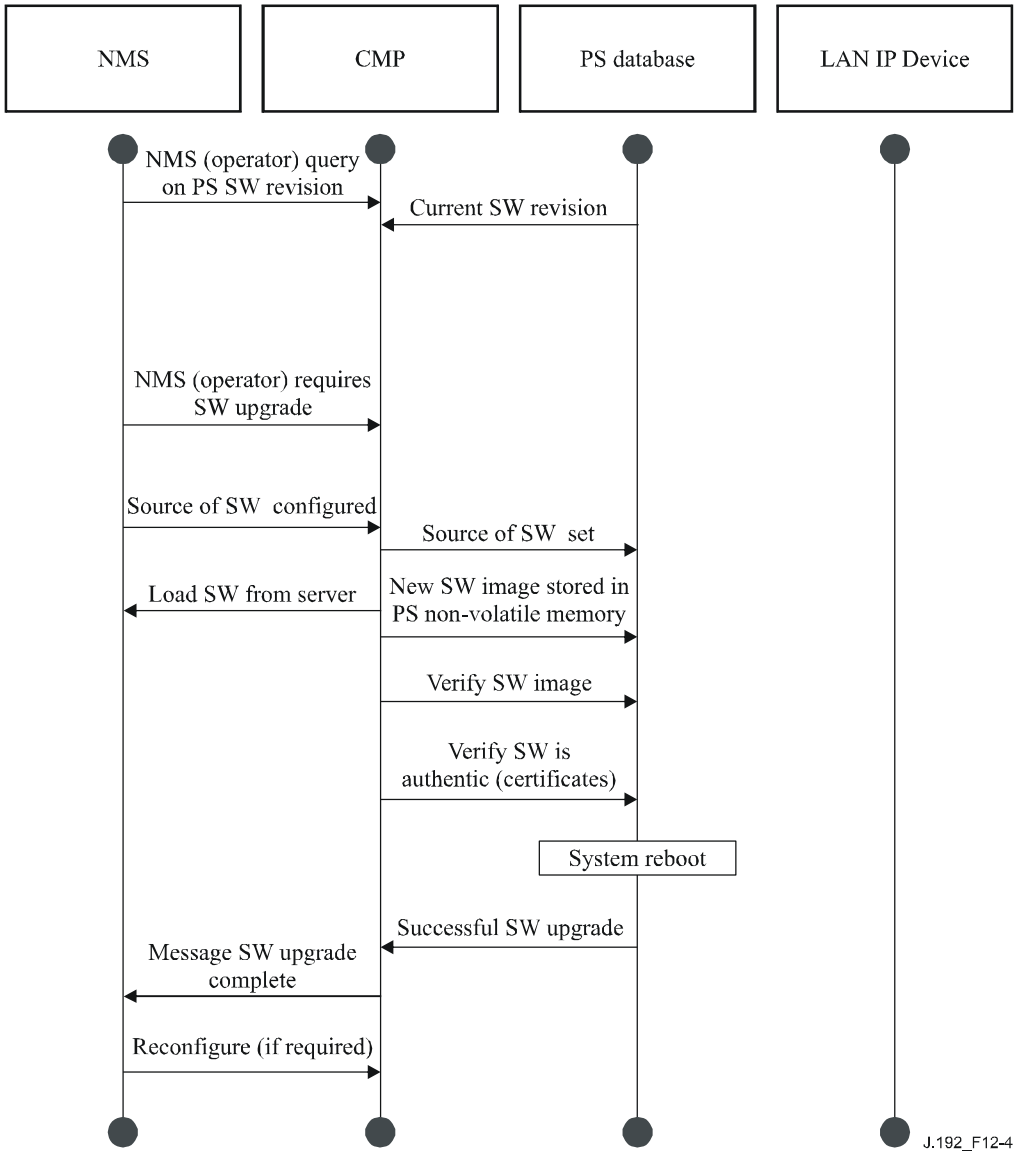


Figure 12-4/J.192 – PS software download sequence diagram

12.3.2.2 PS configuration file download

Figure 12-5 illustrates a reconfiguration of a PS in SNMP Provisioning Mode, via config file download, and is triggered by the NMS. The configuration file is given to the PS by writing the fileserver and filename into the PS, and triggering the PS to download the file. Once the configuration file is loaded, the commands within it are interpreted. If any of the commands are not understood, or are not applicable, they are skipped and an event is generated. When the PS has completed processing the config file, it will record the number of TLV tuples processed and skipped in the appropriate MIB objects.

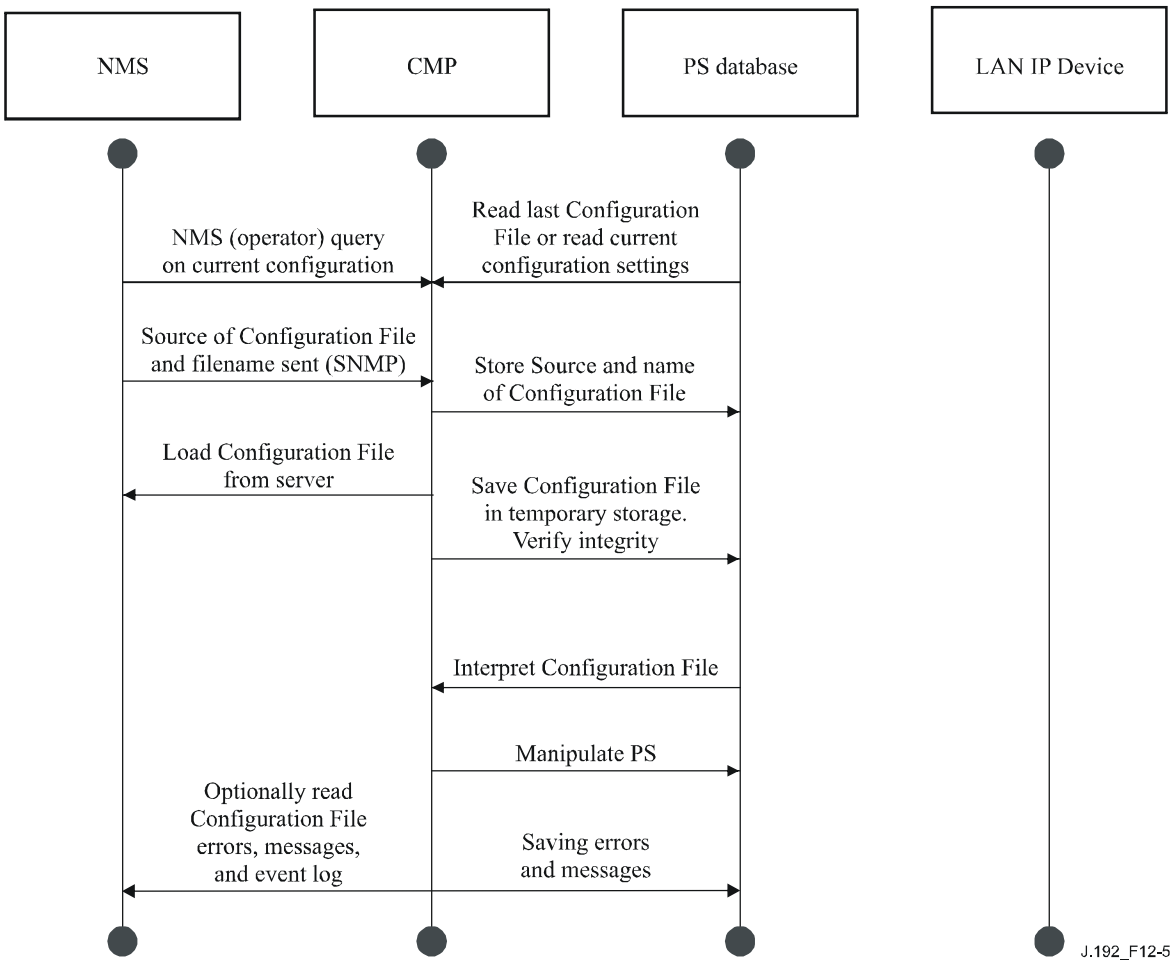


Figure 12-5/J.192 – PS reconfiguration (configuration file download) sequence diagram

12.4 MIB access

12.4.1 VACM configuration

IPCable2Home specifies operator control of the IPCable2Home management domain. An example of the configuration of VACM parameters is shown in Figure 12-6.

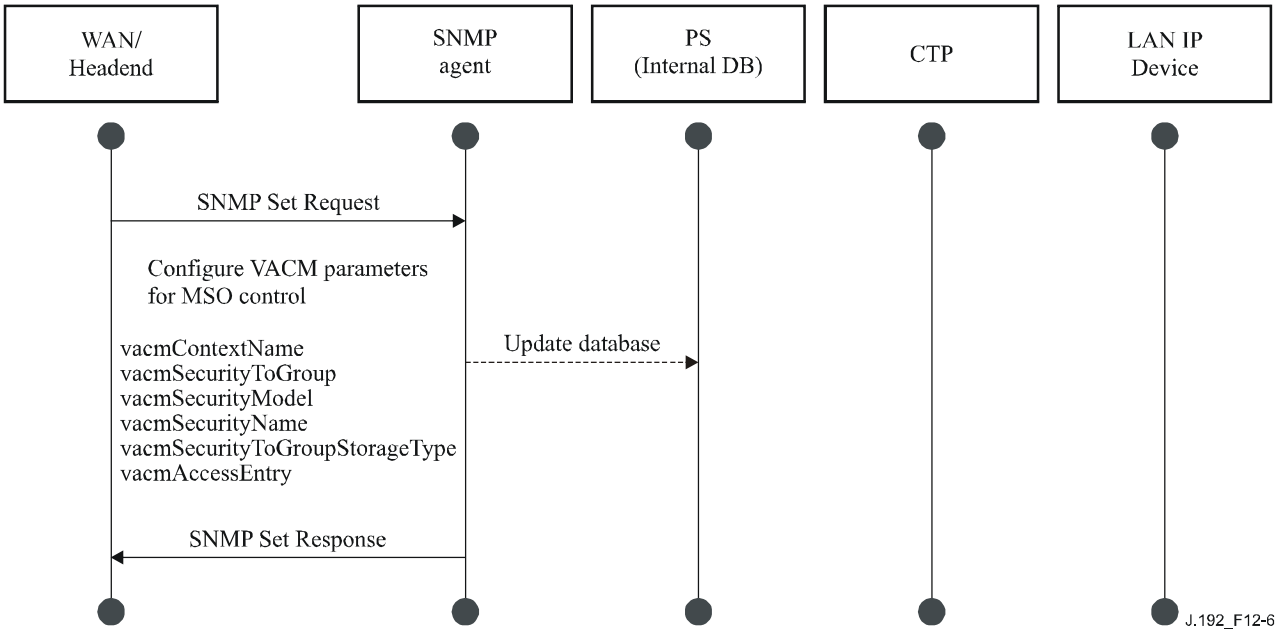


Figure 12-6/J.192 – PS configuration (VACM parameters) sequence

12.4.2 Management event messaging configuration

12.4.2.1 CMP event notification operation

IPCable2Home events are reported through local event logging, SNMP TRAP, SNMP INFORM messages, and SYSLOG. The event notification mechanism can be set or modified by the NMS, by issuing an SNMP Set-Request message to the PS WAN-Man address.

Figure 12-7 illustrates configuring the PS database to store events in local log files. Local log events are of two types: local non-volatile and local volatile. The NMS will read the content of the local log and write that content to the Headend event logging system. A PS reboot causes only the volatile events to be cleared from the PS database. Non-volatile events persist across reboots.

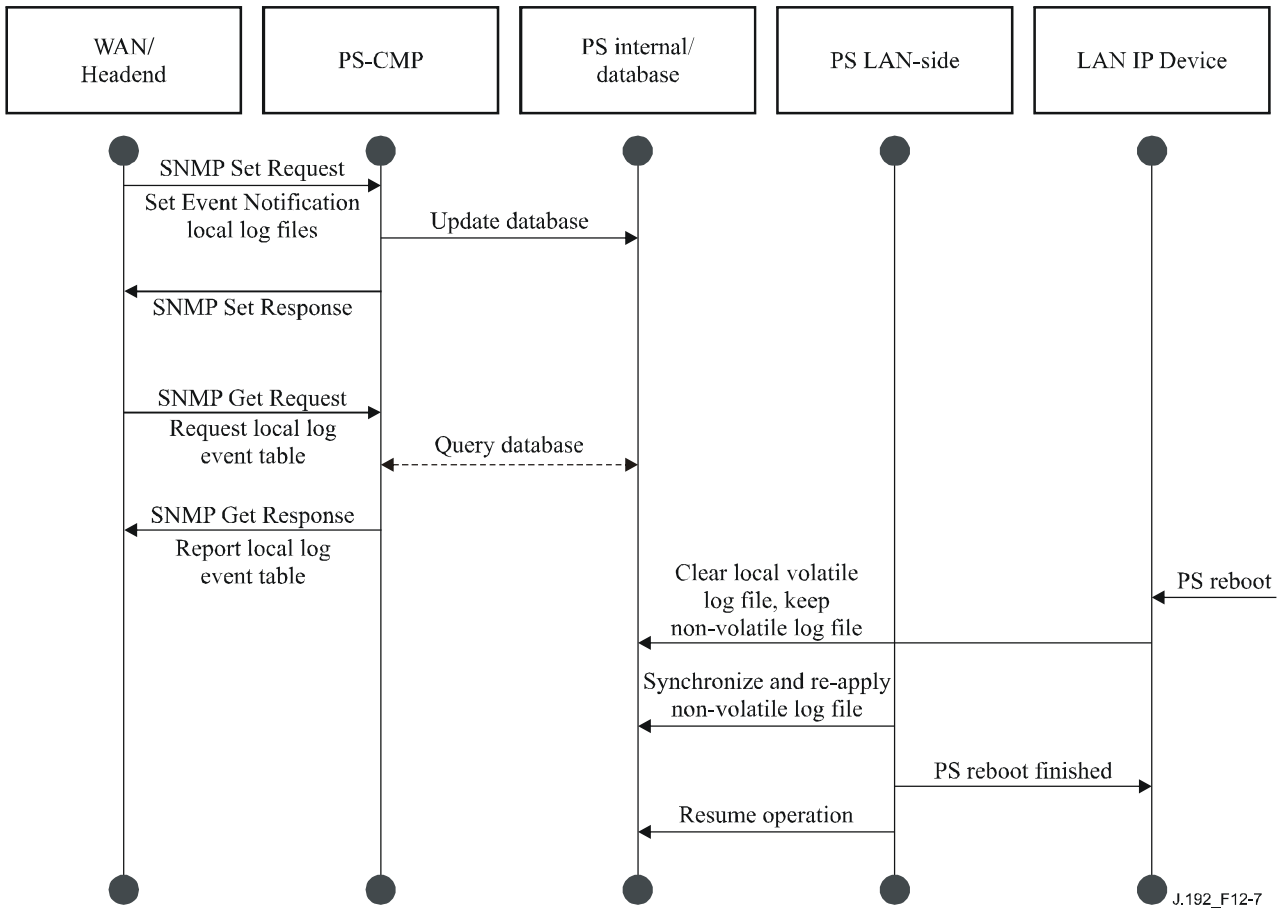


Figure 12-7/J.192 – PS configuration (event control) sequence

Figure 12-8 illustrates the download of a configuration file for a PS in SNMP Provisioning Mode. This process is triggered via an SNMP Set Request. The PS must verify this file before accepting it. In the example, a TLV error exists and is reported. Since the event notification is set to the SNMP TRAP mode, the address of the TRAP server is retrieved from the PS database and the event is sent to that TRAP server.

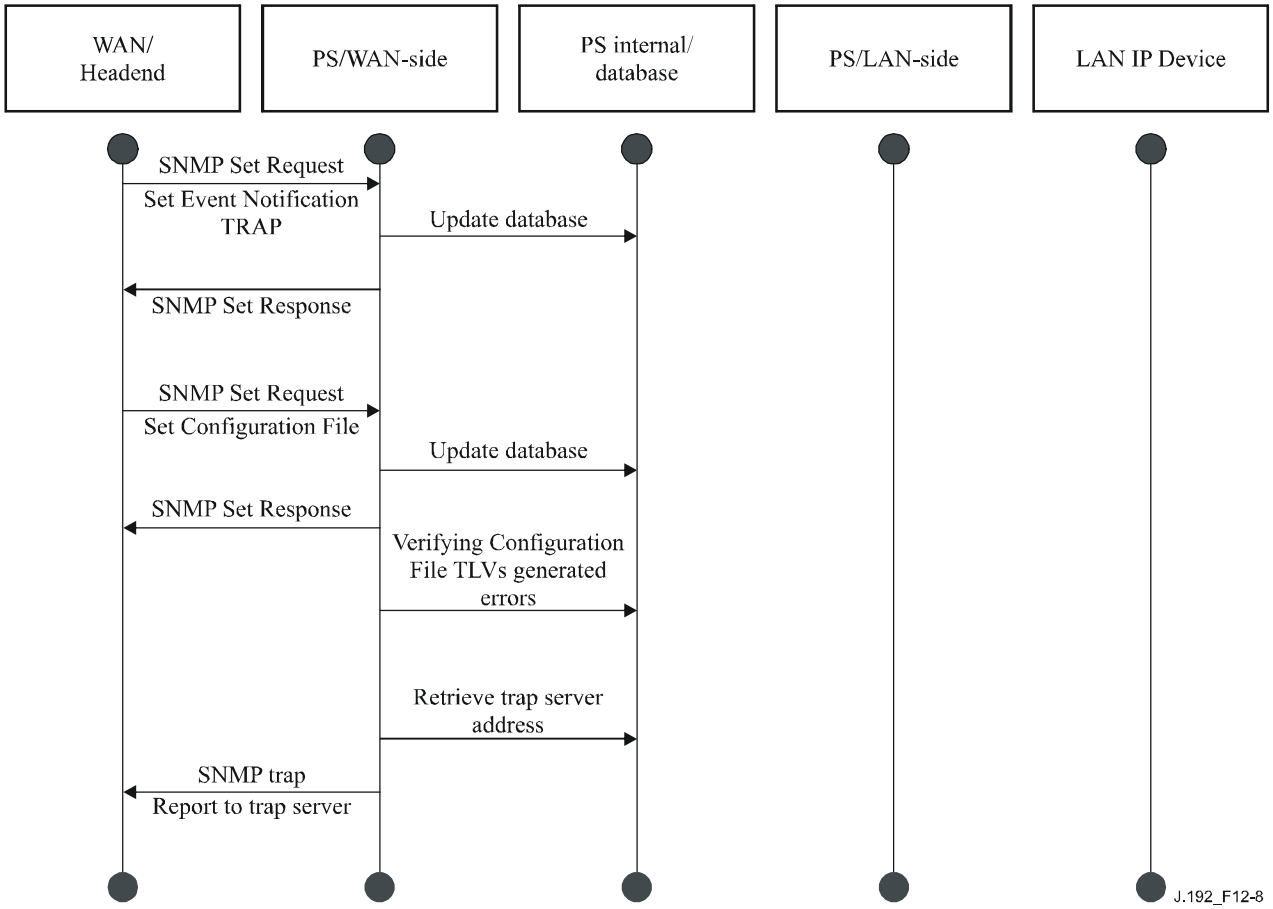


Figure 12-8/J.192 – PS configuration file download (with invalid TLVs) sequence

Figure 12-9 illustrates the process of a LAN IP Device trying to obtain an IP address from the local DHCP server (CDS). The CDS function checks the PS database for an available IP address. In this case, the CDS detects that no IP address is available from the address pool, and generates an event to SYSLOG.

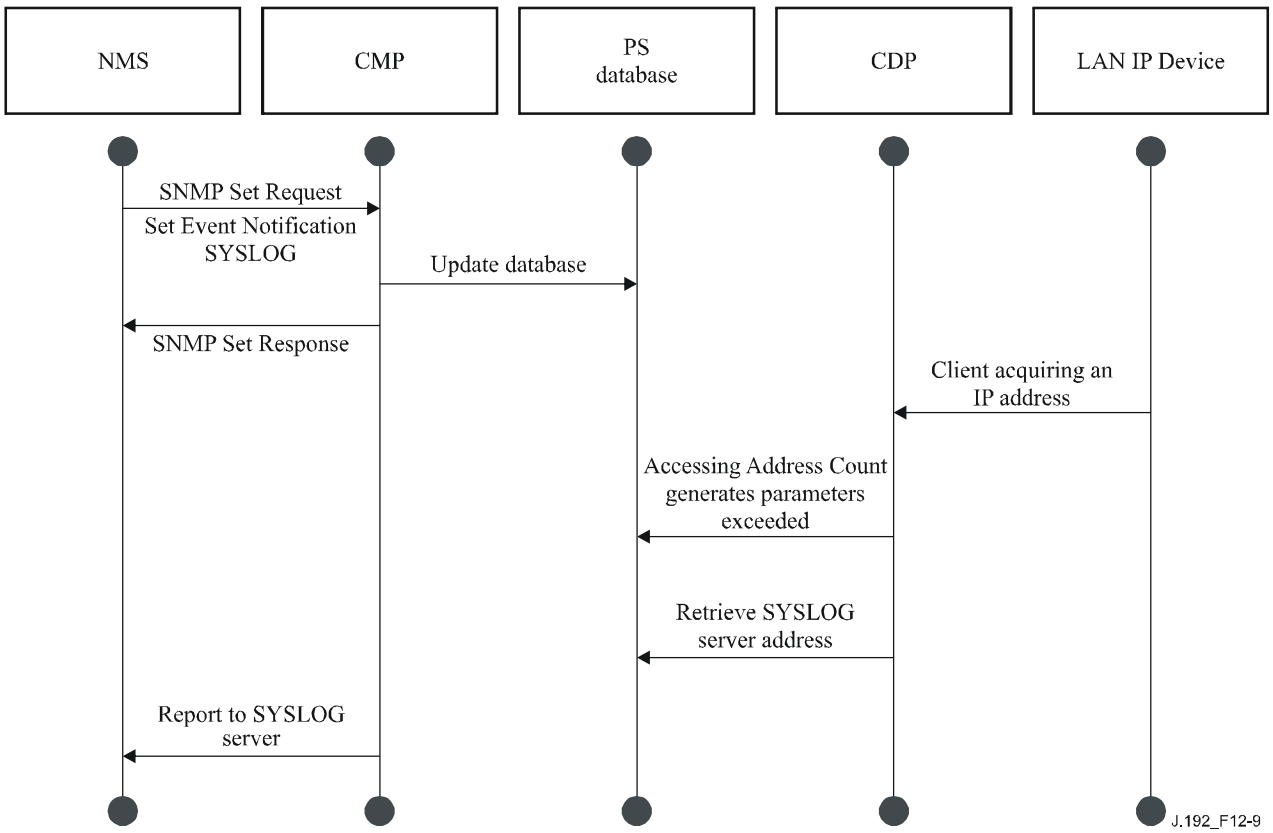


Figure 12-9/J.192 – Address acquisition (request exceeds provisioned count) sequence

12.4.2.2 Example CMP event throttling and limiting operation

This Recommendation provides an event throttling mechanism via the CMP functionality of the PS. Event throttling and limiting is very flexible and can include cases in which all events are reported, and cases in which no events are reported to the NMS. Refer to 6.3.3.2.4.8 for a description of the CMP Event Throttling and Limiting mechanism.

Figure 12-10 illustrates configuring the PS database to return events via the SNMP INFORM method. Initially, several INFORM messages are written to the local log file and delivered to the NMS. The event throttling mechanism sets the limit of the number of events that can be sent to the NMS within a given time-frame. When that limit is reached, the PS will stop sending INFORM messages to the NMS. In order to restart the event notification, the NMS SHOULD re-enable the event reporting.

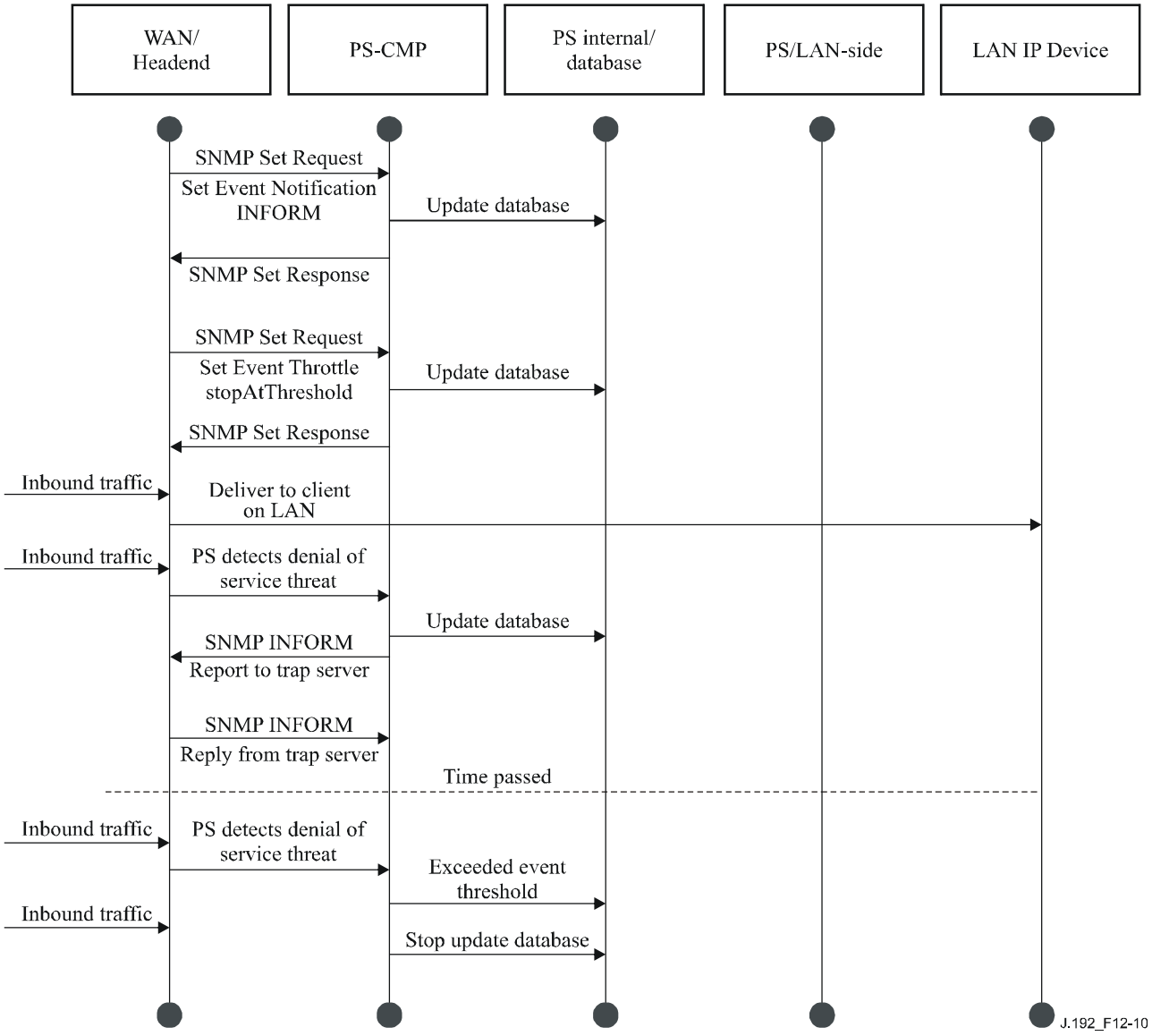


Figure 12-10/J.192 – CMP event throttling and limiting operation

13 Provisioning processes

This clause describes the processes involved when using the Provisioning Tools, described in clause 7, for initial provisioning of LAN IP Device; the PS element Provisioning has the following three tasks:

1) Acquiring network addresses.

2) Acquiring server information.

3) Secure download and processing of the PS Configuration File.

Provisioning processes are described in this clause for each of the following relevant cases:

- PS WAN-Man – Provisioning of the PS WAN based management functionality.
- PS WAN-Data – Provisioning of PS WAN-Data IP addresses to be used for creating CAT Mappings to LAN IP Devices in the LAN-Trans address realm.
- LAN IP Device in the LAN-Trans Realm – Provisioning of a LAN IP Device with a translated IP address.
- LAN IP Device in the LAN-Pass Realm – Provisioning of a LAN IP Device with an IP address that is passed through to the WAN.

Provisioning of the cable modem element of an embedded PS is separate and distinct from IPCable2Home provisioning, and is out of scope for this Recommendation. The reader is referred to CableModem specifications for descriptions of cable modem provisioning.

The functional elements with which the Portal Services element interacts during the provisioning processes listed above are identified in Figure 13-1. The Key Distribution Centre (KDC) functional element is shown with a broken outline, since it is used in SNMP Provisioning Mode, but not in DHCP Provisioning Mode. The other functional elements are used in both provisioning modes.

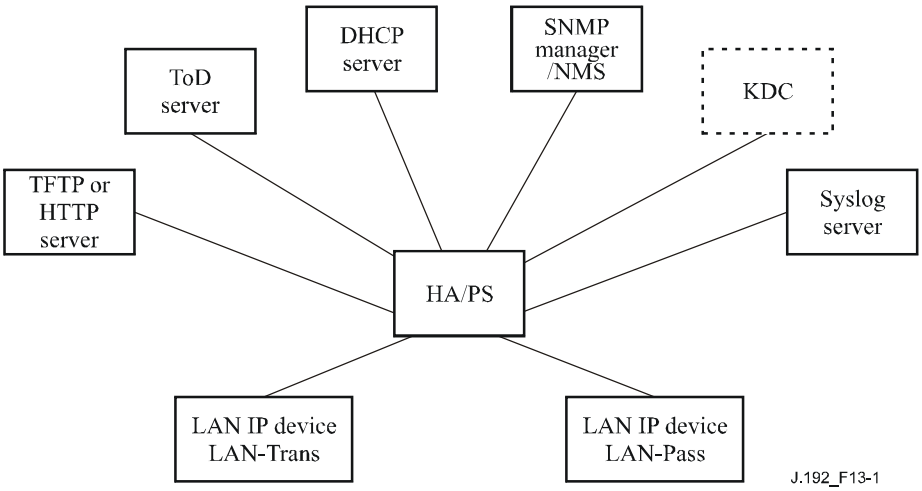


Figure 13-1/J.192 – IPCable2Home provisioning functional elements

The Trivial File Transfer Protocol (TFTP) server or the HyperText Transfer Protocol (HTTP) server provides access to the PS Configuration File for the PS and follows rules described in [RFC 1350]. The Time of Day (ToD) server provides the means for the PS to acquire the current time in UTC format as described in [RFC 868]. The Dynamic Host Configuration Protocol (DHCP) server provides the PS with private and/or global IP addresses following [RFC 2131], as well as providing other information via DHCP options in accordance with [RFC 2132]. The Network Management System (NMS) complies with the Simple Network Management Protocol (SNMP) versions SNMPv1, SNMPv2, and SNMPv3 as described in [RFC 3584]. The System Log (SYSLOG) server handles event messages generated by the PS and by LAN IP Devices in the home. The PS implements clients for these cable data network-based servers, and uses these client functions during the provisioning processes described in this clause to accomplish the tasks listed at the beginning of this clause.

13.1 Provisioning modes

Clauses 5.5 and 7.2.1 introduce two valid provisioning modes supported by the Portal Services element: DHCP Provisioning Mode and SNMP Provisioning Mode. The PS operates in a third mode, Dormant CableHome Mode, if it is not configured to operate in either of the two valid provisioning modes. In this clause the two valid provisioning modes are presented in more detail. Figure 13-2 illustrates a possible event flow for the two provisioning modes and the Dormant CableHome Mode. The key point of Figure 13-2 is the switch used by the PS to determine the mode in which it is to operate.

The PS operates in DHCP Provisioning Mode (DHCP Mode) if the DHCP server in the cable network provides a valid IP address for the TFTP or HTTP server in the DHCP message 'siaddr' field, provides a valid file name for the PS Configuration File in the DHCP message 'file' field, and does NOT provide DHCP option 177 sub-options 3, 6 and 51 to the PS CDC, during the DHCPACK phase of the initialization process. DHCP Provisioning Mode is intended to enable the PS to operate on a DOCSIS 1.0 or a DOCSIS 1.1 infrastructure, 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-options 3, 6 and 51. SNMP Provisioning Mode is intended to enable the PS to take advantage of advanced features of a PacketCable infrastructure.

The PS defaults to Dormant CableHome Mode if it receives none of the fields or sub-options defined as triggers for DHCP Provisioning Mode and for SNMP Provisioning Mode, or if it receives an invalid combination of the fields and sub-options. An embedded PS integrated as an eSAFE with an eDOCSIS [eDOCSIS] compliant embedded cable modem can also be configured through the cable modem's esafePsCableHomeModeControl MIB object to operate in Dormant CableHome Mode. See 7.3.3.2.4.

Not all error conditions are shown in Figures 13-2 and 13-3. Refer to 7.2.2 for a description of PS behaviour in the event of incorrect Provisioning Mode decision criteria.

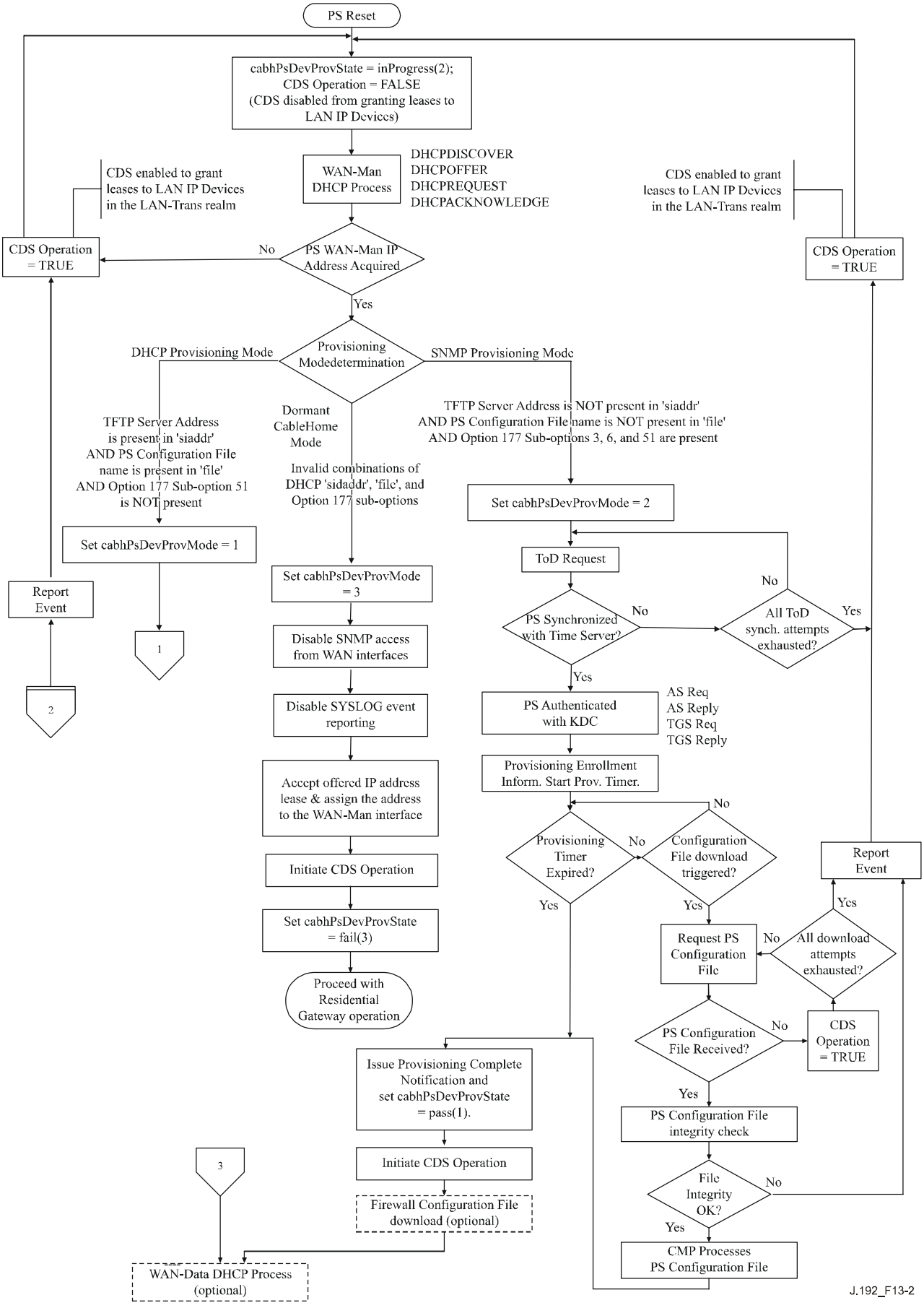


Figure 13-2/J.192 – IPCable2Home provisioning modes (Part 1)

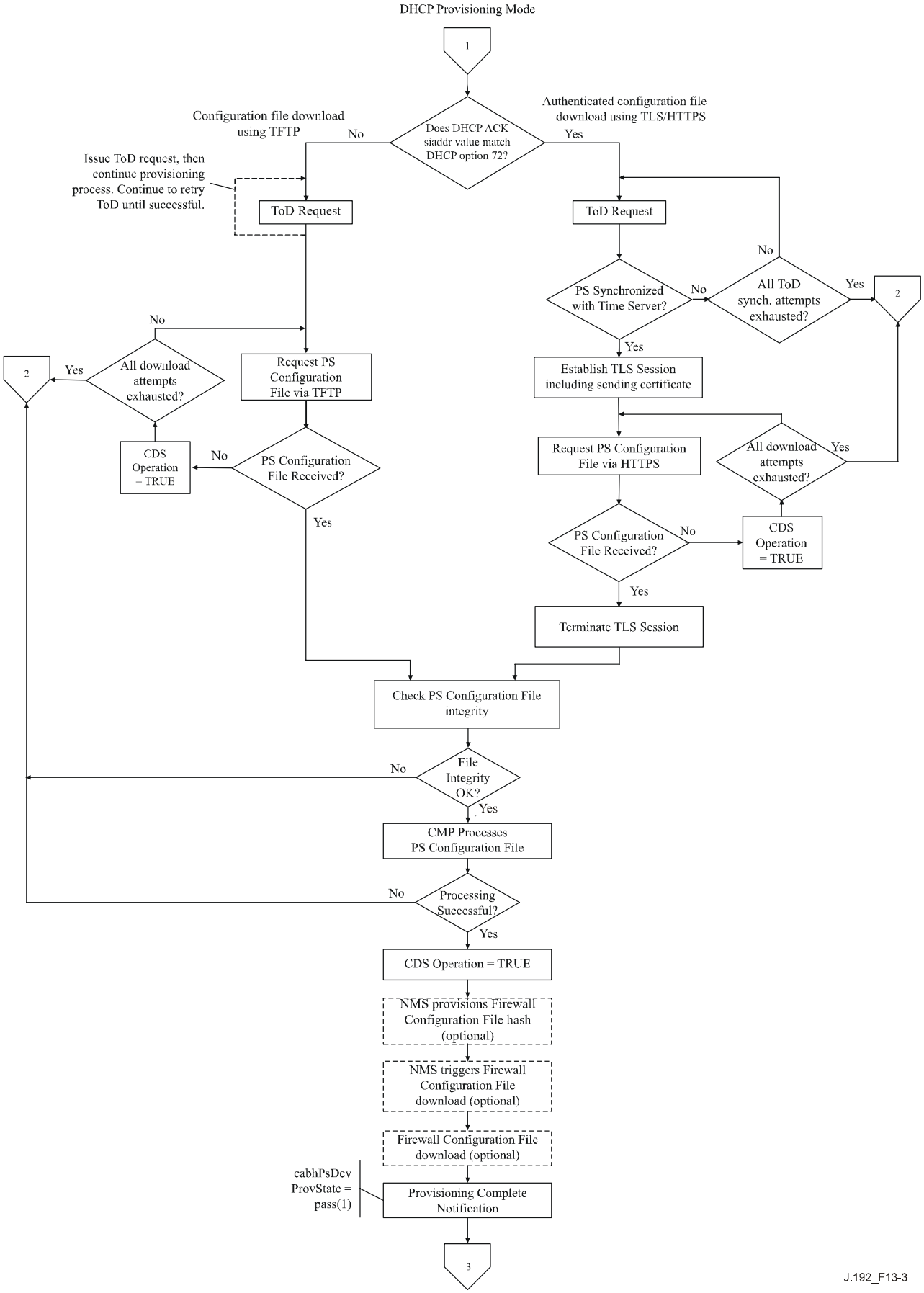


Figure 13-3/J.192 – IPCable2Home provisioning modes (Part 2)

13.2 Process for provisioning the PS for management: DHCP provisioning mode

The PS requests, from the Headend provisioning system, an IP address to be used for the exchange of management messages between the NMS and the PS. The PS parses the DHCP message returned in the DHCP OFFER and makes a determination about the provisioning mode in which it is to operate (see 7.3.3.2.4). Clause 7.3.3.2.3.2 describes 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 7.3. 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.

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. These management messaging modes are described in 6.3.3.

Figure 13-4 and Table 13-1 describe the sequence of messages needed to initialize a PS operating in DHCP Provisioning Mode. The process for provisioning for management of a PS operating in DHCP Provisioning Mode is the same for the PS embedded with a DOCSIS cable modem, as it is for the standalone PS. The provisioning for the Embedded PS MUST NOT occur before the cable modem provisioning process. The standalone PS management provisioning SHOULD occur immediately after power-up/reset.

The optional process of downloading a Firewall Configuration File is shown with shading in Figure 13-4.

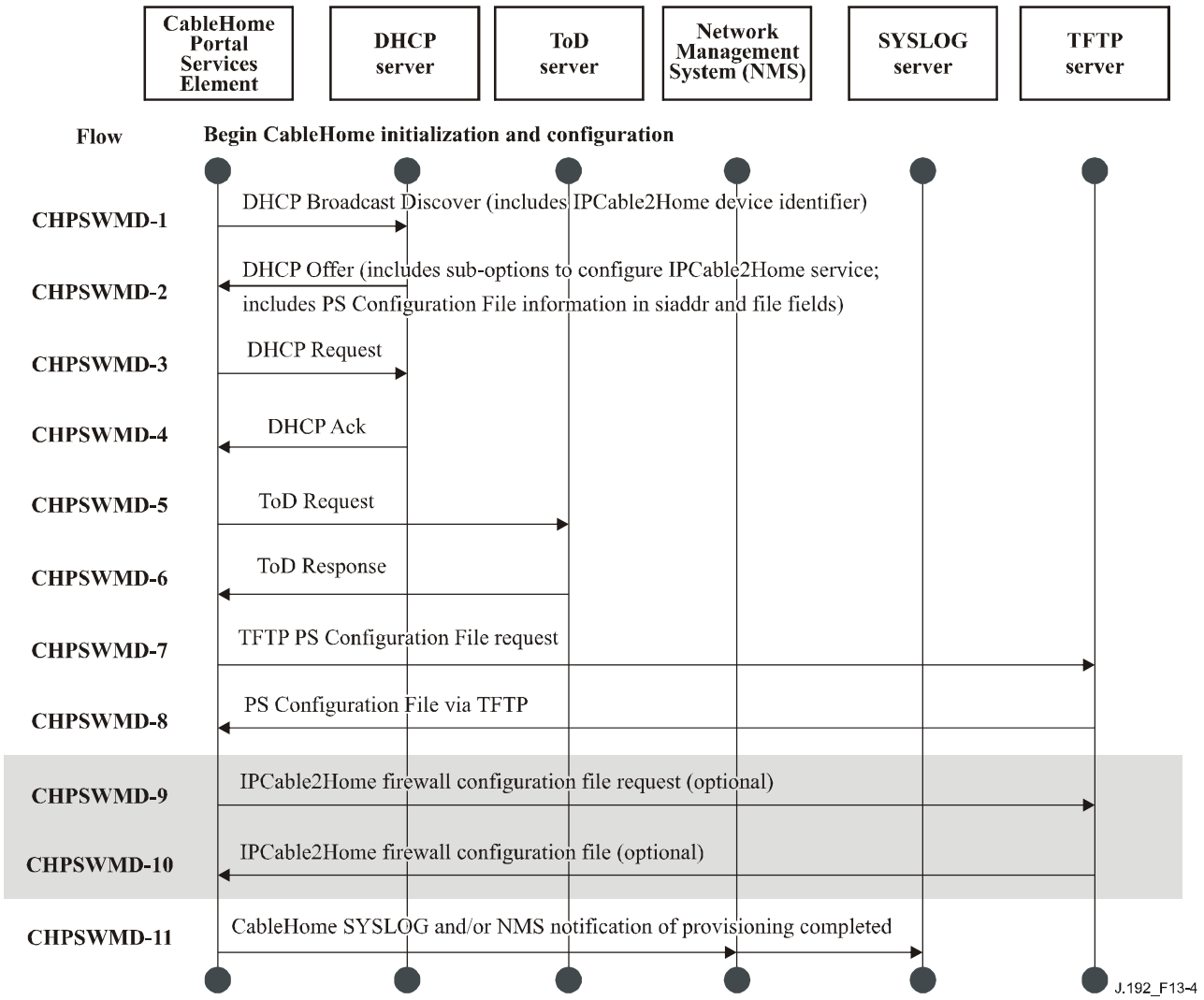


Figure 13-4/J.192 – Provisioning process for PS management – DHCP provisioning mode

Table 13-1 describes the individual messages CHPSWMD-1 – CHPSWMD-11 shown in Figure 13-4.

Table 13-1/J.192 – Flow descriptions for PS WAN-Man provisioning process for DHCP provisioning mode

Flow step	PS WAN-Man provisioning: DHCP provisioning mode	Normal sequence	Failure sequence
CHPSWMD-1	DHCP Broadcast Discover The CDP (CDC) sends a broadcast DHCP DISCOVER message to acquire the WAN-Man IP address as described in 7.3.3.2.4. The DHCP DISCOVER broadcast by the CDP (CDC) includes mandatory options listed in Table 7-10 CDC DHCP Options in DISCOVER and REQUEST Messages. The PS sets 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). If unsuccessful on the first attempt to acquire a WAN-Man IP address, the PS initiates operation of the CDS as specified in 7.3.3.2.4.
CHPSWMD-2	DHCP OFFER	CHPSWMD-2 MUST occur after CHPSWMD-1 completion.	If failure per DHCP protocol, return to CHPSWMD-1 and report an error.
CHPSWMD-3	DHCP REQUEST The CDP MUST send the appropriate DHCP server a DHCP REQUEST message to accept the DHCP OFFER.	CHPSWMD-3 MUST occur after CHPSWMD-2 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 (see 7.3.3.2.4). The PS stores the Time of Day server address in cabhPsDevTimeServerAddr. The PS modifies cabhPsDevProvMode based on information received in the DHCP ACK (see 7.3.3.2.4).	CHPSWMD-4 MUST occur after CHPSWMD-3 completion.	If failure per DHCP protocol, return to CHPSWMD-1 and report an error.
CHPSWMD-5	Time of Day (ToD) Request per [RFC 868] The PS issues a ToD Request to the Time Server identified in option 4 of the DHCP ACK message.	CHPSWMD-5 MUST occur after CHPSWMD-4 completion.	Continue with CHPSWMD-6.

Table 13-1/J.192 – Flow descriptions for PS WAN-Man provisioning process for DHCP provisioning mode

Flow step	PS WAN-Man provisioning: DHCP provisioning mode	Normal sequence	Failure sequence
CHPSWMD-6	ToD Response The ToD server is expected to reply with the current time in UTC format.	CHPSWMD-6 MUST occur after CHPSWMD-5 completion.	Attempt synchronization with the next Time of Day server listed in DHCP option 4 of the DHCP ACK. If an unsuccessful synchronization attempt has been made with each ToD server as part of an initial attempt to synchronize Time of Day, set cabhPsDevTodSyncStatus = false(2), attempt to acquire system time from the cable modem (embedded PS only), update cabhPsDevDateTime, update CDS lease times, and continue with CHPSWMD-7. Refer to 7.5.4 for additional details.
CHPSWMD-7	TFTP Request The PS operating in DHCP Provisioning Mode sends the TFTP Server a TFTP Get Request to request the specified configuration data file as described in 7.4.4.	CHPSWMD-7 MUST occur after CHPSWMD-5 completion. CHPSWMD-7 MAY occur before CHPSWMD-6 completion.	Continue to CHPSWMD-8.
CHPSWMD-8	TFTP server sends PS Configuration File After the PS Configuration File is received, the hash is checked. Refer to 7.4.4.1. The PS Configuration File is then processed. Refer to 7.4.4 for PS Configuration File contents. Optionally, the IP Address of the firewall Configuration File TFTP server, the firewall Configuration File filename and the hash of the firewall Configuration File 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.	CHPSWMD-8 MUST occur after CHPSWMD-7 completion.	If the TFTP download fails, take action, depending upon the nature of the error, as described in 7.4.4.4.

Table 13-1/J.192 – Flow descriptions for PS WAN-Man provisioning process for DHCP provisioning mode

Flow step	PS WAN-Man provisioning: DHCP provisioning mode	Normal sequence	Failure sequence
CHPSWMD-9	TFTP Request – Firewall Configuration File (optional) If the PS receives Firewall Configuration File information (Firewall TFTP server and Firewall Configuration File name) in the PS Configuration File, the PS sends the Firewall Configuration TFTP Server a TFTP Get Request to request a Firewall Configuration File (see 11.6.4.2). If the PS does not receive Firewall Configuration File information in the PS Configuration file, the PS provisioning process (DHCP Provisioning Mode) MUST skip steps CHPSWMD-9 and CHPSWMD-10 and continue with step CHPSWMD-11.	If CHPSWMD-9 occurs, it MUST occur after CHPSWMD-8 completion.	If TFTP fails, continue with PS operation but report an error and continue to retry CHPSWMD-9.
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. The file is then processed. Refer to 11.6.4.	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.
CHPSWMD-11	Provisioning Complete If requested by the provisioning system, the PS is required to inform the provisioning system of the status of PS provisioning. The provisioning system could request the PS to send a SYSLOG message or an SNMP trap, or both. If the PS successfully completes all required steps from CHPSWMD-1 through CHPSWMD-10 and the PS received a SYSLOG server address in the DHCP OFFER, the PS MUST send a provisioning complete message to the SYSLOG server with provisioning state set to PASS. If the PS successfully completes all required provisioning steps from CHPSWMD-1 through CHPSWMD-10 and the PS received valid parameters for the Notification Receiver, then the PS MUST send a provisioning complete notification (cabhPsDevInitTrap) with appropriate parameters to the Notification Receiver. The PS MUST update the value of cabhPsDevProvState with status 'pass' (1) when provisioning flow steps CHPSWMD-1 through CHPSWMD-11 complete successfully.	CHPSWMD-11 MUST occur after CHPSWMD-10 completion.	If the SNMP trap fails, the provisioning server may not know the provisioning process has completed unless it polls the cabhPsProvState object.

13.3 Process for provisioning the PS for management: DHCP provisioning mode with HTTP/TLS

The PS requests from the Headend provisioning system, an IP address to be used for the exchange of management messages between the NMS and the PS. The PS parses the DHCP message returned in the DHCP OFFER and makes a determination about the provisioning mode in which it is to operate (see 7.3.3.2.4). Clause 7.3.3.2.3.2 describes 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. If DHCP option code 72 is present in the DHCP ACK message, and if its contents match the IP address in the siaddr field, the download will occur, using HTTP over TLS, as specified in 11.9.

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

Figure 13-5 and Table 13-2 describe the sequence of messages needed to initialize a PS operating in DHCP Provisioning Mode with HTTP/TLS. The process for provisioning and management of the PS operating in DHCP Provisioning Mode is the same for the PS embedded with a DOCSIS cable modem as it is for the standalone PS. The provisioning for the Embedded PS MUST NOT occur before the cable modem provisioning process. The standalone PS management provisioning SHOULD occur immediately after power-up/reset.

The optional process of downloading a Firewall Configuration File is shown with shading in Figure 13-5.

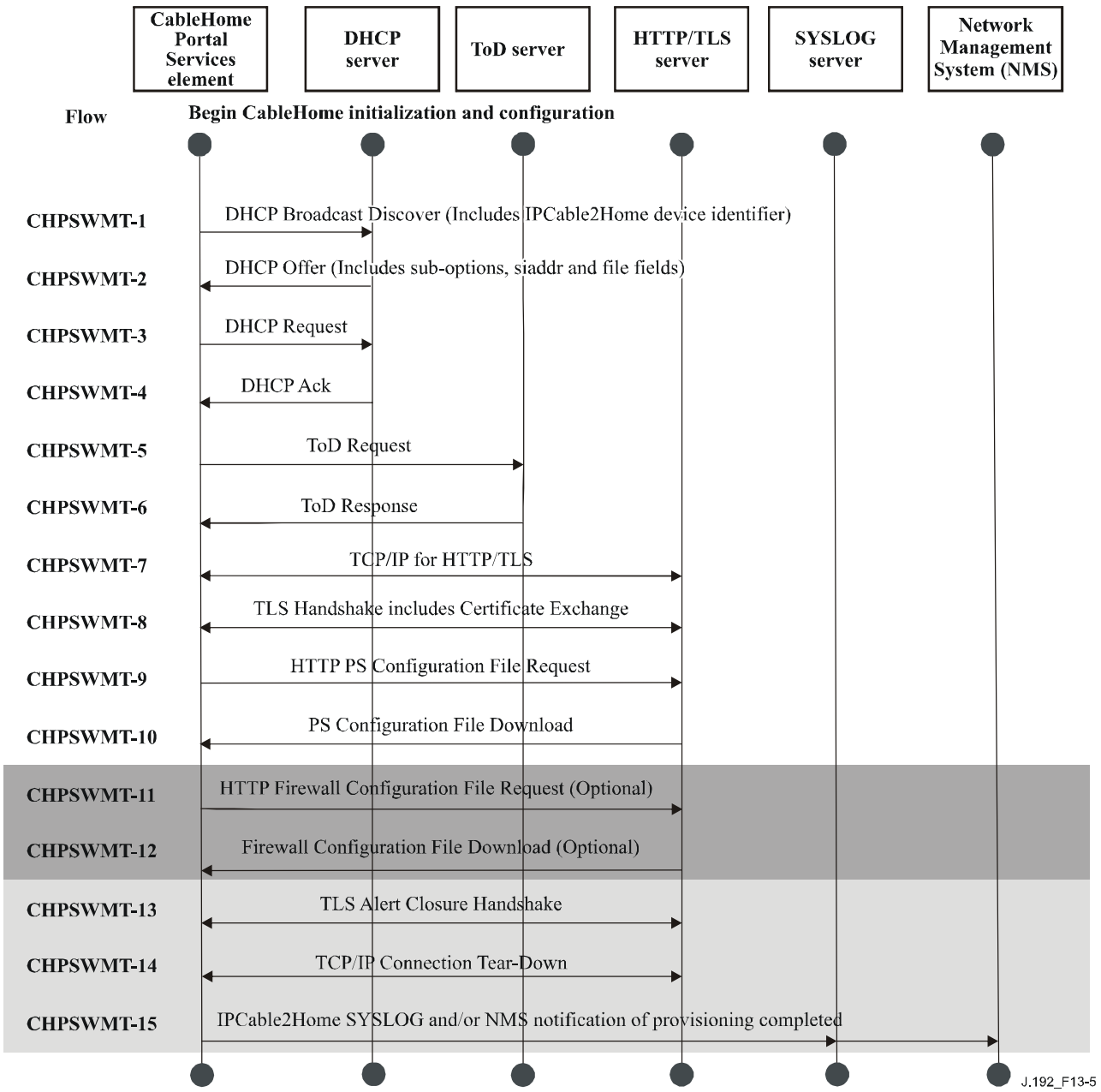


Figure 13-5/J.192 – Provisioning process DHCP provisioning mode using HTTP/TLS

Table 13-2 describes the individual messages CHPSWMT-1 – CHPSWMT-15 shown in Figure 13-5. Refer to 11.9, PS Configuration File Security in DHCP Provisioning Mode, for more information.

Table 13-2/J.192 – Flow Descriptions for DHCP provisioning mode using HTTP/TLS

Flow step	PS WAN-Man provisioning: DHCP provisioning mode	Normal sequence	Failure sequence
CHPSWMT-1	DHCP Broadcast Discover The CDP (CDC) sends a broadcast DHCP DISCOVER message to acquire the WAN-Man IP address as described in 7.3.3.2.4. The DHCP DISCOVER broadcast by the CDP (CDC) includes mandatory options listed in Table 7-10, CDC DHCP Options in DISCOVER and REQUEST messages. The PS sets 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 CHPSWMT-1). If unsuccessful on the first attempt to acquire a WAN-Man IP address, the PS initiates operation of the CDS as specified in 7.3.3.2.4.
CHPSWMT-2	DHCP OFFER	CHPSWMT-2 MUST occur after CHPSWMT-1 completion.	If failure per DHCP protocol, return to CHPSWMT-1 and report an error.
CHPSWMT-3	DHCP REQUEST The CDP sends the appropriate DHCP server a DHCP REQUEST message to accept the DHCP OFFER.	CHPSWMT-3 MUST occur after CHPSWMT-2 completion.	If failure per DHCP protocol, return to CHPSWMT-1 and report an error.
CHPSWMT-4	DHCP ACK The DHCP server sends the CDP a DHCP ACK message which contains the IPv4 address of the PS. The PS stores the Time of Day server address in cabhPsDevTimeServerAddr. If the IP address in the siaddr field of the DHCP ACK matches the first IP address in option 72, the PS initiates a TLS session and downloads the configuration file from the HTTP server. The PS modifies cabhPsDevProvMode based on information received in the DHCP ACK. Refer to 11.9, PS Configuration File Security in DHCP Provisioning Mode.	CHPSWMT-4 MUST occur after CHPSWMT-3 completion.	If failure per DHCP protocol, return to CHPSWMT-1 and report an error.
CHPSWMT-5	Time of Day (ToD) Request per [RFC 868] The PS synchronizes its time with the time server selected from DHCP option 4 (Time Server Option) in the DHCP ACK. Refer to 7.5.4, Time of Day Client Function Requirements.	CHPSWMT-5 MUST occur after CHPSWMT-4 completion.	Continue with CHPSWMT-6.

Table 13-2/J.192 – Flow Descriptions for DHCP provisioning mode using HTTP/TLS

Flow step	PS WAN-Man provisioning: DHCP provisioning mode	Normal sequence	Failure sequence
CHPSWMT-6	ToD Response The ToD server is expected to reply with the current time in UTC format.	CHPSWMT-6 MUST occur after CHPSWMT-5 completion.	Attempt synchronization with the next Time of Day server listed in DHCP option 4 of the DHCP ACK. If an unsuccessful synchronization attempt has been made with each ToD server as part of an initial attempt to synchronize Time of Day, set cabhPsDevTodSyncStatus = false(2), update cabhPsDevDateTime, update CDS lease times, and continue with CHPSWMT-7. Refer to 7.5.4 for additional details.
CHPSWMT-7	TCP/IP Setup The PS operating in DHCP Provisioning Mode establishes a TCP/IP session to exchange HTTP messages with the HTTP server in the cable operator's provisioning system.	CHPSWMT-7 MUST occur after CHPSWMT-5 completion. CHPSWMT-7 MAY occur before CHPSWMT-6 completion.	If failure per TCP/IP, retry per the specification. If retries all fail, return to CHPSWMT-1 and report an error.
CHPSWMT-8	TLS Handshake The PS operating in DHCP Provisioning Mode establishes a TLS session with the HTTPS server.	CHPSWMT-8 MUST occur after CHPSWMT-7 completion.	If failure for TLS, retry per the specification. If retries all fail, return to CHPSWMT-1 and report an error.
CHPSWMT-9	HTTP Configuration File Request The PS operating in DHCP Provisioning Mode requests the configuration file from the HTTP server.	CHPSWMT-9 MUST occur after CHPSWMT-8 completion.	If failure for HTTP, retry per the specification. If retries all fail, return to CHPSWMT-1 and report an error.

Table 13-2/J.192 – Flow Descriptions for DHCP provisioning mode using HTTP/TLS

Flow step	PS WAN-Man provisioning: DHCP provisioning mode	Normal sequence	Failure sequence
CHPSWMT-10	HTTPS server sends PS Configuration File The PS Configuration File is processed. Refer to 7.4.4 for PS Configuration File contents. Optionally, the IP Address of the firewall Configuration File HTTP server and the firewall Configuration File filename of the firewall Configuration File are included in the PS Configuration File.	CHPSWMT-10 MUST occur after CHPSWMT-9 completion.	If the HTTP download fails, report an error and return to CHPSWMT-9 (continue to retry PS Config File download). If processing of the PS Config File produces an error, continue with CHPSWMT-13 and report the error as an event.
CHPSWMT-11	HTTP Request – Firewall Configuration File (Optional) If the PS receives Firewall Configuration File information (Firewall TFTP server and Firewall Configuration File name) in the PS Configuration File, the PS requests the Firewall Configuration File from the HTTP Server. If the PS does not receive Firewall Configuration File information in the PS Configuration file, the PS provisioning process (DHCP Provisioning Mode) MUST skip steps CHPSWMT-11 and CHPSMWT-12 and continue with step CHPSWMT-13.	If CHPSWMT-11 occurs, it MUST occur after CHPSWMT-10 completion.	If HTTP fails, continue with PS operation but report an error and continue to retry CHPSWMT-13.
CHPSWMT-12	HTTP server sends firewall configuration file (Optional) If step CHPSWMD-11 occurs, the HTTP Server sends the PS a HTTP Response containing the requested firewall configuration file.	CHPSWMT-12 MUST occur after CHPSWMT-11 completion.	If the HTTP fails, continue with PS operation but report an error and continue to retry CHPSWMT-11. If processing of the firewall configuration file produces an error, continue and report the error as an event.
CHPSWMT-13	TLS Alert Closure Handshake The PS tears down the TLS session immediately prior to sending the provisioning complete message.	CHPSWMT-13 MUST occur after CHPSWMT-12 completion	Continue to CHPSWMT-14. If failure for HTTP, retry per the specification. If retries all fail, report an error.
CHPSWMT-14	TCP/IP Tear-Down The TCP/IP session between the PS and the HTTP Server is torn down.	CHPSWMT-14 MUST occur after CHPSWMT-13 completion.	If the TCP/IP tear-down fails, report an error. Continue to 15.

Table 13-2/J.192 – Flow Descriptions for DHCP provisioning mode using HTTP/TLS

Flow step	PS WAN-Man provisioning: DHCP provisioning mode	Normal sequence	Failure sequence
CHPSWMT-15	Provisioning Complete If requested by the provisioning system the PS is required to inform the provisioning system of the status of PS provisioning. The provisioning system could request the PS to send a SYSLOG message or an SNMP trap, or both. If the PS successfully completes all required steps from CHPSWMT-1 through CHPSWMT-14 and the PS received a SYSLOG server address in the DHCP OFFER, the PS MUST send a provisioning complete message to the SYSLOG server with provisioning state set to PASS. If the PS successfully completes all required provisioning steps from CHPSWMT-1 through CHPSWMT-12 and the PS received valid parameters for docsDevNmAccessGroup identifying the Trap Receiver (docsDevNmAccessIP) and configuring the provisioning complete trap (cabhPsDevInitTrap) for 'read only with Traps' (set docsDevNmAccess control to '4'. Refer to [RFC 2669].), the PS MUST send a provisioning complete trap (cabhPsDevInitTrap) with appropriate parameters to the Trap Receiver. If the PS provisioning timer expires before all required steps from CHPSWMT-1 through CHPSWMT-14 are completed and the PS received a SYSLOG server address in the DHCP OFFER, the PS MUST send a provisioning complete message to the SYSLOG server with provisioning state set to FAIL. If the PS provisioning timer expires before all required steps from CHPSWMT-1 through CHPSWMT-14 are completed and the PS received valid parameters for docsDevNmAccessGroup identifying the Trap Receiver (docsDevNmAccessIP) and configuring the provisioning complete trap (cabhPsDevInitTrap) for 'read only with Traps' (set docsDevNmAccess control to '4'. Refer to [RFC 2669].), the PS MUST send a provisioning failed trap (cabhPsDevInitRetryTrap) to the Trap receiver. The PS updates the value of cabhPsDevProvState with status 'pass' (1) when provisioning flow steps CHPSWMT-1 through CHPSWMT-14 complete successfully. Refer to 7.5.4.	CHPSWMT-15 MUST occur after CHPSWMT-14 completion.	If the SNMP trap fails, the provisioning server may not know the provisioning process has completed unless it polls the cabhPsDevProvState object.

13.4 Provisioning the PS for management: SNMP provisioning mode

The PS requests a WAN-Man network address from the Headend DHCP server to be used for the exchange of management messages between the PS management functions and the cable network NMS. If the PS determines, based on the procedure described in 7.3.3.2.4, that it is to operate in SNMP Provisioning Mode, the PS will secure its management messages using SNMPv3, following the authentication procedure described in 11.3.2.

The cable network NMS may optionally instruct the PS (CMP) operating in SNMP Provisioning Mode to download a PS Configuration File from the TFTP server. Notification of completion of the provisioning process is provided through the Event Reporting process described in 6.3.3.2. The PS will operate without a PS Configuration File if it is not triggered to download the file.

Figure 13-6 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 Standalone PS. The Standalone PS provisioning SHOULD occur immediately after power-up/reset.

The provisioning process for the WAN-Man interface of a PS operating in SNMP Provisioning Mode MUST occur via the sequence depicted in Figure 13-6 and described in detail in Table 13-3. Optional steps are shown with a shaded background in Figure 13-6. These optional steps may be done immediately following step CHPSWMS-13, at a later time, or not at all.

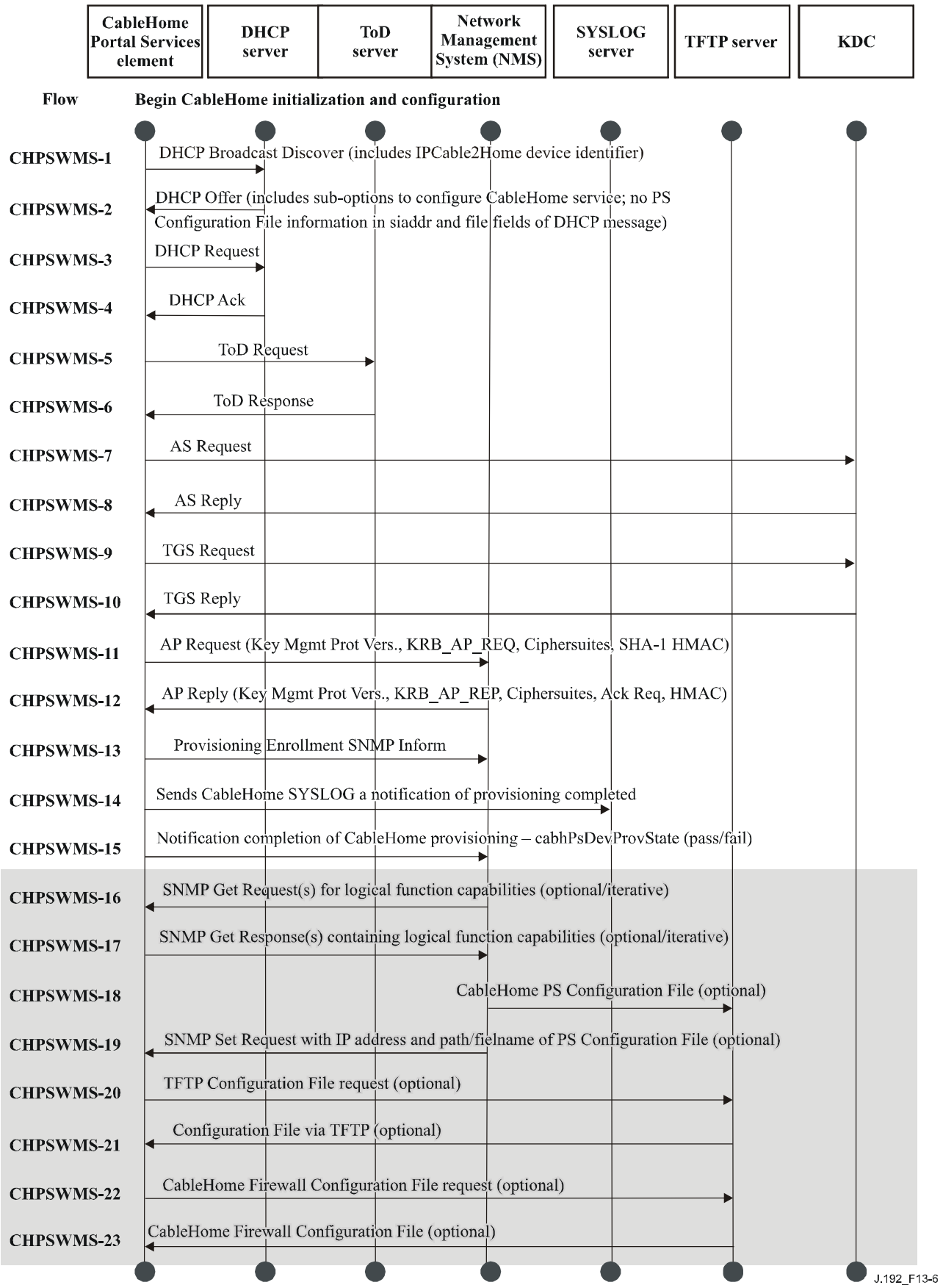


Figure 13-6/J.192 – Provisioning process for PS management – SNMP provisioning mode

Table 13-3 describes the individual steps of the provisioning process depicted in Figure 13-6.

Table 13-3/J.192 – Flow descriptions for PS WAN-Man provisioning process for SNMP provisioning mode

Flow step	PS WAN-Man provisioning: SNMP provisioning mode	Normal sequence	Failure sequence
CHPSWMS-1	DHCP Broadcast Discover The CDP (CDC) broadcasts a DHCP DISCOVER message to acquire the WAN-Man IP address as described in 7.3.3.2.4, CDC Requirements. The DHCP DISCOVER broadcast by the CDC includes mandatory options listed in Table 7-10, CDC DHCP Options in DISCOVER and REQUEST messages. The PS starts monitoring time elapsed AND sets cabhPsDevProvState to status 'inProgress' (2) when the CDC broadcasts its initial DHCP DISCOVER message.	Begin provisioning sequence.	If failure per DHCP protocol, report an error and continue to retry DHCP Broadcast Discover until successful (return to CHPSWMS-1). If the first attempt to acquire an address lease from the cable operator's DHCP server fails, initiate operation of the CDS as specified in 7.3.3.2.4, CDC Requirements.
CHPSWMS-2	DHCP OFFER	CHPSWMS-2 MUST occur after CHPSWMS-1 completion.	If failure per DHCP protocol, return to CHPSWMS-1 and report an error.
CHPSWMS-3	DHCP REQUEST The CDP sends to the appropriate DHCP server a DHCP REQUEST message to accept the DHCP OFFER.	CHPSWMS-3 MUST occur after CHPSWMS-2 completion.	If failure per DHCP protocol, return to CHPSWMS-1.
CHPSWMS-4	DHCP ACK The DHCP server sends the CDC a DHCP ACK message which contains the IPv4 address of the PS WAN-Man Interface and is expected to include the IPCable2Home option code 122 with sub-options 3, 6, & 10 AND no 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 ACK (see 7.3.3.2.4). The PS stores the Time of Day server address in cabhPsDevTimeServerAddr.	CHPSWMS-4 MUST occur after CHPSWMS-3 completion.	If failure per DHCP protocol, return to CHPSWMS-1 and report an error.

Table 13-3/J.192 – Flow descriptions for PS WAN-Man provisioning process for SNMP provisioning mode

Flow step	PS WAN-Man provisioning: SNMP provisioning mode	Normal sequence	Failure sequence
CHPSWMS-5	Time of Day (ToD) Request per [RFC 868] The PS issues a ToD Request message to the Time Server identified in the DHCP option 4 of the DHCP ACK message.	CHPSWMS-5 MUST occur after CHPSWMS-4 completion.	Continue with CHPSWMS-6.
CHPSWMS-6	ToD Response The ToD server is expected to reply with the current time in UTC format.	CHPSWMS-6 MUST occur after CHPSWMS-5 completion.	Retry Time Server synchronization up to a total of four attempts; if not successful in four attempts, attempt synchronization with the next Time Server listed in option 4 of DHCP ACK; if not successful after four attempts with each Time Server report an error, and return to CHPSWMS-1
CHPSWMS-7	AS Request (Note 1) The PS sends the AS Request message to the MSO IPCable2Home KDC provided in DHCP option 122 sub-option 10, to request a Kerberos ticket.	CHPSWMS-7 MUST occur after CHPSWMS-6 completion.	Return to CHPSWMS-1. PS initiates operation of CDS.
CHPSWMS-8	AS Reply The AS Reply Message is received from the MSO IPCable2Home 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 Request (optional) If the PS obtained a Ticket Granting Ticket (TGT) during step CHPSWMS-8, the PS sends the TGS Request message to the MSO KDC server whose address was passed to the PS (CDC) in DHCP option 122 sub-option 10.	If CHPSWMS-9 occurs, it MUST occur after CHPSWMS-8 completion.	Return to CHPSWMS-1. PS initiates operation of CDS.

Table 13-3/J.192 – Flow descriptions for PS WAN-Man provisioning process for SNMP provisioning mode

Flow step	PS WAN-Man provisioning: SNMP provisioning mode	Normal sequence	Failure sequence
CHPSWMS-10	TGS Reply (optional) The TGS Reply message containing the ticket is received from the MSO IPCable2Home KDC.	If CHPSWMS-9 occurs, CHPSWMS-10 MUST occur after CHPSWMS-9 completion.	Return to CHPSWMS-1. PS initiates operation of CDS.
CHPSWMS-11	AP Request The PS sends the AP Request message to the NMS (SNMP manager) to request keying information for SNMPv3, as described in 11.3, PS Device Authentication Infrastructure.	CHPSWMS-11 MUST occur after CHPSWMS-10 completion.	Return to CHPSWMS-1. PS initiates operation of CDS.
CHPSWMS-12	AP reply The AP Reply message is received from the NMS containing the keying information for SNMPv3. Note that the PS MUST establish SNMPv3 keys AND populate the associated SNMPv3 tables before it sends an SNMPv3 Inform message. The keys and tables are established using the information in the AP Reply. Refer to 11.3, PS Device Authentication Infrastructure.	CHPSWMS-12 MUST occur after CHPSWMS-11 completion.	Return to CHPSWMS-1. PS initiates operation of CDS.
CHPSWMS-13	Provisioning Enrollment SNMP Inform After the PS operating in SNMP Provisioning Mode establishes SNMPv3 keys, the PS MUST send an SNMPv3 INFORM (cabhPsDevProvEnrollTrap) requesting enrolment to the SNMP MANAGER whose IP address was provided in DHCP option 122 sub-option 3, in the DHCP ACK message. After the PS sends the cabhPsDevProvEnrollTrap described above, the PS is required to begin monitoring elapsed time, as described in 7.4.4.2.2.	CHPSWMS-13 MUST occur after CHPSWMS-12 completion.	Return to CHPSWMS-1.

Table 13-3/J.192 – Flow descriptions for PS WAN-Man provisioning process for SNMP provisioning mode

Flow step	PS WAN-Man provisioning: SNMP provisioning mode	Normal sequence	Failure sequence
CHPSWMS-14	SNMP Get (optional) If any additional device capabilities are needed by the provisioning system, the provisioning system requests these from the PS via SNMPv3 Get Requests. Iterative: The NMS sends the PS one or more SNMPv3 GET requests to obtain any needed PS capability information. The Provisioning Application may use a GetBulkRequest to obtain several pieces of information in a single message.	CHPSWMS-14 is not expected to occur before CHPSWMS-13 completion.	Return to CHPSWMS-1.
CHPSWMS-15	SNMP Get Response (optional) Iterative: The PS replies to the NMS Get-request or GetBulkRequest 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-14 occurs, CHPSWMS-15 MUST occur after CHPSWMS-14 completes.	N/A
CHPSWMS-16	Configuration File Create Optional: The provisioning system uses information from PS provisioning steps CHPSWMS-16 and CHPSWMS-17 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-15 occurs, CHPSWMS-16 MUST occur after CHPSWMS-15 completes.	N/A

Table 13-3/J.192 – Flow descriptions for PS WAN-Man provisioning process for SNMP provisioning mode

Flow step	PS WAN-Man provisioning: SNMP provisioning mode	Normal sequence	Failure sequence
CHPSWMS-17	SNMP Set (optional) 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 7.4.4.2.2, PS Configuration File Download Trigger for 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 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-16 occurs, CHPSWMS-17 MUST occur after CHPSWMS-16 completes.	Return to CHPSWMS-1 if the set was received, but there was a processing error.
CHPSWMS-18	TFTP Request If the NMS triggers the PS to download a PS Configuration File as described in 7.4.4.2.2, the PS sends the TFTP Server a TFTP Get Request to request the specified PS Configuration File.	If CHPSWMS-17 occurs, CHPSWMS-18 MUST occur after CHPSWMS-17 completes.	Continue with CHPSWMS-17.
CHPSWMS-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-17. The PS then processes the PS Configuration File. Optionally, the IP Address of the Firewall Configuration File TFTP server, the Firewall Configuration File filename and the hash of the firewall configuration file 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-18 occurs, CHPSWMS-19 occurs after CHPSWMS-18 completes.	If the TFTP download fails, report an error, and continue to retry CHPSWMS-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.

Table 13-3/J.192 – Flow descriptions for PS WAN-Man provisioning process for SNMP provisioning mode

Flow step	PS WAN-Man provisioning: SNMP provisioning mode	Normal sequence	Failure sequence
CHPSWMS-20	SYSLOG message If the PS received a SYSLOG server address in the DHCP ACK, the PS MUST send the SYSLOG a "provisioning complete" message. This notification will include the pass-fail result of the provisioning operation. The general format of this message is defined in Table B.1, Defined Events for IPCable2Home, Event ID 73001100 (see Message Notes and Details).	CHPSWMS-20 MUST occur after CHPSWMS-19 completion.	
CHPSWMS-21	SNMP Inform The PS MUST send the NMS an SNMP INFORM (cabhPsDevInitTrap) containing a "provisioning complete" notification and set the value of cabhPsDevProvState to pass(1) in either of the following circumstances: – The PS was not triggered by the NMS to download a PS configuration file before the amount of time specified by the value of cabhPsDevProvisioningTimer elapsed since the enrolment inform described in CHPSWMS-13. – The PS was triggered by the NMS to download a PS configuration file within the time defined by the value of cabhPsDevProvisioningTimer following the enrolment inform, and the PS successfully downloaded and processed the PS configuration file. The configuration file download and processing is not required to complete before the time defined by the value of cabhPsDevProvisioningTimer elapses after the enrolment inform. The PS MUST NOT send an SNMP INFORM (cabhPsDevInitTrap) containing a "provisioning complete" notification, and the PS MUST set the value of cabhPsDevProvState to fail(3), if the PS was triggered by the NMS to download a PS configuration file before the amount of time specified by the value of cabhPsDevProvisioningTimer elapsed since the enrolment inform was issued AND the PS fails to successfully download and process the configuration file within the maximum number of retries defined in 7.4.4.2.4, Post-Trigger Operation.	CHPSWMS-21 MUST occur after CHPSWMS-20 completion.	If the PS does not receive a response to the Provisioning Complete inform, the PS MUST retry to send the cabhPsDevInitTrap inform, for a total of 5 attempts, at an interval of 10 seconds. If all 5 attempts to send the cabhPsDevInitTrap fail, the PS MUST re-start the initialization process: return to CHPSWMS-1 and report an error.

Table 13-3/J.192 – Flow descriptions for PS WAN-Man provisioning process for SNMP provisioning mode

Flow step	PS WAN-Man provisioning: SNMP provisioning mode	Normal sequence	Failure sequence
CHPSWMS-22	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 occurs, it MUST occur after CHPSWMS-21 completes.	Return to CHPSWMS- 1.
CHPSWMS-23	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. The file is then processed. Refer to 7.4.4 for description of PS configuration file contents.	If CHPSWMS-22 occurs, CHPSWMS-23 MUST occur after CHPSWMS-22 completes.	If the TFTP download fails, continue with PS operation but report an error and continue to retry CHPSWMS-22. If processing of the firewall configuration file produces an error, continue and report the error as an event.
NOTE 1 – Steps CHPSWMS-5-CHPSWMS-8 are optional in some cases. Refer to clause 11 for details. NOTE 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.			

13.4.1 PS WAN-Man configuration file download

The PS operating in SNMP Provisioning Mode might contain sufficient factory default information to provide for operation of either or both LAN and WAN sides without a PS Configuration File being downloaded. If the PS is operating in SNMP Provisioning Mode, the NMS might trigger the download of a PS Configuration File for initial provisioning to replace the factory defaults or to provide additional information.

The firewall Configuration File contains information to provision the firewall function. The indication to download a firewall Configuration File will come in either the PS Configuration File or via an SNMP Set during initialization.

13.4.2 PS provisioning timer

A provisioning timer is provided to ensure that the PS will continue with the SNMP Provisioning Mode provisioning process if the PS is not triggered to download a PS Configuration File. The PS is required to monitor time elapsed from the time the PS issues the Provisioning Enrollment SNMP Inform. If the PS is not triggered to download a PS Configuration File in the time defined by the value of the provisioning timer, the PS signals that provisioning is complete by issuing a Provisioning Complete SNMP Inform and setting the value of cabhPsDevProvState to pass(1). The timer object, cabhPsDevProvTimer, has a default value of 5 minutes. Refer to 7.4.4.2.2 for more information.

13.4.3 Provisioning enrolment/Provisioning complete informs

For the PS operating in SNMP Provisioning Mode only, the provisioning enrolment inform (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) indicates whether the provisioning sequence has completed successfully or not.

13.4.4 SYSLOG provisioning

The syslog server IP address MUST be provisioned through the DHCP process. The syslog event will not be sent if the syslog server IP address is not configured.

13.4.5 Provisioning state and error reporting

As indicated in Tables 13-1 and 13-3, failure of the steps in the provisioning process generally results in the process restarting at the first step, CHPSWMD-1 or CHPSWMS-1.

13.5 PS WAN-Data provisioning process

The PS requests zero or more WAN-Data network address(es) from the DHCP server in the cable network to be used for the exchange of data between elements connected to the Internet and LAN IP Devices.

There is no difference in PS WAN-Data operation between the DHCP and SNMP Provisioning Modes.

Figure 13-7 illustrates 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 DOCSIS cable modem as it is for the standalone PS.

If the provisioning process for the PS WAN-Data address(es) occurs, it MUST follow the sequence depicted in Figure 13-7 and described in detail in Table 13-4.

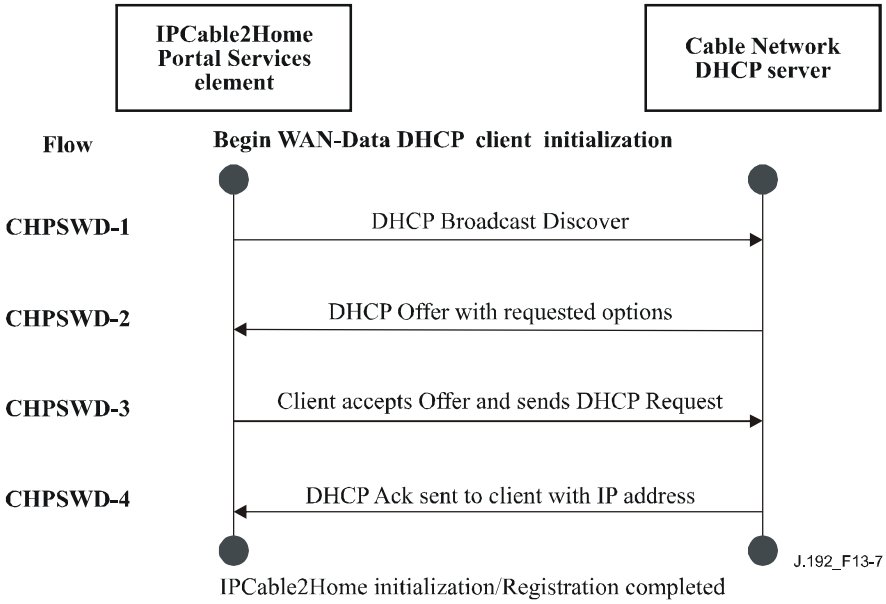


Figure 13-7/J.192 – PS WAN-Data provisioning process

Table 13-4/J.192 – Flow descriptions for PS WAN-Data provisioning process

Flow step	PS WAN-Data address provisioning	Normal sequence	Failure sequence
CHPSWD-1	DHCP Broadcast Discover The PS broadcasts a DHCP DISCOVER message including the mandatory options listed in Table 7-10, CDC DHCP Options in DISCOVER and REQUEST messages.	Proceed to CHPSWD-2.	If failure per DHCP protocol, repeat CHPSWD-1.
CHPSWD-2	DHCP OFFER The DHCP Server at the Headend receives the DHCP DISCOVER packet, assigns an IP address from the WAN-Data pool, builds a DHCP OFFER packet, and transmits the DHCP OFFER to the DHCP Relay Agent [RFC 3046] in the CMTS.	Proceed to CHPSWD-3.	If failure, the client will time out per DHCP protocol and CHPSWD-1 will be repeated.
CHPSWD-3	DHCP REQUEST The CDP sends a DHCP REQUEST message to the selected DHCP server to accept the DHCP OFFER in accordance with client requirements of [RFC 2131].	CHPSWD-3 MUST occur after CHPSWD-2 completion.	If failure per DHCP protocol, return to CHPSWD-1.
CHPSWD-4	DHCP ACK The DHCP server sends the CDP a DHCP ACK message which contains the IPv4 address for the PS WAN Data interface.	CHPSWD-4 MUST occur after CHPSWD-3 completion. Provisioning complete with completion of CHPSWD-4.	If failure per DHCP protocol, return to CHPSWD-1.

13.6 Provisioning process: LAN IP Device in the LAN-Pass realm

Some home LAN applications will not function properly with a translated network address. To accommodate these applications, the PS is enabled to operate in Passthrough (transparent bridging) mode. As described in 8.3.3.1, Packet Handling Modes, bridging occurs when the cable network NMS sets the Primary Packet-handling mode (cabhCapPrimaryMode) to Passthrough, or by writing individual LAN IP Device MAC addresses into the Passthrough Table (cabhCapPassthroughTable). When the PS has been configured to bridge traffic for a LAN IP Device, DHCP DISCOVERs and DHCP REQUESTs issued by that LAN IP Device will be served by the cable network DHCP server, not by the CDS.

A non-IPCable2Home compliant LAN IP Device is assumed to implement a DHCP client and request an IP address lease using DHCP [RFC 2131]. An IPCable2Home compliant LAN IP Device, i.e., one that implements BP functionality defined in this Recommendation, is required to implement a DHCP client and request an IP address lease via DHCP.

Annex A

MIB objects

This annex lists all required MIB objects, as indicated in 6.3.3.1.4.1, SNMP Protocol Requirements, and 6.3.3.1.4.7, IPCable2Home MIB Requirements, and indicates requirement for persistence of each listed object.

The term 'persistent' as it applies to this annex is defined below:

Persistent: The requirement for the PS to retain the value of a configurable (by the manager or by the PS itself) MIB object across a PS reboot or reset.

For MIB objects with entry 'Yes' in the Persistent column, the object's value immediately following a PS reboot or reset MUST be the same as its value immediately preceding the reboot or reset.

For MIB objects with entry 'No' in the Persistent column, the object's value MUST be set to its factory default value (DEFVAL) or, if it has no default value, it MUST be set to zero or null as appropriate, immediately following a PS reboot or reset.

For MIB objects with entry "-" in the Persistent column, one of the following applies:

- the value of the object immediately following PS reboot, or reset is left to vendor implementation because there is no specific requirement for its value following PS reboot or reset; or
- the value of the object is deterministic, based upon the MIB description (the object's value is fixed or can be derived from known values after the PS reboot or reset).

MIB NAME/Parameter	Max-Access	Persistent	No. of Persistent Entries
mib-2 [RFC 1213] system			
sysDescr	read-only	–	N/A
sysObjectID	read-only	–	N/A
sysUpTime	read-only	–	N/A
sysContact	read-write	Yes	1
sysName	read-write	Yes	1
sysLocation	read-write	Yes	1
sysServices	read-only	–	N/A
interfaces [RFC 2863]			
ifNumber	read-only	–	N/A
ifTable/ ifEntry			
ifIndex	read-only	–	N/A
ifDescr	read-only	–	N/A
ifType	read-only	–	N/A
ifMtu	read-only	–	N/A
ifSpeed	read-only	–	N/A

MIB NAME/Parameter	Max-Access	Persistent	No. of Persistent Entries
ifPhysAddress	read-only	–	N/A
ifAdminStatus	read-write	No[2]	N/A
ifOperStatus	read-only	–	N/A
ifLastChange	read-only	–	N/A
ifInOctets	read-only	–	N/A
ifInUcastPkts	read-only	–	N/A
ifInDiscards	read-only	–	N/A
ifInErrors	read-only	–	N/A
ifInUnknownProtos	read-only	–	N/A
ifOutOctets	read-only	–	N/A
ifOutUcastPkts	read-only	–	N/A
ifOutDiscards	read-only	–	N/A
ifOutErrors	read-only	–	N/A

[2] ifAdminStatus is persistent for ifIndex = 254 and is not persistent for other values of ifIndex.

ip [RFC 2011]			
ipForwarding	read-write	No	N/A
ipDefaultTTL	read-write	No	N/A
ipInReceives	read-only	–	N/A
ipInHdrErrors	read-only	–	N/A
ipInAddrErrors	read-only	–	N/A
ipForwDatagrams	read-only	–	N/A
ipInUnknownProtos	read-only	–	N/A
ipInDiscards	read-only	–	N/A
ipInDelivers	read-only	–	N/A
ipOutRequests	read-only	–	N/A
ipOutDiscards	read-only	–	N/A
ipOutNoRoutes	read-only	–	N/A
ipReasmTimeout	read-only	–	N/A
ipReasmReqds	read-only	–	N/A
ipReasmOKs	read-only	–	N/A
ipReasmFails	read-only	–	N/A
ipFragOKs	read-only	–	N/A
ipFragFails	read-only	–	N/A
ipFragCreates	read-only	–	N/A
ipNetToMediaTable/ ipNetToMediaEntry			
ipNetToMediaIfIndex	read-only	No	N/A
ipNetToMediaPhyAddress	read-only	No	N/A

MIB NAME/Parameter	Max-Access	Persistent	No. of Persistent Entries
ipNetToMediaNetAddress	read-only	No	N/A
ipNetToMediaType	read-only	No	N/A
icmp			
icmpInMsgs	read-only	–	N/A
icmpInErrors	read-only	–	N/A
icmpInDestUnreachs	read-only	–	N/A
icmpInTimeExcds	read-only	–	N/A
icmpInParmProbs	read-only	–	N/A
icmpInSrcQuenchs	read-only	–	N/A
icmpInRedirects	read-only	–	N/A
icmpInEchos	read-only	–	N/A
icmpInEchosReps	read-only	–	N/A
icmpInTimestamps	read-only	–	N/A
icmpInTimestampsReps	read-only	–	N/A
icmpInAddrMasks	read-only	–	N/A
icmpInAddrMaskReps	read-only	–	N/A
icmpOutMsgs	read-only	–	N/A
icmpOutErrors	read-only	–	N/A
icmpOutDestUnreachs	read-only	–	N/A
icmpOutTimeExcds	read-only	–	N/A
icmpOutParmProbs	read-only	–	N/A
icmpOutSrcQuenchs	read-only	–	N/A
icmpOutRedirects	read-only	–	N/A
icmpOutEchos	read-only	–	N/A
icmpOutEchosReps	read-only	–	N/A
icmpOutTimestamps	read-only	–	N/A
icmpOutTimestampReps	read-only	–	N/A
icmpOutAddrMasks	read-only	–	N/A
icmpOutAddrMaskReps	read-only	–	N/A
udp [RFC 2013]			
udpInDatagrams	read-only	–	N/A
udpNoPorts	read-only	–	N/A
udpInErrors	read-only	–	N/A
udpOutDatagrams	read-only	–	N/A
udpTable/ udpEntry			
udpLocalAddress	read-only	–	N/A
udpLocalPort	read-only	–	N/A

MIB NAME/Parameter	Max-Access	Persistent	No. of Persistent Entries
transmission [draft-ietf-ipcdn-bpiplus-mib-05]			
docsIfMib			
docsBpi2MIB			
docsBpi2MIBObjects			
docsBpi2CmObjects			
docsBpi2CmCertObjects			
docsBpi2CodeDownloadGroup			
docsBpi2CodeDownloadStatusCode	read-only	–	N/A
docsBpi2CodeDownloadStatusString	read-only	–	N/A
docsBpi2CodeMfgOrgName	read-only	Yes	1
docsBpi2CodeMfgCodeAccessStart	read-only	Yes	1
docsBpi2CodeMfgCvcAccessStart	read-only	Yes	1
docsBpi2CodeCoSignerOrgName	read-only	–	N/A
docsBpi2CodeCoSignerCodeAccessStart	read-only	–	N/A
docsBpi2CodeCoSignerCvcAccessStart	read-only	–	N/A
docsBpi2CodeCvcUpdate	read-write	No	N/A
snmp [RFC 3418]			
snmpInPkts	read-only	–	N/A
snmpInBadVersions	read-only	–	N/A
snmpInBadCommunityNames	read-only	–	N/A
snmpInBadCommunityUses	read-only	–	N/A
snmpInASNParseErrs	read-only	–	N/A
snmpEnableAuthenTraps	read-write	No	N/A
snmpSilentDrops	read-only	–	N/A
ifMIB [RFC 2863]			
ifMIBOjects			
ifXTable/			
ifXEntry			
ifName	read-only	–	N/A
ifInMulticastPkts	read-only	–	N/A
ifInBroadcastPkts	read-only	–	N/A
ifOutMulticastPkts	read-only	–	N/A
ifOutBroadcastPkts	read-only	–	N/A
ifLinkUpDownTrapEnable	read-write	No	N/A
ifHighSpeed	read-only	–	N/A
ifPromiscuousMode	read-write	No	N/A
ifConnectorPresent	read-only	–	N/A

MIB NAME/Parameter	Max-Access	Persistent	No. of Persistent Entries
ifAlias	read-write	No	N/A
ifCounterDiscontinuityTime	read-only	–	N/A
ifStackTable/ ifStackEntry			
ifStackHigherLayer	read-only	–	N/A
ifStackLowerLayer	read-only	–	N/A
ifStackStatus	read-only	–	N/A
docsDev [RFC 2669] docsDevMIBObjects			
docsDevNmAccessTable/ docsDevNmAccessEntry			
docsDevNmAccessIndex	not-accessible	–	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	Yes	1
docsDevSwOperStatus	read-only	Yes	1
docsDevSwCurrentVers	read-only	–	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
docsDevEvThrottleThreshold	read-write	No	N/A
docsDevEvThrottleInterval	read-write	No	N/A
docsDevEvControlTable/ docsDevEvControlEntry			
docsDevEvPriority	not-accessible	–	N/A
docsDevEvReporting	read-write	No	N/A

MIB NAME/Parameter	Max-Access	Persistent	No. of Persistent Entries
docsDevEventTable/ docsDevEventEntry			
docsDevEvIndex	not-accessible	–	N/A
docsDevEvFirstTime	read-only	Yes	10
docsDevEvLastTime	read-only	Yes	10
docsDevEvCounts	read-only	Yes	10
docsDevEvLevel	read-only	Yes	10
docsDevEvId	read-only	Yes	10
docsDevEvText	read-only	Yes	10
docsDevFilter			
docsDevFilterIpTable/ docsDevFilterIpEntry			
docsDevFilterIpIndex	not-accessible	–	N/A
docsDevFilterIpStatus	read-create	Yes	40
docsDevFilterIpControl	read-create	Yes	40
docsDevFilterIpIfIndex	read-create	Yes	40
docsDevFilterIpDirection	read-create	Yes	40
docsDevFilterIpBroadcast	read-create	No	N/A
docsDevFilterIpSaddr	read-create	Yes	40
docsDevFilterIpSmask	read-create	Yes	40
docsDevFilterIpDaddr	read-create	Yes	40
docsDevFilterIpDmask	read-create	Yes	40
docsDevFilterIpProtocol	read-create	Yes	40
docsDevFilterIpSourcePortLow	read-create	Yes	40
docsDevFilterIpSourcePortHigh	read-create	Yes	40
docsDevFilterIpDestPortLow	read-create	Yes	40
docsDevFilterIpDestPortHigh	read-create	Yes	40
docsDevFilterIpMatches	read-only	–	N/A
docsDevFilterIpTos	read-create	No	N/A
docsDevFilterIpTosMask	read-create	No	N/A
docsDevFilterIpContinue	read-only	–	N/A
docsDevFilterIpPolicyId	read-create	No	N/A
dot11			
dot11StationConfigTable/ dot11StationConfigEntry[3]			

MIB NAME/Parameter	Max-Access	Persistent	No. of Persistent Entries
dot11PrivacyOptionImplemented	read-only	–	N/A
dot11DesiredSSID	read-write	Yes	1
dot11OperationalRateSet	read-write/ read-only	Yes/–	1/N/A
dot11BeaconPeriod	read-write/ read-only	Yes/–	1/N/A
dot11DTIMPeriod	read-write/ read-only	Yes/–	1/N/A
dot11WEPDefaultKeysTable/ dot11WEPDefaultKeysEntry[3]			
dot11WEPDefaultKeyIndex	not-accessible	N/A	N/A
dot11WEPDefaultKeyValue	read-write	Yes	4
dot11PrivacyTable/ dot11PrivacyEntry[3]			
dot11PrivacyInvoked	read-write	Yes	1
dot11WEPDefaultKeyID	read-write	Yes	1
dot11OperationTable/ dot11OperationEntry[3]			
dot11MACAddress	read-only	–	N/A
dot11RTSThreshold	read-write/ read-only	Yes/–	1/N/A
dot11FragmentationThreshold	read-write/ read-only	Yes/–	1/N/A
dot11PhyTxPowerTable/ dot11PhyTxPowerEntry[3]			
dot11NumberSupportedPowerLevels	read-only	–	N/A
dot11TxPowerLevel1	read-only	–	N/A
dot11TxPowerLevel2	read-only	–	N/A
dot11TxPowerLevel3	read-only	–	N/A
dot11TxPowerLevel4	read-only	–	N/A
dot11TxPowerLevel5	read-only	–	N/A
dot11TxPowerLevel6	read-only	–	N/A
dot11TxPowerLevel7	read-only	–	N/A
dot11TxPowerLevel8	read-only	–	N/A
dot11CurrentTxPowerLevel	read-write	Yes	1
dot11PhyDSSSTable/ dot11PhyDSSSEntry[3]			
dot11CurrentChannel	read-write	Yes	1

MIB NAME/Parameter	Max-Access	Persistent	No. of Persistent Entries
dot11PhyOFDMTable/ dot11PhyOFDMEntry[3]			
dot11CurrentFrequency	read-write	Yes	1
dot11FrequencyBandsSupported	read-only	–	N/A

[3] MIB Objects with number of persistent requirements are indicated per supported wireless interface supporting the functionality of the MIB entry.

private
enterprises
cableLabs
clabProject
clabProjCableHome
cabhPsDevMib
cabhPsDevBase

cabhPsDevDateTime	read-write	No	N/A
cabhPsDevResetNow	read-write	No	N/A
cabhPsDevSerialNumber	read-only	–	N/A
cabhPsDevHardwareVersion	read-only	–	N/A
cabhPsDevWanManMacAddress	read-only	–	N/A
cabhPsDevWanDataMacAddress	read-only	–	N/A
cabhPsDevTypeIdentifier	read-only	–	N/A
cabhPsDevSetToFactory	read-write	No	N/A
cabhPsDevTodSyncStatus	read-only	–	N/A
cabhPsDevProvMode	read-only	–	N/A
cabhPsDevLastSetToFactory	read-only	–	N/A
cabhPsDevTrapControl	read-write	No	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
cabhPsDevProvConfigFileStatus	read-only	–	N/A
cabhPsDevProvConfigTLVProcessed	read-only	–	N/A
cabhPsDevProvConfigTLVRejected	read-only	–	N/A
cabhPsDevProvSolicitedKeyTimeout	read-write	Yes	1
cabhPsDevProvState	read-only	–	N/A
cabhPsDevProvAuthState	read-only	–	N/A
cabhPsDevTimeServerAddrType	read-only	–	N/A
cabhPsDevTimeServerAddr	read-only	–	N/A

MIB NAME/Parameter	Max-Access	Persistent	No. of Persistent Entries
cabhPsDevAttrib **cabhPsDevPsAttrib**			
cabhPsDevPsDeviceType	read-only	–	N/A
cabhPsDevPsManufacturerURL	read-only	–	N/A
cabhPsDevPsModelURL	read-only	–	N/A
cabhPsDevPsModelUPC	read-only	–	N/A
cabhPsDevPsStats			
cabhPsDevLanIpTrafficCountersReset	read-write	No	N/A
cabhPsDevLanIpTrafficCountersLastReset	read-only	–	N/A
cabhPsDevLanIpTrafficEnabled	read-write	No	N/A
cabhPsDevLanIpTrafficTable/ *cabhPsDevLanIpTrafficEntry*			
cabhPsDevLanIpTrafficIndex	not-accessible	–	N/A
cabhPsDevLanIpTrafficInetAddressType	read-only	–	N/A
cabhPsDevLanIpTrafficInetAddress	read-only	–	N/A
cabhPsDevLanIpTrafficInOctets	read-only	–	N/A
cabhPsDevLanIpTrafficOutOctets	read-only	–	N/A
cabhPsDevPsAccessControl			
cabhPsDevAccessControlEnable	read-write	No	N/A
cabhPsDevAccessControlTable/ *cabhPsDevAccessControlEntry*			
cabhPsDevAccessControlIndex	not-accessible	–	N/A
cabhPsDevAccessControlPhysAddr	read-write	Yes	20
cabhPsDevAccessControlRowStatus	read-create	Yes	20
cabhPsDevPsMisc **cabhPsDevPsUI**			
cabhPsDevUILogin	read-write	Yes	1
cabhPsDevUIPassword	read-write	Yes	1
cabhPsDevUISelection	read-write	Yes	1
cabhPsDevUIServerURL	read-write	Yes	1
cabhPsDevUISelectionDisabledBodyText	read-write	Yes	1
cabhPsDev802dot11BaseTable/ *cabhPsDev802dot11BaseEntry*[4]			
cabhPsDev802dot11BaseSetToDefault	read-write	–	N/A
cabhPsDev802dot11BaseLastSetToDefault	read-only	–	N/A
cabhPsDev802dot11BaseAdvertiseSSID	read-write	Yes	1

MIB NAME/Parameter	Max-Access	Persistent	No. of Persistent Entries
cabhPsDev802dot11BasePhyCapabilities	read-only	–	N/A
cabhPsDev802dot11BasePhyOperMode	read-write	Yes	1
cabhPsDev802dot11SecTable/ cabhPsDev802dot11SecEntry[4]			
cabhPsDev802dot11SecCapabilities	read-only	–	N/A
cabhPsDev802dot11SecOperMode	read-write	Yes	1
cabhPsDev802dot11SecPassPhraseToWEPKey	read-write	Yes	1
cabhPsDev802dot11SecUsePassPhraseToWEPKeyAlg	read-write	Yes	1
cabhPsDev802dot11SecPSKPassPhraseToKey	read-write	Yes	1
cabhPsDev802dot11SecWPAPreSharedKey	read-write	Yes	1
cabhPsDev802dot11SecWPARekeyTime	read-write	Yes	1
cabhPsDev802dot11SecControl	read-write	No	N/A
cabhPsDev802dot11SecCommitStatus	read-only	No	N/A
cabhPsDevUpnp **cabhPsDevUpnpBase**			
cabhPsDevUpnpEnabled	read-write	Yes	1
cabhPsDevUpnpCommands			
cabhPsDevUpnpCommandIpType	read-write	No	N/A
cabhPsDevUpnpCommandIp	read-write	No	N/A
cabhPsDevUpnpCommand	read-write	No	N/A
cabhPsDevUpnpCommandUpdate	read-write	No	N/A
cabhPsDevUpnpLastCommandUpdate	read-only	–	N/A
cabhPsDevUpnpCommandStatus	read-only	–	N/A
cabhPsDevUpnpInfoTable/ cabhPsDevUpnpInfoEntry			
cabhPsDevUpnpInfoXmlFragment	read-only	–	N/A
cabhSecMib *cabhSecCertObjects*			
cabhSecCertPsCert	read-only	–	1
cabhSec2FwObjects **cabhSec2FwBase**			
cabhSec2FwEnable	read-write	Yes	N/A
cabhSec2FwPolicyFileURL	read-write	No	N/A

[4] MIB Objects with number of persistent requirements are indicated per supported wireless interface supporting the functionality of the MIB entry.

MIB NAME/Parameter	Max-Access	Persistent	No. of Persistent Entries
cabhSec2FwPolicyFileHash	read-write	No	N/A
cabhSec2FwPolicyFileOperStatus	read-only	–	N/A
cabhSec2FwPolicyFileCurrentVersion	read-write	Yes	N/A
cabhSec2FwClearPreviousRuleset	read-write	No	N/A
cabhSec2FwPolicySelection	read-write	Yes	N/A
cabhSec2FwEventSetToFactory	read-write	No	N/A
cabhSec2FwEventLastSetToFactory	read-only	–	N/A
cabhSec2FwPolicySuccessfulFileURL	read-only	Yes	1
cabhSec2FwConfiguredRulesetPriority	read-only	Yes	1
cabhSec2FwClearLocalRuleset	read-write	No	N/A
cabhSec2FwEvent			
cabhSec2FwEventControlTable/ cabhSec2FwEventControlEntry			
cabhSec2FwEventType	not-accessible	–	N/A
cabhSec2FwEventEnable	read-write	No	N/A
cabhSec2FwEventThreshold	read-write	No	N/A
cabhSec2FwEventInterval	read-write	No	N/A
cabhSec2FwEventCount	read-only	–	N/A
cabhSec2FwEventLogReset	read-write	No	N/A
cabhSec2FwEventLogLastReset	read-only	–	N/A
cabhSec2FwLogTable/ cabhSec2FwLogEntry			
cabhSec2FwLogIndex	not-accessible	–	N/A
cabhSec2FwLogEventType	read-only	Yes	40
cabhSec2FwLogEventPriority	read-only	Yes	40
cabhSec2FwLogEventId	read-only	Yes	40
cabhSec2FwLogTime	read-only	Yes	40
cabhSec2FwLogIpProtocol	read-only	Yes	40
cabhSec2FwLogIpSourceAddr	read-only	Yes	40
cabhSec2FwLogIpDestAddr	read-only	Yes	40
cabhSec2FwLogIpSourcePort	read-only	Yes	40
cabhSec2FwLogIpDestPort	read-only	Yes	40
cabhSec2FwLogMessageType	read-only	Yes	40
cabhSec2FwLogReplayCount	read-only	Yes	40
cabhSec2FwLogMIBPointer	read-only	Yes	40
cabhSec2FwLogMatchingFilterTableName	read-only	Yes	40
cabhSec2FwLogMatchingFilterTableIndex	read-only	Yes	40
cabhSec2FwLogMatchingFilterDescr	read-only	Yes	40

MIB NAME/Parameter	Max-Access	Persistent	No. of Persistent Entries
cabhSec2FwFilter			
cabhSec2FwFilterScheduleTable/ cabhSec2FwFilterScheduleEntry			
cabhSec2FwFilterScheduleStartTime	read-create	Yes	40
cabhSec2FwFilterScheduleEndTime	read-create	Yes	40
cabhSec2FwFilterScheduleDOW	read-create	Yes	40
cabhSec2FwFilterScheduleDescr	read-create	Yes	40
cabhSec2FwLocalFilterIpTable / cabhSec2FwLocalFilterIpEntry			
cabhSec2FwLocalFilterIpIndex	not-accessible	–	N/A
cabhSec2FwLocalFilterIpStatus	read-create	Yes	40
cabhSec2FwLocalFilterIpControl	read-create	Yes	40
cabhSec2FwLocalFilterIpIfIndex	read-create	Yes	40
cabhSec2FwLocalFilterIpDirection	read-create	Yes	40
cabhSec2FwLocalFilterIpSaddr	read-create	Yes	40
cabhSec2FwLocalFilterIpSmask	read-create	Yes	40
cabhSec2FwLocalFilterIpDaddr	read-create	Yes	40
cabhSec2FwLocalFilterIpDmask	read-create	Yes	40
cabhSec2FwLocalFilterIpProtocol	read-create	Yes	40
cabhSec2FwLocalFilterIpSourcePortLow	read-create	Yes	40
cabhSec2FwLocalFilterIpSourcePortHigh	read-create	Yes	40
cabhSec2FwLocalFilterIpDestPortLow	read-create	Yes	40
cabhSec2FwLocalFilterIpDestPortHigh	read-create	Yes	40
cabhSec2FwLocalFilterIpMatches	read-only	Yes	40
cabhSec2FwLocalFilterIpContinue	read-only	Yes	40
cabhSec2FwLocalFilterIpStartTime	read-create	Yes	40
cabhSec2FwLocalFilterIpEndTime	read-create	Yes	40
cabhSec2FwLocalFilterIpDOW	read-create	Yes	40
cabhSec2FwLocalFilterIpDescr	read-create	Yes	40
cabhSec2FwFactoryDefault			
cabhSec2FwFactoryDefaultTable/ cabhSec2FwFactoryDefaultEntry			
cabhSec2FwFactoryDefaultIndex	not-accessible		
cabhSec2FwFactoryDefaultControl		–	N/A

MIB NAME/Parameter	Max-Access	Persistent	No. of Persistent Entries
cabhSec2FwFactoryDefaultIfIndex		–	N/A
cabhSec2FwFactoryDefaultDirection		–	N/A
cabhSec2FwFactoryDefaultSaddr		–	N/A
cabhSec2FwFactoryDefaultSmask		–	N/A
cabhSec2FwFactoryDefaultDaddr		–	N/A
cabhSec2FwFactoryDefaultDmask		–	N/A
cabhSec2FwFactoryDefaultProtocol		–	N/A
cabhSec2FwFactoryDefaultSourcePortLow		–	N/A
cabhSec2FwFactoryDefaultSourcePortHigh		–	N/A
cabhSec2FwFactoryDefaultDestPortLow		–	N/A
cabhSec2FwFactoryDefaultDestPortHigh		–	N/A
cabhSec2FwFactoryDefaultIFilterContinue		–	N/A
cabhSecKerbBase			
cabhSecKerbPKINITGracePeriod	read-write	No	N/A
cabhSecKerbTGSGracePeriod	read-write	No	N/A
cabhSecKerbUnsolicitedKeyMaxTimeout	read-write	No	N/A
cabhSecKerbUnsolicitedKeyMaxRetries	read-write	No	N/A
cabhCapMib **cabhCapObjects** **cabhCapBase**			
cabhCapTcpTimeWait	read-write	No	N/A
cabhCapUdpTimeWait	read-write	No	N/A
cabhCapIcmpTimeWait	read-write	No	N/A
cabhCapPrimaryMode	read-write	No	N/A
cabhCapSetToFactory	read-write	No	N/A
cabhCapLastSetToFactory	read-only	–	N/A
CabhCapUpnpPortForwardingEnable	read-write	Yes	1
CabhCapUpnpTimeWait	read-write	No	N/A
cabhCapMap			
cabhCapMappingTable/ *cabhCapMappingEntry*			
cabhCapMappingIndex	not-accessible	–	N/A
cabhCapMappingWanAddrType	read-create	Yes[5]	16
cabhCapMappingWanAddr	read-create	Yes[5]	16
cabhCapMappingWanPort	read-create	Yes[5]	16
cabhCapMappingLanAddrType	read-create	Yes[5]	16

MIB NAME/Parameter	Max-Access	Persistent	No. of Persistent Entries
cabhCapMappingLanAddr	read-create	Yes[5]	16
cabhCapMappingLanPort	read-create	Yes[5]	16
cabhCapMappingMethod	read-only	–	N/A
cabhCapMappingProtocol	read-create	Yes[5]	16
cabhCapMappingRowStatus	read-create	Yes	16
cabhCapMappingNumPorts	read-create	Yes	16
cabhCapMappingRowDescr	read-create	Yes	16
cabhCapMappingCreateTime	read-only	No	N/A
cabhCapMappingLastUpdateTime	read-only	No	N/A
cabhCapMappingDuration	read-create	Yes	16
cabhCapMappingRemoteHostAddrType	read-only	No	N/A
CabhCapMappingRemoteHostAddr	read-only	No	N/A
CabhCapMappingEnable	Read-only	No	N/A
cabhCapPassthroughTable/ cabhCapPassthroughEntry			
cabhCapPassthroughIndex	not-accessible	–	N/A
cabhCapPassthroughMacAddr	read-create	Yes	16
cabhCapPassthroughRowStatus	read-create	Yes	16

[5] cabhCapMappingEntry objects are persistent if provisioned by the NMS and non-persistent if created dynamically based on outbound traffic. Refer to 8.3.4.4.

cabhCdpMib
cabhCdpObjects
cabhCdpBase

cabhCdpSetToFactory	read-write	No	N/A
cabhCdpLanTransCurCount	read-only	–	N/A
cabhCdpLanTransThreshold	read-write	No	N/A
cabhCdpLanTransAction	read-write	No	N/A
cabhCdpWanDataIpAddrCount	read-write	No	N/A
cabhCdpLastSetToFactory	read-only		N/A
cabhCdpTimeOffsetSelection	read-write	Yes	1
cabhCdpSnmpSetTimeOffset	read-write	Yes	1
cabhCdpDaylightSavingTimeEnable	read-write	Yes	1

cabhCdpAddr

cabhCdpLanAddrTable/ cabhCdpLanAddrEntry			
cabhCdpLanAddrIpType	not-accessible	–	N/A
cabhCdpLanAddrIp	not-accessible	–	N/A
cabhCdpLanAddrClientID	read-create	Yes	16

MIB NAME/Parameter	Max-Access	Persistent	No. of Persistent Entries
cabhCdpLanAddrLeaseCreateTime	read-only	–	N/A
cabhCdpLanAddrLeaseExpireTime	read-only	–	N/A
cabhCdpLanAddrMethod	read-only	Yes	16
cabhCdpLanAddrHostName	read-only	Yes	16
cabhCdpLanAddrRowStatus	read-create	Yes	16
cabhCdpWanDataAddrTable/ cabhCdpWanDataAddrEntry			
CabhCdpWanDataAddrIndex	not-accessible	–	N/A
CabhCdpWanDataAddrClientId	read-create	No	N/A
CabhCdpWanDataAddrIpType	read-only	–	N/A
CabhCdpWanDataAddrIp	read-only	–	N/A
CabhCdpWanDataAddrRowStatus	read-create	No	N/A
CabhCdpWanDataAddrLeaseCreateTime	read-only	–	N/A
CabhCdpWanDataAddrLeaseExpireTime	read-only	–	N/A
cabhCdpWanDnsServerTable/ cabhCdpWanDnsServerEntry			
cabhCdpWanDnsServerOrder	not-accessible	–	N/A
cabhCdpWanDnsServerIpType	read-only	–	N/A
cabhCdpWanDnsServerIp	read-only	–	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
cabhCdpServerUseCableDataNwDnsAddr	read-write	No	N/A
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

MIB NAME/Parameter	Max-Access	Persistent	No. of Persistent Entries
cabhCdpServerVendorSpecific	read-write	Yes	1
cabhCdpServerLeaseTime	read-write	Yes	1
cabhCdpServerDhcpAddressType	read-only	–	N/A
cabhCdpServerDhcpAddress	read-only	–	N/A
cabhCdpServerControl	read-write	No	N/A
cabhCdpServerCommitStatus	read-only	–	N/A
cabhCtpMib **cabhCtpObjects** **cabhCtpBase**			
cabhCtpSetToFactory	read-write	No	N/A
cabhCtpLastSetToFactory	read-only	–	N/A
cabhCtpConnSpeed			
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
cabhCtpConnPktsSent	read-only	–	N/A
cabhCtpConnPktsRecv	read-only	–	N/A
cabhCtpConnRTT	read-only	–	N/A
cabhCtpConnThroughput	read-only	–	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

MIB NAME/Parameter	Max-Access	Persistent	No. of Persistent Entries
cabhCtpPingNumSent	read-only	–	N/A
cabhCtpPingNumRecv	read-only	–	N/A
cabhCtpPingAvgRTT	read-only	–	N/A
cabhCtpPingMaxRTT	read-only	–	N/A
cabhCtpPingMinRTT	read-only	–	N/A
cabhCtpPingNumIcmpError	read-only	–	N/A
cabhCtpPingIcmpError	read-only	–	N/A
cabhQos2Mib **cabhQos2MibObjects** **cabhQos2Base**			
cabhQos2SetToFactory	read-write	No	N/A
cabhQos2LastSetToFactory	read-only	–	N/A
cabhQos2PsIfAttributes			
cabhQos2PsIfAttribTable/ *cabhQos2PsIfAttribEntry*			
cabhQos2PsIfAttribIfNumPriorities	read-only	–	N/A
cabhQosInterfaceAttribIfNumQueues	read-only	–	N/A
cabhQos2PolicyHolderObjects			
cabhQos2PolicyHolderEnabled	read-write	Yes	1
cabhQos2PolicyAdmissionControl	read-write	Yes	1
cabhQos2NumActivePolicyHolder	read-only	–	N/A
cabhQos2PolicyTable/ *cabhQos2PolicyEntry*			
cabhQos2PolicyOwner	not-accessible	–	N/A
cabhQos2PolicyOwnerRuleId	not-accessible	–	N/A
cabhQos2PolicyRuleOrder	read-create	Yes	32
cabhQos2PolicyAppDomain	read-create	Yes	32
cabhQos2PolicyAppName	read-create	Yes	32
cabhQos2PolicyServiceProvDomain	read-create	Yes	32
cabhQos2PolicyServiceName	read-create	Yes	32
cabhQos2PolicyPortDomain	read-create	Yes	32
cabhQos2PolicyPortNumber	read-create	Yes	32
cabhQos2PolicyIpProtocol	read-create	Yes	32
cabhQos2PolicyIpType	read-create	Yes	32
cabhQos2PolicySrcIp	read-create	Yes	32
cabhQos2PolicyDestIp	read-create	Yes	32
cabhQos2PolicySrcPort	read-create	Yes	32

MIB NAME/Parameter	Max-Access	Persistent	No. of Persistent Entries
cabhQos2PolicyDestPort	read-create	Yes	32
cabhQos2PolicyTraffImpNum	read-create	Yes	32
cabhQos2PolicyUserImportance	read-create	Yes	32
cabhQos2PolicyRowStatus	read-create	Yes	32
cabhQos2DeviceObjects			
cabhQos2TrafficClassTable/ cabhQos2TrafficClassEntry			
cabhQos2TrafficClassMethod	not-accessible	–	N/A
cabhQos2TrafficClassIdx	not-accessible	–	N/A
cabhQos2TrafficClassProtocol	read-create	–	N/A
cabhQos2TrafficClassIpType	read-create	–	N/A
cabhQos2TrafficClassSrcIp	read-create	–	N/A
cabhQos2TrafficClassDestIp	read-create	–	N/A
cabhQos2TrafficClassSrcPort	read-create	–	N/A
cabhQos2TrafficClassDestPort	read-create	–	N/A
cabhQos2TrafficClassImpNum	read-create	–	N/A
cabhQos2TrafficClassRowStatus	read-create	–	N/A
experimental			
snmpUSMDHObjectsMIB [RFC 2786]			
usmDHKeyObjects			
usmDHPublicObjects			
usmDHParamaters	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	–	N/A
usmDHKickstartMyPublic	read-only	–	N/A
usmDHKickstartMgrPublic	read-only	–	N/A
usmDHKickstartSecurityName	read-only	–	N/A

MIB NAME/Parameter	Max-Access	Persistent	No. of Persistent Entries
snmpV2 **snmpModules** **snmpMIB** **snmpMIBObjects** **snmpSet**			
snmpSetSerialNo	read-write	No	N/A
snmpFrameworkMIB [RFC 3411] **snmpEngine**			
snmpEngineID	read-only	Yes	1
snmpEngineBoots	read-only	Yes	1
snmpEngineTime	read-only	–	N/A
snmpEngineMaxMessageSize	read-only	–	N/A
snmpMPDMIB [RFC 3412] **snmpMPDObjects** **snmpMPDStats**			
snmpUnknownSecurityModels	read-only	–	N/A
snmpInvalidMsgs	read-only	–	N/A
snmpUnknownPDUHandlers	read-only	–	N/A
snmpTargetMIB [RFC 3413] **snmpTargetObjects**			
snmpTargetSpinLock	read-write	No	N/A
snmpTargetAddrTable/ *snmpTargetAddrEntry*			
snmpTargetAddrName	not-accessible	–	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	–	N/A
snmpTargetParamsMPModel	read-create	No	N/A

MIB NAME/Parameter	Max-Access	Persistent	No. of Persistent Entries
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
snmpUnknownContexts	read-only	–	N/A
snmpNotificationMIB [RFC 3413]			
snmpNotifyObjects			
snmpNotifyTable/			
snmpNotifyEntry			
snmpNotifyName	not-accessible	–	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	–	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

MIB NAME/Parameter	Max-Access	Persistent	No. of Persistent Entries
snmpUsmMIB [RFC 3414] **usmStats**			
usmStatsUnsupportedSecLevels	read-only	–	N/A
usmStatsNotInTimeWindows	read-only	–	N/A
usmStatsUnknownUserNames	read-only	–	N/A
usmStatsUnknownEngineIDs	read-only	–	N/A
usmStatsWrongDigests	read-only	–	N/A
usmStatsDecryptionErrors	read-only	–	N/A
usmUser			
usmUserSpinLock	read-write	No	N/A
usmUserTable/ *usmUserEntry*			
usmUserEngineID	not-accessible	–	N/A
usmUserName	not-accessible	–	N/A
usmUserSecurityName	read-only	–	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 3415] **snmpVacmMIB** **vacmMIBObjects**			
vacmContextTable/ *vacmContextEntry*			
vacmContextName	read-only	–	N/A
vacmSecurityToGroupTable/ *vacmSecurityToGroupEntry*			
vacmSecurityModel	not-accessible	–	N/A
vacmSecurityName	not-accessible	–	N/A

MIB NAME/Parameter	Max-Access	Persistent	No. of Persistent Entries
vacmGroupName	read-create	No	N/A
vacmSecurityToGroupStorageType	read-create	No	N/A
vacmSecurityToGroupStatus	read-create	No	N/A
vacmAccessTable/ vacmAccessEntry			
vacmAccessContextPrefix	not-accessible	–	N/A
vacmAccessSecurityModel	not-accessible	–	N/A
vacmAccessSecurityLevel	not-accessible	–	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	–	N/A
vacmViewTreeFamilySubtree	not-accessible	–	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 3584] **snmpCommunityMIBObjects**			
snmpCommunityTable/ snmpCommunityEntry			
snmpCommunityIndex	not-accessible	–	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

MIB NAME/Parameter	Max-Access	Persistent	No. of Persistent Entries
snmpTargetAddrExtTable/ snmpTargetAddrExtEntry			
snmpTargetAddrTMask	read-create	No	N/A
snmpTargetAddrMMS	read-create	No	N/A
clabSecCertObject			
clabSrvcPrvdrRootCACert	read-only	–	N/A
clabCVCRootCACert	read-only	–	N/A
clabCVCCACert	read-only	–	N/A
clabMfgCACert	read-only	–	N/A

Annex B

Format and content for event, SYSLOG and SNMP Trap

Table B.1 summarizes the format and content for local log event entries, SYSLOG messages, and SNMP Traps.

Each row in the table specifies an event that the PS must be capable of generating. These events are to be reported by the PS by any or all of the following three means: local event logging as implemented by the local event table in [RFC 2669], SYSLOG and SNMP Trap. The SYSLOG format is specified in 6.3.3.2.4.4 and SNMP Trap format is defined in this annex, following Table B.1.

The first and second columns of Table B.1 indicate in which stage the event happens. The third column indicates the priority assigned to the event. These priorities are the same as reported in the docsDevEvLevel object in [RFC 2669] and in the LEVEL field of a SYSLOG message.

The fourth column specifies the event text, which is reported in the docsDevEvText object of the [RFC 2669] and the text field of a SYSLOG message. The fifth column provides additional information about the event text of the fourth column. For example, some of the event text fields are constants and some event text fields include variable information. Some of the variables are only required in the SYSLOG, as described in the fifth column. The sixth column specifies the error code set.

The seventh column indicates an unique identification number for the event, which is assigned to the docsDevEvId object and the <eventId> field of a syslog message. The eighth column specifies the SNMP Trap, which notifies this event to a SNMP event receiver.

The rules to uniquely generate an event ID from the error code are described in 6.3.3.2.4.4. The event IDs in the table are in decimal format.

To better illustrate the table, the following is an example using the first row in the clause of Software Upgrade events.

The first and second columns are "SW Upgrade" and "SOFTWARE UPGRADE INIT". The event priority is "Notice". The event text is "Software Download INIT – Via NMS". The fifth column reads "For SYSLOG only, append: MAC addr: <P1> P1 = PS Mac Address". This is a note about the SYSLOG. That is to say, the syslog text body will be like "Software Download INIT – Via NMS – MAC addr: x1 x2 x3 x4 x5 x6".

The last column "Trap name" is cabhPsDevSwUpgradeInitTrap, the format for which is given at the end of this annex.

Table B.1/J.192 – Defined events for IPCable2Home

Process	Sub-process	PS priority	Event text	Message notes and details	Error code SET	EventID	Trap name
DHCP errors before provisioning complete							
Init	CDC	Critical	DHCP FAILED – Discover sent, no offer received		D01.0	68000100	
Init	CDC	Critical	DHCP FAILED – Request sent, No response		D02.0	68000200	
Init	CDC	Critical	DHCP FAILED – Requested Info not supported		D03.0	68000300	
Init	CDC	Error	DHCP ERROR – Response does not contain ALL the valid fields OR the PS is unable to determine provisioning mode		D03.1	68000301	
Init	CDC	Warning	DHCP ERROR – Unable to obtain all WAN-Data IP addresses the PS was configured to obtain		P02.0	68000302	cabhPsDevCdpWanDataIpTrap
ToD errors before provisioning complete							
Init	ToD	Warning	ToD Request sent – no response received Time Server address + <P1>	P1 = IP address of time of Day server	D04.1	68000401	cabhPsDevInitTrap
Init	ToD	Warning	ToD Response received – invalid data format Time Server address + <P1>	P1 = IP address of time of Day server	D04.2	68000402	cabhPsDevInitTrap

Table B.1/J.192 – Defined events for IPCable2Home

Process	Sub-process	PS priority	Event text	Message notes and details	Error code SET	EventID	Trap name
TFTP errors before provisioning complete							
Init	TFTP	Error	TFTP failed – Request sent – No Response		D05.0	68000500	cabhPsDevInitTrap (Trap is relevant for SNMP Prov Mode only.)
Init	TFTP	Error	TFTP failed – configuration file NOT FOUND	For SYSLOG only: append: File name = <P1> P1 = requested file name	D06.0	68000600	cabhPsDevInitTrap (Trap is relevant for SNMP Prov Mode only.)
Init	TFTP	Error	TFTP failed – Out of order packets		D07.0	68000700	cabhPsDevInitTrap (Trap is relevant for SNMP Prov Mode only.)
Init	TFTP	Error	TFTP file complete – but failed SHA-1 hash check	For SYSLOG only: append: File name = <P1> P1 = filename of TFTP file	D08.0	68000800	cabhPsDevInitTrap (Trap is relevant for SNMP Prov Mode only.)
Init	TFTP	Error	TFTP Failed Exceeded maximum number of retries	For SYSLOG only: append: Max allowed retries = <P1> P1 = maximum allowed number of retry attempts	D09.0	68000900	cabhPsDevInitTrap (Trap is relevant for SNMP Prov Mode only.)
TFTP success							
Init	TFTP	Notice	TFTP success		D10.0	68001000	

Table B.1/J.192 – Defined events for IPCable2Home

Process	Sub-process	PS priority	Event text	Message notes and details	Error code SET	EventID	Trap name
TLS							
Init	TCP/IP	Critical	PS failed to connect to HTTP/TLS server, when attempting <P1> config file download	P1 = 'PS' or 'Firewall'	D20.0	68002000	
Init	TLS	Critical	TLS Connection timed out and maximum number of retries exceeded, when attempting <P1> config file download	P1 = 'PS' or 'Firewall'	D21.0	68002100	
Init	TLS	Critical	TLS FATAL ERROR <P1>, when attempting <P2> config file download	P1= Error code from [RFC 2246] P2 = 'PS' or 'Firewall'	D22.0	68002200	
HTTP							
Init	HTTP	Critical	Configuration File Download failed, but will retry. HTTP Error. <P1>, when attempting <P2> config file download	P1= Status codes from [RFC 2616] P2 = 'PS' or 'Firewall'	D30.0	68003000	
Init	HTTP	Critical	Configuration file download failed. Due to connection time out and maximum number of retries. Operation aborted, when attempting <P1> config file download	P1 = 'PS' or 'Firewall'	D31.0	68003100	
Init	HTTP	Critical	Secure Configuration file download successfully completed, when attempting <P1> config file download	P1 = 'PS' or 'Firewall'	D32.0	68003200	

Table B.1/J.192 – Defined events for IPCable2Home

Process	Sub-process	PS priority	Event text	Message notes and details	Error code SET	EventID	Trap name
TLV parsing							
Init	TLV PARSING	Warning	TLV-27 or TLV-28 – unrecognized OID when attempting <P1> config file download	P1 = 'PS' or 'Firewall'	I401.0	73040100	cabhPsDevInitTLV UnknownTrap
Init	TLV PARSING	Warning	Unknown TLV	For SYSLOG only, append <P2> config file TLV is <P1>, P1 = the complete TLV in hexadecimal P2 = 'PS' or 'Firewall'	I401.1	73040101	cabhPsDevInitTLV UnknownTrap
Init	TLV PARSING	Error	Invalid TLV Format/content	For SYSLOG only, append: <P2> config file TLV is <P1>, P1+ = the complete TLV in hexadecimal P2 = 'PS' or 'Firewall'	I401.2	73040102	
Provisioning							
Init	Provisioning Complete	Notice	Provisioning complete	For SYSLOG only, append MAC Addr: <P1>. P1 = PS MAC address	I11.0	73001100	cabhPsDevInitTrap
SW UPGRADE INIT (Note)							
SW Upgrade	SW UPGRADE INIT	Notice	SW Download INIT – Via NMS	For SYSLOG only, append: SW file: <P1> – SW server: < P2>. P1 = SW file name and P2 = Tftp server IP address	E101.0	69010100	cabhPsDevSwUpgradeInitTrap
SW Upgrade	SW UPGRADE INIT	Notice	SW Download INIT – Via Config file <P1>	P1 = CM config file name For SYSLOG only, append: SW file: <P2> – SW server: < P3>. P2 = SW file name and P3 = Tftp server IP address	E102.0	69010200	cabhPsDevSwUpgradeInitTrap

Table B.1/J.192 – Defined events for IPCable2Home

Process	Sub-process	PS priority	Event text	Message notes and details	Error code SET	EventID	Trap name
SW UPGRADE GENERAL FAILURE (Note)							
SW Upgrade	SW UPGRADE GENERAL FAILURE	Error	SW Upgrade Failed during download – Max retry exceed (3)	For SYSLOG only, append: SW file: <P1> – SW server: < P2>. P1 = SW file name and P2 = Tftp server IP address	E103.0	69010300	cabhPsDevSwUpgradeFailTrap
SW Upgrade	SW UPGRADE GENERAL FAILURE	Error	SW Upgrade Failed Before Download – Server not Present	For SYSLOG only, append: SW file: <P1> – SW server: < P2>. P1 = SW file name and P2 = Tftp server IP address	E104.0	69010400	cabhPsDevSwUpgradeFailTrap
SW Upgrade	SW UPGRADE GENERAL FAILURE	Error	SW upgrade Failed before download – File not Present	For SYSLOG only, append: SW file: <P1> – SW server: < P2>. P1 = SW file name and P2 = TFTP server IP address	E105.0	69010500	cabhPsDevSwUpgradeFailTrap
SW Upgrade	SW UPGRADE GENERAL FAILURE	Error	SW upgrade Failed before download – TFTP Max Retry Exceeded	For SYSLOG only, append: SW file: <P1> – SW server: < P2>. P1 = SW file name and P2 = TFTP server IP address	E106.0	69010600	cabhPsDevSwUpgradeFailTrap
SW Upgrade	SW UPGRADE GENERAL FAILURE	Error	SW upgrade Failed after download – Incompatible SW file	For SYSLOG only, append: SW file: <P1> – SW server: < P2>. P1 = SW file name and P2 = Tftp server IP address	E107.0	69010700	cabhPsDevSwUpgradeFailTrap

Table B.1/J.192 – Defined events for IPCable2Home

Process	Sub-process	PS priority	Event text	Message notes and details	Error code SET	EventID	Trap name
SW Upgrade	SW UPGRADE GENERAL FAILURE	Error	SW upgrade Failed after download – SW File corruption	For SYSLOG only, append: SW file: <P1> – SW server: < P2>. P1 = SW file name and P2 = TFTP server IP address	E108.0	69010800	cabhPsDevSwUpgradeFailTrap
SW Upgrade	SW UPGRADE GENERAL FAILURE	Error	Disruption during SW download – Power Failure	For SYSLOG only, append: SW file: <P1> – SW server: < P2>. P1 = SW file name and P2 = Tftp server IP address	E109.0	69010900	cabhPsDevSwUpgradeFailTrap
SW UPGRADE SUCCESS (Note)							
SW Upgrade	SW UPGRADE SUCCESS	Notice	SW download Successful – Via NMS	For SYSLOG only, append: SW file: <P1> – SW server: < P2>. P1 = SW file name and P2 = Tftp server IP address	E111.0	69011100	cabhPsDevSwUpgradeSuccessTrap
SW Upgrade	SW UPGRADE SUCCESS	Notice	SW download Successful – Via Config file	For SYSLOG only, append: SW file: <P1> – SW server: < P2>. P1 = SW file name and P2 = Tftp server IP address	E112.0	69011200	cabhPsDevSwUpgradeSuccessTrap
DHCP failure after provisioning complete							
DHCP	CDC	Error	DHCP RENEW sent – No response		D101.0	68010100	cabhPsDevDHCPFailTrap
DHCP	CDC	Error	DHCP REBIND sent – No response		D102.0	68010200	cabhPsDevDHCPFailTrap
DHCP	CDC	Error	DHCP RENEW sent – Invalid DHCP option		D103.0	68010300	cabhPsDevDHCPFailTrap
DHCP	CDC	Error	DHCP REBIND sent – Invalid DHCP option		D104.0	68010400	cabhPsDevDHCPFailTrap

Table B.1/J.192 – Defined events for IPCable2Home

Process	Sub-process	PS priority	Event text	Message notes and details	Error code SET	EventID	Trap name
ToD failure after provisioning complete							
ToD	ToD	Warning	ToD Request sent – No response received		D04.3	68000403	cabhPsDevTODFail Trap
ToD	ToD	Warning	ToD Response received – Invalid data format		D04.4	68000404	cabhPsDevTODFail Trap
Verification of code file							
SW Upgrade	SW UPGRADE GENERAL FAILURE	Error	Improper code file controls	For SYSLOG only, append: Code File: <P1> – Code File Server: <P2>. P1= Code file name, P2 = code file server IP address	E201.0	69020100	cabhPsDevSwUpgradeFailTrap
SW Upgrade	SW UPGRADE GENERAL FAILURE	Error	Code File Manufacturer CVC Validation Failure	For SYSLOG only, append: Code File: <P1> – Code File Server: <P2>. P1= Code file name, P2 = code file server IP address	E202.0	69020200	cabhPsDevSwUpgradeFailTrap
SW Upgrade	SW UPGRADE GENERAL FAILURE	Error	Code File Manufacturer CVS Validation Failure	For SYSLOG only, append: Code File: <P1> – Code File Server: <P2>. P1= Code file name, P2 = code file server IP address	E203.0	69020300	cabhPsDevSwUpgradeFailTrap
SW Upgrade	SW UPGRADE GENERAL FAILURE	Error	Code File Co-Signer CVC Validation Failure	For SYSLOG only, append: Code File: <P1> – Code File Server: <P2>. P1= Code file name, P2 = code file server IP address	E204.0	69020400	cabhPsDevSwUpgradeFailTrap
SW Upgrade	SW UPGRADE GENERAL	Error	Code File Co-Signer CVS Validation Failure	For SYSLOG only, append: Code File: <P1> – Code File Server: <P2>. P1= Code file	E205.0	69020500	cabhPsDevSwUpgradeFailTrap

Table B.1/J.192 – Defined events for IPCable2Home

Process	Sub-process	PS priority	Event text	Message notes and details	Error code SET	EventID	Trap name
	FAILURE			name, P2 = code file server IP address			
Verification of CVC							
SW Upgrade	VERIFICA-TION OF CVC	Error	Improper Configuration File CVC Format – TFTP Server: <P1> – Config File: <P2>	P1 = TFTP Server IP Address P2 = Config File Name	E206.0	69020600	cabhPsDevSwUpgradeCVCFailTrap
SW Upgrade	VERIFICA-TION OF CVC	Error	Configuration File CVC Validation Failure – TFTP Server: <P1> – Config File: <P2>	P1 = TFTP Server IP Address P2 = Config File Name	E207.0	69020700	cabhPsDevSwUpgradeCVCFailTrap
SW Upgrade	VERIFICA-TION OF CVC	Error	Improper SNMP CVC Format – Snmp manager: <P1>	P1 = IP Address of SNMP Manager	E208.0	69020800	cabhPsDevSwUpgradeCVCFailTrap
SW Upgrade	VERIFICA-TION OF CVC	Error	SNMP CVC Validation Failure – Snmp manager: <P1>	P1 = IP Address of SNMP manager	E209.0	69020900	cabhPsDevSwUpgradeCVCFailTrap
CDP events							
CDP	CDS	Notice	Attempt to allocate more LAN TRANS IP addresses than allowed		P01.0	80000100	cabhPsDevCDPThresholdTrap
CDP	CDS	Notice	Unable to provision DHCP LAN client – IP address pool exhausted		P03.0	80000300	cabhPsDevCdpLanIpPoolTrap

Table B.1/J.192 – Defined events for IPCable2Home

Process	Sub-process	PS priority	Event text	Message notes and details	Error code SET	EventID	Trap name
CSP events							
CSP	Firewall	Notice	Change in state for cabhSec2FwEventEnable for Type 1. The new value is <P1>	P1 = value of cabhSec2FwEventTypeEnable for Type 1	P101.1	80010101	cabhPsDevCSPTrap
CSP	Firewall	Notice	Change in state for cabhSec2FwEventEnable for Type 2. The new value is <P1>	P1 = value of cabhSec2FwEventTypeEnable for Type 2	P101.2	80010102	cabhPsDevCSPTrap
CSP	Firewall	Notice	Change in state for cabhSec2FwEventEnable for Type 3. The new value is <P1>	P1 = value of cabhSec2FwEventTypeEnable for Type 3	P101.3	80010103	cabhPsDevCSPTrap
CSP	Firewall	Notice	Change in state for cabhSec2FwEventEnable for Type 4. The new value is <P1>	P1 = value of cabhSec2FwEventTypeEnable for Type 4	P101.4	80010104	cabhPsDevCSPTrap
CSP	Firewall	Notice	Change in state for cabhSec2FwEventEnable for Type 5. The new value is <P1>	P1 = value of cabhSec2FwEventTypeEnable for Type 5	P101.5	80010105	cabhPsDevCSPTrap
CSP	Firewall	Notice	Change in state for cabhSec2FwEventEnable for Type 6. The new value is <P1>	P1 = value of cabhSec2FwEventTypeEnable for Type 6	P101.6	80010106	cabhPsDevCSPTrap
CSP	Firewall	Warning	Firewall Type 1 event threshold reached		P102.1	80010201	cabhPsDevCSPTrap
CSP	Firewall	Warning	Firewall Type 2 event threshold reached		P102.2	80010202	cabhPsDevCSPTrap
CSP	Firewall	Warning	Firewall Type 3 event threshold reached		P102.3	80010203	cabhPsDevCSPTrap

Table B.1/J.192 – Defined events for IPCable2Home

Process	Sub-process	PS priority	Event text	Message notes and details	Error code SET	EventID	Trap name
CSP	Firewall	Warning	Firewall Type 4 event threshold reached Set of <P1> failed. <P2>	P1 = MIB object attempted to be changed (e.g., "cabhSec2FwPolicyFileURL") P2 = Textual description of failure	P102.4	80010204	cabhPsDevCSPTrap
CSP	Firewall	Warning	Firewall Type 5 event threshold reached		P102.5	80010205	cabhPsDevCSPTrap
CSP	Firewall	Warning	Firewall Type 6 event threshold reached		P102.6	80010206	cabhPsDevCSPTrap
CSP	Firewall TFTP	Critical	TFTP download of firewall policy file failed: request sent, no response. Policy file URL: <P1>	P1 = requested firewall policy file URL	P130.0	80013000	cabhPsDevCSPTrap
CSP	Firewall TFTP	Critical	TFTP failed – firewall policy file not found. Policy file URL: <P1>	P1 = requested firewall policy file URL	P131.0	80013100	cabhPsDevCSPTrap
CSP	Firewall TFTP	Critical	TFTP failed – invalid firewall policy file. Policy file URL: <P1>	P1 = requested firewall policy file URL	P132.0	80013200	cabhPsDevCSPTrap

Table B.1/J.192 – Defined events for IPCable2Home

Process	Sub-process	PS priority	Event text	Message notes and details	Error code SET	EventID	Trap name
CSP	Firewall TFTP	Critical	Firewall policy file download complete but failed SHA-1 has check. Policy file URL: <P1> Hash: <P2>	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. Policy file URL: <P1>	P1 = requested firewall policy file URL	P134.0	80013400	cabhPsDevCSPTrap
CSP	Firewall TFTP	Notice	Firewall policy file TFTP download success. Policy file URL: <P1>	P1 = requested firewall policy file URL For SYSLOG only: append: Max allowed retries = <P2> P2 = maximum allowed number of retry attempts	P135.0	80013500	cabhPsDevCSPTrap
CAP events							
CAP	C-NAT	Warning	CAP unable to make C-NAT mapping. No WAN-data IP address available		P201.0	80020100	cabhPsDevCAPTrap
CAP	C-NAPT	Warning	CAP unable to make C-NAPT mapping. No WAN IP address available		P250.0	80025000	cabhPsDevCAPTrap

Table B.1/J.192 – Defined events for IPCable2Home

Process	Sub-process	PS priority	Event text	Message notes and details	Error code SET	EventID	Trap name
CTP events							
CTP	Connection Speed Tool	Notice	Connection Speed Tool test completed successfully. Source IP: <P1>. Dest IP: <P2>. Protocol: <P3>. Throughput: <P4>.	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. Source IP: <P1>. Dest IP: <P2>. Protocol: <P3>. Timer value: <P4> ms.	P1 = IP address of source P2 = IP address of destination P3 = protocol (value of cabhCtpConnProto) P4 = value of the timer measuring the execution time of the Connection Speed Tool (millisec) (Reference: Connection Speed Tool Function Requirements clause)	P302.0	80030200	cabhPsDevCtpTrap
CTP	Connection Speed Tool	Notice	Connection Speed Tool test aborted. Source IP: <P1>. Dest IP: <P2>. Protocol: <P3>. Timer value: <P4> ms	P1 = IP address of source P2 = IP address of destination P3 = protocol (value of cabhCtpConnProto) P4 = value of the timer measuring the execution time of the Connection Speed Tool (ms) (Reference: Connection Speed Tool Function Requirements clause)	P303.0	80030300	cabhPsDevCtpTrap

Table B.1/J.192 – Defined events for IPCable2Home

Process	Sub-process	PS priority	Event text	Message notes and details	Error code SET	EventID	Trap name
CTP	Ping Tool	Notice	Ping Tool test completed successfully. Source IP: <P1>. Dest IP: <P2>. Ave. RTT: <P3> ms.	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. Source IP: <P1>. Dest IP: <P2>. Number of requests: <P3>. Number of responses: <P4>	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. Source IP: <P1>. Dest IP: <P2>. Number of requests: <P3>. Number of responses: <P4>	P1 = IP address of source P2 = IP address of destination P3 = number of requests sent P4 = number of responses received	P322.0	80032200	cabhPsDevCtpTrap
QoS events							
UPnP Discovery	M-Search	Warning	Multiple UPnP Policy Holders active		Q100.0	81010000	cabhPsDevUpnpMultiplePHTrap
NOTE – Software upgrade (secure software download) events apply to standalone Portal Services only. Software upgrade is controlled by the DOCSIS 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 11.8, Software Download for the PS.							

B.1 Trap descriptions

All specified traps are defined in the PS DEV MIB specification, [see E.4].

Annex C

Security threats and preventative measures

When developing a security technology, it is important to understand what the primary threats are for a given application or environment. This information can then be used to select the most effective security tools and technologies for protection and prevention against malicious attacks.

The following primary home networking security threats to subscribers and system operators have been identified:

C.1 Theft of Service: Theft of service comes in two forms; unauthorized access to cable services and unauthorized duplication of service content.

Unauthorized access involves a subscriber or 3rd party (such as a neighbour) having access to cable services for which they have not paid. Devices could be "cloned" or modified to appear as a qualified device on the subscriber's home network. This could also degrade service delivery performance as these devices consume additional transport resources on the HFC and home networks.

Unauthorized duplication usually involves a subscriber or 3rd party (such as a neighbour) making illegal copies of service content. In some cases, these copies are distributed to other consumers without the approval of the operator or content provider.

C.2 Denial of Service (DoS) attacks: Denial of service attacks can occur when a 3rd party entity (attacker, disgruntled customer, etc.) disrupts the normal communication and delivery of services between operators and their subscribers. Offending data transmissions coming from what appears to be a valid device/source could be injected into the home network and severely degrade its normal functions. These offending data transmissions could also extend to the operator's HFC network causing performance problems there.

C.3 Service confidentiality: The service confidentiality threat involves a 3rd party (neighbours, attacker, etc.) monitoring/receiving information about a subscriber and the services they use. This could result in passwords or device configuration information being stolen, allowing attackers to gain further access to a subscriber's network resources and confidential files/data.

There are a number of different methods that can be used to prevent the home network security threats mentioned above. Unfortunately, one method cannot prevent them all, but a combination may be the best line of defense. The following preventative measures can be used:

C.4 Authentication: Authentication involves the verification that the sending and receiving entities are as claimed. This includes the service source, the receiving device, and the subscriber.

Authentication helps prevent theft of service by validating end devices and users, but it does not prevent content from being illegally copied, or prevent unauthorized access by 3rd parties who are monitoring the link. It does do a good job at preventing DoS attacks because traffic can be rejected if it does not come from a valid source. By itself, authentication does not provide any service confidentiality support, encryption must be used.

C.5 Copy protection: Copy protection methods limit the ability of a receiving device to make unauthorized copies of service content.

Copy protection helps prevent theft of service by limiting how many copies can be made, but it does not prevent unauthorized access to services. It also does not prevent DoS or service confidentiality protection. In general, this preventive measure is implemented at higher application layers.

C.6 Data encryption: Data encryption prevents the unauthorized disclosure/access of data.

Data encryption does an excellent job at providing data confidentiality and protection against theft of service. Encryption prevents making data unable to read without the correct decrypting key. However, it does not validate the source/receiving entities and it does not provide copy protection after the data has been decrypted. It also does not prevent DoS attacks.

C.7 Firewall: Firewall applications prevent network traffic from passing from one domain to another, unless it meets certain criteria set by the subscriber or operator. In home networks, firewalls are typically located on residential gateway devices that connect the HFC network to the home network.

A firewall application helps prevent DoS attacks and confidentiality attacks from the wide area network (WAN) side of the firewall, but it does not prevent these kind of attacks coming from the home network side of the firewall. It also does not provide theft of service protection.

C.8 Management message security: This method of prevention involves authentication and encryption of network management messages only. Network management messages are used for device configuration, network monitoring/control, service provisioning, and Quality of Service (QoS) reservations.

Management message security provides a good mechanism to prevent DoS attacks by authenticating and encrypting management messages. Subscriber's personal and network configuration information is also protected from confidentiality attacks, but service content is not. Also, management message security does not prevent theft of service content by unauthorized entities.

Annex D

Applications through CAT and firewall

In the normal operation of address translation and firewall functionality, a number of protocols and applications may be prohibited from working as expected. Firewalls may purposely filter out certain applications and protocols for security purposes. The firewall policy can be explicitly set by the cable operator to allow as many ports to be opened as needed by the customer without opening ports that are not needed for communication between the LAN and WAN. Limiting the open ports and session initiation between the LAN and WAN may provide protection to the home LAN from attacks. If the ports are not allowed to be opened by the firewall policy, an attacker cannot use these ports to attack the LAN. The purpose of this annex is to provide a minimum level of support for commonly used applications under specific scenarios, and to assist the cable operator with common port configuration.

[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 run within a IPCable2Home environment adhere to these guidelines.

The existence of NAT and Firewall functionality are known to disrupt a number of protocols and applications when the end nodes/hosts are not in the same address realm and must traverse an IP Network Address Translator (NAT/CAT) and/or Firewall *en route* to bridge the realms. In many cases, the CAT and Firewall cannot provide the application and protocol transparency desired without the assistance of an Application Layer Gateway (ALG). This Recommendation assumes an ALG is implemented in the Residential Gateway that enables applications listed within this annex to work through the CAT.

Applications through the firewall are described in terms of protocol, specific port numbers, LAN-WAN relationship scenarios and addressing realms. The protocols are divided into two tables: one table is to list the protocols which can be managed by policy alone and is labelled Applications Requiring Firewall Policy Exclusively; the second table is to list the protocols which can only be managed with the combination of policy and ALGs and is labelled Applications Requiring Firewall Policy and an ALG.

According to the policy specified within clause 11, the tables contain information comments for the reader to be able to map the required applications to those with particular policy requirements for IPCable2Home and IPCablecom. IPCable2Home requires factory default settings for the ports to be opened through the firewall for normal Residential gateway operations. The items marked with IPCablecom in the comments column will be included, in addition to the factory defaults in enable IPCablecom through the firewall. The firewall settings to enable IPCablecom are listed in the comments column of each table and are specified within clause 11 in the configuration file clause.

In addition to the specified applications, the PS SHOULD support online gaming applications through the CAT and firewall. Online gaming is considered a typical user application. However, this Recommendation does not specify games, as gaming is a dynamic industry and the online game ports depend upon the current popularity of particular games.

D.1 Relationship scenarios

The specific scenarios may define the number of hosts communicating with each other through the PS, along with the requirements for each protocol and application. Each application/protocol and specific scenario requires support of the CH CAT and firewall to function correctly. The scenarios include an "xxx to xxx" definition that indicates the number of LAN hosts communicating to WAN hosts (e.g., "One-to-Many" defines One LAN host communicating with Many WAN hosts concurrently.). These scenarios include:

- "One-to-One" relationship for a single instance.
- "One-to-One" relationship for multiple instances (the number of required instances may be identified).
- "One-to-Many" relationship for a single instance.
- "One-to-Many" relationship for multiple instances (the number of required instances may be identified).
- "Many-to-One" relationship for a single instance.
- "Many-to-One" relationship for multiple instances (the number of required instances will be identified if necessary).

NOTE – The "Many-to-Many" scenario will be the same as a "One-to-One" relationship for multiple instances, a "One-to-Many" relationship for multiple instances, and/or a "Many-to-One" relationship for multiple instances.

See Figures D.1 to D.3.

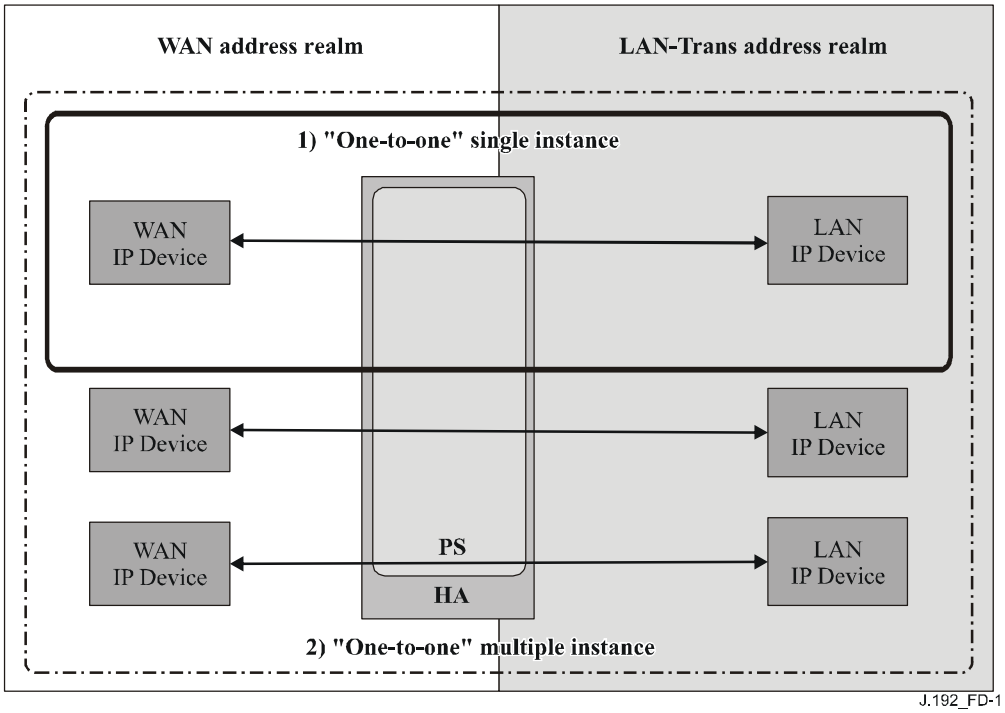


Figure D.1/J.192 – "One-to-One" scenarios

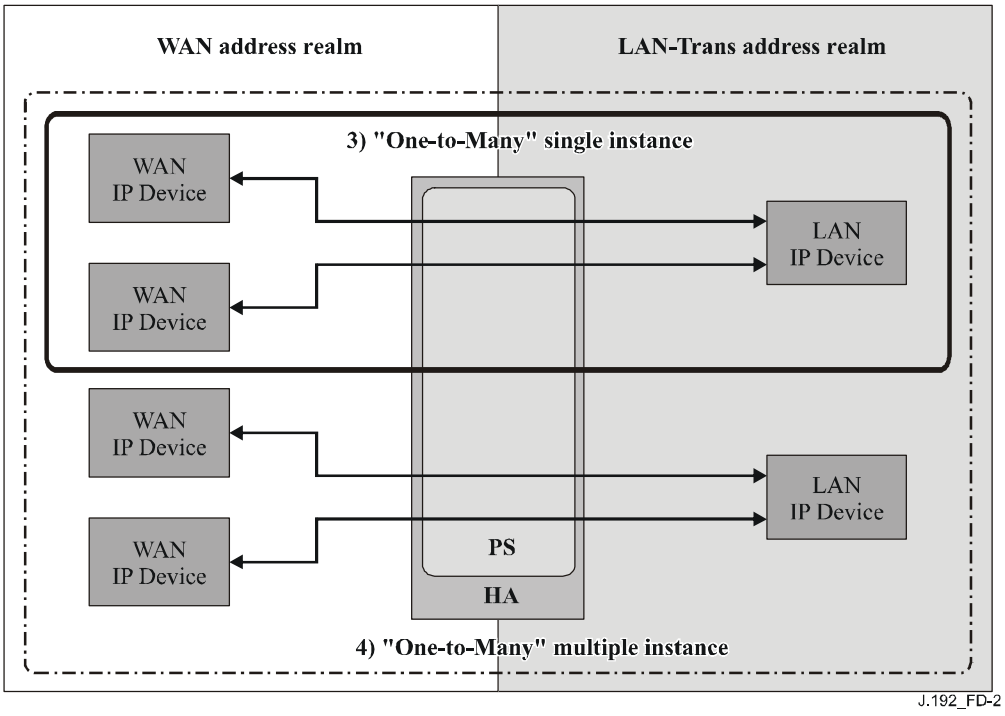


Figure D.2/J.192 – "One-to-Many" scenarios

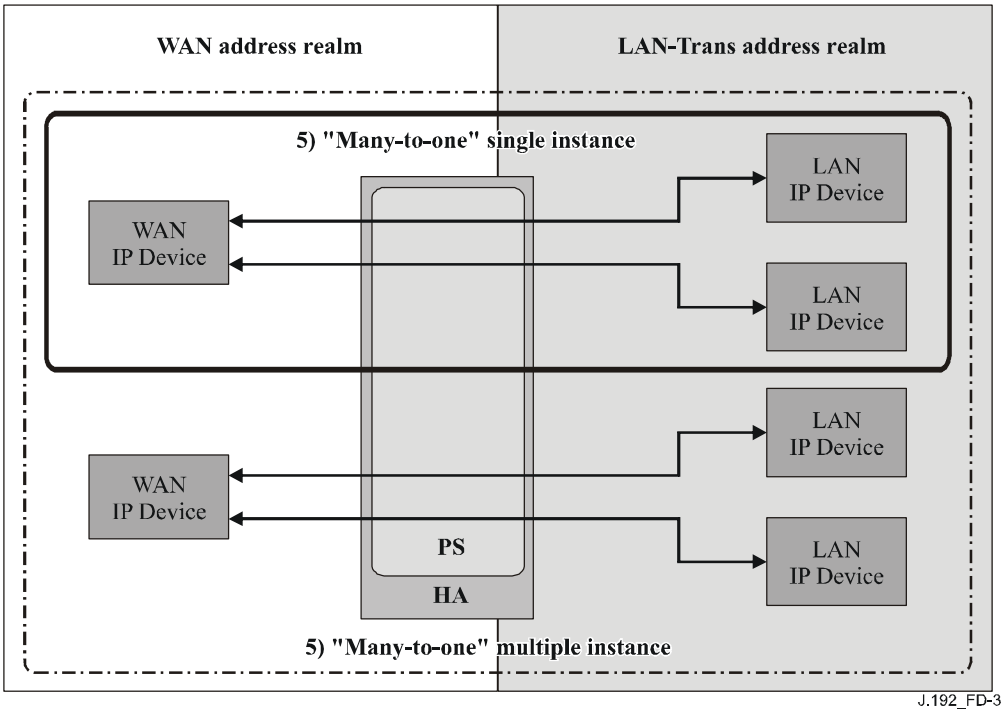


Figure D.3/J.192 – "Many-to-One" scenarios

D.2 Applications requiring firewall policy exclusively

Tables D.1 and D.2 identify the applications and protocols that MUST be supported through the CAT and Firewall. This does not preclude the support of additional applications and protocols. A CAT/Firewall that can support these applications and protocols will be able to support most other applications and protocols that do not embed address, port, or other information affected by network address translation, and do not negotiate inbound sessions.

The following list of protocols and applications in Table D.1 MUST work through CAT and Firewall implementations. The firewall MUST NOT begin operations until after the provisioning complete message is sent by the PS, therefore the protocols needed for the PS to provision are not noted in this table.

NOTE – Applications only requiring Firewall policy configuration exclusively MUST be supported in all six (6) relationship scenarios unless noted in the comments column.

Table D.1/J.192 – Protocols required to work through CAT and CH firewall

Application/Protocol	Ports	Comments
AOL IM	TCP/5190, 5191, 5192, 5193 & 13784	Internet Default
CU-SeeMe	TCP/7648, 7649; UDP/7648, 7649, 24032	
DHCP		Internet Default
DNS	UDP/53	IPCablecom and IPCable2Home
FTPS	989 & 990	
HTTP	TCP/80	Internet Default
HTTPS	TCP/443	Internet Default
IGMP and IP Multicast		CH 1.0 Annex requirement

Table D.1/J.192 – Protocols required to work through CAT and CH firewall

Application/Protocol	Ports	Comments
imap	143	
imap3	220	
IPSec	IKE > UDP/500 – ESP > raw IP/50	IKE key exchange, Tunnel mode, one-to-one single instance (CAT support key) IKE key exchange, Transport mode, one-to-one single instance (Passthrough mode) IPCablecom & LAN Peer Passthrough mode
IRC	TCP/6665-6669	
Kerberos	1293	IPCablecom and IPCable2Home PS Address Realm
L2TP	UDP/1701	
MediaPlayer (Windows)	TCP/80; 1755	
Microsoft Messenger	3330 – 3332	Internet Default mcs-calypsoicf 3330 mcs-messaging 3331 mcs-mailsvr 3332
MGCP	2427, 2727	IPCablecom
Peer-to-Peer (eDonkey)	TCP/4662 UDP/4665	eDonkey
Peer-to-Peer (FastTrack P2P Protocol)	TCP/1214	KaZaA, Grokster, etc.
Peer-to-Peer (Gnutella P2P Protocol)	TCP/6346	Gnutella, LimeWire, BearShare, Morpheus, etc.
Peer-to-Peer (WinMX)	TCP/6699 UDP/6257	WinMX
PING ICMP Echo Request	raw IP/1	IPCable2Home
POP3	TCP/110	Internet Default
PPTP	Control Port > TCP/1723 & GRE > raw IP/47	
RealAudio/RealMedia	TCP: 80;443;554	
RSVP		IPCablecom
RTSP	TCP/554	
RTCP		IPCablecom
RTP		IPCablecom
SMTP	TCP/25	Internet Default
SNMP	TCP/161 UDP/161	IPCable2Home PS Address Realm and IPCablecom

Table D.1/J.192 – Protocols required to work through CAT and CH firewall

Application/Protocol	Ports	Comments
SNMP trap	TCP/162 UDP/162	IPCable2Home PS Address Realm and IPCablecom
SSH	TCP/22 UDP/22	Internet Default
Syslog	UDP/514	IPCable2Home PS Address Realm and IPCablecom
Telnet	UDP/23	Outbound session requests. Internet Default
TFTP	UDP/69	IPCablecom
Traceroute	raw IP/1	Internet Default Reply from all hops between source and destination must be supported
Yahoo Messenger	TCP: 5050, 80 or any available	Internet Default

NOTE – Some port numbers listed in this clause were previously unassigned by IANA, but have been recently assigned and now belong to another application. RTP & Quicktime both list 6970 – 6999, IANA has now assigned 6998 & 6999 to iatp-highpri and iatp-normalpri. IPCable2Home makes no attempt to correct this conflict.

D.3 Application requiring firewall policy and an ALG

There are many cases in which the CAT and Firewall cannot provide the application and protocol transparency desired. Since CAT modifies end node addresses (within the IP header of a packet) *en route*, some applications are unable to function through the CAT without the assistance of an ALG. Where possible, application specific ALGs MUST be used in conjunction with CAT and the appropriate Firewall policy to provide the desired application level transparency. The function of an ALG is application specific, so a list of applications, protocols and the scenarios that MUST be supported is found below.

Table D.2/J.192 – Apps requiring firewall policy and an ALG

Application/ Protocol	Ports	(1) One-to-one single	(2) One-to-one multi	(3) One-to-many single	(4) One-to-many multi	(5) Many-to-one single	(6) Many-to-one multi	Comments
FTP	20/tcp, 21/tcp	X	X	X	X	X	X	
Microsoft Netmeeting (H.323)	TCP/389 ILS 522 ULS 1503 T.120 1720 call set-up 1731 audio call ctrl Dynamic TCP call control Dynamic UDP 1024-65535 RTP over UDP	X	X	X	X	X	X	
MSN Messenger (H.323)	1863/tcp	X	X	X	X	X	X	Internet Default
Net2Phone	6801/udp (also calls for opening 2 additional unspecified ports) UDPPORT=6801 UDPPORT=XXXX TCPPORT=XXXX The Network Administrator needs to make sure UDPPORT 6801 is open. For the other UDPPORT and TCPPORT, the administrator can use anything in the range from 1 – 30000.)	X	X	X	X			
Quicktime 5	RTSP/TCP/554 RTP/UDP 6970-6999	X	X	X	X	X	X	Supporting Quicktime without an ALG via port 80 provides less than optimal performance
Window Messenger (SIP)		X	X					Available on Windows XP only

Annex E

MIBs

E.1 IPCable2Home Address Portal (CAP) MIB requirement

Requirements

The IPCable2Home CAP MIB MUST be implemented as defined below.

```
CABH-CAP-MIB DEFINITIONS ::= BEGIN
IMPORTS
    MODULE-IDENTITY,
    OBJECT-TYPE,
    Unsigned32,
    Integer32           FROM SNMPv2-SMI
    TimeStamp,
    TruthValue,
    RowStatus,
    DateAndTime,
    PhysAddress         FROM SNMPv2-TC
    OBJECT-GROUP,
    MODULE-COMPLIANCE   FROM SNMPv2-CONF
    InetAddressType,
    InetAddress,
    InetPortNumber      FROM INET-ADDRESS-MIB
    clabProjCableHome   FROM CLAB-DEF-MIB
    SnmpAdminString     FROM SNMP-FRAMEWORK-MIB;

cabhCapMib MODULE-IDENTITY
    LAST-UPDATED    "200502110000Z" --February 11, 2005
    ORGANIZATION    "CableLabs Broadband Access Department"
    CONTACT-INFO
            "Kevin Luehrs
            Postal: Cable Television Laboratories, Inc.
            858 Coal Creek Circle
            Louisville, Colorado 80027
            U.S.A.
            Phone:  +1 303-661-9100
            Fax:    +1 303-661-9199
            E-mail: k.luehrs@cablelabs.com; mibs@cablelabs.com"
    DESCRIPTION
            "This MIB module supplies the basic management objects
            for the CableHome Address Portal (CAP) portion of
            the PS."
    ::=  { clabProjCableHome 3 }


    cabhCapObjects      OBJECT IDENTIFIER ::= { cabhCapMib 1 }
    cabhCapBase         OBJECT IDENTIFIER ::= { cabhCapObjects 1 }
    cabhCapMap          OBJECT IDENTIFIER ::= { cabhCapObjects 2 }


--========================================================================
--
--    General CAP Parameters
--
--========================================================================

cabhCapTcpTimeWait OBJECT-TYPE
    SYNTAX      Unsigned32
    UNITS       "seconds"
    MAX-ACCESS  read-write
```

```
    STATUS      current
    DESCRIPTION
            "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 state referred to in [RFC 793]."
    REFERENCE
            "CableHome 1.1 Specification, Packet Handling & Address
            Translation section."
    DEFVAL { 300 }
    ::= { cabhCapBase 1 }

cabhCapUdpTimeWait OBJECT-TYPE
    SYNTAX      Unsigned32
    UNITS       "seconds"
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
            "The inactivity time to wait before destroying
            CAP mappings for UDP."
    REFERENCE
            "CableHome 1.1 Specification, Packet Handling & Address
            Translation section."
    DEFVAL { 300 }  -- 5 minutes
    ::={ cabhCapBase 2 }

cabhCapIcmpTimeWait OBJECT-TYPE
    SYNTAX      Unsigned32
    UNITS       "seconds"
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
            "The inactivity time to wait before destroying
            CAP mappings for ICMP."
    REFERENCE
            "CableHome 1.1 Specification, Packet Handling & Address
            Translation section."
    DEFVAL { 300 }  --  5 minutes
    ::= { cabhCapBase 3 }

cabhCapPrimaryMode OBJECT-TYPE
    SYNTAX      INTEGER {
                    napt(1),        -- NAT with Port Translation Mode
                    nat(2),         -- Traditional NAT Mode
                    passthrough(3)  -- Passthrough/Bridging Mode
                }
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
            "The Primary Packet-handling Mode of the Portal Services
            logical element (PS) of a CableHome compliant residential
            gateway device. This object configures operation of the PS
            packet handling functions.

            When the value of this object is napt(1), the PS is
            required to support the Network Address and Port
            Translation (NAPT) process in accordance with the NAPT
            requirements defined in IETF RFC 3022. When operating in
            NAPT Primary Packet Handling Mode, the PS supports the
            translation of multiple LAN-Trans IP addresses and their
            TCP/UDP ports into a single WAN-Data IP address and its
            TCP/UDP ports.

            When the value of this object is nat(2), the PS is required
            to support the Network Address Translation (NAT) process in
```

```
            accordance with the NAT equirements defined in IETF RFC
            3022. When operating in NAT Primary Packet Handling Mode,
            the PS supports the translation of multiple LAN-Trans IP
            addresses into the same number of unique WAN-Data IP
            addresses.

            When the value of this object is passthrough(3), the PS is
            required to act as a transparent bridge in accordance with
            IEEE 802.1D. When operating in Passthrough Primary Packet
            Handling Mode, the PS does not translate network addresses,
            and bridges all traffic between its LAN and WAN interfaces.

            The PS MUST delete dynamically-created row entries from
            the cabhCapMappingTable, i.e., those with
            cabhCapMappingMethod = dynamic(2), when the value of
            cabhCapPrimaryMode changes. The PS MUST NOT delete
            statically-created row entries from the cabhCapMappingTable
            where cabhCapMappingMethod = static(1), when the value of
            cabhCapPrimaryMode changes."
    REFERENCE
            "CableHome 1.1 Specification, Packet Handling & Address
            Translation section."
    DEFVAL { napt }
    ::= { cabhCapBase 4 }

cabhCapSetToFactory OBJECT-TYPE
    SYNTAX      TruthValue
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
            "Reading this object always returns false(2). When the
            cabhCapSetToFactory object is set to true(1), the PS must
            take the following actions:

            1) Clear all entries in the cabhCapMappingTable and
               cabhCapPassthroughTable.
            2) Reset the following objects to their factory default
               values:
                 cabhCapTcpTimeWait,
                 cabhCapUdpTimeWait,
                 cabhCapIcmpTimeWait,
                 cabhCapPrimaryMode"
    REFERENCE
            "CableHome 1.1 Specification, Packet Handling & Address
            Translation section."
    ::= { cabhCapBase 5 }

cabhCapLastSetToFactory OBJECT-TYPE
    SYNTAX      TimeStamp
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "The value of sysUpTime when cabhCapSetToFactory was
            last set to true. Zero if never reset."
    ::= { cabhCapBase 6 }

cabhCapUpnpPortForwardingEnable OBJECT-TYPE
    SYNTAX      TruthValue
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
            "This MIB is effective only when the PS is performing NAPT.
            If this MIB object is set to false(2), the PS MUST disable
            the UPnP WANIpConnection service in the CableHome PS. If
```

```
            this MIB object is set to true(1), the PS MUST enable the
            WANIpConnection service in the PS. When the primary packet
            handling mode of the PS is C-NAT (2) or Passthrough(3),
            setting this MIB to true(1) MUST return InconsistentValue
            error."
    REFERENCE
            "CableHome 1.1 Specification, Packet Handling & Address
            Translation section."
    DEFVAL { 1 }
    ::= { cabhCapBase 7 }

cabhCapUpnpTimeWait OBJECT-TYPE
    SYNTAX      Unsigned32
    UNITS       "seconds"
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
            "The inactivity time to wait before destroying
            CAP mappings created by UPnP control points. The
            value of 0 indicates inactivity time wait of
            infinity, i.e., a UPnP entry does not get destroyed
            based on inactivity period."
    REFERENCE
            "CableHome 1.1 Specification, Packet Handling & Address
            Translation section."
    DEFVAL { 0 }  --  0 seconds, inactivity time wait of infinity.
    ::= { cabhCapBase 8 }


--========================================================================
--
--    cabhCapMappingTable (CAP Mapping Table)
--
--    The cabhCapMappingTable contains information pertaining to all
--    NAPT and NAT mappings in a CableHome(TM) compliant residential
--    gateway device.
--
--========================================================================


cabhCapMappingTable OBJECT-TYPE
    SYNTAX      SEQUENCE OF CabhCapMappingEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "This table contains IP address mappings between private
            network addresses, or network addresses and port
            numbers/ICMP Identifiers, assigned to devices on the
            subscriber's home LAN, and network addresses, or network
            addresses and port numbers/ICMP Identifiers on the WAN,
            presumed to be on a separate subnetwork than the private
            IP addresses.  The CAP Mapping Table is used by the
            CableHome Address Portal (CAP) function of the PS to make
            packet forwarding decisions."
    REFERENCE
            "CableHome 1.1 Specification, Packet Handling & Address
            Translation section."
    ::= { cabhCapMap 1 }

cabhCapMappingEntry OBJECT-TYPE
    SYNTAX      CabhCapMappingEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
```

```
            "List of the private IP (LAN) address-to-cable
            operator assigned IP (WAN) address mappings stored
            in the PS and used by the PS to make packet
            forwarding decisions."
    INDEX { cabhCapMappingIndex }
    ::= { cabhCapMappingTable 1 }

CabhCapMappingEntry ::= SEQUENCE {
    cabhCapMappingIndex               INTEGER,
    cabhCapMappingWanAddrType         InetAddressType,
    cabhCapMappingWanAddr             InetAddress,
    cabhCapMappingWanPort             InetPortNumber,
    cabhCapMappingLanAddrType         InetAddressType,
    cabhCapMappingLanAddr             InetAddress,
    cabhCapMappingLanPort             InetPortNumber,
    cabhCapMappingMethod              INTEGER,
    cabhCapMappingProtocol            INTEGER,
    cabhCapMappingRowStatus           RowStatus,
    cabhCapMappingNumPorts            Unsigned32,
    cabhCapMappingRowDescr            SnmpAdminString,
    cabhCapMappingCreateTime          DateAndTime,
    cabhCapMappingLastUpdateTime      DateAndTime,
    cabhCapMappingDuration            Integer32,
    cabhCapMappingRemoteHostAddrType InetAddressType,
    cabhCapMappingRemoteHostAddr      InetAddress,
    cabhCapMappingEnable              TruthValue
    }

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  read-create
    STATUS      current
    DESCRIPTION
            "The IP address type assigned on the WAN side."
    DEFVAL { ipv4 }
    ::= { cabhCapMappingEntry 2 }

cabhCapMappingWanAddr OBJECT-TYPE
    SYNTAX      InetAddress
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
            "The IP address assigned by the cable operator's address
            (DHCP) server, and comprising the WAN-side IP address
            of the CAP Mapping tuple. This object is populated
            either dynamically by LAN-to-WAN outbound traffic or
            statically by the cable operator."
    ::= { cabhCapMappingEntry 3 }

cabhCapMappingWanPort OBJECT-TYPE
    SYNTAX      InetPortNumber
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
            "The TCP/UDP port number or ICMP Identifier
            on the WAN side. A port number/Identifier of
```

```
            0 indicates either a NAT or a DMZ mapping.
            A non-zero port number/Identifier indicates
            a NAPT mapping. If the value of
            cabhCapMappingNumPorts MIB object is non-zero,
            this MIB represents a starting TCP/UDP port
            number on the WAN side for which a mapping
            entry is created."
    DEFVAL { 0 }
    ::= { cabhCapMappingEntry 4 }

cabhCapMappingLanAddrType OBJECT-TYPE
    SYNTAX      InetAddressType
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
            "The IP address type assigned on the LAN side."
    DEFVAL { ipv4 }
    ::= { cabhCapMappingEntry 5 }

cabhCapMappingLanAddr OBJECT-TYPE
    SYNTAX      InetAddress
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
            "The IP address of the LAN-Trans IP Device. This object is
            populated either dynamically as a result of LAN-to-WAN
            outbound traffic or statically by the cable operator."
    ::= { cabhCapMappingEntry 6 }

cabhCapMappingLanPort OBJECT-TYPE
    SYNTAX      InetPortNumber
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
            "The TCP/UDP port number or ICMP Identifier
            on the LAN side. A port number/Identifier
            of 0 indicates either a DMZ mapping or a NAT
            mapping. A non-zero port number/Identifier
            indicates a NAPT mapping. If the value of
            cabhCapMappingNumPorts MIB object is non-zero,
            then this MIB represents a starting TCP/UDP port
            number on the LAN side for which a mapping
            entry is created."
    DEFVAL { 0 }
    ::= { cabhCapMappingEntry 7 }

cabhCapMappingMethod OBJECT-TYPE
    SYNTAX      INTEGER {
                    static(1),
                    dynamic(2),
                    upnp(3)
                }
    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. upnp (3) means that the
            CAP mapping entry was created by some UPnP compliant
            application."
    ::= { cabhCapMappingEntry 8 }

cabhCapMappingProtocol OBJECT-TYPE
    SYNTAX      INTEGER {
```

```
                    other(1),    -- any other protocol; e.g. IGMP
                    icmp(2),
                    udp(3),
                    tcp(4),
                    all(255)     -- covers all the protocols
                }
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
            "The protocol for this mapping entry. The value
            of other(1) represents a protocol other
            than ICMP, TCP, and UDP. Thus, when the value
            other(1) is specified for the cabhCapMappingProtocol
            value of a CAP Mapping Table entry,
            TCP, UDP or ICMP packets MUST NOT be forwarded even
            if the WAN and LAN IP address and port tuple
            of the packet matches with mapping entry.
            The value of all(255) represents all protocol types. Thus,
            when the cabhCapMappingProtocol value
            all(255) is specified for an entry in the CAP Mapping
            Table, traffic of all protocol types MUST be forwarded
            accordingly if the WAN and LAN IP address and port tuple
            in the packet matches the mapping entry."
    ::= { 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 PS will prevent modification of this table's
            columns and return an inconsistentValue error when
            cabhCapMappingRowStatus object is active(1).

            The PS must not allow RowStatus to be set to
            notInService(2) by a manager.

            A newly created row cannot be set to active(1) until
            the corresponding instances of cabhCapMappingWanAddr,
            cabhCapMappingLanAddr, and cabhCapMappingProtocol have
            been set.

            If the manager attempts to populate a row entry in
            the table with a non-unique value for the combination
            of cabhCapMappingWanAddr and range of WAN port(s)
            (identified by cabhCapMappingWanPort to
            cabhCapMappingWanPort + cabhCapMappingNumPorts –1),
            or a non-unique value for the combination of
            cabhCapMappingLanAddr and range of LAN port(s)
            (identified by cabhCapMappingLanPort to
            cabhCapMappingLanPort + cabhCapMappingNumPorts - 1),
            the PS MUST prevent the creation of this row and return
            an inconsistentValue error. This prevents creation of
            entries with overlapping port ranges in the CAP table.

            If the manager attempts to populate a row entry with
            a zero value for cabhCapMappingWanPort and a non-zero
            value for cabhCapMappingLanPort or a row entry with a
            zero value for cabhCapMappingLanPort and a non-zero value
            for cabhCapMappingWanPort, the PS MUST prevent the
```

```
            creation of this row and return an inconsistentValue
            error. This prevents creation of invalid NAT or NAPT entries.

            If the manager attempts to populate a row entry with non-zero
            values for both cabhCapMappingWanPort and
            cabhCapMappingLanPort, but a zero value for
            cabhCapMappingNumPorts, the PS MUST prevent
            the creation of this row and return an inconsistentValue
            error. This prevents creation of NAPT entries.

            When Primary Packet-handling Mode is NAPT
            (cabhCapPrimaryMode is napt(1)), provisioned rows can be
            set to active(1) regardless of whether the value to which
            cabhCapMappingWanPort, cabhCapMappingLanPort, and
            cabhCapMappingNumPorts have been set is zero or nonzero.

            When Primary Packet-handling Mode is NAT
            (cabhCapPrimaryMode is nat(2)), a newly created row can
            not be set to active(1) if a non-zero value has been set
            for cabhCapMappingWanPort, cabhCapMappingLanPort and
            cabhCapMappingNumPorts.

            In NAPT Primary Packet-handling mode, a row
            entry with zero values for cabhCapMappingWanPort,
            cabhCapMappingLanPort, and cabhCapMappingNumPorts
            objects represents a DMZ entry."
    ::={ cabhCapMappingEntry 10 }

cabhCapMappingNumPorts OBJECT-TYPE
    SYNTAX      Unsigned32(1..65535)
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
            "This object represents the number of ports
            available for port translation
            on both LAN and WAN side.

            When both cabhCapMappingWanPort and
            cabhCapMappingLanPort are set to zero,
            the PS MUST ignore this MIB object, and
            such a row entry represents either a DMZ entry
            (when primary packet handling mode is NAPT) or
            a NAT entry (when primary packet handling mode is
            NAT).

            When a row entry is created with non-zero
            values for cabhCapMappingWanPort,
            cabhCapMappingLanPort, and cabhCapMappingNumPorts
            the PS MUST translate range of ports on
            the WAN side (identified by cabhCapMappingWanPort
            to cabhCapMappingWanPort + cabhCapMappingNumPorts-1)
            to range of ports on the LAN side (identified by
            cabhCapMappingLanPort to cabhCapMappingLanPort +
            cabhCapMappingNumPorts-1).

            The PS MUST ignore this MIB for a CAP mapping
            entry with the value of cabhCapMappingProtocol
            equal to icmp(2)."
    DEFVAL { 1 }
    ::= { cabhCapMappingEntry 11 }

cabhCapMappingRowDescr OBJECT-TYPE
    SYNTAX      SnmpAdminString (SIZE(0..32))
    MAX-ACCESS  read-create
```

```
    STATUS      current
    DESCRIPTION
            "A string value that can be used to describe
             the purpose or attributes of the CAP Mapping
             entry."
    DEFVAL { "" }
    ::= { cabhCapMappingEntry 12 }

cabhCapMappingCreateTime OBJECT-TYPE
    SYNTAX      DateAndTime
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "For dynamic(2) and upnp(3) CAP mapping entries, the PS MUST
             set this MIB with date and time when the entry is created.
             The PS MUST set the value of this MIB to zero valued
             11-byte string for static CAP mapping entries. This MIB
             object MUST NOT persist across the PS reboot."
    ::= { cabhCapMappingEntry 13 }

cabhCapMappingLastUpdateTime OBJECT-TYPE
    SYNTAX      DateAndTime
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "The PS MUST set the value of this MIB
             to zero valued 11-byte string for static
             CAP mapping entries. For dynamic(2) CAP
             Mapping entries, the PS MUST set the value
             of this MIB to the value of cabhCapMappingCreateTime.
             For upnp(3) CAP mapping entries, the PS MUST
             set this MIB with date and time when the entry
             is last updated. When the upnp(3)entry is first
             created, the PS MUST set this MIB with the value
             of cabhCapMappingCreateTime MIB. This MIB object
             MUST NOT persist across the PS reboot."
    ::= { cabhCapMappingEntry 14 }

cabhCapMappingDuration OBJECT-TYPE
    SYNTAX      Integer32 (-1|0..2147483647)
    UNITS       "seconds"
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
            "When a value greater than zero
             is assigned to this object, the PS MUST
             remove the CAP entry after the time
             duration, represented by
             this object, elapses starting from
             cabhCapMappingLastUpdateTime.

             When a value of 0 is assigned to this object,
             the PS MUST retain the CAP mapping entry
             until reboot or reset. The PS MUST retain
             a CAP mapping entry with cabhCapMappingDuration
             MIB set to 0 and cabhCapMappingMethod set
             to static(1) across the reboots. The PS MUST
             NOT retain a CAP mapping entry with
             cabhCapMappingDuration MIB set to 0 and
             cabhCapMappingMethod set to upnp(3) across
             the reboots.

             When a value of -1 is assigned for this
             MIB, the PS MUST ignore this MIB and
```

```
                MUST remove the CAP mapping entries
                based on TCP, UDP and ICMP inactivity
                time-wait depending upon their protocol
                type.

                When the cabhCapMappingMethod object is
                static(1), the default value for this object
                is 0.

                When the cabhCapMappingMethod object is
                dynamic(2), the PS MUST set the value of
                this object to -1.

                When the cabhCapMappingMethod object is
                upnp(3), the default value for this object
                is -1."

    ::= { cabhCapMappingEntry 15 }

cabhCapMappingRemoteHostAddrType OBJECT-TYPE
    SYNTAX      InetAddressType
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "The IP address type for a remote host on the WAN side."
    DEFVAL { ipv4 }
    ::= { cabhCapMappingEntry 16 }

cabhCapMappingRemoteHostAddr OBJECT-TYPE
    SYNTAX      InetAddress
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "The IP address of the remote host for
            a CAP mapping entry. The packet traversing
            through the PS is either originated
            from or is destined to this remote host.
            The value of all zeros for this MIB object
            indicates any IP address for a remote host."
    DEFVAL { '00000000'h }
    ::= { cabhCapMappingEntry 17 }

cabhCapMappingEnable OBJECT-TYPE
    SYNTAX      TruthValue
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "This MIB allows the PS to enable or disable
            a particular CAP mapping entry. When this MIB
            is set to true(1) for a CAP mapping entry, the
            PS MUST correctly route the traffic that
            matches this entry. When this MIB is set to
            false(2) for a CAP mapping entry, the PS MUST
            NOT route the traffic that matches this entry."
    DEFVAL { true }
    ::= { cabhCapMappingEntry 18 }

--================================================================
--
--    cabhCapPassthroughTable (CAP Passthrough Table)
--
--    The cabhCapPassthroughTable contains the hardware addresses
--    for all LAN IP Devices for which the PS will bridge traffic at
--    OSI Layer 2 when the PS's cabhCapPrimaryMode is set to forward
```

```
--    traffic at OSI Layer 3 (NAPT/NAT) for all other hardware
--    addresses.
--
--============================================================================

cabhCapPassthroughTable OBJECT-TYPE
    SYNTAX      SEQUENCE OF CabhCapPassthroughEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "This table contains hardware addresses of LAN IP Devices
            for which the PS will bridge traffic at OSI Layer 2."
    REFERENCE
            "CableHome 1.1 Specification, Packet Handling & Address
            Translation section."
    ::= {  cabhCapMap 2 }

cabhCapPassthroughEntry OBJECT-TYPE
    SYNTAX      CabhCapPassthroughEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "List of hardware addresses of LAN IP Devices for which
            the PS will bridge traffic at OSI Layer 2."
    INDEX { cabhCapPassthroughIndex }
::= { cabhCapPassthroughTable 1 }

CabhCapPassthroughEntry::=SEQUENCE {
    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 OBJECT-TYPE
    SYNTAX      PhysAddress (SIZE(0..16))
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
            "Hardware address of the LAN IP Device for which the PS
            MUST bridge traffic at OSI Layer 2."
    ::={ 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.
            Any writable object in each row can be modified
            at any time while the row is active(1)."
    ::= { 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
            the CableHome Portal Services functionality."
    MODULE   --cabhCapMib

-- unconditionally mandatory groups

MANDATORY-GROUPS {
    cabhCapGroup
    }

OBJECT cabhCapMappingProtocol
    SYNTAX      INTEGER { icmp(2) }
    WRITE-SYNTAX  INTEGER { other(1), udp(3), tcp(4), all(255) }
    DESCRIPTION
            "icmp(2) applies only to dynamic entries."

    ::= { cabhCapCompliances 1 }


cabhCapGroup OBJECT-GROUP
    OBJECTS {
    cabhCapTcpTimeWait,
    cabhCapUdpTimeWait,
    cabhCapIcmpTimeWait,
    cabhCapPrimaryMode,
    cabhCapSetToFactory,
    cabhCapLastSetToFactory,
    cabhCapMappingWanAddrType,
    cabhCapMappingWanAddr,
    cabhCapMappingWanPort,
    cabhCapMappingLanAddrType,
    cabhCapMappingLanAddr,
    cabhCapMappingLanPort,
    cabhCapMappingMethod,
    cabhCapMappingProtocol,
    cabhCapMappingRowStatus,
    cabhCapPassthroughMacAddr,
    cabhCapPassthroughRowStatus,
    cabhCapMappingNumPorts,
    cabhCapMappingRowDescr,
    cabhCapMappingCreateTime,
    cabhCapMappingLastUpdateTime,
    cabhCapMappingDuration,
    cabhCapUpnpPortForwardingEnable,
    cabhCapUpnpTimeWait,
```

```
    cabhCapMappingRemoteHostAddrType,
    cabhCapMappingRemoteHostAddr,
    cabhCapMappingEnable
    }
    STATUS      current
    DESCRIPTION
            "Group of objects for CableHome CAP MIB."
    ::= { cabhCapGroups 1 }

END
```

E.2 IPCable2Home DHCP Portal (CDP) MIB requirement

The IPCable2Home CDP MIB MUST be implemented as defined below.

```
CABH-CDP-MIB DEFINITIONS ::= BEGIN
IMPORTS
    MODULE-IDENTITY,
    OBJECT-TYPE,
    Integer32,
    Unsigned32              FROM SNMPv2-SMI
    PhysAddress,
    TruthValue,
    DateAndTime,
    TimeStamp,
    RowStatus               FROM SNMPv2-TC --RFC2579
    OBJECT-GROUP,
    MODULE-COMPLIANCE       FROM SNMPv2-CONF
    InetAddressType,
    InetAddress             FROM INET-ADDRESS-MIB
    SnmpAdminString         FROM SNMP-FRAMEWORK-MIB
    clabProjCableHome       FROM CLAB-DEF-MIB;

cabhCdpMib MODULE-IDENTITY
    LAST-UPDATED    "200412160000Z"  -- December 16, 2004
    ORGANIZATION    "CableLabs Broadband Access Department"
    CONTACT-INFO
            "Kevin Luehrs
            Postal: Cable Television Laboratories, Inc.
            858 Coal Creek Circle
            Louisville, Colorado 80027
            U.S.A.
            Phone:  +1 303-661-9100
            Fax:    +1 303-661-9199
            E-mail: k.luehrs@cablelabs.com; mibs@cablelabs.com"
    DESCRIPTION
            "This MIB module supplies the basic management objects
            for the CableHome DHCP Portal (CDP) portion of the PS
            database."
    ::=  { clabProjCableHome 4 }

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 CableHome
--    DHCP Portal.  The rest of this group deals with addresses defined
--    on the LAN side.
--
```

```
cabhCdpSetToFactory OBJECT-TYPE
    SYNTAX      TruthValue
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
            "Reading this object always returns false(2). When the
            cabhCdpSetToFactory object is set to true(1), the PS must
            take the following actions:
            1) Clear all cabhCdpLanAddrEntries in the CDP LAN Address
               Table.
            2) The CDS must offer the factory default DHCP options
               at the next lease renewal time.
            3) Reset the following objects to their factory default
               values:

               cabhCdpLanTransThreshold,
               cabhCdpLanTransAction,
               cabhCdpWanDataIpAddrCount,
               cabhCdpTimeOffsetSelection,
               cabhCdpSnmpSetTimeOffset,
               cabhCdpDaylightSavingTimeEnable,
               cabhCdpLanPoolStartType,
               cabhCdpLanPoolStart,
               cabhCdpLanPoolEndType,
               cabhCdpLanPoolEnd,
               cabhCdpServerNetworkNumberType,
               cabhCdpServerNetworkNumber,
               cabhCdpServerSubnetMaskType,
               cabhCdpServerSubnetMask,
               cabhCdpServerTimeOffset,
               cabhCdpServerRouterType,
               cabhCdpServerRouter,
               cabhCdpServerDnsAddressType,
               cabhCdpServerDnsAddress,
               cabhCdpServerSyslogAddressType,
               cabhCdpServerSyslogAddress,
               cabhCdpServerDomainName,
               cabhCdpServerTTL,
               cabhCdpServerInterfaceMTU,
               cabhCdpServerVendorSpecific,
               cabhCdpServerLeaseTime,
               cabhCdpServerDhcpAddressType,
               cabhCdpServerDhcpAddress,
               cabhCdpServerCommitStatus"
::= { cabhCdpBase 1 }

cabhCdpLanTransCurCount OBJECT-TYPE
    SYNTAX      Unsigned32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "The current number of active leases in the
            cabhCdpLanAddrTable (the number of row entries in the
            table that have a cabhCdpLanAddrMethod value of
            reservationActive(2) or dynamicActive (4)).  This count
            does not include expired leases or reservations not
            associated with a current lease."
    ::= { cabhCdpBase 2 }

cabhCdpLanTransThreshold OBJECT-TYPE
    SYNTAX      INTEGER (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. Whenever an attempt is made to allocate a
            LAN-Trans IP address when cabhCdpLanTransCurCount is
            greater than or equal to cabhCdpLanTransThreshold, an
            event is generated. A value of 0 indicates that the CDP
            sets the threshold at the highest number of addresses in
            the LAN address pool."
    DEFVAL { 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 as would
            normally occur if the threshold was not exceeded.
            noAssignment - do not assign a LAN-Trans IP address."
    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 the
            PS's CDC must attempt to acquire via DHCP. When
            this MIB object is incremented, the CDC MUST
            immediately attempt to acquire additional WAN-Data
            IP addresses. When this MIB object is decremented, the
            CDC MUST not renew the leases for the appropriate
            number of WAN-Data IP addresses."
    DEFVAL { 0 }
    ::= { cabhCdpBase 5 }

cabhCdpLastSetToFactory    OBJECT-TYPE
    SYNTAX      TimeStamp
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "The value of sysUpTime when cabhCdpSetToFactory was
            last set to true. Zero if never reset."
    ::= { cabhCdpBase 6 }

cabhCdpTimeOffsetSelection OBJECT-TYPE
    SYNTAX      INTEGER {
                useDhcpOption2 (1),
                useSnmpSetOffset(2)
                }
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
            "This object selects the source to be used by the PS in
            determining the time offset to the time of day acquired
            from the time server. It is intended to be used in cases
```

```
            where the time zone information provisioned by the ToD
            server or DHCP Server (in DHCP Option 2) is different from
            the time zone where the provisioned device is physically
            located.

            Setting this object to useDhcpOption2(1) configures the PS
            to use the value of DHCP option 2 from the DHCP ACK message
            for time of day offset. Setting this object to
            useSnmpSetOffset(2) configures the PS to use the value of
            cabhCdpServerSnmpSetTimeOffset for time of day offset, and
            to ignore DHCP option 2. When the value of this object is
            changed, the PS MUST immediately begin using the time
            offset specified by the value of this object, regardless
            of which time offset the PS was using before the update
            occurred."
    DEFVAL { useDhcpOption2 }
 ::= { cabhCdpBase 7 }

cabhCdpSnmpSetTimeOffset OBJECT-TYPE
    SYNTAX      Integer32 (-43200..46800)  --  -12 to +13 hours (seconds)
    UNITS       "seconds"
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
            "This object is intended to be used in cases where the
            service provider's provisioning system serves devices in
            multiple time zones, or for other times when the service
            provider wants UTC time offset to be provisioned in a
            device other than from the ToD server or from the DHCP
            Server (in DHCP option 2).

            This object allows a manager to set a value for UTC time
            offset. If DHCP option 2 is not present in the DHCP ACK
            message, or if the value of DHCP option 2 is null, and time
            offset information is not provided in the response received
            from the time of day server, the PS MUST add the value of
            cabhCdpServerTimeOffset to the UTC time acquired from the
            time of day server to create the current time of day.

            If the value of cabhCdpServerTimeOffsetSelection is
            useSnmpSetOffset(2), the PS adds the value of
            cabhCdpServerSnmpSetTimeOffset to the UTC time acquired
            from the time of day server to create the current time
            of day.

            If the value of cabhCdpServerTimeOffsetSelection is
            useDhcpOption2(1), the PS ignores
            cabhCdpServerSnmpSetTimeOffset."
    DEFVAL { 0 }
    ::= { cabhCdpBase 8 }

cabhCdpDaylightSavingTimeEnable OBJECT-TYPE
    SYNTAX      INTEGER{
                enabled(1),
                disabled(2)
                }
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
            "This object allows a manager to configure the PS to adjust
            the current time of day based on Daylight Saving Time. If
            the value of this object is enabled(1), the PS adds 3600
            seconds and the time offset specified by
            cabhCdpServerTimeOffsetSelection to the UTC time acquired
```

```
            from the time of day server to create the current time of
            day during Daylight Saving Time, and adds only the time
            offset specified by cabhCdpServerTimeOffsetSelection to the
            UTC time acquired from the time of day server during
            standard time. The PS is responsible for knowing the date
            and time of each transition between Daylight Saving Time
            and standard time.

            If the value of this object is disabled(2), the PS adds
            only the time offset specified by
            cabhCdpServerTimeOffsetSelection to the UTC time acquired
            from the time of day server."
    DEFVAL { disabled }
    ::= { cabhCdpBase 9 }
--
--    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 table has one row entry for each allocated
               LAN-Trans IP address.  Each row must have at least a
               valid cabhCdpLanAddrMethod, a cabhCdpLanAddrIpType, a
               unique cabhCdpLanAddrIp, and a unique
               cabhCdpLanAddrClientId value.

               Static/Manual address assignment: To create a new DHCP
               address reservation, the NMS creates a row with: an
               index comprised of a new cabhCdpLanAddrIp and its
               cabhCdpLanAddrIpType, a new unique
               cabhCdpLanAddrClientID, (an empty LeaseCreateTime and
               empty LeaseExpireTime,) and a
               cabhCdpLanDataAddrRowStatus of createAndGo(4).  If the
               syntax and values of the new row - indicating a
               reservation - are valid, the PS must set
               cabhCdpLanAddrMethod to reservationInactive(1) and
               cabhCdpLanDataAddrRowStatus to active(1).  When the PS
               grants a lease for a reserved IP, it must set the
               cabhCdpLanAddrMethod object for that row to
               reservationActive(2).  When a lease for a reserved IP
               expires, the PS must set the corresponding row's
               cabhCdpLanAddrMethod object to reservationInactive(1).
               For row entries that represent lease reservations - rows
               in which the cabhCdpLanAddrMethod object has a value of
               either reservationInactive(1) or reservationActive(2) -
               the cabhCdpLanAddrIpType, cabhCdpLanAddrIp,
               cabhCdpLanAddrClientID, cabhCdpLanAddrMethod, and
               cabhCdpLanAddrHostName object values must persist across
```

```
                PS reboots.

                Dynamic address assignment:  When the PS grants a lease
                for a non-reserved IP, it must set the
                cabhCdpLanAddrMethod object for that row to
                dynamicActive(4).  When a lease for a non-reserved IP
                expires, the PS must set the corresponding row's
                cabhCdpLanAddrMethod object to dynamicInactive(3).  The
                PS must create new row entries using cabhCdpLanAddrIp
                values that are unique to this table.  If all
                cabhCdpLanAddrIp values in the range defined by
                cabhCdpLanPoolStart and cabhCdpLanPoolEnd are in use in
                this table, the PS may overwrite the
                cabhCdpLanAddrClientId of a row that has a
                cabhCdpLanAddrMethod object with a value of
                dynamicInactive(3) with a new cabhCdpLanAddrClientId
                value and use that cabhCdpLanAddrIp as part of a new
                lease.  For row entries that represent active leases -
                rows in which the cabhCdpLanAddrMethod object has a
                value of dynamicActive(4) - the cabhCdpLanAddrIpType,
                cabhCdpLanAddrIp, cabhCdpLanAddrClientID,
                cabhCdpLanAddrMethod, and cabhCdpLanAddrHostName object
                values must persist across PS reboots."
    ::= { cabhCdpAddr 1 }

cabhCdpLanAddrEntry OBJECT-TYPE
    SYNTAX      CabhCdpLanAddrEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "List of general parameters pertaining to LAN-Trans IP
            address reservations and leases."
    INDEX { cabhCdpLanAddrIpType, cabhCdpLanAddrIp }
    ::= { cabhCdpLanAddrTable 1 }

CabhCdpLanAddrEntry ::= SEQUENCE {
    cabhCdpLanAddrIpType              InetAddressType,
    cabhCdpLanAddrIp                  InetAddress,
    cabhCdpLanAddrClientID            PhysAddress,
    cabhCdpLanAddrLeaseCreateTime     DateAndTime,
    cabhCdpLanAddrLeaseExpireTime     DateAndTime,
    cabhCdpLanAddrMethod              INTEGER,
    cabhCdpLanAddrHostName            SnmpAdminString,
    cabhCdpLanAddrRowStatus           RowStatus
    }

cabhCdpLanAddrIpType OBJECT-TYPE
    SYNTAX      InetAddressType
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "The type of IP address assigned to the LAN IP Device
            in the LAN-Trans Realm."
    ::= { cabhCdpLanAddrEntry 1 }

cabhCdpLanAddrIp OBJECT-TYPE
    SYNTAX      InetAddress
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "The address assigned to the LAN IP Device. 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
```

```
            entered by the NMS through the CMP, when the NMS creates
            a new DHCP address reservation.  Each cabhCdpLanAddrIp
            in the table must fall within the range of IPs defined
            inclusively by cabhCdpLanPoolStart and
            cabhCdpLanPoolEnd.  The PS must return an
            inconsistentValue error if the NMS attempts to
            create a row entry with a cabhCdpLanAddrIP value that falls
            outside of this range or is not unique from all existing
            cabhCdpLanAddrIP entries in this table. The address type of
            this object is specified by cabhCdpLanAddrIpType."
    ::= { cabhCdpLanAddrEntry 2 }

cabhCdpLanAddrClientID OBJECT-TYPE
    SYNTAX      PhysAddress
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
            "The client's (i.e., LAN IP Device's) hardware address as
            indicated in the chaddr field of its DHCP REQUEST message.
            There is a one-to-one relationship between the hardware
            address and the LAN IP Device. This parameter is entered
            by the PS (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
            cabhCdpLanDataAddrRowStatus object with an index
            comprised of a unique cabhCdpLanAddrIp and creating
            a row with a unique cabhCdpLanAddrClientID."
    ::= { cabhCdpLanAddrEntry 3 }

cabhCdpLanAddrLeaseCreateTime OBJECT-TYPE
    SYNTAX      DateAndTime
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "This is the date and time when the LAN IP lease was
            created (if it has not yet been renewed) or last renewed.
            This MIB object contains a zero valued 11-byte string
            when a reservation is created for a LAN IP address
            and it maintains this value until the LAN IP Device
            acquires its lease and cabhCdpLanAddrMethod
            becomes reservationActive(2)."
    ::= { cabhCdpLanAddrEntry 4 }

cabhCdpLanAddrLeaseExpireTime OBJECT-TYPE
    SYNTAX      DateAndTime
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "This is the date and time when the LAN IP address lease
            expired or will expire. This MIB object contains a zero
            valued 11-byte string when a reservation is created for
            a LAN IP address and it maintains this value until the
            LAN IP Device acquires its lease and
            cabhCdpLanAddrMethod becomes reservationActive(2)."
    ::= { cabhCdpLanAddrEntry 5 }

cabhCdpLanAddrMethod OBJECT-TYPE
    SYNTAX      INTEGER {
                    mgmtReservationInactive(1),
                    mgmtReservationActive(2),
                    dynamicInactive(3),
                    dynamicActive(4),
```

```
                    psReservationInactive(5),
                    psReservationActive(6)
                }
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "The IP allocation method indicated by this row.

            The value of mgmtReservationInactive(1)
            indicates an externally provisioned IP address
            reservation that has not yet been leased or that
            has an expired lease. This indicates an IP address
            lease reservation created either by an operator or
            a user.

            The value of mgmtReservationActive(2)
            indicates an externally provisioned IP address
            reservation that has an active lease. This indicates
            an IP address lease reservation created either
            by an operator or a user.

            The value of dynamicInactive(3) indicates an
            IP address that was once dynamically assigned to a
            LAN-Trans by the PS device but currently
            has an expired lease.

            The value of dynamicActive(4)indicates an IP
            Address that was dynamically assigned to a
            LAN-Trans device by the PS and has a current
            active lease.

            The value of psReservationInactive(5)
            indicates an IP address reservation created by some
            internal process of the PS and has not yet been
            leased or has an expired lease.

            The value of psReservationActive(6)
            indicates an IP address reservation created by some
            internal process of the PS that has an active lease."
    ::= { cabhCdpLanAddrEntry 6 }

cabhCdpLanAddrHostName OBJECT-TYPE
    SYNTAX      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 of row
            entries. The PS must not allow the NMS to set RowStatus
            to notInService(2). The PS must assign a RowStatus of
            notInService(2) to any new row entry created with a
            non-unique, cabhCdpLanAddrClientID value. The PS must
            assign a RowStatus of notReady(3) to any new row entry
            created without a cabhCdpLanAddrClientID.  The PS will
            prevent modification of this table's columns and return an
            inconsistentValue error, if the NMS attempts to make such
```

```
            modifications while the RowStatus is active(1)."
    ::= { 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,
    cabhCdpWanDataAddrLeaseCreateTime DateAndTime,
    cabhCdpWanDataAddrLeaseExpireTime DateAndTime
    }

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 attempting
            to acquire a WAN-Data IP Address via DHCP."
    ::= { cabhCdpWanDataAddrEntry 2 }

cabhCdpWanDataAddrIpType OBJECT-TYPE
    SYNTAX      InetAddressType
    MAX-ACCESS  read-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-only
    STATUS      current
    DESCRIPTION
            "The address assigned on the WAN-Data side."
    ::= { cabhCdpWanDataAddrEntry 4 }

cabhCdpWanDataAddrRenewalTime OBJECT-TYPE
    SYNTAX      Integer32
    MAX-ACCESS  read-only
    STATUS      deprecated
    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 of row
            entries.  Any writable object in a row can be modified at
            any time while the row is active(1).  The PS must assign a
            RowStatus of notInService(2) to any new row entry created
            with a cabhCdpWanDataAddrClientId that is not unique within
            this table."
    ::= { cabhCdpWanDataAddrEntry 6 }

cabhCdpWanDataAddrLeaseCreateTime OBJECT-TYPE
    SYNTAX      DateAndTime
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "This is the date and time when the WAN-Data address lease
            was created (if it has not yet been renewed) or last
            renewed."
    ::= { cabhCdpWanDataAddrEntry 7 }

cabhCdpWanDataAddrLeaseExpireTime OBJECT-TYPE
    SYNTAX      DateAndTime
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "This is the date and time when the WAN-Data address
             lease expired or will expire."
    ::= { cabhCdpWanDataAddrEntry 8 }

--------------------------------------------------------------------
--
-- cabhCdpWanDnsServerTable (CDP WAN DNS Server Table)
--
-- The cabhCdpWanDnsServerTable is a table of 3 cable network
-- and Internet DNS Servers.
--
--------------------------------------------------------------------
cabhCdpWanDnsServerTable OBJECT-TYPE
    SYNTAX SEQUENCE OF CabhCdpWanDnsServerEntry
    MAX-ACCESS not-accessible
    STATUS current
```

```
    DESCRIPTION
            "This table contains the IP addresses of cable network and
            Internet DNS servers, in the order of preference in which
            the PS's CNP will query them, when it cannot resolve a DNS
            query using local information. Entries in this table are
            updated with the information contained in DHCP option 6,
            received during both the WAN-Man and WAN-Data IP
            acquisition processes."
    ::= { cabhCdpAddr 3 }

cabhCdpWanDnsServerEntry OBJECT-TYPE
    SYNTAX      CabhCdpWanDnsServerEntry
    MAX-ACCESS  not-accessible
    STATUS     current
    DESCRIPTION
            "List of cable network and Internet DNS servers."
    INDEX { cabhCdpWanDnsServerOrder }
    ::= { cabhCdpWanDnsServerTable 1 }

CabhCdpWanDnsServerEntry ::= SEQUENCE {
    cabhCdpWanDnsServerOrder  INTEGER,
    cabhCdpWanDnsServerIpType InetAddressType,
    cabhCdpWanDnsServerIp     InetAddress
    }

cabhCdpWanDnsServerOrder OBJECT-TYPE
    SYNTAX      INTEGER {
                    primary(1),
                    secondary(2),
                    tertiary(3)
                }
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "The order of preference for cable network and Internet DNS
            servers, as listed in DHCP option 6 (Domain Server). Any
            time the CDC receives valid IP address information within
            DHCP option 6, as part of lease acquisition or renewal of
            a WAN-Man or WAN-Data IP, it must update this information
            into this table.  As entries in DHCP option 6 are listed in
            order of preference, the highest priority entry in DHCP
            option 6 must correspond to the row with a
            cabhCdpWanDnsServerOrder with a value of 1. If DHCP
            option 6 contains 1 valid IP address, the PS MUST update
            the row with a cabhCdpWanDnsServerOrder value of 1 and
            MUST NOT modify rows with
            cabhCdpWanDnsServerOrder values of 2 & 3
            (if they exist). If DHCP option 6 contains 2 valid
            IP addresses, the PS MUST update the rows with
            cabhCdpWanDnsServerOrder values of 1 and 2
            and MUST NOT modify the row with cabhCdpWanDnsServerOrder
            value of 3 (if it exists). If DHCP option 6 contains 3
            valid IP addresses, the PS MUST update rows with
            cabhCdpWanDnsServerOrder values of 1, 2, and 3.
            Any DNS server information included in DHCP option 6
            beyond primary, secondary and tertiary will not be
            represented in this table."
    ::= { cabhCdpWanDnsServerEntry 1 }

cabhCdpWanDnsServerIpType OBJECT-TYPE
    SYNTAX      InetAddressType
    MAX-ACCESS read-only
    STATUS current
```

```
    DESCRIPTION
            "This parameter indicates the IP address type of a
            WAN DNS server."
    DEFVAL { ipv4 }
    ::= { cabhCdpWanDnsServerEntry 2 }

cabhCdpWanDnsServerIp OBJECT-TYPE
    SYNTAX      InetAddress
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "This parameter indicates the IP address of a WAN DNS
            server. The type of this address is specified by
            cabhCdpWanDnsServerIpType."
    ::= { cabhCdpWanDnsServerEntry 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. The type of
            this address is specified by cabhCdpLanPoolStartType."
    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. The type of
            this address is specified by cabhCdpLanPoolEndType."
    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. The type of this address is
            specified by cabhCdpServerNetworkNumberType."
    DEFVAL { 'c0a80000'h } --192.168.0.0
    ::= { cabhCdpServer 6 }

cabhCdpServerSubnetMaskType OBJECT-TYPE
    SYNTAX      InetAddressType
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
            "Type of LAN-Trans Subnet Mask."
    DEFVAL { ipv4 }
    ::= { cabhCdpServer 7 }

cabhCdpServerSubnetMask OBJECT-TYPE
    SYNTAX      InetAddress
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
            "The PS MUST provide the value of this MIB
            object in option 1 (Subnet Mask) of
            DHCP OFFER and ACK messages sent to a LAN IP Device."
    DEFVAL { 'ffffff00'h }    -- 255.255.255.0
    ::= { cabhCdpServer 8 }

cabhCdpServerTimeOffset OBJECT-TYPE
    SYNTAX      Integer32 (-86400..86400) -- 0 to 24 hours (in seconds)
    UNITS       "seconds"
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
            "The PS MUST provide the value of this MIB object in
            option 2 (Time Offset from Coordinated
            Universal Time-UTC) in the DHCP OFFER and ACK
            messages sent to the LAN IP Device."
    DEFVAL { 0 }    -- UTC
    ::= { cabhCdpServer 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 10 }

cabhCdpServerRouter OBJECT-TYPE
    SYNTAX      InetAddress
```

```
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
            "The type of this address is specified by
            cabhCdpServerRouterType. The PS MUST
            provide the value of this MIB object in
            option 3 (Router IP address) of the DHCP
            OFFER and ACK messages sent to the LAN IP Device."
    DEFVAL { 'c0a80001'h }    -- 192.168.0.1
    ::= { cabhCdpServer 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 12 }

cabhCdpServerDnsAddress OBJECT-TYPE
    SYNTAX      InetAddress
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
            "The default value of this MIB object is the
            same as the value of the cabhCdpServerRouter
            object. The NMS may set the value of this
            object to a value different than the value
            of cabhCdpServerRouter (e.g., DNS server in the
            cable data network) so that a LAN IP Device can direct its
            DNS queries to a server other than the PS DNS
            server. The type of this address is specified
            by cabhCdpServerDnsAddressType. The PS MUST
            provide the value of this MIB object in option 6
            (Domain Name Server) of DHCP OFFER and ACK
            messages sent to a LAN IP Device."
    ::= { cabhCdpServer 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 14 }

cabhCdpServerSyslogAddress OBJECT-TYPE
    SYNTAX      InetAddress
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
            "If the value of this object is non-zero, the PS will
            include the value of this object in DHCP option 7
            (Log Servers) in DHCP OFFER and DHCP ACK messages
            sent to the LAN IP Device."
    DEFVAL { '00000000'h }    -- 0.0.0.0
    ::= { cabhCdpServer 15 }

cabhCdpServerDomainName OBJECT-TYPE
    SYNTAX      SnmpAdminString(SIZE(0..128))
    MAX-ACCESS  read-write
```

```
    STATUS      current
    DESCRIPTION
            "The PS MUST provide the value of this MIB object
            in option 15 (Domain Name Option) of the DHCP
            OFFER and ACK messages sent to the LAN IP Device."
    DEFVAL { "" }
    ::= { cabhCdpServer 16 }

cabhCdpServerTTL OBJECT-TYPE
    SYNTAX      INTEGER (1..255)
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
            "The PS MUST provide the value of this MIB
             object in option 23 (Default IP TTL) of
             DHCP OFFER and ACK messages sent to a LAN IP Device."
    DEFVAL { 64 }
    ::= { cabhCdpServer 17 }

cabhCdpServerInterfaceMTU OBJECT-TYPE
    SYNTAX      Integer32 (0 | 68..4096)
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
            "The PS MUST provide the value of this MIB object in
            option 26 (Interface MTU Option) of the DHCP OFFER
            and ACK messages sent to the LAN IP Device. If the value
            of this object is 0, the PS must not include this option
            in its DHCP OFFER or DHCP ACK messages to LAN IP Devices."
    DEFVAL { 0 }
    ::= { cabhCdpServer 18 }

cabhCdpServerVendorSpecific OBJECT-TYPE
    SYNTAX      OCTET STRING (SIZE(0..255))
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
            "The PS MUST provide the value of this MIB object in
            option 43 (Vendor Specific Information) of the DHCP OFFER
            and ACK messages sent to the LAN IP Device. If the value of
            this object is ' 'h,  then the PS MUST NOT include this
            option in its DHCP OFFER or DHCP ACK messages to LAN IP
            Devices."
    DEFVAL { ''h }
    ::= { cabhCdpServer 19 }

cabhCdpServerLeaseTime OBJECT-TYPE
    SYNTAX      Unsigned32
    UNITS       "seconds"
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
            "The PS MUST provide the value of this MIB object in
            option 51 (IP Address lease time) of the DHCP OFFER and
            ACK messages sent to the LAN IP Device."
    DEFVAL { 3600 }
    ::= { cabhCdpServer 20 }

cabhCdpServerDhcpAddressType OBJECT-TYPE
    SYNTAX      InetAddressType
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "Type of LAN DHCP server IP address. The
```

```
            IP address of LAN DHCP server is provided by
            the PS in option 54 of DHCP OFFER or ACK."
    DEFVAL { ipv4 }
    ::= { cabhCdpServer 21 }

cabhCdpServerDhcpAddress OBJECT-TYPE
    SYNTAX      InetAddress
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "The value of this MIB object is always the
            same as the value of the cabhCdpServerRouter
            object. The type of this address is specified
            by cabhCdpServerDhcpAddressType.
            The PS MUST provide the value of this MIB
            object in option 54 (DHCP server identifier)
            field of DHCP OFFER and ACK messages
            sent to a LAN IP device."
    ::= { cabhCdpServer 22 }


cabhCdpServerControl OBJECT-TYPE
    SYNTAX      INTEGER {
                    restoreConfig(1),
                    commitConfig(2)
                }
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
            "The control for the CDS (DHCP Server) configuration.
            All changes to the cabhCdpServer MIB objects are reflected
            when reading the value of the MIB objects; however, those
            changes are NOT applied to the running configuration of
            the CDS until they are successfully committed via use
            of the cabhCdpServerControl object.

            If changes are made to the cabhCdpServer MIB objects which
            are not yet successfully committed to the CDS, the
            cabhCdpServerControl object can be used to roll back all
            changes to the last valid CDS configuration and discard
            all intermediate changes.

            restoreConfig - Setting cabhCdpServerControl to this value
            will cause any changes to the cabhCdpServer objects not yet
            committed be reset to the values from the current running
            configuration of the CDS.

            commitConfig - Setting cabhCdpServerControl to this value
            will cause the CDS to validate and apply the valid
            cabhCdpServer MIB settings to its running configuration.
            The cabhCdpServerCommitStatus object will detail the
            status of this operation."
    DEFVAL { restoreConfig }
    ::= { cabhCdpServer 23 }

cabhCdpServerCommitStatus OBJECT-TYPE
    SYNTAX      INTEGER {
                    commitSucceeded(1),
                    commitNeeded(2),
                    commitFailed(3)
                }
    MAX-ACCESS  read-only
    STATUS      current
```

```
    DESCRIPTION
            "Indicates the status of committing the current
            cabhCdpServer MIB object values to the running
            configuration of the CDS (DHCP Server).

            commitSucceeded - indicates the current cabhCdpServer
            MIB object values are valid and have been successfully
            committed to the running configuration of the CDS.

            commitNeeded - indicates that the value of one or more
            objects in cabhCdpServer MIB group have been changed
            but not yet committed to the running configuration
            of the CDS.

            commitFailed - indicates the PS was unable to commit the
            cabhCdpServer MIB object values to the running
            configuration of the CDS due to conflicts in those
            values."
    DEFVAL { commitSucceeded }
    ::= { cabhCdpServer 24 }

cabhCdpServerUseCableDataNwDnsAddr OBJECT-TYPE
    SYNTAX      TruthValue
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
            "If the value of this object is false(2), the PS will
            provide the DNS Server IP address as specified in
            cabhCdpServerDnsAddress MIB object in option 6
            (Domain Name Server) of the DHCP OFFER and ACK messages
            sent to a LAN IP Device.

            When the object cabhCdpServerUseCableDataNwDnsAddr is set
            to true(1), the PS must take the following actions:
            The PS will provide in option 6 (Domain Name Server), of
            the DHCP OFFER and ACK messages sent to a LAN IP Device,
            the DNS server address(es) which is/are being used by the
            PS itself, i.e., the DNS server address(es) provided to the
            PS in DHCP option 6 and made available through PS MIB
            object cabhCdpWanDnsServerIp.

            The LAN IP Device can then direct its DNS queries to a
            server other than the PS DNS server. The PS MUST provide
            the value of this."
    DEFVAL { false }
    ::= { cabhCdpServer 25 }

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

cabhCdpNotification OBJECT IDENTIFIER ::= { cabhCdpMib 2 }
cabhCdpNotifications OBJECT IDENTIFIER ::= { cabhCdpNotification 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
            the CableHome Portal Services functionality."
    MODULE   --cabhCdpMib


-- unconditionally mandatory groups

MANDATORY-GROUPS {
    cabhCdpGroup
    }

    ::= { cabhCdpCompliances 3 }

cabhCdpGroup OBJECT-GROUP
    OBJECTS {
    cabhCdpSetToFactory,
    cabhCdpLanTransCurCount,
    cabhCdpLanTransThreshold,
    cabhCdpLanTransAction,
    cabhCdpWanDataIpAddrCount,
    cabhCdpLastSetToFactory,
    cabhCdpTimeOffsetSelection,
    cabhCdpSnmpSetTimeOffset,
    cabhCdpDaylightSavingTimeEnable,

    cabhCdpLanAddrClientID,
    cabhCdpLanAddrLeaseCreateTime,
    cabhCdpLanAddrLeaseExpireTime,
    cabhCdpLanAddrMethod,
    cabhCdpLanAddrHostName,
    cabhCdpLanAddrRowStatus,

    cabhCdpWanDataAddrClientId,
    cabhCdpWanDataAddrIpType,
    cabhCdpWanDataAddrIp,
    -- cabhCdpWanDataAddrRenewalTime,
    cabhCdpWanDataAddrRowStatus,
    cabhCdpWanDataAddrLeaseCreateTime,
    cabhCdpWanDataAddrLeaseExpireTime,

    cabhCdpWanDnsServerIpType,
    cabhCdpWanDnsServerIp,

    cabhCdpLanPoolStartType,
    cabhCdpLanPoolStart,
    cabhCdpLanPoolEndType,
    cabhCdpLanPoolEnd,
    cabhCdpServerNetworkNumberType,
    cabhCdpServerNetworkNumber,
    cabhCdpServerSubnetMaskType,
    cabhCdpServerSubnetMask,
    cabhCdpServerTimeOffset,
    cabhCdpServerRouterType,
    cabhCdpServerRouter,
    cabhCdpServerDnsAddressType,
    cabhCdpServerDnsAddress,
    cabhCdpServerSyslogAddressType,
    cabhCdpServerSyslogAddress,
    cabhCdpServerDomainName,
    cabhCdpServerTTL,
    cabhCdpServerInterfaceMTU,
```

```
    cabhCdpServerVendorSpecific,
    cabhCdpServerLeaseTime,
    cabhCdpServerDhcpAddressType,
    cabhCdpServerDhcpAddress,
    cabhCdpServerControl,
    cabhCdpServerCommitStatus,
    cabhCdpServerUseCableDataNwDnsAddr
    }
    STATUS      current
    DESCRIPTION
            "Group of objects for CableHome CDP MIB."
    ::= { cabhCdpGroups 1 }

END
```

E.3 IPCable2Home Test Portal (CTP) MIB requirement

The IPCable2Home CTP MIB MUST be implemented as defined below:

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

cabhCtpMib MODULE-IDENTITY
    LAST-UPDATED  "200404090000Z" -- April 9, 2004
    ORGANIZATION    "CableLabs Broadband Access Department"
    CONTACT-INFO
            "Kevin Luehrs
            Postal: Cable Television Laboratories, Inc.
            858 Coal Creek Circle
            Louisville, Colorado 80027
            U.S.A.
            Phone:  +1 303-661-9100
            Fax:    +1 303-661-9199
            E-mail: k.luehrs@cablelabs.com or mibs@cablelabs.com"
    DESCRIPTION
            "This MIB module defines control and monitoring objects
            for remote diagnostic tools for a CableHome LAN
            supported by the CableHome Test Portal (CTP) as
            defined and described in CableLabs' CableHome
            specifications."
    ::=  { clabProjCableHome 5 }

-- Textual conventions


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 CableHome
--    Management Portal.
--
```

```
cabhCtpSetToFactory OBJECT-TYPE
    SYNTAX      TruthValue
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
            "Setting this object to true(1) causes all 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)."
    ::={ cabhCtpBase 1 }

cabhCtpLastSetToFactory OBJECT-TYPE
    SYNTAX      TimeStamp
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "The value of sysUpTime when cabhCtpSetToFactory
            was last set to true. Zero if never reset."
    ::={ cabhCtpBase 2 }

--
--    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.
            The PS MUST NOT allow the value of cabhCtpConnSrcIpType
            to be changed if cabhCtpConnStatus = running(2). The PS
            MUST return inconsistentValue error to a manager that
            attempts to set the value of cabhCtpConnSrcIpType when the
            value of cabhCtpConnStatus is running(2)."
    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. The default value is the value
            of cabhCdpServerRouter (192.168.0.1). The type of
            this address is specified by cabhCtpConnSrcIpType.
            The PS MUST NOT allow the value of cabhCtpConnSrcIp
            to be changed if cabhCtpConnStatus = running(2). The PS
            MUST return inconsistentValue error to a manager that
            attempts to set the value of cabhCtpConnSrcIp when the
            value of cabhCtpConnStatus is running(2)."
    REFERENCE
            "CableHome Specification, Management Tools - PS
            Logical Element CableHome Test Portal (CTP) section."
    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 for the CTP Connection Speed Tool
            destination address.
            The PS MUST NOT allow the value of cabhCtpConnDestIpType
            to be changed if cabhCtpConnStatus = running(2). The PS
            MUST return inconsistentValue error to a manager that
            attempts to set the value of cabhCtpConnDestIpType when the
            value of cabhCtpConnStatus is running(2)."
    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. The type of this address is
            specified by cabhCtpConnDestIpType. The PS MUST NOT allow
            the value of cabhCtpConnDestIp to be changed if
            cabhCtpConnStatus = running(2). The PS MUST return
            inconsistentValue error to a manager that attempts
            to set the value of cabhCtpConnDestIp when the value of
            cabhCtpConnStatus is running(2)."
    ::= { 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.
            The PS MUST NOT allow the value of cabhCtpConnProto
            to be changed if cabhCtpConnStatus = running(2). The PS
            MUST return inconsistentValue error to a manager that
            attempts to set the value of cabhCtpConnProto when the
            value of cabhCtpConnStatus is running(2)."
    DEFVAL { udp }
    ::= { cabhCtpConnSpeed 5 }

cabhCtpConnNumPkts OBJECT-TYPE
    SYNTAX      INTEGER (1..65535)
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
            "The number of OSI Layer 3 (IP) packets the CTP is to
            send when triggered to execute the Connection Speed Tool.
            The PS MUST NOT allow the value of cabhCtpConnNumPkts
            to be changed if cabhCtpConnStatus = running(2). The PS
            MUST return inconsistentValue error to a manager that
            attempts to set the value of cabhCtpConnNumPkts when the
            value of cabhCtpConnStatus is running(2)."
    DEFVAL { 100 }
    ::= { cabhCtpConnSpeed 6 }

cabhCtpConnPktSize OBJECT-TYPE
    SYNTAX      INTEGER (64..1518)
    MAX-ACCESS  read-write
    STATUS      current
```

```
    DESCRIPTION
            "The size of each OSI Layer 2 frame to be
            sent by the PS CableHome Test Portal
            function when configured to execute the
            Connection Speed remote diagnostic tool.
            The PS MUST NOT allow the value of cabhCtpConnPktSize
            to be changed if cabhCtpConnStatus = running(2).
            The PS MUST return inconsistentValue error
            to a manager that attempts to set the value of
            cabhCtpConnPktSize when the value of cabhCtpConnStatus
            is running(2)."
    REFERENCE
            "CableHome Specification, Management Tools - PS
            Logical Element CableHome Test Portal (CTP) section."
    DEFVAL { 1518 }
    ::= { cabhCtpConnSpeed 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.
            The PS MUST NOT allow the value of cabhCtpConnTimeOut
            to be changed if cabhCtpConnStatus = running(2). The PS
            MUST return inconsistentValue error to a manager that
            attempts to set the value of cabhCtpConnTimeOut when the
            value of cabhCtpConnStatus is running(2)."
    DEFVAL {30000}  -- 30 seconds
    ::= { cabhCtpConnSpeed 8 }

cabhCtpConnControl OBJECT-TYPE
    SYNTAX      INTEGER {
                    start(1),
                    abort(2)
                }
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
            "The control for the Connection Speed Tool. Setting this
            object to start(1) causes the Connection Speed Tool to
            execute. Setting this object to abort(2) causes the
            Connection Speed Tool to stop running. This parameter
            should only be set via SNMP."
    DEFVAL  {abort }
    ::={ cabhCtpConnSpeed 9 }

cabhCtpConnStatus OBJECT-TYPE
    SYNTAX      INTEGER {
                    notRun(1),
                    running(2),
                    complete(3),
                    aborted(4),
                    timedOut(5)
                }
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "This object returns the status of the Connection Speed
            Tool. The value notRun(1) indicates that the Connection
            Speed Tool has not been run since the Portal Services
            element of the CableHome residential gateway was
```

```
            initialized or reset.

            The value running(2) indicates that the Connection Speed
            Tool was initiated by a manager
            (cabhCtpConnControl = start(1)) and the test has not
            timed out and the PS has not yet completed sending
            all the packets it was configured to send or it has not
            received all responses.

            The value complete(3) indicates that the Connection Speed
            Tool was initiated by a manager, successfully sent all the
            packets it was configured to send, received all responses,
            and is no longer sending packets or waiting for responses.

            The value aborted(4) indicates that the Connection Speed
            Tool was initiated by a manager then was terminated by the
            manager by setting cabhCtpConnControl = abort(2).
            The Connection Speed Tool is no longer sending packets or
            waiting for responses.

            The value timedOut(5) indicates that the Connection Speed
            Tool was initiated by a manager and had not received all
            responses from the client but the amount of time allowed
            for the Connection Speed Tool to execute, defined by the
            value of cabhCtpConnTimeOut, has transpired. The Connection
            Speed Tool is no longer sending packets or waiting
            for responses."
    DEFVAL { notRun }
    ::={ cabhCtpConnSpeed 10 }

cabhCtpConnPktsSent OBJECT-TYPE
    SYNTAX      INTEGER (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 11 }

cabhCtpConnPktsRecv OBJECT-TYPE
    SYNTAX      INTEGER (0..65535)
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "The number 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 }

cabhCtpConnThroughput OBJECT-TYPE
    SYNTAX      INTEGER (0..65535)
    MAX-ACCESS  read-only
    STATUS      current
```

```
    DESCRIPTION
            "The average round-trip throughput measured in
            kilobits per second."
    ::= { cabhCtpConnSpeed 14 }

--
--    Parameters and Results for Ping Command
--

cabhCtpPingSrcIpType OBJECT-TYPE
    SYNTAX      InetAddressType
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
            "The IP Address Type for CTP Ping Tool source address.
            The PS MUST NOT allow the value of cabhCtpPingSrcIpType
            to be changed if cabhCtpPingStatus = running(2). The PS
            MUST return inconsistentValue error to a manager that
            attempts to set the value of cabhCtpPingSrcIpType when the
            value of cabhCtpPingStatus is running(2)."
    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. The default value is the value of
            CabhCdpServerRouter (192.168.0.1). The type of this
            address is specified by cabhCtpPingSrcIpType.
            The PS MUST NOT allow the value of cabhCtpPingSrcIp
            to be changed if cabhCtpPingTimeOut = running(2). The PS
            MUST return inconsistentValue error to a manager that
            attempts to set the value of cabhCtpPingSrcIp when the
            value of cabhCtpPingTimeOut is running(2)."
    REFERENCE
            "CableHome Specification, Management Tools - PS
            Logical Element CableHome Test Portal (CTP) section."
    DEFVAL { 'c0a80001'h } --192.168.0.1
    ::= { cabhCtpPing 2 }

cabhCtpPingDestIpType OBJECT-TYPE
    SYNTAX      InetAddressType
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
            "The IP Address Type for the CTP Ping Tool destination
            address.
            The PS MUST NOT allow the value of cabhCtpPingDestIpType
            to be changed if cabhCtpPingStatus = running(2). The PS
            MUST return inconsistentValue error to a manager that
            attempts to set the value of cabhCtpPingDestIpType when the
            value of cabhCtpPingStatus is running(2)."
    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. The type of this address
            is specified by cabhCtpPingDestIpType. The PS MUST NOT
            allow the value of cabhCtpPingDestIp to be changed if
            cabhCtpPingStatus = running(2). The PS MUST return
            inconsistentValue error to a manager that attempts
            to set the value of cabhCtpPingDestIp when the value of
            cabhCtpPingStatus is running(2)."
    ::= { cabhCtpPing 4 }

cabhCtpPingNumPkts OBJECT-TYPE
    SYNTAX      INTEGER (1..4)
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
            "The number of ICMP Echo Request messages to send to
            the destination defined by cabhCtpPingDestIp. The PS
            MUST NOT allow the value of cabhCtpPingNumPkts to be
            changed if cabhCtpPingStatus = running(2).  The PS MUST
            return inconsistentValue error to a manager that attempts
            to set the value of cabhCtpPingNumPkts when the value of
            cabhCtpPingStatus is running(2)."
    DEFVAL { 1 }
    ::= { cabhCtpPing 5 }

cabhCtpPingPktSize OBJECT-TYPE
    SYNTAX      INTEGER (64..1518)
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
            "The size of the ICMP Echo Request packets to send to
            the destination defined by cabhCtpPingDestIp. The PS
            MUST NOT allow the value of cabhCtpPingPktSize to be
            changed if cabhCtpPingStatus = running(2).  The PS MUST
            return inconsistentValue error to a manager that attempts
            to set the value of cabhCtpPingPktSize when the value of
            cabhCtpPingStatus is running(2)."
    DEFVAL { 64 }
    ::= { cabhCtpPing 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.
            The PS MUST NOT allow the value of cabhCtpPingTimeBetween
            to be changed if the value of cabhCtpPingStatus is
            running(2).  The PS MUST return inconsistentValue error
            to a manager that attempts to set the value of
            cabhCtpPingTimeBetween when the value of
            cabhCtpPingStatus is running(2)."
    DEFVAL { 1000 }
    ::= { cabhCtpPing 7 }

cabhCtpPingTimeOut OBJECT-TYPE
    SYNTAX      INTEGER (1..600000)
    UNITS       "milliseconds"
    MAX-ACCESS  read-write
    STATUS      current
```

```
    DESCRIPTION
            "The time out for ping response (ICMP reply) for a
            single transmitted ping message (ICMP request).
            The PS MUST NOT allow the value of cabhCtpPingTimeOut
            to be changed if cabhCtpPingStatus = running(2). The PS
            MUST return inconsistentValue error to a manager that
            attempts to set the value of cabhCtpPingTimeOut when the
            value of cabhCtpPingStatus is running(2)."
    DEFVAL { 1000 } -- 1 second
    ::={ cabhCtpPing 8 }

cabhCtpPingControl OBJECT-TYPE
    SYNTAX      INTEGER {
                    start(1),
                    abort(2)
                }
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
            "The control for the Ping Tool. Setting this object
            to start(1) causes the 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 {abort }
    ::={ cabhCtpPing 9 }

cabhCtpPingStatus OBJECT-TYPE
    SYNTAX      INTEGER {
                    notRun(1),
                    running(2),
                    complete(3),
                    aborted(4),
                    timedOut(5)
                }
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "This object returns the status of the Ping Tool.

            The value notRun(1) indicates that the Ping Tool
            has not been run since the Portal Services element of the
            CableHome residential gateway was initialized or reset.

            The value running(2) indicates that the Ping Tool
            was initiated by a manager (cabhCtpPingControl = start(1))
            and the test has not timed out and the PS has not yet
            completed sending all the packets it was configured to
            send or it has not received all responses.

            The value complete(3) indicates that the Ping Tool
            was initiated by a manager, successfully sent all the
            packets it was configured to send, received all responses,
            and is no longer sending packets or waiting for responses.

            The value aborted(4) indicates that the Ping Tool was
            initiated by a manager then was terminated by the manager
            by setting cabhCtpPingControl = abort(2). The Ping Tool
            is no longer sending packets or waiting for responses.

            The value timedOut(5) indicates that the Ping Tool was
            initiated by a manager and had not received all responses
            from the client but the amount of time allowed for the
            Ping Tool to execute, defined by the value of
            cabhCtpPingTimeOut, has transpired. The Ping Tool is no
```

```
            longer sending packets or waiting for responses."
    DEFVAL { notRun }
    ::={ cabhCtpPing 10 }

cabhCtpPingNumSent OBJECT-TYPE
    SYNTAX      INTEGER (0..4)
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "The number of Pings sent."
    ::={ cabhCtpPing 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 }
cabhCtpNotifications OBJECT IDENTIFIER ::= { cabhCtpNotification 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 {

        cabhCtpSetToFactory,
        cabhCtpLastSetToFactory,
        cabhCtpConnSrcIpType,
        cabhCtpConnSrcIp,
        cabhCtpConnDestIpType,
        cabhCtpConnDestIp,
        cabhCtpConnProto,
        cabhCtpConnNumPkts,
        cabhCtpConnPktSize,
        cabhCtpConnTimeOut,
        cabhCtpConnControl,
        cabhCtpConnStatus,
        cabhCtpConnPktsSent,
        cabhCtpConnPktsRecv,
        cabhCtpConnRTT,
        cabhCtpConnThroughput,

        cabhCtpPingSrcIpType,
        cabhCtpPingSrcIp,
        cabhCtpPingDestIpType,
        cabhCtpPingDestIp,
        cabhCtpPingNumPkts,
```

```
        cabhCtpPingPktSize,
        cabhCtpPingTimeBetween,
        cabhCtpPingTimeOut,
        cabhCtpPingControl,
        cabhCtpPingStatus,
        cabhCtpPingNumSent,
        cabhCtpPingNumRecv,
        cabhCtpPingAvgRTT,
        cabhCtpPingMinRTT,
        cabhCtpPingMaxRTT,
        cabhCtpPingNumIcmpError,
        cabhCtpPingIcmpError
        }
    STATUS      current
    DESCRIPTION
            "Group of objects for CableHome CTP MIB."
    ::= { cabhCtpGroups 1 }

END
```

E.4 IPCable2Home Portal Services Device (PSDev) MIB requirement

The IPCable2Home PSDev MIB MUST be implemented as defined below.

```
CABH-PS-DEV-MIB DEFINITIONS ::= BEGIN
IMPORTS
    MODULE-IDENTITY,
    OBJECT-TYPE,
    Integer32,
    Unsigned32,
    TimeTicks,
    NOTIFICATION-TYPE            FROM SNMPv2-SMI

    TruthValue,
    PhysAddress,
    DateAndTime,
    TimeStamp,
    RowStatus                    FROM SNMPv2-TC

    SnmpAdminString              FROM SNMP-FRAMEWORK-MIB

    OBJECT-GROUP,
    MODULE-COMPLIANCE,
    NOTIFICATION-GROUP           FROM SNMPv2-CONF

    ifIndex                      FROM IF-MIB

    InetAddressType,
    InetAddress                  FROM INET-ADDRESS-MIB

    IANAifType                   FROM IANAifType-MIB

    docsDevSwCurrentVers,
    docsDevEvLevel,
    docsDevEvId,
    docsDevEvText,
    docsDevSwFilename,
    docsDevSwServer              FROM DOCS-CABLE-DEVICE-MIB -- RFC 2669

    cabhCdpServerDhcpAddress,
    cabhCdpWanDataAddrClientId,
    cabhCdpLanTransThreshold,
```

```
    cabhCdpLanTransCurCount      FROM CABH-CDP-MIB

    ZeroBasedCounter32           FROM RMON2-MIB

    cabhQos2NumActivePolicyHolder,
    cabhQos2PolicyHolderEnabled,
    cabhQos2PolicyAdmissionControl FROM CABH-QOS2-MIB

    clabProjCableHome            FROM CLAB-DEF-MIB;

cabhPsDevMib MODULE-IDENTITY
    LAST-UPDATED    "200504080000Z" -- April 8, 2005
    ORGANIZATION    "CableLabs Broadband Access Department"
    CONTACT-INFO
            "Kevin Luehrs
            Postal: Cable Television Laboratories, Inc.
            858 Coal Creek Circle
            Louisville, Colorado 80027
            U.S.A.
            Phone:  +1 303-661-9100
            Fax:    +1 303-661-9199
            E-mail: k.luehrs@cablelabs.com; mibs@cablelabs.com"
    DESCRIPTION
            "This MIB module supplies the basic management objects for
            the Portal Services logical element of a CableHome
            compliant Residential Gateway device. The PS device
            parameters describe general PS Device attributes and
            behaviour characteristics.
            Most the PS Device MIB is needed for configuration
            download."
    ::= { clabProjCableHome 1 }

-- Textual Conventions

cabhPsDevMibObjects     OBJECT IDENTIFIER ::= { cabhPsDevMib 1 }
cabhPsDevBase           OBJECT IDENTIFIER ::= { cabhPsDevMibObjects 1 }
cabhPsDevProv           OBJECT IDENTIFIER ::= { cabhPsDevMibObjects 2 }
cabhPsDevAttrib         OBJECT IDENTIFIER ::= { cabhPsDevMibObjects 3 }
cabhPsDevPsAttrib       OBJECT IDENTIFIER ::= { cabhPsDevAttrib 1 }
cabhPsDevBpAttrib       OBJECT IDENTIFIER ::= { cabhPsDevAttrib 2 }
cabhPsDevStats          OBJECT IDENTIFIER ::= { cabhPsDevMibObjects 4 }
cabhPsDevAccessControl  OBJECT IDENTIFIER ::= { cabhPsDevMibObjects 5 }
cabhPsDevMisc           OBJECT IDENTIFIER ::= { cabhPsDevMibObjects 6 }
cabhPsDevUI             OBJECT IDENTIFIER ::= { cabhPsDevMisc 1 }
cabhPsDev802dot11       OBJECT IDENTIFIER ::= { cabhPsDevMisc 2 }
cabhPsDevUpnp           OBJECT IDENTIFIER ::= { cabhPsDevMisc 3 }
cabhPsDevUpnpBase       OBJECT IDENTIFIER ::= { cabhPsDevUpnp 1 }
cabhPsDevUpnpCommands   OBJECT IDENTIFIER ::= { cabhPsDevUpnp 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 standalone or
            embedded PS device to reboot. Device code initializes as if
            starting from a power-on reset. The CMP ensures that MIB
            object values persist as specified in Annex A. Reading this
            object always returns false(2)."
    ::= { cabhPsDevBase 2 }

cabhPsDevSerialNumber OBJECT-TYPE
    SYNTAX      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."
    ::= { cabhPsDevBase 3 }

cabhPsDevHardwareVersion OBJECT-TYPE
    SYNTAX      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 }

cabhPsDevWanManMacAddress OBJECT-TYPE
    SYNTAX      PhysAddress (SIZE (0..16))
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "The PS WAN-Man MAC address. This is the PS hardware
            address 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 (SIZE (0..16))
    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      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."
```

```
    REFERENCE
            "CableHome Specification, CDC Function System
            Description section."
    ::= { 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      deprecated
    DESCRIPTION
            "This is the client ID used for WAN-MAN DHCP requests.
            The default value is the 6 byte MAC address."
    ::= { cabhPsDevBase 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."
    DEFVAL { false }
    ::= { cabhPsDevBase 10 }

cabhPsDevProvMode OBJECT-TYPE
    SYNTAX      INTEGER
    {
        dhcpmode(1),
        snmpmode(2),
        dormantCHmode(3)
    }
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "This object indicates the provisioning mode in which the
            PS is operating. If the PS is operating in DHCP
            Provisioning Mode as described in the CableHome
            specification, 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). If the PS is not
            configured to operate in either dhcpmode or snmpmode,
            it will fall back to Dormant CableHome Mode and set
            the value of cabhPsDevProvMode to dormantCHmode(3)."
    ::= { cabhPsDevBase 11 }

cabhPsDevLastSetToFactory OBJECT-TYPE
    SYNTAX      TimeStamp
    MAX-ACCESS  read-only
    STATUS      current
```

```
    DESCRIPTION
            "The value of sysUpTime when cabhPsDevSetToFactory was
            last set to true. Zero if never reset."
    ::= { cabhPsDevBase 12 }

cabhPsDevTrapControl OBJECT-TYPE
    SYNTAX BITS {
        cabhPsDevInitTLVUnknownTrap(0),
        cabhPsDevInitTrap(1),
        cabhPsDevInitRetryTrap(2),
        cabhPsDevDHCPFailTrap(3),
        cabhPsDevSwUpgradeInitTrap(4),
        cabhPsDevSwUpgradeFailTrap(5),
        cabhPsDevSwUpgradeSuccessTrap(6),
        cabhPsDevSwUpgradeCVCFailTrap(7),
        cabhPsDevTODFailTrap(8),
        cabhPsDevCdpWanDataIpTrap(9),
        cabhPsDevCdpThresholdTrap(10),
        cabhPsDevCspTrap(11),
        cabhPsDevCapTrap(12),
        cabhPsDevCtpTrap(13),
        cabhPsDevProvEnrollTrap(14),
        cabhPsDevCdpLanIpPoolTrap(15),
        cabhPsDevUpnpMultiplePHTrap(16)
    }
    MAX-ACCESS read-write
    STATUS current
    DESCRIPTION
            "The object is used to enable PS notifications.
            From left to right, the set bit indicates
            the corresponding PS notification is enabled.
            For example, if the first bit is set, then
            cabhPsDevInitTLVUnknownTrap is enabled.
            If the bit is zero, the trap is disabled."
    DEFVAL { '0000'h }
    ::= { cabhPsDevBase 13 }

--
--    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      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(0|20))
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
            "Hash of the contents of the PS config file, which is
            calculated by the NMS and sent to the PS. For the SHA-1
            authentication algorithm, the hash length is 160 bits. This
            hash value is encoded in binary format."
    ::= { 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 6 }

cabhPsDevProvConfigTLVRejected OBJECT-TYPE
    SYNTAX      INTEGER (0..16383)
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "Number of TLVs rejected in config file."
    ::= { cabhPsDevProv 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 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 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 10 }

cabhPsDevProvCorrelationId OBJECT-TYPE
    SYNTAX      Integer32
    MAX-ACCESS  read-only
    STATUS      deprecated
    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 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 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 13 }

--============================================================
```

```
--
--   PS Device Profile Group
--
--  The cabhPsDevPsProfile contains the Residential Gateway's
--  device attributes. This set of attributes is analogous to
--  some attributes of the BP Device profile.
--
--================================================================

cabhPsDevPsDeviceType OBJECT-TYPE
    SYNTAX      SnmpAdminString (SIZE(1..32))
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "The type of device, as defined in the CableHome
            specifications (Residential Gateway Device or CableHome
            Host Device), that implements this OID."
    DEFVAL { "CableHome Residential Gateway" }
    ::= { cabhPsDevPsAttrib 1 }

cabhPsDevPsManufacturerUrl OBJECT-TYPE
    SYNTAX      SnmpAdminString (SIZE(0..32))
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "Universal Resource Locator to the Residential Gateway
            device manufacturer's web site."
    ::= { cabhPsDevPsAttrib 3 }

cabhPsDevPsModelUrl OBJECT-TYPE
    SYNTAX      SnmpAdminString (SIZE(0..32))
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "Universal Resource Locator to the web site describing this
            CableHome compliant residential gateway device."
    ::= { cabhPsDevPsAttrib 7 }

cabhPsDevPsModelUpc OBJECT-TYPE
    SYNTAX      SnmpAdminString (SIZE(0..32))
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "Universal Product Code of the CableHome compliant
            residential gateway device.
            See: Uniform Code Council  www.uc-council.org"
    ::= { cabhPsDevPsAttrib 8 }

--================================================================
--
--   CableHome Host/BP Device Profile Table
--
--  The cabhPsDevBpProfile contains the list of the CableHome Host
--  device attributes provided to the PS by BPs passing their Device
--  Profile XML schema via SOAP/HTTP.
--
--================================================================

cabhPsDevBpProfileTable OBJECT-TYPE
    SYNTAX SEQUENCE OF CabhPsDevBpProfileEntry
    MAX-ACCESS  not-accessible
    STATUS      obsolete
    DESCRIPTION
            "This table contains the information for the CableHome Host
```

```
            Device Profiles. Attributes of a device make up a Device
            Profile."
    ::= { cabhPsDevBpAttrib 1 }


cabhPsDevBpProfileEntry OBJECT-TYPE
    SYNTAX      CabhPsDevBpProfileEntry
    MAX-ACCESS  not-accessible
    STATUS      obsolete
    DESCRIPTION
            "The table that describes the CableHome Host Device
            Profile."
    INDEX { cabhPsDevBpIndex }
    ::= { cabhPsDevBpProfileTable 1 }

CabhPsDevBpProfileEntry ::= SEQUENCE {
        cabhPsDevBpIndex                     INTEGER,
        cabhPsDevBpDeviceType                SnmpAdminString,
        cabhPsDevBpManufacturer              SnmpAdminString,
        cabhPsDevBpManufacturerUrl           SnmpAdminString,
        cabhPsDevBpSerialNumber              SnmpAdminString,
        cabhPsDevBpHardwareVersion           SnmpAdminString,
        cabhPsDevBpHardwareOptions           SnmpAdminString,
        cabhPsDevBpModelName                 SnmpAdminString,
        cabhPsDevBpModelNumber               SnmpAdminString,
        cabhPsDevBpModelUrl                  SnmpAdminString,
        cabhPsDevBpModelUpc                  SnmpAdminString,
        cabhPsDevBpModelSoftwareOs           SnmpAdminString,
        cabhPsDevBpModelSoftwareVersion      SnmpAdminString,
        cabhPsDevBpLanInterfaceType          IANAifType,
        cabhPsDevBpNumberInterfacePriorities INTEGER,
        cabhPsDevBpPhysicalLocation          SnmpAdminString,
        cabhPsDevBpPhysicalAddress           PhysAddress
        }

cabhPsDevBpIndex OBJECT-TYPE
    SYNTAX      INTEGER (1..65535)
    MAX-ACCESS  not-accessible
    STATUS      obsolete
    DESCRIPTION
            "Integer index into the CableHome Host Device Profile
            Table."
    ::= { cabhPsDevBpProfileEntry 1 }

cabhPsDevBpDeviceType OBJECT-TYPE
    SYNTAX      SnmpAdminString (SIZE(0..32))
    MAX-ACCESS  read-only
    STATUS      obsolete
    DESCRIPTION
            "The type of device, as defined by the CableHome
            specifications (CableHome Residential Gateway or CableHome
            Host Device), that passed the Device Profile whose
            information is made available through this table row."
    DEFVAL { "CableHome Host" }
    ::= { cabhPsDevBpProfileEntry 2 }

cabhPsDevBpManufacturer OBJECT-TYPE
    SYNTAX      SnmpAdminString (SIZE(0..32))
    MAX-ACCESS  read-only
    STATUS      obsolete
    DESCRIPTION
            "The name of the CableHome Host Device's manufacturer."
    DEFVAL { "" }
    ::= { cabhPsDevBpProfileEntry 3 }
```

```
cabhPsDevBpManufacturerUrl OBJECT-TYPE
    SYNTAX      SnmpAdminString (SIZE(0..32))
    MAX-ACCESS  read-only
    STATUS      obsolete
    DESCRIPTION
            "Universal Resource Locator to the CableHome Host device
            manufacturer's web site."
    DEFVAL { "" }
    ::= { cabhPsDevBpProfileEntry 4 }

cabhPsDevBpSerialNumber OBJECT-TYPE
    SYNTAX      SnmpAdminString (SIZE(0..32))
    MAX-ACCESS  read-only
    STATUS      obsolete
    DESCRIPTION
            "The serial number assigned by the manufacturer for this
            CableHome Host Device."
    DEFVAL { "" }
    ::= { cabhPsDevBpProfileEntry 5 }

cabhPsDevBpHardwareVersion OBJECT-TYPE
    SYNTAX      SnmpAdminString (SIZE(0..32))
    MAX-ACCESS  read-only
    STATUS      obsolete
    DESCRIPTION
            "The hardware version number assigned by the manufacturer
            for this CableHome Host Device."
    DEFVAL { "" }
    ::= { cabhPsDevBpProfileEntry 6 }

cabhPsDevBpHardwareOptions OBJECT-TYPE
    SYNTAX      SnmpAdminString (SIZE(0..32))
    MAX-ACCESS  read-only
    STATUS      obsolete
    DESCRIPTION
            "The hardware options implemented on this CableHome Host
            Device."
    DEFVAL { "" }
    ::= { cabhPsDevBpProfileEntry 7 }

cabhPsDevBpModelName OBJECT-TYPE
    SYNTAX      SnmpAdminString (SIZE(0..32))
    MAX-ACCESS  read-only
    STATUS      obsolete
    DESCRIPTION
            "The model name assigned by the manufacturer for this
            CableHome Host Device."
    DEFVAL  { "" }
    ::= { cabhPsDevBpProfileEntry 8 }

cabhPsDevBpModelNumber OBJECT-TYPE
    SYNTAX      SnmpAdminString (SIZE(0..32))
    MAX-ACCESS  read-only
    STATUS      obsolete
    DESCRIPTION
            "The model number assigned by the manufacturer for this
            CableHome Host Device."
    DEFVAL { "" }
    ::= { cabhPsDevBpProfileEntry 9 }

cabhPsDevBpModelUrl OBJECT-TYPE
    SYNTAX      SnmpAdminString (SIZE(0..32))
    MAX-ACCESS  read-only
```

```
    STATUS      obsolete
    DESCRIPTION
            "The Universal Resource Locator to the web site describing
            this CableHome Host Device model."
    DEFVAL { "" }
    ::= { cabhPsDevBpProfileEntry 10 }

cabhPsDevBpModelUpc OBJECT-TYPE
    SYNTAX      SnmpAdminString (SIZE(0..32))
    MAX-ACCESS  read-only
    STATUS      obsolete
    DESCRIPTION
            "Universal Product Code of the CableHome Host Device."
    DEFVAL { "" }
    ::= { cabhPsDevBpProfileEntry 11 }

cabhPsDevBpModelSoftwareOs OBJECT-TYPE
    SYNTAX      SnmpAdminString (SIZE(0..32))
    MAX-ACCESS  read-only
    STATUS      obsolete
    DESCRIPTION
            "Software operating system implemented on the CableHome
            Host Device."
    DEFVAL { "" }
    ::= { cabhPsDevBpProfileEntry 12 }

cabhPsDevBpModelSoftwareVersion OBJECT-TYPE
    SYNTAX      SnmpAdminString (SIZE(0..32))
    MAX-ACCESS  read-only
    STATUS      obsolete
    DESCRIPTION
            "Version of the operating system implemented on the
            CableHome Host Device."
    DEFVAL { "" }
    ::= { cabhPsDevBpProfileEntry 13 }

cabhPsDevBpLanInterfaceType OBJECT-TYPE
    SYNTAX      IANAifType
    MAX-ACCESS  read-only
    STATUS      obsolete
    DESCRIPTION
            "The ifType for the LAN Interface implemented on the
            CableHome Host Device."
    REFERENCE
            "http://www.iana.org/assignments/ianaiftype-mib."
    DEFVAL { other }
    ::= { cabhPsDevBpProfileEntry 14 }

cabhPsDevBpNumberInterfacePriorities OBJECT-TYPE
    SYNTAX      INTEGER (1..8)
    MAX-ACCESS  read-only
    STATUS      obsolete
    DESCRIPTION
            "Number of QoS priorities supported by the LAN technology
            (Data Link Layer) implemented in the CableHome Host
            Device."
    DEFVAL { 1 }
    ::= { cabhPsDevBpProfileEntry 15 }

cabhPsDevBpPhysicalLocation OBJECT-TYPE
    SYNTAX      SnmpAdminString (SIZE(0..32))
    MAX-ACCESS  read-only
    STATUS      obsolete
    DESCRIPTION
```

```
            "Physical location of the CableHome Host Device."
    DEFVAL { "" }
    ::= { cabhPsDevBpProfileEntry 16 }

cabhPsDevBpPhysicalAddress OBJECT-TYPE
    SYNTAX      PhysAddress (SIZE (0..16))
    MAX-ACCESS  read-only
    STATUS      obsolete
    DESCRIPTION
            "The CableHome Host Device's hardware address."
    DEFVAL { ''h }
    ::= { cabhPsDevBpProfileEntry 17 }

--================================================================
--
--  LAN IP Traffic Statistics Table
--
--  The cabhPsDevLanIpTrafficTable contains the Traffic Statistics
--  for all LAN IP Devices connected to the PS. When the PS learns a
--  new LAN IP address, an entry is added to this table.
--
--================================================================

cabhPsDevLanIpTrafficCountersReset OBJECT-TYPE
    SYNTAX      INTEGER
    {
        clearCounters(1),
        clearTable(2)
    }
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
            "Setting this object to clearCounters(1) resets all the
            traffic statistic counter entries to zero in the
            cabhPsDevLanIpTrafficTable. Setting this object to
            clearTable(2) removes all entries in the
            cabhPsDevLanIpTrafficTable. Reading this object always
            returns clearCounters(1)."
    DEFVAL { clearCounters }
    -- Default read value
    ::= { cabhPsDevStats 1 }

cabhPsDevLanIpTrafficCountersLastReset OBJECT-TYPE
    SYNTAX      TimeStamp
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "The value of sysUpTime when
            cabhPsDevLanIpTrafficCountersReset was last written to.
            Zero if never written to."
    ::= { cabhPsDevStats 2 }

cabhPsDevLanIpTrafficEnabled OBJECT-TYPE
    SYNTAX      TruthValue
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
            "Setting this object to true(1) turns on the IP traffic
            counters. Setting this object to false(2) turns off the IP
            traffic counters."
    DEFVAL { false } -- IP traffic counters are off by default
    ::= { cabhPsDevStats 3 }

cabhPsDevLanIpTrafficTable OBJECT-TYPE
```

```
    SYNTAX      SEQUENCE OF CabhPsDevLanIpTrafficEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "This table contains IP-layer Traffic Statistics for all
            LAN IP Devices connected to the PS."
    ::= { cabhPsDevStats 4 }

cabhPsDevLanIpTrafficEntry OBJECT-TYPE
    SYNTAX      CabhPsDevLanIpTrafficEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "List of Traffic Statistics for LAN IP Devices."
    INDEX { cabhPsDevLanIpTrafficIndex }
    ::= { cabhPsDevLanIpTrafficTable 1 }

CabhPsDevLanIpTrafficEntry ::= SEQUENCE {
    cabhPsDevLanIpTrafficIndex              INTEGER,
    cabhPsDevLanIpTrafficInetAddressType    InetAddressType,
    cabhPsDevLanIpTrafficInetAddress        InetAddress,
    cabhPsDevLanIpTrafficInOctets           ZeroBasedCounter32,
    cabhPsDevLanIpTrafficOutOctets          ZeroBasedCounter32
    }

cabhPsDevLanIpTrafficIndex OBJECT-TYPE
    SYNTAX      INTEGER   (1..65535)
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "The Index into the LAN IP Traffic Statistics Table."
    ::= { cabhPsDevLanIpTrafficEntry 1 }

cabhPsDevLanIpTrafficInetAddressType OBJECT-TYPE
    SYNTAX      InetAddressType
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "The type of IP address assigned to the LAN IP device to
            which the statistics in this table row apply.  IP version
            4 is typically used."
    DEFVAL { ipv4 }
    ::= { cabhPsDevLanIpTrafficEntry 2 }

cabhPsDevLanIpTrafficInetAddress OBJECT-TYPE
    SYNTAX      InetAddress
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "The IP address of the LAN IP device to which the
            statistics in this table row apply.  An IPv4 IP
            address is typically used."
    ::= { cabhPsDevLanIpTrafficEntry 3 }

cabhPsDevLanIpTrafficInOctets OBJECT-TYPE
    SYNTAX      ZeroBasedCounter32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "The total number of octets the PS forwarded from the WAN
            interfaces to the LAN IP device associated with the value
            of cabhPsDevLanIpTrafficInetAddress. This counter object
            does not include LAN-to-LAN traffic."
    ::= { cabhPsDevLanIpTrafficEntry 4 }
```

```
cabhPsDevLanIpTrafficOutOctets OBJECT-TYPE
    SYNTAX      ZeroBasedCounter32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "The total number of octets the PS forwarded from the LAN
            IP device associated with the value of
            cabhPsDevLanIpTrafficInetAddress, to the WAN interfaces.
            This counter object does not include LAN-to-LAN traffic."
    ::= { cabhPsDevLanIpTrafficEntry 5 }

--========================================================================
--
--    CableHome Interface Access Control Table
--
--  The cabhPsDevAccessControlTable lists the physical addresses
--  of all LAN IP Devices for which the PS will forward traffic to
--  or from an interface type for which the Table is enabled.
--  If an interface type is enabled, the PS will not forward traffic
--  to or from any device on that interface whose physical address
--  is not listed in the Access Control Table. If an interface type
--  is disabled, the PS does apply forwarding restrictions based on
--  entires of the Access Control Table.
--
--========================================================================

cabhPsDevAccessControlEnable OBJECT-TYPE
    SYNTAX      BITS {
        hpna(0),  -- most significant bit
        ieee80211(1),
        ieee8023(2),
        homeplug(3),
        usb(4),
        ieee1394(5),
        scsi(6),
        other(7)
        }
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
            "This object specifies the interface type(s) for which the
            PSDev Access Control Table access rules are enabled. If a
            bit field is set to 1, the PS MUST only forward traffic
            received through that interface type if the source physical
            address is an entry in the cabhPsDevAccessControlTable. If
            a bit field is set to 1, the PS MUST only forward traffic
            destined to a device on that interface type if the
            destination physical address is an entry in the
            cabhPsDevAccessControlTable. If the bit field for an
            interface type is not set, i.e., if it is equal to 0, the
            PS MUST NOT apply forwarding restrictions for that
            interface type based on the Access Control Table. The PS
            MUST implement cabhPsDevAccessControlEnable for bit 1
            (wireless LAN) and for bit 3 (HomePlug). If the PS does not
            implement cabhPsDevAccessControlEnable for any of the other
            defined bits, the PS MUST return inconsistent value error,
            and not allow the bit to be set, if an attempt is made to
            set a bit that is not implemented.

            If the PS implements a HomePNA interface and implements the
            PSDev Access Control Table enable functionality for the
            HomePNA interface, then if bit 0 is set, the PS MUST apply
            PSDev Access Control Table access rules to any PS interface
```

```
        of  IANAifType 220 (Home Phoneline Networking Alliance). If
        the PS does not implement PSDev Access Control Table enable
        functionality for the HomePNA interface, and an attempt is
        made to set bit 0 to value '1', the PS MUST return
        'Inconsistent Value' error and MUST NOT set bit 0 to
        value '1'.

        If bit 1 (ieee80211) is set, the PS MUST apply PSDev
        Access Control Table access rules to any PS interface of
        IANAifType 71 (radio spread spectrum).

        If the PS implements an IEEE 802.3/CSMA-CD interface and
        implements the PSDev Access Control Table enable
        functionality for the IEEE 802.3/CSMA-CD interface, then
        if bit 2 is set, the PS MUST apply PSDev Access Control
        Table access rules to any PS interface of  IANAifType 6
        (ethernetCsmacd). If the PS does not implement PSDev
        Access Control Table enable functionality for a
        IEEE 802.3/CSMA-CD interface, and an attempt is made to set
        bit 2 to value '1', the PS MUST return 'Inconsistent
        Value' error and MUST NOT set bit 2 to value '1'.

        If bit 3 (homeplug) is set, the PS MUST apply PSDev
        Access Control Table access rules to any PS HomePlug
        Powerline Alliance (HomePlug) interface as defined by
        HomePlug Powerline Alliance (www.homeplug.org).

        If the PS implements a USB interface and implements the
        PSDev Access Control Table enable functionality for the
        USB interface, then if bit 4 is set, the PS MUST apply
        PSDev Access Control Table access rules to any PS
        interface of  IANAifType 160 (USB). If the PS does not
        implement PSDev Access Control Table enable functionality
        for the USB interface, and an attempt is made to set
        bit 4 to value '1', the PS MUST return 'Inconsistent Value'
        error and MUST NOT set bit 4 to value '1'.

        If the PS implements an IEEE 1394 interface and
        implements the PSDev Access Control Table enable
        functionality for the IEEE 1394 interface, then if bit 5
        is set, the PS MUST apply PSDev Access Control Table
        access rules to any PS interface of  IANAifType 144
        (IEEE 1394 High Performance Serial Bus). If the PS does
        not implement PSDev Access Control Table enable
        functionality for the IEEE 1394 interface, and an attempt
        is made to set bit 5 to value '1', the PS MUST return
        'Inconsistent Value' error and MUST NOT set bit 5 to
        value '1'.

        If the PS implements a SCSI interface and implements the
        PSDev Access Control Table enable functionality for the
        SCSI interface, then if bit 6 is set, the PS MUST apply
        PSDev Access Control Table access rules to any PS SCSI-2
        or SCSI-3 interface. If the PS does not implement PSDev
        Access Control Table enable functionality for the SCSI
        interface, and an attempt is made to set bit 6 to value
        '1', the PS MUST return 'Inconsistent Value' error and
        MUST NOT set bit 6 to value '1'.

        If bit 7 (other) is set, the PS MAY apply PSDev Access
        Control Table filter access to any PS interface of a type
        other than the types defined by bits 0 - 6."
    DEFVAL { '00'h }  -- null, all interface types disabled
    ::= { cabhPsDevAccessControl 1 }
```

```
cabhPsDevAccessControlTable OBJECT-TYPE
    SYNTAX      SEQUENCE OF CabhPsDevAccessControlEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "This table contains a list of the physical addresses of
            LAN IP Devices to and from which the PS will forward
            traffic through a LAN interface if
            cabhPsDevAccessControlEnable is enabled(1) for that
            interface type."
    REFERENCE
            "CableHome specification, Packet Handling & Address
            Translation section."
    ::= { cabhPsDevAccessControl 2 }

cabhPsDevAccessControlEntry OBJECT-TYPE
    SYNTAX      CabhPsDevAccessControlEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "List of the physical addresses for LAN IP Devices
            to and from which the PS will forward traffic when
            the PSDev Access Control Table is enabled."
    INDEX { cabhPsDevAccessControlIndex }
    ::= { cabhPsDevAccessControlTable 1 }

CabhPsDevAccessControlEntry ::= SEQUENCE {
    cabhPsDevAccessControlIndex     INTEGER,
    cabhPsDevAccessControlPhysAddr  PhysAddress,
    cabhPsDevAccessControlRowStatus RowStatus
    }

cabhPsDevAccessControlIndex OBJECT-TYPE
    SYNTAX      INTEGER (1..65535)
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "Integer index into the CableHome PSDev Access Control
            Table."
    ::= { cabhPsDevAccessControlEntry 1 }

cabhPsDevAccessControlPhysAddr OBJECT-TYPE
    SYNTAX      PhysAddress (SIZE (1..16))
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
            "The physical address of the LAN IP Device for which the PS
            will forward traffic when the PSDev Access Control
            Table is enabled. The PS will not forward traffic
            from any LAN IP Device whose physical address is
            not an entry of the PSDev Access Control Table when the
            PSDev Access Control Table is enabled for the
            corresponding interface."
    ::= { cabhPsDevAccessControlEntry 2 }

cabhPsDevAccessControlRowStatus OBJECT-TYPE
    SYNTAX      RowStatus
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
            "The RowStatus interlock for the creation and deletion
            of a cabhPsDevAccessControlTable entry. Any writable
            object in each row of the cabhPsDevAccessControlTable
```

```
            can be modified at any time while the row is active(1)."
    ::= { cabhPsDevAccessControlEntry 3 }

--========================================================================
--
--    CableHome Miscellaneous MIB
--
--  This branch of cabhPsDevMib contains extensions related to
--  functionalities defined for other standards bodies or outside
--  of CableHome fully defined features.
--
--========================================================================

--========================================================================
--
--    CableHome User Interface Miscellaneous MIB
--
--  PS MIB objects for controlling features of the CableHome compliant
--  residential gateways User Interface (UI) if present.
--
--========================================================================

cabhPsDevUILogin OBJECT-TYPE
    SYNTAX      OCTET STRING (SIZE(0..32))
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
            "This parameter specifies the value of the user login name
            required for access to the CableHome compliant residential
            gateway device's user interface."
    ::= { cabhPsDevUI 1 }

cabhPsDevUIPassword OBJECT-TYPE
    SYNTAX      OCTET STRING (SIZE(4..32))
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
            "This parameter specifies the value of the user password
            required for access to the CableHome compliant residential
            gateway device's user interface."
    ::= { cabhPsDevUI 2 }

cabhPsDevUISelection OBJECT-TYPE
    SYNTAX      INTEGER {
                    manufacturerLocal(1),
                    cableOperatorLocal(2),
                    cableOperatorServer(3),
                    disabledUI(4)
                }
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
            "Indicates the type of Web user interface (UI)
            to present to the user if Web interface is supported:
            manufacturerLocal:
              PS uses the vendor UI shipped with the device.
            cableOperatorLocal:
              PS uses a cable operator defined UI interface.
              To operate properly, it should require a special code
              image downloaded into the PS.  By default, if no cable
              operator UI is being defined, selecting this option
              points to 'manufacturerLocal' selection.
            cableOperatorServer:
              PS redirects HTTP requests to its UI to the URL specified
```

```
              in cabhPsDevUIServerUrl.
            disabledUI:
              PS responds to HTTP requests to its UI with an HTTP page
              containing the value of
              cabhPsDevUISelectionDisabledBodyText as the body tag;
              or with a vendor specific message or HTTP error if that
              value is null."
    DEFVAL { manufacturerLocal }
    ::= { cabhPsDevUI 3 }

cabhPsDevUIServerUrl OBJECT-TYPE
    SYNTAX      SnmpAdminString (SIZE(0..255))
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
            "The Uniform Resource Locator (URL) provisioned by the cable
            operator to which the PS re-directs the subscriber's LAN IP Device
            for presentation of the PS User Interface when the value of
            cabhPsDevUISelection is cableOperatorServer(3). This object is valid
            and applicable only when the value of cabhPsDevUISelection is
            cableOperatorServer(3)."
    DEFVAL { "" }
    ::= { cabhPsDevUI 4 }

cabhPsDevUISelectionDisabledBodyText OBJECT-TYPE
    SYNTAX      SnmpAdminString (SIZE(0..255))
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
            "Default text for the HTTP body tag to include in the
            response to UI requests when the object
            cabhPsDevUISelection is set to 'disabledUI'.
            An example of a body tag is below:
            <body>Feature currently disabled by Cable Operator</body>."
    ::= { cabhPsDevUI 5 }

-- ====================================================
-- IEEE802dot11-MIB CableHome extension
-- ====================================================

cabhPsDev802dot11BaseTable OBJECT-TYPE
    SYNTAX      SEQUENCE OF CabhPsDev802dot11BaseEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "CableHome specifics controls for 80211 wireless
            interfaces."
    ::= { cabhPsDev802dot11 1 }

cabhPsDev802dot11BaseEntry OBJECT-TYPE
    SYNTAX      CabhPsDev802dot11BaseEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "An entry in cabhPsDev802dot11BaseTable associated to a
            wireless interface of IANAifType ieee80211.(71)"
    INDEX { ifIndex }
    ::= { cabhPsDev802dot11BaseTable 1 }

CabhPsDev802dot11BaseEntry ::=
    SEQUENCE {
              cabhPsDev802dot11BaseSetToDefault     TruthValue,
              cabhPsDev802dot11BaseLastSetToDefault TimeStamp,
              cabhPsDev802dot11BaseAdvertiseSSID    TruthValue,
```

```
              cabhPsDev802dot11BasePhyCapabilities  BITS,
              cabhPsDev802dot11BasePhyOperMode      INTEGER
             }

cabhPsDev802dot11BaseSetToDefault OBJECT-TYPE
    SYNTAX      TruthValue
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
            "When set  to true(1), the PS MUST reset to default values
            the MIB objects of IEEE802dot11-MIB module and others under
            cabhPsDev802dot11 for this entry related IfIndex.
            Reading this object always return false(2)."
    DEFVAL { false }
    ::= { cabhPsDev802dot11BaseEntry 1 }

cabhPsDev802dot11BaseLastSetToDefault OBJECT-TYPE
    SYNTAX      TimeStamp
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "The value of sysUpTime when
            cabhPsDev802dot11MIBSetToDefault was last set to true.
            Zero if never reset."
    ::= { cabhPsDev802dot11BaseEntry 2 }

cabhPsDev802dot11BaseAdvertiseSSID OBJECT-TYPE
    SYNTAX      TruthValue
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
            "When set to false(2) the PS does not advertise the BSS
            SSID in a proprietary manner. To avoid interoperability
            problems and service disruption, it is RECOMMENDED to set
            this object always to true. This feature does not provide
            any security, and does not prevent Wireless Stations to
            obtain the SSID by sniffing frames from other stations in
            the ESS. If the device does not support the feature of
            turning on/off the SSID advertisement, this object always
            reports 'true(1)' and reports the error 'wrongValue' when
            set to 'false(2)."
    DEFVAL { true }
    ::= { cabhPsDev802dot11BaseEntry 3 }

cabhPsDev802dot11BasePhyCapabilities OBJECT-TYPE
    SYNTAX      BITS {
                    --ieee80211DSSS(0) , not interest
                      ieee80211a(0),
                      ieee80211b(1),
                      ieee80211g(2)
                    --ieee80211FHSS(8),
                    --ieee80211IR(16)
                    --values with comments are not requirements
                    --included for completeness of 80211 spec.
                    }
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "Indicates the PHY capabilities of the wireless interface."
    ::= { cabhPsDev802dot11BaseEntry 4 }

cabhPsDev802dot11BasePhyOperMode OBJECT-TYPE
    SYNTAX      INTEGER {
                        ieee80211a(1),
```

```
                        ieee80211b(2),
                        ieee80211g(4),
                        ieee80211bg(24)
                       }
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
            "Indicates the PHY mode of operation being set for the
            wireless interface. Setting this object will update the
            value of dot11PhyType. Accordingly (if implemented), as
            well as the object dot11OperationalRateSet to the 80211
            mandatory rates for dot11PhyType.

            It is left to vendors the option to update the values of
            PS optional dot11SupportedDataRatesTxEntry and
            dot11SupportedDataRatesRxEntry tables based on the
            operational mode.

            In the case of selecting ieee80211bg(14), dot11PhyType
            reports erp(6) and dot11OperationalRateSet should report
            HRDSSS and ERP mandatory rates and in addition 54 Mbit/s rate
            if supported by PS. e.g. : (this example assumes 54 Mbit/s
            OFDM is supported.
            HR-DSSS :
              Mandatory:
                     1 Mbit/s '80'H + '01'H
                     2 Mbit/s '80'H + '02'H
                     5.5 Mbit/s '80'H + '0B'H
                    11 Mbit/s '80'H + '16'H
            ERP :
              Mandatory:
                     6 Mbit/s '80'H + '0C'H
                    12 Mbit/s '80'H + '18'H
                    24 Mbit/s '80'H + '30'H
     (if supported) 54 Mbit/s '80'H + '6C'
              Optional:
                    22 Mbit/s '00'H + '2C'H
                    33 Mbit/s '00'H + '42'H
                    18 Mbit/s '00'H + '24'H
                    36 Mbit/s '00'H + '48'H
                    48 Mbit/s '00'H + '60'H

          Combined operational rates in :

            dot11OperationalRateSet value in rate order regardless
            of '80'H flag and using dots for clarity :
            + means flagged '80'H, - not flagged.
          Rates Mbit/s: +1,+2,+5.5,+6,+11,+12,-18,-22,+24,-33,-36,-48,+54
                Hex:  '81.82.8B.8C.96.98. 24.2C.B0.48.42. 60.EC'H

            The default value of this object is left to the vendor to
            accommodate the factory defaults for the device."
    REFERENCE
            "IEEE Std 802.11, 1999 Edition,
            IEEE Std 802.11a-1999,
            IEEE Std 802.11b-1999/Cor 1-2001,
            IEEE Std 802.11g-2003."
    ::= { cabhPsDev802dot11BaseEntry 5 }

-- ============================================================
-- IEEE802dot11MIB CableHome extension for security configuration
-- ============================================================

cabhPsDev802dot11SecTable OBJECT-TYPE
```

```
    SYNTAX      SEQUENCE OF CabhPsDev802dot11SecEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "CableHome specifics controls for configuring the
            security mechanisms of 80211 wireless interfaces."
    ::= { cabhPsDev802dot11 2 }

cabhPsDev802dot11SecEntry OBJECT-TYPE
    SYNTAX      CabhPsDev802dot11SecEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "An entry in cabhPsDev802dot11SecTable associated to a
            wireless interface of IANAifType ieee80211(71)."
    INDEX { ifIndex }
    ::= { cabhPsDev802dot11SecTable 1 }

CabhPsDev802dot11SecEntry ::=
    SEQUENCE {
              cabhPsDev802dot11SecCapabilities              BITS,
              cabhPsDev802dot11SecOperMode                  BITS,
              cabhPsDev802dot11SecPassPhraseToWEPKey        OCTET STRING,
              cabhPsDev802dot11SecUsePassPhraseToWEPKeyAlg  TruthValue,
              cabhPsDev802dot11SecPSKPassPhraseToKey        OCTET STRING,
              cabhPsDev802dot11SecWPAPreSharedKey           OCTET STRING,
              cabhPsDev802dot11SecWPARekeyTime              Unsigned32,
              cabhPsDev802dot11SecControl                   INTEGER,
              cabhPsDev802dot11SecCommitStatus              INTEGER
             }

cabhPsDev802dot11SecCapabilities OBJECT-TYPE
    SYNTAX      BITS {
                      wep64(0),
                      wep128(1),
                      wpaPSK(2)
                    --wpa2PSK(3)
                     }
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "The PS capabilities for Authentication and encryption used
            to authenticate 802.11 clients."
    ::= { cabhPsDev802dot11SecEntry 1 }

cabhPsDev802dot11SecOperMode OBJECT-TYPE
    SYNTAX      BITS {
                      wep64(0),
                      wep128(1),
                      wpaPSK(2)
                   -- wpa2PSK(3)
                     }
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
            "Indicates the Authentication and encryption mechanism to
            be enabled for the users and advertised in Beacon messages.
            Bits set to this object and not supported by the PS in
            cabhPsDev802dot11SecCapabilities are set to '0' without
            failing the SNMP set. Setting two bits that the PS does not
            support in combination returns an error 'wrongValue'.
            In particular:
              Setting to '1' both wep64(0)and wep128(1) bits returns an
                error'wrongValue'.
```

```
              Setting a combination of WEP bits (wep64(0) or wep128(1))
                and wpaPSK bit returns is not a mandatory requirement,
                therefore an error 'wrongValue' may be reported.

            Setting any bit to '1' must not affect the value of object
            dot11PrivacyInvoked.

            If dot11PrivacyInvoked is set to 'false', the 80211 WEP
            security mechanism is disabled (see dot11PrivacyInvoked
            description) and the value of this object is not used.

            Setting the wpaPSK(2) bit to '1' indicates the usage of
            WPA-PSK TKIP.

            Note that to enable the PSK security mechanism, the value
            of cabhPsDev802dot11SecWPAPreSharedKey must be a non-zero
            length string."
    ::= { cabhPsDev802dot11SecEntry 2 }

cabhPsDev802dot11SecPassPhraseToWEPKey OBJECT-TYPE
    SYNTAX      OCTET STRING (SIZE(0|5..63))
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
            "The Password used for PS to derive WEP encryption keys.
            After a successful set, the values of
            dot11WEPDefaultKeyValue are populated as described below:

            For wep64:
              If cabhPsDev802dot11SecOperMode wep64 bit is set to '1'
              This object value (x) is used as a generator of a 4-octet
              seed.

              seed[i%4] = XOR(seed[i%4],x[i]); i from 1 to len(x) -1

            The values of the four dot11WEPDefaultKeyValue are
            calculated as indicated below :
              loop j 1..4
              loop k 0..4
              seed = seed * (((26*8+1)*256-1)*4+1) + 2531011
              The value is always truncated at 32 bits.
              OCTETk = (seed >> 16 )& 0xFF  -lowest octet-
              end loop
              dot11WEPDefaultKeyValue(j) = OCTET0,OCTET1, ... OCTET4
              end loop

              Note that seed value is constantly re-computed when
              calculating each octet of each default WEP key.

            For wep128:
              If cabhPsDev802dot11SecOperMode wep128 bit is set to '1'
              This object value (x) fills a 64-octet buffer y :
              y = x,x,x...up to 64 octets.
              Calculate the 128-bit MD5 digest of y
              the values of all dot11WEPDefaultKeyValue (1..4)
              are calculated by truncating the first 13 octets
              of MD5y.

            dot11WEPDefaultKeyValue = MD5y0,MD5y1, .. MD5y12

              This object value is normally read by issuing SNMP
              request PDUs. This object can be cleared with an SNMP
              SET to an empty string Value and the PS MUST not update
              the type of keys being set to '1' in
```

```
    cabhPsDev802dot11SecOperMode.

  If cabhPsDev802dot11SecUsePassPhraseToKeyAlg is set to
  false(2), the behaviour of a SET to this object depends
  on the bits set for cabhPsDev802dot11SecOperMode as
  follows:


    If cabhPsDev802dot11SecOperMode bit wep64 is set to '1'
    and this object value length is 5 octets, the MIB object
    dot11WEPDefaultKeyValue.1 (WEP key 0) is populated with
    this object value, otherwise an error 'inconsistentValue'
    is reported.

    If cabhPsDev802dot11SecOperMode bit wep128 is set to '1'
    and this object value length is 13 octets, the MIB object
    dot11WEPDefaultKeyValue.1 (WEP key 0) is populated with
    this object value, otherwise an error 'inconsistentValue'
    is reported.

  Vector examples for wep64 and wep128 key derivation:

    Note:
    % refers to the module operation (reminder of the
    quotient of i and 4); XOR is the OR exclusive boolean
    operation.

For wep64:
passphrase:
        'ABCD4321' ( hex code 0x41.42.43.44.34.33.32.31 )


First loop: (octets 0..3)
   XOR (0x00,A) -> XOR(0x00,0x41) -> 0x41
   XOR (0x00,B) -> XOR(0x00,0x42) -> 0x42
   XOR (0x00,C) -> XOR(0x00,0x43) -> 0x43
   XOR (0x00,D) -> XOR(0x00,0x44) -> 0x44

Second loop: (octets 4..7)
   XOR (A,4) -> XOR(0x41,0x34) -> 0x75
   XOR (B,3) -> XOR(0x42,0x33) -> 0x71
   XOR (C,2) -> XOR(0x43,0x32) -> 0x71
   XOR (D,1) -> XOR(0x44,0x31) -> 0x75

  initial seed 0x75717175 -> 1970368885

  DefaultKeys calculation

  key1
  seed : 0x16545E64 -> 2nd MSB byte : 0x54
  seed : 0x41681397 -> 2nd MSB byte : 0x68
  seed : 0x1BE77FFE -> 2nd MSB byte : 0xE7
  seed : 0xAA6996C9 -> 2nd MSB byte : 0x69
  seed : 0xD1523E68 -> 2nd MSB byte : 0x52
  dot11WEPDefaultKeyValue.1 = 0x5468E76952

  key2
  seed : 0x1FFB838B -> 2nd MSB byte : 0xFb
  seed : 0xF9C60022 -> 2nd MSB byte : 0xC6
  seed : 0xAB43A65D -> 2nd MSB byte : 0x43
  seed : 0xE9A35FAC -> 2nd MSB byte : 0xA3
  seed : 0xE7AA2FBF -> 2nd MSB byte : 0xAA
  dot11WEPDefaultKeyValue.2 = 0xFBC643A3AA
```

```
            key3
            seed : 0x6D13CB86 -> 2nd MSB byte : 0x13
            seed : 0x5D8CD431 -> 2nd MSB byte : 0x8C
            seed : 0xCC702630 -> 2nd MSB byte : 0x70
            seed : 0xD78AEC33 -> 2nd MSB byte : 0x8A
            seed : 0x24DC662A -> 2nd MSB byte : 0xDC
            dot11WEPDefaultKeyValue.3 = 0x138C708ADC

            key4
            seed : 0x4F329445 -> 2nd MSB byte : 0x32
            seed : 0x3EC035F4 -> 2nd MSB byte : 0xC0
            seed : 0xF416CCE7 -> 2nd MSB byte : 0x16
            seed : 0x9904940E -> 2nd MSB byte : 0x04
            seed : 0x28969A99 -> 2nd MSB byte : 0x96
            dot11WEPDefaultKeyValue.4 = 0x32C0160496

            For wep128:
            passphrase:
                   'ABCD4321' ( hex code 0x41.42.43.44.34.33.32.31 )
            128-bit MD-5 digest 0xFECBACF05B42F7A138A5F3928E
            dot11WEPDefaultKeyValue.1..4 = 0xFECBACF05B42F7A138A5"
    ::= { cabhPsDev802dot11SecEntry 3 }

cabhPsDev802dot11SecUsePassPhraseToWEPKeyAlg OBJECT-TYPE
    SYNTAX      TruthValue
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
            "When this object value is true(1), the WEP Pass Phrase to
            key mechanism described in
            cabhPsDev802dot11SecPassPhraseToWEPKey applies. When this
            object is set to false(2), the Pass Phrase to WEP Key
            mechanism is ignored and the password is used as WEP key
            to populate the MIB object keydot11WEPDefaultKeyValue
            object as indicated in
            cabhPsDev802dot11SecPassPhraseToWEPKey description."
    DEFVAL { true }
    ::= { cabhPsDev802dot11SecEntry 4 }

cabhPsDev802dot11SecPSKPassPhraseToKey OBJECT-TYPE
    SYNTAX      OCTET STRING (SIZE(8..63))
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
            "The Password used for PS to derive WPA PSK encryption key.
            After a successful set, the values of
            cabhPsDev802dot11SecWPAPreSharedKey are updated as described
            below:

            For wpaPSK:
            If cabhPsDev802dot11SecOperMode wpaPSK bit is set to '1',
            the value of cabhPsDev802dot11SecWPAPreSharedKey is updated
            with the Password Base Key Derivation Function from  the
            Password-based Cryptographic Specification PKCS #5 v2.0
            RFC 2898 (PBKDF2) with the following specific parameters:

            PSK = PBKDF2(PassPhrase, ssid, ssidLength, 4096, 256);
            PassPhrase is the value of this object;
            ssid is the PS SSID value used as the function salt;
            ssidLength is the number of octets of ssid;
            the iterations count is 4096 and the key generation length
            is 256 bits (32 octets).

            This object value is normally read by issuing SNMP request
```

```
            PDUs. This object can be cleared with an SNMP SET to an
            empty string Value and the PS MUST not update the type of
            keys being set to '1' in cabhPsDev802dot11SecOperMode.

            Vector examples for wpaPSK:

            for wpaPSK:
            passphrase:
                   'ABCD4321' ( hex code 0x41.42.43.44.34.33.32.31 )
            SSID: 'ABCD4321' ( hex code 0x41.42.43.44.34.33.32.31 )

            256 bit PBKDF2('ABCD4321', 'ABCD4321', 8, 4096, 32)
            cabhPsDev802dot11SecWPAPreSharedKey =
            0x7C199CF2FEF9AF206C8EE0E9703920C2
              3517068B3F96B011E0F975C9131BDB58"
    ::= { cabhPsDev802dot11SecEntry 5 }

cabhPsDev802dot11SecWPAPreSharedKey OBJECT-TYPE
    SYNTAX      OCTET STRING (SIZE(0|32))
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
            "The Pre-shared key used for the PS when the bit 'wpaPSK'
            is set to '1'. This object can be set directly or derived
            from the password phrase set in
            cabhPsDev802dot11SecPSKPassPhraseToKey. This object is
            meaningful when the bit wpaPSK is set to '1'.

            If the value of this object is the zero-length string, the
            PS must not activate the PSK security mechanism."
    DEFVAL { ''H }
    ::= { cabhPsDev802dot11SecEntry 6 }

cabhPsDev802dot11SecWPARekeyTime OBJECT-TYPE
    SYNTAX      Unsigned32 (1..4294967295)
    UNITS       "seconds"
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
            "Time interval to initiate WPA Group Keys (GTK) updates."
    DEFVAL { 86400 }
    ::= { cabhPsDev802dot11SecEntry 7 }

cabhPsDev802dot11SecControl OBJECT-TYPE
    SYNTAX INTEGER {
        restoreConfig(1),
        commitConfig(2)
        }
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
            "The control for the indexed 80211 device configuration.
            All changes to the cabhPsDev802dot11SecEntry MIB objects
            are reflected when reading the value of the MIB objects;
            however, those changes are NOT applied to the running
            configuration of the indexed 80211 device until they are
            successfully committed via use of the
            cabhPsDev802dot11SecControl object.

            If changes are made to the cabhPsDev802dot11SecEntry MIB
            objects which are not yet successfully committed to the
            indexed 80211 device, the cabhPsDev802dot11SecControl
            object can be used to roll back all changes to the last
            valid 80211 device configuration and discard all
```

```
            intermediate changes.

            restoreConfig - Setting cabhPsDev802dot11SecControl to this
            value will cause any changes to the
            cabhPsDev802dot11SecEntry objects not yet committed be
            reset to the values from the current running
            configuration of the indexed 80211 device.

            commitConfig - Setting cabhPsDev802dot11SecControl to this
            value will cause the indexed 80211 device to validate and
            apply the valid cabhPsDev802dot11SecEntry MIB settings to
            its running configuration. The
            cabhPsDev802dot11SecCommitStatus object will detail the
            status of this operation."
    DEFVAL { restoreConfig }
    ::= { cabhPsDev802dot11SecEntry 8 }

cabhPsDev802dot11SecCommitStatus OBJECT-TYPE
    SYNTAX INTEGER {
        commitSucceeded(1),
        commitNeeded(2),
        commitFailed(3)
        }
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "Indicates the status of committing the current
            cabhPsDev802dot11SecEntry MIB object values to the running
            configuration of the indexed 80211 device.

            commitSucceeded - indicates the current
            cabhPsDev802dot11SecEntry MIB object values are valid and
            have been successfully committed to the running
            configuration of the indexed 80211 device.

            commitNeeded - indicates that the value of one or more
            objects in cabhPsDev802dot11SecEntry MIB group have been
            changed but not yet committed to the running configuration
            of the indexed 80211 device.

            commitFailed - indicates the PS was unable to commit the
            cabhPsDev802dot11SecEntry MIB object values to the running
            configuration of the indexed 80211 device due to conflicts
            in those values."
    DEFVAL { commitSucceeded }
    ::= { cabhPsDev802dot11SecEntry 9 }

-- ============================================================
--
-- UPnP Services
-- Contains CableHome Portal Server UPnP information of LAN hosts
--
-- ============================================================

cabhPsDevUpnpEnabled OBJECT-TYPE
    SYNTAX      TruthValue
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
            "Setting this object to false(1) disables PS UPnP
            services and UPnP MIB objects related functionality.
            When this object reports 'false', any set to
            UPnP read-write or read-create objects returns error
            'InconsistentValue'.  Transitions of this object from
```

```
            'true' to 'false' and vice versa does not alter the content
            of persistent MIB objects and may clear dynamically UPnP
            created entries. This object value persists upon system
            reinitialization."
    DEFVAL { true }
    ::= { cabhPsDevUpnpBase 1 }

cabhPsDevUpnpCommandIpType OBJECT-TYPE
    SYNTAX      InetAddressType
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
            "The type of InetAddress for cabhPsDevUpnpCommandIp."
    DEFVAL { ipv4 }
    ::= { cabhPsDevUpnpCommands 1 }

cabhPsDevUpnpCommandIp OBJECT-TYPE
    SYNTAX      InetAddress
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
            "The IP address of the device for which the UPnP
            information is being requested. This may be an IPv4 or
            IPv6 prefix. When quering specific information about the
            PS itself, the PS router IP address 192.168.0.1
            should be specified ."
    DEFVAL { 'C0A80001'h } -- 192.168.0.1
    ::= { cabhPsDevUpnpCommands 2 }

cabhPsDevUpnpCommand OBJECT-TYPE
    SYNTAX      INTEGER {
        discoveryInfo(1),
        qosDeviceCapabilities(2),
        qosDeviceState(3)
        }
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
            "The type of information to be retrieved from the Upnp
            Devices in the LAN side and stored in
            cabhPsDevUpnpInfoTable.
            The following selections are supported:

            - discoveryInfo:
            PS retrieves the Discovery information of UPnP devices.
            If the Ip address specified in
            cabhPsDevUpnpCommandIp is 255.255.255.255,
            the PS executes an M-search command and then
            retrieves the discovery information of the
            responding devices. The data stored in
            cabhPsDevUpnpInfoTable also contain UPnP
            discovery data of the PS itself.

            - qosDeviceCapabilities:
            This command is executed for unicast address only
            and will trigger the PS to retrieve the QoS device
            information pertaining to QoS capabilities.

            - qosDeviceState:
            This command is executed for unicast address only
            and will trigger the PS to retrieve the QoS device
            information pertaining to QoS Device state."
    DEFVAL { discoveryInfo }
    ::= { cabhPsDevUpnpCommands 3 }
```

```
cabhPsDevUpnpCommandUpdate OBJECT-TYPE
    SYNTAX      TruthValue
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
            "If set to 'true' triggers the execution of the command
            indicated in cabhPsDevUpnpCommand for the host(s) in
            cabhPsDevUpnpCommandIp. Setting to true this object will
            return error 'wrongValue'  if host IP corresponds to
            255.255.255.255 and cabhPsDevUpnpCommand value is not
            'discoveryInfo'. Reading this value always returns 'false'."
    ::= { cabhPsDevUpnpCommands 4 }

cabhPsDevUpnpLastCommandUpdate OBJECT-TYPE
    SYNTAX      TimeTicks
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "The sysUpTime value of the last time the object
            cabhPsDevUpnpLastCommandUpdate was set to 'true'."
    ::= { cabhPsDevUpnpCommands 5 }

cabhPsDevUpnpCommandStatus OBJECT-TYPE
    SYNTAX      INTEGER {
        none(1),
        inProgress(2),
        complete(3),
        failed(4)
        }
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "The status of cabhPsDevUpnpCommandUpdate trigger
            none(1)
            initial state.
            inProgress(2)
            the information is being acquired by the
            device, PS does not change from 'inProgress'
            to the final state (complete, failed)
            until the execution has finished.
            complete(3) The overall execution is finished with
            no error conditions.
            failed(4).
            The UPnP Device has experienced a timeout. In the
            case of multiple devices query
            (cabhPsDevUpnpCommand set to 'discoveryInfo')
            The failed devices are stored with content information
            empty. At system initialization this object returns
            'none'."
    DEFVAL { none }
    ::= { cabhPsDevUpnpCommands 6}

cabhPsDevUpnpInfoTable OBJECT-TYPE
    SYNTAX      SEQUENCE OF CabhPsDevUpnpInfoEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "This table contains QoS related information of LAN
            UPnP devices or the PS itself."
    ::= { cabhPsDevUpnpCommands 7 }

cabhPsDevUpnpInfoEntry OBJECT-TYPE
    SYNTAX      CabhPsDevUpnpInfoEntry
```

```
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "The Indexes for this entries
            Entries are created after setting to 'true' the
            value of cabhPsDevUpnpCommand."
    INDEX { cabhPsDevUpnpInfoIpType, cabhPsDevUpnpInfoIp,
            cabhPsDevUpnpInfoXmlFragmentIndex }
    ::= { cabhPsDevUpnpInfoTable 1 }

CabhPsDevUpnpInfoEntry::= SEQUENCE {
    cabhPsDevUpnpInfoIpType            InetAddressType,
    cabhPsDevUpnpInfoIp                InetAddress,
    cabhPsDevUpnpInfoXmlFragmentIndex  Unsigned32,
    cabhPsDevUpnpInfoXmlFragment       OCTET STRING
    }

cabhPsDevUpnpInfoIpType OBJECT-TYPE
    SYNTAX      InetAddressType
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "The type of InetAddress for cabhPsDevUpnpInfoIp."
    ::= { cabhPsDevUpnpInfoEntry 1 }

cabhPsDevUpnpInfoIp OBJECT-TYPE
    SYNTAX      InetAddress
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "The IP address of the device for which the UPnP
            information is being stored. This may be a DNS name
            (LAN Host name), an IPv4 or IPv6 prefix. Information
            pertaining to the PS itself is indicated by the PS
            well-known LAN IP address interface 192.168.0.1."
    ::= { cabhPsDevUpnpInfoEntry 2 }

cabhPsDevUpnpInfoXmlFragmentIndex OBJECT-TYPE
    SYNTAX      Unsigned32 (1..4294967295)
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "The index of the sequence of entries of
            cabhPsDevUpnpInfoXmlFragment for a specific
            cabhPsDevUpnpInfoIp IP address starting with '1'."
    ::= { cabhPsDevUpnpInfoEntry 3 }

cabhPsDevUpnpInfoXmlFragment OBJECT-TYPE
    SYNTAX      OCTET STRING (SIZE(0..400))
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "The UPnP Device information being requested by
            cabhPsDevUpnpCommand for the IP addresses specified
            in cabhPsDevUpnpInfoIp for LAN host(s). If the
            information is greater than 400 bytes,
            cabhPsDevUpnpInfoXmlFragmentIndex indicates the
            sequence of the consecutive portions per host identified in
            the table."
    ::= { cabhPsDevUpnpInfoEntry 4 }

--

cabhPsNotification     OBJECT IDENTIFIER ::= { cabhPsDevMib 2 }
```

```
cabhPsDevNotifications OBJECT IDENTIFIER ::= { cabhPsNotification 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,
        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
            cabhPsDevWanManMacAddress indicates the WAN-Man MAC address
            of the PS. This part of the information is uniform across
            all PS Traps."
    ::= { cabhPsDevNotifications 1 }

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

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

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

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

cabhPsDevSwUpgradeInitTrap NOTIFICATION-TYPE
    OBJECTS {
        docsDevEvLevel,
        docsDevEvId,
        docsDevEvText,
        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 IP address of the
            server from which the image was downloaded."
    ::= { cabhPsDevNotifications 5 }

cabhPsDevSwUpgradeFailTrap NOTIFICATION-TYPE
    OBJECTS {
        docsDevEvLevel,
        docsDevEvId,
        docsDevEvText,
        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 IP
            address of the server from which the image was downloaded."
    ::= { cabhPsDevNotifications 6 }

cabhPsDevSwUpgradeSuccessTrap NOTIFICATION-TYPE
    OBJECTS {
        docsDevEvLevel,
        docsDevEvId,
        docsDevEvText,
        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 IP address of the
            server from which the image was downloaded."
    ::= { cabhPsDevNotifications 7 }

cabhPsDevSwUpgradeCVCFailTrap NOTIFICATION-TYPE
    OBJECTS {
        docsDevEvLevel,
        docsDevEvId,
        docsDevEvText,
        cabhPsDevWanManMacAddress
```

```
    }
    STATUS      current
    DESCRIPTION
            "An event to report the failure of the verification of code
            file happened during a secure software upgrade attempt."
    ::= { cabhPsDevNotifications 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."
    ::= { cabhPsDevNotifications 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."
    ::= { cabhPsDevNotifications 10 }

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

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

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

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

cabhPsDevProvEnrollTrap NOTIFICATION-TYPE
    OBJECTS {
        cabhPsDevHardwareVersion,
        docsDevSwCurrentVers,
        cabhPsDevTypeIdentifier,
        cabhPsDevWanManMacAddress
    }
    STATUS      current
    DESCRIPTION
            "This notification is issued to initiate the CableHome
            provisioning process for SNMP Provisioning Mode."
    ::= { cabhPsDevNotifications 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."
    ::= { cabhPsDevNotifications 16 }

cabhPsDevUpnpMultiplePHTrap NOTIFICATION-TYPE
    OBJECTS {
        docsDevEvLevel,
        docsDevEvId,
        docsDevEvText,
        cabhQos2NumActivePolicyHolder,
        cabhQos2PolicyHolderEnabled,
        cabhQos2PolicyAdmissionControl
    }
    STATUS      current
    DESCRIPTION
```

```
            "To report that more than one active UPnP Policy Holders
            have been detected.
            This notification is triggered in the case the PS
            has cabhPsDevUpnpEnabled true."
    ::= { cabhPsDevNotifications 17 }

-- compliance statements

cabhPsBasicCompliance MODULE-COMPLIANCE
    STATUS      current
    DESCRIPTION
            "The compliance statement for devices that implement the
            CableHome Portal Services logical element."
    MODULE   -- cabhPsMib

-- unconditionally mandatory groups

MANDATORY-GROUPS {
        cabhPsDevBaseGroup,
        cabhPsDevProvGroup,
        cabhPsNotificationGroup,
        cabhPsDevAttribGroup,
        cabhPsDevStatsGroup,
        cabhPsDevAccessControlGroup,
        cabhPsDevUpnpGroup
        }

--  conditionally mandatory groups

GROUP cabhPsDev802dot11Group
    DESCRIPTION
            "This group is implemented only if PS
            supports interfaces of ifType ieee80211(71)."

GROUP cabhPsDevUIGroup
    DESCRIPTION
            "This group is implemented only in CableHome compliant
            residential gateways that implement a User Interface (UI)."

OBJECT cabhPsDevTimeServerAddrType
    SYNTAX      InetAddressType { ipv4(1) }
    DESCRIPTION
            "An implementation is only required to support IPv4
            addresses. "

OBJECT cabhPsDevTimeServerAddr
    SYNTAX      InetAddress (SIZE(4))
    DESCRIPTION
            "An implementation is only required to support IPv4
            addresses."

OBJECT cabhPsDevLanIpTrafficInetAddress
    SYNTAX      InetAddress (SIZE(4))
    DESCRIPTION
            "An implementation is only required to support IPv4
            addresses."

OBJECT cabhPsDevUpnpCommandIpType
    SYNTAX      InetAddressType { ipv4(1) }
    DESCRIPTION
            "An implementation is only required to support IPv4
            addresses."

OBJECT cabhPsDevUpnpCommandIp
```

```
    SYNTAX      InetAddress (SIZE(4))
    DESCRIPTION
            "An implementation is only required to support IPv4
            addresses."


OBJECT cabhPsDevUpnpInfoIpType
    SYNTAX      InetAddressType { ipv4(1) }
    DESCRIPTION
            "An implementation is only required to support IPv4
            addresses. "

OBJECT cabhPsDevUpnpInfoIp
    SYNTAX      InetAddress (SIZE(4))
    DESCRIPTION
            "An implementation is only required to support IPv4
            addresses."

    ::= { cabhPsCompliances 1 }

cabhPsDeprecatedCompliance MODULE-COMPLIANCE
    STATUS      deprecated
    DESCRIPTION
            "The compliance statement for deprecated MIB objects."
    MODULE   -- cabhPsMib

-- deprecated groups

GROUP cabhPsDevDeprecatedGroup
    DESCRIPTION
            "Group containing deprecated MIB objects."
    ::= { cabhPsCompliances 2 }

cabhPsObsoleteCompliance MODULE-COMPLIANCE
    STATUS      obsolete
    DESCRIPTION
            "The compliance statement for obsolete MIB objects."
    MODULE   -- cabhPsMib

GROUP cabhPsDevObsoleteGroup
    DESCRIPTION
            "Group containing obsolete MIB objects."

    ::= { cabhPsCompliances 3 }

cabhPsDevBaseGroup OBJECT-GROUP
    OBJECTS {
        cabhPsDevDateTime,
        cabhPsDevResetNow,
        cabhPsDevSerialNumber,
        cabhPsDevHardwareVersion,
        cabhPsDevWanManMacAddress,
        cabhPsDevWanDataMacAddress,
        cabhPsDevTypeIdentifier,
        cabhPsDevSetToFactory,
        cabhPsDevTodSyncStatus,
        cabhPsDevProvMode,
        cabhPsDevLastSetToFactory,
        cabhPsDevTrapControl
    }
    STATUS      current
    DESCRIPTION
            "A collection of objects for providing device status and
            control."
```

```
    ::= { cabhPsGroups 1 }

cabhPsDevProvGroup OBJECT-GROUP
    OBJECTS {
        cabhPsDevProvisioningTimer,
        cabhPsDevProvConfigFile,
        cabhPsDevProvConfigHash,
        cabhPsDevProvConfigFileSize,
        cabhPsDevProvConfigFileStatus,
        cabhPsDevProvConfigTLVProcessed,
        cabhPsDevProvConfigTLVRejected,
        cabhPsDevProvSolicitedKeyTimeout,
        cabhPsDevProvState,
        cabhPsDevProvAuthState,
        cabhPsDevTimeServerAddrType,
        cabhPsDevTimeServerAddr
    }
    STATUS      current
    DESCRIPTION
            "A collection of objects for controlling and providing
            status on provisioning."
    ::= { cabhPsGroups 2 }

cabhPsDevAttribGroup OBJECT-GROUP
    OBJECTS {
        cabhPsDevPsDeviceType,
        cabhPsDevPsManufacturerUrl,
        cabhPsDevPsModelUrl,
        cabhPsDevPsModelUpc
    }
    STATUS      current
    DESCRIPTION
            "A collection of objects for providing information on
            LAN IP devices known to the PS."
    ::= { cabhPsGroups 3 }

cabhPsDevStatsGroup OBJECT-GROUP
    OBJECTS {
        cabhPsDevLanIpTrafficCountersReset,
        cabhPsDevLanIpTrafficCountersLastReset,
        cabhPsDevLanIpTrafficEnabled,
        cabhPsDevLanIpTrafficInetAddressType,
        cabhPsDevLanIpTrafficInetAddress,
        cabhPsDevLanIpTrafficInOctets,
        cabhPsDevLanIpTrafficOutOctets
    }
    STATUS      current
    DESCRIPTION
            "A collection of objects for providing information
            on LAN IP traffic."
    ::= { cabhPsGroups 4 }

cabhPsDevDeprecatedGroup OBJECT-GROUP
    OBJECTS {
        cabhPsDevWanManClientId,
        cabhPsDevProvCorrelationId
    }
    STATUS      deprecated
    DESCRIPTION
            "Group of deprecated PSDev MIB objects."
    ::= { cabhPsGroups 5 }

cabhPsNotificationGroup NOTIFICATION-GROUP
    NOTIFICATIONS {
```

```
        cabhPsDevInitTLVUnknownTrap,
        cabhPsDevInitTrap,
        cabhPsDevInitRetryTrap,
        cabhPsDevDHCPFailTrap,
        cabhPsDevSwUpgradeInitTrap,
        cabhPsDevSwUpgradeFailTrap,
        cabhPsDevSwUpgradeSuccessTrap,
        cabhPsDevSwUpgradeCVCFailTrap,
        cabhPsDevTODFailTrap,
        cabhPsDevCdpWanDataIpTrap,
        cabhPsDevCdpThresholdTrap,
        cabhPsDevCspTrap,
        cabhPsDevCapTrap,
        cabhPsDevCtpTrap,
        cabhPsDevProvEnrollTrap,
        cabhPsDevCdpLanIpPoolTrap,
        cabhPsDevUpnpMultiplePHTrap
    }
    STATUS      current
    DESCRIPTION
            "These notifications indicate change in status of the
            Portal Services set of functions in a device complying
            with ITU-T Rec. J.192."
    ::= { cabhPsGroups 6 }

cabhPsDevAccessControlGroup OBJECT-GROUP
    OBJECTS {
        cabhPsDevAccessControlEnable,
        cabhPsDevAccessControlPhysAddr,
        cabhPsDevAccessControlRowStatus
    }
    STATUS      current
    DESCRIPTION
            "Group of Access Control objects for the CableHome PSDev
            MIB."
    ::= { cabhPsGroups 7 }

cabhPsDevUIGroup OBJECT-GROUP
    OBJECTS {
        cabhPsDevUILogin,
        cabhPsDevUIPassword,
        cabhPsDevUISelection,
        cabhPsDevUIServerUrl,
        cabhPsDevUISelectionDisabledBodyText
    }
    STATUS      current
    DESCRIPTION
            "A collection of objects for configuring the selection and
            operation of the User Interface displayed to an HTTP
            client, if a UI is implemented."
    ::= { cabhPsGroups 8 }

cabhPsDev802dot11Group OBJECT-GROUP
    OBJECTS {
               cabhPsDev802dot11BaseSetToDefault,
               cabhPsDev802dot11BaseLastSetToDefault,
               cabhPsDev802dot11BaseAdvertiseSSID,
               cabhPsDev802dot11BasePhyCapabilities,
               cabhPsDev802dot11BasePhyOperMode,
               cabhPsDev802dot11SecCapabilities,
               cabhPsDev802dot11SecOperMode,
               cabhPsDev802dot11SecPassPhraseToWEPKey,
               cabhPsDev802dot11SecUsePassPhraseToWEPKeyAlg,
               cabhPsDev802dot11SecPSKPassPhraseToKey,
```

```
                cabhPsDev802dot11SecWPAPreSharedKey,
                cabhPsDev802dot11SecWPARekeyTime,
                cabhPsDev802dot11SecControl,
                cabhPsDev802dot11SecCommitStatus
            }
    STATUS      current
    DESCRIPTION
            "Group of CableHome proprietary objects for the
            management of IEEE 80211 interfaces."
    ::= { cabhPsGroups 9 }

cabhPsDevUpnpGroup OBJECT-GROUP
    OBJECTS {
                cabhPsDevUpnpEnabled,
                cabhPsDevUpnpCommandIpType,
                cabhPsDevUpnpCommandIp,
                cabhPsDevUpnpCommand,
                cabhPsDevUpnpCommandUpdate,
                cabhPsDevUpnpLastCommandUpdate,
                cabhPsDevUpnpCommandStatus,
                cabhPsDevUpnpInfoXmlFragment
              }
    STATUS      current
    DESCRIPTION
            "Group of MIB objects for the
            management interface of UPnP Services."
    ::= { cabhPsGroups 10 }

cabhPsDevObsoleteGroup OBJECT-GROUP
    OBJECTS {
        cabhPsDevBpDeviceType,
        cabhPsDevBpManufacturer,
        cabhPsDevBpManufacturerUrl,
        cabhPsDevBpSerialNumber,
        cabhPsDevBpHardwareVersion,
        cabhPsDevBpHardwareOptions,
        cabhPsDevBpModelName,
        cabhPsDevBpModelNumber,
        cabhPsDevBpModelUrl,
        cabhPsDevBpModelUpc,
        cabhPsDevBpModelSoftwareOs,
        cabhPsDevBpModelSoftwareVersion,
        cabhPsDevBpLanInterfaceType,
        cabhPsDevBpNumberInterfacePriorities,
        cabhPsDevBpPhysicalLocation,
        cabhPsDevBpPhysicalAddress
    }
    STATUS      obsolete
    DESCRIPTION
            "Group of BP related objects with obsoleted status."
    ::= { cabhPsGroups 11 }

END
```

E.5 IPCable2Home Security (SEC) MIB requirement

The CableHome™ SEC MIB MUST be implemented as defined below.

```
CABH-SEC-MIB DEFINITIONS ::= BEGIN
IMPORTS
    MODULE-IDENTITY,
    Unsigned32,
    zeroDotZero,
```

```
    Counter32,
    OBJECT-TYPE                FROM SNMPv2-SMI  -- RFC 2578

    DateAndTime,
    TruthValue,
    TimeStamp,
    RowStatus,
    VariablePointer            FROM SNMPv2-TC  -- RFC 2579

    OBJECT-GROUP,
    MODULE-COMPLIANCE          FROM SNMPv2-CONF -- RFC 2580
    InetPortNumber,
    InetAddress                FROM INET-ADDRESS-MIB -- RFC 3291

    SnmpAdminString            FROM SNMP-FRAMEWORK-MIB -- RFC 2571

    X509Certificate            FROM DOCS-BPI2-MIB

    ZeroBasedCounter32         FROM RMON2-MIB
    docsDevFilterIpEntry       FROM DOCS-CABLE-DEVICE-MIB
    InterfaceIndexOrZero       FROM IF-MIB

    clabProjCableHome          FROM CLAB-DEF-MIB;

    cabhSecMib MODULE-IDENTITY
    LAST-UPDATED    "200408060000Z" -- August 6, 2004
    ORGANIZATION    "CableLabs Broadband Access Department"
    CONTACT-INFO
            "Kevin Luehrs
            Postal: Cable Television Laboratories, Inc.
            858 Coal Creek Circle
            Louisville, Colorado 80027
            U.S.A.
            Phone:  +1 303-661-9100
            Fax:    +1 303-661-9199
            E-mail: k.luehrs@cablelabs.com; mibs@cablelabs.com"
    DESCRIPTION
            "This MIB module supplies the basic management
            objects for the Security Portal Services."
    ::=  { clabProjCableHome 2 }


-- Textual conventions

   cabhSecMibObjects  OBJECT IDENTIFIER ::= { cabhSecMib 5 }
   cabhSecFwObjects   OBJECT IDENTIFIER ::= { cabhSecMib 1 }
   cabhSecFwBase      OBJECT IDENTIFIER ::= { cabhSecFwObjects 1 }
   cabhSecFwLogCtl    OBJECT IDENTIFIER ::= { cabhSecFwObjects 2 }

   cabhSecCertObjects OBJECT IDENTIFIER ::= { cabhSecMib 2 }
   cabhSecKerbObjects OBJECT IDENTIFIER ::= { cabhSecMibObjects 3 }
   cabhSecKerbBase    OBJECT IDENTIFIER ::= { cabhSecKerbObjects 1 }

   cabhSec2FwObjects  OBJECT IDENTIFIER ::= { cabhSecMibObjects 4 }
   cabhSec2FwBase     OBJECT IDENTIFIER ::= { cabhSec2FwObjects 1 }
   cabhSec2FwEvent    OBJECT IDENTIFIER ::= { cabhSec2FwObjects 2 }
   cabhSec2FwLog      OBJECT IDENTIFIER ::= { cabhSec2FwObjects 3 }
   cabhSec2FwFilter   OBJECT IDENTIFIER ::= { cabhSec2FwObjects 4 }

--
--    CableHome 1.0 Base Firewall Functions
--

cabhSecFwPolicyFileEnable OBJECT-TYPE
```

```
    SYNTAX      INTEGER {
                    enable (1),
                    disable(2)
                }
    MAX-ACCESS  read-write
    STATUS      deprecated
    DESCRIPTION
            "This parameter indicates whether or not to enable
            the firewall functionality."
    DEFVAL { enable }
    ::= { cabhSecFwBase 1 }

cabhSecFwPolicyFileURL OBJECT-TYPE
    SYNTAX      SnmpAdminString
    MAX-ACCESS  read-write
    STATUS      deprecated
    DESCRIPTION
            "A policy rule set file download is triggered when the
            value used to set this object is different than the value
            in the cabhSecFwPolicySuccessfulFileURL object."
    REFERENCE
            "CableHome 1.0 Specification, CH-SP-CH1.0-I05-030801,
            11.3.5.2 of ITU-T Rec. J.191, Firewall Rule Set Management
            Parameters."
    DEFVAL { "" }
    ::= { cabhSecFwBase 2 }

cabhSecFwPolicyFileHash OBJECT-TYPE
    SYNTAX      OCTET STRING (SIZE(0|20))
    MAX-ACCESS  read-write
    STATUS      deprecated
    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 length of the hash is 160 bits.
            This hash value is encoded in binary format."
    DEFVAL { ''h }
    ::= { cabhSecFwBase 3 }

cabhSecFwPolicyFileOperStatus OBJECT-TYPE
    SYNTAX      INTEGER {
                    inProgress(1),
                    complete(2),
                 -- completeFromMgt(3), deprecated
                    failed(4)
                }
    MAX-ACCESS  read-only
    STATUS      deprecated
    DESCRIPTION
            "inProgress(1) indicates a firewall configuration
            file download is under way.
            complete (2) indicates the firewall configuration
            file downloaded and configured successfully.
            completeFromMgt(3). This state is deprecated.
            failed(4) indicates the last attempted firewall
            configuration file download or processing
            failed ordinarily due to TFTP timeout."
    ::= { cabhSecFwBase 4 }

cabhSecFwPolicyFileCurrentVersion OBJECT-TYPE
    SYNTAX      SnmpAdminString
    MAX-ACCESS  read-only
    STATUS      deprecated
```

```
    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 }

cabhSecFwPolicySuccessfulFileURL OBJECT-TYPE
    SYNTAX      SnmpAdminString
    MAX-ACCESS  read-only
    STATUS      deprecated
    DESCRIPTION
            "Contains the location of the last successful downloaded
            policy rule set file in the format pointed in the
            reference. If a successful download has never occurred,
            this MIB object MUST report empty string."
    REFERENCE
            "CableHome 1.0 Specification, CH-SP-CH1.0-I05-030801,
            11.3.5.2 of ITU-T Rec. J.191, Firewall Rule Set Management
            Parameters."
    DEFVAL { "" }
    ::= { cabhSecFwBase 6 }

--
-- CableHome 1.0 Firewall Event MIBs
--

cabhSecFwEventType1Enable OBJECT-TYPE
    SYNTAX      INTEGER {
                    enable(1), -- log event
                    disable(2) -- do not log event
                }
    MAX-ACCESS  read-write
    STATUS      deprecated
    DESCRIPTION
            "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      deprecated
    DESCRIPTION
            "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      deprecated
    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      deprecated
    DESCRIPTION
            "If the number of type 1 or 2 hacker attacks
            exceeds this threshold in the period defined
            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      deprecated
    DESCRIPTION
            "Indicates the period to be used (in 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 }

--
-- CableHome PS device certificate
--

    cabhSecCertPsCert OBJECT-TYPE
    SYNTAX      X509Certificate
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "The X509 DER-encoded PS certificate."
    ::= { cabhSecCertObjects 1 }

--
--  CableHome 1.1 Firewall Management MIBs
--

cabhSec2FwEnable OBJECT-TYPE
    SYNTAX      INTEGER    {
                    enabled(1),
```

```
                    disabled(2)
                }
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
            "This parameter indicates whether to enable or disable the
            firewall."
    DEFVAL { enabled }
    ::= { cabhSec2FwBase 1 }

cabhSec2FwPolicyFileURL OBJECT-TYPE
    SYNTAX      SnmpAdminString
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
            "A policy rule set file download is triggered when the
            value used to set this object is different than the value
            in the cabhSec2FwPolicySuccessfulFileURL object."
    REFERENCE
            "CableHome 1.1 Specification, CH-SP-CH1.1-I05-040806,
            11.6.4.9.1 of ITU-T Rec. J.192, Firewall Rule Set Management
            MIB Objects."
    DEFVAL { "" }
    ::= { cabhSec2FwBase 2 }

cabhSec2FwPolicyFileHash OBJECT-TYPE
    SYNTAX      OCTET STRING (SIZE(0|20))
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
            "Hash of the contents of the firewall
            configuration file. For the SHA-1 authentication
            algorithm, the length of the hash is 160 bits.
            This hash value is encoded in binary format."
    DEFVAL { ''h }
    ::= { cabhSec2FwBase 3 }

cabhSec2FwPolicyFileOperStatus OBJECT-TYPE
    SYNTAX      INTEGER {
                    inProgress(1),
                    complete(2),
                    failed(3)
                }
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "InProgress(1) indicates a firewall configuration
            file download is under way. Complete(2) indicates
            the firewall configuration file was downloaded
            and processed successfully. Failed(3) indicates
            that the last attempted firewall configuration
            file download or processing failed."
    ::= { cabhSec2FwBase 4 }

cabhSec2FwPolicyFileCurrentVersion OBJECT-TYPE
    SYNTAX      SnmpAdminString
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
            "A label set by the cable operator that can be
            used to track various versions of configured
            rulesets. Once the label is set and configured
            rules are changed, it may not accurately reflect
            the version of configured rules running on the box.
```

```
            If this object has never been configured, it MUST
            contain an empty string."
    DEFVAL { "" }
    ::= { cabhSec2FwBase 5 }

cabhSec2FwClearPreviousRuleset OBJECT-TYPE
    SYNTAX      TruthValue
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
            "If set to 'true', the PS MUST clear all entries in the
            docsDevFilterIpTable. Reading this value always returns
            false."
    REFERENCE
            "CableHome specification - Security section"
    DEFVAL { false }
    ::= { cabhSec2FwBase 6 }

cabhSec2FwPolicySelection OBJECT-TYPE
    SYNTAX      INTEGER {
                    factoryDefault(1),
                    configuredRulesetBoth(2),
                    factoryDefaultAndConfiguredRulesetBoth(3),
                    configuredRulesetDocsDevFilterIpTable(4),
                    configuredRulesetCabhSec2FwLocalFilterIpTable (5),
                    factoryDefaultAndDocsDevFilterIpTable (6),
                    factoryDefaultAndCabhSec2FwLocalFilterIpTable (7)
                }
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
            "This object allows for selection of the filtering policy
            as defined by the following options:

            factoryDefault (1) The firewall filters against the Factory
            Default Ruleset in the cabhSec2FwFactoryDefaultFilterTable.

            configuredRulesetBoth (2) The firewall filters against the
            Configured Ruleset defined by both the
            docsDevFilterIpTable and the cabhSec2FwLocalFilterIpTable.

            factoryDefaultAndConfiguredRulesetBoth (3) The firewall
            filters against the CableHome specified Factory Default
            Ruleset in the cabhSec2FwFactoryDefaultFilterTable and
            the Configured Ruleset in the docsDevFilterIpTable and
            the cabhSec2FwLocalFilterIpTable.

            configuredRulesetDocsDevFilterIpTable(4) The firewall
            filters against the Configured Ruleset defined by the
            docsDevFilterIpTable.

            configuredRulesetCabhSec2FwLocalFilterIpTable (5) The
            firewall filters against the Configured Ruleset defined by
            the cabhSec2FwLocalFilterIpTable.

            factoryDefaultAndDocsDevFilterIpTable (6) The firewall
            filters against the Factory Default Ruleset and the
            Configured Ruleset defined by the DocsDevFilterIpTable.

            factoryDefaultAndCabhSec2FwLocalFilterIpTable (7) The
            firewall filters against the Factory Default Ruleset and
            the Configured Ruleset defined by the
            cabhSec2FwLocalFilterIpTable."
```

```
    REFERENCE
            "CableHome specification - Security section."
    DEFVAL { factoryDefault }
    ::= { cabhSec2FwBase 7 }

cabhSec2FwEventSetToFactory OBJECT-TYPE
    SYNTAX      TruthValue
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
            "If set to 'true', entries in cabhSec2FwEventControlEntry
            are set to their default values.
            Reading this value always returns false."
    DEFVAL { false }
    ::= { cabhSec2FwBase 8 }

cabhSec2FwEventLastSetToFactory OBJECT-TYPE
    SYNTAX      TimeStamp
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "The value of sysUpTime when cabhSec2FwEventSetToFactory
            was Last set to true. Zero if never reset."
    ::= { cabhSec2FwBase 9 }

cabhSec2FwPolicySuccessfulFileURL OBJECT-TYPE
    SYNTAX      SnmpAdminString
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "Contains the location of the last successful downloaded
            policy rule set file in the format pointed in the
            reference. If a successful download has not yet
            occurred, this MIB object should report empty string."
    REFERENCE
            "CableHome 1.1 Specification, CH-SP-CH1.1-I05-040806,
            11.6.4.9.1 of ITU-T Rec. J.192, Firewall Rule Set Management MIB
Objects."
    DEFVAL { "" }
    ::= { cabhSec2FwBase 10 }

cabhSec2FwConfiguredRulesetPriority OBJECT-TYPE
    SYNTAX      INTEGER {
                    docsDevFilterIpTable (1),
                    cabhSec2FwLocalFilterIpTable (2)
                }
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
            "This object defines which Configured Ruleset filter rule
             has priority when a conflict exists between a filter rule
             in the docsDevFilterIpTable and a filter rule in the
             cabhSec2FwLocalFilterIpTable as indicated by the following
             options:

             docsDevFilterIpTable (1) - indicates that filter rules in
             the docsDevFilterIpTable have priority over any
             conflicting filters that may exist in the
             cabhSec2FwLocalFilterIpTable.

             cabhSec2FwLocalFilterIpTable (2) - indicates that filter
             rules in the cabhSec2FwLocalFilterIpTable have priority
             over any conflicting filters that may exist in the
             docsDevFilterIpTable."
```

```
    REFERENCE
             "CableHome specification - Security section."
    DEFVAL { cabhSec2FwLocalFilterIpTable }
    ::= { cabhSec2FwBase 11 }

cabhSec2FwClearLocalRuleset OBJECT-TYPE
    SYNTAX      TruthValue
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
            "If set to 'true', the PS MUST clear all entries in the
            cabhSec2FwLocalFilterIpTable. Reading this value always
            returns false."
    REFERENCE
            "CableHome specification - Security section"
    DEFVAL { false }
    ::= { cabhSec2FwBase 12 }

-- ++++++++++

--
-- CableHome 1.1 Firewall Event MIBs
--


cabhSec2FwEventControlTable OBJECT-TYPE
    SYNTAX      SEQUENCE OF CabhSec2FwEventControlEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "This table controls the reporting of the
            Firewall Attacks events"
    ::= { cabhSec2FwEvent 1 }

cabhSec2FwEventControlEntry OBJECT-TYPE
    SYNTAX      CabhSec2FwEventControlEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "Allows configuration of the reporting mechanisms
            for a particular type of attack."
    INDEX { cabhSec2FwEventType }
    ::= { cabhSec2FwEventControlTable 1 }

CabhSec2FwEventControlEntry ::= SEQUENCE {
    cabhSec2FwEventType         INTEGER,
    cabhSec2FwEventEnable       INTEGER,
    cabhSec2FwEventThreshold    Unsigned32,
    cabhSec2FwEventInterval     Unsigned32,
    cabhSec2FwEventCount        ZeroBasedCounter32,
    cabhSec2FwEventLogReset     TruthValue,
    cabhSec2FwEventLogLastReset TimeStamp
    }

cabhSec2FwEventType OBJECT-TYPE
    SYNTAX      INTEGER     {
                    type1(1),
                    type2(2),
                    type3(3),
                    type4(4),
                    type5(5),
                    type6(6)
                }
    MAX-ACCESS  not-accessible
```

```
    STATUS      current
    DESCRIPTION
            "Classification of the different types of
            attacks.
            Type 1  logs all attempts from both LAN and WAN
            clients to traverse the Firewall that violate the
            Security Policy.
            Type 2 logs identified Denial of Service attack
            attempts.
            Type 3 logs all changes made to the
            cabhSec2FwPolicyFileURL,
            cabhSec2FwPolicyFileCurrentVersion or
            cabhSec2FwPolicyFileEnable objects.
            Type 4 logs all failed attempts to modify
            cabhSec2FwPolicyFileURL and
            cabhSec2FwPolicyFileEnable objects.
            Type 5 logs allowed inbound packets from the WAN.
            Type 6 logs allowed outbound packets from the
            LAN."
    ::= { cabhSec2FwEventControlEntry 1 }

cabhSec2FwEventEnable OBJECT-TYPE
    SYNTAX      INTEGER    {
                    enabled(1),
                    disabled(2)
                }
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
            "Enables or disables counting and logging of
            firewall events by type as assigned by
            cabhSec2FwEventType."
    DEFVAL { disabled }
    ::= { cabhSec2FwEventControlEntry 2 }

cabhSec2FwEventThreshold OBJECT-TYPE
    SYNTAX       Unsigned32 (0..65535)
    MAX-ACCESS   read-write
    STATUS       current
    DESCRIPTION
            "Number of attacks to count before sending the
            appropriate event by type as assigned by
            cabhSec2FwEventType."
    DEFVAL { 0 }
    ::= { cabhSec2FwEventControlEntry 3 }

cabhSec2FwEventInterval OBJECT-TYPE
    SYNTAX      Unsigned32 (0..744)
    UNITS       "hours"
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
            "Indicates the time interval in hours to count and log
            occurrences of a firewall event type as assigned in
            cabhSec2FwEventType. If this MIB has a value of zero,
            then there is no interval assigned and the PS will not
            count or log events."
    DEFVAL { 0 }
    ::= { cabhSec2FwEventControlEntry 4 }

cabhSec2FwEventCount OBJECT-TYPE
    SYNTAX       ZeroBasedCounter32
    MAX-ACCESS   read-only
    STATUS       current
```

```
    DESCRIPTION
           "Indicates the current count up to the
            cabhSec2FwEventThreshold value by type as
            assigned by cabhSec2FwEventType."
    ::= { cabhSec2FwEventControlEntry 5 }

cabhSec2FwEventLogReset OBJECT-TYPE
    SYNTAX       TruthValue
    MAX-ACCESS   read-write
    STATUS       current
    DESCRIPTION
            "Setting this object to true clears the log table
            for the specified event type. Reading this object
            always returns false."
    DEFVAL { false }
    ::= { cabhSec2FwEventControlEntry 6 }

cabhSec2FwEventLogLastReset OBJECT-TYPE
    SYNTAX       TimeStamp
    MAX-ACCESS   read-only
    STATUS       current
    DESCRIPTION
            "The value of sysUpTime when cabhSec2FwEventLogReset was
            last set to true. Zero if never reset."
    ::= { cabhSec2FwEventControlEntry 7 }

--
-- CableHome 1.1 Firewall Log Tables
--

cabhSec2FwLogTable OBJECT-TYPE
    SYNTAX      SEQUENCE OF CabhSec2FwLogEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "Contains a log of packet information as related
            to events enabled by the cable operator. The types
            are defined in the CableHome 1.1 specification and
            require various objects to be included in the log.
            The following is a description for what is
            expected in the log for each type Type 1, Type 2,
            Type 5 and Type 6 table MUST include
            cabhSec2FwEventType, cabhSec2FwEventPriority,
            cabhSec2FwEventId, cabhSec2FwLogTime,
            cabhSec2FwIpProtocol, cabhSec2FwIpSourceAddr,
            cabhSec2FwIpDestAddr, cabhSec2FwIpSourcePort,
            cabhSec2FwIpDestPort, cabhSec2Fw,
            cabhSec2FwReplayCount.  The other values not used
            by Types 1, 2, 5 and 6 are default values. Type 3
            and Type 4 MUST include cabhSec2FwEventType,
            cabhSec2FwEventPriority, cabhSec2FwEventId,
            cabhSec2FwLogTime, cabhSec2FwIpSourceAddr,
            cabhSec2FwLogMIBPointer.  The other values not used
            by type 3 and 4 are default values. When applicable,
            Type 1, Type 5,and Type 6 MUST also include
            cabhSec2FwLogMatchingFilterTableName,
            cabhSec2FwLogMatchingFilterTableIndex,
            cabhSec2FwLogMatchingFilterDescr."
    ::= { cabhSec2FwLog 1 }

cabhSec2FwLogEntry OBJECT-TYPE
    SYNTAX      CabhSec2FwLogEntry
    MAX-ACCESS  not-accessible
    STATUS      current
```

```
    DESCRIPTION
            "Each entry contains the log of firewall events"
    INDEX {cabhSec2FwLogIndex}
    ::= { cabhSec2FwLogTable 1 }

CabhSec2FwLogEntry ::= SEQUENCE {
    cabhSec2FwLogIndex                      Unsigned32,
    cabhSec2FwLogEventType                  INTEGER,
    cabhSec2FwLogEventPriority              INTEGER,
    cabhSec2FwLogEventId                    Unsigned32,
    cabhSec2FwLogTime                       DateAndTime,
    cabhSec2FwLogIpProtocol                 Unsigned32,
    cabhSec2FwLogIpSourceAddr               InetAddress,
    cabhSec2FwLogIpDestAddr                 InetAddress,
    cabhSec2FwLogIpSourcePort               InetPortNumber,
    cabhSec2FwLogIpDestPort                 InetPortNumber,
    cabhSec2FwLogMessageType                Unsigned32,
    cabhSec2FwLogReplayCount                Unsigned32,
    cabhSec2FwLogMIBPointer                 VariablePointer,
    cabhSec2FwLogMatchingFilterTableName    INTEGER,
    cabhSec2FwLogMatchingFilterTableIndex   Unsigned32,
    cabhSec2FwLogMatchingFilterDescr        SnmpAdminString
    }

cabhSec2FwLogIndex OBJECT-TYPE
    SYNTAX      Unsigned32 (1..2147483647)
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "A sequence number for the specific events
            under a cabhSec2FwEventType."
    ::= { cabhSec2FwLogEntry 1 }

cabhSec2FwLogEventType OBJECT-TYPE
    SYNTAX INTEGER     {
               type1(1),
               type2(2),
               type3(3),
               type4(4),
               type5(5),
               type6(6)
           }
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "Classification of the different types of
            attacks.
            Type 1 logs all attempts from both LAN and WAN
            clients to traverse the Firewall that violate
            the Security Policy.
            Type 2 logs identified Denial of Service attack
            attempts.
            Type 3 logs all changes made to the
            cabhSec2FwPolicyFileURL,
            cabhSec2FwPolicyFileCurrentVersion or
            cabhSec2FwPolicyFileEnable objects.
            Type 4 logs all failed attempts to modify
            cabhSec2FwPolicyFileURL and
            cabhSec2FwPolicyFileEnable objects.
            Type 5 logs allowed inbound packets from the WAN.
            Type 6 logs allowed outbound packets from the
            LAN."
    ::= { cabhSec2FwLogEntry 2 }
```

```
cabhSec2FwLogEventPriority OBJECT-TYPE
    SYNTAX     INTEGER     {
                   emergency(1),
                   alert(2),
                   critical(3),
                   error(4),
                   warning(5),
                   notice(6),
                   information(7),
                   debug(8)
               }
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "The priority level of this event as defined
            by CableHome Specification. If a priority is
            not assigned in the CableHome specification for
            a particular event, then the vendor or cable
            operator may assign priorities. These are
            ordered from most serious (emergency)to least
            serious (debug)."
    ::= { cabhSec2FwLogEntry 3 }

cabhSec2FwLogEventId  OBJECT-TYPE
    SYNTAX      Unsigned32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "The assigned event ID."
    ::= { cabhSec2FwLogEntry 4 }

cabhSec2FwLogTime OBJECT-TYPE
    SYNTAX      DateAndTime
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "The time that this entry was created by the PS."
    ::= { cabhSec2FwLogEntry 5 }

cabhSec2FwLogIpProtocol OBJECT-TYPE
    SYNTAX      Unsigned32 (0..256)
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "The IP Protocol."
    ::= { cabhSec2FwLogEntry 6 }

cabhSec2FwLogIpSourceAddr OBJECT-TYPE
    SYNTAX      InetAddress
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "The Source IP Address of the packet logged."
    ::= { cabhSec2FwLogEntry 7 }

cabhSec2FwLogIpDestAddr OBJECT-TYPE
    SYNTAX      InetAddress
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "The Destination IP Address of the packet logged."
    ::= { cabhSec2FwLogEntry 8 }

cabhSec2FwLogIpSourcePort OBJECT-TYPE
```

```
    SYNTAX      InetPortNumber
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "The Source IP Port of the packet logged."
    ::= { cabhSec2FwLogEntry 9 }

cabhSec2FwLogIpDestPort OBJECT-TYPE
    SYNTAX      InetPortNumber
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "The Source IP Port of the packet logged."
    ::= { cabhSec2FwLogEntry 10 }

cabhSec2FwLogMessageType OBJECT-TYPE
    SYNTAX      Unsigned32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "The ICMP defined types."
    ::= { cabhSec2FwLogEntry 11}

cabhSec2FwLogReplayCount OBJECT-TYPE
    SYNTAX      Unsigned32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "The number of identical attack packets that
            were seen by the firewall based on
            cabhSec2FwLogIpProtocol, cabhSec2FwLogIpSourceAddr,
            cabhSec2FwLogIpDestAddr, cabhSec2FwLogIpSourcePort,
            cabhSec2FwLogIpDestPort and cabhSec2FwLogMessageType."
    DEFVAL { 0 }
    ::= { cabhSec2FwLogEntry 12 }

cabhSec2FwLogMIBPointer OBJECT-TYPE
    SYNTAX      VariablePointer
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "Identifies if the cabhSec2FwPolicyFileURL or the
            cabhSec2FwEnable MIB object changed or an attempt
            was made to change it."
    DEFVAL { zeroDotZero }
    ::= { cabhSec2FwLogEntry 13 }

cabhSec2FwLogMatchingFilterTableName OBJECT-TYPE
    SYNTAX      INTEGER     {
                   cabhSec2FwFactoryDefaultFilterTable(1),
                   docsDevFilterIpTable(2),
                   cabhSec2FwLocalFilterIpTable(3),
                   none(4)
                }
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "When applicable, cabhSec2FwLogMatchingFilterTableName
            indicates the filter table name containing the last filter
            rule matched that caused the event to be generated."
    DEFVAL { none }
    ::= { cabhSec2FwLogEntry 14 }

cabhSec2FwLogMatchingFilterTableIndex OBJECT-TYPE
```

```
    SYNTAX      Unsigned32 (0..2147483647)
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "When applicable, cabhSec2FwLogMatchingFilterTableIndex
            indicates the filter table index if the last filter
            rule matched that caused the event to be generated. If
            the value is 0, the event was not caused by a filter
            rule match. "
    DEFVAL { 0 }
    ::= { cabhSec2FwLogEntry 15 }

cabhSec2FwLogMatchingFilterDescr OBJECT-TYPE
    SYNTAX      SnmpAdminString (SIZE(0..32))
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "When applicable, cabhSec2FwLogMatchingFilterDesc
            contains the description value found in the
            cabhSec2FwFilterScheduleDesc MIB object or the
            cabhSec2FwLocalFilterIpDesc MIB object of the last
            filter rule matched that caused the event to be
            generated."
    DEFVAL { "" }
    ::= { cabhSec2FwLogEntry 16 }

-- ============================================================
--
--  CableHome 1.1 PS IP Filter Scheduling Table
--
--  The cabhSec2FwFilterScheduleTable contains the firewall
--  policy identification and links that policy as defined
--  in RFC 2669 to specific time of day restrictions.
--
-- ============================================================

cabhSec2FwFilterScheduleTable OBJECT-TYPE
    SYNTAX SEQUENCE OF CabhSec2FwFilterScheduleEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "Extends the filtering matching parameters of
            docsDevFilterIpTable defined in RFC 2669 for CableHome
            Residential Gateways to include time day intervals and days
            of the week."
    ::= { cabhSec2FwFilter 1 }


cabhSec2FwFilterScheduleEntry OBJECT-TYPE
    SYNTAX      CabhSec2FwFilterScheduleEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "Extended values for entries of docsDevFilterIpTable.
            If the PS has not acquired ToD, the entire
            docsDevFilterIpEntry rule set is ignored.
            Note - A filter time period may include two days
            (e.g., from 10 PM to 4 AM).  A filter time period that
            includes two days is identified by the absolute value
            of the cabhSec2FwFilterScheduleEndTime being less than the
            absolute value of the cabhSec2FwFilterScheduleStartTime.
            The cabhSec2FwFilterScheduleDOW setting and the
            cabhSec2FwFilterScheduleStartTime value indicate what day
            and time the filter becomes active. The
```

```
            cabhSec2FwFilterScheduleEndTime indicates when the filter
            becomes inactive on the second day. The maximum filter
            time period that includes two days is 24 hours.
            If cabhSec2FwFilterScheduleStartTime is less than or
            equal to the cabhSec2FwFilterScheduleEndTime, the time
            period of the filter falls in the same day."
    AUGMENTS { docsDevFilterIpEntry }
    ::= { cabhSec2FwFilterScheduleTable 1 }

CabhSec2FwFilterScheduleEntry ::= SEQUENCE {
    cabhSec2FwFilterScheduleStartTime    Unsigned32,
    cabhSec2FwFilterScheduleEndTime      Unsigned32,
    cabhSec2FwFilterScheduleDOW          BITS,
    cabhSec2FwFilterScheduleDescr        SnmpAdminString
    }

cabhSec2FwFilterScheduleStartTime OBJECT-TYPE
    SYNTAX      Unsigned32 (0..2359)
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
            "The start time for matching the filter ruleset in the
            specified days indicated in cabhSec2FwFilterScheduleDOW.
            Time is represented in Military Time, e.g., 8:30 AM is
            represented as 830 and 11:45 PM as 2345. An attempt to set
            this object to an invalid military time value, e.g., 1182,
            returns 'wrongValue' error."
    DEFVAL { 0 }
    ::= { cabhSec2FwFilterScheduleEntry 1 }

cabhSec2FwFilterScheduleEndTime OBJECT-TYPE
    SYNTAX      Unsigned32 (0..2359)
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
            "The end time for matching the filter rule for the
            days indicated in cabhSec2FwFilterScheduleDOW. The filter
            rule associated with this end time MUST not be disabled
            until the minute following the time indicated by this
            MIB object. If the time period is for two days,
            identified by cabhSec2FwFilterScheduleEndTime being
            less than cabhSec2FwFilterScheduleStartTime, then
            the cabhSec2FwFilterScheduleDOW settings
            do not apply to this MIB object.
            Time is represented in the same manner as in
            cabhSec2FwFilterScheduleStartTime. An attempt to set
            this object to an invalid military time value, e.g., 1182,
            returns 'wrongValue' error."
    DEFVAL { 2359 }
    ::= { cabhSec2FwFilterScheduleEntry 2 }

cabhSec2FwFilterScheduleDOW OBJECT-TYPE
    SYNTAX BITS {
                    sunday(0),
                    monday(1),
                    tuesday(2),
                    wednesday(3),
                    thursday(4),
                    friday(5),
                    saturday(6)
                }
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
```

```
            "If the day of week bit associated with the PS given day
            is '1', this object criteria matches."
    DEFVAL { 'fe'h }  -- 11111110 Sun-Sat
    ::= { cabhSec2FwFilterScheduleEntry 3 }

cabhSec2FwFilterScheduleDescr OBJECT-TYPE
    SYNTAX      SnmpAdminString (SIZE(0..32))
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
            "A filter rule description configured by the
            cable operator or subscriber."
    DEFVAL { "" }
    ::= { cabhSec2FwFilterScheduleEntry 4 }

-- ============================================================
--
--  CableHome 1.1 PS Firewall Factory Default Filter Table
--
--  The cabhSec2FwFactoryDefaultFilterTable contains the
--  firewall factory default ruleset in a read only table as
--  defined by the CableLabs CableHome 1.1 Specification.
--
-- ============================================================

cabhSec2FwFactoryDefaultFilterTable OBJECT-TYPE
    SYNTAX SEQUENCE OF CabhSec2FwFactoryDefaultFilterEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "Contains the firewall factory default ruleset as
            defined by the CableLabs CableHome 1.1 Specification."
    ::= { cabhSec2FwFilter 2 }

cabhSec2FwFactoryDefaultFilterEntry OBJECT-TYPE
    SYNTAX      CabhSec2FwFactoryDefaultFilterEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "Contains the firewall factory default ruleset."
    INDEX {cabhSec2FwFactoryDefaultFilterIndex }
    ::= { cabhSec2FwFactoryDefaultFilterTable 1 }

CabhSec2FwFactoryDefaultFilterEntry ::= SEQUENCE {
    cabhSec2FwFactoryDefaultFilterIndex           Unsigned32,
    cabhSec2FwFactoryDefaultFilterControl         INTEGER,
    cabhSec2FwFactoryDefaultFilterIfIndex         InterfaceIndexOrZero,
    cabhSec2FwFactoryDefaultFilterDirection       INTEGER,
    cabhSec2FwFactoryDefaultFilterSaddr           InetAddress,
    cabhSec2FwFactoryDefaultFilterSmask           InetAddress,
    cabhSec2FwFactoryDefaultFilterDaddr           InetAddress,
    cabhSec2FwFactoryDefaultFilterDmask           InetAddress,
    cabhSec2FwFactoryDefaultFilterProtocol        Unsigned32,
    cabhSec2FwFactoryDefaultFilterSourcePortLow   Unsigned32,
    cabhSec2FwFactoryDefaultFilterSourcePortHigh  Unsigned32,
    cabhSec2FwFactoryDefaultFilterDestPortLow     Unsigned32,
    cabhSec2FwFactoryDefaultFilterDestPortHigh    Unsigned32,
    cabhSec2FwFactoryDefaultFilterContinue        TruthValue
}

cabhSec2FwFactoryDefaultFilterIndex OBJECT-TYPE
    SYNTAX      Unsigned32 (1..2147483647)
    MAX-ACCESS  not-accessible
    STATUS      current
```

```
    DESCRIPTION
            "Index used to order the application of filters.
            The filter with the lowest index is always applied
            first."
    ::= { cabhSec2FwFactoryDefaultFilterEntry 1 }

cabhSec2FwFactoryDefaultFilterControl OBJECT-TYPE
    SYNTAX      INTEGER {
                    deny(1),
                    allow(2)
                }
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "If set to deny(1), all packets matching this filter
            will be discarded. If set to allow(2), all
            packets matching this filter will be accepted.
            The cabhSec2FwFactoryDefaultFilterContinue object is
            set to true, and therefore the PS MUST continue to
            scan the table for other matches to apply the match
            with the highest cabhSec2FwFactoryDefaultFilterIndex
            value."
    ::= { cabhSec2FwFactoryDefaultFilterEntry 2 }

cabhSec2FwFactoryDefaultFilterIfIndex OBJECT-TYPE
    SYNTAX      InterfaceIndexOrZero
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "The index number assigned to this object MUST
            match the IfIndex numbering assigned in the
            ifTable from the Interfaces Group MIB [RFC 2863],
            and as specified in CH 1.1 Spec, Table 6-17 of
            ITU-T Rec. J.192, Numbering Interfaces in the
            ifTable. If the value is zero, the filter applies
            to all interfaces. This object MUST be specified
            to create a row in this table."
    ::= { cabhSec2FwFactoryDefaultFilterEntry 3 }

cabhSec2FwFactoryDefaultFilterDirection OBJECT-TYPE
    SYNTAX      INTEGER {
                    inbound(1),
                    outbound(2),
                    both(3)
                }
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "This value represents direction in relationship
            to the assigned
            cabhSec2FwFactoryDefaultFilterIfIndex
            in this particular rule, meaning that the PS
            MUST represent traffic direction as follows:
            inbound(1)traffic, outbound(2) traffic, or
            both(3)inbound and outbound traffic."
    ::= { cabhSec2FwFactoryDefaultFilterEntry 4 }

cabhSec2FwFactoryDefaultFilterSaddr OBJECT-TYPE
    SYNTAX      InetAddress
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "The source IP address, or portion thereof, that is
            to be matched for this filter. The source address
```

```
            is first masked (and'ed) against
            cabhSec2FwFactoryDefaultFilterSmask
            before being compared  to this value. A value of 0
            for this object and 0 for the mask matches all IP
            addresses."
    DEFVAL { '00000000'h }
    ::= { cabhSec2FwFactoryDefaultFilterEntry 5 }

cabhSec2FwFactoryDefaultFilterSmask OBJECT-TYPE
    SYNTAX      InetAddress
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "A bit mask that is to be applied to the source
            address prior to matching. This mask is not
            necessarily the same as a subnet mask, but 1's
            bits must be leftmost and contiguous."
    DEFVAL { '00000000'h }
    ::= { cabhSec2FwFactoryDefaultFilterEntry 6 }

cabhSec2FwFactoryDefaultFilterDaddr OBJECT-TYPE
    SYNTAX      InetAddress
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "The destination IP address, or portion thereof, that
            is to be matched for this filter. The destination
            address is first masked (and'ed) against
            cabhSec2FwFactoryDefaultFilterDmask
            before being compared  to this value. A value of 0
            for this object and 0 for the mask matches all
            IP addresses."
    DEFVAL { '00000000'h }
    ::= { cabhSec2FwFactoryDefaultFilterEntry 7 }

cabhSec2FwFactoryDefaultFilterDmask OBJECT-TYPE
    SYNTAX      InetAddress
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "A bit mask that is to be applied to the destination
            address prior to matching. This mask is not necessarily
            the same as a subnet mask, but 1's bits must be leftmost
            and contiguous."
    DEFVAL { '00000000'h }
    ::= { cabhSec2FwFactoryDefaultFilterEntry 8 }

cabhSec2FwFactoryDefaultFilterProtocol OBJECT-TYPE
    SYNTAX      Unsigned32 (0..65535)
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "The protocol value that is to be matched. For example:
            icmp is 1, tcp is 6, udp is 17. A value of 65535 matches
            ANY protocol."
    DEFVAL { 65535 }
    ::= { cabhSec2FwFactoryDefaultFilterEntry 9 }

cabhSec2FwFactoryDefaultFilterSourcePortLow OBJECT-TYPE
    SYNTAX      Unsigned32 (0..65535)
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "If cabhSec2FwFactoryDefaultFilterProtocol is udp
```

```
            or tcp, this is the inclusive lower bound of the
            transport-layer source port range that is to be
            matched, otherwise it is ignored during matching."
    DEFVAL { 0 }
    ::= { cabhSec2FwFactoryDefaultFilterEntry 10 }

cabhSec2FwFactoryDefaultFilterSourcePortHigh OBJECT-TYPE
    SYNTAX      Unsigned32 (0..65535)
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "If cabhSec2FwFactoryDefaultFilterProtocol is
            udp or tcp, this is the inclusive upper bound
            of the transport-layer source port range that
            is to be matched, otherwise it is ignored
            during matching."
    DEFVAL { 65535 }
    ::= { cabhSec2FwFactoryDefaultFilterEntry 11 }

cabhSec2FwFactoryDefaultFilterDestPortLow OBJECT-TYPE
    SYNTAX      Unsigned32 (0..65535)
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "If cabhSec2FwFactoryDefaultFilterProtocol is
            udp or tcp, this is the inclusive lower bound
            of the transport-layer destination port range
            that is to be matched, otherwise it is ignored
            during matching."
    DEFVAL { 0 }
    ::= { cabhSec2FwFactoryDefaultFilterEntry 12 }

cabhSec2FwFactoryDefaultFilterDestPortHigh OBJECT-TYPE
    SYNTAX      Unsigned32 (0..65535)
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "If cabhSec2FwFactoryDefaultFilterProtocol is
            udp or tcp, this is the inclusive upper bound
            of the transport-layer destination port range
            that is to be matched, otherwise it is ignored
            during matching."
    DEFVAL { 65535 }
    ::= { cabhSec2FwFactoryDefaultFilterEntry 13 }

cabhSec2FwFactoryDefaultFilterContinue OBJECT-TYPE
    SYNTAX      TruthValue
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "This value is always set to true so the PS MUST continue
            scanning and applying rules."
    DEFVAL { true }
    ::= { cabhSec2FwFactoryDefaultFilterEntry 14 }

-- ============================================================
--
--  CableHome 1.1 PS Firewall Local Filter Table
--
--  The cabhSec2FwLocalFilterIpTable can be configured to contain
--  a filtering Ruleset for the PS firewall.  It can be used to
--  support subscriber specific or local filtering rules that
--  are separate from general filtering rules that may be
--  be configured in the docsDevFilterIpTable.
```

```
-- ============================================================

cabhSec2FwLocalFilterIpTable OBJECT-TYPE
    SYNTAX SEQUENCE OF CabhSec2FwLocalFilterIpEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "Contains a configured filtering Ruleset for the
            PS firewall."
    ::= { cabhSec2FwFilter 3 }

cabhSec2FwLocalFilterIpEntry OBJECT-TYPE
    SYNTAX      CabhSec2FwLocalFilterIpEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "Contains a configured filter rule for the PS
            firewall.

            If the PS has not acquired ToD, entries that do not have
            default time settings are ignored.

            Note that a filter time period may include two days
            (e.g., from 10 PM to 4 AM).  A filter time period that
            includes two days is identified by the absolute value of
            the cabhSec2FwLocalFilterIpEndTime being less than the
            absolute value of the cabhSec2FwLocalFilterIpStartTime.
            The cabhSec2FwLocalFilterIpDOW setting and the
            cabhSec2FwLocalFilterIpStartTime value indicate what day
            and time the filter becomes active.  The
            cabhSec2FwLocalFilterIpEndTime indicates when the filter
            becomes inactive on the second day. The maximum filter time
            period that includes two days is 24 hours.

            If cabhSec2FwLocalFilterIpStartTime is less than or equal
            to the cabhSec2FwLocalFilterIpEndTime, the time period
            of the filter falls in the same day."

    INDEX { cabhSec2FwLocalFilterIpIndex }
    ::= { cabhSec2FwLocalFilterIpTable 1 }

CabhSec2FwLocalFilterIpEntry ::= SEQUENCE {
    cabhSec2FwLocalFilterIpIndex           Unsigned32,
    cabhSec2FwLocalFilterIpStatus          RowStatus,
    cabhSec2FwLocalFilterIpControl         INTEGER,
    cabhSec2FwLocalFilterIpIfIndex         InterfaceIndexOrZero,
    cabhSec2FwLocalFilterIpDirection       INTEGER,
    cabhSec2FwLocalFilterIpSaddr           InetAddress,
    cabhSec2FwLocalFilterIpSmask           InetAddress,
    cabhSec2FwLocalFilterIpDaddr           InetAddress,
    cabhSec2FwLocalFilterIpDmask           InetAddress,
    cabhSec2FwLocalFilterIpProtocol        Unsigned32,
    cabhSec2FwLocalFilterIpSourcePortLow   Unsigned32,
    cabhSec2FwLocalFilterIpSourcePortHigh  Unsigned32,
    cabhSec2FwLocalFilterIpDestPortLow     Unsigned32,
    cabhSec2FwLocalFilterIpDestPortHigh    Unsigned32,
    cabhSec2FwLocalFilterIpMatches         Counter32,
    cabhSec2FwLocalFilterIpContinue        TruthValue,
    cabhSec2FwLocalFilterIpStartTime       Unsigned32,
    cabhSec2FwLocalFilterIpEndTime         Unsigned32,
    cabhSec2FwLocalFilterIpDOW             BITS,
    cabhSec2FwLocalFilterIpDescr           SnmpAdminString
}
```

```
cabhSec2FwLocalFilterIpIndex OBJECT-TYPE
    SYNTAX      Unsigned32 (1..2147483647)
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "Index used to order the application of filters.
            The filter with the lowest index is always applied
            first."
    ::= { cabhSec2FwLocalFilterIpEntry 1 }

cabhSec2FwLocalFilterIpStatus OBJECT-TYPE
    SYNTAX      RowStatus
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
            "Controls and reflects the status of rows in this
             table.  Creation of the
             rows may be done via either create-and-wait or
             create-and-go, but the filter is not applied until this
             object is set to (or changes to) active. There is no
             restriction in changing any object in a row while this
             object is set to active."
    ::= { cabhSec2FwLocalFilterIpEntry 2 }

cabhSec2FwLocalFilterIpControl OBJECT-TYPE
    SYNTAX      INTEGER {
                    deny(1),
                    allow(2)
                }
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
            "If set to deny(1), all packets matching this filter
            will be discarded. If set to allow(2), all
            packets matching this filter will be accepted.
            The cabhSec2FwLocalFilterIpContinue object is
            set to true, and therefore the PS MUST continue to
            scan the table for other matches to apply the match
            with the highest cabhSec2FwLocalFilterIpIndex
            value."
    ::= { cabhSec2FwLocalFilterIpEntry 3 }

cabhSec2FwLocalFilterIpIfIndex OBJECT-TYPE
    SYNTAX      InterfaceIndexOrZero
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
            "The index number assigned to this object MUST
            match the IfIndex numbering assigned in the
            ifTable from the Interfaces Group MIB [RFC 2863],
            and as specified in CH 1.1 Spec, Table 6-17 of
            ITU-T Rec. J.192, Numbering Interfaces in the ifTable."
    DEFVAL { 255 }
    ::= { cabhSec2FwLocalFilterIpEntry 4 }

cabhSec2FwLocalFilterIpDirection OBJECT-TYPE
    SYNTAX      INTEGER {
                    inbound(1),
                    outbound(2),
                    both(3)
                }
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
```

```
            "This value represents direction in relationship
            to the assigned cabhSec2FwLocalFilterIpIfIndex
            in this particular rule, meaning that the PS
            MUST represent traffic direction as follows:
            inbound(1)traffic, outbound(2) traffic, or
            both(3)inbound and outbound traffic."
    ::= { cabhSec2FwLocalFilterIpEntry 5 }

cabhSec2FwLocalFilterIpSaddr OBJECT-TYPE
    SYNTAX      InetAddress
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
            "The source IP address, or portion thereof, that is
            to be matched for this filter. The source address
            is first masked (and'ed) against
            cabhSec2FwLocalFilterIpSmask before being compared to this
            value. A value of 0 for this object and 0 for the mask
            matches all IP addresses."
    DEFVAL { '00000000'h }
    ::= { cabhSec2FwLocalFilterIpEntry 6 }

cabhSec2FwLocalFilterIpSmask OBJECT-TYPE
    SYNTAX      InetAddress
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
            "A bit mask that is to be applied to the source
            address prior to matching. This mask is not
            necessarily the same as a subnet mask, but 1's
            bits must be leftmost and contiguous."
    DEFVAL { '00000000'h }
    ::= { cabhSec2FwLocalFilterIpEntry 7 }

cabhSec2FwLocalFilterIpDaddr OBJECT-TYPE
    SYNTAX      InetAddress
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
            "The destination IP address, or portion thereof, that
            is to be matched for this filter. The destination
            address is first masked (and'ed) against
            cabhSec2FwLocalFilterIpDmask
            before being compared  to this value. A value of 0
            for this object and 0 for the mask matches all
            IP addresses."
    DEFVAL { '00000000'h }
    ::= { cabhSec2FwLocalFilterIpEntry 8 }

cabhSec2FwLocalFilterIpDmask OBJECT-TYPE
    SYNTAX      InetAddress
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
            "A bit mask that is to be applied to the destination
            address prior to matching. This mask is not necessarily
            the same as a subnet mask, but 1's bits must be leftmost
            and contiguous."
    DEFVAL { '00000000'h }
    ::= { cabhSec2FwLocalFilterIpEntry 9 }

cabhSec2FwLocalFilterIpProtocol OBJECT-TYPE
    SYNTAX      Unsigned32 (0..65535)
    MAX-ACCESS  read-create
```

```
    STATUS      current
    DESCRIPTION
            "The protocol value that is to be matched. For example:
            icmp is 1, tcp is 6, udp is 17. A value of 65535 matches
            ANY protocol."
    DEFVAL { 65535 }
    ::= { cabhSec2FwLocalFilterIpEntry 10 }

cabhSec2FwLocalFilterIpSourcePortLow OBJECT-TYPE
    SYNTAX      Unsigned32 (0..65535)
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
            "If cabhSec2FwLocalFilterIpProtocol is udp
            or tcp, this is the inclusive lower bound of the
            transport-layer source port range that is to be
            matched, otherwise it is ignored during matching."
    DEFVAL { 0 }
    ::= { cabhSec2FwLocalFilterIpEntry 11 }

cabhSec2FwLocalFilterIpSourcePortHigh OBJECT-TYPE
    SYNTAX      Unsigned32 (0..65535)
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
            "If cabhSec2FwLocalFilterIpProtocol is
            udp or tcp, this is the inclusive upper bound
            of the transport-layer source port range that
            is to be matched, otherwise it is ignored
            during matching."
    DEFVAL { 65535 }
    ::= { cabhSec2FwLocalFilterIpEntry 12 }

cabhSec2FwLocalFilterIpDestPortLow OBJECT-TYPE
    SYNTAX      Unsigned32 (0..65535)
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
            "If cabhSec2FwLocalFilterIpProtocol is
            udp or tcp, this is the inclusive lower bound
            of the transport-layer destination port range
            that is to be matched, otherwise it is ignored
            during matching."
    DEFVAL { 0 }
    ::= { cabhSec2FwLocalFilterIpEntry 13 }

cabhSec2FwLocalFilterIpDestPortHigh OBJECT-TYPE
    SYNTAX      Unsigned32 (0..65535)
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
            "If cabhSec2FwLocalFilterIpProtocol is
            udp or tcp, this is the inclusive upper bound
            of the transport-layer destination port range
            that is to be matched, otherwise it is ignored
            during matching."
    DEFVAL { 65535 }
    ::= { cabhSec2FwLocalFilterIpEntry 14 }

cabhSec2FwLocalFilterIpMatches OBJECT-TYPE
    SYNTAX      Counter32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
```

```
            "Counts the number of times this filter was matched.
             This object is initialized to 0 at boot, or at row
             creation, and is reset only upon reboot."
    ::= { cabhSec2FwLocalFilterIpEntry 15 }

cabhSec2FwLocalFilterIpContinue OBJECT-TYPE
    SYNTAX      TruthValue
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "This value is always set to true so the PS MUST continue
            scanning and applying rules."
    DEFVAL { true }
    ::= { cabhSec2FwLocalFilterIpEntry 16 }

cabhSec2FwLocalFilterIpStartTime OBJECT-TYPE
    SYNTAX      Unsigned32 (0..2359)
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
            "The start time for matching the filter ruleset in the
            specified days indicated in cabhSec2FwLocalFilterIpDOW.
            Time is represented in Military Time, e.g., 8:30 AM is
            represented as 830 and 11:45 PM as 2345. An attempt to set
            this object to an invalid military time value, e.g., 1182,
            returns 'wrongValue' error."
    DEFVAL { 0 }
    ::= { cabhSec2FwLocalFilterIpEntry 17 }

cabhSec2FwLocalFilterIpEndTime OBJECT-TYPE
    SYNTAX      Unsigned32 (0..2359)
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
            "The end time for matching the filter ruleset for the
            days indicated in cabhSec2FwLocalFilterIpDOW.  The filter
            rule associated with this end time MUST not be disabled
            until the minute following the time indicated by this
            MIB object.  If the time period is for two days, identified
            by cabhSec2FwLocalFilterIpEndTime being less than
            cabhSec2FwLocalFilterIpStartTime, then the
            cabhSec2FwLocalFilterIpDOW settings do not apply to this
            MIB object. Time is represented in the same manner as in
            cabhSec2FwLocalFilterIpStartTime. An attempt to set
            this object to an invalid military time value, e.g., 1182,
            returns 'wrongValue' error."
    DEFVAL { 2359 }
    ::= { cabhSec2FwLocalFilterIpEntry 18 }

cabhSec2FwLocalFilterIpDOW OBJECT-TYPE
    SYNTAX BITS {
                    sunday(0),
                    monday(1),
                    tuesday(2),
                    wednesday(3),
                    thursday(4),
                    friday(5),
                    saturday(6)
                }
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
            "If the day of week bit associated with the PS given day
            is '1', this object criteria matches."
```

```
    DEFVAL { 'fe'h }  -- 11111110 Sun-Sat
    ::= { cabhSec2FwLocalFilterIpEntry 19 }

cabhSec2FwLocalFilterIpDescr OBJECT-TYPE
    SYNTAX      SnmpAdminString (SIZE(0..32))
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
            "A filter rule description configured by the
            cable operator or subscriber."
    DEFVAL { "" }
    ::= { cabhSec2FwLocalFilterIpEntry 20 }

--
-- Kerberos MIBs
--

cabhSecKerbPKINITGracePeriod OBJECT-TYPE
    SYNTAX      Unsigned32 (15..600)
    UNITS       "minutes"
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
            "The PKINIT Grace Period is needed by the PS
            to know when it should start retrying to get
            a new ticket. The PS MUST obtain a new Kerberos
            ticket (with a PKINIT exchange),this, many minutes
            before the old ticket expires."
    DEFVAL { 30 }
    ::= { cabhSecKerbBase 1}

cabhSecKerbTGSGracePeriod OBJECT-TYPE
    SYNTAX      Unsigned32 (1..600)
    UNITS       "minutes"
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
            "The TGS Grace Period is needed by the PS to
            know when it should start retrying to get a new
            ticket. The PS MUST obtain a new Kerberos ticket
            (with a TGS Request), this, many minutes before the
            old ticket expires."
    DEFVAL { 10 }
    ::= { cabhSecKerbBase 2 }

cabhSecKerbUnsolicitedKeyMaxTimeout OBJECT-TYPE
    SYNTAX      Unsigned32 (15..600)
    UNITS       "seconds"
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
            "This timeout applies to PS initiated AP-REQ/REP
            key management exchange with NMS. The maximum
            timeout is the value which may not be exceeded in
            the exponential backoff algorithm."
    DEFVAL { 600 }
    ::= { cabhSecKerbBase 3 }

cabhSecKerbUnsolicitedKeyMaxRetries OBJECT-TYPE
    SYNTAX      Unsigned32 (1..32)
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
            "The number of retries the PS is allowed for
```

```
            AP-REQ/REP key management exchange initiation
            with the NMS. This is the maximum number of
            retries before the PS gives up attempting to
            establish an SNMPv3 security association
            with NMS."
    DEFVAL { 8 }
    ::= { cabhSecKerbBase 4 }

cabhSecNotification OBJECT IDENTIFIER ::= { cabhSecMib 3 }
cabhSecConformance  OBJECT IDENTIFIER ::= { cabhSecMib 4 }
cabhSecCompliances  OBJECT IDENTIFIER ::= { cabhSecConformance 1 }
cabhSecGroups       OBJECT IDENTIFIER ::= { cabhSecConformance 2 }

--
--    Notification Group for future extension
--

-- compliance statements

cabhSecCompliance MODULE-COMPLIANCE
    STATUS      deprecated
    DESCRIPTION
            "The compliance statement for CableHome Security."
    MODULE   --cabhSecMib

-- unconditionally mandatory groups

    MANDATORY-GROUPS {
        cabhSecCertGroup,
        cabhSecKerbGroup
        }

-- conditional mandatory groups

    GROUP cabhSecGroup
    DESCRIPTION
            "This group is implemented only for CH 1.0 gateways."
    ::= { cabhSecCompliances 1 }

    cabhSec2Compliance MODULE-COMPLIANCE
    STATUS      current
    DESCRIPTION
            "The compliance statement for CableHome 1.1 Security."
    MODULE   --cabhSecMib

-- unconditionally mandatory groups

    MANDATORY-GROUPS {
        cabhSecCertGroup,
        cabhSecKerbGroup,
        cabhSec2Group
        }
    ::= { cabhSecCompliances 2 }

cabhSecGroup OBJECT-GROUP
    OBJECTS {
        cabhSecFwPolicyFileEnable,
        cabhSecFwPolicyFileURL,
        cabhSecFwPolicyFileHash,
        cabhSecFwPolicyFileOperStatus,
        cabhSecFwPolicyFileCurrentVersion,
        cabhSecFwPolicySuccessfulFileURL,
        cabhSecFwEventType1Enable,
        cabhSecFwEventType2Enable,
```

```
        cabhSecFwEventType3Enable,
        cabhSecFwEventAttackAlertThreshold,
        cabhSecFwEventAttackAlertPeriod
        }
    STATUS      deprecated
    DESCRIPTION
            "Group of objects in CableHome 1.0 Firewall MIB."
    ::= { cabhSecGroups 1 }

cabhSecCertGroup OBJECT-GROUP
    OBJECTS {
        cabhSecCertPsCert
        }
    STATUS      current
    DESCRIPTION
            "Group of objects in CableHome gateway for PS
            Certificate."
    ::= { cabhSecGroups 2 }

cabhSecKerbGroup OBJECT-GROUP
    OBJECTS {
        cabhSecKerbPKINITGracePeriod,
        cabhSecKerbTGSGracePeriod,
        cabhSecKerbUnsolicitedKeyMaxTimeout,
        cabhSecKerbUnsolicitedKeyMaxRetries
        }
    STATUS      current
    DESCRIPTION
            "Group of objects in CableHome gateway for Kerberos."
    ::= { cabhSecGroups 3 }

cabhSec2Group OBJECT-GROUP
    OBJECTS {
        cabhSec2FwEnable,
        cabhSec2FwPolicyFileURL,
        cabhSec2FwPolicyFileHash,
        cabhSec2FwPolicyFileOperStatus,
        cabhSec2FwPolicyFileCurrentVersion,
        cabhSec2FwClearPreviousRuleset,
        cabhSec2FwPolicySelection,
        cabhSec2FwEventSetToFactory,
        cabhSec2FwEventLastSetToFactory,
        cabhSec2FwPolicySuccessfulFileURL,
        cabhSec2FwEventEnable,
        cabhSec2FwEventThreshold,
        cabhSec2FwEventInterval,
        cabhSec2FwEventCount,
        cabhSec2FwEventLogReset,
        cabhSec2FwEventLogLastReset,
        cabhSec2FwLogEventType,
        cabhSec2FwLogEventPriority,
        cabhSec2FwLogEventId,
        cabhSec2FwLogTime,
        cabhSec2FwLogIpProtocol,
        cabhSec2FwLogIpSourceAddr,
        cabhSec2FwLogIpDestAddr,
        cabhSec2FwLogIpSourcePort,
        cabhSec2FwLogIpDestPort,
        cabhSec2FwLogMessageType,
        cabhSec2FwLogReplayCount,
        cabhSec2FwLogMIBPointer,
        cabhSec2FwFilterScheduleStartTime,
        cabhSec2FwFilterScheduleEndTime,
        cabhSec2FwFilterScheduleDOW,
```

```
        cabhSec2FwFactoryDefaultFilterControl,
        cabhSec2FwFactoryDefaultFilterIfIndex,
        cabhSec2FwFactoryDefaultFilterDirection,
        cabhSec2FwFactoryDefaultFilterSaddr,
        cabhSec2FwFactoryDefaultFilterSmask,
        cabhSec2FwFactoryDefaultFilterDaddr,
        cabhSec2FwFactoryDefaultFilterDmask,
        cabhSec2FwFactoryDefaultFilterProtocol,
        cabhSec2FwFactoryDefaultFilterSourcePortLow,
        cabhSec2FwFactoryDefaultFilterSourcePortHigh,
        cabhSec2FwFactoryDefaultFilterDestPortLow,
        cabhSec2FwFactoryDefaultFilterDestPortHigh,
        cabhSec2FwFactoryDefaultFilterContinue
        }
    STATUS      current
    DESCRIPTION
            "Group of objects in CableHome 1.1 Firewall MIB."
    ::= { cabhSecGroups 4 }

END
```

E.6 Cablelabs definition MIB

The CableLabs Definition MIB MUST be implemented as defined below.

```
CLAB-DEF-MIB DEFINITIONS ::= BEGIN
IMPORTS
    MODULE-IDENTITY,
    OBJECT-TYPE,
    enterprises
                    FROM SNMPv2-SMI
    DocsX509ASN1DEREncodedCertificate
                    FROM DOCS-IETF-BPI2-MIB;

cableLabs MODULE-IDENTITY
    LAST-UPDATED "200504081700Z" -- April 8, 2005
    ORGANIZATION "Cable Television Laboratories, Inc."
    CONTACT-INFO
           "Editor: Jean-Francois Mule
            Postal: Cable Television Laboratories, Inc.
                    858 Coal Creek Circle
                    Louisville, Colorado 80027-9750
                    U.S.A.
            Phone:  +1 303-661-9100
            Fax:    +1 303-661-9199
            E-mail: jfm@cablelabs.com
                    mibs@cablelabs.com"
    DESCRIPTION
            "This MIB module defines the namespace organization for the
            CableLabs enterprise OID registry.

            Copyright 1999-2005 Cable Television Laboratories, Inc.
            All rights reserved."

    REVISION "200504081700Z" -- April 8, 2005
    DESCRIPTION
            "This revision, published as CL-SP-MIB-CLABDEF-I05."
    ::= { enterprises 4491 }

-- Sub-tree for Registrations
clabFunction            OBJECT IDENTIFIER ::= { cableLabs 1 }
clabFuncMib2            OBJECT IDENTIFIER ::= { clabFunction 1 }
clabFuncProprietary     OBJECT IDENTIFIER ::= { clabFunction 2 }
```

```
-- Sub-tree for Project Definitions
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 }

-- Sub-tree for Global Security Definitions
clabSecurity            OBJECT IDENTIFIER ::= { cableLabs 3 }
clabSecCertObject       OBJECT IDENTIFIER ::= { clabSecurity 1 }

-- Sub tree for CableLabs cross project common MIB definitions
clabCommonMibs          OBJECT IDENTIFIER ::= { cableLabs 4 }

--
-- CableLabs DOCSIS Project Sub-tree Definitions
--
dsgMIB OBJECT IDENTIFIER
    -- DOCSIS Set-top Gateway (DSG) MIB module
    -- This object identifier points to the MIB module
    -- DOCSIS-SETTOP-GATEWAY-MIB, which is being deprecated by
    -- DSG-IF-MIB MIB module (dsgIfMib).
    -- Reference:
    -- CableLabs DOCSIS Set-top Gateway (DSG) Interface Specification
    ::= { clabProjDocsis 1 }

docsLoadBalMib OBJECT IDENTIFIER
    -- DOCSIS MIB module defining the CMTS configuration parameters to
    -- support Load Balancing requirements."
    ::= { clabProjDocsis 2 }

dsgIfMIB OBJECT IDENTIFIER
    -- DOCSIS Set-top Gateway (DSG) MIB module
    -- Obsoletes DOCSIS-SETTOP-GATEWAY-MIB Module (dsgMib)
    -- defined initially in DOCSIS Set-top Gateway (DSG) Interface
    -- Specification SP-DSG-I01-020228.
    -- Reference:
    -- CableLabs DOCSIS Set-top Gateway (DSG) Interface Specification
    ::= { clabProjDocsis 3 }

dsgIfStdMib OBJECT IDENTIFIER
    -- DOCSIS Set-top Device (DSG) MIB module.
    -- Reference:
    -- CableLabs DOCSIS Set-top Gateway (DSG) Interface Specification
    ::= { clabProjDocsis 4 }

docsIfExt2Mib OBJECT IDENTIFIER
    -- This MIB module contains the management objects that enhance
    -- DOCSIS RFI Interface Extensions. Contains Enhancements to
    -- DOCSIS RFI interface MIB module.
    -- Reference:
    -- CableLabs DOCSIS RFI Interface Specification.
    ::= { clabProjDocsis 5 }

docsTestMIB OBJECT IDENTIFIER
    -- DOCSIS Test MIB module supporting programmable test features
    -- for DOCSIS 2.0 compliant Cable Modems (CM) and Cable Modems
    -- Termination Systems (CMTS).
    -- Reference:
    -- CableLabs DOCSIS 2.0 Testing MIB Specification
    ::= { clabProjDocsis 12 }

sledMib OBJECT IDENTIFIER
    -- eDOCSIS MIB module supporting the Software Loopback Application
```

```
    -- for eDOCSIS (SLED).
    -- Reference:
    -- CableLabs eDOCSIS Specification
    ::= { clabProjDocsis 13 }

--
-- CableLabs CableHome Project Sub-tree Definitions
-- Reference
-- CableLabs CableHome Specification
--
cabhPsDevMib OBJECT IDENTIFIER
    -- CableHome MIB module defining the basic management objects for
    -- the Portal Services logical element of a CableHome compliant
    -- Residential Gateway device. The PS device parameters describe
    -- general PS Device attributes and behaviour characteristics
    ::= { clabProjCableHome 1 }

cabhSecMib OBJECT IDENTIFIER
    -- CableHome MIB module defining the basic management objects for
    -- the firewall and other security features of the Portal Services
    -- element.
    ::= { clabProjCableHome 2 }

cabhCapMib OBJECT IDENTIFIER
    -- CableHome MIB module defining the basic management objects for
    -- the CableHome Address Portal (CAP) function of the Portal
    -- Services element.
    ::= { clabProjCableHome 3 }

cabhCdpMib OBJECT IDENTIFIER
    -- This MIB module supplies the basic management objects for the
    -- CableHome DHCP Portal (CDP) function of the Portal Services
    -- element.
    ::= { clabProjCableHome 4 }

cabhCtpMib OBJECT IDENTIFIER
    -- CableHome MIB module supporting the remote LAN diagnostic
    -- features provided by the CableHome Test Portal (CTP) function
    -- of the Portal Services element.
    ::= { clabProjCableHome 5 }

cabhQosMib OBJECT IDENTIFIER
    -- CABLEHOME QOS MIB Module (cabhQosMib).
    -- This object identifier points to the MIB module
    -- CABH-QOS-MIB, which is being deprecated by
    -- CABH-QOS2-MIB module (cabhQos2Mib).
    -- Reference:
    -- CableLabs CableHome 1.1 Specification
    ::= { clabProjCableHome 6 }

cabhCsaMib OBJECT IDENTIFIER
    -- CableHome MIB module defining management objects for the
    -- configuration and monitoring of CableHome Commercial Services
    -- Annex.
    -- Reference:
    -- CableLabs CableOffice Commercial Services Annex MIB
    -- Specification
    ::= { clabProjCableHome 7 }

cabhQos2Mib OBJECT IDENTIFIER
    -- Obsoletes CABH-QOS-MIB module (cabhQosMib)
    -- defined initially in CABLEHOME 1.1 Interface Specification.
    -- This MIB module defines the Quality of Service Management
    -- Information Base (MIUB) for CableHome UPnP QOS-compliant
```

```
    -- devices.
    -- Reference:
    -- CableLabs CableHome 1.1 Specification
    ::= { clabProjCableHome 8 }

--
-- CableLabs PacketCable Project Sub-tree Definitions
--
pktcMtaMib OBJECT IDENTIFIER
    -- PacketCable MIB module defining the basic management object for
    -- the Multimedia Terminal Adapter (MTA) devices compliant with
    -- PacketCable requirements.
    -- Reference
    -- CableLabs PacketCable MTA Device Provisioning Specification
    ::= { clabProjPacketCable 1 }

pktcSigMib OBJECT IDENTIFIER
    -- PacketCable MIB module defining the basic management object for
    -- the PacketCable MTA Signalling protocols. This version of the MIB
    -- includes common signalling and Network Call Signalling (NCS)
    -- related signalling objects.
    -- Reference
    -- CableLabs PacketCable MTA Device Provisioning Specification
    ::= { clabProjPacketCable 2 }

pktcEventMib OBJECT IDENTIFIER
    -- PacketCable MIB module defining the basic management objects for
    -- event reporting.
    -- Reference
    -- CableLabs PacketCable Management Event Specification
    ::= { clabProjPacketCable 3 }

pktcSecurity OBJECT IDENTIFIER
    -- CableLabs OID reserved for security and used to specify errors
    -- that can be returned for the Kerberos KDC - Provisioning
    -- Server interface, or the MTA-CMS Kerberized IPSec interface, or
    -- the MTA-Provisioning Server Kerberized SNMPv3 interface.
    -- CableLabs PacketCable Security Specification
    ::= { clabProjPacketCable 4 }

pktcLawfulIntercept OBJECT IDENTIFIER
    -- CableLabs OID reserved for the PacketCable Electronic
    -- Surveillance Protocol (PCESP) between the Delivery Function
    -- and Collection Function. This OID is used to define the ASN.1
    -- PCESP messages.
    -- CableLabs PacketCable Electronic Surveillance Protocol
    -- Specification
    ::= { clabProjPacketCable 5 }

--
-- Sub-tree for PacketCable MIB Enhancements
--

pktcEnhancements OBJECT IDENTIFIER ::= { clabProjPacketCable 6 }


-- The following MIB OBJECTS are being introduced for
-- incorporation of new MIB objects (MIB enhancements)
-- proposed to the PacketCable MIB group.
-- This includes new MIB objects being introduced
-- as part of the PacketCable MIB Enhancement efforts
-- and as a place holder for future revisions.
-- This sub-division would facilitate easier incorporation
-- of proposed IETF Drafts/RFCs by keeping enhancements
```

```
-- independent of RFC/Draft changes.
-- For new MIB tables that use previously used indices, it is
-- recommended that the AUGMENT CLAUSE be used to aid SNMP Operations,
-- as deemed necessary.


pktcEnMtaMib OBJECT IDENTIFIER
    -- PacketCable MIB module enhancements to the basic management
    -- objects defined by the MIB group pktcMtaMib for the Multimedia
    -- Terminal Adapter (MTA) devices compliant with PacketCable
    -- requirements.
    -- Reference:
    -- CableLabs PacketCable MTA Device Provisioning Specification.
    ::= { pktcEnhancements 1 }

pktcEnSigMib OBJECT IDENTIFIER
    -- PacketCable MIB module enhancements to the basic management
    -- objects defined by the MIB group pktcSigMib for the
    -- PacketCable MTA Signalling protocols.
    -- Reference:
    -- CableLabs PacketCable MTA Device Provisioning Specification.
    ::= { pktcEnhancements 2 }

pktcEnEventMib OBJECT IDENTIFIER
    -- PacketCable MIB module enhancements to the basic management
    -- objects defined by the MIB group pktcEventMib for event reporting.
    -- Reference:
    -- CableLabs PacketCable Management Event Specification.
    ::= { pktcEnhancements 3 }

pktcEnSecurityMib OBJECT IDENTIFIER
    -- PacketCable MIB module enhancements to the basic management
    -- objects defined by the reserved MIB group pktcSecurity.
    -- Reference:
    -- CableLabs PacketCable Security Specification.
    ::= { pktcEnhancements 4 }

--
--
-- Definition of CableLabs Security Certificate Objects
--
clabSrvcPrvdrRootCACert OBJECT-TYPE
    SYNTAX      DocsX509ASN1DEREncodedCertificate
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "The X509 DER-encoded CableLabs Service Provider Root CA
            Certificate."
    REFERENCE
            "CableLabs CableHome Specification;
            CableLabs PacketCable Security Specification."
    ::= { clabSecCertObject 1 }

clabCVCRootCACert OBJECT-TYPE
    SYNTAX      DocsX509ASN1DEREncodedCertificate
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "The X509 DER-encoded CableLabs CVC Root CA Certificate."
    REFERENCE
            "CableLabs CableHome Specification;
            CableLabs PacketCable Security Specification."
    ::= { clabSecCertObject 2 }
```

```
clabCVCCACert OBJECT-TYPE
    SYNTAX      DocsX509ASN1DEREncodedCertificate
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "The X509 DER-encoded CableLabs CVC CA Certificate."
    REFERENCE
            "CableLabs CableHome Specification;
            CableLabs PacketCable Security Specification."
    ::= { clabSecCertObject 3 }

clabMfgCVCCert OBJECT-TYPE
    SYNTAX      DocsX509ASN1DEREncodedCertificate
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "The X509 DER-encoded Manufacturer CVC Certificate."
    REFERENCE
            "CableLabs CableHome Specification;
            CableLabs PacketCable Security Specification."
    ::= { clabSecCertObject 4 }

clabMfgCACert OBJECT-TYPE
    SYNTAX      DocsX509ASN1DEREncodedCertificate
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "The X509 DER-encoded Manufacturer CA Certificate."
    REFERENCE
            "CableLabs CableHome Specification;
            CableLabs PacketCable Security Specification."
    ::= { clabSecCertObject 5 }

--
-- CableLabs cross project common MIB sub-tree definitions
--

clabUpsMib OBJECT IDENTIFIER
    -- CableLabs cross project MIB module defining the basic management
    -- objects for the configuration and monitoring of the battery
    -- backup and UPS functionality for CableLabs compliant devices.
    ::= { clabCommonMibs 1 }

END
```

E.7 IPCable2Home QoS Portal (CQP) MIB requirements

The IPCable2Home CQP MIB MUST be implemented as defined below.

```
CABH-QOS2-MIB DEFINITIONS ::= BEGIN
IMPORTS
    MODULE-IDENTITY,
    OBJECT-TYPE,
    Unsigned32,
    Gauge32                  FROM SNMPv2-SMI

    TruthValue,
    TimeStamp,
    RowStatus                FROM SNMPv2-TC

    SnmpAdminString          FROM SNMP-FRAMEWORK-MIB
```

```
    OBJECT-GROUP,
    MODULE-COMPLIANCE        FROM SNMPv2-CONF

    InetPortNumber,
    InetAddressType,
    InetAddress              FROM INET-ADDRESS-MIB

    ifIndex                  FROM IF-MIB

    clabProjCableHome        FROM CLAB-DEF-MIB;

cabhQos2Mib MODULE-IDENTITY
    LAST-UPDATED    "200504080000Z" -- April 8, 2005
    ORGANIZATION    "CableLabs Broadband Access Department"
    CONTACT-INFO
            "Kevin Luehrs
            Postal: Cable Television Laboratories, Inc.
            858 Coal Creek Circle
            Louisville, Colorado 80027
            U.S.A.
            Phone:  +1 303-661-9100
            Fax:    +1 303-661-9199
            E-mail: k.luehrs@cablelabs.com; mibs@cablelabs.com"
    DESCRIPTION
            "This MIB module supplies parameters for the
            configuration and monitoring of CableHome
            QoS capabilities."
    ::= { clabProjCableHome 8 }

   -- Textual conventions

-- Notifications
cabhQos2Mib2Notifications   OBJECT IDENTIFIER ::= { cabhQos2Mib 0 }

-- Objects definitions

cabhQos2MibObjects          OBJECT IDENTIFIER ::= { cabhQos2Mib 1 }
cabhQos2Base                OBJECT IDENTIFIER ::= {
                                                    cabhQos2MibObjects 1 }
cabhQos2PsIfAttributes      OBJECT IDENTIFIER ::= {
                                                    cabhQos2MibObjects 2 }
cabhQos2PolicyHolderObjects OBJECT IDENTIFIER ::= {
                                                    cabhQos2MibObjects 3 }
cabhQos2DeviceObjects       OBJECT IDENTIFIER ::= {
                                                    cabhQos2MibObjects 4 }

   --================================================================
   --
   -- PS QoS basic control and configuration
   --
   --================================================================

cabhQos2SetToFactory OBJECT-TYPE
    SYNTAX      TruthValue
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
            "When this object is set to true(1), the PS MUST clear
            all the entries in cabhQos2PolicyTable and
            cabhQos2TrafficClassTable. Reading this object always
            returns false(2)."
    ::= { cabhQos2Base 1 }
```

```
cabhQos2LastSetToFactory OBJECT-TYPE
    SYNTAX      TimeStamp
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "The value of sysUpTime when cabhQos2SetToFactory
            was last set to true. Zero if never reset."
    ::= { cabhQos2Base 2 }

   --================================================================
   --
   --  PS Interface Attributes Table
   --
   --  The cabhQos2PsIfAttribTable replaces the deprecated
   --  cabhPriorityQosPsIfAttribTable and contains the number of
   --  media access priorities and number of queues associated with
   --  each PS interface.
   --
   --================================================================

cabhQos2PsIfAttribTable OBJECT-TYPE
    SYNTAX      SEQUENCE OF CabhQos2PsIfAttribEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "This table contains interface attributes. It includes
            the number of media access priorities and number of
            queues associated with each PS interface in the
            Residential Gateway."
    ::= { cabhQos2PsIfAttributes 1 }

cabhQos2PsIfAttribEntry OBJECT-TYPE
    SYNTAX      CabhQos2PsIfAttribEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "Number of media access priorities and number
            of queues for each PS interface in the Residential
            Gateway. PS does not need to provide support for entries
            associated with Aggregated LAN interfaces (ifIndex 255 and
            254). The PS WAN interfaces are assigned as ifIndex 1 for
            Wan Management and ifIndex 2 for Wan Data; both interfaces
            are indicated in this table as 'WAN interface' with
            ifIndex 1 as the entry identifier."
    INDEX { ifIndex }
    ::= { cabhQos2PsIfAttribTable 1 }

CabhQos2PsIfAttribEntry::= SEQUENCE {
    cabhQos2PsIfAttribNumPriorities   Unsigned32,
    cabhQos2PsIfAttribNumQueues       Unsigned32
    }

cabhQos2PsIfAttribNumPriorities OBJECT-TYPE
    SYNTAX      Unsigned32 (1..8)
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "The number of media access priorities supported
            by this interface."
    ::= { cabhQos2PsIfAttribEntry 1 }

cabhQos2PsIfAttribNumQueues OBJECT-TYPE
    SYNTAX      Unsigned32 (1..8)
    MAX-ACCESS  read-only
```

```
    STATUS      current
    DESCRIPTION
            "The number of queues associated with this interface."
    ::= { cabhQos2PsIfAttribEntry 2 }

   --================================================================
   --
   --  PS UPnP Policy Holder Information
   --
   --  Provides the UPnP Qos admission control and Upnp Policy Holder
   -- control and information to be used by the policy manager.
   --
   --================================================================

cabhQos2PolicyHolderEnabled OBJECT-TYPE
    SYNTAX      TruthValue
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
            "The value true indicates that the Policy Holder entity is
            active and advertised in PS UPnP standard discovery
            mechanisms; false indicates it is disabled."
    DEFVAL { true }
    ::= { cabhQos2PolicyHolderObjects 1 }

cabhQos2PolicyAdmissionControl OBJECT-TYPE
    SYNTAX      INTEGER {
                enabled(1),
                disabled(2)
                }
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
            "Indicates if the QoS Policy Admission Control
            is enabled or disabled for all the traffic requests."
    DEFVAL { disabled }
    ::= { cabhQos2PolicyHolderObjects 2 }

cabhQos2NumActivePolicyHolder OBJECT-TYPE
    SYNTAX      Gauge32 (0..4294967295)
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "Indicates the number of active policy holders the PS
            have discovered in the LAN. This object includes the PS
            Policy Holder if active."
    ::= { cabhQos2PolicyHolderObjects 3 }

cabhQos2PolicyTable OBJECT-TYPE
    SYNTAX      SEQUENCE OF CabhQos2PolicyEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "This table contains the operator and user created
            policies for the management of QoS for applications.
            PS creates non-persistent entries (of type 'upnp') for
            the QoS-aware applications and services discovered
            through UPnP actions in the user part of this table which
            could be converted to persistent entries by user (of type
            'user' or by cable operator of type
            'operatorForHomeUserOnly)."
    ::= { cabhQos2PolicyHolderObjects 4 }

cabhQos2PolicyEntry OBJECT-TYPE
```

```
    SYNTAX      CabhQos2PolicyEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "The indices for these entries."
    INDEX { cabhQos2PolicyOwner, cabhQos2PolicyOwnerRuleId }
    ::= { cabhQos2PolicyTable 1 }

CabhQos2PolicyEntry ::= SEQUENCE {
    cabhQos2PolicyOwner             INTEGER,
    cabhQos2PolicyOwnerRuleId       Unsigned32,
    cabhQos2PolicyRuleOrder         Unsigned32,
    cabhQos2PolicyAppDomain         SnmpAdminString,
    cabhQos2PolicyAppName           SnmpAdminString,
    cabhQos2PolicyServiceProvDomain SnmpAdminString,
    cabhQos2PolicyServiceName       SnmpAdminString,
    cabhQos2PolicyPortDomain        SnmpAdminString,
    cabhQos2PolicyPortNumber        InetPortNumber,
    cabhQos2PolicyIpType            InetAddressType,
    cabhQos2PolicyIpProtocol        Unsigned32,
    cabhQos2PolicySrcIp             InetAddress,
    cabhQos2PolicyDestIp            InetAddress,
    cabhQos2PolicySrcPort           InetPortNumber,
    cabhQos2PolicyDestPort          InetPortNumber,
    cabhQos2PolicyTraffImpNum       Unsigned32,
    cabhQos2PolicyUserImportance    Unsigned32,
    cabhQos2PolicyRowStatus         RowStatus
    }

cabhQos2PolicyOwner OBJECT-TYPE
    SYNTAX      INTEGER {
                operatorOnly(1),
                homeUser(2),
                operatorForHomeUser(3),
                upnp(4)
                }
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "This Index defines the policy creation owner. The entries
            of type 'upnp' are dynamically created by the PS for
            the applications, services and devices that it discovers
            on the LAN with UPnP QoS actions."
    ::= { cabhQos2PolicyEntry 1 }

cabhQos2PolicyOwnerRuleId OBJECT-TYPE
    SYNTAX      Unsigned32 (1..4294967295)
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "Index for the set of rules related to an
            owner index."
    ::= { cabhQos2PolicyEntry 2 }

cabhQos2PolicyRuleOrder OBJECT-TYPE
    SYNTAX      Unsigned32 (0..4294967295)
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
            "The order in which the policy rules are processed within
             An owner."
    DEFVAL { 0 }
    ::= { cabhQos2PolicyEntry 3 }
```

```
cabhQos2PolicyAppDomain OBJECT-TYPE
    SYNTAX      SnmpAdminString (SIZE (0..32))
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
            "Vendor domain name from the Vendor
            application name URN."
    DEFVAL { "" }
    ::= { cabhQos2PolicyEntry 4 }

cabhQos2PolicyAppName OBJECT-TYPE
    SYNTAX      SnmpAdminString (SIZE (0..32))
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
            "Text description of the application."
    DEFVAL { "" }
    ::= { cabhQos2PolicyEntry 5 }

cabhQos2PolicyServiceProvDomain OBJECT-TYPE
    SYNTAX      SnmpAdminString (SIZE (0..32))
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
            "The service Provider Service Domain Name from the
            service Provider URN."
    DEFVAL { "" }
    ::= { cabhQos2PolicyEntry 6 }

cabhQos2PolicyServiceName OBJECT-TYPE
    SYNTAX      SnmpAdminString (SIZE (0..32))
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
            "Text description of the Service."
    DEFVAL { "" }
    ::= { cabhQos2PolicyEntry 7 }

cabhQos2PolicyPortDomain OBJECT-TYPE
    SYNTAX      SnmpAdminString (SIZE (0..32))
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
            "Domain name from the Port URN."
    DEFVAL { "" }
    ::= { cabhQos2PolicyEntry 8 }

cabhQos2PolicyPortNumber OBJECT-TYPE
    SYNTAX      InetPortNumber
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
            "Well known IP transport port of the application."
    DEFVAL { 0 }
    ::= { cabhQos2PolicyEntry 9 }

cabhQos2PolicyIpType OBJECT-TYPE
    SYNTAX      InetAddressType
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
            "The type of InetAddress for cabhQos2PolicySrcIp,
            and cabhQos2PolicyDestIp."
    DEFVAL { ipv4 }
```

```
    ::= { cabhQos2PolicyEntry 10 }

cabhQos2PolicyIpProtocol OBJECT-TYPE
    SYNTAX         Unsigned32 (0..255)
    MAX-ACCESS     read-create
    STATUS         current
    DESCRIPTION
            "The IANA-defined IP protocol number representing
            the IP protocol to match against the IPv4 protocol
            number or the IPv6 Next-Header number in the packet.
            '0' means no protocol is specified as matching criteria
            for policy determination, i.e., QoS policy is
            irrespective of IP protocol."
    REFERENCE
            "http://www.iana.org/assignments/protocol-numbers"
    DEFVAL { 0 }
    ::= { cabhQos2PolicyEntry 11 }

cabhQos2PolicySrcIp OBJECT-TYPE
    SYNTAX      InetAddress
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
            "The IP address to match against the packet's source IP
            address. This may not be a DNS name, but may be an IPv4 or
            IPv6 prefix."
    DEFVAL { '00000000'h }
    ::= { cabhQos2PolicyEntry 12 }

cabhQos2PolicyDestIp OBJECT-TYPE
    SYNTAX      InetAddress
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
            "The IP address to match against the packet's source IP
            address. This may not be a DNS name, but may be an IPv4 or
            IPv6 prefix."
    DEFVAL { '00000000'h }
    ::= { cabhQos2PolicyEntry 13 }

cabhQos2PolicySrcPort OBJECT-TYPE
    SYNTAX      InetPortNumber
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
            "The value that the layer-4 source port number in the
            packet must have in order to match this policy entry."
    DEFVAL { 0 }
    ::= { cabhQos2PolicyEntry 14 }

cabhQos2PolicyDestPort OBJECT-TYPE
    SYNTAX      InetPortNumber
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
            "The value that the layer-4 destination port number in the
             packet must have in order to match this policy entry."
    DEFVAL { 0 }
    ::= { cabhQos2PolicyEntry 15 }

cabhQos2PolicyTraffImpNum OBJECT-TYPE
    SYNTAX      Unsigned32 (0..7)
    MAX-ACCESS  read-create
    STATUS      current
```

```
    DESCRIPTION
            "The Traffic priority being assigned to this policy. The
            final packet tagging is determined by 802.1D rules with
            the priority hierarchy order (highest to lowest priority)
            as defined in 802.1D-2004 table G-2:
            7, 6, 5, 4, 3, 0, 2, 1.
            Note that traffic type '1' and '2' has lower priority than
            '0' (best effort)."
    DEFVAL { 0 }
    ::= { cabhQos2PolicyEntry 16 }

cabhQos2PolicyUserImportance OBJECT-TYPE
    SYNTAX      Unsigned32 (0..255)
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
            "The UPnP relative value to determine the allocation or
            reallocation of resources to multiple streams."
    DEFVAL { 0 }
    ::= { cabhQos2PolicyEntry 17 }

cabhQos2PolicyRowStatus OBJECT-TYPE
    SYNTAX      RowStatus
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
            "The status of this conceptual row. All writable objects
            in this row may be modified at any time. The PS MUST
            NOT allow creation of new entry or modification to an
            existing active entry such that the resulting entry is a
            duplicate entry with respect to the following MIBs
            in an entry:
            cabhQos2PolicyAppDomain,
            cabhQos2PolicyAppNameSnmpAdminString,
            cabhQos2PolicyServiceProvDomainSnmpAdminString,
            cabhQos2PolicyServiceName SnmpAdminString,
            cabhQos2PolicyPortDomain SnmpAdminString,
            cabhQos2PolicyPortNumber InetPortNumber,
            cabhQos2PolicyIpType InetAddressType,
            cabhQos2PolicyIpProtocol Unsigned32,
            cabhQos2PolicySrcIp InetAddress,
            cabhQos2PolicyDestIp InetAddress,
            cabhQos2PolicySrcPort InetPortNumber,
            cabhQos2PolicyDestPort InetPortNumber,

            The entries of type 'upnp' are not persistent while others
            are persistent. The user or the operator can change the
            'upnp' entries and in that case the PS MUST change the
            entry to either 'homeUser' or 'operatorForHomeUser',
            respectively. The PS MUST NOT change the entries of type
            'upnp' to 'operatorOnly'."
    ::= { cabhQos2PolicyEntry 18 }

   --================================================================
   --
   --  PS UPnP QoS Device Information
   --
   --  Contains PS QoS device traffic descriptors as classifiers when
   --  acting as an intermediate device for traffic flows
   --  Qos Device information retrieval from the SNMP WAN interface is
   --  defined in PSDEV-MIB module
   --
   --================================================================
```

```
cabhQos2TrafficClassTable OBJECT-TYPE
    SYNTAX      SEQUENCE OF CabhQos2TrafficClassEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "This table contains the Classifiers being configured
            in the PS as an intermediate QoS device.
            For matching classifiers the PS processes entries
            in a sorted manner, first entries with
            cabhQos2TrafficClassMethod 'static' and then
            'dynamic' entries."
    ::= { cabhQos2DeviceObjects 1 }

cabhQos2TrafficClassEntry OBJECT-TYPE
    SYNTAX      CabhQos2TrafficClassEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "The conceptual row definition of this table.
            Only entries with cabhQos2TrafficClassMethod
            'static' do persist after PS reboot."
    INDEX { cabhQos2TrafficClassMethod, cabhQos2TrafficClassIdx }
    ::= { cabhQos2TrafficClassTable 1 }

CabhQos2TrafficClassEntry::= SEQUENCE {
    cabhQos2TrafficClassMethod         INTEGER,
    cabhQos2TrafficClassIdx            Unsigned32,
    cabhQos2TrafficClassProtocol       Unsigned32,
    cabhQos2TrafficClassIpType         InetAddressType,
    cabhQos2TrafficClassSrcIp          InetAddress,
    cabhQos2TrafficClassDestIp         InetAddress,
    cabhQos2TrafficClassSrcPort        InetPortNumber,
    cabhQos2TrafficClassDestPort       InetPortNumber,
    cabhQos2TrafficClassImpNum         Unsigned32,
    cabhQos2TrafficClassRowStatus      RowStatus
    }

cabhQos2TrafficClassMethod OBJECT-TYPE
    SYNTAX      INTEGER {
                static(1),
                upnp(2)
                }
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "Indicates how this entry has been created.
            'static' indicates that the entry has been
             provisioned via SNMP or related mechanisms
             like a config file.
            'upnp' indicates that the entry was created via UPnP
            Qos actions."
    ::= { cabhQos2TrafficClassEntry 1 }

cabhQos2TrafficClassIdx OBJECT-TYPE
    SYNTAX      Unsigned32 (1..4294967295)
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "The index of this conceptual row entry."
    ::= { cabhQos2TrafficClassEntry 2 }

cabhQos2TrafficClassProtocol OBJECT-TYPE
    SYNTAX      Unsigned32 (0..256)
    MAX-ACCESS  read-create
```

```
    STATUS      current
    DESCRIPTION
            "The IANA IP transport protocol designated for this
            classifier. '0' means no protocol is specified as
            matching criteria."
    DEFVAL { 0 }
    ::= { cabhQos2TrafficClassEntry 3 }

cabhQos2TrafficClassIpType OBJECT-TYPE
    SYNTAX      InetAddressType
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
            "The type of InetAddress for cabhQos2TrafficClassSrcIp,
            and cabhQos2TrafficClassDestIp."
    DEFVAL { ipv4 }
    ::= { cabhQos2TrafficClassEntry 4 }

cabhQos2TrafficClassSrcIp OBJECT-TYPE
    SYNTAX      InetAddress
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
            "The IP address to match against the packet's source IP
            address for this classifier. This may not be a DNS name,
            but may be an IPv4 or IPv6 prefix."
    DEFVAL { '00000000'h }
    ::= { cabhQos2TrafficClassEntry 5 }

cabhQos2TrafficClassDestIp OBJECT-TYPE
    SYNTAX      InetAddress
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
            "The IP address to match against the packet's source IP
            address for this classifier. This may not be a DNS name,
            but may be an IPv4 or IPv6 prefix."
    DEFVAL { '00000000'h }
    ::= { cabhQos2TrafficClassEntry 6 }

cabhQos2TrafficClassSrcPort OBJECT-TYPE
    SYNTAX      InetPortNumber
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
            "The value that the layer-4 source port number in the
            packet must have in order to match this classifier entry."
    DEFVAL { 0 }
    ::= { cabhQos2TrafficClassEntry 7 }

cabhQos2TrafficClassDestPort OBJECT-TYPE
    SYNTAX      InetPortNumber
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
            "The value that the layer-4 destination port number in the
            packet must have in order to match this classifier entry."
    DEFVAL { 0 }
    ::= { cabhQos2TrafficClassEntry 8 }

cabhQos2TrafficClassImpNum OBJECT-TYPE
    SYNTAX      Unsigned32 (0..7)
    MAX-ACCESS  read-create
    STATUS      current
```

```
    DESCRIPTION
            "The traffic priority assigned to this classifier and used
            for the tagging of the packet streams."
    DEFVAL { 0 }
    ::= { cabhQos2TrafficClassEntry 9 }

cabhQos2TrafficClassRowStatus OBJECT-TYPE
    SYNTAX      RowStatus
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
            "The status of this conceptual row. All writable objects
            in rows with cabhQosTrafficMethod 'static' may be
            modified at any time. An SNMP Set to Entries with
            cabhQosTrafficMethod 'upnp' returns an error
            'wrongValue'with the exception of the RowStatus
            object when set to 'destroy'.
            An attempt to create an entry via SNMP with
            cabhQosTrafficMethod UPnP returns error 'wrongValue'."
    ::= { cabhQos2TrafficClassEntry 10 }

   -- Placeholder for notifications.
   --
   --
   -- Conformance definitions
   --
cabhQos2Conformance    OBJECT IDENTIFIER  ::= { cabhQos2Mib 2 }
cabhQos2Compliances    OBJECT IDENTIFIER  ::= { cabhQos2Conformance 1 }
cabhQos2Groups         OBJECT IDENTIFIER  ::= { cabhQos2Conformance 2 }

   -- ==================

   -- compliance statements

cabhQos2Compliance MODULE-COMPLIANCE
    STATUS      current
    DESCRIPTION
            "The compliance statement for devices that implement
            CableHome QoS UPnP capabilities."
    MODULE   --cabhQos2Mib

   -- unconditionally mandatory groups

    MANDATORY-GROUPS {
          cabhQos2Group
          }

 -- conditionally groups

GROUP cabhQos2ClassifierGroup
    DESCRIPTION
            "This group is optional and implemented only for
             traffic between LAN and WAN."

OBJECT cabhQos2PolicyIpType
    SYNTAX InetAddressType { ipv4(1) }
    DESCRIPTION
            "An implementation is only required to support IPv4
            addresses."

OBJECT cabhQos2PolicySrcIp
    SYNTAX InetAddress (SIZE(4))
```

```
    DESCRIPTION
            "An implementation is only required to support IPv4
            addresses."

OBJECT cabhQos2PolicyDestIp
    SYNTAX InetAddress (SIZE(4))
    DESCRIPTION
            "An implementation is only required to support IPv4
            addresses."

OBJECT cabhQos2TrafficClassIpType
    SYNTAX InetAddressType { ipv4(1) }
    DESCRIPTION
            "An implementation is only required to support IPv4
            addresses. "

OBJECT cabhQos2TrafficClassSrcIp
    SYNTAX InetAddress (SIZE(4))
    DESCRIPTION
            "An implementation is only required to support IPv4
            addresses."

OBJECT cabhQos2TrafficClassDestIp
    SYNTAX InetAddress (SIZE(4))
    DESCRIPTION
            "An implementation is only required to support IPv4
            addresses."
    ::= { cabhQos2Compliances 1 }

cabhQos2Group OBJECT-GROUP
    OBJECTS {
    cabhQos2SetToFactory,
    cabhQos2LastSetToFactory,
    cabhQos2PsIfAttribNumPriorities,
    cabhQos2PsIfAttribNumQueues,
    cabhQos2PolicyHolderEnabled,
    cabhQos2PolicyAdmissionControl,
    cabhQos2NumActivePolicyHolder,
    cabhQos2PolicyRuleOrder,
    cabhQos2PolicyAppDomain,
    cabhQos2PolicyAppName,
    cabhQos2PolicyServiceProvDomain,
    cabhQos2PolicyServiceName,
    cabhQos2PolicyPortDomain,
    cabhQos2PolicyPortNumber,
    cabhQos2PolicyIpProtocol,
    cabhQos2PolicyIpType,
    cabhQos2PolicySrcIp,
    cabhQos2PolicyDestIp,
    cabhQos2PolicySrcPort,
    cabhQos2PolicyDestPort,
    cabhQos2PolicyTraffImpNum,
    cabhQos2PolicyUserImportance,
    cabhQos2PolicyRowStatus,
    cabhQos2TrafficClassProtocol,
    cabhQos2TrafficClassIpType,
    cabhQos2PolicySrcIp,
    cabhQos2PolicyDestIp,
    cabhQos2PolicySrcPort,
    cabhQos2PolicyDestPort,
    cabhQos2PolicyTraffImpNum,
    cabhQos2PolicyUserImportance,
    cabhQos2PolicyRowStatus
    }
```

```
    STATUS      current
    DESCRIPTION
            "Group of objects for CableHome QoS management."
    ::= { cabhQos2Groups 1 }

cabhQos2ClassifierGroup OBJECT-GROUP
    OBJECTS {
    cabhQos2TrafficClassProtocol,
    cabhQos2TrafficClassIpType,
    cabhQos2TrafficClassSrcIp,
    cabhQos2TrafficClassDestIp,
    cabhQos2TrafficClassSrcPort,
    cabhQos2TrafficClassDestPort,
    cabhQos2TrafficClassImpNum,
    cabhQos2TrafficClassRowStatus
    }
    STATUS      current
    DESCRIPTION
            "Group of objects for cableHome QoS Packet
            classification."
    ::= { cabhQos2Groups 2 }
END
```

Appendix I

Example of UPnP root device description of IPCable2Home PS

The following XML document provides an example of UPnP Root Device description of IPCable2Home PS.

```
<?xml version="1.0" encoding="utf-8" ?>
- <root xmlns="urn:schemas-upnp-org:device-1-0">
  - <specVersion>
      <major>1</major>
      <minor>0</minor>
    </specVersion>
    <URLBase>URLBase</URLBase>
  - <device>
      <deviceType>urn:schemas-cablelabs-com:device:CableHomePSDevice:1</deviceType>
      <friendlyName>friendlyName</friendlyName>
      <manufacturer>manufacturer</manufacturer>
      <manufacturerURL>manufacturerURL</manufacturerURL>
      <modelDescription>modelDescription</modelDescription>
      <modelName>modelName</modelName>
      <modelNumber>modelNumber</modelNumber>
      <modelURL>modelURL</modelURL>
      <serialNumber>serialNumber</serialNumber>
      <UDN>uuid:CableHomePSDevice-1_0-00AABBCCDDEE</UDN>
      <UPC>upc</UPC>
    - <serviceList>
      - <service>
          <serviceType>urn:schemas-upnp-org:service:QosManager:1</serviceType>
          <serviceId>urn:upnp-org:serviceId:QosManager</serviceId>
          <SCPDURL>/QosManager.xml</SCPDURL>
          <controlURL>/QosManager</controlURL>
          <eventSubURL>/QosManager</eventSubURL>
        </service>
      - <service>
          <serviceType>urn:schemas-upnp-org:service:QosPolicyHolder:1</serviceType>
          <serviceId>urn:upnp-org:serviceId:QosPolicyHolder</serviceId>
          <SCPDURL>/QosPolicyHolder.xml</SCPDURL>
          <controlURL>/QosPolicyHolder</controlURL>
          <eventSubURL>/QosPolicyHolder</eventSubURL>
        </service>
      - <service>
          <serviceType>urn:schemas-upnp-org:service:QosDevice:1</serviceType>
          <serviceId>urn:upnp-org:serviceId:QosDevice</serviceId>
          <SCPDURL>/QosDevice.xml</SCPDURL>
          <controlURL>/QosDevice</controlURL>
          <eventSubURL>/QosDevice</eventSubURL>
        </service>
      </serviceList>
    - <deviceList>
      - <device>
          <deviceType>urn:schemas-upnp-org:device:InternetGatewayDevice:1</deviceType>
          <friendlyName>friendlyName</friendlyName>
          <manufacturer>manufacturer</manufacturer>
          <manufacturerURL>manufacturerURL</manufacturerURL>
          <modelDescription>modelDescription</modelDescription>
          <modelName>modelName</modelName>
          <modelNumber>modelNumber</modelNumber>
          <modelURL>modelURL</modelURL>
          <serialNumber>serialNumber</serialNumber>
          <UDN>uuid:InternetGatewayDevice-1_0-00AABBCCDDEE</UDN>
          <UPC>upc</UPC>
        - <deviceList>
          - <device>
              <deviceType>urn:schemas-upnp-org:device:WANDevice:1</deviceType>
              <friendlyName>friendlyName</friendlyName>
              <manufacturer>manufacturer</manufacturer>
```

```
                <manufacturerURL>manufacturerURL</manufacturerURL>
                <modelDescription>modelDescription</modelDescription>
                 <modelName>modelName</modelName>
                 <modelNumber>modelNumber</modelNumber>
                 <modelURL>modelURL</modelURL>
                 <serialNumber>serialNumber</serialNumber>
                 <UDN>uuid:upnp-WANDevice-1_0-XXXX</UDN>
                 <UPC>upc</UPC>
               - <serviceList>
                  - <service>
                      <serviceType>urn:schemas-upnp-org:service:WANCommonInterfaceConfig:1</serviceType>
                      <serviceId>urn:upnp-org:serviceId:WANCommonInterfaceConfig</serviceId>
                       <SCPDURL>/WANCommonInterfaceConfig.xml</SCPDURL>
                       <controlURL>/WANCommonInterfaceConfig</controlURL>
                      <eventSubURL>/WANCommonInterfaceConfig</eventSubURL>
                    </service>
                 </serviceList>
               - <deviceList>
                  - <device>
                      <deviceType>urn:schemas-upnp-org:device:WANConnectionDevice:1</deviceType>
                       <friendlyName>friendlyName</friendlyName>
                       <manufacturer>manufacturer</manufacturer>
                       <manufacturerURL>manufacturerURL</manufacturerURL>
                       <modelDescription>modelDescription</modelDescription>
                       <modelName>modelName</modelName>
                       <modelNumber>modelNumber</modelNumber>
                       <modelURL>modelURL</modelURL>
                       <serialNumber>serialNumber</serialNumber>
                       <UDN>uuid:upnp-WANConnectionDevice-1_0-XXXX</UDN>
                       <UPC>upc</UPC>
                     - <serviceList>
                        - <service>
                            <serviceType>urn:schemas-upnp-org:service:WANIPConnection:1</serviceType>
                            <serviceId>urn:upnp-org:serviceId:WANIPConnection</serviceId>
                             <SCPDURL>/WANIPConnection.xml</SCPDURL>
                             <controlURL>/WANIPConnection</controlURL>
                             <eventSubURL>/WANIPConnection</eventSubURL>
                          </service>
                       </serviceList>
                    </device>
                 </deviceList>
              </device>
           </deviceList>
           <presentationURL>/index.htm</presentationURL>
        </device>
     </deviceList>
     <presentationURL>/index.htm</presentationURL>
  </device>
</root>
```

SERIES OF ITU-T RECOMMENDATIONS

Series A	Organization of the work of ITU-T
Series D	General tariff principles
Series E	Overall network operation, telephone service, service operation and human factors
Series F	Non-telephone telecommunication services
Series G	Transmission systems and media, digital systems and networks
Series H	Audiovisual and multimedia systems
Series I	Integrated services digital network
Series J	**Cable networks and transmission of television, sound programme and other multimedia signals**
Series K	Protection against interference
Series L	Construction, installation and protection of cables and other elements of outside plant
Series M	Telecommunication management, including TMN and network maintenance
Series N	Maintenance: international sound programme and television transmission circuits
Series O	Specifications of measuring equipment
Series P	Telephone transmission quality, telephone installations, local line networks
Series Q	Switching and signalling
Series R	Telegraph transmission
Series S	Telegraph services terminal equipment
Series T	Terminals for telematic services
Series U	Telegraph switching
Series V	Data communication over the telephone network
Series X	Data networks, open system communications and security
Series Y	Global information infrastructure, Internet protocol aspects and next-generation networks
Series Z	Languages and general software aspects for telecommunication systems

Printed in Switzerland
Geneva, 2007