Data-Over-Cable Service Interface Specifications

Operations Support System Interface Specification

SP-OSSIv2.0-I02-020617

ISSUED
SPECIFICATION

Document Status Sheet

Document Control Number:	SP-OSSIv2.0-I02-020617			
Reference:	Operations Support System Interface Specification			
Revision History:	I01 — First Issued Release, December 31, 2001 I02 — Second Issued Release, June 17, 2002			
Date:	June 17, 2002			
Status Code:	Work in Process	Draft	Issued	Closed
Distribution Restrictions:	CableLabs only	CL Reviewers	CL Vendor	Public

Key to Document Status Codes

Work in Process An incomplete document, designed to guide discussion and generate feedback, that may include several alternative requirements for consideration.

Draft A document in specification format considered largely complete, but lacking review by cable industry and vendors. Drafts are susceptible to substantial change during the review process.

Issued A stable document, which has undergone rigorous member and vendor review and is suitable for product design and development, cross-vendor interoperability, and for certification testing.

Closed A static document, reviewed, tested, validated, and closed to further engineering change requests to the specification through CableLabs.

Contents

Figures

This page intentionally left blank.

Tables

This page intentionally left blank.

1 Scope and Purpose

1.1 Scope

This Specification defines the Network Management requirements to support a DOCSIS 2.0 environment. More specifically, the specification details the SNMPv3 protocol and how it coexists with SNMP v1/v2. The RFCs and Management Information Bases (MIBs) required as a minimum for all advertising, configuring, event notifications, etc. basic network-management principles such as fault, configuration, and performance management are incorporated in this specification for the understanding and management of high-speed cable modem environment.

1.2 Requirements

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

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

This document defines many features and parameters, and a valid range for each parameter is usually specified. Equipment (CM and CMTS) requirements are always explicitly stated. Equipment must comply with all mandatory (MUST and MUST NOT) requirements to be considered compliant with this specification. Support of non-mandatory features and parameter values is optional.

This page intentionally left blank.

2 References (normative/informative)

[DOCSIS 1] DOCSIS Cable Modem Termination System - Network-Side Interface Specification SP-CMTS-NSI-I01-960702

[DOCSIS 2] DOCSIS Cable Modem to Customer Premise Equipment Interface Specification SP-CMCI-I07-020301

[DOCSIS 4] DOCSIS Telephony Return Interface Specification SP-CMTRI-I01-970804

[DOCSIS 5] DOCSIS Radio Frequency Interface Specification SP-RFIv2.0-I02-020617

[DOCSIS 6] DOCSIS Baseline Privacy Plus Interface Specification SP-BPI+-I08-020301

[ID-IGMP] Fenner, W., IGMP-based Multicast Forwarding (“IGMP Proxying”), IETF Internet Draft, http://www.ietf.org/internet-drafts/draft-fenner-igmp-proxy-03.txt.

[IETF3] draft-ietf-ipcdn-igmp-mib-00.txt, H. Abramson, “DOCSIS 1.1 IGMP MIB”, June 1999

[IETF4] draft-ietf-ipcdn-qos-mib-04.txt, Mike Patrick, “Data Over Cable System Quality of Service Management Information Base”, Oct. 18, 2000

[IETF6] Proposed Standard RFC version of BPI+ MIB, “draft-ietf-ipcdn-bpiplus-05.txt”

[IETF7] Proposed Standard RFC version of USB MIB, “draft-ietf-xxxx-xxxx-xxxx-00.txt”

[IETF9] Proposed Standard RFC version of Customer Management MIB, “draft-ietf-ipcdn-subscriber-mib-02.txt”

[NDM-U 3.1] “Network Data Management – Usage (NDM-U) For IP-Based Services”, Version 3.1, IPDR.org, April 15, 2002.

[RFC 1157] Schoffstall, M., Fedor, M., Davin, J. and Case, J., A Simple Network Management Protocol (SNMP), IETF RFC 1157, May, 1990

[RFC 1213] K. McCloghrie and M. Rose. Management Information Base for Network Management of TCP/IP-base internets: MIB-II, IETF RFC 1213, March, 1991

[RFC 1224] L. Steinberg., Techniques for Managing Asynchronously Generated Alerts, IETF RFC 1224, May, 1991

[RFC 1493] E. Decker, P. Langille, A. Rijsinghani, and K.McCloghrie., Definitions of Managed Objects for Bridges, IETF RFC 1493, July, 1993

[RFC 1901] Case, J., McCloghrie, K., Rose, M. and S. Waldbusser, “Introduction to Community-based SNMPv2”, RFC 1901, January, 1996.

[RFC 1903] Case, J., McCloghrie, K., Rose, M., and S. Waldbusser, “Textual Conventions for Version 2 of the Simple Network Management Protocol (SNMPv2)”, RFC 1903, January, 1996.

[RFC 1905] Case, J., McCloghrie, K., Rose, M. and S. Waldbusser, “Protocol Operations for Version 2 of the Simple Network Management Protocol (SNMPv2)”, RFC 1905, January, 1996.

[RFC 1906] Case, J., McCloghrie, K., Rose, M. and S. Waldbusser, "Transport Mappings for Version 2 of the Simple Network Management Protocol (SNMPv2)", RFC 1906, January 1996

[RFC 1907] Case, J., McCloghrie, K., Rose, M. and S. Waldbusser, "Management Information Base for Version 2 of the Simple Network Management Protocol (SNMPv2)", RFC 1907, January 1996.

[RFC 1952] Deutsch, P., "GZIP file format specification version 4.3", RFC 1952, May, 1996.

[RFC 2011] K. McCloghrie, "Category: Standards Track SNMPv2 Management Information Base for the Internet Protocol using SMIv2", November 1996

[RFC 2013] K. McCloghrie, "Category: Standards Track SNMPv2 Management Information Base for the User Datagram Protocol using SMIv2", November 1996

[RFC 2132] S. Alexander, R. Droms. DHCP Options and BOOTP Vendor Extensions. IETF RFC 2132. March, 1997.

[RFC 2233] K. McCloghrie, F. Kastenholz, "The Interfaces Group MIB using SMIv2", November 1997

[RFC 2358] J. Flick, J. Johnson, "Definitions of Managed Objects for the Ethernet-like Interface Types", June 1998

[RFC 2570] J. Case, R. Mundy, D. Partain, B. Stewart, "Introduction to Version 3 of the Internet-standard Network Management Framework", April 1999

[RFC 2571] Harrington, D., Presuhn, R. and B. Wijnen, "An Architecture for Describing SNMP Management Frameworks", RFC 2571, April 1999.

[RFC 2572] Case, J., Harrington, D., Presuhn, R. and B. Wijnen, "Message Processing and Dispatching for the Simple Network Management Protocol (SNMP)", RFC 2572, April 1999

[RFC 2573] Levi, D., Meyer, P. and B. Stewart, "SNMP Applications", RFC 2573, April 1999.

[RFC 2574] Blumenthal, U. and B. Wijnen, "The User-Based Security Model for Version 3 of the Simple Network Management Protocol (SNMPv3)", RFC 2574, April 1999

[RFC 2575] Wijnen, B., Presuhn, R. and K. McCloghrie, "View-based Access Control Model for the Simple Network Management Protocol (SNMP)", RFC 2575, April 1999

[RFC 2576] R. Frye, D. Levi, S. Routhier, B. Wijnen, "Coexistence between Version 1, Version 2, and Version 3 of the Internet-Standard and Network Management Framework", RFC 2576, March 2000.

[RFC 2578] McCloghrie, K., Perkins, D., Schoenwaelder, J., Case, J., Rose, M. and S. Waldbusser, "Structure of Management Information Version 2 (SMIv2)", STD 58, RFC 2578, April 1999

[RFC 2579] McCloghrie, K., Perkins, D., Schoenwaelder, J., Case, J., Rose, M. and S. Waldbusser, "Textual Conventions for SMIv2", STD 58, RFC 2579, April 1999

[RFC 2580] McCloghrie, K., Perkins, D., Schoenwaelder, J., Case, J., Rose, M. and S. Waldbusser, "Conformance Statements for SMIv2", STD 58, RFC 2580, April 1999

[RFC 2669] M. St. Johns, "DOCSIS Cable Device MIB Cable Device Management Information Base for DOCSIS compliant Cable Modems and Cable Modem Termination Systems", August 1999

[RFC 2670] M. St. Johns, “Radio Frequency (RF) Interface Management Information Base for MCNS/ DOCSIS compliant RF interfaces”, August 1999

[RFC 2786] stjohns-snmpv3-dhkeychange-mib-01.txt, Michael C. St. Johns, “Diffie-Helman USM Key MIB August 1999 Diffie-Helman USM Key Management Information Base and Textual Convention”, Aug. 6, 1999

[RFC 2863] K. McCloghrie, F. Kastenholz, “The Interfaces Group MIB”, June 2000.

[RFC 2933] McCloghrie, K., Farinacci, D., Thaler, D., “Internet Group Management Protocol MIB”, RFC 2933

[RFC 3083] R. Woundy, “Baseline Privacy Interface Management Information Base for DOCSIS Compliant Cable Modems and Cable Modem Termination Systems”, RFC 3083, March 2001.

[RFI-MIB-IPCDN-DRAFT] draft-ietf-ipcdn-docs-rfimibv2-04.txt, David Raftus, Aviv Goren, Radio Frequency (RF) Interface Management Information Base for MCNS/DOCSIS compliant RF interfaces”, April, 2002.

This page intentionally left blank.

3 Glossary (informative)

Active Service Flow An admitted Service Flow from the CM to the CMTS which is available for packet transmission.

Address Resolution Protocol (ARP) A protocol of the IETF for converting network addresses to 48-bit Ethernet addresses.

Admitted Service Flow A Service Flow, either provisioned or dynamically signaled, which is authorized and for which resources have been reserved but is not active.

Allocation A group of contiguous mini-slots in a MAP which constitute a single transmit opportunity.

American National Standards Institute (ANSI) A US standards body.

Asynchronous Transfer Mode (ATM) A protocol for the transmission of a variety of digital signals using uniform 53-byte cells.

A-TDMA DOCSIS 2.0 TDMA mode (as distinguished from DOCSIS 1.x TDMA).

Authorization Module The authorization module is an abstract module that the CMTS can contact to authorize Service Flows and Classifiers. The authorization module tells the CMTS whether the requesting CM is authorized for the resources it is requesting.

Availability In cable television systems, availability is the long-term ratio of the actual RF channel operation time to scheduled RF channel operation time (expressed as a percent value) and is based on a bit error rate (BER) assumption.

Bandwidth Allocation Map The MAC Management Message that the CMTS uses to allocate transmission opportunities to CMs.

Bridge Protocol Data Unit (BDU) Spanning tree protocol messages as defined in [ISO/IEC10038].

Broadcast Addresses A predefined destination address that denotes the set of all data network service access points.

Burst A single continuous RF signal from the upstream transmitter, from transmitter on to transmitter off.

Burst Error Second Any Errored Second containing at least 100 errors.

Cable Modem (CM) A modulator-demodulator at subscriber locations intended for use in conveying data communications on a cable television system.

Cable Modem Termination System (CMTS) Cable modem termination system, located at the cable television system head-end or distribution hub, which provides complementary functionality to the cable modems to enable data connectivity to a wide-area network.

Cable Modem Termination System - Network Side Interface (CMTS-NSI) The interface, defined in [DOCSIS3], between a CMTS and the equipment on its network side.

Cable Modem to CPE Interface (CMCI) The interface, defined in [DOCSIS4], between a CM and CPE.

Carrier Hum Modulation The peak-to-peak magnitude of the amplitude distortion relative to the RF carrier signal level due to the fundamental and low-order harmonics of the power-supply frequency.

Carrier-to-Noise Ratio (C/N or CNR) The ratio of signal power to noise power in the defined measurement bandwidth. For digital modulation, CNR = E_s/N_0, the energy-per-symbol to noise-density ratio; the signal power is measured in the occupied bandwidth, and the noise power is normalized to the modulation-rate bandwidth. For video, the measurement bandwidth is 4 MHz.

CCCM CPE Controlled Cable Modem. Refer to the DOCSIS Cable Modem to Customer Premise Equipment Interface (CMCI) specification.

Channel The frequency spectrum occupied by a signal. Usually specified by center frequency and bandwidth parameters.

Chip Each of the 128 bits comprising the S-CDMA spreading codes.

Chip Duration The time to transmit one chip of the S-CDMA spreading code. The inverse of the chip rate.

Chip Rate The rate at which individual chips of the S-CDMA spreading codes are transmitted. (1280 to 5120 kHz)

Classifier A set of criteria used for packet matching according to TCP, UDP, IP, LLC, and/or 802.1P/Q packet fields. A classifier maps each packet to a Service Flow. A Downstream Classifier is used by the CMTS to assign packets to downstream service flows. An Upstream Classifier is used by the CM to assign packets to upstream service flows.

Code Hopping Matrix A shifted version of the reference code matrix (see below) that is used when code hopping is employed to vary the codes used by each CM. The Code Hopping Matrix is either 128 rows by 128 columns (when all 128 codes are active) or is 127 rows by 128 columns (when less than 128 codes are active in the S-CDMA spreader-on frame). When less than 128 codes are active, Code 0 (all ones) is deleted from the matrix, but all remaining codes are still cycled through even if less than 127 codes are active in a frame.

Composite Second Order Beat (CSO) The peak of the average level of distortion products due to second-order non-linearities in cable system equipment.

Composite Triple Beat (CTB) The peak of the average level of distortion components due to third-order non-linearities in cable system equipment.

Cross-Modulation A form of television signal distortion where modulation from one or more television channels is imposed on another channel or channels.

Customer See End User.

Customer Premises Equipment (CPE) Equipment at the end user’s premises; MAY be provided by the end user or the service provider.

Data Link Layer Layer 2 in the Open System Interconnection (OSI) architecture; the layer that provides services to transfer data over the transmission link between open systems.

Distribution Hub A location in a cable television network which performs the functions of a head-end for customers in its immediate area, and which receives some or all of its television program material from a Master Head-end in the same metropolitan or regional area.

Downstream In cable television, the direction of transmission from the head-end to the subscriber.

Drop Cable Coaxial cable that connects to a residence or service location from a directional coupler (tap) on the nearest coaxial feeder cable.

Dynamic Host Configuration Protocol (DHCP) An Internet protocol used for assigning network-layer (IP) addresses.

Dynamic Range The ratio between the greatest signal power that can be transmitted over a multichannel analog transmission system without exceeding distortion or other performance limits, and the least signal power that can be utilized without exceeding noise, error rate or other performance limits.

Electronic Industries Association (EIA) A voluntary body of manufacturers which, among other activities, prepares and publishes standards.

End User A human being, organization, or telecommunications system that accesses the network in order to communicate via the services provided by the network.

Engineering Change Notice The final step in the procedure to change specifications.

Engineering Change Order The second step in the procedure to change specifications. DOCSIS posts ECO to web site EC table and ECO page (with indication of ECO Comment Deadline). DOCSIS issues ECO announcement to DOCSIS-announce and working group mail lists (with indication of ECO Comment Deadline).

Engineering Change Request The first step in the procedure to change specifications. DOCSIS issues ECR number, posts to web site EC table and ECR page. DOCSIS sends ECR to subject area working group mail list (and author).

Errored Second Any 1-sec interval containing at least one bit error.

Extended Subsplit A frequency division scheme that allows bidirectional traffic on a single coaxial cable. Reverse path signals come to the head-end from 5 to 42 MHz. Forward path signals go from the head-end from 50 or 54 MHz to the upper frequency limit.

Feeder Cable Coaxial cables that run along streets within the served area and connect between the individual taps which serve the customer drops.

Fiber Distributed Data Interface (FDDI) A fiber-based LAN standard.

Fiber Node A point of interface between a fiber trunk and the coaxial distribution.

Forward Channel The direction of RF signal flow away from the head-end toward the end user; equivalent to Downstream.

Frame See MAC frame, S-CDMA frame, and MPEG frame.

Group Delay The difference in transmission time between the highest and lowest of several frequencies through a device, circuit or system.

Guard Time Minimum time allocated between bursts in the upstream referenced from the symbol center of the last symbol of a burst to the symbol center of the first symbol of the following burst. The guard time should be at least the duration of five symbols plus the maximum system timing error.

Harmonic Related Carrier (HRC) A method of spacing television channels on a cable television system in exact 6-MHz increments, with all carrier frequencies harmonically related to a common reference.

Head-end The central location on the cable network that is responsible for injecting broadcast video and other signals in the downstream direction. See also Master Head-End, Distribution Hub.

Header Protocol control information located at the beginning of a protocol data unit.

High Frequency (HF) Used in this document to refer to the entire subsplit (5-30 MHz) and extended subsplit (5-42 MHz) band used in reverse channel communications over the cable television network.

High Return A frequency division scheme that allows bi-directional traffic on a single coaxial cable. Reverse channel signals propagate to the head-end above the downstream passband.

Hum Modulation Undesired modulation of the television visual carrier by the fundamental or low-order harmonics of the power supply frequency, or other low-frequency disturbances.

Hybrid Fiber/Coax (HFC) System A broadband bidirectional shared-media transmission system using fiber trunks between the head-end and the fiber nodes, and coaxial distribution from the fiber nodes to the customer locations.

Incremental Related Carriers (IRC) A method of spacing NTSC television channels on a cable television system in which all channels except 5 and 6 correspond to the standard channel plan, used to reduce composite triple beat distortions.

Institute of Electrical and Electronic Engineers (IEEE) A voluntary organization which, among other things, sponsors standards committees and is accredited by the American National Standards Institute.

International Electrotechnical Commission (IEC) An international standards body.

International Organization for Standardization (ISO) An international standards body, commonly known as the International Standards Organization.

Internet Control Message Protocol (ICMP) An Internet network-layer protocol.

Internet Engineering Task Force (IETF) A body responsible, among other things, for developing standards used in the Internet.

Internet Group Management Protocol (IGMP) A network-layer protocol for managing multicast groups on the Internet

Impulse Noise Noise characterized by non-overlapping transient disturbances.

Information Element The fields that make up a MAP and define individual grants, deferred grants, etc.

Internet Protocol (IP) An Internet network-layer protocol.

Interval Usage Code A field in MAPs and UCDs to link burst profiles to grants.

Latency The time, expressed in quantity of symbols, taken for a signal element to pass through a device.

Layer A subdivision of the Open System Interconnection (OSI) architecture, constituted by subsystems of the same rank

Link Layer See Data Link Layer.

Local Area Network (LAN) A non-public data network in which serial transmission is used for direct data communication among data stations located on the user's premises.

Logical Link Control (LLC) procedure In a local area network (LAN) or a Metropolitan Area Network (MAN), that part of the protocol that governs the assembling of data link layer frames and their exchange between data stations, independent of how the transmission medium is shared.

Logical (Upstream) Channel A MAC entity identified by a unique channel ID and for which bandwidth is allocated by an associated MAP message. A physical upstream channel may support multiple logical upstream channels. The associated UCD and MAP messages completely describe the logical channel.

MAC Frame MAC header plus optional PDU.

MAC Service Access Point An attachment to a MAC-sublayer domain.

MAP See Bandwidth Allocation Map.

Master Head-End A head-end which collects television program material from various sources by satellite, microwave, fiber and other means, and distributes this material to Distribution Hubs in the same metropolitan or regional area. A Master Head-End MAY also perform the functions of a Distribution Hub for customers in its own immediate area.

Mean Time to Repair (MTTR) In cable television systems, the MTTR is the average elapsed time from the moment a loss of RF channel operation is detected up to the moment the RF channel operation is fully restored.

Media Access Control (MAC) address The "built-in" hardware address of a device connected to a shared medium.

Media Access Control (MAC) procedure In a subnetwork, that part of the protocol that governs access to the transmission medium independent of the physical characteristics of the medium, but taking into account the topological aspects of the subnetworks, in order to enable the exchange of data between nodes. MAC procedures include framing, error protection, and acquiring the right to use the underlying transmission medium.

Media Access Control (MAC) sublayer The part of the data link layer that supports topology-dependent functions and uses the services of the Physical Layer to provide services to the logical link control (LLC) sublayer.

Micro-reflections Echoes in the forward transmission path due to departures from ideal amplitude and phase characteristics.

Mid Split A frequency division scheme that allows bi-directional traffic on a single coaxial cable. Reverse channel signals propagate to the head-end from 5 to 108 MHz. Forward path signals go from the head-end from 162 MHz to the upper frequency limit. The diplex crossover band is located from 108 to 162 MHz.

Mini-Slot A "mini-slot" is an integer multiple of 6.25-microsecond increments.

Modulation Rate The signaling rate of the upstream modulator (1280 to 5120 kHz). In S-CDMA, the chip rate. In TDMA, the channel symbol rate.

Moving Picture Experts Group (MPEG) A voluntary body which develops standards for digital compressed moving pictures and associated audio.

Multipoint Access User access in which more than one terminal equipment is supported by a single network termination.

Multipoint Connection A connection among more than two data network terminations.

National Cable Television Association (NCTA) A voluntary association of cable television operators which, among other things, provides guidance on measurements and objectives for cable television systems in the USA.

National Television Systems Committee (NTSC) Committee which defined the analog color television broadcast standard used today in North America.

Network Layer Layer 3 in the Open Systems Interconnection (OSI) architecture; the layer that provides services to establish a path between open systems.

Network Management The functions related to the management of data link layer and physical layer resources and their stations across the data network supported by the hybrid fiber/coax system.

Number Of Allocated Codes The total number of codes which a single CM uses in a single S-CDMA frame. This number is determined by the size of the grants in minislots and the mapping of these minislots to S-CDMA frames (note that a CM may receive multiple grants which are mapped to a single S-CDMA frame). The number of allocated codes can be in the range of the number of Codes per Mini-slot to the number of active codes, and may vary from frame to frame, but is constant within an S-CDMA frame.

Open Systems Interconnection (OSI) A framework of ISO standards for communication between different systems made by different vendors, in which the communications process is organized into seven different categories that are placed in a layered sequence based on their relationship to the user. Each layer uses the layer immediately below it and provides a service to the layer above. Layers 7 through 4 deal with end-to-end communication between the message source and destination, and layers 3 through 1 deal with network functions.

Organizationally Unique Identifier (OUI) A 3-octet IEEE assigned identifier that can be used to generate Universal LAN MAC addresses and Protocol Identifiers per ANSI/IEEE Std 802 for use in Local and Metropolitan Area Network applications.

Packet Identifier (PID) A unique integer value used to identify elementary streams of a program in a single- or multi-program MPEG-2 stream.

Partial Grant A grant that is smaller than the corresponding bandwidth request from the CM.

Payload Header Suppression The suppression of the header in a payload packet. (*e.g.*, the suppression of the Ethernet header in forwarded packets)

Payload Unit Start Indicator (PUSI) A flag in an MPEG header. A value of 1 indicates the presence of a pointer field as the first byte of the payload.

Physical (PHY) Layer Layer 1 in the Open System Interconnection (OSI) architecture; the layer that provides services to transmit bits or groups of bits over a transmission link between open systems and which entails electrical, mechanical and handshaking procedures.

Physical Media Dependent (PMD) Sublayer A sublayer of the Physical Layer which is concerned with transmitting bits or groups of bits over particular types of transmission link between open systems and which entails electrical, mechanical and handshaking procedures.

Primary Service Flow All CMs have a Primary Upstream Service Flow and a Primary Downstream Service Flow. They ensure that the CM is always manageable and they provide a default path for forwarded packets that are not classified to any other Service Flow

Program-Specific Information (PSI) In MPEG-2, normative data necessary for the demultiplexing of Transport Streams and the successful regeneration of programs.

Program Stream In MPEG-2, a multiplex of variable-length digital video and audio packets from one or more program sources having a common time-base.

Protocol A set of rules and formats that determines the communication behavior of layer entities in the performance of the layer functions.

Provisioned Service Flow A Service Flow that has been provisioned as part of the Registration process, but has not yet been activated or admitted. It may still require an authorization exchange with a policy module or external policy server prior to admission.

QoS Parameter Set The set of Service Flow Encodings that describe the Quality of Service attributes of a Service Flow or a Service Class.

Quadrature Amplitude Modulation (QAM) A method of modulating digital signals onto a radio-frequency carrier signal involving both amplitude and phase coding.

Quadrature Phase-Shift Keying (QPSK) A method of modulating digital signals onto a radio-frequency carrier signal using four phase states to code two digital bits.

Radio Frequency (RF) In cable television systems, this refers to electromagnetic signals in the range 5 to 1000 MHz.

Reference Code Matrix A 128-by-128 element matrix formed by stacking successive spreading codes on top of each other, i.e., the bottom row of the reference code matrix is Code 0 (all ones) and the top row is Code 127. The code elements are placed in the matrix from right to left, i.e., the right-most column of the code matrix is the first element of each code, and the left-most column is the last element of each code.

Request For Comments (RFC) A technical policy document of the IETF; these documents can be accessed on the World Wide Web at http://www.rfc-editor.org/.

Return Loss The parameter describing the attenuation of a guided wave signal (*e.g.*, via a coaxial cable) returned to a source by a device or medium resulting from reflections of the signal generated by the source.

Reverse Channel The direction of signal flow towards the head-end, away from the subscriber; equivalent to Upstream.

Routing Information Protocol (RIP) A protocol of the IETF for exchanging routing information about IP networks and subnets.

S-CDMA Frame A two dimensional representation of mini-slots, where the dimensions are codes and time. An S-CDMA frame is composed of p active codes in the code dimension and K spreading intervals in the time dimension. Within the S-CDMA frame, the number of mini-slots is determined by the number of codes per mini-slot (c) and p, the number of active codes in the S-CDMA frame. Each S-CDMA frame thus contains s mini-slots, where s=p/c, and each mini-slot contains c*K information (QAM) symbols.

S-CDMA Subframe A subframe is a vertically-smaller subset of an S-CDMA frame over which interleaving is performed, where the vertical dimension is R' codes, where $R' \leq p$ (the number of active codes). A subframe is generally used to constrain the interleaving region to be of a similar size to the Reed-Solomon codeword in order to provide protection from impulse noise.

Security Association Identifier A Baseline Privacy security identifier between a CMTS and a CM.

Service Access Point (SAP) The point at which services are provided by one layer, or sublayer to the layer immediately above it.

Service Class A set of queuing and scheduling attributes that is named and that is configured at the CMTS. A Service Class is identified by a Service Class Name. A Service Class has an associated QoS Parameter Set.

Service Class Name An ASCII string by which a Service Class may be referenced in modem configuration files and protocol exchanges.

Service Data Unit (SDU) Information that is delivered as a unit between peer service access points

Service Flow A MAC-layer transport service which:

- Provides unidirectional transport of packets from the upper layer service entity to the RF;
- Shapes, polices, and prioritizes traffic according to QoS traffic parameters defined for the Flow.

Service Flow Identifier (SFID) An identifier assigned to a service flow by the CMTS. [32 bits]

Service Flow Reference A message parameter in Configuration Files and Dynamic Service MAC messages used to associate Classifiers and other objects in the message with the Service Flow Encodings of a requested Service Flow.

Service Identifier (SID) A Service Flow Identifier assigned by the CMTS (in addition to a Service Flow Identifier) to an Active or Admitted Upstream Service Flow. [14 bits]

Simple Network Management Protocol (SNMP) A network management protocol of the IETF.

Spectrum Management System (SMS) A system, defined in [SMS], for managing the RF cable spectrum.

Spread Symbol Or Spreading Interval At the output of the spreader, a group of 128 chips which comprise a single S-CDMA spreading code, and are the result of spreading a single information (QAM) symbol. One spread symbol = one spreading interval = 128 chips = one information (QAM) symbol.

Spreader-Off S-CDMA Burst A transmission from a single CM in a spreader-off frame on an S-CDMA channel defined by the time in which the CM's transmitter turns on to the time it turns off. There will generally be several spreader off bursts in a spreader-off frame.

Spreader-Off S-CDMA Frame TDMA mini-slots on an S-CDMA channel in which the spreader is turned off. These are differentiated from TDMA bursts on a TDMA channel in that, for example, the number of mini-slots per spreader-off SCDMA burst frame is constrained to be the same as the number of mini-slots in a spreader-on SCDMA frame (s). This number of mini-slots will be less than the number of TDMA mini-slots in a TDMA channel over the same time interval if the number of active codes is significantly less than 128.

Spreading Interval Time to transmit a single complete S-CDMA spreading code, equal to the time to transmit 128 chips. Also, time to transmit a single information (QAM) symbol on an S-CDMA channel. See also Spread Symbol.

Sub-Channel A logical channel sharing same upstream spectrum (RF center frequency and RF channel) with other logical channels.

Sublayer A subdivision of a layer in the Open System Interconnection (OSI) reference model.

Subnetwork Subnetworks are physically formed by connecting adjacent nodes with transmission links.

Subnetwork Access Protocol (SNAP) An extension of the LLC header to accommodate the use of 802-type networks as IP networks.

Subscriber See End User.

Subsplit A frequency-division scheme that allows bi-directional traffic on a single cable. Reverse path signals come to the head-end from 5 to 30 (up to 42 on Extended Subsplit systems) MHz. Forward path signals go from the head-end from 50 or 54 MHz to the upper frequency limit of the cable network.

Subsystem An element in a hierarchical division of an Open System that interacts directly with elements in the next higher division or the next lower division of that open system.

System Clock Period The period of the 10.24 MHz system clock, nominally 97.65625 ns.

Systems Management Functions in the application layer related to the management of various open systems Interconnection (OSI) resources and their status across all layers of the OSI architecture.

Tick 6.25-microsecond time intervals that are the reference for upstream mini-slot definition and upstream transmission times.

Tilt Maximum difference in transmission gain of a cable television system over a given bandwidth (typically the entire forward operating frequency range).

Transit Delay The time difference between the instant at which the first bit of a PDU crosses one designated boundary, and the instant at which the last bit of the same PDU crosses a second designated boundary.

Transmission Control Protocol (TCP) A transport-layer Internet protocol which ensures successful end-to-end delivery of data packets without error.

Transmission Convergence Sublayer A sublayer of the Physical Layer that provides an interface between the Data Link Layer and the PMD Sublayer.

Transmission Link The physical unit of a subnetwork that provides the transmission connection between adjacent nodes.

Transmission Medium The material on which information signals may be carried; *e.g.*, optical fiber, coaxial cable, and twisted-wire pairs.

Transmission System The interface and transmission medium through which peer physical layer entities transfer bits.

Transmit On/Off Ratio In multiple-access systems, the ratio between the signal powers sent to line when transmitting and when not transmitting.

Transport Stream In MPEG-2, a packet-based method of multiplexing one or more digital video and audio streams having one or more independent time bases into a single stream.

Trivial File-Transfer Protocol (TFTP) An Internet protocol for transferring files without the requirement for user names and passwords that is typically used for automatic downloads of data and software.

Trunk Cable Cables that carry the signal from the head-end to groups of subscribers. The cables can be either coaxial or fiber depending on the design of the system.

Type/Length/Value (TLV) An encoding of three fields, in which the first field indicates the type of element, the second the length of the element, and the third field the value.

Upstream The direction from the subscriber location toward the head-end.

Upstream Channel Descriptor (UCD) The MAC Management Message used to communicate the characteristics of the upstream physical layer to the cable modems.

4 Abbreviations

ANSI American National Standards Institute

ARP Address Resolution Protocol

ASCII American Standard Code for Information Interchange

ATM Asynchronous Transfer Mode

BPDU Bridge Protocol Data Unit

BPI Baseline Privacy Interface

CA Certificate Authority

CATV Community Access Television, Cable Television

CCCM CPE Controlled Cable Modem

CM Cable Modem

CMCI Cable Modem to CPE Interface

CMTS Cable Modem Termination System

CMTS-NSI Cable Modem Termination System - Network Side Interface

CPE Customer Premises Equipment

CSA Code Signing Agent

CVC Code Verification Certificate

DCC Dynamic Channel Change

DES Digital Encryption Standard

DH Diffie-Helman

DHCP Dynamic Host Configuration Protocol

DOCSIS 1.x Abbreviation for “DOCSIS 1.0 or 1.1”

DOCSIS Data-Over-Cable Service Interface Specifications

ECN Engineering Change Notice

ECO Engineering Change Order

ECR Engineering Change Request

FDDI Fiber Distributed Data Interface

FTP File Transfer Protocol

HFC Hybrid Fiber/Coax (HFC) System

ICMP Internet Control Message Protocol

IE Information Element

IEEE Institute of Electrical and Electronic Engineers

IETF Internet Engineering Task Force

IGMP Internet Group Management Protocol

IP Internet Protocol

IPCDN Internet Protocol over Cable Data Network (IETF working group)

IPDR Internet Protocol Detail Record

IUC Interval Usage Code

LAN Local Area Network

LLC Logical Link Control procedure

MAC Media Access Control procedure

MIB Management Information Base

MPEG Moving Picture Experts Group

MSAP MAC Service Access Point

MSO Mulitple System Operator

MTA Multimedia Terminal Adapter

NMS Network Management System

OID Object Identifier

OSI Open Systems Interconnection

OSSI Operations Support System Interface

OUI Organization Unique Identifier

PCI Peripheral Component Interconnect

PDU Payload Data Unit

PHS Payload Header Suppression

PHY Physical (PHY) Layer

PID Packet Identifier

PMD Physical Media Dependent (PMD) Sublayer

PSI Program-Specific Information

QAM Quadrature Amplitude Modulation

QoS Quality of Service

QPSK Quadrature Phase-Shift Keying

RFC Request for Comments

RFI Radio Frequency Interface

RO Read Only

RW Read/Write

SAID Security Association Identifier

SCN Service Class Name

SF Service Flow

SFID Service Flow Identifier

SID Service Identifier

SLA Service Level Agreement

SMI Structure of Management Informationhh

SMS Spectrum Management System

SNAP Subnetwork Access Protocol

SNMP Simple Network Management Protocol

SNMPv1 Version 1 of the Simple Network Management Protocol

SNMPv2c Version 2C of the Simple Network Management Protocol

SNMPv3 Version 3 of the Simple Network Management Protocol

STP Spanning Tree Protocol

TCP Transmission Control Protocol

TFTP Trivial File-Transfer Protocol

TLV Type/Length/Value

UCC Upstream Channel Change

UDP User Datagram Protocol

USB Universal Synchronous Bus

USM User-based Security Model (USM) for version 3 of the Simple Network Management Protocol

VACM View-based Access Control Model for the simple Network Management Protocol (SNMP)

VoIP Voice over Internet Protocol

XML Extensible Markup Language

5 SNMP Protocol

The SNMPv3 protocol has been selected as the communication protocol for management of data-over-cable services and MUST be implemented. Although SNMPv3 offers advantages, many management systems may not be capable of supporting SNMPv3 agents; therefore, support of SNMPv1 and SNMPv2c is also required and MUST be implemented.

The following IETF SNMP-related RFCs MUST be implemented:

RFC 2570	Introduction to Version 3 of the internet-standard Network Management
RFC 2571	An Architecture for Describing SNMP Management Frameworks
RFC 2572	Message Processing and Dispatching for the Simple Network Management Protocol (SNMP)
RFC 2573	SNMP Applications
RFC 2574	User-based Security Model (USM) for version 3 of the Simple Network Management Protocol (SNMPv3)
RFC 2575	View-based Access Control Model (VACM) for the simple Network Management Protocol (SNMP)
RFC 1905	Protocol Operations for Version 2 of the Simple Network Management Protocol (SNMPv2)
RFC 1906	Transport Mappings for Version 2 of the Simple Network Management Protocol (SNMPv2)
RFC 1907	Management Information Base for Version 2 of the Simple Network Management Protocol (SNMPv2)
RFC 1901	Introduction to Community-based SNMPv2
RFC 1157	A Simple Network Management Protocol

For support of SMIv2, the following IETF SNMP-related RFCs MUST be implemented:

RFC 2578	Structure of Management Information Version 2 (SMIv2)
RFC 2579	Textual Conventions for SMIv2
RFC 2580	Conformance Statements for SMIv2
RFC 2786	Diffie-Helman USM Key

5.1 SNMP Mode for DOCSIS 2.0-compliant CMTSes

DOCSIS 2.0-compliant CMTSes MUST support SNMPv1, SNMPv2c, and SNMPv3 and SNMP coexistence as described by RFC 2571 and RFC 2576, and MAY support SNMPv1 and SNMPv2c vendor proprietary solutions, including SNMP v1/v2c NmAccess mode, with the following requirements:

a) DOCSIS 2.0 compliant CMTS MUST operate in SNMP coexistence mode (not using docsDevNmAccessTable); additionally, SNMP coexistence mode MAY be disabled, by vendor proprietary configuration control, to allow the CMTS to support SNMPv1, SNMPv2c vendor proprietary solutions, including SNMP v1/v2c NmAccess mode (using docsDevNmAccessTable).

b) CMTSes in SNMPv1/v2c NmAccess mode (using Cable Device MIB docsDevNmAccessTable) MUST operate subject to the following requirements and limitations:

- Only SNMPv1/v2c packets are processed
- SNMPv3 packets are dropped
- The docsDevNmAccessTable controls SNMP access and SNMP trap destinations as described in RFC 2669
- None of the SNMPv3 MIBs (Community MIB, TARGET-MIB, VACM-MIB, USM-MIB, NOTIFICATION-MIB) is accessible

c) CMTS SNMPv1, SNMPv2c vendor-proprietary solutions MUST operate subject to the following requirements and limitations:

- Only SNMPv1/v2c packets are processed
- SNMPv3 packets are dropped
- Vendor-proprietary solutions MUST control SNMP access and SNMP trap destinations
- None of the SNMPv3 MIBs (Community MIB, TARGET-MIB, VACM-MIB, USM-MIB, NOTIFICATION-MIB) are accessible.

d) CMTS SNMP Coexistence Mode MUST operate subject to the following requirements and limitations:

- SNMP v1/v2c/v3 Packets are processed as described by RFC2571-RFC2576.
- docsDevNmAccessTable is not accessible. (If the CMTS also support Cable Device MIB)
- Access control and trap destinations are determined by the snmpCommunityTable, Notification MIB, Target MIB, VACM-MIB, and USM-MIB
- Community MIB controls the translation of SNMPv1/v2c packet community string into securityName which select entries in the USM MIB. Access control is provided by the VACM MIB.
- USM MIB and VACM MIB controls SNMPv3 packets
- Trap destinations are specified in the Target MIB and Notification MIB

5.1.1 Key Change Mechanism

DOCSIS 2.0-compliant CMTSes SHOULD use the key-change mechanism specified in RFC 2786. CMTSes MUST always support the key-change mechanism described in RFC 2574 to comply with the industry-wide SNMPv3 standard.

5.2 SNMP Mode for DOCSIS 2.0-compliant CMs

DOCSIS 2.0-compliant CMs (in 2.0, 1.1, and 1.0 modes) MUST support SNMPv1, SNMPv2c, and SNMPv3 as well as SNMP-coexistence (RFC 2576) subject to the following requirements:

a) Before completion of registration, the CM MUST operate as follows (in some CCCM implementations, SNMP MAY be made inaccessible from CPE for security reasons; in such implementations, access to a similar set of MIB objects SHOULD be provided in a diagnostic utility):

- SNMPv1/v2c read-only Access to all MIB variables which are required to be in view during SNMPv1/v2c operation is allowed from the CMCI port. No access is allowed from the RF port.
- SNMPv1/v2c packets are accepted which contain any community string
- All SNMPv3 packets are dropped
- Access SHOULD be prohibited to any MIB variable that would allow determination of the modem's IP address, like the MIB-2 IpAddrTable
- None of the SNMPv3 MIBs (Community MIB, TARGET-MIB, VACM-MIB, USM-MIB, NOTIFICATION-MIB) is accessible, except that they may be set from the config file

- None of the elements in the SNMP-USM-DH-OBJECTS-MIB is accessible except that they may be set from the configuration file
- The registration request MUST be sent and registration MUST be completed after successful processing of all MIB elements in the config file, but before beginning the calculation of the public values in the USMDHKickstart Table

b) The content of the CM config file determines the CM SNMP mode after registration.

- CM is in SNMPv1/v2c docsDevNmAccess mode if the CM configuration file contains ONLY docsDevNmAccessTable setting for SNMP access control
- If the configuration file does not contain SNMP access control items (docsDevNmAccessTable or snmpCommunityTable or TLV 34.1/34.2 orTLV38), the CM is in NmAccess mode
- CM is in SNMP coexistence mode if the CM configuration file contains snmpCommunityTable setting and/or TLV type 34.1 and 34.2. and/or TLV type 38. In this case, any entries made to the docsDevNmAccessTable are ignored.

c) After completion of registration, the modem operates in one of 2 modes. The operating mode is determined by the contents of the config file as described above.

SNMPv1/v2c NmAccess Mode (using docsDevNmAccess Table)

- Only SNMPv1/v2c packets are processed
- SNMPv3 packets are dropped
- docsDevNmAccessTable controls access and trap destinations as described in RFC 2669
- None of the SNMPv3 MIBs (Community MIB, TARGET-MIB, VACM-MIB, USM-MIB, NOTIFICATION-MIB) is accessible.

SNMP Coexistence Mode

During calculation of USMDHKickstartTable public values:

- The modem MUST NOT allow any SNMP access from the RF port
- The modem MAY continue to allow access from the CPE port with the limited access as configured by USM MIB, community MIB and VACM-MIB

After calculation of USMDHKickstartTable public values:

- The modem MUST send the cold start or warm start trap to indicate that the modem is now fully SNMPv3 manageable
- SNMPv1/v2c/v3 Packets are processed as described by RFC 2571 and RFC 2576
- docsDevNmAccessTable is not accessible
- Access control and trap destinations are determined by the snmpCommunityTable, Notification MIB, Target MIB, VACM-MIB, and USM-MIB
- Community MIB controls the translation of SNMPv1/v2c packet community string into security name which select entries in the USM MIB. Access control is provided by the VACM MIB.
- USM MIB and VACM MIB controls SNMPv3 packets
- Trap destinations are specified in the Target MIB and Notification MIB.

d) In case of failure to complete SNMPv3 initialization (i.e., NMS cannot access CM via SNMPv3 PDU), the CM is in co-existence mode and will allow SNMPv1/v2c access if and only if the community MIB entries (and related entries) are configured.

5.2.1 SNMPv3 Initialization and Key changes

DOCSIS 2.0-compliant CMs MUST support the “SNMPv3 Initialization” and “DH Key Changes” requirements specified in the following sections.

5.2.2 SNMPv3 Initialization

For each of up to 5 different security names, the Manager generates a pair of numbers. First, the Manager generates a random number Rm.

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

z = g ^ Rm MOD p

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

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

TLV type 34.1 (SNMPv3 Kickstart Security Name) = docsisManager
TLV type 34.2 (SNMPv3 Kickstart Public Number) = z

The CM MUST support the VACM entries defined in Section 5.2.4. Only VACM entries specified by the corresponding security name in the CM configuration file will (MUST) be active.

During the CM 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 = “docsisManager”

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
usmuserAuthKeyChange: (derived from set value)
usmUserOwnAuthKeyChange: (derived from set value)
usmUserPrivProtocol: usmDESPrivProtocol
usmUserPrivKeyChange: (derived from set value)
usmUserOwnPrivKeyChange: (derived from set value
usmUserPublic
usmUserStorageType: permanent
usmUserStatus: active

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

After the CM has registered with the CMTS:

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

   c = g ^ xa MOD p
   where g is the from the set of Diffie-Helman parameters, and p is the prime from those parameters

At this point:

usmDHKickstartMyPublic.1 = “c” (octet string)
usmDHKickstartMgrPublic.1 = “z” (octet string)
usmDHKickstartSecurityName.1 = “docsisManager”

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

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

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

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

   At this point the CM 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.

   DOCSIS 2.0-compliant CMs 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 CM’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.

   DOCSIS 2.0-compliant CMs 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.

   The SNMP manager gets the value of the CM’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 CM.

5.2.3 DH Key Changes

DOCSIS 2.0-compliant CMs MUST support the key-change mechanism specified in RFC 2786.

5.2.4 VACM Profile

This section addresses the default VACM profile for DOCSIS CMs operating in SNMP Coexistence mode.

The following VACM entries MUST be included by default in a compliant CM:

- The system manager, with full read/write/config access:

  vacmSecurityModel: 3 (USM)
  vacmSecurityName: docsisManager
  vacmGroupName: docsisManager
  vacmSecurityToGroupStorageType: permanent
  vacmSecurityToGroupStatus: active

- An operator/CSR with read/reset access to full modem:

  vacmSecurityModel: 3 (USM)
  vacmSecurityName: docsisOperator
  vacmGroupName: docsisOperator
  vacmSecurityToGroupStorageType: permanent
  vacmSecurityToGroupStatus: active

- RF Monitoring with read access to RF plant statistics:

  vacmSecurityModel: 3 (USM)
  vacmSecurityName: docsisMonitor
  vacmGroupName: docsisMonitor
  vacmSecurityToGroupStorageType: permanent
  vacmSecurityToGroupStatus: active

- User debugging with read access to ‘useful’ variables:

  vacmSecurityModel: 3 (USM)
  vacmSecurityName: docsisUser
  vacmGroupName: docsisUser
  vacmSecurityToGroupStorageType: permanent
  vacmSecurityToGroupStatus: active

- Group name to view translations

  vacmGroupName: docsisManager
  vacmAccessContextPrefix: ‘’
  vacmAccessSecurityModel: 3 (USM)
  vacmAccessSecurityLevel: AuthPriv
  vacmAccessContextMatch: exact
  vacmAccessReadViewName: docsisManagerView
  vacmAccessWriteViewName: docsisManagerView
  vacmAccessNotifyViewName: docsisManagerView
  vacmAccessStorageType: permanent
  vacmAccessStatus: active

  vacmGroupName: docsisOperator
  vacmAccessContextPrefix: ‘’
  vacmAccessSecurityModel: 3 (USM)
  vacmAccessSecurityLevel: AuthPriv & AuthNoPriv
  vacmAccessContextMatch: exact
  vacmAccessReadViewName: docsisManagerView
  vacmAccessWriteViewName: docsisOperatorWriteView
  vacmAccessNotifyViewName: docsisManagerView
  vacmAccessStorageType: permanent
  vacmAccessStatus: active

  vacmGroupName: docsisMonitor
  vacmAccessContextPrefix: ‘’
  vacmAccessSecurityModel: 3 (USM)

vacmAccessSecurityLevel: AuthNoPriv
vacmAccessContextMatch: exact
vacmAccessReadViewName: docsisMonitorView
vacmAccessWriteViewName: ‘’
vacmAccessNotifyViewName: docsisMonitorView
vacmAccessStorageType: permanent
vacmAccessStatus: active

vacmGroupName: docsisUser
vacmAccessContextPrefix: ‘’
vacmAccessSecurityModel: 3 (USM)
vacmAccessSecurityLevel: AuthNoPriv
vacmAccessContextMatch: exact
vacmAccessReadViewName: docsisUserView
vacmAccessWriteViewName: ‘’
vacmAccessNotifyViewName: ‘’
vacmAccessStorageType: permanent
vacmAccessStatus: active

- The views:

docsisManagerView

subtree: 1.3.6.1 (Entire MIB)

docsisOperatorWriteView

subtree: docsDevBase
subtree: docsDevSoftware
subtree: docsDevEvControl
subtree: docsDevEvThrottleAdminStatus

docsisMonitorView

subtree: 1.3.6.1.2.1.1 (system)
subtree: docsIfBaseObjects
subtree: docsIfCmObjects

docsisUserView

subtree 1.3.6.1.2.1.1 (system)
subtree: docsDevBase
subtree: docsDevSwOperStatus
subtree: docsDevSwCurrentVersion
subtree docsDevServerConfigFile
subtree: docsDevEventTable
subtree: docsDevCpeTable
subtree: docsIfUpstreamChannelTable
subtree: docsIfDownstreamChannelTable
subtree: docsIfSignalQualityTable
subtree: docsIfCmStatusTable

DOCSIS 2.0-compliant CMs MUST also support additional VACM users as they are configured via an SNMP-embedded configuration file.

This page intentionally left blank.

6 Management Information Bases (MIBs)

This section defines the minimum set of managed objects required to support the management of CM and CMTS. Vendors MAY augment this MIB with objects from other standard or vendor-specific MIBs where appropriate.

The DOCSIS OSSI 2.0 specification has priority over the IETF MIBs and all objects. Though deprecated or optional in the IETF MIB, the object can be required by this specification as mandatory. Regardless of having either a status of deprecated or optional in the IETF MIB, the CM and CMTS MUST implement MIB requirements in accordance with the OSSI 2.0 specification.

If not required by this specification, deprecated objects are optional. If a CM or CMTS implements a deprecated object, the object MUST be implemented correctly according to the MIB definition. If a CM or CMTS does not implement a deprecated object, the agent MUST NOT instantiate the object and when accessed, MUST respond with the appropriate error/exception condition, such as no such object for SNMPv2c.

If not required by this specification, optional objects are optional. If a CM or CMTS implements an optional object, the object MUST be implemented correctly according to the MIB definition. If a CM or CMTS does not implement an optional object, the agent MUST NOT instantiate the object and when accessed, MUST respond with the appropriate error/exception condition, such as no such object for SNMPv2c.

If not required by this specification, obsolete objects are optional. If a CM or CMTS implements an obsolete object, the object MUST be implemented correctly according to the MIB definition. If a CM or CMTS does not implement an obsolete object, the agent MUST NOT instantiate the object and when accessed, MUST respond with the appropriate error/exception condition, such as no such object for SNMPv2c.

Sections 6.1 and 6.2 include an overview of the MIB modules required for management of the facilities specified in the DOCSIS RFI 2.0 and BPI+ specifications.

6.1 IPCDN drafts and others[1]

Table 6-1 IPCDN Drafts

MIB	Applicable Device(s)
IETF Proposed Standard RFC version of Qos MIB, “draft-ietf-ipcdn-qos-mib-04.txt”	CM and CMTS
IETF Proposed Standard RFC version of BPI+ MIB, “draft-ietf-ipcdn-bpiplus-mib-05.txt”	CM and CMTS
IETF Proposed Standard RFC version of USB MIB, “draft-ietf-xxxx-xxxx-xxxx-00.txt”	CM only
IETF Proposed Standard RFC version of Subscriber Management MIB, “draft-ietf-ipcdn-subscriber-mib-02.txt”	CMTS only
RFI-MIB-IPCDN-DRAFT, “draft-ietf-ipcdn-docs-rfmibv2-04.txt” supersedes RFC 2670 for DOCSIS 2.0.	CM and CMTS

6.2 IETF RFCs

Table 6-2 IETF RFCs

MIB	Applicable Device(s)
RFC 2933: Internet Group Management Protocol MIB	CM and CMTS
RFC 2669: DOCSIS Cable Device MIB	CM and CMTS
RFC 2665: Ethernet Interface MIB.	CM and CMTS
RFC 2233: The Interfaces Group MIB using SMIv2	CM and CMTS
RFC 1493: Bridge MIB	CM and CMTS
RFC 2011: SNMPv2 Management Information Base for the Internet Protocol using SMIv2	CM and CMTS
RFC 2013: SNMPv2 Management Information Base for the User Datagram Protocol using SMIv2	CM and CMTS
RFC 1907: Management Information Base for Version 2 of the Simple Network Management Protocol (SNMPv2)	CM and CMTS
RFC 2786: Diffie-Helman USM Key	CM and CMTS
RFC 3083: Baseline Privacy Interface MIB	CM
RFC 2863: The Interfaces Group MIB - supersedes RFC 2233 for DOCSIS 2.0	CM and CMTS

6.3 Managed objects requirements

The following sections detail any additional implementation requirements for the RFCs listed. Refer to Annex A for specific object implementation requirements.

The CM and CMTS MUST support a minimum of 10 available SNMP table rows unless otherwise specified by RFC or DOCSIS specification. The CM/CMTS minimum number of available SNMP table rows SHOULD mean rows (per table) that are available to support device configuration. CM/CMTS used (default) SNMP table row entries MUST NOT apply to the minimum number of available SNMP table rows.

1. table updated per ossi2-n-02016, 05/14/02, ab
also per oss2-n-02107, 06/04/02, ab

6.3.1 CMTS MIB requirements

DOCSIS 2.0-compliant CMTSes MUST implement the Subscriber Management MIB.

6.3.2 Requirements for RFC 2669

RFC 2669 MUST be implemented by DOCSIS 2.0-compliant CMs. DOCSIS 2.0-compliant CMTSes MUST implement the mandatory required objects (as specified by Annex A), and SHOULD implement the other non-mandatory required objects.

6.3.3 Requirements for RFI-MIB-IPCDN-DRAFT

RFI-MIB-IPCDN-DRAFT MUST be implemented by DOCSIS 2.0-compliant CMTSes and CMs. It replaces RFC 2670 for DOCSIS 2.0.

The docsIfDownChannelPower object-type MUST be implemented in a CMTS that provides an integrated RF upconverter. If the CMTS relies on an external upconverter, then the CMTS SHOULD implement the docsIfDownChannelPower object-type. The CMTS transmit power reported in the MIB object MUST be within 2 dB of the actual transmit power in dBmV when implemented. If transmit power management is not implemented, the MIB object will be read-only and report the value of 0 (zero).

The docsIfDownChannelPower object-type MUST be implemented in DOCSIS 2.0-conforming CMs. This object is read-only. When operated at nominal line voltage, at normal room temperature, the reported power MUST be within 3 dB of the actual received channel power. Across the input power range from -15 dBmV to +15 dBmV, for any 1 dB change in input power, the CM MUST report a power change in the same direction that is not less than 0.5 dB and not more than 1.5 dB.

The access of docsIfDownChannelFrequency object MUST be implemented as RW if a CMTS is in control of the downstream frequency. But if a CMTS provides IF output, docsIfDownChannelFrequency MUST be implemented as read-only and return 0.

6.3.4 Requirements for RFC 2863

RFC 2863 MUST be implemented by DOCSIS 2.0-compliant CMTSes and CMs.

The CMTS/CM ifAdminStatus object MUST provide administrative control over both MAC interfaces and individual channel and MUST be implemented as RW.

The ifType object has been assigned the following enumerated values for each instance of a Data Over Cable Service (DOCS) interface:

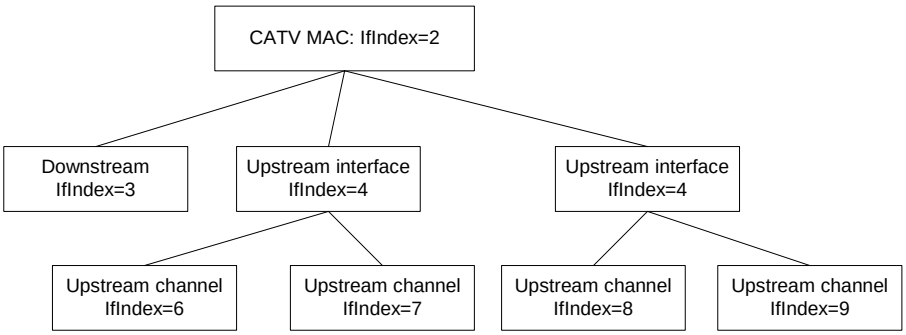
CATV MAC interface: docsCableMacLayer (127)
CATV downstream channel: docsCableDownstream (128)
CATV upstream interface: docsCableUpStream (129)
CATV upstream logical channel: docsCableUpstreamChannel (205)

6.3.4.1 Interface organization and numbering

Assigned interface numbers for CATV-MAC and Ethernet (Ethernet-like interface) are used in both the NMAccessTable and IP/LLC filtering table to configure access and traffic policy at these interfaces. These configurations are generally encoded in the configuration file using TLV encoding. To avoid provisioning complexity the interface-numbering scheme MUST comply with the following requirements:

- A CM supports only one upstream interface. At the CM, an instance of IfEntry MUST exist for each CATV-MAC interface, downstream channel, upstream interface, and each LAN interface enabled. The enabling of LAN interfaces MAY be fixed a priori during the manufacturing process or MAY be determined dynamically during operation by the CM according to whether or not an interface has a CPE device attached to it.
- If the CM has multiple CPE interfaces but only one CPE interface can be enabled at any given time, the ifTable MUST contain only the entry corresponding to the enabled or the default CPE interface. If a MAC interface consists of more than one upstream and downstream channel, a separate instance of ifEntry MUST exist for each channel.
- A 2.0 CMTS supports more than one one upstream logical channel per upstream interface. At the CMTS, an instance of IfEntry MUST exist for each CATV-MAC interface, downstream channel, upstream interface, upstream logical channel, and any other interface enabled.
- For CM/CMTS, the ifStack group ([RFC 2863]) must be implemented to identify relationships among sub-interfaces. Note that the CATV-MAC interface MUST exist, even though it is broken out into sub-interfaces.

The following example illustrates a MAC interface with one downstream, two upstream interfaces each with two upstream logical channels for a CMTS.

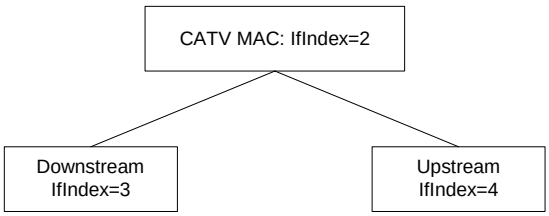


Implementation of ifStackTable for this example:

ifStackHigherLayer	ifStackLowerLayer
0	2
2	3
2	4
2	5
3	0
4	9
4	7
5	9
5	9
6	0
7	0
8	0
9	0

Figure 6-1 ifIndex example for CMTS

The following example illustrates a MAC interface with one downstream and one upstream interface for a CM.



Implementation of ifStackTable for this example:

ifStackHigherLayer	ifStackLowerLayer
0	2
2	3
2	4
3	0
4	0

Figure 6-2 ifIndex example for CM

At the CMTS, interface number is at the discretion of the vendor, and SHOULD correspond to the physical arrangement of connections. If table entries exist separately for upstream and downstream channels, then the ifStack group ([RFC 2863]) MUST be implemented to identify the relationship among sub-interfaces. Note that the CATV MAC interface(s) MUST exist, even if further broken out into sub-interfaces.

At the CM, interfaces MUST be numbered as follows:

Table 6-3 CM interface numbering

Interface	Type
1	Primary CPE interface
2	CATV-MAC
3	RF-down
4	RF-up
4+n	Other interfaces

If the CM has more than one CPE interface, the vendor MUST define which of the n CPE interfaces is the primary CPE interface. The definition of the primary CPE interface MAY be fixed a priori during manufacturing process or MAY be determined dynamically during operation by the CM according to which interface has a CPE device attached to it. Regardless of the number of CPE interfaces the CM has, or how the primary CPE interface is defined, the primary interface MUST be interface number 1.

The definition of the secondary CPE interface MAY be fixed a priori during manufacturing process or MAY be determined dynamically during operation by the CM according to which interface has a CPE device attached to it. The secondary CPE, and other interfaces, will start at 5.

DOCSIS CMs may have multiple interfaces. If filter(s) (Ip, LLC, or NmAccess) are applied to the CM's IfIndex 1, the same filter(s) MUST also be applied to each CPE interface; however, filters are never used to limit traffic between multiple CPE interfaces within the CM.

6.3.4.2 docsIfCmStatusValue and ifOperStatus Relationship

For the CM's RF downstream, RF upstream (upstream interface and logical channel) and RF MAC interfaces, the following are the expected relationship of ifOperStatus and docsIfCmStatusValue when ifAdminStatus = up (taken from RFI-MIB-IPCDN-DRAFT).

Table 6-4 docsIfCmStatusValue and ifOperStatus relationship

IfOperStatus	docsIfCmStatusValue
down(2):	other(1), notReady(2)
dormant(5):	notSynchronized(3), phySynchronized(4), usParametersAcquired(5), rangingComplete(6), ipComplet(7), todEstablished(8), paramTransferComplete(10),
up(1):	registrationComplete(11), securityEstablished(9), operational(12), accessDenied(13)

6.3.4.2.1 ifAdminStatus and traffic

If the CM and CMTS interface's ifAdminStatus = down, the interface MUST NOT accept or forward any traffic (traffic includes data and MAC management traffic).

6.3.5 Interface MIB and Trap Enable

The Interface MIB and Trap Enable specified in RFC 2863 MUST be implemented by DOCSIS 2.0-compliant CMTSes and CMs.

If a multi-layer interface model is present in the device, each sub-layer for which there is an entry in the ifTable can generate linkUp/Down traps. Since interface state changes would tend to propagate through the interface stack (from top to bottom, or bottom to top), it is likely that several traps would be generated for each linkUp/Down occurrence. The CM and CMTS MUST implement the ifLinkUpDownTrapEnable object to allow managers to control trap generation, and configure only the interface sub-layers of interest.

The default setting of ifLinkUpDownTrapEnable MUST limit the number of traps generated to one, per interface, per linkUp/Down event. Interface state changes, of most interest to network managers, occur at the lowest level of an interface stack.

On CM linkUp/Down event a trap SHOULD be generated by the CM MAC interface and not by any sub-layers of the interface. Therefore, the default setting of ifLinkUpDownTrapEnable for CM MAC MUST be set to enable, and the default setting of ifLinkUpDownTrapEnable for CM RF-Up MUST be set to disable, and the default setting of ifLinkUpDownTrapEnable for CM RF-Down MUST be set to disable.

On CMTS interfaces (MAC, RF-Downstream(s), RF-Upstream(s)) the linkUp/Down event/trap SHOULD be generated by each CMTS interface. Therefore, the default setting of ifLinkUpDownTrapEnable for each CMTS interface (MAC, RF-Downstream(s), RF-Upstream(s)) MUST be set to enable.

6.3.6 Requirements for RFC 2665

RFC 2665 MUST be implemented by DOCSIS 2.0-compliant CMTSes and CMs if Ethernet or Fast Ethernet interfaces are present.

6.3.7 Requirements for RFC 1493

RFC 1493 MUST be implemented by DOCSIS 2.0-compliant CMTS and CMs.

In both the CM and the CMTS (if the CMTS implements transparent bridging), the Bridge MIB ([RFC 1493]) MUST be implemented to manage the bridging process.

In CMTSes that implement transparent bridging, the Bridge MIB MUST be used to represent information about the MAC Forwarder states.

6.3.8 Requirements for RFC 2011

RFC 2011 MUST be implemented by DOCSIS 2.0-compliant CMTSes and CMs.

6.3.8.1 The IP Group

The IP group MUST be implemented. It does not apply to IP packets forwarded by the device as a link-layer bridge. For the CM, it applies only to the device as an IP host. At the CMTS, it applies to the device as an IP host, and as a routers if IP routing is implemented.

6.3.8.2 The ICMP Group

The ICMP group MUST be implemented.

6.3.9 Requirements for RFC 2013

RFC 2013 MUST be implemented by DOCSIS 2.0-compliant CMTSes and CMs.

The UDP group in RFC 2013 MUST be implemented.

6.3.10 Requirements for RFC 1907

RFC 1907 MUST be implemented by DOCSIS 2.0-compliant CMTSes and CMs.

6.3.10.1 The System Group

The System Group from RFC 1907 MUST be implemented. See Section 7.2.1 for sysObjectID requirements.

6.3.10.2 The SNMP Group

The SNMP Group from RFC 1907 MUST be implemented.

6.3.11 Requirements for “draft-ietf-ipcdn-qos-mib-04.txt”

“draft-ietf-ipcdn-qos-mib-04.txt” MUST be implemented by DOCSIS 2.0-compliant CMTSes and CMs.

6.3.12 Requirements for “draft-ietf-ipcdn-igmp-mib-01.txt”

“draft-ietf-ipcdn-igmp-mib-01.txt” requirements have been deleted for CMTSes and CMs.

6.3.13 Requirements for RFC 2933

RFC 2933 MUST be implemented by DOCSIS 2.0-compliant CMTSes and CMs.

Refer to Annex E for DOCSIS 2.0 IGMP cable device implementation details.

6.3.14 Requirements for BPI+ MIB

"draft-ietf-ipcdn-bpiplus-mib-05.txt" MUST be implemented by DOCSIS 2.0-compliant CMTSes and CMs as specified in Annex A.

6.3.15 Requirements for "draft-ietf-xxxx-xxxx-xxxx-00.txt" USB MIB

Note: Until the USB MIB becomes an IETF RFC, the draft text will be available on the DOCSIS website.

6.3.16 Requirements for Subscriber Management MIB

"draft-ietf-ipcdn-subscriber-mib-02-.txt" MUST be implemented by DOCSIS 2.0-compliant CMTSes.

DOCSIS 2.0-compliant CMTSes MUST support a minimum number of thirty (30) filter groups of twenty (20) filters each.

6.3.17 Requirements for RFC 2786 Diffie-Helman USM Key

RFC 2786 MUST be implemented by DOCSIS 2.0-compliant CMs. RFC 2786 MAY be implemented on the CMTS.

6.3.18 Requirements for RFC 3083 Baseline Privacy Interface MIB

RFC 3083 MUST be implemented by DOCSIS 2.0-compliant CMs as specified in Annex A.

Due to the editorial error in RFC 3083, DOCSIS 2.0-compliant CMs MUST use the following definition for docsBpiCmAuthState and not the definition in RFC 3083:

```
docsBpiCmAuthState  OBJECT-TYPE
SYNTAX  INTEGER {
           start(1),
           authWait(2),
           authorized(3),
           reauthWait(4),
           authRejectWait(5)
       }

MAX-ACCESS  read-only

STATUS  current

DESCRIPTION
"The value of this object is the state of the CM authorization FSM. The start state indicates
that FSM is in its initial state."

REFERENCE
"DOCSIS Baseline Privacy Interface Specification, Section 4.1.2.1."
::= { docsBpiCmBaseEntry 3 }
```

6.3.19 Requirements for DOCS-IF-EXT-MIB

A Docsis 2.0-compliant CM/CMTS MUST NOT support the DOCS-IF-EXT MIB, which is defined in Annex G.[1]

6.3.20 Requirements for DOCS-CABLE-DEVICE-TRAP-MIB

DOCSIS 2.0-compliant CMs and CMTSes MUST implement DOCS-CABLE-DEVICE-TRAP-MIB, as specified in Annex H.

6.3.21 Requirements for SNMPv3 MIBs

DOCSIS 1.1-compliant CMs/CMTSes MUST implement the MIBs defined in [RFC 2571] through [RFC 2576].

For CMs, the default value for any SNMPv3 object with a storageType textual convention MUST be 'volatile(2)'. This overides the default value specified in [RFC 2573] through [RFC 2576]. The CM MUST only accept the value of 'volatile(2)' on any SNMPv3 storageType object. An attempted set to a value of other(1), nonVolatile(3), permanent(4), or readOnly(5) will result in an 'inconsistantValue' error. Values other than the valid range (1-5) would result in a 'wrongValue' error.

6.4 CM configuration files, TLV-11 and MIB OIDs/values

The following sections define the use of CM configuration file TLV-11 elements and the CM rules for translating TLV-11 elements into SNMP PDU (SNMP MIB OID/instance and MIB OID/instance value combinations; also referred to as SNMP varbinds).

This section also defines the CM behaviors, or state transitions, after either pass or fail of the CM configuration process.

For TLV-11 definitions refer to Annex C of [DOCSIS 5].

6.4.1 CM configuration file TLV-11 element translation (to SNMP PDU)

TLV-11 translation defines the process used by the CM to convert CM configuration file information (TLV-11 elements) into SNMP PDU (varbinds). The CM MUST translate CM configuration file TLV-11 elements into a single SNMP PDU containing (n) MIB OID/instance and value components (SNMP varbinds). Once a single SNMP PDU is constructed, the CM processes the SNMP PDU and determines the CM configuration pass/fail based on the rules for CM configuration file processing, described below. However, if a CM is not physically capable of processing a potentially large single CM configuration file-generated SNMP PDU, the CM MUST still behave as if all MIB OID/instance and value components (SNMP varbinds) from CM configuration file TLV-11 elements are processed as a single SNMP PDU.

In accordance with [RFC 1905], the single CM configuration file-generated SNMP PDU will be treated "as if simultaneous" and the CM must behave consistently, regardless of the order in which TLV-11 elements appear in the CM configuration file or SNMP PDU. The single CM configuration file-generated SNMP PDU requirement is consistent with SNMP PDU packet behaviors received from an SNMP manager; SNMP PDU varbind order does not matter, and there is no defined MAX SNMP PDU limit.

1. changed per rfi-n-02016, 05/14/02, ab

The CM configuration file MUST NOT contain duplicate TLV-11 elements (duplicate means SNMP MIB object has either identical OID or OID from the old and new MIB that actually point to the same SNMP MIB object). If duplicate TLV-11 elements are received by the CM, from the CM configuration file, then the CM MUST fail CM configuration.

6.4.1.1 Rules for CreateAndGo and CreateAndWait

The CM MUST support CreateAndGo for row creation.

The CM MAY support CreateAndWait, with the constraint that CM configuration file TLV-11 elements MUST NOT be duplicated (all SNMP MIB OID/instance must be unique). For instance, an SNMP PDU constructed from CM configuration file TLV-11 elements, which contains an SNMP CreateAndWait value for a given SNMP MIB OID/instance, MUST NOT also contain an SNMP Active value for the same SNMP MIB OID/instance (and vice versa). A CM configuration file MAY contain a TLV-11 CreateAndWait element if the intended result is to create an SNMP table row which will remain in the SNMP NotReady or SNMP NotInService state until a non-configuration file SNMP PDU is issued, from an SNMP manager, to update the SNMP table row status.

Both SNMP NotReady and SNMP NotInService states are valid table row states after an SNMP CreateAndWait instruction.

6.4.2 CM configuration TLV-11 elements not supported by the CM

If any CM configuration file TLV-11 elements translate to SNMP MIB OIDs that are not MIB OID elements supported by the CM, then those SNMP varbinds MUST be ignored, and treated as if they had not been present, for the purpose of CM configuration. This means that the CM will ignore SNMP MIB OIDs for other vendors' private MIBs as well as standard MIB elements that the CM does not support.

CMs that do not support SNMP CreateAndWait for a given SNMP MIB table MUST ignore, and treat as if not present, the set of columns associated with the SNMP table row.

If any CM configuration file TLV-11 element(s) are ignored, then the CM MUST report via the CM configured notification mechanism(s), after the CM is registered. The CM notification method MUST be in accordance with Section 7.4.2.3.

6.4.3 CM state after CM configuration file processing success

After successful CM configuration, via CM configuration file, the CM MUST proceed to register with the CMTS and pass data.

6.4.4 CM state after CM configuration file processing failure

If any CM configuration file generated SNMP PDU varbind performs an illegal set operation (illegal, bad, or inconsistent value) to any MIB OID/instance supported by the CM, processing of the CM configuration file MUST fail. Any CM configuration file generated SNMP PDU varbind set failure MUST cause a CM configuration failure, and the CM MUST NOT proceed with CM registration.

6.5 Treatment and interpretation of MIB counters on the CM

Octet and packet counters implemented as counter32 and counter64 MIB objects are monotonically increasing positive integers with no specific initial value and a maximum value based on the counter size that will roll-over to zero when it is exceeded. In particular, counters are defined such that the only meaningful value is the

difference between counter values as seen over a sequence of counter polls. However, there are two situations that can cause this consistent monotonically increasing behavior to change: 1) resetting the counter due to a system or interface reinitialization, or 2) a rollover of the counter when it reaches its maximum value of 2^{32}-1 or 2^{64}-1. In these situations, it must be clear what the expected behavior of the counters should be.

Case 1: Whenever the state of an interface changes resulting in an "interface counter discontinuity" as defined in RFC 2863. In this case the value of the ifXTable.ifXEntry.ifCounterDiscontinuityTime for the affected interface MUST be set to the current value of sysUpTime and ALL counters for the affected interface MUST be set to ZERO. Setting the ifAdminStatus of specified interface to down(2) MUST NOT be considered as an interface reset.

Case 2: SNMP Agent Reset. In this case, the value of the sysUpTime MUST be set to ZERO, all interface ifCounterDiscontinuityTime values MUST be set to ZERO, and all interface counters MUST be set to ZERO. Also, all other counters being maintained by the SNMP Agent MUST be set to ZERO.

Case 3: Counter Rollover. When a counter32 object reaches its maximum value of 4,294,967,295, the next value MUST be ZERO. When a counter64 object reaches its maximum value of 18,446,744,073,709,551,615, the next value MUST be ZERO. Note that unless a CM or CMTS vendor provides a means outside of SNMP to preset a counter64 or counter32 object to an arbitrary value, it will not be possible to test any rollover scenarios for counter64 objects (and many counter32 objects as well). This is because it is not possible for these counters to rollover during the service life of the device (see discussion in Section 3.1.6 of RFC 2863).

6.6 SNMPv3 Notification Receiver config file element

This section specifies processing requirements on the CM when one or SNMPv3 Notification Receiver TLVs are present in the configuration file. The SNMPv3 Notification Receiver TLV is used to configure SNMPv3 tables for notification transmission. The CM MUST process this TLV only if the CM is in SNMPv3 Coexistence Mode.

Based on the TLV, the CM 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. The mapping from the TLV to these tables is described in the following section.

6.6.1 Mapping of TLV fields into created SNMPv3 table rows[1]

The following tables illustrate how the fields from the config file TLV elements are placed into the SNMPv3 tables. The TLV fields are shown below as:

<IP Address> A 32-bit IP address of a notification receiver

<Port> A 16-bit UDP Port number on the notification receiver to receive the notifications

<Trap type> Defines the notification type as explained above

<Timeout> 16-bit timeout, in milliseconds to wait before sending a retry of an Inform Notification

<Retries> 16-bit number of times to retry an Inform after the first Inform transmission

1. value of all "StorageType" and "StorType" objects changed to "volatile (2)" per oss2-n-02079, 06/04/02, ab.

<Filter OID> The OID of the snmpTrapOID value that is the root of the MIB subtree that defines all of the notifications to be sent to the Notification Receiver.

<Security Name> The security name specified on the TLV element, or "@config" if not specified.

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

In configuring entries in these SNMPv3 tables, note the following:

- The Community Name for traps in SNMPv1 and SNMPv2 packets is configured as "public". The Security Name in traps and informs in SNMPv3 packets where no security name has been specified is configured as "@Config", in which case the security level is "noAuthNoPriv".
- Several columnar objects are configured with a value beginning with the string "@config". If these tables are configured through other mechanisms, Network operators should not use values beginning with "@config" to avoid conflicts with the mapping process specified here.

6.6.1.1 snmpNotifyTable

The snmpNotifyTable is defined in RFC 2573, in the "Notification MIB Module" section.

Create 2 rows with fixed values if 1 or more TLV elements are present.

Table 6-5 snmpNotifyTable

Column Name (* = Part of Index)	1st Row Column Value	2nd Row Column Value
* snmpNotifyName	"@config_inform"	"@config_trap"
snmpNotifyTag	"@config_inform"	"@config_ trap "
snmpNotifyType	inform (2)	trap (1)
snmpNotifyStorageType	volatile (2)	volatile (2)
snmpNotifyRowStatus	Active (1)	active (1)

6.6.1.2 snmpTargetAddrTable

The snmpTargetAddrTable is defined in RFC 2573, in the "Management Target MIB Module" section.

Create 1 row for each TLV element in the config file.

Table 6-6 snmpTargetAddrTable

Column Name (* = Part of Index)	Column Value
* snmpTargetAddrName	"@config_n" Where n ranges from 0 to m-1 where m is the number of notification receiver TLV elements in the config file
snmpTargetAddrTDomain	snmpUDPDomain = snmpDomains.1
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

Table 6-6 snmpTargetAddrTable

Column Name (* = Part of Index)	Column Value
snmpTargetAddrTagList	"@config_trap" if <Trap type> is 1, 2, or 4 "@config_inform" if <Trap type> is 3 or 5
snmpTargetAddrParams	"@config_n" (Same as snmpTargetAddrName value)
snmpTargetAddrStorageType	volatile (2)
snmpTargetAddrRowStatus	active (1)

6.6.1.3 snmpTargetAddrExtTable

The snmpTargetAddrExtTable is defined in RFC 2576, in the "SNMP Community MIB Module" section.

Create 1 row for each TLV element in the config file.

Table 6-7 snmpTargetAddrExtTable

Column Name (* = Part of Index)	Column Value
* snmpTargetAddrName	"@config_n" where n ranges from 0 to m-1, and m is the number of notification receiver TLV elements in the config file
snmpTargetAddrTMask	<zero-length octet string>
snmpTargetAddrMMS	0

6.6.1.4 snmpTargetParamsTable

The snmpTargetParamsTable is defined in RFC 2573, in the "Management Target MIB Module" section.

Create 1 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 6-8 snmpTargetParamsTable for <Trap type> 1, 2, or 3

Column Name (* = Part of Index)	Column Value
* snmpTargetParamsName	"@config_n" where n ranges from 0 to m-1, and m is the number of notification receiver TLV elements in the config file
snmpTargetParamsMPModel SYNTAX: SnmpMessageProcessingModel	SNMPv1 (0) if <Trap type> is 1 SNMPv2c (1) if <Trap type> is 2 or 3 SNMPv3 (3) if <Trap type> is 4 or 5
snmpTargetParamsSecurityModel SYNTAX: SnmpSecurityModel	SNMPv1 (1) if <Trap type> is 1 SNMPv2c (2) If <Trap type> is 2 or 3 USM (3) if <Trap type> is 4 or 5 NOTE: The mapping of SNMP protocol types to value here are different from snmpTargetParamsMPModel
snmpTargetParamsSecurityName	"@config"
snmpTargetParamsSecurityLevel	noAuthNoPriv
snmpTargetParamsStorageType	volatile (2)
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 6-9 snmpTargetParamsTable for <Trap type> 4 or 5

Column Name (* = Part of Index)	Column Value
* snmpTargetParamsName	"@config_n" where n ranges from 0 to m-1, and m is the number of notification receiver TLV elements in the config file
snmpTargetParamsMPModel SYNTAX: SnmpMessageProcessingModel	SNMPv1 (0) if <Trap type> is 1 SNMPv2c (1) if <Trap type> is 2 or 3 SNMPv3 (3) if <Trap type> is 4 or 5
snmpTargetParamsSecurityModel SYNTAX: SnmpSecurityModel	SNMPv1 (1) if <Trap type> is 1 SNMPv2c (2) if <Trap type> is 2 or 3 USM (3) if <Trap type> is 4 or 5 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 (2)
snmpTargetParamsRowStatus	active (1)

6.6.1.5 snmpNotifyFilterProfileTable

The snmpNotifyFilterProfileTable is defined in RFC 2573, in the "Notification MIB Module" section.

Create 1 row for each TLV that has a non-zero <Filter Length>.

Table 6-10 snmpNotifyFilterProfileTable

Column Name (* = Part of Index)	Column Value
* snmpTargetParamsName	"@config_n" where n ranges from 0 to m-1, and m is the number of notification receiver TLV elements in the config file
snmpNotifyFilterProfileName	"@config_n" where n ranges from 0 to m-1, and m is the number of notification receiver TLV elements in the config file
snmpNotifyFilterProfileStorType	volatile (2)
snmpNotifyFilterProfileRowStatus	active (1)

6.6.1.6 snmpNotifyFilterTable

The snmpNotifyFilterTable is defined in RFC 2573, in the "Notification MIB Module" section.

Create 1 row for each TLV that has a non-zero <Filter Length>.

Table 6-11 snmpNotifyFilterTable

Column Name (* = Part of Index)	Column Value
* snmpNotifyFilterProfileName	"@config_n" where n ranges from 0 to m-1, and m is the number of notification receiver TLV elements in the config file
* snmpNotifyFilterSubtree	<Filter OID> from the TLV

Table 6-11 snmpNotifyFilterTable

Column Name (* = Part of Index)	Column Value
snmpNotifyFilterMask	<zero-length octet string>
snmpNotifyFilterType	included (1)
snmpNotifyFilterStorageType	volatile (2)
snmpNotifyFilterRowStatus	active (1)

6.6.1.7 snmpCommunityTable

The snmpCommunityTable is defined in RFC 2576, in the "SNMP Community MIB Module" section.

Create 1 row with fixed values if 1 or more TLVs are present. This causes SNMPv1 and v2c notifications to contain the community string in snmpCommunityName.

Table 6-12 snmpCommunityTable

Column Name (* = Part of Index)	Column Value
* snmpCommunityIndex	"@config"
snmpCommunityName	"public"
snmpCommunitySecurityName	"@config"
snmpCommunityContextEngineID	<the engineID of the cable modem>
snmpCommunityContextName	<zero-length octet string>
snmpCommunityTransportTag	<zero-length octet string>
snmpCommunityStorageType	volatile (2)
snmpCommunityStatus	active (1)

6.6.1.8 usmUserTable

The usmUserTable is defined in RFC 2574, in the "Definitions" section.

Create 1 row with fixed values if 1 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 which notifications are to be sent.

One row in the usmUserTable is created. 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 6-13 usmUserTable

Column Name (* = Part of Index)	Column Value
* usmUserEngineID	0x00
* usmUserName	"@config" When other rows are created, this is replaced with the <Security Name> field from the TLV element.
usmUserSecurityName	"@config" When other rows are created, this is replaced with the <Security Name> field from the TLV element.

Table 6-13 usmUserTable (Continued)

Column Name (* = Part of Index)	Column Value
usmUserCloneFrom	<don't care> This row cannot be cloned.
usmUserAuthProtocol	None When other rows are created, this is replaced with None or MD5, depending on 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>
usmUserStorageType	volatile (2)
usmUserStatus	active (1)

6.6.1.9 vacmSecurityToGroupTable

The vacmSecurityToGroupTable is defined in RFC 2575, in the "Definitions" section.

Create 3 rows with fixed values if 1 or more TLVs are present.

These are the 3 rows with fixed values. These are used for the TLV entries with <Trap Type> set to 1, 2, or 3, or with a zero-length <Security Name>. The TLV entries with <Trap Type> set to 4 or 5 and a non-zero length <Security Name> will use the rows created in the vacmSecurityToGroupTable by the DH Kickstart process.

Table 6-14 vacmSecurityToGroupTable

Column Name (* = Part of Index)	First Row Column Value	Second Row Column Value	Third Row Column Value
* vacmSecurityModel	SNMPV1 (1)	SNMPV2c (2)	USM (3)
* vacmSecurityName	"@config"	"@config"	"@config"
vacmGroupName	"@configV1"	"@configV2"	"@configUSM"
vacmSecurityToGroupStorageType	volatile (2)	volatile (2)	volatile (2)
vacmSecurityToGroupStatus	active (1)	active (1)	active (1)

6.6.1.10 vacmAccessTable

The vacmAccessTable is defined in RFC 2575, in the "Definitions" section.

Create 3 rows with fixed values, if 1 or more TLVs are present.

These are the 3 rows with fixed values. These are used for the TLV entries with <Trap Type> set to 1, 2, or 3, or with a zero-length <Security Name>. The TLV entries with <Trap Type> set to 4 or 5 and a non-zero length <Security Name> will use the rows created in the vacmAccessTable by the DH Kickstart process.

Table 6-15 vacmAccessTable

Column Name (* = Part of Index)	Column Value	Column Value	Column Value
* vacmGroupName	"@configV1"	"@configV2"	"@configUSM"
* vacmAccessContextPrefix	<zero-length string>	<zero-length string>	<zero-length string>
* vacmAccessSecurityModel	SNMPV1 (1)	SNMPV2c (2)	USM (3)
* vacmAccessSecurityLevel	noAuthNoPriv (1)	noAuthNoPriv (1)	noAuthNoPriv (1)
vacmAccessContextMatch	exact (1)	exact (1)	exact (1)
vacmAccessReadViewName	<zero-length octet string>	<zero-length octet string>	<zero-length octet string>
vacmAccessWriteViewName	<Zero length octet string>	<Zero length octet string>	<Zero length octet string>
vacmAccessNotifyViewName	"@config"	"@config"	"@config"
vacmAccessStorageType	volatile (2)	volatile (2)	volatile (2)
vacmAccessStatus	active (1)	active (1)	active (1)

6.6.1.11 vacmViewTreeFamilyTable

The vacmViewTreeFamilyTable is defined in RFC 2575, in the "a" section.

Create 1 row with fixed values if 1 or more TLVs are present.

This row is used for the TLV entries with <Trap Type> set to 1, 2, or 3 or with a zero-length <Security Name>. The TLV entries with <Trap Type> set to 4 or 5 and a non-zero length <Security Name> will use the rows created in the vacmViewTreeFamilyTable by the DH Kickstart process.

Table 6-16 vacmViewTreeFamilyTable

Column Name (* = Part of Index)	Column Value
* vacmViewTreeFamilyViewName	"@config"
* vacmViewTreeFamilySubtree	1.3
vacmViewTreeFamilyMask	<default from MIB>
vacmViewTreeFamilyType	included (1)
vacmViewTreeFamilyStorageType	volatile (2)
vacmViewTreeFamilyStatus	active (1)

This page intentionally left blank.

7 OSSI for Radio Frequency Interface

7.1 Subscriber Account Management Interface specification

Note: The Subscriber Account Management Interface specification is OPTIONAL for CMTS vendors at this time. However, if a billing interface is provided by a CMTS vendor, it MUST conform to the specification in this section.

The Subscriber Account Management Interface Specification is defined to enable prospective vendors of cable modems and cable modem termination systems to address the operational requirements of subscriber account management in a uniform and consistent manner. It is the intention that this will enable operators and other interested parties to define, design, and develop Operations and Business Support Systems (OBSS) necessary for the commercial deployment of different classes of services over cable networks with accompanying usage-based billing of services for each individual subscriber.

The Subscriber Account Management described herein refers to the following business processes and terms:

- Class of Service Provisioning Processes, which are involved in the automatic and dynamic provisioning and enforcement of subscribed class of policy-based service level agreements (SLAs)
- Usage-Based Billing Processes, which are involved in the processing of bills based on services rendered to and consumed by paying subscribers. This Specification focuses primarily on bandwidth-centric usage-based billing scenarios. It complements the Telephony Billing Specification that is being developed within the PacketCable architecture.

In order to develop the DOCSIS-OSS Subscriber Account Management Specification, it is necessary to consider high-level business processes common to cable operators and the associated operational scenarios. These issues are discussed in Appendix I.

7.1.1 Service Flows, Service Classes, and Subscriber Usage Billing

The DOCSIS 2.0 RFI specification provides a mechanism for a Cable Modem (CM) to register with its Cable Modem Termination System (CMTS) and configure itself based on external Quality of Service (QoS) parameters when it is powered up or reset. To quote (in part) from Section 10.1, “Theory of Operation”:

> The principal mechanism for providing enhanced QoS is to classify packets traversing the RF MAC interface into a Service Flow. A Service Flow is a unidirectional flow of packets that is provided a particular Quality of Service. The CM and the CMTS provide this QoS by shaping, policing, and prioritizing traffic according to the QoS Parameter Set defined for the Service Flow.
>
> The requirements for Quality of Service include:
>
> - A configuration and registration function for pre-configuring CM-based QoS Service Flows and traffic parameters.
> […]
> - Utilization of QoS traffic parameters for downstream Service Flows.
> - Classification of packets arriving from the upper layer service interface to a specific active Service Flow.
> - Grouping of Service Flow properties into named Service Classes, so upper layer entities and external applications (at both the CM and the CMTS) can request Service Flows with desired QoS parameters in a globally consistent way.

A Service Class Name (SCN) is defined in the CMTS via provisioning (see DOCS-QOS-MIB). An SCN provides a handle to an associated QoS Parameter Set (QPS) template. Service Flows created using an SCN are considered to be “named” Service Flows. The SCN identifies the service characteristics of a Service Flow to

external systems such as a billing system or customer service system. SCNs MUST be unique within an MSO's operation, and a descriptive SCN might be something like PrimaryUp, GoldTCPUp, VoiceDown, or BronzeUDPDown to indicate the nature and direction of the Service Flow to the external system.

A Service Package implements a Service Level Agreement (SLA) between the MSO and its Subscribers on the RFI interface. A Service Package might be known by a name such as Gold, Silver, or Bronze. A Service Package is itself implemented by the set of named Service Flows (using SCNs) that are placed into a CM Configuration File[1] that is stored on a TFTP server. The set of Service Flows defined in the CM Config File are used to create active Service Flows when the CM registers with the CMTS. Note that many Subscribers are assigned to the same Service Package, therefore, many CMs use the same CM Config File to establish their active Service Flows. Also, note that a Service Package MUST define at least two Service Flows known as Primary Service Flows that are used by default when a packet matches none of the classifiers for the other Service Flows. A CM Config File that implements a Service Package, therefore, MUST define the two primary Service Flows using SCNs (e.g., PrimaryUp and PrimaryDown) that are known to the CMTS if these Service Flows are to be visible via the billing interface to external systems.

The DOCSIS 2.0 RFI specification also provides for dynamically-created Service Flows. An example could be a set of dynamic Service Flows created by an embedded PacketCable Multimedia Terminal Adapter (MTA) to manage VoIP signaling and media flows. All dynamic Service Flows MUST be created using an SCN known to the CMTS if they are to be visible to the billing system. These dynamic SCNs do not need to appear in the CM Config File but the MTA may refer to them directly during its own initialization and operation.

During initialization, a CM communicates with a DHCP Server that provides the CM with its assigned IP address and, in addition, provides a pointer to the TFTP Server that stores the assigned CM Config File for that CM. The CM reads the CM Config File and forwards the set of Service Flow definitions (using SCNs) up to the CMTS. The CMTS then performs a macro-expansion on the SCNs (using the provisioned SCN templates) into QoS Parameter Sets needed to establish active Service Flows for the CM. Internally, each active Service Flow is identified by a 32-bit SFID assigned by the CMTS to a specific CM. For billing purposes, however, the SFID is not sufficient as the only identifier of a Service Flow because the billing system cannot distinguish the service being delivered by one SFID from another. Therefore, the SCN is necessary, in addition to the SFID, to identify the Service Flow's class of service characteristics to the billing system. The billing system can then rate the charges differently for each of the Service Flow traffic counts based on its Service Class (e.g., Gold octet counts are likely to be charged more than Bronze octet counts). Thus, the billing system obtains from the CMTS the traffic counts for each named Service Flow (identified by SFID and SCN) that a subscriber's CM uses during the billing data collection interval. This is true even if multiple active Service Flows (i.e., SFIDs) are created using the same SCN for a given CM over time. This will result in multiple billing records for the CM for Service Flows that have the same SCN (but different SFIDs). Note that the SFID is the primary key to the Service Flow. When an active Service Flow exists across multiple sequential billing files the SFID allows the sequence of recorded counter values to be correlated to the same Service Flow.

7.1.2 Requirements for Subscriber Usage Billing Records

The CMTS, or its supporting Element Management System (EMS), MUST provide formatted Subscriber Usage Billing Records for all subscribers attached to the CMTS on demand to a mediation system or a billing system. It is expected that the billing record collection interval could be as short as 10 minutes. The following are the requirements for processing and transmitting Subscriber Usage Billing Records:

1. The CM Configuration File contains several kinds of information needed to properly configure the CM and its relationship with the CMTS, but for the sake of this discussion only the Service Flow and Quality of Service components are of interest.

1. The Subscriber Usage Billing File MUST identify the CMTS by host name and IP address and the time that the billing file was created. The sysUpTime value for the CMTS MUST also be recorded.
2. Subscriber billing records MUST be organized by CM MAC address (but not necessarily sorted). The Subscriber's current CM IP address and all current CPE IP addresses MUST be present in the billing record for the Subscriber.
3. Subscriber billing records MUST have entries for each active Service Flow (identified by SFID and Service Class Name) used by the CM during the collection interval. This includes all currently active Service Flows as well as all active Service Flows that were deleted and logged during the collection interval. Note well that a provisioned or admitted state SF that was deleted before it became active is not recorded in the billing file, even though it was logged by the CMTS. It MUST be possible to distinguish continuing Service Flows from deleted Service Flows in the billing records.
4. It MUST be possible to identify the Service Flow direction as upstream or downstream without reference to the SCN. The number of packets and octets passed MUST be collected for each upstream and downstream Service Flow. The number of packets dropped and the number of packets delayed due to enforcement of QoS maximum throughput parameters MUST also be collected for each downstream Service Flow. Note that since it is possible for a Subscriber to change from one service package to another and back again or to have dynamic service flows occur multiple times, it is possible that there will be multiple entries for a given SCN within a Subscriber's billing record for the collection period. This could also occur if a CM ranges and re-registers for any reason (such as CPE power failure).
5. All traffic counters MUST be based on absolute 64-bit counters as maintained by the CMTS. These counters MUST be reset to zero by the CMTS if it reinitializes its management interface. The CMTS sysUpTime value is used to determine if the management interface has been reset between adjacent collection intervals.
6. To facilitate processing of the Subscriber Usage Billing Records by a large number of diverse billing and mediation systems an Extensible Markup Language (XML) format is required. Specifically, the IP Detail Record (IPDR) standard as described in IPDR.org's Network Data Management -Usage, Version 3.1 ([NDM-U 3.1]) as extended for XML Schema format Cable Data Systems Subscriber Usage Billing Records MUST be used. See Annex B for the Cable Data Systems Subscriber Usage Billing Records Service Specification submission to IPDR.org and an example IPDR XML Schema billing file. Refer to http://www.ipdr.org/ for more information on the NDM-U specification and Service Specification Guidelines.
7. To improve the performance of storage and transmission of the NDM-U XML format billing records a compressed file format is required. NDM-U 3.1 describes a compact encoding of IPDRDocs, utilizing the IETF XDR specification language. Optionally, subscriber usage billing records contained in IPDRDocs may be encoded in this format to gain efficiency. Furthermore, lossless compression in GZIP 4.3 format as described in [RFC 1952] MUST be used to store and transmit the billing file. It is expected that an IPDR XML format billing file will compress on the order of 15:1 or better. See also http://www.gnu.org/software/gzip/ for more information.
8. To improve the network performance of the billing collection activity, a reliable high-throughput TCP stream MUST be used to transfer billing records between the record formatter and the collection system. Standard FTP GET of the compressed (and optionally encrypted) billing file from the record formatter by the collection system MUST be supported.
9. To allow for decoupled scheduling, the billing collection cycle MUST be driven by the collection system through the standard FTP GET and FTP DELETE operations. Since the collection interval may vary over time, the record formatter is only required to maintain one current billing file in its FTP file system. The collection system (operating on its own schedule) may retrieve the current billing file using FTP GET at any time after it has been constructed and placed in the FTP file system by the record formatter. The collection system MUST explicitly FTP DELETE the billing file when it no longer needs it. The record formatter MUST begin construction of a new billing file when it detects that the current billing file has been deleted. The record formatter MUST protect the billing file until it is deleted by the collection system. The record formatter MUST support a minimum 15 minute collection interval. If the collection system attempts to retrieve the billing file before the record formatter has completed building it, the collection system will

determine that the billing file does not exist yet in the formatter's FTP file system. In this case, the collection system MUST back off and try again at a later time. If multiple retrievals of the billing file by multiple collectors are desired, then the last collector must delete the billing file. How this is coordinated between the multiple collectors is beyond the scope of this specification.

10. The NDM-U specification provides a mapping of its document transfer protocol to a file sharing approach as described above. There is a "contract" interface between the producer and consumer of IPDRDocs, embodied in a Capabilities Record. Use of this mapping ensures reliable file transfer and maximum interoperability among systems. In order to meet the requirements stated above, the NDM-U 3.1 protocol MUST be used.

7.1.3 Billing collection interval[1]

Subscriber Usage Billing Records report the absolute traffic counter values for each active Service Flow used by a Cable Modem (Subscriber) during billing collection interval as seen at the end of the interval. The collection interval is defined as the time between the creation of the previous billing file (Tprev) and the creation of the current billing file (Tnow). See Figure 7-1 below. There are two kinds of Service Flows that are reported in the current billing file: 1) SFs that are still active at the time the billing file is created and 2) active SFs that have been deleted and logged during the collection interval. A provisioned or admitted state SF that was deleted before it became active is not recorded in the billing file, even though it was logged by the CMTS. A currently active SF is recorded using Tnow as the timestamp for its counters and are identified in the IPDR element as type=Interim. Deleted SFs that have a deletion time (Tdel) later than Tprev are the only ones recorded in the current billing file (a deleted SF is reported only once). A deleted SF is recorded using its Tdel from the log as the timestamp for its counters and is identified in the IPDR element as type=Stop. Note that the timestamps are based on the formatter's recording times, not the collection system's retrieval times. Since the collection cycle may vary over time, the recording times in the billing file may be used to construct an accurate time base over sequences of billing files.

A currently active SF is recorded using Tnow as the timestamp for its counters and are identified in the IPDR UE element as type=Interim. Deleted SFs that have a deletion time (Tdel) later than Tprev are the only ones recorded in the current billing file (a deleted SF is reported only once). A deleted SF is recorded using its Tdel from the log as the timestamp for its counters and is identified in the IPDR UE element as type=Stop. Note that the timestamps are based on the formatter's recording times, not the collection system's retrieval times. Since the collection cycle may vary over time, the recording times in the billing file may be used to construct an accurate time base over sequences of billing files.

1. section replaced per oss2-n-02069, 05/22/02, ab

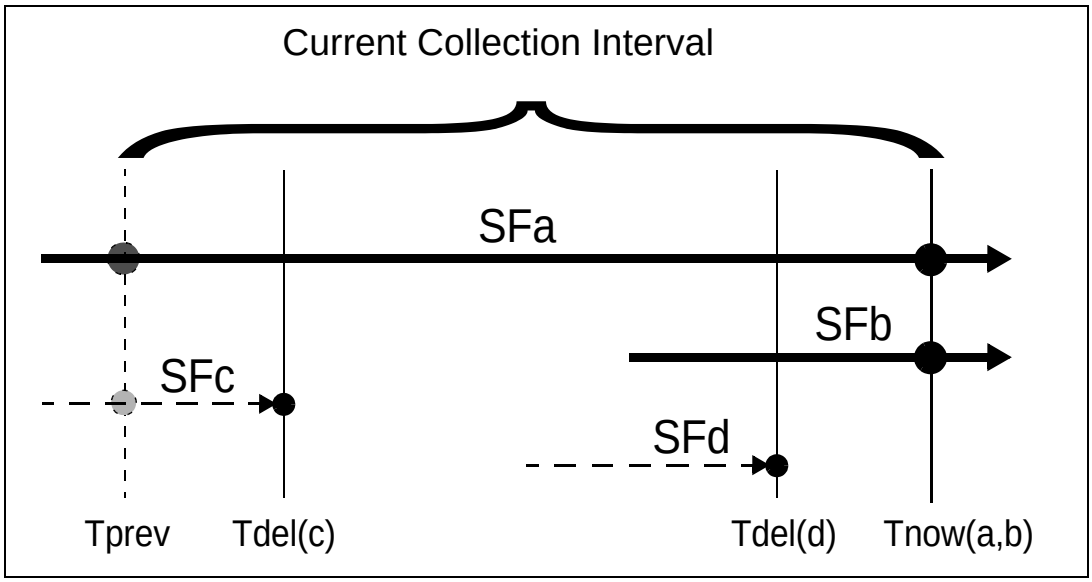


Figure 7-1 Collection interval time base

In the example in Figure 7-1 there are four Service Flows recorded for a Subscriber in the current billing file being created at Tnow. SFa is a long-running SF that was active during the previous collection interval (it has the same SFID in both the current and the previous billing files). SFa was recorded using Tprev in the previous billing file and is recorded using Tnow in the current file. SFb is an active SF that was created during the current collection interval. SFb is recorded for the first time using Tnow in the current file. SFc is a deleted SF that was active during the previous collection interval but was deleted and logged during the current collection interval. SFc was recorded using Tprev in the previous billing file and is recorded using the logged Tdel(c) in the current file. SFd is a deleted SF that was both created and deleted during the current collection interval. SFd is recorded using the logged Tdel(d) in the current billing file only.

7.1.4 Billing file retrieval model[1]

Billing files are built by the record formatter and retrieved by the collection system in a decoupled manner. There is no explicit signaling protocol between them and no prior arrangement regarding the frequency of billing collection.

The formatter is responsible for creating the current billing file and placing it into its FTP file system only when the file is completely built. The formatter only creates one billing file which it must protect until the collection system is done with it. The collection system may retrieve the current billing file via FTP GET at any time after the file becomes available in the formatter's FTP file system. When the collection system has successfully retrieved the billing file, it removes the file via FTP DELETE in the formatter's FTP file system.

The formatter monitors the existence of the billing file in its FTP file system and when it no longer exists, the formatter begins to create the next billing file. The formatter must finish constructing the next billing file and have it ready for retrieval in its FTP file system within 10 minutes of the previous file's deletion. If the billing file does not yet exist in the formatter's FTP file system when the collection system comes to retrieve it, the collection system will back off and return at a later time to try again. Note that if the collection system fails for

1. section replaced per oss2-n-02069, 05/22/02, ab

any reason, the formatter will retain the "current" billing file until such time as the collection system returns to retrieve the file. In this case, even though the recording timestamps in the current billing file may be quite old, the collection system will still retrieve the current file and delete it in the standard manner. The formatter will then immediately begin construction of a new billing file based on the current values of the CMTS's internal absolute 64-bit counters and the current timestamp. The collection system may then return at any time after the minimum cycle time (i.e., 10 minutes) and retrieve the new billing file with the current timestamps. The absolute values of the counters will always be preserved by the CMTS, only the collection interval will be extended due to the outage on the collection system. The billing system can use the recording timestamps in the two files to accurately reconstruct the time base of the counters. Furthermore, the collection system may deliberately vary its collection cycles based on time of day or day of week. This decoupled billing file retrieval model works well for this case also.

The decoupled billing file retrieval model also supports multiple retrievals by multiple collection systems so long as the last collection system deletes the billing file when it is done with it. However, there is no requirement to support multiple simultaneous file transfers from the formatter. How the multiple collection systems coordinate this between themselves is beyond the scope of this specification.

7.1.5 Billing file security model

The billing file security model has two components: authorization to control access to the billing file in the formatter's FTP file system, and encryption to ensure the privacy of the billing data and guarantee its integrity both while it is stored and in transit.

Authorization is provided by the standard FTP userid and password mechanism. The collection system needs read and delete access permissions to the billing file in the formatter's FTP file system. The collection system's userid and password are locally administered by the formatter's FTP implementation.

Encryption is provided by a 40-bit single DES encryption algorithm using a locally administered "shared secret" encryption key in the formatter. The billing file is first compressed and then encrypted before it is finally stored in the formatter's FTP file system. This provides end-to-end security during any intervening storage or transmission steps between the formatter and the billing system. Note that intermediate collection systems that store or transmit the billing file may or may not have the encryption key if the operator so chooses. It is only required that the billing system or mediation system responsible for parsing the billing records must have the shared encryption key. Also, the billing encryption key must be provided solely for the use of the billing interface and not be shared with other applications in the formatter host.

The formatter must provide authorization and encryption capabilities, but the operator might not utilize them. In this case the formatter must turn off authorization when the userid and password are null, and must turn off encryption when the billing file encryption key is null.

7.1.6 IP Detail Record (IPDR) standard[1]

The IPDR Organization (see http://www.ipdr.org/) has defined a generic model for using XML Schema in IP Detail Recording applications. Industry specific IP billing applications such as the Cable Data Systems Subscriber Usage Billing Record can be added to the IPDR standard by mapping the application semantics onto the NDM-U XML Schema syntax. See Annex B for the DOCSIS OSSI Service Specification submission to IPDR.org for the Cable Data Systems Subscriber Usage Billing Record. Annex B also contains an example IPDR XML Schema format Subscriber Usage Billing file and the IPDR standard XML Schema file that describes the IPDR syntax.

1. section replaced per oss2-n-02069, 05/22/02, ab

7.1.6.1 IPDR Network Model[1]

The IPDR Network Model is given in the ipdr.org standard [NDM-U 3.1] and is portrayed in Figure 7-2, below.

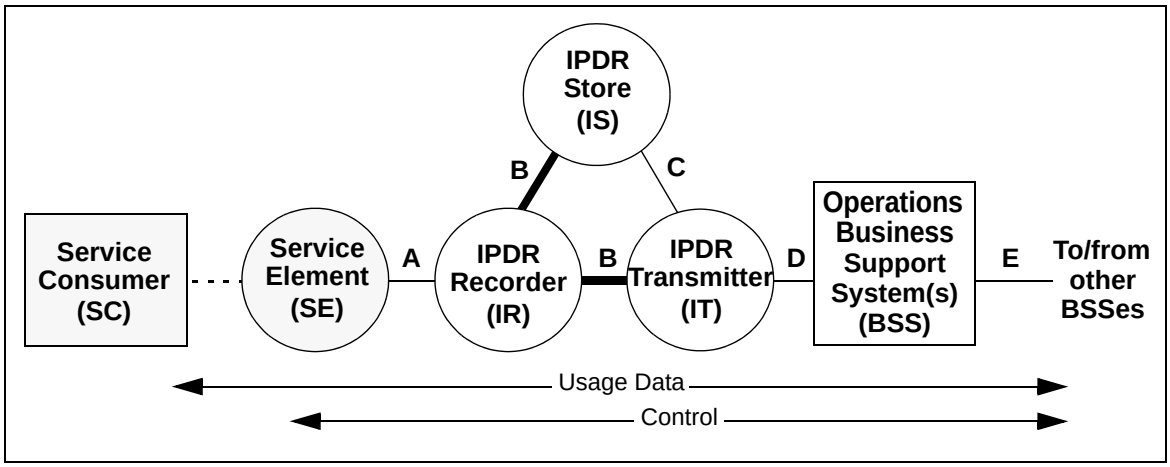


Figure 7-2 IPDR Basic Network Model (ref. [NDM-U 3.1] from www.ipdr.org)

In this model, the Service Consumer (SC) is the Cable Data Service Subscriber identified by a Cable Modem MAC address, current CM IP address, and current CPE IP addresses. The Service Element (SE) is the CMTS identified by its host name and IP address. The IPDR Recorder (IR) is the billing record formatter function that creates the IPDR XML format billing records from the internal counters maintained by the CMTS for each Subscriber and active Service Flow. The IPDR Recorder is a function that may be implemented within the CMTS or hosted on another platform such as an Element Management System (EMS). The IPDR Store (IS) and the IPDR Transmitter (IT) represent the billing record collectors that retrieve the billing records from the IPDR Recorder. In this specification the IS or IT would retrieve the billing file from the IR on a collection cycle determined by the IT or IS. The A-interface is not specified by the IPDR standard because it is an internal interface between the SE and the IR. However, the B-interface is specified by the IPDR standard as a file of IPDR records formatted according to the IPDRdoc XML Data Type Definition (DTD) file (see Annex B). The B-interface MUST be implemented using the Cable Data Systems Subscriber Usage Billing Record submission to the IPDR standard as defined in Annex B. The C-, D-, and E-interfaces are beyond the scope of this specification.

In this model, the Service Consumer (SC) is the Cable Data Service Subscriber identified by their Cable Modem MAC address, current CM IP address, and current CPE IP addresses. The Service Element (SE) is the CMTS identified by its host name and IP address. The IPDR Recorder (IR) is the billing record formatter function that creates the [NDM-U 3.1] XML Schema format IPDRs from the internal counters maintained by the CMTS for each Subscriber and active Service Flow. The IPDR Recorder is a function that represents the mediation component on a platform such as a Record Keeping Server (RKS). The IPDR Store (IS) and the IPDR Transmitter (IT) represent, respectively, the persistence and document transfer functions of such a platform. In this specification, the OBSS would retrieve the billing file from the IT on a collection cycle as specified in Section 7.1.2. The A-interface is not specified by [NDM-U 3.1] because it is an internal interface between the SE and the IR. However, the D-interface is specified by the NDM-U specification as a file of IPDR records formatted according to the IPDRdoc XML Schema file (see Annex B). The D-interface MUST be implemented

1. section replaced per oss2-n-02069, 05/22/02, ab

using the Cable Data Systems Subscriber Usage Billing Record Service Specification submission to the IPDR standard as defined in Annex B. The B- and E-interfaces are beyond the scope of this specification.

7.1.6.2 IPDR Record Structure[1]

The IPDR standard specifies the IPDRDoc record structure. The IPDRDoc XML DTD (see Appendix C) defines the hierarchy of elements as shown in Figure 5 below.

The [NDM-U 3.1] specification specifies the IPDRDoc record structure. The IPDRDoc XML Schema in Annex B defines the hierarchy of elements as shown in Figure 7-3 below.

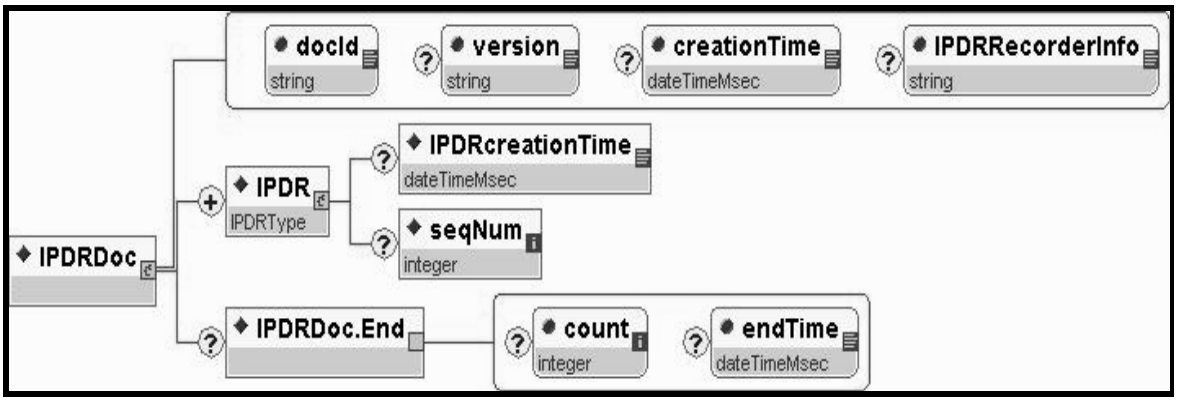


Figure 7-3 IPDR XML Element Hierarchy (ref. [NDM-U 3.1] from www.ipdr.org)

The following elements are used by the Cable Data Systems Subscriber Usage Billing Record (see Annex B):

1. *IPDRDoc* is the outermost element that describes the IPDR billing file itself. It defines the version of the specification and the timestamp for the file. An IPDRDoc is composed of multiple IPDR records.
2. *IPDRRec* describes the IPDR Recorder (IR) from the network model in Figure 7-2. This element identifies the billing record formatter by the fully-qualified host name of the CMTS or the EMS where the formatter resides.
3. *IPDR* describes a single Subscriber Usage Billing Record. This element identifies the time of the billing collection event. While the IPDR record structure is designed to describe most time based and event-oriented IP services, this feature is not particularly relevant to the Cable Data Service Subscriber Usage Billing Records and is largely ignored. This is because a Service Session at the CMTS is just the aggregate usage of an active Service Flow during the billing collection interval. Another way to look at it is as if there is really only one event being recorded: the billing collection event itself.
4. *SS* describes the Service Session which in a CMTS is an active Service Flow. This element identifies the Service Flow by its Service Class Name (SCN). The Service Session is further structured into the Service Consumer (SC), which is the Subscriber, and the Service Element (SE), which is the CMTS, that are related by this Service Session.

1. section replaced per oss2-n-02069, 05/22/02, ab

5. *SC* describes the Service Consumer from the network model in Figure 7-2. This is the Subscriber identified by its Cable Modem MAC address and its current IP address plus the current IP addresses of all CPEs using the Service Flows during the collection interval. Since several IPDRs will describe the Service Flows used by the Subscriber, the SC occurs just once. The common SC is then is referenced in each subsequent IPDR by an *SCRef* element.
6. *SE* describes the Service Element from the network model in Figure 7-2. This is the CMTS identified by its host name and its IP address and includes the sysUpTime for the CMTS. Since all the IPDRs in a given IPDRDoc are for a single CMTS, the SE occurs just once at the top of the document. The common SE is then referenced in each subsequent IPDR by an *SERef* element.
7. *UE* describes the Usage Event details. This element identifies the counters that are associated with the Service Flow and whether the Service Flow has terminated (type=Stop) or is continuing (type=Interim). A set of <v…/> elements identify the usage attribute values including the *SFID* and the actual 64-bit CMTS counters for the Service Flow in decimal format ASCII notation. The SFID facilitates the correlation of sequential counter sets for the same Service Flow from one IPDRDoc file to the next. Note well that the direction of a Service Session in the IPDR model is from the SC's frame of reference—this means that a Service Flow is seen from the CM's point of view. Thus, downstream Service Flows in the CMTS are received by the CM while upstream Service Flows in the CMTS are sent by the CM. Also, note that since a Service Flow is unidirectional a UE may either have send-counters or receive-counters, but not both. The usage attribute value names for downstream Service Flows are *recvPkts*, *recvOctets*, *recvSLADropPkts*, and *recvSLADelayPkts*. The usage attribute value names for upstream Service Flows are *sendPkts* and *sendOctets*. There are no drop or delay counters in the upstream direction because these are not known to the CMTS.

IPDRDoc.End is the last element inside IPDRDoc that describes the IPDR document itself. It defines the count of IPDRs that are contained in the document and the ending timestamp for the document creation.

7.2 Configuration Management

Configuration management is concerned with initializing, maintaining, adding and updating network components. In a DOCSIS environment, this includes a cable modem and/or CMTS. Unlike performance, fault, and account management, which emphasize network monitoring, configuration management is primarily concerned with network control. Network control, as defined by this interface specification, is concerned with modifying parameters in and causing actions to be taken by the cable modem and/or CMTS. Configuration parameters could include both identifiable physical resources (for example, Ethernet Interface) and logical objects (for example, IP Filter Table).

Modifying the configuration information of a CM and/or CMTS can be categorized as *non-operational* or *operational*.

Non-operational changes occur when a manager issues a modify command to a CM/CMTS, and the change doesn't affect the operating environment. For example, a manager may change contact information, such as the name and address of the person responsible for a CMTS.

Operational changes occur when a manager issues a modify command to a CM/CMTS, and the change affects the underlying resource or environment. For example, a manager may change the docsDevResetNow object from false to true, which in turn will cause the CM to reboot.

To adjust the necessary attribute values, the CM and CMTS MUST support MIB objects as specified in Section 6 of this document.

While the network is in operation, configuration management is responsible for monitoring the configuration and making changes in response to commands via SNMP or in response to other network management functions.

For example, a performance management function may detect that response time is degrading due to a high number of uncorrected frames, and may issue a configuration management change to modify the modulation type from 16Qam to QPSK. A fault management function may detect and isolate a fault and may issue a configuration management change to bypass the fault.

7.2.1 Version Control

The CM MUST support software revision and operational parameter configuration interrogation.

The CM MUST include at least the hardware version, Boot ROM image version, vendor name, software version, and model number in the sysDescr object (from [RFC 1907]). The CM MUST support docsDevSwCurrentVers MIB object and the object MUST contain the same software revision information as shown in the software information included in the sysDescr object.

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

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

Each type-value pair MUST be separated with a colon and blank space. Each pair is separated by a ";" followed by a blank. For instance, a sysDescr of a CM of vendor X, hardware version 5.2, Boot ROM version 1.4, SW version 2.2, and model number X

MUST appear as following:

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

The CM MUST report at least all of the information necessary in determining what SW the CM is capable of being upgraded to. If any fields are not applicable, the CM MUST report "NONE" as the value. For example; CM with no BOOTR, CM will report BOOTR: NONE.

The CM MUST implement the docsDevSwCurrentVers object ([RFC 2669]) to report the current software version.

The intent of specifying the format of sysObjectID and sysDescr is to define how to report information in a consistent manner so that sysObjectID and sysDescr field information can be programmatically parsed. This format specification does not intend to restrict the vendor's hardware version numbering policy.

The CMTS MUST implement the sysDescr object (from [RFC 1907]). For the CMTS, the format and content of the information in sysDescr is vendor-dependent.

7.2.2 System Initialization and Configuration

There are several methods available to configure CM and CMTS including console port, SNMP set, configuration file, and configuration-file-based SNMP encoded object. The CM MUST support system initialization and configuration via configuration file, configuration-file-based SNMP encoded object and SNMP set. The CMTS MUST support system initialization and configuration via telnet connection, console port, and SNMP set. The CM and CMTS (only CMTS that support configuration by configuration file) MUST support any valid configuration file regardless of configuration file size.

7.2.3 Secure Software Upgrades

The CM secure software upgrade process is documented in detail in Appendix D of the DOCSIS BPI+ specification.

DOCSIS 2.0 CMs MUST use the secure software upgrade mechanism to perform software upgrade regardless of the version (1.0, 1.1, or 2.0) of the CMTS to which it is connected.

When a 2.0 CM is connected to a 2.0 CMTS, the CM MUST operate in either DOCSIS 2.0 mode, DOCSIS 1.1 mode, or DOCSIS 1.0 mode.

When a 2.0 CM is connected to a 1.1 CMTS, the CM MUST operate in either DOCSIS 1.1 mode or DOCSIS 1.0 mode.

When a 2.0 CM is connected to a 1.0 CMTS, the CM MUST operate in DOCSIS 1.0 mode.

This means that a DOCSIS 2.0 CM MUST use secure software upgrade mechanism to perform software upgrade regardless of what mode it operates in (1.0 mode, 1.1 mode or 2.0 mode). There are two available secure software download schemes: the manufacturer control scheme and the operator control scheme.

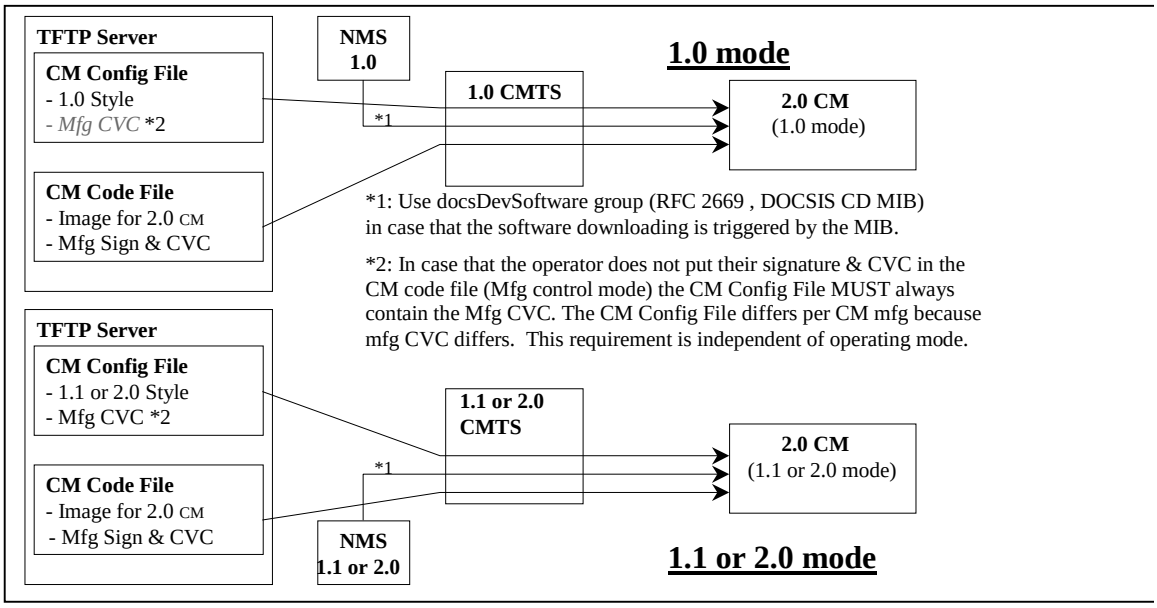


Figure 7-4 Manufacturer control scheme

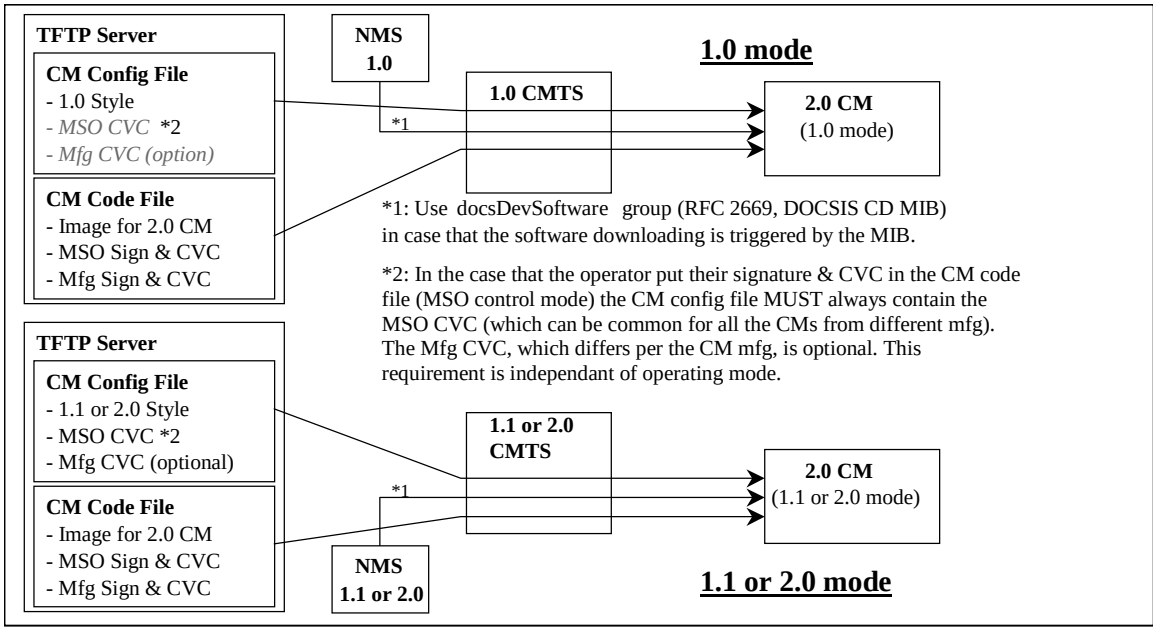


Figure 7-5 Operator control scheme

Prior to secure software upgrade initialization, CVC information needs to be initialized at the CM for software upgrade. Depending on the scheme (described above) that the operator chooses to implement, appropriate CVC information MUST be included in the configuration file. It is recommended that CVC information always be present in the configuration file so that a device will always have the CVC information initialized and read if the operator decides to use a SNMP-initiated upgrade as a method to trigger a secure software upgrade operation. If the operator decides to use a configuration-file-initiated upgrade as a method to trigger secure software download, CVC information needs to be present in the configuration file at the time the modem is rebooted to get the configuration file that will trigger the upgrade only.

There are two methods to trigger secure software download: SNMP-initiated and configuration-file-initiated. Both methods MUST be supported by CMs and MAY be supported by CMTSes.

The following describes the SNMP-initiated mechanism. Prior to a SNMP-initiated upgrade, a CM MUST have valid X.509-compliant code verification certificate information. 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 CM 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 CM suffers a loss of power or resets during SNMP-initiated upgrade, the CM MUST resume the upgrade without requiring manual intervention, and when the CM 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 CM

If the CM reaches the maximum number of retries (3) resulting from multiple losses of power or resets during either an SNMP-initiated upgrade or a configuration-file-initiated upgrade, the CM MUST behave as specified in [DOCSIS 5]; in addition, the CM'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 CM

If a CM exhausts the required number of TFTP retries by issuing a total of 16 consecutive retries, the CM MUST behave as specified in [DOCSIS 5] and then fall back to last known working image and proceed to an operational state and adhere to the following requirements:

- docDevSwAdminStautus MUST be allowProvisioningUpgrade(2)
- docDevSwFilename 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 CM

After the CM has completed the SNMP-initiated secure software upgrade, the CM MUST behave as specified in [DOCSIS 5] and MUST reboot and become operational with the correct software image. After the CM is registered, it MUST adhere to the following requirements:

- set its docsDevSwAdminStatus to ignoreProvisioningUpgrade(3)
- set its docsDevOperStatus to completeFromMgt(3)
- reboot

The CM MUST properly use ignoreProvisioningUpgrade status to ignore software upgrade value that may be included in the CM configuration file and become operation with the correct software image. After the CM is registered, it MUST adhere to the following requirements:

- docsDevSwAdminStatus MUST be ignoreProvisioningUpgrade(3)
- docsDevSwFilename MAY be the filename of the software currently operating on the CM
- docsDevSwServer MAY be the address of the TFTP server containing the software that is currently operating on the CM
- docsDevSwOperStatus MUST be completeFromMgt(3)
- docsDevSwCurrentVer MUST be the current version of the software that is operating on the CM

After the CM has completed the configuration-file-initiated secure software upgrade, the CM MUST behave as specified in [DOCSIS 5] and MUST reboot and become operational with the correct software image. After the CM is registered, it MUST adhere to the following requirements:

- docsDevSwAdminStatus MUST be allowProvisioningUpgrade(2)
- docsDevSwFilename MAY be the filename of the software currently operating on the CM
- docsDevSwServer MAY be the address of the TFTP server containing the software that is currently operating on the CM
- docsDevSwOperStatus MUST be completeFromProvisioning(2)
- docsDevSwCurrentVer MUST be the current version of the software that is operating on the CM

In the case where the CM successfully downloads (or detects during download) an image that is not intended for the CM device, the CM MUST behave as specified (refer to [DOCSIS 5], section 12.1 “Downloading Cable Modem Operating Software”):

- DocsDevSwAdminStatus MUST be allowProvisioingUpgrade(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 CM

In the case where the CM determines that the download image is damaged or corrupted, the CM MUST reject the newly downloaded image. The CM MAY re-attempt to download if the MAX number of TFTP sequence retries has not been reached. If the CM chooses not to retry and the MAX number of TFTP sequence retries has not been reached, the CM MUST fall back to the last known working image and proceed to an operational state, generate appropriate event notification as specified in Annex D, and adhere to the following requirements:

- DocsDevSwAdminStauts 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 CM

In the case where the CM determines that the image is damaged or corrupted, the CM MUST reject the newly downloaded image. The CM 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 CM software download attempt, the CM MUST fall back to the last known working image and proceed to an operational state. In this case, the CM 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 CM reaches the operational state, the CM MUST adhere to the following requirements:

- DocsDevSwAdminStauts 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 CM

7.3 Protocol Filters

The CM MUST implement LLC, SNMP Access, and IP protocol filters. The LLC protocol filter entries can be used to limit CM forwarding to a restricted set of network-layer protocols (such as IP, IPX, NetBIOS, and AppleTalk). The IP protocol filter entries can be used to restrict upstream or downstream traffic based on source and destination IP addresses, transport-layer protocols (such as TCP, UDP, and ICMP), and source and destination TCP/UDP port numbers.

CM MUST apply filters (or more properly, classifiers) in an order appropriate to the following layering model; specifically, the inbound MAC (or LLC) layer filters are applied first, then the "special" filters, then the IP layer inbound filters, then the IP layer outbound filters, then any final LLC outbound filters. Note that LLC outbound filters are expected future requirements of the Cable Device MIB.

7.3.1 LLC filters

Inbound LLC filters, from docsDevFilterLLCTable, MUST be applied to layer-2 frames entering the CM from either the CATV MAC interface{2} and/or any CM CPE interface.

The object docsDevFilterLLCUnmatchedAction MUST apply to all (CM) interfaces. The default value of the (CM) docsDevFilterLLCUnmatchedAction MUST be set to accept.

7.3.1.1 docsDevFilterLLCUnmatchedAction

If the CM docsDevFilterLLCUnmatchedAction is set to discard(1), any L2 packet that does not match any LLC filters will be discarded, otherwise accepted. If CM docsDevFilterLLCUnmatchedAction is set to accept, any L2 packet that does not match any LLC filters will be accepted, otherwise discarded.

Another way to interpret this is the following:

```
action = UnMatchedAction
Iterate through the table
    if there is a match (packet.protocol = row.protocol)
    {
        reverse the action (accept becomes discard, discard becomes accept)
        apply action to the packet
        terminate the iteration
    }
```

LLC (CM) filters MUST apply to the inbound traffic direction only. Traffic generated from the CM MUST not be applied to LLC filters (i.e., ARP requests, SNMP responses).

The CM MUST support a minimum of ten LLC protocol filter entries.

7.3.2 Special filters

Special filters are IP spoofing filters and SNMP access filters. IP spoofing filters MUST only be applied to packets entering the CM from CMCI interface(s). SNMP access filters are in effect when the CM is not running in SNMPv3 agent mode and can be applied to both CMCI and CATV interfaces.

According to the interface number section of this document, the CMCI interface is a generic reference to any current or future form of CM CPE interface port technology.

7.3.3 IP spoofing filter[1]

DOCSIS 2.0 CMs MAY implement an IP spoofing filter as specified in RFC 2669.

If a CM supports the IP Spoofing filter functionality specified in RFC 2669, the CM MUST adhere to the following requirements:

- Implement all MIB objects in the docsDevCpeGroup
- The default value of docsDevCpeIpMax = -1

7.3.3.1 Additional requirement on dot1dTpFdbTable (RFC 1493)[2]

CM CPE MAC addresses learned via the CM configuration file MUST set the dot1dTpFdbStatus to “mgmt”. It is assumed that the number of “mgmt”-configured CM CPE MAC addresses is less than, or equal to, the TLV type-18 value (Maximum Number of CPE).

7.3.4 SNMP Access Filter

The SNMP access filters MUST be applied to SNMP packets entering from any interfaces and destined for the CM. SNMP access filter MUST be applied after IP spoofing filters for the packets entering the CM from the CMCI interface. Since SNMP access filter function is controlled by docsDevNmAccessTable, SNMP access filter is available and applies only when the CM is in SNMP v1/v2c NmAccess mode.

When the CM is running in SNMP Coexistence mode, SNMP access MUST be controlled and specified by the MIB Objects in [RFC 2571] and [RFC2576].

7.3.4.1 docsDevNmAccessIP and docsDevNmAccessIpMask[3]

A device that implements docsDevNmAccessTable applies the following rules in order to determine whether to permit SNMP access from a given source IP address (SrcIpAddr):

1. If (docsDevNmAccessIp == "255.255.255.255"), the CMTS/CM MUST permit the access from any SrcIpAddr.
2. If ((docsDevNmAccessIp AND docsDevNmAccessIpMask) == (SrcIpAddr AND docsDevNmAccesIpMask)), the CMTS/CM MUST permit the access from SrcIpAddr.
3. If neither #1 nor #2 is applied, the CMTS/CM MUST NOT permit the access from SrcIpAddr.

The CMTS/CM’s default value of the docsDevNmAccessIpMask MUST be set to “0.0.0.0”.

---

1. original section 7.3.3.1, “DocsDevCpeIpMax, TLV type-18, and FilterCpeTable”, deleted per oss2-n-02053, 05/17/02, ab
2. section gutted per oss2-n-02088, 06/05/02, ab
3. section replaced per oss2-n-02054, 05/17/02, ab

The following table contains sample MIB values and the access granted.

Table 7-1 Sample docsDevNmAccessIp values

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

7.3.5 IP filter

The object docsDevFilterIPDefault MUST apply to all CM interfaces. DOCSIS 2.0-compliant CMs MUST support a minimum 16 IP filters.

7.4 Fault management

The goals of fault management are remote monitoring/detection, diagnosis, and correction of problems. Network Management operators rely on the ability to monitor and detect problemss (such as ability to trace and identify faults, accept and act on error-detection events), as well as the ability to diagnose and correct problems (such as perform a sequences of diagnostic tests, correct faults, and display/maintain event logs).

This section defines what MUST be available to support remote monitoring/detection, and diagnosis and correction of problems.

7.4.1 SNMP Usage

In the DOCSIS environment, the goals of fault management are the remote detection, diagnosis, and correction of network problems. Therefore, the standalone CM MUST support SNMP management traffic across both the CPE and CATV MAC interfaces regardless of the CM's connectivity state. CCCMs MAY ignore the CPE management traffic, and MUST support SNMP on the CATV MAC interface once connectivity to CMTS is established. CM SNMP access may be restricted to support policy goals. CM installation personnel can use SNMP queries from a station on the CMCI side to perform on-site CM and diagnostics and fault classification (note that this may require temporary provisioning of the CM from a local DHCP server). Further, future CMCI side customer applications, using SNMP queries, can diagnose simple post-installation problems, avoiding visits from service personnel and minimizing help desk telephone queries.

Standard MIB-II support MUST be implemented to instrument interface status, packet corruption, protocol errors, etc. The transmission MIB for Ethernet-like objects [RFC 2665] MUST be implemented on each cable device (CMTS/CM) Ethernet and Fast Ethernet port. Each cable device (CMTS/CM) MUST implement the ifXTable [RFC 2863] to provide discrimination between broadcast and multicast traffic.

The cable device (CMTS/CM) MUST support managed objects for fault management of the PHY and MAC layers. The RFI-MIB-IPCDN-DRAFT MIB includes variables to track PHY state such as codeword collisions and corruption, signal-to-noise ratios, transmit and receive power levels, propagation delays, micro-reflections, in channel response, and Sync loss. The RFI-MIB-IPCDN-DRAFT MIB also includes variables to track MAC state, such as collisions and excessive retries for requests, immediate data transmits, and initial ranging requests.

For fault management at all layers, the cable device (CMTS/CM) MUST generate replies to SNMP queries (subject to policy filters) for counters and status. The cable device (CMTS/CM) MUST send SNMP traps to one

or more trap NMSs (subject to policy), and MUST send SYSLOG events to a SYSLOG server (if a SYSLOG server is defined).

When the cable device (CM) is operating in SNMP v1/v2c NmAccess mode it MUST support the capability of sending traps as specify 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 { Disable SNMPv2trap }
    ::= { docsDevNmAccessEntry 8 }
```

Any cable device (CMTS/CM) SHOULD implement the ifTestTable [RFC 2863] for any diagnostic test procedures that can be remotely initiated.

7.4.2 Event Notification

A cable device (CMTS/CM) MUST generate asynchronous events that indicate malfunction situations and notify about important non-fault events. Events could be stored in CMTS/CM device internal event LOG file, in non-volatile memory, get reported to other SNMP entities (as TRAP or INFORM SNMP messages), or be sent as a SYSLOG event message to a pre-defined SYSLOG server. Events MAY also be sent to the cable device (CMTS/CM) console; as a duplicate (identical) message to the optional console destination.

Event notification implemented by a cable device (CMTS/CM) MUST be fully configurable, by priority class; including the ability to disable SNMP Trap, SYSLOG transmission, and local logging. CMTS/CM MUST implement docsDevEvControlTable to control reporting of event classes. The object docsDevEvReporting MUST be implemented as RW for CMTS/CM.

A cable device (CMTS/CM) MUST support the following event notification mechanisms (regardless of the cable device's SNMP mode):

- local event logging
- SNMP TRAP/INFORM (trap-versions/targets/limiting/throttling)
- SYSLOG (targets/limiting/throttling)

Refer to the following sections for event notification implementation details.

When a CM is in SNMP v1/v2c NmAccess mode, the CM MUST support event notification functions including local event logging, SYSLOG (targets/limiting/throttling) and SNMP TRAP (trap-versions/targets/limiting/throttling) as specified in RFC 2669 and the current specification. When CM is in SNMP coexistence mode, CM MUST support event notification functions including local event logging, SYSLOG (targets/limiting/throttling) and SNMP TRAP (limiting/throttling) as specified in RFC 2669 and the current specification, and SNMP notification functions as specified in RFC 2573.

If the CMTS supports, and is in SNMP v1/v2c NmAccess mode, the CMTS MUST support event notification functions including local event logging, SYSLOG (targets/limiting/throttling) and SNMP TRAP (limiting/throttling) as specified in RFC 2669 and the current specification; however, SNMP TRAP (trap-versions/targets)

MAY be implemented as specified in RFC 2669 and OSSI 1.1, or a vendor-proprietary MIB. When the CMTS is in SNMP Coexistence mode, the CMTS MUST support event notification functions including local event logging, SYSLOG (targets/limiting/throttling) and SNMP TRAP (limiting/throttling) as specified in RFC 2669 and the current specification, and SNMP notification functions as specified in RFC 2573.

7.4.2.1 Local Event Logging

Event logging provides a mechanism to store events in local-volatile and optionally local-nonvolatile memory. The event log storage and access mechanism MUST be implemented in a cable device (CM or CMTS) as described below. A DOCSIS 2.0 compliant cable device MUST implement docsDevEventTable with additional requirements as specified by the OSSI Specification 12.0.

The cable device event log MUST be organized as a cyclic buffer with a minimum of ten entries, and MAY persist across reboots. The event log table MUST be accessible through the DocsDevEventTable [RFC 2669] by the cable device (CM or CMTS).

Aside from the procedures defined in this document, event recording must conform to the requirements of RFC 2669. Event descriptions must appear in English and must not be longer than 255 characters, which is the maximum defined for SnmpAdminString.

Events are identical if their EventIds are identical. For identical events occurring consecutively, the CM MAY choose to store only a single event. In such a case, the event description recorded MUST reflect the most recent event.[1]

The EventId digit is a 32-bit unsigned integer. EventIds ranging from 0 to (2^{31} - 1) are reserved by DOCSIS. The EventId MUST be converted from the error codes defined in Annex D.

The EventIds ranging from 2^{31} to (2^{32} - 1) MUST be used as vendor-specific EventIds using the following format:

- Bit 31 is set to indicate vendor-specific event
- Bits 30-16 contain the lower 15 bits of the vendor's SNMP enterprise number
- Bits 15-0 are used by the vendor to number events

Section 7.4.2.2.2 describes rules to generate unique EventIds from the error code.

The RFC 2669 docsDevEvIndex object provides relative ordering of events in the log. The creation of local-volatile and local-nonvolatile logs necessitates a method for synchronizing docsDevEvIndex values between the two local logs after reboot. The following procedure MUST be used after reboot:

- The values of docsDevEvIndex maintained in the local non-volatile log MUST be renumbered beginning with 1.
- The local-volatile log MUST then be initialized with the contents of the local non-volatile log.
- The first event recorded in the new active session's local-volatile log MUST use as its docsDevEvIndex the value of (last restored non-volatile docsDevEvIndex + 1).

A reset of the log initiated through an SNMP SET of the RFC 2669 docsDevEvControl object MUST clear both the local-volatile and local-nonvolatile logs.

---

1. para modified per oss2-n-02062, 05/17/02, ab

7.4.2.2 Format of Events

Annex D lists all DOCSIS events.

The following sections explain in detail how to report these events via any of the three mechanisms (local event logging, SNMP trap and syslog).

7.4.2.2.1 SNMP TRAP/INFORM

A cable device (CMTS or CM) MUST send the following generic SNMP traps, as defined in standard MIB [RFC 1907] and [RFC 2863]:

- coldStart (warmStart is optional) [RFC 1907]
- linkUp [RFC 2863]
- linkDown [RFC 2863]
- SNMP authentication-Failure [RFC 1907]

A cable device (CMTS or CM) MUST implement the SNMP traps defined in the DOCS-CABLE-DEVICE-TRAP-MIB, which is complementary to the existing standard DOCSIS MIBs (CABLE-DEVICE-MIB, BPI-PLUS-MIB, and DOCS-IF-MIB) and defined in Annex H:

- A CM or CMTS in SNMP v1/v2c NmAccess mode MUST support SNMPv1 and SNMPv2c Traps.
- A CM or CMTS in SNMP Coexistence mode MUST support SNMPv1, SNMPv2c, and SNMPv3 Traps.
- Cable devices (CMTS or CM) MUST support INFORM.

INFORM is a variation of trap and requires the receiving host to acknowledge the arrival of an InformRequest-PDU with an InformResponse-PDU. An InformRequest-PDU is exactly the same as a trap-PDU except that the value in the PDU-type field is 6 for InformRequest-PDU instead of 7 for SNMPv2-trap-PDU. SNMPv1 does not support INFORM.

When an SNMP trap defined in the DOCS-CABLE-DEVICE-TRAP-MIB is enabled in a CM, it MUST send notifications for any event in its category whose priority is either “error” or “notice”. See Table 7-2, “Default event priorities for the Cable Modem device,” on page 69. It MAY notify (optionally) events with other priorities when it is possible.

When the SNMP trap defined in the DOCS-CABLE-DEVICE-TRAP-MIB is enabled in a CMTS, it MUST send notifications for an event whose priority is “critical” or “error” or “warning” or “notice”. See Table 7-3, “Default Event priorities for the CMTS Device,” on page 69. It MAY send (optionally) events with other priorities.

Vendor-specific events reportable via SNMP TRAP MUST be described in the vendor documents. Vendors can also define vendor-specific SNMP traps and MUST do so in the private MIBs.

When defining a vendor-specific SNMP trap, the OBJECTS statement of the private trap definition SHOULD contain at least the objects explained below. For CM traps, docsDevEvLevel, docsDevEvId, docsDevText, docsIfDocsisCapability, docsIfDocsisCapability, ifPhysAddress, and docsIfCmCmtsAddress SHOULD be included. For CMTS traps, docsDevEvLevel, docsDevEvId, docsDevEvText, docsIfCmtsCmStatusDocsisMode, docsIfCmtsCmStatusMacAddress, docsIfDocsisOperMode, and ifPhysAddress SHOULD be included. For a description of the usage of these objects, please refer to the DOCS-CABLE-DEVICE-TRAP-MIB. More objects may be contained in the OBJECTS body as desired.

Since the objects contained in these SNMP traps are the same objects in the SNMP local event table, CM MUST turn on local event logging on a particular priority whenever the SNMP traps are configured on that event priority.

7.4.2.2.2 SYSLOG message format

For DOCSIS events, the CM's Syslog message MUST be sent in the following format; for non-DOCSIS events, it is optional:

```
<level>CABLEMODEM[vendor]: <eventId> text vendor-specific-text
```

Where *level* is an ASCII representation of the event priority, enclosed in angle brackets, which is constructed as an OR of the default Facility (128) and event priority (0-7). The resulting level has the range between 128 and 135.

The CMTS's Syslog message MUST be sent in the following format:

For DOCSIS events, the CMTS's Syslog message MUST be sent in the following format; for non-DOCSIS events, it is optional:

```
<level>TIMESTAMP HOSTNAME CMTS[vendor]: <eventId> text vendor-specific-text
```

Where:

- *level* is an ASCII representation of the event priority, enclosed in angle brackets, which is constructed as an OR of the default Facility (128) and event priority (0-7). The resulting level ranges between 128 and 135.
- *TIMESTAMP* and *HOSTNAME* MAY be sent after <*level*> by the CMTS. If the *TIMESTAMP* and *HOSTNAME* fields are sent, they MUST be in the same format as the IETF proposed "draft-ietf-syslog-syslog-06.txt" *TIMESTAMP* and *HOSTNAME* format and MUST be sent together. The one space after *TIMESTAMP* is part of the *TIMESTAMP* field. The one space after *HOSTNAME* is part of the *HOSTNAME* field.
- *vendor* is the vendor name for the vendor-specific SYSLOG messages or DOCSIS for the standard DOCSIS messages.
- *eventId* is an ASCII representation of the INTEGER number in decimal format, enclosed in angle brackets, which uniquely identifies the type of event. This number MUST be the same number that is stored in the docsDevEvId object in docsDevEventTable, and is also associated with SNMP TRAP in the "SNMP TRAP/Inform" section.

  For the standard DOCSIS 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 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 gap 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 gap in the left side.

  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 DOCSIS (0 to 2^{31}-1). The first letter of an error code is always in upper-case.
- *text*: for the standard DOCSIS messages, this string MUST contain the textual description as defined in Annex D.

- *vendor-specific-text* MAY be provided by vendors for vendor-specific information.

There are products in the marketplace that expect existing syslog messages in their current format for fault management, which the DOCSIS syslog message format would break. So, for CM and CMTS, it is optional for the syslog message format of the non-DOCSIS events to follow the above formats.

For example, the syslog event for the event D04.2, “Time of the day received in invalid format”, is as follows:

```
<132>CABLEMODEM[DOCSIS]: <44000402> Time of Day Response but invalid data/format.
```

The number 44000402 in the example is the number assigned by DOCSIS to this particular event.

7.4.2.3 Standard DOCSIS events for CMs

The DOCSIS cable device MIB [RFC 2669] defines 8 priority levels and a corresponding reporting mechanism for each level. The standard DOCSIS events specified in this document utilize a subset of these priority levels.

Emergency event (priority 1) Reserved for vendor-specific ‘fatal’ hardware or software errors that prevents normal system operation and causes reporting system to reboot.

Every vendor may define their own set of emergency events. Examples of such events might be ‘no memory buffers available’, ‘memory test failure’, and so on. (Such basic cross-vendor type events should be included in the DOCSIS 2.0 “Events for Notification” Appendix F so that vendors do not define many overlapping EventIds in vendor-private scope.)

Alert event (priority 2) A serious failure, which causes reporting system to reboot but it is not caused by h/w or s/w malfunctioning. After recovering from the critical event, the system MUST send a cold/warm start notification. The alert event could not be reported as a Trap or SYSLOG message and MUST be stored in the internal log file. The code of this event MUST be saved in non-volatile memory and reported later through the docsIfCmStatusCode SNMP variable [RFI-MIB-IPCDN-DRAFT].

Critical event (priority 3) A serious failure that requires attention and prevents the device from transmitting data but could be recovered without rebooting the system. After recovering from the error event Cable Modem Device MUST send the Link Up notification. Critical events could not be reported as a Trap or SYSLOG message and MUST be stored in the internal log file. The code of this event MUST be reported later through docsIfCmStatusCode SNMP variable [RFI-MIB-IPCDN-DRAFT]. Examples of such events might be configuration file problems detected by the modem or the inability to get an IP address from the DHCP server.

Error event (priority 4) A failure occurred that could interrupt the normal data flow but will not cause the modem to re-register. Error events could be reported in real time by using the trap or SYSLOG mechanism.

Warning event (priority 5) A failure occurred that could interrupt the normal data flow but will not cause the modem to re-register. ‘Warning’ level is assigned to events both modem and CMTS have information about. To prevent sending the same event both from the CM and the CMTS, the trap and Syslog reporting mechanism is disabled by default for this level.

Notice event (priority 6) The event is important, but is not a failure and could be reported in real time by using 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) The event is of marginal importance, and is not a failure, but could be helpful for tracing the normal modem operation. By default, these events are not saved into the local event log and no Syslog/trap message is sent.

Debug event (priority 8) Reserved for vendor-specific non-critical events.

The priority associated with the event is hard-coded and cannot be changed. The reporting mechanism for each priority could be changed from the default reporting mechanism (Table 1) by using docsDevEvReporting object in cable device MIB [RFC2669].

Table 7-2 Default event priorities for the Cable Modem device

Event Priority	Local-volatile	Trap	Syslog	Note
1 Emergency	Yes	No	No	Vendor-specific
2 Alert	Yes	No	No	DOCSIS
3 Critical	Yes	No	No	DOCSIS
4 Error	Yes	Yes	Yes	DOCSIS
5 Warning	Yes	No	No	DOCSIS
6 Notice	Yes	Yes	Yes	DOCSIS
7 Informational	No	No	No	DOCSIS/Vendor-specific
8 Debug	No	No	No	Vendor-specific

Use of the local-nonvolatile logging option is at the discretion of the vendor, but when implemented MUST be accompanied by a local-volatile log. Notifications for standard DOCSIS events generated by the CM MUST be in the format specified in Annex D.

7.4.2.4 Standard DOCSIS events for CMTSes

CMTSes use the same levels of the event priorities as CMs; however, the severity definition of the events is different. Events with a severity level of Warning and less specify problems that could affect individual users (for example, an individual CM registration problem).

A severity level of ‘Error’ indicates problems with a group of CMs (for example, CMs that share the same upstream channel).

A severity level of ‘Critical’ indicates a problem that affects the operation of the whole cable system, but is not a faulty condition of the CMTS device. In all these cases, the CMTS MUST be able to send a SYSLOG event and/or an SNMP TRAP to the NMS.

A severity level of ‘Emergency’ is vendor-specific and indicates problems with the CMTS hardware or software which prevent CMTS operation.

Table 7-3 Default Event priorities for the CMTS Device

Event Priority	Local-volatile	Trap	Syslog	Note
1 Emergency	Yes	No	No	Vendor-specific
2 Alert	Yes	No	No	Vendor-specific
3 Critical	Yes	Yes	Yes	DOCSIS
4 Error	Yes	Yes	Yes	DOCSIS
5 Warning	Yes	Yes	Yes	DOCSIS
6 Notice	Yes	Yes	Yes	DOCSIS
7 Informational	No	No	No	DOCSIS/Vendor-specific
8 Debug	No	No	No	Vendor-specific

Use of the local-nonvolatile logging option is at the discretion of the vendor, but when implemented MUST be accompanied by a local-volatile log.

Notifications for standard DOCSIS events generated by the CMTS MUST be in the format specified in Annex D.

7.4.3 Throttling, limiting and priority for event, trap and Syslog

7.4.3.1 Trap and Syslog throttling, trap and Syslog limiting

DOCSIS 2.0-compliant cable devices (CMTS or CM) MUST support SNMP TRAP/INFORM and SYSLOG throttling and limiting as described in RFC 2669, regardless of SNMP mode.

7.4.3.2 Maximum priorities for event reporting

Table 7-2 and Table 7-3 define the default required event reporting capacity for events with different priorities for CMs and CMTSes. This capacity can be considered the minimum requirement for vendors to implement. Vendors may choose to report an event with more mechanisms than required in these tables. According to the priority definitions, there is a maximum level at which an event can be reported. Table 7-4 shows that maximum level for CM events and Table 7-5 displays that for CMTS events.

Vendor-specific priorities can be handled as each vendor sees fit.

Table 7-4 Maximum Level of Support for CM Events

Event Priority	Local-volatile	SNMP Trap	SYSLOG	Note
1 Emergency				Vendor-specific
2 Alert	Yes			DOCSIS
3 Critical	Yes			DOCSIS
4 Error	Yes	Yes	Yes	DOCSIS
5 Warning	Yes	Yes	Yes	DOCSIS
6 Notice	Yes	Yes	Yes	DOCSIS
7 Informational	Yes	Yes	Yes	DOCSIS/Vendor-specific
8 Debug	Yes	Yes	Yes	Vendor-specific

Table 7-5 Maximum Level of Support for CMTS Events

Event Priority	Local-volatile	SNMP Trap	SYSLOG	Note
1 Emergency				Vendor-specific
2 Alert				Vendor-specific
3 Critical	Yes	Yes	Yes	DOCSIS
4 Error	Yes	Yes	Yes	DOCSIS
5 Warning	Yes	Yes	Yes	DOCSIS
6 Notice	Yes	Yes	Yes	DOCSIS
7 Informational	Yes	Yes	Yes	DOCSIS/Vendor-specific
8 Debug				Vendor-specific

7.4.3.3 BIT Values for docsDevEvReporting (RFC 2669)

Permissible BIT values for RFC 2669 docsDevEvReporting objects include:

- 1: local-nonvolatile(0)
- 2: traps(1)
- 3: syslog(2)
- 4: local-volatile(3)

Note: To maintain compatibility with the meaning of default values used by DOCSIS 1.0, the term local-nonvolatile should be interpreted as (local-nonvolatile and volatile). The term local-volatile should be interpreted to mean local-volatile only. If a vendor implements the local-nonvolatile log, the default setting of the local-nonvolatile logging bit in the docsDevEvReporting object is at the discretion of the vendor.

An event reported by trap or syslog or local non-volatile MUST be accompanied by a local-volatile log. The following BITS type values for RFC2669 object docsDevEvReporting MUST NOT be accepted:

- 0x20 = syslog only
- 0x40 = trap only
- 0x60 = trap and syslog

Note that the lower nibble MUST be zero in all cases, resulting in thirteen acceptable values.

SNMP SET requests for unacceptable values MUST result in a 'Wrong Value' error for SNMPv2c/v3 PDUs or a 'Bad Value' error for SNMPv1 PDUs.

A device possessing only non-volatile memory can accept the local-volatile + local-nonvolatile mapping, since active functionality will be identical.

7.4.4 Non-SNMP Fault Management Protocols

The OSS can use a variety of tools and techniques to examine faults at multiple layers. For the IP layer, useful non-SNMP based tools include ping (ICMP Echo and Echo Reply), traceroute (UDP and various ICMP Destination Unreachable flavors). Pings to a CM from its CMCI side MUST be supported to enable local connectivity testing from a customer's PC to the modem. The CM and CMTS MUST support IP end-station generation of ICMP error messages and processing of all ICMP messages.

7.5 Performance management

At the CATV MAC and PHY layers, performance management focuses on the monitoring of the effectiveness of cable plant segmentation and rates of upstream traffic and collisions. Instrumentation is provided in the form of the standard interface statistics [RFC 2863], as well as the docsifCmtsServiceTable and docsifCmServiceTable entries. It is not anticipated that the CMTS upstream bandwidth allocation function will require active network management intervention and tuning.

At the LLC layer, the performance management focus is on bridge traffic management. The CM and CMTS (if the CMTS implements transparent bridging) MUST implement the Bridge MIB RFC 1493, including the dot1dBase and dot1dTp groups. The CM and CMTS MUST implement a managed object that controls whether the 802.1d spanning tree protocol (STP) is run and topology update messages are generated; STP is unnecessary in hierarchical, loop-free topologies. If the STP is enabled for the CM/CMTS, then the CM/CMTS MUST implement the dot1dStp group. These MIB groups' objects allow the NMS to detect when bridge forwarding tables are full, and enable the NMS to modify aging timers.

A final performance concern is the ability to diagnose unidirectional loss. Both the CM and CMTS MUST implement the MIB-2 [RFC 2863] Interfaces group. When there exists more than one upstream or downstream channel, the CM/CMTS MUST implement an instance of IfEntry for each channel. The ifStack group [RFC 2863] MUST be used to define the relationships among the CATV MAC interfaces and their channels.

7.5.1 Additional MIB implementation requirements

To support performance monitoring and data collection for capacity, fault, and performance management, CMs and CMTSes MUST support MIB objects with:

- Accurate in measurement
- Counter properly working (i.e. counter roll over at maximum)
- Correct counter capacity
- Counter reset properly
- Update rate of 1 second

7.6 Coexistence

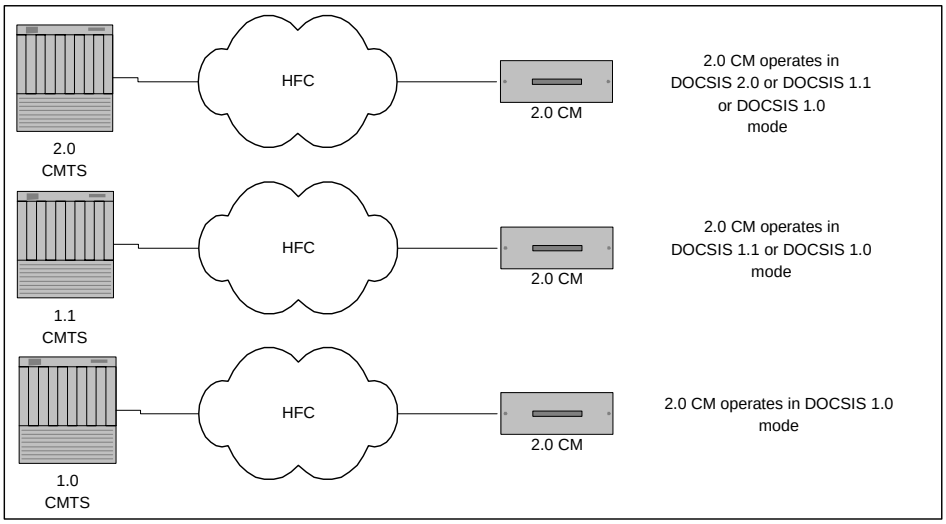


Figure 7-6 Coexistence (DOCSIS 1.0 mode vs. DOCSIS 1.1 mode vs. DOCSIS 2.0 mode)

When a DOCSIS 2.0-compliant CM is connected to a 2.0 CMTS, it can operate in DOCSIS 2.0, 1.1, or 1.0 mode.

When a DOCSIS 2.0-compliant CM is connected to a 1.1 CMTS, it can operate in either DOCSIS 1.1 or 1.0 mode.

When a DOCSIS 2.0-compliant CM is connected to a 1.0 CMTS, it operates in DOCSIS 1.0 mode.

Refer to [DOCSIS 5] and the BPI+ specification for more detailed descriptions of features available for DOCSIS 2.0-compliant CM operating modes.

7.6.1 Coexistence and MIBs

The following table summarizes the requirements for MIB support for a DOCSIS 2.0 CM operating in DOCSIS 2.0, 1.1, or 1.0 mode.

The table also addresses the cases where different sections of a MIB have different support requirements across CM operational modes.

After registration

Table 7-6 DOCSIS 2.0 CM Modes and MIB Requirements

SNMP version:	v1/v2c RO	v1/v2c	v1/v2c/v3	v1/v2c	v1/v2c/v3	v1/v2c	v1/v2c/v3
2.0 CM	**Before Registration (RO from CMCI only)**	**Docsis 1.0 Mode + NmAccess**	**Docsis 1.0 Mode + SNMP Coexistence**	**Docsis 1.1 Mode + NmAccess**	**Docsis 1.1 Mode + SNMP Coexistence**	**Docsis 2.0 Mode + NmAccess**	**Docsis 2.0 Mode + SNMP Coexistence**
Accessible MIBs	RFI-MIB-IPCDN-DRAFT IF-EXT (9) RFC 2863 RFC 1493 RFC 2669(1) RFC 2011(1) RFC 2013 RFC 1907 RFC 2665 RFC 2933(1,7) RFC 3083 QOS BPI+ USB TRAP(2)	RFI-MIB-IPCDN-DRAFT RFC2863 RFC1493 RFC 2669 RFC 2011 RFC 2013 RFC 1907 RFC 2665 RFC 2933(7) RFC 3083 BPI+ (6) USB TRAP(2)	RFI-MIB-IPCDN-DRAFT RFC 2863 RFC 1493 RFC 2669(1) RFC 2011 RFC 2013 RFC 1907 RFC 2665 RFC 2933(7) RFC 3083 BPI+ (6) USB RFC 2786 RFC 2571 RFC 2572 RFC 2573 RFC 2574 RFC 2575 RFC 2576 TRAP(2)	RFI-MIB-IPCDN-DRAFT IF-EXT (9) RFC 2863 RFC 1493 RFC 2669 RFC 2011 RFC 2013 RFC 1907 RFC 2665 RFC 2933 QOS BPI+ USB TRAP	RFI-MIB-IPCDN-DRAFT IF-EXT (9) RFC 2863 RFC 1493 RFC 2669(1) RFC 2011 RFC 2013 RFC 1907 RFC 2665 RFC 2933 QOS BPI+ RFC 2786 RFC 2571 RFC 2572 RFC 2573 RFC 2574 RFC 2575 RFC 2576 USB TRAP	RFI-MIB-IPCDN-DRAFT IF-EXT (9) RFC 2863 RFC 1493 RFC 2669 RFC 2011 RFC 2013 RFC 1907 RFC 2665 RFC 2933 QOS BPI+ USB TRAP	RFI-MIB-IPCDN-DRAFT IF-EXT (9) RFC 2863 RFC 1493 RFC 2669 (1) RFC 2011 RFC 2013 RFC 1907 RFC 2665 RFC 2933 QOS BPI+ RFC 2786 RFC 2571 RFC 2572 RFC 2573 RFC 2574 RFC 2575 RFC 2576 USB TRAP

Table 7-6 DOCSIS 2.0 CM Modes and MIB Requirements (Continued)

	After registration						
SNMP version:	v1/v2c RO	v1/v2c	v1/v2c/v3	v1/v2c	v1/v2c/v3	v1/v2c	v1/v2c/v3
2.0 CM	**Before Registration (RO from CMCI only)**	**Docsis 1.0 Mode + NmAccess**	**Docsis 1.0 Mode + SNMP Coexistence**	**Docsis 1.1 Mode + NmAccess**	**Docsis 1.1 Mode + SNMP Coexistence**	**Docsis 2.0 Mode + NmAccess**	**Docsis 2.0 Mode + SNMP Coexistence**
Inaccessible MIBs	RFC 2669 NmAccessTable RFC 2669 CpeTable RFC 2011 ipAddrTable (8) RFC 2011 ipNetToMedia (8) RFC 2933 InterfaceQuerier (8) RFC 2933 IGMP CacheTable (8) RFC 2786 RFC 2571 RFC 2572 RFC 2573 RFC 2574 RFC 2575 RFC 2576	QOS(5) RFC 2786(4) RFC 2571(5) RFC 2572(5) RFC 2573(5) RFC 2574(5) RFC 2575(5) RFC 2576(5)	RFC 2669 NmAccessTable (3) QOS(5)	RFC 2786(4) RFC 2571(5) RFC 2572(5) RFC 2573(5) RFC 2574(5) RFC 2575(5) RFC 2576(5) RFC 3083(5)	RFC 2669 NmAccessTable(3) RFC 3083	RFC 2786(4) RFC 2571(5) RFC 2572(5) RFC 2573(5) RFC 2574(5) RFC 2575(5) RFC 2576(5) RFC 3083(5)	RFC 2669 NmAccessTable(3) RFC 3083

Notes to Table 7-6:

1. Part of a mib is not accessible. See Inaccessible section for inaccessible objects.
2. Supporting this MIB is optional. When DOCSIS 2.0 CM operates at 1.0 mode, it MAY (optionally) support DOCS_CABLE-DEVICE-TRAP-MIB. Some of the traps will not be applicable. Please see Appendix A for details.
3. RFC 2669 - When CM is in SNMP Coexistence mode, the CM MUST respond with “NoSuchName” or corresponding SNMPv2c error code “NoAccess” for all requests to tables and objects in docsDevNmAccessTable.
4. RFC 2786 - When CM is in SNMP V1/V2c NmAccess mode, CM MUST respond with “NoSuchName” or corresponding SNMPv2c error code “NoAccess” for all requests to tables and objects in this MIB.
5. When CM is in SNMP v1/v2c NmAccess mode, CM MUST respond with “NoSuchName” or corresponding SNMPv2c error code “NoAccess” for all requests to tables and objects defined.
6. BPI+ MIB. Part of the BPI+ MIB MUST be supported to enable secure software download. Refer to Appendix A for specific MIB object requirements. For all other objects in the BPI+ MIB CM MUST respond with “NoSuchName” or corresponding SNMPv2c error code “NoAccess” for all requests.
7. Supporting this MIB is optional. If CM in 1.0 mode supports IGMP, it must implement RFC 2933.
8. Access to this object SHOULD be prohibited as it contains IP address object(s). Refer to Section 5.2 for more details.
9. This MIB has been deprecated and MUST NOT be supported.[1]

1. table modified and note added per ossi2-n-02016, 05/15/02, ab

The following MIB Identities are used in Table 7-6:

RFC 1493 - Bridge
RFC 1907 - SNMP/System
RFC 2011 - IP/ICMP
RFC 2013 - UDP
RFC 2571 - SNMP Engine
RFC 2572 - SNMP MPD
RFC 2573 - SNMPTarget/Notification
RFC 2574 - USM
RFC 2575 - VACM
RFC 2576 - Community
RFC 2665 - Ethernet
RFC 2669 - Cable Device
RFC 2786 - Diffie-Helman
RFC 2863 - Interfaces
RFC 2933 - IGMP
RFC 3083 - BPI
BPI+ - Baseline Privacy Plus
IF-EXT - DOCS-IF-EXT-MIB (see Annex G)
QoS - Quality of Service
RFI-MIB-IPCDN-DRAFT - Supercedes RFC 2670 draft for DOCSIS 2.0 DOCSIS device.
Trap - DOCS-CABLE-DEVICE-TRAP-MIB (See Annex H)
USB - USB draft

7.6.2 Coexistence and SNMP

A DOCSIS 2.0-compliant CM MUST support SNMPv3 and SNMPv1/v2c functionality as specified in Section 5 regardless of what mode (DOCSIS 1.0, 1.1, or 2.0) the CM operates in.

This page intentionally left blank.

8 OSS for BPI+

This section provides the requirements, guidelines, and/or examples related to the Digital Certificate management process and policy.

8.1 DOCSIS Root CA

The DOCSIS Root CA issues two kinds of digital certificates as specified by the BPI+ specification. One is the Manufacturer CA Certificate embedded in the DOCSIS 2.0-compliant CM and verified by the CMTS in order to authenticate the CM during the CM initialization when the CM is provisioned to enable BPI+. The other is the Manufacturer Code Verification Certificate (CVC) embedded in the CM Code File and verified by the CM in order to authenticate the CM Code File during Secure Software Downloading regardless of whether the BPI+ is provisioned or not.

The legitimate DOCSIS Root CA Certificate needs to be delivered to cable operators and/or CMTS vendors because the legitimate DOCSIS Root CA Certificate MUST be provisioned in the CMTS in order to realize the correct CM Authentication. The legitimate DOCSIS Root CA Certificate also needs to be delivered to CM vendors because the legitimate DOCSIS Root CA Public Key extracted from the legitimate DOCSIS Root CA Certificate MUST be embedded in the CM in order for the CM to verify the CVC in the CM Code File. Since the DOCSIS Root CA Certificate is not a secret, the DOCSIS Root CA MAY disclose the DOCSIS Root CA Certificate to any organization including cable operators, CMTS vendors, and CM vendors.

8.2 Digital Certificate Validity Period and Re-issuance

8.2.1 DOCSIS Root CA Certificate

The validity period of the DOCSIS Root CA Certificate is 30 years. The re-issuance process is TBD.

8.2.2 DOCSIS Manufacturer CA Certificate

When the DOCSIS Root CA newly issues the DOCSIS Manufacturer CA Certificate, the following conditions apply:

- the tbsCertificate.validity.notBefore MUST be the actual issuance date and time
- tbsCertificate.validity.notAfter MUST be the actual issuance date and time plus 20 years

Before the DOCSIS Manufacturer CA Certificate expires, a certificate with the same information except the tbsCertificate.validity.notAfter and tbsCertificate.serialNumber needs to be re-issued. DOCSIS 2.0-compliant CM vendors MUST obtain the re-issued DOCSIS Manufacturer CA Certificate from the DOCSIS Root CA at least two years before the tbsCertificate.validity.notAfter value of the current DOCSIS Manufacturer CA Certificate.

When the DOCSIS Root CA re-issues the DOCSIS Manufacturer CA Certificate, the following attribute values MUST be the same as the current DOCSIS Manufacturer CA Certificate:

- tbsCertificate.issuer
- tbsCertificate.subject
- tbsCertificate.subjectPublicKeyInfo

As well, the tbsCertificate.validity.notAfter value MUST be the actual re-issuance date and time plus 20 years.

8.2.3 DOCSIS CM Certificate

The requirements for the DOCSIS CM Certificate including the validity period are specified by the BPI+ specification.

8.2.4 DOCSIS Code Verification Certificate

When the DOCSIS Root CA initially issues the DOCSIS Manufacturer Code Verification Certificate (CVC), the following conditions apply:

- the tbsCertificate.validity.notBefore value MUST be the actual issuance date and time
- the tbsCertificate.validity.notAfter value MUST be the actual issuance date and time plus 2 years

Before the DOCSIS Manufacturer CVC expires, the certificate with the same information except the tbsCertificate.validity.notBefore, the tbsCertificate.validity.notAfter and tbsCertificate.serialNumber needs to be re-issued. DOCSIS 2.0-compliant CM vendors MUST obtain the re-issued DOCSIS Manufacturer CVC from the DOCSIS Root CA at least 6 months before the tbsCertificate.validity.notAfter value of the current DOCSIS Manufacturer CVC.

When the DOCSIS Root CA re-issues the DOCSIS Manufacturer CVC, the following attribute values MUST be the same as the current DOCSIS Manufacturer CVC:

- tbsCertificate.issuer
- tbsCertificate.subject
- tbsCertificate.subjectPublicKeyInfo

As well, the tbsCertificate.validity.notBefore value MUST be between the tbsCertificate.validity.notBefore value of the current DOCSIS Manufacturer CVC, and the actual issuance date and time. In addition, the tbsCertificate.validity.notAfter value MUST be the actual re-issuance date and time plus 2 years.

8.3 CM Code File Signing Policy

CM vendors and cable operators can control the Secure Software Download process based on their policies by updating the Manufacturer/Co-Signer CVC or by changing the signingTime in the Manufacturer/Co-Signer CVS (Code Verification Signature). At this time, the DOCSIS 2.0 specifications do not specify the policy related to the CM Code File signing process. However, an example of the policy is specified in this section.

8.3.1 Manufacturer CM Code File Signing Policy

A DOCSIS 2.0-compliant CM vendor and its Manufacturer Code Signing Agent (Mfg CSA), which securely stores the RSA private key corresponding to the RSA public key in the Manufacturer CVC and generates the CVS for the CM Code File, MAY employ the following policy for the CM Code File signing process.

The Mfg CSA continues to put exactly the same date and time value (T1) in the signingTime field in the Mfg CVS of the CM Code File as long as the vendor does not have any CM Code File to revoke.

Once the vendor realizes there are certain issues in one or more CM Code File(s) and wants to revoke them, the vendor chooses the current date and time value (T2) and starts using T2 as the signingTime value in the Mfg CVS for all the newly created CM Code File from that point. In addition, it re-signs all the still-good old CM Code Files using the T2.

Under this policy, because the multiple CM Code Files make a group of the CM Code Files with the exact same signingTime value in the Msg CVS, the operator can download any CM Code File in the group in any order. That is, among the CM Code Files in the same group, the CM's software can be downgraded if necessary.

This page intentionally left blank.

9 OSSI for CMCI

This section defines the operational mechanisms needed to support the transmission of data over cable services between a cable modem and customer premise equipment. More specifically, this section outlines the following:

- SNMP access via CMCI
- Console Access
- CM diagnostic capabilities
- Protocol Filtering
- Required MIBs

Currently, the CMCI is categorized as internal, external, and CPE-Controlled cable modem functional reference models. The external cable modems MAY have either an Ethernet 10Base-T, a Universal Serial Bus (USB) CMCI interface, or both. If both interfaces are present on a CM, they MAY be active at the same time.

Internal cable modems MUST utilize the Peripheral Component Interconnect (PCI) bus for transparent bi-directional IP traffic forwarding. The PCI interface MUST be defined and accessible from an SNMP manager for both operational and security purposes.

A CPE-Controlled Cable modem's (CCCM) CMCI MAY be either a Peripheral Component Interconnect (PCI) or Universal Serial Bus (USB) interface. If PCI is utilized, the interface MUST be defined and accessible from an SNMP manager for both operational and security purposes.

9.1 SNMP Access via CMCI

A CM device providing CPE SNMP access, prior to completing of the CMTS registration process, MUST comply with the SNMP access requirement specified in Section 5.2.

9.2 Console Access

An external cable modem MUST NOT allow access to the CM functions via a console port. In this specification, a console port is defined as a communication path, either hardware or software, that allows a user to issue commands to modify the configuration or operational status of the CM. Access to the external CM MUST only be allowed using DOCSIS 2.0-defined RF interfaces and operator-controlled SNMP access via the CMCI.

9.3 CM Diagnostic Capabilities

The cable modem MAY have read-only diagnostic interfaces for debugging and troubleshooting purposes. The read-only diagnostic interface MUST NOT display any network addressing or operational information.

9.4 Protocol Filtering

The CM MUST be capable of filtering all broadcast traffic from the host CPE, with the exception of DHCP and ARP packets. This filtering function must adhere to Section 7.3, "Protocol Filters," on page 61. All ICMP type packets MUST be forwarded from the CMCI interface to the RF upstream interface. The CMCI MUST also adhere to the data forwarding rules defined in [DOCSIS 5].

9.5 Management Information Base (MIB) Requirements

All Cable Modems MUST implement the MIBs detailed in Section 6 of this specification, with the following exceptions:

- An external CM with only USB interface(s) MUST NOT implement RFC 2665, the Ethernet Interface MIB.
- An external CM with only USB interface(s) MUST implement the IETF Proposed Standard RFC version of the USB MIB.
- An internal CM MAY implement RFC 2665, the Ethernet Interface MIB.

Annex A Detailed MIB Requirements (normative)[1]

The following abbreviations and rules apply in this Annex:

ACC-FN Accessible for Notify.

ATRAP Accessible through SNMP trap.

D Deprecated. Deprecated objects are optional. That is, a vendor can choose to implement or not implement the object. If a vendor chooses to implement the object, the object MUST be implemented correctly according to the MIB definition. If a vendor chooses not to implement the object, an agent MUST NOT instantiate such object and MUST respond with the appropriate error/exception condition (e.g., no such object for SNMPv2c).

M Mandatory. The object MUST be implemented correctly according to the MIB definition.

N-Acc Not accessible. The object is not accessible and is usually an index in a table.

NA Not Applicable. Not applicable to the device.

N-Sup MUST not support. The device MUST NOT support the object. That is, an agent MUST NOT instantiate such object and MUST respond with the appropriate error/exception condition (e.g., no such object for SNMPv2c).

O Optional. A vendor can choose to implement or not implement the object. If a vendor chooses to implement the object, the object MUST be implemented correctly according to the MIB definition. If a vendor chooses not to implement the object, an agent MUST NOT instantiate such object and MUST respond with the appropriate error/exception condition (e.g., no such object for SNMPv2c).

Ob Obsolete. It is optional. Though in SNMP convention, obsolete objects should not be implemented, DOCSIS 2.0 OSSI lets vendors choose whether or not to support the obsolete object. That is, a vendor can choose to implement or not implement the object. If a vendor chooses to implement the object, the object MUST be implemented correctly according to the MIB definition. If a vendor chooses not to implement the object, the SNMP agent MUST NOT instantiate such object and MUST respond with the appropriate error/exception condition (e.g., no such object for SNMPv2c).

RC Read-Create. The access of the object MUST be implemented as Read-Create.

RO Read-Only. The access of the object MUST be implemented as Read-Only.

RW Read-Write. The access of the object MUST be implemented as Read-Write.

RC/RO Read-Create or Read-Only. The access of the object MUST be implemented as either Read-Create or Read-Only as described in the MIB definition.

RW/RO Read-Write or Read-Only. The access of the object MUST be implemented as either Read-Write or Read-Only as described in the MIB definition.

---

1. DOCS-IF-EXT-MIB, docsIfDocsisBaseCapability, docsIfUpstreamChannelTable, and docsIfCmStatusTable changed per ossi2-n-02016, 05/15/02, ab.
draft-ietf-ipcdn-docs-rfmibv2-01.txt changed to -04.txt per oss2-n-02107, 06/05/02, ab.

DOCS-IF-MIB (RFI-MIB-IPCDN-DRAFT: draft-ietf-ipcdn-docs-rfmibv2-04.txt)								
docsIfDownstreamChannelTable								
Object	**2.0 CM in 1.0 mode**	**Access**	**2.0 CM in 1.1 mode**	**Access**	**2.0 CM in 2.0 mode**	**Access**	**CMTS**	**Access**
docsIfDownChannelId	M	RO	M	RO	M	RO	M	RO
docsIfDownChannelFrequency	M	RO	M	RO	M	RO	M	RW/RO
docsIfDownChannelWidth	M	RO	M	RO	M	RO	M	RW/RO
docsIfDownChannelModulation	M	RO	M	RO	M	RO	M	RW
docsIfDownChannelInterleave	M	RO	M	RO	M	RO	M	RW
docsIfDownChannelPower	M	RO	M	RO	M	RO	M	RW/RO
docsIfDownChannelAnnex	O	RO	O	RO	M	RO	M	RW/RO
docsIfUpstreamChannelTable								
Object	**2.0 CM in 1.0 mode**	**Access**	**2.0 CM in 1.1 mode**	**Access**	**2.0 CM in 2.0 mode**	**Access**	**CMTS**	**Access**
docsIfUpChannelId	M	RO	M	RO	M	RO	M	RO
docsIfUpChannelFrequency	M	RO	M	RO	M	RO	M	RC
docsIfUpChannelWidth	M	RO	M	RO	M	RO	M	RC
docsIfUpChannelModulationProfile	M	RO	M	RO	M	RO	M	RC
docsIfUpChannelSlotSize	M	RO	M	RO	M	RO	M	RC/RO
docsIfUpChannelTxTimingOffset	M	RO	M	RO	M	RO	M	RO
docsIfUpChannelRangingBackoffStart	M	RO	M	RO	M	RO	M	RC
docsIfUpChannelRangingBackoffEnd	M	RO	M	RO	M	RO	M	RC
docsIfUpChannelTxBackoffStart	M	RO	M	RO	M	RO	M	RC
docsIfUpChannelTxBackoffEnd	M	RO	M	RO	M	RO	M	RC
docsIfUpChannelScdmaActiveCodes	O	RO	O	RO	M	RO	M	RC
docsIfUpChannelScdmaCodesPerSlot	O	RO	O	RO	M	RO	M	RC
docsIfUpChannelScdmaFrameSize	O	RO	O	RO	M	RO	M	RC
docsIfUpChannelScdmaHoppingSeed	O	RO	O	RO	M	RO	M	RC
docsIfUpChannelType	O	RO	O	RO	M	RO	M	RC
docsIfUpChannelCloneFrom	O	RO	O	RO	M	RO	M	RC
docsIfUpChannelUpdate	O	RO	O	RO	M	RO	M	RC

docsIfUpChannelStatus	O	RO	O	RO	M	RO	M	RC
docsIfQosProfileTable								
Object	**2.0 CM in 1.0 mode**	**Access**	**2.0 CM in 1.1 mode**	**Access**	**2.0 CM in 2.0 mode**	**Access**	**CMTS**	**Access**
docsIfQosProfIndex	M	N-Acc	O	N-Acc	O	N-Acc	O	N-Acc
docsIfQosProfPriority	M	RO	O	RO	O	RO	O	RC/RO
docsIfQosProfMaxUpBandwidth	M	RO	O	RO	O	RO	O	RC/RO
docsIfQosProfGuarUpBandwidth	M	RO	O	RO	O	RO	O	RC/RO
docsIfQosProfMaxDownBandwidth	M	RO	O	RO	O	RO	O	RC/RO
docsIfQosProfMaxTxBurst	D	RO	D	RO	D	RO	D	RC/RO
docsIfQosProfBaselinePrivacy	M	RO	O	RO	O	RO	O	RC/RO
docsIfQosProfStatus	M	RO	O	RO	O	RO	O	RC/RO
docsIfQosProfMaxTransmitBurst	M	RO	O	RO	O	RO	O	RC/RO
docsIfSignalQualityTable								
Object					**CM**	**Access**	**CMTS**	**Access**
docsIfSigQIncludesContention					M	RO	M	RO
docsIfSigQUnerroreds					M	RO	M	RO
docsIfSigQCorrecteds					M	RO	M	RO
docsIfSigQUncorrectables					M	RO	M	RO
docsIfSigQSignalNoise					M	RO	M	RO
docsIfSigQMicroreflections					M	RO	M	RO
docsIfSigQEqualizationData					M	RO	M	RO
(no table)								
Object	**2.0 CM in 1.0 mode**	**Access**	**2.0 CM in 1.1 mode**	**Access**	**2.0 CM in 2.0 mode**	**Access**	**CMTS**	**Access**
docsIfDocsisBaseCapability	O	RO	M	RO	M	RO	M	RO
docsIfCmMacTable								
Object					**CM**	**Access**	**CMTS**	**Access**
docsIfCmCmtsAddress					M	RO	NA	NA
docsIfCmCapabilities					M	RO	NA	NA
docsIfCmRangingTimeout					Ob	N-Sup	NA	NA

docsIfCmRangingTimeout					M	RW	NA	NA
docsIfCmStatusTable								
Object	**2.0 CM in 1.0 mode**	**Access**	**2.0 CM in 1.1 mode**	**Access**	**2.0 CM in 2.0 mode**	**Access**	**CMTS**	**Access**
docsIfCmStatusValue	M	RO	M	RO	M	RO	NA	NA
docsIfCmStatusCode	M	RO	M	RO	M	RO	NA	NA
docsIfCmStatusTxPower	M	RO	M	RO	M	RO	NA	NA
docsIfCmStatusResets	M	RO	M	RO	M	RO	NA	NA
docsIfCmStatusLostSyncs	M	RO	M	RO	M	RO	NA	NA
docsIfCmStatusInvalidMaps	M	RO	M	RO	M	RO	NA	NA
docsIfCmStatusInvalidUcds	M	RO	M	RO	M	RO	NA	NA
docsIfCmStatusInvalidRanging Responses	M	RO	M	RO	M	RO	NA	NA
docsIfCmStatusInvalidRegistration Responses	M	RO	M	RO	M	RO	NA	NA
docsIfCmStatusT1Timeouts	M	RO	M	RO	M	RO	NA	NA
docsIfCmStatusT2Timeouts	M	RO	M	RO	M	RO	NA	NA
docsIfCmStatusT3Timeouts	M	RO	M	RO	M	RO	NA	NA
docsIfCmStatusT4Timeouts	M	RO	M	RO	M	RO	NA	NA
docsIfCmStatusRangingAborteds	M	RO	M	RO	M	RO	NA	NA
docsIfCmStatusDocsisOperMode	O	RO	M	RO	M	RO	NA	NA
docsIfCmStatusModulationType	O	RO	M	RO	M	RO	NA	NA
docsIfCmServiceTable								
Object	**2.0 CM in 1.0 mode**	**Access**	**2.0 CM in 1.1 mode**	**Access**	**2.0 CM in 2.0 mode**	**Access**	**CMTS**	**Access**
docsIfCmServiceId	M	N-Acc	M	N-Acc	M	N-Acc	NA	NA
docsIfCmServiceQosProfile	M	RO	M	RO	M	RO	NA	NA
docsIfCmServiceTxSlotsImmed	M	RO	M	RO	M	RO	NA	NA
docsIfCmServiceTxSlotsDed	M	RO	M	RO	M	RO	NA	NA
docsIfCmServiceTxRetries	M	RO	M	RO	M	RO	NA	NA
docsIfCmServiceTxExceeds	M	RO	M	RO	M	RO	NA	NA
docsIfCmServiceRqRetries	M	RO	M	RO	M	RO	NA	NA
docsIfCmServiceRqExceeds	M	RO	M	RO	M	RO	NA	NA
docsIfCmServiceExtTxSlotsImmed	O	RO	O	RO	M	RO	NA	NA

docsIfCmServiceExtTxSlotsDed	O	RO	O	RO	M	RO	NA	NA

docsIfCmtsMacTable

Object	CM	Access	CMTS	Access
docsIfCmtsCapabilities	NA	NA	M	RO
docsIfCmtsSyncInterval	NA	NA	M	RW/RO
docsIfCmtsUcdInterval	NA	NA	M	RW/RO
docsIfCmtsMaxServiceIds	NA	NA	M	RO
docsIfCmtsInsertionInterval	NA	NA	Ob	N-Sup
docsIfCmtsInvitedRangingAttempts	NA	NA	M	RW/RO
docsIfCmtsInsertionInterval	NA	NA	M	RW/RO

docsIfCmtsStatusTable

Object	CM	Access	CMTS	Access
docsIfCmtsStatusInvalidRangeReqs	NA	NA	M	RO
docsIfCmtsStatusRangingAborteds	NA	NA	M	RO
docsIfCmtsStatusInvalidRegReqs	NA	NA	M	RO
docsIfCmtsStatusFailedRegReqs	NA	NA	M	RO
docsIfCmtsStatusInvalidDataReqs	NA	NA	M	RO
docsIfCmtsStatusT5Timeouts	NA	NA	M	RO

docsIfCmtsCmStatusTable

Object	CM	Access	CMTS	Access
docsIfCmtsCmStatusIndex	NA	NA	M	N-Acc
docsIfCmtsCmStatusMacAddress	NA	NA	M	RO
docsIfCmtsCmStatusIpAddress	NA	NA	D	RO
docsIfCmtsCmStatusDownChannelIfIndex	NA	NA	M	RO
docsIfCmtsCmStatusUpChannelIfIndex	NA	NA	M	RO
docsIfCmtsCmStatusRxPower	NA	NA	M	RO
docsIfCmtsCmStatusTimingOffset	NA	NA	M	RO
docsIfCmtsCmStatusEqualizationData	NA	NA	M	RO
docsIfCmtsCmStatusValue	NA	NA	M	RO
docsIfCmtsCmStatusUnerroreds	NA	NA	M	RO
docsIfCmtsCmStatusCorrecteds	NA	NA	M	RO
docsIfCmtsCmStatusUncorrectables	NA	NA	M	RO
docsIfCmtsCmStatusSignalNoise	NA	NA	M	RO

docsIfCmtsCmStatusMicroreflections	NA	NA	M	RO
docsIfCmtsCmStatusExtUnerroreds	NA	NA	M	RO
docsIfCmtsCmStatusExtCorrecteds	NA	NA	M	RO
docsIfCmtsCmStatusExtUncorrectables	NA	NA	M	RO
docsIfCmtsCmStatusDocsisRegMode	NA	NA	M	RO
docsIfCmtsCmStatusModulationType	NA	NA	M	RO
docsIfCmtsCmStatusInetAddressType	NA	NA	M	RO
docsIfCmtsCmStatusInetAddress	NA	NA	M	RO

docsIfCmtsServiceTable

Object	**CM**	**Access**	**CMTS**	**Access**
docsIfCmtsServiceId	NA	NA	M	N-Acc
docsIfCmtsServiceCmStatusIndex	NA	NA	D	RO
docsIfCmtsServiceAdminStatus	NA	NA	M	RW/RO
docsIfCmtsServiceQosProfile	NA	NA	M	RO
docsIfCmtsServiceCreateTime	NA	NA	M	RO
docsIfCmtsServiceInOctets	NA	NA	M	RO
docsIfCmtsServiceInPackets	NA	NA	M	RO
docsIfCmtsServiceNewCmStatusIndex	NA	NA	M	RO

docsIfCmtsModulationTable

Object	**CM**	**Access**	**CMTS**	**Access**
docsIfCmtsModIndex	NA	NA	M	N-Acc
docsIfCmtsModIntervalUsageCode	NA	NA	M	N-Acc
docsIfCmtsModControl	NA	NA	M	RC
docsIfCmtsModType	NA	NA	M	RC
docsIfCmtsModPreambleLen	NA	NA	M	RC
docsIfCmtsModDifferentialEncoding	NA	NA	M	RC
docsIfCmtsModFECErrorCorrection	NA	NA	M	RC
docsIfCmtsModFECCodewordLength	NA	NA	M	RC
docsIfCmtsModScramblerSeed	NA	NA	M	RC
docsIfCmtsModMaxBurstSize	NA	NA	M	RC
docsIfCmtsModGuardTimeSize	NA	NA	M	RO
docsIfCmtsModLastCodewordShortened	NA	NA	M	RC
docsIfCmtsModScrambler	NA	NA	M	RC
docsIfCmtsModByteInterleaverDepth	NA	NA	M	RC

docsIfCmtsModByteInterleaverBlockSize	NA	NA	M	RC
docsIfCmtsModPreambleType	NA	NA	M	RC
docsIfCmtsModTcmErrorCorrectionOn	NA	NA	M	RC
docsIfCmtsModScdmaInterleaverStepSize	NA	NA	M	RC
docsIfCmtsModScdmaSpreaderEnable	NA	NA	M	RO
docsIfCmtsModScdmaSubframeCodes	NA	NA	M	RC
docsIfCmtsModChannelType	NA	NA	M	RC
(no table)				
Object	**CM**	**Access**	**CMTS**	**Access**
docsIfCmtsQosProfilePermissions	NA	NA	M	RW / RO
docsIfCmtsMacToCmTable				
Object	**CM**	**Access**	**CMTS**	**Access**
docsIfCmtsCmMac	NA	NA	M	N-Acc
docsIfCmtsCmPtr	NA	NA	M	RO
IF-MIB (RFC 2863)				
(no table)				
Object	**CM**	**Access**	**CMTS**	**Access**
ifNumber	M	RO	M	RO
IfTableLastChange	M	RO	M	RO
ifTable				
Object	**CM**	**Access**	**CMTS**	**Access**
IfIndex	M	RO	M	RO
ifDescr	M	RO	M	RO
ifType	M	RO	M	RO
ifMtu	M	RO	M	RO
ifSpeed	M	RO	M	RO
ifPhysAddress	M	RO	M	RO
ifAdminStatus	M	RW	M	RW
ifOperStatus	M	RO	M	RO
ifLastChange	M	RO	M	RO

ifInOctets	M	RO	M	RO
ifInUcastPkts	M	RO	M	RO
ifInNUcastPkts	D	RO	D	RO
ifInDiscards	M	RO	M	RO
ifInErrors	M	RO	M	RO
ifInUnknownProtos	M	RO	M	RO
ifOutOctets	M	RO	M	RO
ifOutUcastPkts	M	RO	M	RO
ifOutNUcastPkts	D	RO	D	RO
ifOutDiscards	M	RO	M	RO
ifOutErrors	M	RO	M	RO
ifOutQLen	D	RO	D	RO
ifSpecific	D	RO	D	RO
ifXTable				
Objects	**CM**	**Access**	**CMTS**	**Access**
ifName	M	RO	M	RO
ifInMulticastPkts	M	RO	M	RO
ifInBroadcastPkts	M	RO	M	RO
ifOutMulticastPkts	M	RO	M	RO
ifOutBroadcastPkts	M	RO	M	RO
ifHCInOctets	O	RO	O	RO
ifHCInUcastPkts	O	RO	O	RO
ifHCInMulticastPkts	O	RO	O	RO
ifHCInBroadcastPkts	O	RO	O	RO
ifHCOutOctets	O	RO	O	RO
ifHCOutUcastPkts	O	RO	O	RO
ifHCOutMulticastPkts	O	RO	O	RO
ifHCOutBroadcastPkts	O	RO	O	RO
ifLinkUpDownTrapEnable	M	RW	M	RW
ifHighSpeed	M	RO	M	RO
ifPromiscuousMode	M	RW/RO	M	RW/RO
ifConnectorPresent	M	RO	M	RO
ifAlias	M	RW/RO	M	RW/RO
ifCounterDiscontinuityTime	M	RO	M	RO

ifStackTable

Objects	**CM**	**Access**	**CMTS**	**Access**
ifStackHigherLayer	M	N-Acc	M	N-Acc
ifStackLowerLayer	M	N-Acc	M	N-Acc
ifStackStatus	M	RC/RO	M	RC/RO

(no table)

Object	**CM**	**Access**	**CMTS**	**Access**
ifStackLastChange	M	RO	M	RO

ifRcvAddressTable

Object	**CM**	**Access**	**CMTS**	**Access**
ifRcvAddressAddress	O	N-Acc	O	N-Acc
ifRcvAddressStatus	O	RC	O	RC
IfRcvAddressType	O	RC	O	RC

Notification	**CM**	**Access**	**CMTS**	**Access**
linkUp	M		M	
linkDown	M		M	

ifTestTable

Objects	**CM**	**Access**	**CMTS**	**Access**
ifTestId	O	RW	O	RW
ifTestStatus	O	RW	O	RW
ifTestType	O	RW	O	RW
ifTestResult	O	RO	O	RO
ifTestCode	O	RO	O	RO
ifTestOwner	O	RW	O	RW

BRIDGE-MIB (RFC 1493)

NOTE: Implementation of BRIDGE MIB is required ONLY if device is a bridging device

dot1dBase group

Objects	**CM**	**Access**	**CMTS**	**Access**
dot1dBaseBridgeAddress	M	RO	M	RO

dot1dBaseNumPorts	M	RO	M	RO
dot1dBaseType	M	RO	M	RO
dot1dBasePortTable				
Objects	**CM**	**Access**	**CMTS**	**Access**
dot1dBasePort	M	RO	M	RO
dot1dBasePortIfIndex	M	RO	M	RO
dot1dBasePortCircuit	M	RO	M	RO
dot1dBasePortDelayExceededDiscards	M	RO	M	RO
dot1dBasePortMtuExceededDiscards	M	RO	M	RO
dot1dStp group NOTE: This group is required ONLY if STP is implemented				
Objects	**CM**	**Access**	**CMTS**	**Access**
dot1dStpProtocolSpecification	M	RO	M	RO
dot1dStpPriority	M	RW	M	RW
dot1dStpTimeSinceTopologyChange	M	RO	M	RO
dot1dStpTopChanges	M	RO	M	RO
dot1dStpDesignatedRoot	M	RO	M	RO
dot1dStpRootCost	M	RO	M	RO
dot1dStpRootPort	M	RO	M	RO
dot1dStpMaxAge	M	RO	M	RO
dot1dStpHelloTime	M	RO	M	RO
dot1dStpHoldTime	M	RO	M	RO
dot1dStpForwardDelay	M	RO	M	RO
dot1dStpBridgeMaxAge	M	RW	M	RW
dot1dStpBridgeHelloTime	M	RW	M	RW
dot1dStpBridgeForwardDelay	M	RW	M	RW
dot1dStpPortTable NOTE:This table is required ONLY if STP is implemented				
Objects	**CM**	**Access**	**CMTS**	**Access**
dot1dStpPort	M	RO	M	RO
dot1dStpPortPriority	M	RW	M	RW
dot1dStpPortState	M	RO	M	RO
dot1dStpPortEnable	M	RW	M	RW

dot1dStpPortPathCost	M	RW	M	RW
dot1dStpPortDesignatedRoot	M	RO	M	RO
dot1dStpPortDesignatedCost	M	RO	M	RO
dot1dStpPortDesignatedBridge	M	RO	M	RO
dot1dStpPortDesignatedPort	M	RO	M	RO
dot1dStpPortForwardTransitions	M	RO	M	RO

dot1dTp group

Note: This group is required ONLY if transparent bridging is implemented.

Objects	**CM**	**Access**	**CMTS**	**Access**
dot1dTpLearnedEntryDiscards	M	RO	M	RO
dot1dTpAgingTime	M	RW	M	RW

dot1dTpFdbTable

Objects	**CM**	**Access**	**CMTS**	**Access**
dot1dTpFdbAddress	M	RO	M	RO
dot1dTpFdbPort	M	RO	M	RO
dot1dTpFdbStatus	M	RO	M	RO

dot1dTpPortTable

Objects	**CM**	**Access**	**CMTS**	**Access**
dot1dTpPort	M	RO	M	RO
dot1dTpPortMaxInfo	M	RO	M	RO
dot1dTpPortInFrames	M	RO	M	RO
dot1dTpPortOutFrames	M	RO	M	RO
dot1dTpPortInDiscards	M	RO	M	RO

dot1dStaticTable

Note: Implementation of dot1dStaticTable is OPTIONAL

Objects	**CM**	**Access**	**CMTS**	**Access**
dot1dStaticAddress	O	RW	O	RW
dot1dStaticReceivePort	O	RW	O	RW
dot1dStaticAllowedToGoTo	O	RW	O	RW
dot1dStaticStatus	O	RW	O	RW

DOCS-CABLE-DEVICE-MIB (RFC 2669)				
docsDevBaseGroup				
Objects	**CM**	**Access**	**CMTS**	**Access**
docsDevRole	M	RO	O	RO
docsDevDateTime	M	RW	O	RW
docsDevResetNow	M	RW	O	RW
docsDevSerialNumber	M	RO	O	RO
docsDevSTPControl	M	RW/ RO	O	RW/RO
docsDevNmAccessGroup NOTE: docsDevNmAccessGroup is NOT accessible when the device is in SNMP Coexistence mode.				
docsDevNmAcessTable				
Objects	**CM**	**Access**	**CMTS**	**Access**
docsDevNmAccessIndex	M	N-Acc	O	N-Acc
docsDevNmAccessIp	M	RC	O	RC
docsDevNmAccessIpMask	M	RC	O	RC
docsDevNmAccessCommunity	M	RC	O	RC
docsDevNmAccessControl	M	RC	O	RC
docsDevNmAccessInterfaces	M	RC	O	RC
docsDevNmAccessStatus	M	RC	O	RC
docsDevNmAccessTrapVersion (Note: This object is currently not in RFC 2669)	M	RC	O	RC
docsDevSoftwareGroup				
Objects	**CM**	**Access**	**CMTS**	**Access**
docsDevSwServer	M	RW	O	RW
docsDevSwFilename	M	RW	O	RW
docsDevSwAdminStatus	M	RW	O	RW
docsDevSwOperStatus	M	RO	O	RO
docsDevSwCurrentVers	M	RO	O	RO
docsDevServerGroup				
Objects	**CM**	**Access**	**CMTS**	**Access**

docsDevServerBootState	M	RO	N-Sup	
docsDevServerDhcp	M	RO	N-Sup	
docsDevServerTime	M	RO	N-Sup	
docsDevServerTftp	M	RO	N-Sup	
docsDevServerConfigFile	M	RO	N-Sup	
docsDevEventGroup				
Objects	**CM**	**Access**	**CMTS**	**Access**
docsDevEvControl	M	RW	M	RW
docsDevEvSyslog	M	RW	M	RW
docsDevEvThrottleAdminStatus	M	RW	M	RW
docsDevEvThrottleInhibited	M	RO	M	RO
docsDevEvThrottleThreshold	M	RW	M	RW
docsDevEvThrottleInterval	M	RW	M	RW
docsDevEvControlTable				
Objects	**CM**	**Access**	**CMTS**	**Access**
docsDevEvPriority	M	N-Acc	M	N-Acc
docsDevEvReporting (Mandatory RW by DOCSIS 1.1 and DOCSIS 2.0; exception to RFC 2669)	M	RW	M	RW
docsDevEventTable				
Objects	**CM**	**Access**	**CMTS**	**Access**
docsDevEvIndex	M	N-Acc	M	N-Acc
docsDevEvFirstTime	M	RO	M	RO
docsDevEvLastTime	M	RO	M	RO
docsDevEvCounts	M	RO	M	RO
docsDevEvLevel	M	RO	M	RO
docsDevEvId	M	RO	M	RO
docsDevEvText	M	RO	M	RO
docsDevFilterGroup				
Objects	**CM**	**Access**	**CMTS**	**Access**
docsDevFilterLLCUnmatchedAction	M	RW	O	RW

docsDevFilterLLCTable

Objects	CM	Access	CMTS	Access
docsDevFilterLLCIndex	M	N-Acc	O	N-Acc
docsDevFilterLLCStatus	M	RC	O	RC
docsDevFilterLLCIfIndex	M	RC	O	RC
docsDevFilterLLCProtocolType	M	RC	O	RC
docsDevFilterLLCProtocol	M	RC	O	RC
docsDevFilterLLCMatches	M	RO	O	RO

(no table)

Objects	CM	Access	CMTS	Access
docsDevFilterIpDefault	M	RW	O	RW

docsDevFilterIpTable

Objects	CM	Access	CMTS	Access
docsDevFilterIpIndex	M	N-Acc	O	N-Acc
docsDevFilterIpStatus	M	RC	O	RC
docsDevFilterIpControl	M	RC	O	RC
docsDevFilterIpIfIndex	M	RC	O	RC
docsDevFilterIpDirection	M	RC	O	RC
docsDevFilterIpBroadcast	M	RC	O	RC
docsDevFilterIpSaddr	M	RC	O	RC
docsDevFilterIpSmask	M	RC	O	RC
docsDevFilterIpDaddr	M	RC	O	RC
docsDevFilterIpDmask	M	RC	O	RC
docsDevFilterIpProtocol	M	RC	O	RC
docsDevFilterIpSourcePortLow	M	RC	O	RC
docsDevFilterIpSourcePortHigh	M	RC	O	RC
docsDevFilterIpDestPortLow	M	RC	O	RC
docsDevFilterIpDestPortHigh	M	RC	O	RC
docsDevFilterIpMatches	M	RO	O	RO
docsDevFilterIpTos	M	RC	O	RC
docsDevFilterIpTosMask	M	RC	O	RC
docsDevFilterIpContinue	M	RC	O	RC
docsDevFilterIpPolicyId	M	RC	O	RC

docsDevFilterPolicyTable

Objects	CM	Access	CMTS	Access
docsDevFilterPolicyIndex	M	N-Acc	O	N-Acc
docsDevFilterPolicyId	M	RC	O	RC
docsDevFilterPolicyStatus	M	RC	O	RC
docsDevFilterPolicyPtr	M	RC	O	RC

docsDevFilterTosTable

Objects	CM	Access	CMTS	Access
docsDevFilterTosIndex	M	N-Acc	O	N-Acc
docsDevFilterTosStatus	M	RC	O	RC
docsDevFilterTosAndMask	M	RC	O	RC
docsDevFilterTosOrMask	M	RC	O	RC

docsDevCpeGroup

NOTE: CMs supporting the IP spoofing function MUST implement this group. CMs not supporting the IP spoofing filter MUST NOT implement this group.

Objects	CM	Access	CMTS	Access
docsDevCpeEnroll	O	RW	N-Sup	
docsDevCpeIpMax	O	RW	N-Sup	

docsDevCpeTable

Objects	CM	Access	CMTS	Access
docsDevCpeIp	O	N-Acc	N-Sup	
docsDevCpeSource	O	RO	N-Sup	
docsDevCpeStatus	O	RC	N-Sup	

IP-MIB (RFC 2011)

IP Group

Objects	CM	Access	CMTS	Access
ipForwarding	M	RW	M	RW
ipDefaultTTL	M	RW	M	RW
ipInreceives	M	RO	M	RO

ipInHdrErrors	M	RO	M	RO
ipInAddrErrors	M	RO	M	RO
ipForwDatagrams	M	RO	M	RO
ipinUnknownProtos	M	RO	M	RO
ipInDiscards	M	RO	M	RO
ipInDelivers	M	RO	M	RO
ipOutRequests	M	RO	M	RO
ipOutDiscards	M	RO	M	RO
ipOutNoRoutes	M	RO	M	RO
ipReasmTimeout	M	RO	M	RO
ipReasmReqds	M	RO	M	RO
ipReasmOKs	M	RO	M	RO
ipReasmFails	M	RO	M	RO
ipFragOKs	M	RO	M	RO
ipFragFails	M	RO	M	RO
ipFragCreates	M	RO	M	RO
ipAddrTable				
Objects	**CM**	**Access**	**CMTS**	**Access**
ipAdEntAddr	M	RO	M	RO
ipAdEntIfIndex	M	RO	M	RO
ipAdEntNetMask	M	RO	M	RO
ipAdEntBcastAddr	M	RO	M	RO
ipAdEntReasmMaxSize	M	RO	M	RO
IpNetToMediaTable				
Objects	**CM**	**Access**	**CMTS**	**Access**
ipNetToMediaIfIndex	M	RC	M	RC
ipNetToMediaPhysAddress	M	RC	M	RC
ipNetToMediaNetAddress	M	RC	M	RC
ipNetToMediaType	M	RC	M	RC
(no table)				
Object	**CM**	**Access**	**CMTS**	**Access**
ipRoutingDiscards	M	RO	M	RO

ICMP Group

Objects	CM	Access	CMTS	Access
icmpInMsgs	M	RO	M	RO
icmpInErrors	M	RO	M	RO
icmpInDestUnreachs	M	RO	M	RO
icmpInTimeExcds	M	RO	M	RO
icmpInParmProbs	M	RO	M	RO
icmpInSrcQuenchs	M	RO	M	RO
icmpInRedirects	M	RO	M	RO
icmpInEchos	M	RO	M	RO
icmpInEchosReps	M	RO	M	RO
icmpInTimestamps	M	RO	M	RO
icmpInTimeStampreps	M	RO	M	RO
icmpInAddrMasks	M	RO	M	RO
icmpInAddrMaskReps	M	RO	M	RO
icmpOutMsgs	M	RO	M	RO
icmpOutErrors	M	RO	M	RO
icmpOutDestUnreachs	M	RO	M	RO
icmpOutTimeExcds	M	RO	M	RO
icmpOutParmProbs	M	RO	M	RO
icmpOutSrcQuenchs	M	RO	M	RO
icmpOutRedirects	M	RO	M	RO
icmpOutEchos	M	RO	M	RO
icmpOutEchoReps	M	RO	M	RO
icmpOutTimestamps	M	RO	M	RO
icmpOutTimestampReps	M	RO	M	RO
icmpOutAddrMasks	M	RO	M	RO
icmpOutAddrMaskReps	M	RO	M	RO

UDP-MIB (RFC 2013)

UDP Group

Objects	CM	Access	CMTS	Access
udpInDatagrams	M	RO	M	RO
udpNoPorts	M	RO	M	RO

udpInErrors	M	RO	M	RO
udpOutDatagrams	M	RO	M	RO
UDP Listener Table				
Objects	**CM**	**Access**	**CMTS**	**Access**
udpLocalAddress	M	RO	M	RO
udpLocalPort	M	RO	M	RO
SNMPv2-MIB (RFC 1907)				
System Group				
Objects	**CM**	**Access**	**CMTS**	**Access**
sysDescr	M	RO	M	RO
sysObjectID	M	RO	M	RO
sysUpTime	M	RO	M	RO
sysContact	M	RW	M	RW
sysName	M	RW	M	RW
sysLocation	M	RW	M	RW
sysServices	M	RO	M	RO
sysORLastChange	M	RO	M	RO
sysORTable				
Object	**CM**	**Access**	**CMTS**	**Access**
sysORIndex	M	N-Acc	M	N-Acc
sysORID	M	RO	M	RO
sysORDescr	M	RO	M	RO
sysORUpTime	M	RO	M	RO
SNMP Group				
Objects	**CM**	**Access**	**CMTS**	**Access**
snmpInPkts	M	RO	M	RO
SnmpInBadVersions	M	RO	M	RO
snmpOutPkts	Ob	RO	Ob	RO
snmpInBadCommunityNames	M	RO	M	RO
snmpInBadCommunityUses	M	RO	M	RO
snmpInASNParseErrs	M	RO	M	RO

snmpInTooBigs	Ob	RO	Ob	RO
snmpInNoSuchNames	Ob	RO	Ob	RO
snmpInBadValues	Ob	RO	Ob	RO
snmpInReadOnlys	Ob	RO	Ob	RO
snmpInGenErrs	Ob	RO	Ob	RO
snmpInTotalReqVars	Ob	RO	Ob	RO
snmpInTotalSetVars	Ob	RO	Ob	RO
snmpInGetRequests	Ob	RO	Ob	RO
snmpInGetNexts	Ob	RO	Ob	RO
snmpInSetRequests	Ob	RO	Ob	RO
snmpInGetResponses	Ob	RO	Ob	RO
snmpInTraps	Ob	RO	Ob	RO
snmpOutTooBigs	Ob	RO	Ob	RO
snmpOutNoSuchNames	Ob	RO	Ob	RO
snmpOutBadValues	Ob	RO	Ob	RO
snmpOutGenErrs	Ob	RO	Ob	RO
snmpOutGetRequests	Ob	RO	Ob	RO
snmpOutGetNexts	Ob	RO	Ob	RO
snmpOutSetRequests	Ob	RO	Ob	RO
snmpOutGetResponses	Ob	RO	Ob	RO
snmpOutTraps	Ob	RO	Ob	RO
snmpEnableAuthenTraps	M	RW	M	RW
snmpSilentDrops	M	RO	M	RO
snmpProxyDrops	M	RO	M	RO
snmpSet Group				
Object	**CM**	**Access**	**CMTS**	**Access**
snmpSetSerialNo	M	RW	M	RW
Etherlike-MIB (RFC 2665)				
dot3StatsTable				
Objects	**CM**	**Access**	**CMTS**	**Access**
dot3StatsIndex	M	RO	M	RO
dot3StatsAlignmentErrors	M	RO	M	RO
dot3StatsFCSErrors	M	RO	M	RO

dot3StatsSingleCollisionFrames	M	RO	M	RO
dot3StatsMultipleCollisionFrames	M	RO	M	RO
dot3StatsSQETestErrors	M	RO	M	RO
dot3StatsDeferredTransmissions	M	RO	M	RO
dot3StatsLateCollisions	M	RO	M	RO
dot3StatsExcessiveCollisions	M	RO	M	RO
dot3StatsInternalMacTransmitErrors	M	RO	M	RO
dot3StatsCarrierSenseErrors	M	RO	M	RO
dot3StatsFrameTooLongs	M	RO	M	RO
dot3StatsInternalMacReceiveErrors	M	RO	M	RO
dot3StatsEtherChipSet	D	RO	D	RO
dot3StatsSymbolErrors	M	RO	M	RO
dot3StatsDuplexStatus	M	RO	M	RO
dot3CollTable				
Objects	**CM**	**Access**	**CMTS**	**Access**
dot3CollCount	O	NA	O	NA
dot3CollFrequencies	O	RO	O	RO
dot3ControlTable				
Objects	**CM**	**Access**	**CMTS**	**Access**
dot3ControlFunctionsSupported	O	RO	O	RO
dot3ControlInUnknownOpcodes	O	RO	O	RO
dot3PauseTable				
Objects	**CM**	**Access**	**CMTS**	**Access**
dot3PauseAdminMode	O	RW	O	RW
dot3PauseOperMode	O	RO	O	RO
dot3InPauseFrames	O	RO	O	RO
dot3OutPauseFrames	O	RO	O	RO
USB MIB				
NOTE: This MIB is required for CMs that support USB only.				
Object	**CM**	**Access**	**CMTS**	**Access**
usbNumber	M	RO	NA	

usbPortTable

Object	CM	Access	CMTS	Access
usbPortIndex	M	RO	NA	
usbPortType	M	RO	NA	
usbPortRate	M	RO	NA	

usbDeviceTable

Object	CM	Access	CMTS	Access
usbDeviceIndex	M	RO	NA	
usbDevicePower	M	RO	NA	
usbDeviceVendorID	M	RO	NA	
usbDeviceProductID	M	RO	NA	
usbDeviceNumberConfigurations	M	RO	NA	
usbDeviceActiveClass	M	RO	NA	
usbDeviceStatus	M	RO	NA	
usbDeviceEnumCounter	M	RO	NA	
usbDeviceRemoteWakeup	M	RO	NA	
usbDeviceRemoteWakeupOn	M	RO	NA	

usbCDCTable

Object	CM	Access	CMTS	Access
usbCDCIndex	M	RO	NA	
usbCDCIfIndex	M	RO	NA	
usbCDCSubclass	M	RO	NA	
usbCDCVersion	M	RO	NA	
usbCDCDataTransferType	M	RO	NA	
usbCDCDataEndpoints	M	RO	NA	
usbCDCStalls	M	RO	NA	

usbCDCEtherTable

Object	CM	Access	CMTS	Access
usbCDCEtherIndex	M	RO	NA	
usbCDCEtherIfIndex	M	RO	NA	
usbCDCEtherMacAddress	M	RO	NA	
usbCDCEtherPacketFilter	M	RO	NA	

usbCDCEtherDataStatisticsCapabilities		M	RO	NA	
usbCDCEtherDataCheckErrs		M	RO	NA	
DOCS-QOS-MIB (draft-ietf-ipcdn-qos-mib-04.txt)					
NOTE: 2.0 CMs in 1.0 mode MUST NOT support this MIB.					
docsQosPktClassTable					
Object	**2.0 CM in 1.1 mode**	**Access**	**2.0 CM in 2.0 mode**	**Access**	**CMTS**	**Access**
docsQosPktClassId	M	N-Acc	M	N-Acc	M	N-Acc
docsQosPktClassDirection	M	RO	M	RO	M	RO
docsQosPktClassPriority	M	RO	M	RO	M	RO
docsQosPktClassIpTosLow	M	RO	M	RO	M	RO
docsQosPktClassIpTosHigh	M	RO	M	RO	M	RO
docsQosPktClassIpTosMask	M	RO	M	RO	M	RO
docsQosPktClassIpProtocol	M	RO	M	RO	M	RO
docsQosPktClassIpSourceAddr	M	RO	M	RO	M	RO
docsQosPktClassIpSourceMask	M	RO	M	RO	M	RO
docsQosPktClassIpDestAddr	M	RO	M	RO	M	RO
docsQosPktClassIpDestMask	M	RO	M	RO	M	RO
docsQosPktClassSourcePortStart	M	RO	M	RO	M	RO
docsQosPktClassSourcePortEnd	M	RO	M	RO	M	RO
docsQosPktClassDestPortStart	M	RO	M	RO	M	RO
docsQosPktClassDestPortEnd	M	RO	M	RO	M	RO
docsQosPktClassDestMacAddr	M	RO	M	RO	M	RO
docsQosPktClassDestMacMask	M	RO	M	RO	M	RO
docsQosPktClassSourceMacAddr	M	RO	M	RO	M	RO
docsQosPktClassEnetProtocolType	M	RO	M	RO	M	RO
docsQosPktClassEnetProtocol	M	RO	M	RO	M	RO
docsQosPktClassUserPriLow	M	RO	M	RO	M	RO
docsQosPktClassUserPriHigh	M	RO	M	RO	M	RO
docsQosPktClassVlanId	M	RO	M	RO	M	RO
docsQosPktClassState	M	RO	M	RO	M	RO
docsQosPktClassPkts	M	RO	M	RO	M	RO
docsQosPktClasBitMap	M	RO	M	RO	M	RO

docsQosParamSetTable

Object	2.0 CM in 1.1 mode	Access	2.0 CM in 2.0 mode	Access	CMTS	Access
docsQosParamSetServiceClassName	M	RO	M	RO	M	RO
docsQosParamSetPriority	M	RO	M	RO	M	RO
docsQosParamSetMaxTrafficRate	M	RO	M	RO	M	RO
docsQosParamSetMaxTrafficBurst	M	RO	M	RO	M	RO
docsQosParamSetMinReservedRate	M	RO	M	RO	M	RO
docsQosParamSetMinReservedPkt	M	RO	M	RO	M	RO
docsQosParamSetActiveTimeout	M	RO	M	RO	M	RO
docsQosParamSetAdmittedTimeout	M	RO	M	RO	M	RO
docsQosParamSetMaxConcatBurst	M	RO	M	RO	M	RO
docsQosParamSetSchedulingType	M	RO	M	RO	M	RO
docsQosParamSetNomPollInterval	M	RO	M	RO	M	RO
docsQosParamSetTolPollJitter	M	RO	M	RO	M	RO
docsQosParamSetUnsolicitGrantSize	M	RO	M	RO	M	RO
docsQosParamSetNomGrantInterval	M	RO	M	RO	M	RO
docsQosParamSetTolGrantJitter	M	RO	M	RO	M	RO
docsQosParamSetGrantsPerInterval	M	RO	M	RO	M	RO
docsQosParamSetTosAndMask	M	RO	M	RO	M	RO
docsQosParamSetTosOrMask	M	RO	M	RO	M	RO
docsQosParamSetMaxLatency	M	RO	M	RO	M	RO
docsQosParamSetType	M	NA	M	NA	M	NA
docsQosParamSetRequestPolicyOct	M	RO	M	RO	M	RO
docsQosParamSetBitMap	M	RO	M	RO	M	RO

docsQosServiceFlowTable

Object	2.0 CM in 1.1 mode	Access	2.0 CM in 2.0 mode	Access	CMTS	Access
docsQosServiceFlowId	M	N-Acc	M	N-Acc	M	N-Acc
docsQosServiceFlowSID	M	RO	M	RO	M	RO
docsQosServiceFlowDirection	M	RO	M	RO	M	RO
docsQosServiceFlowPrimary	M	RO	M	RO	M	RO

docsQosServiceFlowStatsTable

Object	2.0 CM in 1.1 mode	Access	2.0 CM in 2.0 mode	Access	CMTS	Access
docsQosServiceFlowPkts	M	RO	M	RO	M	RO
docsQosServiceFlowOctets	M	RO	M	RO	M	RO
docsQosServiceFlowTimeCreated	M	RO	M	RO	M	RO
docsQosServiceFlowTimeActive	M	RO	M	RO	M	RO
docsQosServiceFlowPHSUnknowns	M	RO	M	RO	M	RO
docsQosServiceFlowPolicedDropPkts	M	RO	M	RO	M	RO
docsQosServiceFlowPolicedDelayPkts	M	RO	M	RO	M	RO

docsQosUpstreamStatsTable

Object	2.0 CM in 1.1 mode	Access	2.0 CM in 2.0 mode	Access	CMTS	Access
docsQosSID	N-Sup		N-Sup		M	N-Acc
docsQosUpstreamFragments	N-Sup		N-Sup		M	RO
docsQosUpstreamFragDiscards	N-Sup		N-Sup		M	RO
docsQosUpstreamConcatBursts	N-Sup		N-Sup		M	RO

docsQosDynamicServiceStatsTable

Object	2.0 CM in 1.1 mode	Access	2.0 CM in 2.0 mode	Access	CMTS	Access
docsQosIfDirection	M	N-Acc	M	N-Acc	M	N-Acc
docsQosDSAReqs	M	RO	M	RO	M	RO
docsQosDSARsps	M	RO	M	RO	M	RO
docsQosDSAAcks	M	RO	M	RO	M	RO
docsQosDSCReqs	M	RO	M	RO	M	RO
docsQosDSCRsps	M	RO	M	RO	M	RO
docsQosDSCAcks	M	RO	M	RO	M	RO
docsQosDSDReqs	M	RO	M	RO	M	RO
docsQosDSDRsps	M	RO	M	RO	M	RO

docsQosDynamicAdds	M	RO	M	RO	M	RO
docsQosDynamicAddFails	M	RO	M	RO	M	RO
docsQosDynamicChanges	M	RO	M	RO	M	RO
docsQosDynamicChangeFails	M	RO	M	RO	M	RO
docsQosDynamicDeletes	M	RO	M	RO	M	RO
docsQosDynamicDeleteFails	M	RO	M	RO	M	RO
docsQosDCCReqs	M	RO	M	RO	M	RO
docsQosDCCRsps	M	RO	M	RO	M	RO
docsQosDCCAcks	M	RO	M	RO	M	RO
docsQosDCCs	M	RO	M	RO	M	RO
docsQosDCCFails	M	RO	M	RO	M	RO

docsQosServiceFlowLogTable

Object	**2.0 CM in 1.1 mode**	**Access**	**2.0 CM in 2.0 mode**	**Access**	**CMTS**	**Access**
docsQosServiceFlowLogIndex	N-Sup		N-Sup		M	N-Acc
docsQosServiceFlowLogIfIndex	N-Sup		N-Sup		M	RO
docsQosServiceFlowLogSFID	N-Sup		N-Sup		M	RO
docsQosServiceFlowLogCmMac	N-Sup		N-Sup		M	RO
docsQosServiceFlowLogPkts	N-Sup		N-Sup		M	RO
docsQosServiceFlowLogOctets	N-Sup		N-Sup		M	RO
docsQosServiceFlowLogTimeDeleted	N-Sup		N-Sup		M	RO
docsQosServiceFlowLogTimeCreated	N-Sup		N-Sup		M	RO
docsQosServiceFlowLogTimeActive	N-Sup		N-Sup		M	RO
docsQosServiceFlowLogDirection	N-Sup		N-Sup		M	RO
docsQosServiceFlowLogPrimary	N-Sup		N-Sup		M	RO
docsQosServiceFlowLogServiceClassName	N-Sup		N-Sup		M	RO
docsQosServiceFlowLogPolicedDropPkts	N-Sup		N-Sup		M	RO

docsQosServiceFlowLogPolicedDelayPkts	N-Sup		N-Sup		M	RO
docsQosServiceFlowLogControl	N-Sup		N-Sup		M	RW
docsQosServiceClassTable						
Object	**2.0 CM in 1.1 mode**	**Access**	**2.0 CM in 2.0 mode**	**Access**	**CMTS**	**Access**
docsQosServiceClassName	N-Sup		N-Sup		M	N-Acc
docsQosServiceClassStatus	N-Sup		N-Sup		M	RC
docsQosServiceClassPriority	N-Sup		N-Sup		M	RC
docsQosServiceClassMaxTrafficRate	N-Sup		N-Sup		M	RC
docsQosServiceClassMaxTrafficBurst	N-Sup		N-Sup		M	RC
docsQosServiceClassMinReservedRate	N-Sup		N-Sup		M	RC
docsQosServiceClassMinReservedPkt	N-Sup		N-Sup		M	RC
docsQosServiceClassMaxConcatBurst	N-Sup		N-Sup		M	RC
docsQosServiceClassNomPollInterval	N-Sup		N-Sup		M	RC
docsQosServiceClassTolPollJitter	N-Sup		N-Sup		M	RC
docsQosServiceClassUnsolicitGrantSize	N-Sup		N-Sup		M	RC
docsQosServiceClassNomGrantInterval	N-Sup		N-Sup		M	RC
docsQosServiceClassTolGrantJitter	N-Sup		N-Sup		M	RC
docsQosServiceClassGrantsPerInterval	N-Sup		N-Sup		M	RC
docsQosServiceClassMaxLatency	N-Sup		N-Sup		M	RC
docsQosServiceClassActiveTimeout	N-Sup		N-Sup		M	RC
docsQosServiceClassAdmittedTimeout	N-Sup		N-Sup		M	RC
docsQosServiceClassSchedulingTime	N-Sup		N-Sup		M	RC

docsQosServiceClassRequestPolicy	N-Sup		N-Sup		M	RC
docsQosServiceClassTosAndMask	N-Sup		N-Sup		M	RC
docsQosServiceClassTosOrMask	N-Sup		N-Sup		M	RC
docsQosServiceClassDirection	N-Sup		N-Sup		M	RC
docsQosServiceClassPolicyTable						
Object	**2.0 CM in 1.1 mode**	**Access**	**2.0 CM in 2.0 mode**	**Access**	**CMTS**	**Access**
docsQosServiceClassPolicyIndex	O	N-Acc	O	N-Acc	O	N-Acc
docsQosServiceClassPolicyName	O	RC	O	RC	O	RC
docsQosServiceClassPolicyRulePriority	O	RC	O	RC	O	RC
docsQosServiceClassPolicyStatus	O	RC	O	RC	O	RC
docsQosPHSTable						
Object	**2.0 CM in 1.1 mode**	**Access**	**2.0 CM in 2.0 mode**	**Access**	**CMTS**	**Access**
docsQosPHSField	O	RO	O	RO	O	RO
docsQosPHSMask	O	RO	O	RO	O	RO
docsQosPHSSize	O	RO	O	RO	O	RO
docsQosPHSVerify	O	RO	O	RO	O	RO
docsQosPHSIndex	O	RO	O	RO	O	RO
docsQosCmtsMacToSrvFlowTable						
Object	**2.0 CM in 1.1 mode**	**Access**	**2.0 CM in 2.0 mode**	**Access**	**CMTS**	**Access**
docsQosCmtsCmMac	N-Sup		N-Sup		M	N-Acc
docsQosCmtsServiceFlowId	N-Sup		N-Sup		M	N-Acc
docsQosCmtsIfIndex	N-Sup		N-Sup		M	RO

DOCS-SUBMGT-MIB (draft-ietf-ipcdn-subscriber-mib-02.txt) Subscriber Management MIB				
docsSubMgtCpeControlTable				
Object	**CM**	**Access**	**CMTS**	**Access**
docsSubMgtCpeControlMaxCpeIp	NA	NA	M	RW
docsSubMgtCpeControlActive	NA	NA	M	RW
docsSubMgtCpeControlLearnable	NA	NA	M	RW
docsSubMgtCpeControlReset	NA	NA	M	RW
docsSubMgtCpeMaxIpDefault	NA	NA	M	RW
docsSubMgtCpeActiveDefault	NA	NA	M	RW
docsSubMgtCpeIpTable				
Object	**CM**	**Access**	**CMTS**	**Access**
docsSubMgtCpeIpIndex	NA	NA	M	N-Acc
docsSubMgtCpeIpAddr	NA	NA	M	RO
docsSubMgtCpeIpLearned	NA	NA	M	RO
docsSubMgtPktFilterTable				
Object	**CM**	**Access**	**CMTS**	**Access**
docsSubMgtPktFilterGroup	NA	NA	M	N-Acc
docsSubMgtPktFilterIndex	NA	NA	M	N-Acc
docsSubMgtPktFilterSrcAddr	NA	NA	M	RC
docsSubMgtPktFilterSrcMask	NA	NA	M	RC
docsSubMgtPktFilterDstAddr	NA	NA	M	RC
docsSubMgtPktFilterDstMask	NA	NA	M	RC
docsSubMgtPktFilterUlp	NA	NA	M	RC
docsSubMgtPktFilterTosValue	NA	NA	M	RC
docsSubMgtPktFilterTosMask	NA	NA	M	RC
docsSubMgtPktFilterAction	NA	NA	M	RC
docsSubMgtPktFilterMatches	NA	NA	M	RO
docsSubMgtPktFilterStatus	NA	NA	M	RC
docsSubMgtTcpUdpFilterTable				
Object	**CM**	**Access**	**CMTS**	**Access**
docsSubMgtTcpUdpSrcPort	NA	NA	M	RC

docsSubMgtTcpUdpDstPort	NA	NA	M	RC
docsSubMgtTcpFlagValues	NA	NA	M	RC
docsSubMgtTcpFlagMask	NA	NA	M	RC
docsSubMgtTcpUdpStatus	NA	NA	M	RC

docsSubMgtCmFilterTable

Object	**CM**	**Access**	**CMTS**	**Access**
docsSubMgtSubFilterDownstream	NA	NA	M	RW
docsSubMgtSubFilterUpstream	NA	NA	M	NW
docsSubMgtCmFilterDownstream	NA	NA	M	RW
docsSubMgtCmFilterUpstream	NA	NA	M	RW

(no table)

Objects	**CM**	**Access**	**CMTS**	**Access**
docsSubMgtSubFilterDownDefault	NA	NA	M	RW
docsSubMgtSubFilterUpDefault	NA	NA	M	RW
docsSubMgtCmFilterDownDefault	NA	NA	M	RW
docsSubMgtCmFilterUpDefault	NA	NA	M	RW

IGMP-STD-MIB (RFC 2933)

This MIB is optional for Bridging CMTSes.

NOTE: 2.0 CMs in 1.0 mode are not required to implement RFC 2933.

IgmpInterfaceTable

Object	**2.0 CM in 1.0 mode**	**Access**	**2.0 CM in 1.1 mode**	**Access**	**2.0 CM in 2.0 mode**	**Access**	**CMTS**	**Access**
IgmpInterfaceIfIndex	O	N-Acc	M	N-Acc	M	N-Acc	M	N-Acc
igmpInterfaceQueryInterval	O	RC	M	RC	M	RC	M	RC
igmpInterfaceStatus	O	RC	M	RC	M	RC	M	RC
igmpInterfaceVersion	O	RC	M	RC	M	RC	M	RC
igmpInterfaceQuerier	O	RO	M	RO	M	RO	M	RO
igmpInterfaceQueryMaxResponseTime	O	RO	M	RO	M	RO	M	RO
igmpInterfaceVersion1QuerierTimer	O	RO	M	RO	M	RO	M	RO
igmpInterfaceWrongVersionQueries	O	RO	M	RO	M	RO	M	RO
igmpInterfaceJoins	O	RO	M	RO	M	RO	M	RO

IgmpInterfaceGroups	O	RO	M	RO	M	RO	M	RO
igmpInterfaceRobustness	O	RC	M	RC	M	RC	M	RC
igmpInterfaceLastMembQueryIntvl	O	RC	M	RC	M	RC	M	RC
igmpInterfaceProxyIfIndex	O	RC	M	RC	M	RC	M	RC
igmpInterfaceQuerierUpTime	O	RO	M	RO	M	RO	M	RO
igmpInterfaceQuerierExpiryTime	O	RO	M	RO	M	RO	M	RO
igmpCacheTable								
Object	**2.0 CM in 1.0 mode**	**Access**	**2.0 CM in 1.1 mode**	**Access**	**2.0 CM in 2.0 mode**	**Access**	**CMTS**	**Access**
igmpCacheAddress	O	N-Acc	M	N-Acc	M	N-Acc	M	N-Acc
igmpCacheIfIndex	O	N-Acc	M	N-Acc	M	N-Acc	M	N-Acc
IgmpCacheSelf	O	RC	M	RC	M	RC	M	RC
igmpCacheLastReporter	O	RO	M	RO	M	RO	M	RO
igmpCacheUpTime	O	RO	M	RO	M	RO	M	RO
igmpCacheExpiryTime	O	RO	M	RO	M	RO	M	RO
igmpCacheStatus	O	RC	M	RC	M	RC	M	RC
igmpCacheVersion1HostTimer	O	RO	M	RO	M	RO	M	RO
Account Management MIB (MIB defining work is still in progress.)								
docsCpeSegmentTable								
Object					**CM**	**Access**	**CMTS**	**Access**
docsCpeSegmentID					NA	NA	O	RO
docsCpeSegmentIp					NA	NA	O	RC
docsCpeTrafficData Table								
Objects					**CM**	**Access**	**CMTS**	**Access**
docsCpeIpAddress					NA	NA	O	RO
docsCpeTrafficDataUpStreamPackets					NA	NA	O	RC
docsCpeTrafficDataDownStreamPackets					NA	NA	O	RC
docsCpeTrafficDataUpStreamOctets					NA	NA	O	RC
docsCpeTrafficDataDownStreamOctets					NA	NA	O	RC
docsCpeTrafficDataUpStreamDropPackets					NA	NA	O	RC
docsCpeTrafficDataDownStreamDropPackets					NA	NA	O	RC

docsCmCpeTable	**CM**	**Access**	**CMTS**	**Access**
docsCmMacAddress	NA	NA	O	RC
docsCmIpAddress	NA	NA	O	RC
docsCpeMACAddress	NA	NA	O	RC
docsCpeIpAddress	NA	NA	O	RC

DOCS-BPI-MIB RFC 3083

docsBpiCmBaseTable

Object	**2.0 CM in 1.0 mode**	**Access**	**2.0 CM in 1.1 mode**	**Access**	**2.0 CM in 2.0 mode**	**Access**	**CMTS**	**Access**
docsBpiCmPrivacyEnable	M	RO	N-Sup		N-Sup		NA	
docsBpiCmPublicKey	M	RO	N-Sup		N-Sup		NA	
docsBpiCmAuthState	M	RO	N-Sup		N-Sup		NA	
docsBpiCmAuthKeySequenceNumber	M	RO	N-Sup		N-Sup		NA	
docsBpiCmAuthExpires	M	RO	N-Sup		N-Sup		NA	
docsBpiCmAuthReset	M	RW	N-Sup		N-Sup		NA	
docsBpiCmAuthGraceTime	M	RO	N-Sup		N-Sup		NA	
docsBpiCmTEKGraceTime	M	RO	N-Sup		N-Sup		NA	
docsBpiCmAuthWaitTimeout	M	RO	N-Sup		N-Sup		NA	
docsBpiCmReauthWaitTimeout	M	RO	N-Sup		N-Sup		NA	
docsBpiCmOpWaitTimeout	M	RO	N-Sup		N-Sup		NA	
docsBpiCmRekeyWaitTimeout	M	RO	N-Sup		N-Sup		NA	
docsBpiCmAuthRejectWaitTimeout	M	RO	N-Sup		N-Sup		NA	
docsBpiCmAuthRequests	M	RO	N-Sup		N-Sup		NA	
docsBpiCmAuthReplies	M	RO	N-Sup		N-Sup		NA	

docsBpiCmAuthRejects	M	RO	N-Sup		N-Sup		NA	
docsBpiCmAuthInvalids	M	RO	N-Sup		N-Sup		NA	
docsBpiCmAuthRejectErrorCode	M	RO	N-Sup		N-Sup		NA	
docsBpiCmAuthRejectErrorString	M	RO	N-Sup		N-Sup		NA	
docsBpiCmAuthInvalidErrorCode	M	RO	N-Sup		N-Sup		NA	
docsBpiCmAuthInvalidErrorString	M	RO	N-Sup		N-Sup		NA	
docsBpiCmTEKTable								
Object	**2.0 CM in 1.0 mode**	**Access**	**2.0 CM in 1.1 mode**	**Access**	**2.0 CM in 2.0 mode**	**Access**	**CMTS**	**Access**
docsBpiCmTEKPrivacyEnable	M	RO	N-Sup		N-Sup		NA	
docsBpiCmTEKState	M	RO	N-Sup		N-Sup		NA	
docsBpiCmTEKExpiresOld	M	RO	N-Sup		N-Sup		NA	
docsBpiCmTEKExpiresNew	M	RO	N-Sup		N-Sup		NA	
docsBpiCmTEKKeyRequests	M	RO	N-Sup		N-Sup		NA	
docsBpiCmTEKKeyReplies	M	RO	N-Sup		N-Sup		NA	
docsBpiCmTEKKeyRejects	M	RO	N-Sup		N-Sup		NA	
docsBpiCmTEKInvalids	M	RO	N-Sup		N-Sup		NA	
docsBpiCmTEKAuthPends	M	RO	N-Sup		N-Sup		NA	
docsBpiCmTEKKeyRejectErrorCode	M	RO	N-Sup		N-Sup		NA	
docsBpiCmTEKKeyRejectErrorString	M	RO	N-Sup		N-Sup		NA	
docsBpiCmTEKInvalidErrorCode	M	RO	N-Sup		N-Sup		NA	
docsBpiCmTEKInvalidErrorString	M	RO	N-Sup		N-Sup		NA	

docsBpiCmtsBaseTable

Object	2.0 CM in 1.0 mode	Access	2.0 CM in 1.1 mode	Access	2.0 CM in 2.0 mode	Access	CMTS	Access
docsBpiCmtsDefaultAuthLifetime	NA		NA		NA		N-Sup	
docsBpiCmtsDefaultTEKLifetime	NA		NA		NA		N-Sup	
docsBpiCmtsDefaultAuthGraceTime	NA		NA		NA		N-Sup	
docsBpiCmtsDefaultTEKGraceTime	NA		NA		NA		N-Sup	
docsBpiCmtsAuthRequests	NA		NA		NA		N-Sup	
docsBpiCmtsAuthReplies	NA		NA		NA		N-Sup	
docsBpiCmtsAuthRejects	NA		NA		NA		N-Sup	
docsBpiCmtsAuthInvalids	NA		NA		NA		N-Sup	

docsBpiCmtsAuthTable

Object	2.0 CM in 1.0 mode	Access	2.0 CM in 1.1 mode	Access	2.0 CM in 2.0 mode	Access	CMTS	Access
docsBpiCmtsAuthCmMacAddress	NA		NA		NA		N-Sup	
docsBpiCmtsAuthCmPublicKey	NA		NA		NA		N-Sup	
docsBpiCmtsAuthCmKeySequence-Number	NA		NA		NA		N-Sup	
docsBpiCmtsAuthCmExpires	NA		NA		NA		N-Sup	
docsBpiCmtsAuthCmLifetime	NA		NA		NA		N-Sup	
docsBpiCmtsAuthCmGraceTime	NA		NA		NA		N-Sup	
docsBpiCmtsAuthCmReset	NA		NA		NA		N-Sup	
docsBpiCmtsAuthCmRequests	NA		NA		NA		N-Sup	
docsBpiCmtsAuthCmReplies	NA		NA		NA		N-Sup	
docsBpiCmtsAuthCmRejects	NA		NA		NA		N-Sup	
docsBpiCmtsAuthCmInvalids	NA		NA		NA		N-Sup	
docsBpiCmtsAuthRejectErrorCode	NA		NA		NA		N-Sup	
docsBpiCmtsAuthRejectErrorString	NA		NA		NA		N-Sup	
docsBpiCmtsAuthInvalidErrorCode	NA		NA		NA		N-Sup	
docsBpiCmtsAuthInvalidErrorString	NA		NA		NA		N-Sup	

docsBpiCmtsTEKTable

Object	2.0 CM in 1.0 mode	Access	2.0 CM in 1.1 mode	Access	2.0 CM in 2.0 mode	Access	CMTS	Access
docsBpiCmtsTEKLifetime	NA		NA		NA		N-Sup	
docsBpiCmtsTEKGraceTime	NA		NA		NA		N-Sup	
docsBpiCmtsTEKExpiresOld	NA		NA		NA		N-Sup	
docsBpiCmtsTEKExpiresNew	NA		NA		NA		N-Sup	
docsBpiCmtsTEKReset	NA		NA		NA		N-Sup	
docsBpiCmtsKeyRequests	NA		NA		NA		N-Sup	
docsBpiCmtsKeyReplies	NA		NA		NA		N-Sup	
docsBpiCmtsKeyRejects	NA		NA		NA		N-Sup	
docsBpiCmtsTEKInvalids	NA		NA		NA		N-Sup	
docsBpiCmtsKeyRejectErrorCode	NA		NA		NA		N-Sup	
docsBpiCmtsKeyRejectErrorString	NA		NA		NA		N-Sup	
docsBpiCmtsTEKInvalidErrorCode	NA		NA		NA		N-Sup	
docsBpiCmtsTEKInvalidErrorString	NA		NA		NA		N-Sup	

docsBpiIpMulticastMapTable

Object	2.0 CM in 1.0 mode	Access	2.0 CM in 1.1 mode	Access	2.0 CM in 2.0 mode	Access	CMTS	Access
docsBpiIpMulticastAddress	NA		NA		NA		N-Sup	
docsBpiIpMulticastprefixLength	NA		NA		NA		N-Sup	
docsBpiIpMulticastServiceId	NA		NA		NA		N-Sup	
docsBpiIpMulticastMapControl	NA		NA		NA		N-Sup	

docsBpiMulticastAuthTable

Object	2.0 CM in 1.0 mode	Access	2.0 CM in 1.1 mode	Access	2.0 CM in 2.0 mode	Access	CMTS	Access
docsBpiMulticastServiceId	NA		NA		NA		N-Sup	
docsBpiMulticastCmMacAddress	NA		NA		NA		N-Sup	
docsBpiMulticastAuthControl	NA		NA		NA		N-Sup	

BPI+ MIB (draft-ietf-ipcdn-bpiplus-mib-05.txt)								
docsBpi2CmBaseTable								
Object	**2.0 CM in 1.0 mode**	**Access**	**2.0 CM in 1.1 mode**	**Access**	**2.0 CM in 2.0 mode**	**Access**	**CMTS**	**Access**
docsBpi2CmPrivacyEnable	O	RO	M	RO	M	RO	NA	
docsBpi2CmPublicKey	O	RO	M	RO	M	RO	NA	
docsBpi2CmAuthState	O	RO	M	RO	M	RO	NA	
docsBpi2CmAuthKeySequenceNumber	O	RO	M	RO	M	RO	NA	
docsBpi2CmAuthExpiresOld	O	RO	M	RO	M	RO	NA	
docsBpi2CmAuthExpiresNew	O	RO	M	RO	M	RO	NA	
docsBpi2CmAuthReset	O	RW	M	RW	M	RW	NA	
docsBpi2CmAuthGraceTime	O	RO	M	RO	M	RO	NA	
docsBpi2CmTEKGraceTime	O	RO	M	RO	M	RO	NA	
docsBpi2CmAuthWaitTimeout	O	RO	M	RO	M	RO	NA	
docsBpi2CmReauthWaitTimeout	O	RO	M	RO	M	RO	NA	
docsBpi2CmOpWaitTimeout	O	RO	M	RO	M	RO	NA	
docsBpi2CmRekeyWaitTimeout	O	RO	M	RO	M	RO	NA	
docsBpi2CmAuthRejectWaitTimeout	O	RO	M	RO	M	RO	NA	
docsBpi2CmSAMapWaitTimeout	O	RO	M	RO	M	RO	NA	
docsBpi2CmSAMapMaxRetries	O	RO	M	RO	M	RO	NA	
docsBpi2CmAuthentInfos	O	RO	M	RO	M	RO	NA	
docsBpi2CmAuthRequests	O	RO	M	RO	M	RO	NA	
docsBpi2CmAuthReplies	O	RO	M	RO	M	RO	NA	
docsBpi2CmAuthRejects	O	RO	M	RO	M	RO	NA	
docsBpi2CmAuthInvalids	O	RO	M	RO	M	RO	NA	
docsBpi2CmAuthRejectErrorCode	O	RO	M	RO	M	RO	NA	
docsBpi2CmAuthRejectErrorString	O	RO	M	RO	M	RO	NA	
docsBpi2CmAuthInvalidErrorCode	O	RO	M	RO	M	RO	NA	
docsBpi2CmAuthInvalidErrorString	O	RO	M	RO	M	RO	NA	

docsBpi2CmTEKTable

Object	2.0 CM in 1.0 mode	Access	2.0 CM in 1.1 mode	Access	2.0 CM in 2.0 mode	Access	CMTS	Access
docsBpi2CmTEKSAId	O	N-Acc	M	N-Acc	M	N-Acc	NA	
docsBpi2CmTEKSAType	O	RO	M	RO	M	RO	NA	
docsBpi2CmTEKDataEncryptAlg	O	RO	M	RO	M	RO	NA	
docsBpi2CmTEKDataAuthentAlg	O	RO	M	RO	M	RO	NA	
docsBpi2CmTEKState	O	RO	M	RO	M	RO	NA	
docsBpi2CmTEKKeySequenceNumber	O	RO	M	RO	M	RO	NA	
docsBpi2CmTEKExpiresOld	O	RO	M	RO	M	RO	NA	
docsBpi2CmTEKExpiresNew	O	RO	M	RO	M	RO	NA	
docsBpi2CmTEKKeyRequests	O	RO	M	RO	M	RO	NA	
docsBpi2CmTEKKeyReplies	O	RO	M	RO	M	RO	NA	
docsBpi2CmTEKKeyRejects	O	RO	M	RO	M	RO	NA	
docsBpi2CmTEKInvalids	O	RO	M	RO	M	RO	NA	
docsBpi2CmTEKAuthPends	O	RO	M	RO	M	RO	NA	
docsBpi2CmTEKKeyRejectErrorCode	O	RO	M	RO	M	RO	NA	
docsBpi2CmTEKKeyRejectErrorString	O	RO	M	RO	M	RO	NA	
docsBpi2CmTEKInvalidErrorCode	O	RO	M	RO	M	RO	NA	
docsBpi2CmTEKInvalidErrorString	O	RO	M	RO	M	RO	NA	

docsBpi2CmIpMulticastMapTable

Object	2.0 CM in 1.0 mode	Access	2.0 CM in 1.1 mode	Access	2.0 CM in 2.0 mode	Access	CMTS	Access
docsBpi2CmIpMulticastIndex	O	N-Acc	M	N-Acc	M	N-Acc	NA	
docsBpi2CmIpMulticastAddressType	O	RO	M	RO	M	RO	NA	
docsBpi2CmIpMulticastAddress	O	RO	M	RO	M	RO	NA	
docsBpi2CmIpMulticastSAId	O	RO	M	RO	M	RO	NA	
docsBpi2CmIpMulticastSAMapState	O	RO	M	RO	M	RO	NA	
docsBpi2CmIpMulticastSAMapRequests	O	RO	M	RO	M	RO	NA	
docsBpi2CmIpMulticastSAMapReplies	O	RO	M	RO	M	RO	NA	
docsBpi2CmIpMulticastSAMapRejects	O	RO	M	RO	M	RO	NA	
docsBpi2CmIpMulticastSAMapRejectErrorCode	O	RO	M	RO	M	RO	NA	

docsBpi2CmIpMulticastSAMapRejectErrorString	O	RO	M	RO	M	RO	NA	
docsBpi2CmDeviceCertTable								
Object	**2.0 CM in 1.0 mode**	**Access**	**2.0 CM in 1.1 mode**	**Access**	**2.0 CM in 2.0 mode**	**Access**	**CMTS**	**Access**
docsBpi2CmDeviceCmCert	M	RW/RO	M	RW/RO	M	RW/RO	NA	
docsBpi2CmDeviceManufCert	M	RO	M	RO	M	RO	NA	
docsBpi2CmCryptoSuiteTable								
Object	**2.0 CM in 1.0 mode**	**Access**	**2.0 CM in 1.1 mode**	**Access**	**2.0 CM in 2.0 mode**	**Access**	**CMTS**	**Access**
docsBpi2CmCryptoSuiteIndex	M	N-Acc	M	N-Acc	M	N-Acc	NA	
docsBpi2CmCryptoSuiteDataEncryptAlg	M	RO	M	RO	M	RO	NA	
docsBpi2CmCryptoSuiteDataAuthentAlg	M	RO	M	RO	M	RO	NA	
docsBpi2CmtsBaseEntryTable								
Object	**2.0 CM in 1.0 mode**	**Access**	**2.0 CM in 1.1 mode**	**Access**	**2.0 CM in 2.0 mode**	**Access**	**CMTS**	**Access**
docsBpi2CmtsDefaultAuthLifetime	NA		NA		NA		M	RW
docsBpi2CmtsDefaultTEKLifetime	NA		NA		NA		M	RW
docsBpi2CmtsDefaultSelfSignedManufCertTrust	NA		NA		NA		M	RW
docsBpi2CmtsCheckCertValidityPeriods	NA		NA		NA		M	RW
docsBpi2CmtsAuthentInfos	NA		NA		NA		M	RO
docsBpi2CmtsAuthRequests	NA		NA		NA		M	RO
docsBpi2CmtsAuthReplies	NA		NA		NA		M	RO
docsBpi2CmtsAuthRejects	NA		NA		NA		M	RO
docsBpi2CmtsAuthInvalids	NA		NA		NA		M	RO
docsBpi2CmtsSAMapRequests	NA		NA		NA		M	RO
docsBpi2CmtsSAMapReplies	NA		NA		NA		M	RO
docsBpi2CmtsSAMapRejects	NA		NA		NA		M	RO

docsBpi2CmtsAuthEntryTable

Object	2.0 CM in 1.0 mode	Access	2.0 CM in 1.1 mode	Access	2.0 CM in 2.0 mode	Access	CMTS	Access
docsBpi2CmtsAuthCmMacAddress	NA		NA		NA		M	N-Acc
docsBpi2CmtsAuthCmBpiVersion	NA		NA		NA		M	RO
docsBpi2CmtsAuthCmPublicKey	NA		NA		NA		M	RO
docsBpi2CmtsAuthCmKeySequence Number	NA		NA		NA		M	RO
docsBpi2CmtsAuthCmExpiresOld	NA		NA		NA		M	RO
docsBpi2CmtsAuthCmExpiresNew	NA		NA		NA		M	RO
docsBpi2CmtsAuthCmLifetime	NA		NA		NA		M	RW
docsBpi2CmtsAuthCmGraceTime	NA		NA		NA		Ob	RO
docsBpi2CmtsAuthCmReset	NA		NA		NA		M	RW
docsBpi2CmtsAuthCmInfos	NA		NA		NA		M	RO
docsBpi2CmtsAuthCmRequests	NA		NA		NA		M	RO
docsBpi2CmtsAuthCmReplies	NA		NA		NA		M	RO
docsBpi2CmtsAuthCmRejects	NA		NA		NA		M	RO
docsBpi2CmtsAuthCmInvalids	NA		NA		NA		M	RO
docsBpi2CmtsAuthRejectErrorCode	NA		NA		NA		M	RO
docsBpi2CmtsAuthRejectErrorString	NA		NA		NA		M	RO
docsBpi2CmtsAuthInvalidErrorCode	NA		NA		NA		M	RO
docsBpi2CmtsAuthInvalidErrorString	NA		NA		NA		M	RO
docsBpi2CmtsAuthPrimarySAId	NA		NA		NA		M	RO
docsBpi2CmtsAuthBpkmCmCertValid	NA		NA		NA		M	RO
docsBpi2CmtsAuthBpkmCmCert	NA		NA		NA		M	RO

docsBpi2CmtsTEKTable

Object	2.0 CM in 1.0 mode	Access	2.0 CM in 1.1 mode	Access	2.0 CM in 2.0 mode	Access	CMTS	Access
docsBpi2CmtsTEKSAId	NA		NA		NA		M	N-Acc
docsBpi2CmtsTEKSAType	NA		NA		NA		M	RO
docsBpi2CmtsTEKDataEncryptAlg	NA		NA		NA		M	RO
docsBpi2CmtsTEKDataAuthentAlg	NA		NA		NA		M	RO
docsBpi2CmtsTEKLifetime	NA		NA		NA		M	RW

docsBpi2CmtsTEKGraceTime	NA		NA		NA		Ob	RO
docsBpi2CmtsTEKKeySequenceNumber	NA		NA		NA		M	RO
docsBpi2CmtsTEKExpiresOld	NA		NA		NA		M	RO
docsBpi2CmtsTEKExpiresNew	NA		NA		NA		M	RO
docsBpi2CmtsTEKReset	NA		NA		NA		M	RW
docsBpi2CmtsKeyRequests	NA		NA		NA		M	RO
docsBpi2CmtsKeyReplies	NA		NA		NA		M	RO
docsBpi2CmtsKeyRejects	NA		NA		NA		M	RO
docsBpi2CmtsTEKInvalids	NA		NA		NA		M	RO
docsBpi2CmtsKeyRejectErrorCode	NA		NA		NA		M	RO
docsBpi2CmtsKeyRejectErrorString	NA		NA		NA		M	RO
docsBpi2CmtsTEKInvalidErrorCode	NA		NA		NA		M	RO
docsBpi2CmtsTEKInvalidErrorString	NA		NA		NA		M	RO
docsBpi2CmtsIpMulticastMapTable								
Object	**2.0 CM in 1.0 mode**	**Access**	**2.0 CM in 1.1 mode**	**Access**	**2.0 CM in 2.0 mode**	**Access**	**CMTS**	**Access**
docsBpi2CmtsIpMulticastIndex	NA		NA		NA		M	N-Acc
docsBpi2CmtsIpMulticastAddressType	NA		NA		NA		M	RC/RO
docsBpi2CmtsIpMulticastAddress	NA		NA		NA		M	RC/RO
docsBpi2CmtsIpMulticastMaskType	NA		NA		NA		M	RC/RO
docsBpi2CmtsIpMulticastMask	NA		NA		NA		M	RC/RO
docsBpi2CmtsIpMulticastSAId	NA		NA		NA		M	RC/RO
docsBpi2CmtsIpMulticastSAType	NA		NA		NA		M	RC/RO
docsBpi2CmtsIpMulticastDataEncryptAlg	NA		NA		NA		M	RC/RO
docsBpi2CmtsIpMulticastDataAuthentAlg	NA		NA		NA		M	RC/RO
docsBpi2CmtsIpMulticastSAMapRequests	NA		NA		NA		M	RO
docsBpi2CmtsIpMulticastSAMapReplies	NA		NA		NA		M	RO
docsBpi2CmtsIpMulticastSAMapRejects	NA		NA		NA		M	RO
docsBpi2CmtsIpMulticastSAMapRejectErrorCode	NA		NA		NA		M	RO
docsBpi2CmtsIpMulticastSAMapRejectErrorString	NA		NA		NA		M	RO

docsBpi2CmtsIpMulticastMapControl	NA		NA		NA		M	RC/RO
docsBpi2CmtsMulticastAuthTable								
Object	**2.0 CM in 1.0 mode**	**Access**	**2.0 CM in 1.1 mode**	**Access**	**2.0 CM in 2.0 mode**	**Access**	**CMTS**	**Access**
docsBpi2CmtsMulticastAuthSAId	NA		NA		NA		M	N-Acc
docsBpi2CmtsMulticastAuthCmMacAddress	NA		NA		NA		M	N-Acc
docsBpi2CmtsMulticastAuthControl	NA		NA		NA		M	RC/RO
docsBpi2CmtsProvisionedCmCertTable								
Object	**2.0 CM in 1.0 mode**	**Access**	**2.0 CM in 1.1 mode**	**Access**	**2.0 CM in 2.0 mode**	**Access**	**CMTS**	**Access**
docsBpi2CmtsProvisionedCmCertMac Address	NA		NA		NA		M	N-Acc
docsBpi2CmtsProvisionedCmCertTrust	NA		NA		NA		M	RC
docsBpi2CmtsProvisionedCmCert Source	NA		NA		NA		M	RO
docsBpi2CmtsProvisionedCmCert Status	NA		NA		NA		M	RC
docsBpi2CmtsProvisionedCmCert	NA		NA		NA		M	RC
docsBpi2CmtsCACertTable								
Object	**2.0 CM in 1.0 mode**	**Access**	**2.0 CM in 1.1 mode**	**Access**	**2.0 CM in 2.0 mode**	**Access**	**CMTS**	**Access**
docsBpi2CmtsCACertIndex	NA		NA		NA		M	N-Acc
docsBpi2CmtsCACertSubject	NA		NA		NA		M	RO
docsBpi2CmtsCACertIssuer	NA		NA		NA		M	RO
docsBpi2CmtsCACertSerialNumber	NA		NA		NA		M	RO
docsBpi2CmtsCACertTrust	NA		NA		NA		M	RC
docsBpi2CmtsCACertSource	NA		NA		NA		M	RO
docsBpi2CmtsCACertStatus	NA		NA		NA		M	RC
docsBpi2CmtsCACert	NA		NA		NA		M	RC
docsBpi2CmtsCACertThumprint	NA		NA		NA		M	RO

docsBpi2CodeDownloadGroup

Object	2.0 CM in 1.0 mode	Access	2.0 CM in 1.1 mode	Access	2.0 CM in 2.0 mode	Access	CMTS	Access
docsBpi2CodeDownloadStatusCode	M	RO	M		M	RO	O	RO
docsBpi2CodeDownloadStatusString	M	RO	M		M	RO	O	RO
docsBpi2CodeMfgOrgName	M	RO	M		M	RO	O	RO
docsBpi2CodeMfgCodeAccessStart	M	RO	M		M	RO	O	RO
docsBpi2CodeMfgCvcAccessStart	M	RO	M		M	RO	O	RO
docsBpi2CodeCoSignerOrgName	M	RO	M		M	RO	O	RO
docsBpi2CodeCoSignerCodeAccessStart	M	RO	M		M	RO	O	RO
docsBpi2CodeCoSignerCvcAccessStart	M	RO	M		M	RO	O	RO
docsBpi2CodeCvcUpdate	M	RW	M		M	RW	O	RW

SNMP-USM-DH-OBJECTS-MIB (RFC 2786)

NOTE: SNMP-USM-DH-OBJECTS-MIB is only accessible when the device is in SNMP Coexistence Mode.

Object	CM	Access	CMTS	Access
usmDHParameters	M	RW	O	RW

usmDHUserKeyTable

Object	CM	Access	CMTS	Access
usmDHUserAuthKeyChange	M	RC	O	RC
smDHUserOwnAuthKeyChange	M	RC	O	RC
usmDHUserPrivKeyChange	M	RC	O	RC
usmDHUserOwnPrivKeyChange	M	RC	O	RC

usmDHKickstartTable

Object	CM	Access	CMTS	Access
usmDHKickstartIndex	M	N-Acc	O	N-Acc
usmDHKickstartMyPublic	M	RO	O	RO
usmDHKickstartMgrPublic	M	RO	O	RO
usmDHKickstartSecurityName	M	RO	O	RO

SNMP-VIEW-BASED-ACM-MIB (RFC 2575)				
(Note: SNMP-VIEW-BASED-ACM-MIB is ONLY accessible when the device is in SNMP Coexistence mode.)				
vacmContextTable				
Object	**CM**	**Access**	**CMTS**	**Access**
vacmContextName	M	RO	M	RO
vacmSecurityToGroupTable				
Object	**CM**	**Access**	**CMTS**	**Access**
vacmSecurityModel	M	N-Acc	M	N-Acc
vacmSecurityName	M	N-Acc	M	N-Acc
vacmGroupName	M	RC	M	RC
vacmSecurityToGroupStorageType	M	RC	M	RC
vacmSecurityToGroupStatus	M	RC	M	RC
vacmAccessTable				
Object	**CM**	**Access**	**CMTS**	**Access**
vacmAccessContextPrefix	M	N-Acc	M	N-Acc
vacmAccessSecurityModel	M	N-Acc	M	N-Acc
vacmAccessSecurityLevel	M	N-Acc	M	N-Acc
vacmAccessContextMatch	M	RC	M	RC
vacmAccessReadViewName	M	RC	M	RC
vacmAccessWriteViewName	M	RC	M	RC
vacmAccessNotifyViewName	M	RC	M	RC
vacmAccessStorageType	M	RC	M	RC
vacmAccessStatus	M	RC	M	RC
vacmViewSpinLock	M	RW	M	RW
vacmViewTreeFamilyTable				
Object	**CM**	**Access**	**CMTS**	**Access**
vacmViewTreeFamilyViewName	M	N-Acc	M	N-Acc
vacmViewTreeFamilySubtree	M	N-Acc	M	N-Acc
vacmViewTreeFamilyMask	M	RC	M	RC
vacmViewTreeFamilyType	M	RC	M	RC

vacmViewTreeFamilyStorageType	M	RC	M	RC
vacmViewTreeFamilyStatus	M	RC	M	RC

SNMP-COMMUNITY-MIB (RFC2576)

Note: The SNMP-COMMUNITY-MIB is ONLY accessible when the device is in SNMP Coexistence mode.

snmpCommunityTable

Object	CM	Access	CMTS	Access
snmpCommunityIndex	M	N-Acc	M	N-Acc
snmpCommunityName	M	RC	M	RC
snmpCommunitySecurityName	M	RC	M	RC
snmpCommunityContextEngineID	M	RC	M	RC
snmpCommunityContextName	M	RC	M	RC
snmpCommunityTransportTag	M	RC	M	RC
snmpCommunityStorageType	M	RC	M	RC
snmpCommunityStatus	M	RC	M	RC

SnmpTargetExtTable

Object	CM	Access	CMTS	Access
snmpTargetAddrTMask	M	RC	M	RC
snmpTargetAddrMMS	M	RC	M	RC
snmpTrapAddress	O	ACC-FN	O	ACC-FN
snmpTrapCommunity	O	ACC-FN	O	ACC-FN

SNMP Management Framework architecture (RFC 2571)

snmpEngine Group

Object	CM	Access	CMTS	Access
snmpEngineID	M	RO	M	RO
snmpEngineBoots	M	RO	M	RO
snmpEngineTime	M	RO	M	RO
snmpEngineMaxMessageSize	M	RO	M	RO

SNMP Message Processing and Dispatching MIB (RFC 2572)

Note: The SNMP Message Processing and Dispatching MIB is ONLY accessible when the device is in SNMP Coexistence mode.

snmpMPDStats

Object	CM	Access	CMTS	Access
snmpUnknownSecurityModels	M	RO	M	RO
snmpInvalidMsgs	M	RO	M	RO
snmpUnknownPDUHandlers	M	RO	M	RO

RFC 2573

Note: RFC 2573 is ONLY accessible when the device is in SNMP Coexistence mode.

Object	CM	Access	CMTS	Access
snmpTargetSpinLock	M	RW	M	RW

snmpTargetAddrTable

Object	CM	Access	CMTS	Access
snmpTargetAddrName	M	N-Acc	M	N-Acc
snmpTargetAddrTDomain	M	RC	M	RC
SnmpTargetAddrTAddress	M	RC	M	RC
SnmpTargetAddrTimeout	M	RC	M	RC
SnmpTargetAddrRetryCount	M	RC	M	RC
SnmpTargetAddrTagList	M	RC	M	RC
SnmpTargetAddrParams	M	RC	M	RC
SnmpTargetAddrStorageType	M	RC	M	RC
SnmpTargetAddrRowStatus	M	RC	M	RC

snmpTargetParamsTable

Object	CM	Access	CMTS	Access
SnmpTargetParamsName	M	N-Acc	M	N-Acc
SnmpTargetParamsMPModel	M	RC	M	RC
SnmpTargetParamsSecurityModel	M	RC	M	RC
SnmpTargetParamsSecurityName	M	RC	M	RC
SnmpTargetParamsSecurityLevel	M	RC	M	RC

SnmpTargetParamsStorageType	M	RC	M	RC
SnmpTargetParamsRowStatus	M	RC	M	RC
SnmpUnavailableContexts		RO	M	RO
snmpUnknownContexts	M	RO	M	RO
snmpNotifyTable				
Object	**CM**	**Access**	**CMTS**	**Access**
snmpNotifyName	M	N-Acc	M	N-Acc
snmpNotifyTag	M	RC	M	RC
SnmpNotifyType	M	RC	M	RC
snmpNotifyStorageType	M	RC	M	RC
SnmpNotifyRowStatus	M	RC	M	RC
snmpNotifyFilterProfileTable				
Object	**CM**	**Access**	**CMTS**	**Access**
SnmpNotifyFilterProfileName	M	RC	M	RC
snmpNotifyFilterProfileStorType	M	RC	M	RC
snmpNotifyFilterProfileRowStatus	M	RC	M	RC
snmpNotifyFilterTable				
Object	**CM**	**Access**	**CMTS**	**Access**
SnmpNotifyFilterSubtree	M	N-Acc	M	N-Acc
SnmpNotifyFilterMask	M	RC	M	RC
SnmpNotifyFilterType	M	RC	M	RC
SnmpNotifyFilterStorageType	M	RC	M	RC
SnmpNotifyFilterRowStatus	M	RC	M	RC
RFC 2574				
Note: The RFC 2574 MIB is ONLY accessible when the device is in SNMP Coexistence mode.				
usmStats				
Object	**CM**	**Access**	**CMTS**	**Access**
usmStatsUnsupportedSecLevels	M	RO	M	RO
usmStatsNotInTimeWindows	M	RO	M	RO
usmStatsUnknownUserNames	M	RO	M	RO

usmStatsUnknownEngineIDs	M	RO	M	RO
usmStatsWrongDigests	M	RO	M	RO
usmStatsDecryptionErrors	M	RO	M	RO
usmUser				
Object	**CM**	**Access**	**CMTS**	**Access**
usmUserSpinLock	M	RW	M	RW
usmUserTable				
Object	**CM**	**Access**	**CMTS**	**Access**
usmUserEngineID	M	N-Acc	M	N-Acc
usmUserName	M	N-Acc	M	N-Acc
usmUserSecurityName	M	RO	M	RO
usmUserCloneFrom	M	RC	M	RC
usmUserAuthProtocol	M	RC	M	RC
usmUserAuthKeyChange	M	RC	M	RC
usmUserOwnAuthKeyChange	M	RC	M	RC
usmUserPrivProtocol	M	RC	M	RC
usmUserPrivKeyChange	M	RC	M	RC
usmUserOwnPrivKeyChange	M	RC	M	RC
usmUserPublic	M	RC	M	RC
usmUserStorageType	M	RC	M	RC
usmUserStatus	M	RC	M	RC

DOCS-IF-EXT-MIB

Object	**2.0 CM in 1.0 Mode**	**Access**	**2.0 CM in 1.1 Mode**	**Access**	**2.0 CM in 2.0 Mode**	**Access**	**CMTS**	**Access**
docsIfDocsisCapability	D	RO	D	RO	N-Sup		N-Sup	
docsIfDocsisOperMode	D	RO	D	RO	N-Sup		N-Sup	
docsIfCmtsCmStatusDocsisMode	N/A	N/A	N/A	N/A	N/A	N/A	N-Sup	

DOCS-CABLE-DEVICE-TRAP-MIB								
Object	**2.0CM in 1.0 Mode**	**Access**	**2.0 CM in 1.1 Mode**	**Access**	**2.0 CM in 2.0 Mode**	**Access**	**CMTS**	**Access**
docsDevCmTrapControl	O	RW	M	RW	M	RW	NA	
docsDevCmtsTrapControl	NA		NA		NA		M	RW
docsDevCmInitTLVUnknownTrap	NA		M	ATRAP	M	ATRAP	NA	
docsDevCmDynServReqFailTrap	NA		M	ATRAP	M	ATRAP	NA	
docsDevCmDynServRspFailTrap	NA		M	ATRAP	M	ATRAP	NA	
docsDevCmDynServAckFailTrap	NA		M	ATRAP	M	ATRAP	NA	
docsDevCmBpiInitTrap	NA		M	ATRAP	M	ATRAP	NA	
docsDevCmBPKMTrap	NA		M	ATRAP	M	ATRAP	NA	
docsDevCmDynamicSATrap	NA		M	ATRAP	M	ATRAP	NA	
docsDevCmDHCPFailTrap	O	ATRAP	M	ATRAP	M	ATRAP	NA	
docsDevCmSwUpgradeInitTrap	O	ATRAP	M	ATRAP	M	ATRAP	NA	
docsDevCmSwUpgradeFailTrap	O	ATRAP	M	ATRAP	M	ATRAP	NA	
docsDevCmSwUpgradeSuccessTrap	O	ATRAP	M	ATRAP	M	ATRAP	NA	
docsDevCmSwUpgradeCVCFailTrap	O	ATRAP	M	ATRAP	M	ATRAP	NA	
docsDevCmTODFailTrap	O	ATRAP	M	ATRAP	M	ATRAP	NA	
docsDevCmDCCReqFailTrap	O	ATRAP	M	ATRAP	M	ATRAP		
docsDevCmDCCRspFailTrap	O	ATRAP	M	ATRAP	M	ATRAP		
docsDevCmDCCAckFailTrap	O	ATRAP	M	ATRAP	M	ATRAP		
docsDevCmtsInitRegReqFailTrap			NA		NA		M	ATRAP
docsDevCmtsInitRegRspFailTrap			NA		NA		M	ATRAP
docsDevCmtsInitRegAckFailTrap			NA		NA		M	ATRAP
docsDevCmtsDynServReqFailTrap			NA		NA		M	ATRAP
docsDevCmtsDynServRspFailTrap			NA		NA		M	ATRAP
docsDevCmtsDynServAckFailTrap			NA		NA		M	ATRAP
docsDevCmtsBpiInitTrap			NA		NA		M	ATRAP
docsDevCmtsBPKMTrap			NA		NA		M	ATRAP
docsDevCmtsDynamicSATrap			NA		NA		M	ATRAP
docsDevCmtsDCCReqFailTrap			NA		NA		M	ATRAP
docsDevCmtsDCCRspFailTrap			NA		NA		M	ATRAP
docsDevCmtsDCCAckFailTrap			NA		NA		M	ATRAP

A.1 RFI-MIB-IPCDN-DRAFT ifTable MIB-Object details

RFI-MIB-IPCDN-DRAFT MIB-Object details for Cable Device using 10 Meg Ethernet	**CMTS-Ethernet-10**	**CMTS-MAC**	**CMTS-Downstream**	**CMTS-Upstream Physical Interface**	**CMTS-Upstream Logical Channel**	**CM-Ethernet-10**	**CM-MAC**	**CM-Downstream**	**CM-Upstream**	**CM-USB**	**CM-CPEOther Type**
ifIndex: "A unique value, greater than zero, for each interface. It is recommended that values are assigned contiguously starting from 1. [The Primary CPE MUST be Interface number 1] The value for each interface sub-layer must remain constant at least from one reinitialization of the entity's network management system to the next reinitialization."	(n)	(n)	(n)	(n)	(n)	1 or [4+(n)]	2	3	4	1 or [4+(n)]	1 or [4+(n)]
ifType: "The type of interface. Additional values for ifType are assigned by the Internet Assigned Numbers Authority (IANA), through updating the syntax of the IANAifType textual convention."	6	127	128	129	205	6	127	128	129	160	(IANA num)
ifSpeed: "An estimate of the interface's current bandwidth in bits per second. [For RF Downstream; This is the symbol rate multiplied by the number of bits per symbol. For RF Upstream; This is the raw band-width in bits per second of this interface, regarding the highest speed modulation profile that is defined. This is the symbol rate multiplied with the number of bits per symbol for this modulation profile. For MAC Layer; Return zero.] For interfaces which do not vary in bandwidth or for those where no accurate estimation can be made, this object should contain the nominal bandwidth. If the bandwidth of the interface is greater than the maximum value reportable by this object then this object should report its maximum value (4,294,967,295) and ifHighSpeed must be used to report the interface's speed. For a sub-layer which has no concept of bandwidth, this object should be zero."	10,000,000	0	~64-QAM=30,341,646 ~256-QAM=42,884,296	(n)	(n)	10,000,000	0	~64-QAM=30,341,646 ~256-QAM=42,884,296	(n)	12,500,000	speed

RFI-MIB-IPCDN-DRAFT MIB-Object details for Cable Device using 10 Meg Ethernet	**CMTS-Ethernet-10**	**CMTS-MAC**	**CMTS-Downstream**	**CMTS-Upstream Physical Interface**	**CMTS-Upstream Logical Channel**	**CM-Ethernet-10**	**CM-MAC**	**CM-Downstream**	**CM-Upstream**	**CM-USB**	**CM-CPEOther Type**
ifHighSpeed: "An estimate of the interface's current bandwidth in units of 1,000,000 bits per second. If this object reports a value of 'n' then the speed of the interface is somewhere in the range of 'n-500,000' to `n+499,999'. [For RF Downstream; This is the symbol rate multiplied with the number of bits per symbol. For RF Upstream; This is the raw bandwidth in bits per second of this interface, regarding the highest speed modulation profile that is defined. This is the symbol rate multiplied with the number of bits per symbol for this modulation profile. For MAC Layer; Return zero.] For interfaces which do not vary in bandwidth or for those where no accurate estimation can be made, this object should contain the nominal bandwidth. For a sub-layer which has no concept of bandwidth, this object should be zero."	10	0	~64-QAM=30,~256-QAM=42	(n)*	(n)**	10	0	~64-QAM=30,~256-QAM=42	(n)	12	speed
ifPhysAddress: "The interface's address at its protocol sub-layer. [For RF Upstream/ Downstream; return empty string. For MAC Layer; return the physical address of this interface.] For example, for an 802.x interface, this object normally contains a MAC address. The interface's media-specific MIB must define the bit and byte ordering and the format of the value of this object. For interfaces which do not have such an address (e.g., a serial line), this object should contain an octet string of zero length.	Enet-MAC	CATV-MAC	Empty-String	Empty-String	Empty-String	Enet-MAC	CATV-MAC	Empty-String	Empty-String	USB-PhysAddr.	PhysAddr.

RFI-MIB-IPCDN-DRAFT MIB-Object details for Cable Device using 10 Meg Ethernet	**CMTS-Ethernet-10**	**CMTS-MAC**	**CMTS-Downstream**	**CMTS-Upstream Physical Interface**	**CMTS-Upstream Logical Channel**	**CM-Ethernet-10**	**CM-MAC**	**CM-Downstream**	**CM-Upstream**	**CM-USB**	**CM-CPEOther Type**
ifAdminStatus: "The desired state of the interface. The testing(3) state indicates that no operational packets can be passed. [For CM: When a managed system initializes, all interfaces start with ifAdminStatus in the up(1) state. As a result of explicit management action, ifAdminStatus is then changed to either the down(2) or testing(3) states (or remains in the up(1) state). For CMTS: When a managed system initializes, all interface start with ifAdminStatus in the up(1) state. As a result of either explicit management or configuration information saved via other non SNMP method (i.e. CLI commands) retained by the managed system, ifAdminStatus is then changed to either the down(2) or testing(3) states (or remains in the up(1) state).]"	Up(1), Down(2), Testing(3)	Up(1), Down(2), Testing(3)	Up(1), Down(2), Testing(3)	Up(1), Down(2), Testing(3)	Up(1), Down(2), Testing(3)	Up(1), Down(2), Testing(3)	Up(1), Down(2), Testing(3)	Up(1), Down(2), Testing(3)	Up(1), Down(2), Testing(3)	Up(1), Down(2), Testing(3)	Up(1), Down(2), Testing(3)
ifOperStatus: "The current operational state of the interface. The testing(3) state indicates that no operational packets can be passed. If ifAdminStatus is down(2) then ifOperStatus should be down(2). If ifAdminStatus is changed to up(1) then ifOperStatus should change to up(1) if the interface is ready to transmit and receive network traffic; it should change to dormant(5) if the interface is waiting for external actions (such as a serial line waiting for an incoming connection); it should remain in the down(2) state if and only if there is a fault that prevents it from going to the up(1) state; it should remain in the notPresent(6) state if the interface has missing (typically, hardware) components."	Up(1), Down(2), Testing(3), Dormant(5), notPresent(6)	Up(1), Down(2), Testing(3), Dormant(5), notPresent(6)	Up(1), Down(2), Testing(3), Dormant(5), notPresent(6)	Up(1), Down(2), Testing(3), Dormant(5), notPresent(6)	Up(1), Down(2), Testing(3), Dormant(5), notPresent(6)	Up(1), Down(2), Testing(3), Dormant(5), notPresent(6)	Up(1), Down(2), Testing(3), Dormant(5), notPresent(6)	Up(1), Down(2), Testing(3), Dormant(5), notPresent(6)	Up(1), Down(2), Testing(3), Dormant(5), notPresent(6)	Up(1), Down(2), Testing(3), Dormant(5), notPresent(6)	Up(1), Down(2), Testing(3), Dormant(5), notPresent(6)

RFI-MIB-IPCDN-DRAFT MIB-Object details for Cable Device using 10 Meg Ethernet	**CMTS-Ethernet-10**	**CMTS-MAC**	**CMTS-Downstream**	**CMTS-Upstream Physical Interface**	**CMTS-Upstream Logical Channel**	**CM-Ethernet-10**	**CM-MAC**	**CM-Downstream**	**CM-Upstream**	**CM-USB**	**CM-CPEOther Type**
ifMtu: "The size of the largest packet which can be sent/received on the interface, specified in octets. [For RF Upstream/Downstream; the value includes the length of the MAC header. For MAC Layer; return 1500.] For interfaces that are used for transmitting network datagrams, this is the size of the largest network datagram that can be sent on the interface."	1500	1500	1764	1764	1764	1500	1500	1764	1764	1500	1500?
ifInOctets: "The total number of octets received on the interface, including framing characters. [For RF Upstream/Downstream (where not zero*); This includes MAC packets as well as data packets, and includes the length of the MAC header, this does not include any PHY overhead. For MAC Layer; The total number of data octets received on this interface, targeted for upper protocol layers. For MAC; The total number of data octets (bridge data, data target for the managed device) received on this interface from RF-downstream interface and before application of protocol filters defined in RFC 2669.] Discontinuities in the value of this counter can occur at re-initialization of the management system, and at other times as indicated by the value of ifCounter DiscontinuityTime."	(n)	(n)	MUST be 0	(n)	(n)	(n)	(n) = low 32-bits of the 64-bit count	(n) = low 32-bits of the 64-bit count	MUST be 0	(n)	(n)

RFI-MIB-IPCDN-DRAFT MIB-Object details for Cable Device using 10 Meg Ethernet	**CMTS-Ethernet-10**	**CMTS-MAC**	**CMTS-Downstream**	**CMTS-Upstream Physical Interface**	**CMTS-Upstream Logical Channel**	**CM-Ethernet-10**	**CM-MAC**	**CM-Downstream**	**CM-Upstream**	**CM-USB**	**CM-CPEOther Type**
IfHCInOctets: (usage**) "The total number of octets received on the interface, including framing characters. [For RF Upstream/Downstream (where not zero*); This includes MAC packets as well as data packets, and includes the length of the MAC header, this does not include any PHY overhead. For MAC Layer; The total number of data octets received on this interface, targeted for upper protocol layers.] This object is a 64-bit version of ifInOctets. Discontinuities in the value of this counter can occur at re-initialization of the management system, and at other times as indicated by the value of ifCounter DiscontinuityTime."	0 or (n) = 64-bit count	0 or (n) = 64-bit count ***	MUST be 0	0 or (n) = 64-bit count ***	0 or (n) = 64-bit count ***	0 or (n) = 64-bit count ***	(n) = 64-bit count	(n) = 64-bit count	MUST be 0	0 or (n) = 64-bit count ***	0 or (n) = 64-bit count ***
ifOutOctets: "The total number of octets transmitted out of the interface, including framing characters. [For RF Upstream/ Downstream (where not zero*); This includes MAC packets as well as data packets, and includes the length of the MAC header, this does not include any PHY overhead. For MAC Layer; The total number of octets, received from upper protocol layers and transmitted on this interface. For MAC; The total number of data octets (bridge data, data generated from the managed device) transmitted on this interface to RF-upstream interface after application of protocol filters defined in RFC 2669.] Discontinuities in the value of this counter can occur at re-initialization of the management system, and at other times as indicated by the value of ifCounter DiscontinuityTime."	(n)	(n) = low 32-bits of the 64-bit count	(n) = low 32-bits of the 64-bit count	MUST be 0	MUST be 0	(n)	(n)	MUST be 0	(n)	(n)	(n)

RFI-MIB-IPCDN-DRAFT MIB-Object details for Cable Device using 10 Meg Ethernet	**CMTS-Ethernet-10**	**CMTS-MAC**	**CMTS-Downstream**	**CMTS-Upstream Physical Interface**	**CMTS-Upstream Logical Channel**	**CM-Ethernet-10**	**CM-MAC**	**CM-Downstream**	**CM-Upstream**	**CM-USB**	**CM-CPEOther Type**
ifHCOutOctets: (usage**) "The total number of octets transmitted out of the interface, including framing characters. [For RF Upstream/ Downstream (where not zero*); This includes MAC packets as well as data packets, and includes the length of the MAC header, this does not include any PHY overhead. For MAC Layer; The total number of octets, received from upper protocol layers and transmitted on this interface.] This object is a 64-bit version of ifOutOctets. Discontinuities in the value of this counter can occur at re-initialization of the management system, and at other times as indicated by the value of ifCounter DiscontinuityTime."	0 or (n) = 64-bit count ***	(n) = 64-bit count	(n) = 64-bit count	MUST be 0	MUST be 0	0 or (n) = 64-bit count ***	0 or (n) = 64-bit count ***	MUST be 0	0 or (n) = 64-bit count ***	0 or (n) = 64-bit count ***	0 or (n) = 64-bit count ***
ifInUcastPkts: "The number of packets, delivered by this sub-layer to a higher (sub-) layer, which were not addressed to a multicast or broadcast address at this sublayer. [For RF Upstream/ Downstream (where not zero*); This includes data packets as well as MAC layer packets, this does not include any PHY overhead. For MAC Layer; The number of Unicast packets received on this interface, targeted for upper protocol layers. For MAC layer; the number of Unicast data packets (bridge data, data target for the managed device) received on this interface from RF-downstream interface before application of protocol filters defined in RFC 2669.] Discontinuities in the value of this counter can occur at re-initialization of the management system, and at other times as indicated by the value of ifCounter DiscontinuityTime."	(n)	(n)	MUST be 0	(n)	(n)	(n)	(n)	MUST be 0	MUST be 0	(n)	(n)

RFI-MIB-IPCDN-DRAFT MIB-Object details for Cable Device using 10 Meg Ethernet	CMTS-Ethernet-10	CMTS-MAC	CMTS-Downstream	CMTS-Upstream Physical Interface	CMTS-Upstream Logical Channel	CM-Ethernet-10	CM-MAC	CM-Downstream	CM-Upstream	CM-USB	CM-CPEOther Type
ifHCInUcastPkts: "The number of packets, delivered by this sub-layer to a higher (sub-) layer, which were not addressed to a multicast or broadcast address at this sub-layer. [For RF Upstream/ Downstream (where not zero*); This includes data packets as well as MAC layer packets, this does not include any PHY overhead. For MAC Layer; The number of Unicast packets received on this interface, targeted for upper protocol layers. For MAC layer; the number of Unicast data packets (bridge data, data target for the managed device) received on this interface from RF-downstream interface before application of protocol filters defined in RFC 2669.] This object is a 64-bit version of ifInUcastPkts. Discontinuities in the value of this counter can occur at re-initialization of the management system, and at other times as indicated by the value of ifCounterDiscontinuityTime."	0 or (n) = 64-bit count ***	0 or (n) = 64-bit count ***	MUST be 0	0 or (n) = 64-bit count ***	0 or (n) = 64-bit count ***	0 or (n) = 64-bit count ***	0 or (n) = 64-bit count ***	MUST be 0	MUST be 0	0 or (n) = 64-bit count ***	0 or (n) = 64-bit count ***
ifInMulticastPkts: "The number of packets, delivered by this sub-layer to a higher (sub-) layer, which were addressed to a multicast address at this sublayer. [For RF Upstream/ Downstream (where not zero*); This includes data packets as well as MAC layer packets, this does not include any PHY overhead. For MAC Layer; The number of Multicast packets received on this interface, targeted for upper protocol layers. For MAC layer; the number of Multicast data packets (bridge data, data targeted for the managed device) received on this interface from RF-downstream interface before application of protocol filter defined in RFC 2669.] For a MAC layer protocol, this includes both Group and Functional addresses. Discontinuities in the value of this counter can occur at re-initialization of the management system, and at other times as indicated by the value of ifCounterDiscontinuityTime."	(n)	(n)	MUST be 0	(n)	(n)	(n)	(n)	MUST be 0	MUST be 0	(n)	(n)

RFI-MIB-IPCDN-DRAFT MIB-Object details for Cable Device using 10 Meg Ethernet	**CMTS-Ethernet-10**	**CMTS-MAC**	**CMTS-Downstream**	**CMTS-Upstream Physical Interface**	**CMTS-Upstream Logical Channel**	**CM-Ethernet-10**	**CM-MAC**	**CM-Downstream**	**CM-Upstream**	**CM-USB**	**CM-CPEOther Type**
ifHCInMulticastPkts: "The number of packets, delivered by this sub-layer to a higher (sub-) layer, which were addressed to a multicast address at this sub-layer. [For RF Upstream/ Downstream (where not zero*); This includes data packets as well as MAC layer packets, this does not include any PHY overhead. For MAC Layer; The number of Multicast packets received on this interface, targeted for upper protocol layers. For MAC layer; the number of Multicast data packets (bridge data, data targeted for the managed device) received on this interface from RF-downstream interface before application of protocol filter defined in RFC 2669.] For a MAC layer protocol, this includes both Group and Functional addresses. This object is a 64-bit version of ifInMulticastPkts. Discontinuities in the value of this counter can occur at re-initialization of the management system, and at other times as indicated by the value of ifCounter DiscontinuityTime."	0 or (n) = 64-bit count ***	0 or (n) = 64-bit count ***	MUST be 0	0 or (n) = 64-bit count ***	0 or (n) = 64-bit count ***	0 or (n) = 64-bit count ***	0 or (n) = 64-bit count ***	MUST be 0	MUST be 0	0 or (n) = 64-bit count ***	0 or (n) = 64-bit count ***
ifInBroadcastPkts: "The number of packets, delivered by this sub-layer to a higher (sub-) layer, which were addressed to a broadcast address at this sub-layer. [For RF Upstream/ Downstream (where not zero*); This includes data packets as well as MAC layer packets, this does not include any PHY overhead. For MAC Layer; The number of Broadcast packets received on this interface, targeted for upper protocol layers. For MAC layer; The number of Broadcast data packets (bridge data, data targeted for the managed device) received on this interface from RF-downstream interface before application of protocol filter defined in RFC 2669.] Discontinuities in the value of this counter can occur at re-initialization of the management system, and at other times as indicated by the value of ifCounter DiscontinuityTime."	(n)	(n)	MUST be 0	(n)	(n)	(n)	(n)	(n)	MUST be 0	(n)	(n)

RFI-MIB-IPCDN-DRAFT MIB-Object details for Cable Device using 10 Meg Ethernet	**CMTS-Ethernet-10**	**CMTS-MAC**	**CMTS-Downstream**	**CMTS-Upstream Physical Interface**	**CMTS-Upstream Logical Channel**	**CM-Ethernet-10**	**CM-MAC**	**CM-Downstream**	**CM-Upstream**	**CM-USB**	**CM-CPEOther Type**
ifHCInBroadcastPkts: "The number of packets, delivered by this sub-layer to a higher (sub-) layer, which were addressed to a broadcast address at this sub-layer. [For RF Upstream/ Downstream (where not zero*); This includes data packets as well as MAC layer packets, this does not include any PHY overhead. For MAC Layer; The number of Broadcast packets received on this interface, targeted for upper protocol layers. For MAC layer; The number of Broadcast data packets (bridge data, data targeted for the managed device) received on this interface from RF-downstream interface before application of protocol filter defined in RFC 2669.] This object is a 64-bit version of ifInBroadcastPkts. Discontinuities in the value of this counter can occur at re-initialization of the management system, and at other times as indicated by the value of ifCounterDiscontinuity Time."	0 or (n) = 64-bit count ***	0 or (n) = 64-bit count ***	MUST be 0	0 or (n) = 64-bit count ***	0 or (n) = 64-bit count ***	0 or (n) = 64-bit count ***	0 or (n) = 64-bit count ***	0 or (n) = 64-bit count ***	MUST be 0	0 or (n) = 64-bit count ***	0 or (n) = 64-bit count ***
ifInDiscards: "The number of inbound packets which were chosen to be discarded even though no errors had been detected to prevent their being deliverable to a higher-layer protocol. One possible reason for discarding such a packet could be to free up buffer space. Discontinuities in the value of this counter can occur at re-initialization of the management system, and at other times as indicated by the value of ifCounterDiscontinuityTime."	(n)	(n)	MUST be 0	(n)	(n)	(n)	(n)	(n)	MUST be 0	(n)	(n)

RFI-MIB-IPCDN-DRAFT MIB-Object details for Cable Device using 10 Meg Ethernet	**CMTS-Ethernet-10**	**CMTS-MAC**	**CMTS-Downstream**	**CMTS-Upstream Physical Interface**	**CMTS-Upstream Logical Channel**	**CM-Ethernet-10**	**CM-MAC**	**CM-Downstream**	**CM-Upstream**	**CM-USB**	**CM-CPEOther Type**
ifInErrors: "For packet-oriented interfaces, the number of inbound packets that contained errors preventing them from being deliverable to a higher-layer protocol. For character-oriented or fixed-length interfaces, the number of inbound transmission units that contained errors preventing them from being deliverable to a higher-layer protocol.Discontinuities in the value of this counter can occur at re-initialization of the management system, and at other times as indicated by the value of ifCounter DiscontinuityTime."	(n)	(n)	MUST be 0	(n)	(n)	(n)	(n)	(n)	MUST be 0	(n)	(n)
ifInUnknownProtos: "For packet-oriented interfaces, the number of packets received via the interface which were discarded because of an unknown or unsupported protocol. For character-oriented or fixed-length interfaces that support protocol multiplexing the number of transmission units received via the interface which were discarded because of an unknown or unsupported protocol. For any interface that does not support protocol multiplexing, this counter will always be 0. Discontinuities in the value of this counter can occur at re-initialization of the management system, and at other times as indicated by the value of ifCounter DiscontinuityTime."	(n)	(n)	MUST be 0	(n)	(n)	(n)	(n)	(n)	MUST be 0	(n)	(n)

RFI-MIB-IPCDN-DRAFT MIB-Object details for Cable Device using 10 Meg Ethernet	**CMTS-Ethernet-10**	**CMTS-MAC**	**CMTS-Downstream**	**CMTS-Upstream Physical Interface**	**CMTS-Upstream Logical Channel**	**CM-Ethernet-10**	**CM-MAC**	**CM-Downstream**	**CM-Upstream**	**CM-USB**	**CM-CPEOther Type**
ifOutUcastPkts: "The total number of packets that higher-level protocols requested be transmitted, and which were not addressed to a multicast or broadcast address at this sub-layer, including those that were discarded or not sent. [For RF Upstream/ Downstream (where not zero*); This includes MAC packets as well as data packets, this does not include any PHY overhead. For MAC Layer; The number of Unicast packets, received from upper protocol layers and transmitted on this interface. For MAC layer; The number of Unicast data packets (bridge data, data generated from the managed device) transmitted on this interface to RF-upstream interface after application of protocol filters defined in RFC 2669.] Discontinuities in the value of this counter can occur at re-initialization of the management system, and at other times as indicated by the value of ifCounter DiscontinuityTime."	(n)	(n)	(n)	MUST be 0	MUST be 0	(n)	(n)	MUST be 0	(n)	(n)	(n)

RFI-MIB-IPCDN-DRAFT MIB-Object details for Cable Device using 10 Meg Ethernet	**CMTS-Ethernet-10**	**CMTS-MAC**	**CMTS-Downstream**	**CMTS-Upstream Physical Interface**	**CMTS-Upstream Logical Channel**	**CM-Ethernet-10**	**CM-MAC**	**CM-Downstream**	**CM-Upstream**	**CM-USB**	**CM-CPEOther Type**
ifHCOutUcastPkts: "The total number of packets that higher-level protocols requested be transmitted, and which were not addressed to a multicast or broadcast address at this sub-layer, including those that were discarded or not sent. [For RF Upstream/ Downstream (where not zero*); This includes MAC packets as well as data packets, this does not include any PHY overhead. For MAC Layer; The number of Unicast packets, received from upper protocol layers and transmitted on this interface. For MAC layer; The number of Unicast data packets (bridge data, data generated from the managed device) transmitted on this interface to RF-upstream interface after application of protocol filters defined in RFC 2669.] This object is a 64-bit version of ifOutUcastPkts. Discontinuities in the value of this counter can occur at re- initialization of the management system, and at other times as indicated by the value of ifCounterDiscontinuityTime."	0 or (n) = 64-bit count ***	0 or (n) = 64-bit count ***	0 or (n) = 64-bit count ***	MUST be 0	MUST be 0	0 or (n) = 64-bit count ***	0 or (n) = 64-bit count ***	MUST be 0	0 or (n) = 64-bit count ***	0 or (n) = 64-bit count ***	0 or (n) = 64-bit count ***

RFI-MIB-IPCDN-DRAFT MIB-Object details for Cable Device using 10 Meg Ethernet	**CMTS-Ethernet-10**	**CMTS-MAC**	**CMTS-Downstream**	**CMTS-Upstream Physical Interface**	**CMTS-Upstream Logical Channel**	**CM-Ethernet-10**	**CM-MAC**	**CM-Downstream**	**CM-Upstream**	**CM-USB**	**CM-CPEOther Type**
ifOutMulticastPkts: "The total number of packets that higher-level protocols requested be transmitted, and which were addressed to a multicast address at this sub-layer, including those that were discarded or not sent. [For RF Upstream/ Downstream (where not zero*); This includes MAC packets as well as data packets, this does not include any PHY overhead. For MAC Layer; The number of Multicast packets received from upper protocol layers and transmitted on this interface. For MAC layer; The number of Multicast data packets (bridge data, data generated from the managed device) transmitted on this interface to RF-upstream interface after application of protocol filters defined in RFC 2669.] For a MAC layer protocol, this includes both Group and Functional addresses. Discontinuities in the value of this counter can occur at reinitialization of the management system, and at other times as indicated by the value of ifCounter DiscontinuityTime."	(n)	(n)	(n)	MUST be 0	MUST be 0	(n)	(n)	MUST be 0	MUST be 0	(n)	(n)

RFI-MIB-IPCDN-DRAFT MIB-Object details for Cable Device using 10 Meg Ethernet	**CMTS-Ethernet-10**	**CMTS-MAC**	**CMTS-Downstream**	**CMTS-Upstream Physical Interface**	**CMTS-Upstream Logical Channel**	**CM-Ethernet-10**	**CM-MAC**	**CM-Downstream**	**CM-Upstream**	**CM-USB**	**CM-CPEOther Type**
ifHCOutMulticastPkts: "The total number of packets that higher-level protocols requested be transmitted, and which were addressed to a multicast address at this sub-layer, including those that were discarded or not sent. [For RF Upstream/ Downstream (where not zero*); This includes MAC packets as well as data packets, this does not include any PHY overhead. For MAC Layer; The number of Multicast packets received from upper protocol layers and transmitted on this interface. For MAC layer; The number of Multicast data packets (bridge data, data generated from the managed device) transmitted on this interface to RF-upstream interface after application of protocol filters defined in RFC 2669.] For a MAC layer protocol, this includes both Group and Functional addresses. This object is a 64-bit version of ifOutMulticastPkts. Discontinuities in the value of this counter can occur at re-initialization of the management system, and at other times as indicated by the value of ifCounter DiscontinuityTime."	0 or (n) = 64-bit count ***	0 or (n) = 64-bit count ***	0 or (n) = 64-bit count ***	MUST be 0	MUST be 0	0 or (n) = 64-bit count ***	0 or (n) = 64-bit count ***	MUST be 0	MUST be 0	0 or (n) = 64-bit count ***	0 or (n) = 64-bit count ***

RFI-MIB-IPCDN-DRAFT MIB-Object details for Cable Device using 10 Meg Ethernet	**CMTS-Ethernet-10**	**CMTS-MAC**	**CMTS-Downstream**	**CMTS-Upstream Physical Interface**	**CMTS-Upstream Logical Channel**	**CM-Ethernet-10**	**CM-MAC**	**CM-Downstream**	**CM-Upstream**	**CM-USB**	**CM-CPEOther Type**
ifOutBroadcastPkts: "The total number of packets that higher-level protocols requested be transmitted, and which were addressed to a broadcast address at this sub-layer, including those that were discarded or not sent. [For RF Upstream/ Downstream (where not zero*); This includes MAC packets as well as data packets, this does not include any PHY overhead. For MAC Layer; The number of Broadcast packets, received from upper protocol layers and transmitted on this interface. For MAC layer; The number of Broadcast data packets (bridge data, data generated from the managed device) transmitted on this interface to RF-upstream interface after application of protocol filters defined in RFC 2669.] Discontinuities in the value of this counter can occur at re-initialization of the management system, and at other times as indicated by the value of ifCounter DiscontinuityTime."	(n)	(n)	(n)	MUST be 0	MUST be 0	(n)	(n)	MUST be 0	MUST be 0	(n)	(n)

RFI-MIB-IPCDN-DRAFT MIB-Object details for Cable Device using 10 Meg Ethernet	**CMTS-Ethernet-10**	**CMTS-MAC**	**CMTS-Downstream**	**CMTS-Upstream Physical Interface**	**CMTS-Upstream Logical Channel**	**CM-Ethernet-10**	**CM-MAC**	**CM-Downstream**	**CM-Upstream**	**CM-USB**	**CM-CPEOther Type**
ifHCOutBroadcastPkts: "The total number of packets that higher-level protocols requested be transmitted, and which were addressed to a broadcast address at this sub-layer, including those that were discarded or not sent. [For RF Upstream/ Downstream (where not zero*); This includes MAC packets as well as data packets, this does not include any PHY overhead. For MAC Layer; The number of Broadcast packets, received from upper protocol layers and transmitted on this interface. For MAC layer; The number of Broadcast data packets (bridge data, data generated from the managed device) transmitted on this interface to RF-upstream interface after application of protocol filters defined in RFC 2669.] This object is a 64-bit version of ifOutBroadcastPkts. Discontinuities in the value of this counter can occur at re-initialization of the management system, and at other times as indicated by the value of ifCounterDiscontinuityTime."	0 or (n) = 64-bit count ***	0 or (n) = 64-bit count ***	0 or (n) = 64-bit count ***	MUST be 0	MUST be 0	0 or (n) = 64-bit count ***	0 or (n) = 64-bit count ***	MUST be 0	MUST be 0	0 or (n) = 64-bit count ***	0 or (n) = 64-bit count ***
ifOutDiscards: "The number of outbound packets which were chosen to be discarded even though no errors had been detected to prevent their being transmitted. One possible reason for discarding such a packet could be to free up buffer space. Discontinuities in the value of this counter can occur at re-initialization of the management system, and at other times as indicated by the value of ifCounter DiscontinuityTime."	(n)	(n)	(n)	MUST be 0	MUST be 0	(n)	(n)	MUST be 0	(n)	(n)	(n)

RFI-MIB-IPCDN-DRAFT MIB-Object details for Cable Device using 10 Meg Ethernet	**CMTS-Ethernet-10**	**CMTS-MAC**	**CMTS-Downstream**	**CMTS-Upstream Physical Interface**	**CMTS-Upstream Logical Channel**	**CM-Ethernet-10**	**CM-MAC**	**CM-Downstream**	**CM-Upstream**	**CM-USB**	**CM-CPEOther Type**
ifOutErrors: "For packet-oriented interfaces, the number of outbound packets that could not be transmitted because of errors. For character-oriented or fixed-length interfaces, the number of outbound transmission units that could not be transmitted because of errors. Discontinuities in the value of this counter can occur at re-initialization of the management system, and at other times as indicated by the value of ifCounterDiscontinuityTime."	(n)	(n)	(n)	MUST be 0	MUST be 0	(n)	(n)	MUST be 0	(n)	(n)	(n)
ifPromiscuousMode: "This object has a value of false(2) if this interface only accepts packets/frames that are addressed to this station. This object has a value of true(1) when the station accepts all packets/frames transmitted on the media. The value true(1) is only legal on certain types of media. If legal, setting this object to a value of true(1) may require the interface to be reset before becoming effective. The value of ifPromiscuousMode does not affect the reception of broadcast and multicast packets/frames by the interface."	true(1) false(2)	true(1) false(2)	false(2)	true(1) false(2)	true(1) false(2)	true(1) false(2)	true(1) false(2)	true(1) false(2)	false(2)	true(1) false(2)	true(1) false(2)

Annex B IPDR Standards Submission for Cable Data Systems Subscriber Usage Billing Records (normative)

B.1 Service Definition

Cable Data Systems consist of Cable Modem Termination Systems (CMTSes) (located at a Multiple Service Operator's (MSO) head-end office) that provide broadband Internet access to subscribers connected via Cable Modems (CMs) through the cable plant. These Cable Data Systems comply with the Data-Over-Cable Service Interface Specifications (DOCSIS) sponsored by Cable Television Laboratories, Inc. The IPDR format for Cable Data Systems Subscriber Usage Billing Records specified herein support the DOCSIS 2.0 Operations Support System Interface specification (OSSI). The DOCSIS 2.0 OSSI requires the CMTS to provide usage-billing records for all bandwidth consumed by the subscribers connected to it via their Cable Modems when polled by the MSO's billing or mediation system.

B.1.1 Service Requirements

1. Cable Data Service is "always on". Thus, from the CMTS perspective, there are no subscriber logon events to track, but rather, in a manner similar to electric power utilities, there are only data traffic flows to meter and police.
2. A Cable Data Subscriber is uniquely identified by their Cable Modem MAC address (i.e., Ethernet address). Note that a CM is usually assigned a dynamic IP address via DHCP, so the IP address of a subscriber changes over time. Since the CM MAC address is constant, it must be used to identify the subscriber's usage billing records. All Internet traffic generated by the subscriber's Customer Premises Equipment (CPE) is bridged by the CM to and from the CMTS. The subscriber's packet and byte (octet) traffic counts are recorded by the CMTS in counters associated with the CM MAC address. Note that the current IP addresses of the CM and all the CPE in use during the collection interval are recorded for auditing purposes.
3. Cable Data Service is metered and enforced against a Service Level Agreement (SLA) that specifies the Quality of Service (QoS) that an MSO provides to a subscriber. An MSO typically has several Service Packages to offer to their subscribers, such as "Gold", "Silver", or "Bronze". Each of the Service Packages implements a specific SLA and is available for a specific price. A Service Package is implemented by a set of Service Flows that are known to the billing system by their Service Flow IDs (SFIDs) and Service Class Names (SCNs). Service Flows are the unit of billing data collection for a Cable Data Subscriber. In addition, since a subscriber may change their Service Package over time, it is very likely that a given subscriber will have several IPDRs, one for each Service Flow they have used during the collection interval.
4. Bandwidth in a Cable Data System is measured separately in both the downstream and upstream directions (relative to the CMTS). Each Service Flow is unidirectional and is associated with packet traffic of a specific type (e.g., TCP or UDP). Since most SLAs provide for asymmetric bandwidth guarantees, it is necessary to separate the downstream and upstream traffic flows in the billing usage records. Bandwidth used is measured in both packets and octets.
5. The bandwidth guarantee component of the SLA is enforced and metered by the CMTS with the assistance of the CM. However, the CM is not considered a trusted device because of its location on the Customer's Premises, so the CMTS is expected to provide all of the usage billing information for each subscriber connected to it.
6. Since an SLA may require the CMTS to enforce bandwidth limits by dropping or delaying packets that exceed the maximum throughput bandwidth for a Service Flow, the SLA dropped packets counters and delayed packets counters are also included in the usage records for each Service Flow. These counters are not used to compute billable subscriber usage but rather are available to the billing and customer care systems to enable "up-selling" to subscribers who try to exceed their subscribed service level. Thus, subscribers whose

usage patterns indicate a large number of dropped octets are probably candidates for an upgrade to a higher SLA that supports their true application bandwidth demands which, in turn, generates more revenue for the MSO.

7. The packet and octet values in the usage billing records are based on absolute 64-bit counters maintained in the CMTS. These counters may be reset when the CMTS system resets, therefore the CMTS System Up Time (sysUpTime) is included in the IPDRdoc so that the billing or mediation system can correlate counters that appear to regress.

B.1.2 Service Usage Attribute List

B.1.2.1 Service Session (SS)

The Service Session records the usage for a Service Consumer (i.e., Subscriber) associated with a specific Service Flow as seen at this collection interval. The standard SS attribute name **service** identifies the Service Class Name (SCN) of the Service Flow associated with this bandwidth usage. Note that the SFID for the Service Flow is recorded as a Usage Event (UE) attribute (see Section B.1.2.2, below). See Table B-1 for a summary of all service usage attribute value names.

B.1.2.1.1 Service Consumer (SC)

The Service Consumer (Subscriber) is identified by their Cable Modem MAC Address and their current Cable Modem IP address (as assigned by DHCP). The standard usage attribute value names **subscriberId** and **ipAddress** are used to record this information. Additionally, each CPE IP address that was in use during the collection interval is also recorded. A new usage attribute value name **cpeIpAddress** is used to record these addresses. Each Subscriber's SC element is identified by a unique sequential reference value.

B.1.2.1.2 Service Element (SE)

The CMTS is the single Service Element that records all of the subscriber usage in this IPDRdoc. The CMTS is identified by its IP address and its DNS host name. The standard usage attribute value names **ipAddress** and **hostName** are used to record this information. In addition, the current value of the CMTS System Up Time is included so the billing or mediation system can determine if the CMTS has been reset since the last record collection cycle. A new usage attribute value name **sysUpTime** is used to record this information. The format of sysUpTime is a 32-bit integer counting the number of hundredths of a second since the management interface of the CMTS was initialized. The SE reference ID is usually the host name of the CMTS.

B.1.2.2 Usage Entry (UE)

The Usage Entry records the absolute value of the packet and octet counters associated with a single active Service Flow for a given Subscriber (i.e., CM) as seen during this collection interval. The UE **type** keyword is **Interim** if the Service Flow is currently active or **Stop** if the Service Flow has been deleted during this collection interval. Note that the IPDR **time** value for an Interim record is always the same as the IPDRDoc **startTime** value, but a Stop record always has a **time** value earlier than the IPDRDoc.

A single UE represents the absolute bandwidth consumed by the Subscriber since the Service Flow was started. Bandwidth consumed during the interval must be computed by the billing system based on counters from adjacent collection intervals. The CMTS maintains the absolute values in 64-bit counters which are reported as usage attribute values in the IPDR formatted in ASCII decimal representation as described below. The internal 32-bit Service Flow ID is recorded as the new usage attribute value name **SFID** to faciltate correlation of counter sets for the same Service Flow in sequential IPDRDoc files.

Note well in the discussion that follows that *downstream* and *upstream* are relative to the CMTS while *receive* and *send* are relative to the CM. A Usage Entry is always seen from the Subscriber's (i.e. CM's) frame of reference, therefore receive and send are the directional modifiers of the usage attribute value names in an IPDR. In addition, since a Service Flow is unidirectional there should be either receive-counts or send-counts for that Service Flow, but not both. Note also that the directional modifiers of the usage attribute value names are the only true indicators of the Service Flow direction for the billing system as the SCN is chosen arbitrarily by the MSO and cannot be relied on to encode Service Flow direction in its name. For an **upstream Service Flow**, packet traffic is recorded as bandwidth sent from the CM to the CMTS. The bandwidth-consumed counters are in both packets and octets so the standard usage attribute value names *sendPkts* and *sendOctets* are used to record this information. For a **downstream Service Flow**, packet traffic is recorded as bandwidth received by the CM from the CMTS. The bandwidth-consumed counters are in both packets and octets so the standard usage attribute value names *recvPkts* and *recvOctets* are used to record this information. In addition, for downstream Service Flows only, the CMTS records the number of received and sent packets dropped and delayed due to the subscriber exceeding the maximum SLA bandwidth limit associated with a Service Flow. Two new usage attribute value names are needed to record this information: *recvSLADropPkts* and *recvSLADelayPkts*.

Table B-1 Service Usage Attribute Value names

Category	Attribute or Usage Attribute Value Name	Type	Presence	Units/Values	Remarks
What	service	String	Required	e.g., GoldTCPDown, BronzeUDPUp	Service Class Name (SCN) of the Service Flow
Who	subscriberId	String	Required	hh-hh-hh-hh-hh-hh	Cable Modem MAC address in dash delimited hex notation
What	SCipAddress	IPV4Addr	Required	nnn.nnn.nnn.nnn	CM's current IP Address. Canonical IP address in period delimited decimal notation
What	CPEipAddress	IPV4Addr	Required	nnn.nnn.nnn.nnn	Current IP address of a CPE using this CM. One per CPE active during the collection interval.
What	SEipAddress	IPV4Addr	Required	nnn.nnn.nnn.nnn	CMTS's IP Address. Canonical IP address in period delimited decimal notation
Who	hostName	String	Required	e.g., cmts-01.mso.com	CMTS's fully-qualified domain name
What	sysUpTime	unsignedInt	Required	nnnnnnnnn	32-bit count of hundredths of a second since system initialization, in decimal notation

Table B-1 Service Usage Attribute Value names (Continued)

Category	Attribute or Usage Attribute Value Name	Type	Presence	Units/Values	Remarks
What	type	String	Required	Interim | Stop	Interim identifies running SFs. Stop identifies deleted SFs.
What	SFID	unsignedInt	Required	nnnnnnnnn	32-bit Service Flow ID of the SF in decimal notation
What	recvOctets	unsignedLong	Required	64-bit counter, in decimal notation	Downstream Octets
What	recvPkts	unsignedLong	Required	64-bit counter, in decimal notation	Downstream packets
What	recvSLADropPkts	unsignedLong	Required	64-bit counter, in decimal notation	Downstream dropped packets exceeding SLA
What	recvSLADelayPkts	unsignedLong	Required	64-bit counter, in decimal notation	Downstream delayed packets exceeding SLA
What	sendOctets	unsignedLong	Optional	64-bit counter, in decimal notation	Upstream Octets
What	sendPkts	unsignedLong	Optional	64-bit counter, in decimal notation	Upstream packets

B.2 Example IPDR XML Subscriber Usage Billing Records

The example Subscriber Usage Billing File can be viewed easily via a standard web browser (such as Microsoft Internet Explorer 5.0) if the [NDM-U 3.1] standard XML Schema document (.xsd) is placed in the same directory as the billing file.

B.2.1 Schema

```
<?xml version = "1.0" encoding = "UTF-8"?>
<schema xmlns = "http://www.w3.org/2001/XMLSchema"
     targetNamespace = "http://www.ipdr.org/namespaces/ipdr"
     xmlns:ipdr = "http://www.ipdr.org/namespaces/ipdr"
     version = "3.0"
     elementFormDefault = "qualified"
     attributeFormDefault = "unqualified">
    <include schemaLocation = "http://www.ipdr.org/public/IPDRDoc3.0.xsd"/>
    <element name = "service" type = "string">
        <annotation>
            <documentation>
              Service Class Name (SCN) of the Service Flow
            </documentation>
        </annotation>
    </element>
    <element name = "subscriberId" type = "string">
        <annotation>
            <documentation>
              Cable Modem MAC address, in dash delimited hex notation
            </documentation>
        </annotation>
    </element>
```

```
    <element name = "SCipAddress" type = "ipdr:ipV4Addr">
        <annotation>
            <documentation>
              CM current IP address. Canonical IP address in period delimited decimal
notation.
            </documentation>
        </annotation>
    </element>
    <element name = "CPEipAddress" type = "ipdr:ipV4Addr">
        <annotation>
            <documentation>
              Current IP address of a CPE using this CM.  One per CPE active during the
collection interval.
            </documentation>
        </annotation>
    </element>
    <element name = "SEipAddress" type = "ipdr:ipV4Addr">
        <annotation>
            <documentation>
              CMTS IP address.  Canonical IP address in period delimited decimal notation.
            </documentation>
        </annotation>
    </element>
    <element name = "hostName" type = "string">
        <annotation>
            <documentation>
              CMTS fully qualified domain name
            </documentation>
        </annotation>
    </element>
    <element name = "sysUpTime" type = "ipdr:unsignedInt">
        <annotation>
            <documentation>
              32-bit count of hundredths of a second since system initialization, in decimal
notation.
            </documentation>
        </annotation>
    </element>
    <element name = "type">
        <simpleType>
            <annotation>
                <documentation>
                  Interim identifies running SFs.  Stop identifies deleted SFs.
                </documentation>
            </annotation>
            <restriction base = "string">
                <enumeration value = "Interim"/>
                <enumeration value = "Stop"/>
            </restriction>
        </simpleType>
    </element>
    <element name = "SFID" type = "ipdr:unsignedInt">
        <annotation>
            <documentation>
              32-bit Service Flow ID of the SF, in decimal notation
            </documentation>
        </annotation>
    </element>
    <element name = "recvOctets" type = "ipdr:unsignedLong">
        <annotation>
            <documentation>
              Downstream octets
            </documentation>
        </annotation>
    </element>
```

```
        <element name = "recvPkts" type = "ipdr:unsignedLong">
        <annotation>
            <documentation>
              Downstream packets
            </documentation>
        </annotation>
    </element>
    <element name = "recvSLADropPkts" type = "ipdr:unsignedLong">
        <annotation>
            <documentation>
              Downstream dropped packets exceeding SLA
            </documentation>
        </annotation>
    </element>
    <element name = "recvSLADelayPkts" type = "ipdr:unsignedLong">
        <annotation>
            <documentation>
              Downstream delayed packets exceeding SLA
            </documentation>
        </annotation>
    </element>
    <element name = "sendOctets" type = "ipdr:unsignedLong">
        <annotation>
            <documentation>
              Upstream octets
            </documentation>
        </annotation>
    </element>
    <element name = "sendPkts" type = "ipdr:unsignedLong">
        <annotation>
            <documentation>
              Upstream packets
            </documentation>
        </annotation>
    </element>
    <complexType name = "DOCSIS-1.1-Type">
        <complexContent>
            <extension base = "ipdr:IPDRType">
                <sequence>
                    <element ref = "ipdr:service"/>
                    <element ref = "ipdr:subscriberId"/>
                    <element ref = "ipdr:SCipAddress"/>
                    <element ref = "ipdr:CPEipAddress"/>
                    <element ref = "ipdr:SEipAddress"/>
                    <element ref = "ipdr:hostName"/>
                    <element ref = "ipdr:sysUpTime"/>
                    <element ref = "ipdr:type"/>
                    <element ref = "ipdr:recvOctets"/>
                    <element ref = "ipdr:recvPkts"/>
                    <element ref = "ipdr:recvSLADropPkts"/>
                    <element ref = "ipdr:recvSLADelayPkts"/>
                    <element ref = "ipdr:sendOctets" minOccurs = "0"/>
                    <element ref = "ipdr:sendPkts" minOccurs = "0"/>
                </sequence>
            </extension>
        </complexContent>
    </complexType>
</schema>
```

B.2.2 Sample Instance Document

```
<?xml version="1.0" ?>
```

```
<IPDRDoc xmlns="http://www.ipdr.org/namespaces/ipdr"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://www.ipdr.org/namespaces/ipdr DOCSIS1.1-3.0-A.0.xsd"
         docId="f9c0ca84-1111-11b2-a222-90ef-fd7354696bb"
         creationTime="2002-03-25T16:17:33Z"
         IPDRRecorderInfo="RKSxyz"
         version="3.0">
    <IPDR xsi:type=" DOCSIS-1.1-Type">
        <service>GoldTCPDown</service>
        <subscriberId>0A-1B-2C-3D-4E-5F-60</subscriberId>
        <SCipAddress>192.168.0.1</SCipAddress>
        <CPEipAddress>192.168.0.2</CPEipAddress>
        <SEipAddress>192.168.0.3</SEipAddress>
        <hostName>cmts-01.mso.com</hostName>
        <sysUpTime>123456789</sysUpTime>
        <type>Interim</type>
        <recvOctets>256</recvOctets>
        <recvPkts>4</recvPkts>
        <recvSLADropPkts>1</recvSLADropPkts>
        <recvSLADelayPkts>0</recvSLADelayPkts>
    </IPDR>
</IPDRDoc>
```

This page intentionally left blank.

Annex C SNMPv2c INFORM Request Definition for Subscriber Account Management (SAM) (normative)

The INFORM Request definition of account management will be specified in this section by the ECR/ECO/ECN process.

This page intentionally left blank.

Annex D Format and Content for Event, SYSLOG, and SNMP Trap (normative)[1]

The list in this Annex summarizes the format and content for event, syslog, and SNMP trap.

Each row specifies a possible event that may appear in the CM or CMTS. These events are to be reported by a cable device through local event logging, and may be accompanied by syslog or SNMP trap.

The first and second columns indicate in which stage the event happens. The third and fourth columns indicate the priority it is assigned in the CM or CMTS. These priorities are the same is reported in the docsDevEvLevel object in the cable device MIB and in the LEVEL field of a syslog.

The fifth column specifies the event text, which is reported in the docsDevEvText object of the cable device MIB and the text field of the syslog. The sixth column provides additional information about the event text in the fifth column. Some of the text fields include variable information. The variables are explained in the sixth column. Some of the variables are only required in the SYSLOG and are described in the sixth column too.

The next column specifies the error code. The eighth column indicates a unique identification number for the event, which is assigned to the docsDevEvId object in the MIB and the <eventId> field of a syslog . The final 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 Section 7.4.2.2.2. Please note that the algorithm in Section 7.4.2.2.2 will generate a hexadecimal number. The event IDs in this list have been converted to decimal integers from the hexadecimal number.

The syslog format is specified in Section 7.4.2.2.2, “SYSLOG message format,” on page 67 of this document.

The SNMP traps are defined in the cable device trap MIB.

To better illustrate the table, let us take the example of the first row in the section of DYNAMIC SERVICE REQUEST.

The first and second columns are “Dynamic Services” and “Dynamic Service Request”. The event priority is “Error” in a cable modem and “Warning” in a cable modem termination system. The event Id is 1392509184. The event text is “Service Add rejected - Unspecified reason”. The sixth column reads “For SYSLOG only append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)”. This is a note about the SYSLOG. That is to say, the syslog text body will be of the form “Service Add rejected - Unspecified reason - MAC addr: x1 x2 x3 x4 x5 x6”.

The last column, “TRAP NAME”, is docsDevCmDynServReqFailTrap, docsDevCmtsDynServReqFailTrap. This indicates that the event is notified by the SNMP trap docsDevCmDynServReqFailTrap in a cable modem and docsDevCmtsDynServReqFailTrap in a CMTS.

1. Table modified per oss2-n-02024, 06/10/02, ab

Process	Sub-Process	CM Priority	CMTS Priority	Event Messsage	Message Notes And Details	Error Code Set	Event ID	Trap Name
DOWNSTREAM ACQUISITION FAILED								
Init	DOWN-STREAM ACQUISI-TION	Critical		SYNC Timing Synchronization failure - Failed to acquire QAM/ QPSK symbol timing		T01.0	84000100	
Init	DOWN-STREAM ACQUISI-TION	Critical		SYNC Timing Synchronization failure - Failed to acquire FEC framing		T02.0	84000200	
Init	DOWN-STREAM ACQUISI-TION	Critical		SYNC Timing Synchronization failure, Acquired FEC framing - Failed to ac-quire MPEG2 Sync		T02.1	84000201	
Init	DOWN-STREAM ACQUISI-TION	Critical		SYNC Timing Synchronization failure - Failed to acquire MAC framing		T03.0	84000300	
Init	DOWN-STREAM ACQUISI-TION	Critical		SYNC Timing Synchronization failure - Failed to receive MAC SYNC frame within time-out period		T04.0	84000400	
Init	DOWN-STREAM ACQUISI-TION	Critical		SYNC Timing Synchronization failure - Loss of Sync		T05.0	84000500	
FAILED TO OBTAIN UPSTREAM PARAMETERS								
Init	OBTAIN UP-STREAM PARAME-TERS	Critical		No UCDs Re-ceived - Timeout		U01.0	85000100	
Init	OBTAIN UP-STREAM PARAME-TERS	Critical		UCD invalid or channel unus-able		U02.0	85000200	
Init	OBTAIN UP-STREAM PARAME-TERS	Critical		UCD & SYNC valid - NO MAPS for this channel		U04.0	85000400	

Process	Sub-Process	CM Priority	CMTS Priority	Event Messsage	Message Notes And Details	Error Code Set	Event ID	Trap Name
Init	OBTAIN UP-STREAM PARAME-TERS	Critical		US channel wide parameters not set before Burst Descriptors		U06.0	85000600	
MAP Upstream Bandwidth Allocation								
Any	Any	informa-tional	Infor-mation-al	A transmit op-portunity was missed because the MAP arrived too late.		M01.0	77000100	
RANGING FAILED : RNG-REQ RANGING REQUEST								
Init	RANGING	Critical		No Maintenance Broadcasts for Ranging oppor-tunities received - T2 time-out		R01.0	82000100	
Init	RANGING	Critical		Received Re-sponse to Broadcast Main-tenance Re-quest, But no Unicast Mainte-nance opportuni-ties received - T4 timeout		R04.0	82000400	
Init	RANGING		Warn-ing	No Ranging Re-quests received from POLLED CM (CMTS gen-erated polls).		R101.0	82010100	
Init	RANGING		Warn-ing	Retries exhaust-ed for polled CM (report MAC ad-dress). After 16 R101.0 errors.		R102.0	82010200	
Init	RANGING		Warn-ing	Unable to Suc-cessfully Range CM (report MAC address) Re-tries Exhausted.	Note: this is different from R102.0 in that it was able to try, i.e. got REQs but failed to Range prop-erly.	R103.0	82010300	
Init	RANGING		Warn-ing	Failed to re-ceive Periodic RNG-REQ from modem (SID X), timing-out SID.		R104.0	82010400	
RANGING FAILED : RNG-RSP RANGING RESPONSE								
Init	RANGING	Critical		No Ranging Re-sponse received - T3 time-out		R02.0	82000200	

Process	Sub-Process	CM Priority	CMTS Priority	Event Messsage	Message Notes And Details	Error Code Set	Event ID	Trap Name
Init	RANGING	Critical		Ranging Request Retries exhausted		R03.0	82000300	
Init	RANGING	Critical		Started Unicast Maintenance Ranging - No Response received - T3 timeout		R05.0	82000500	
Init	RANGING	Critical		Unicast Maintenance Ranging attempted - No response - Retries exhausted		R06.0	82000600	
Init	RANGING	Critical		Unicast Ranging Received Abort Response - Re-initializing MAC		R07.0	82000700	
TOD FAILED Before Registration								
Init	TOD	Warning		ToD request sent - No Response received		D04.1	68000401	
Init	TOD	Warning		ToD Response received - Invalid data format		D04.2	68000402	
TOD FAILED After Registration								
TOD		Error		ToD request sent- No Response received		D04.3	68000403	docsDevCmTODFailTrap
TOD		Error		ToD Response received - Invalid data format		D04.4	68000404	docsDevCmTODFailTrap
DHCP and TFTP FAILED - before registration								
Init	DHCP	Critical		DHCP FAILED - Discover sent, no offer received		D01.0	68000100	
Init	DHCP	Critical		DHCP FAILED - Request sent, No response		D02.0	68000200	
Init	DHCP	Critical		DHCP FAILED - Requested Info not supported.		D03.0	68000300	
Init	DHCP	Critical		DHCP FAILED - Response doesn't contain ALL the valid fields as described in the RFI spec Annex D		D03.1	68000301	
Init	TFTP	Critical		TFTP failed - Request sent - No Response		D05.0	68000500	

Process	Sub-Process	CM Priority	CMTS Priority	Event Messsage	Message Notes And Details	Error Code Set	Event ID	Trap Name
Init	TFTP	Critical		TFTP failed - configuration file NOT FOUND	For SYS-LOG only: append: File name = <P1> P1 = requested file name	D06.0	68000600	
Init	TFTP	Critical		TFTP Failed - OUT OF OR-DER packets		D07.0	68000700	
Init	TFTP	Critical		TFTP file complete - but failed Message Integrity check MIC	For SYS-LOG only: append: File name = <P1> P1 = filename of TFTP file	D08.0	68000800	
Init	TFTP	Critical		TFTP file complete - but missing mandatory TLV		D09.0	68000900	
Init	TFTP	Critical		TFTP Failed - file too big		D10.0	68001000	
Init	TFTP	Critical		TFTP file complete- but doesn't enable 2.0 Mode - conflicts with current US channel type	For SYS-LOG only: append: File name = <P1> P1 = filename of TFTP file	D11.0	68001100	
REGISTRATION FAILED (REG-REQ REGISTRATION REQUEST)								
Init	REGIS-TRATION REQUEST		Warn-ing	Service unavail-able - Other	For CMTS SYSLOG only, ap-pend: MAC Addr: <P1>. P1 = CM MAC ad-dress	I04.0	73000400	docsDevCmtsI nitRegReqFail Trap
Init	REGIS-TRATION REQUEST		Warn-ing	Service unavail-able - Unrecog-nized configura-tion setting	For CMTS SYSLOG only, ap-pend: MAC Addr: <P1>. P1 = CM MAC ad-dress	I04.1	73000401	docsDevCmtsI nitRegReqFail Trap
Init	REGIS-TRATION REQUEST		Warn-ing	Service unavail-able - Tempo-rarily unavailable	For CMTS SYSLOG only, ap-pend: MAC Addr: <P1>. P1 = CM MAC ad-dress	I04.2	73000402	docsDevCmtsI nitRegReqFail Trap

Process	Sub-Process	CM Priority	CMTS Priority	Event Messsage	Message Notes And Details	Error Code Set	Event ID	Trap Name
Init	REGIS-TRATION REQUEST		Warn-ing	Service unavail-able - Perma-nent	For CMTS SYSLOG only, ap-pend: MAC Addr: <P1>. P1 = CM MAC ad-dress	I04.3	73000403	docsDevCmtsI nitRegReqFail Trap
Init	REGIS-TRATION REQUEST		Warn-ing	Registration re-jected authenti-cation failure: CMTS MIC in-valid	For CMTS SYSLOG only, ap-pend: MAC Addr: <P1>. P1 = CM MAC ad-dress	I05.0	73000500	docsDevCmtsI nitRegReqFail Trap
Init	REGIS-TRATION REQUEST		Warn-ing	REG REQ has Invalid MAC header	For CMTS SYSLOG only, ap-pend: MAC Addr: <P1>. P1 = CM MAC ad-dress	I101.0	73010100	docsDevCmtsI nitRegReqFail Trap
Init	REGIS-TRATION REQUEST		Warn-ing	REG REQ has Invalid SID or not in use	For CMTS SYSLOG only, ap-pend: MAC Addr: <P1>. P1 = CM MAC ad-dress	I102.0	73010200	docsDevCmtsI nitRegReqFail Trap
Init	REGIS-TRATION REQUEST		Warn-ing	REG REQ missed Required TLVs	For CMTS SYSLOG only, ap-pend: MAC Addr: <P1>. P1 = CM MAC ad-dress	I104.0	73010400	docsDevCmtsI nitRegReqFail Trap
Init	REGIS-TRATION REQUEST		Warn-ing	Bad DS FREQ - Format Invalid	For CMTS SYSLOG only, ap-pend: MAC Addr: <P1>. P1 = CM MAC ad-dress	I105.0	73010500	docsDevCmtsI nitRegReqFail Trap
Init	REGIS-TRATION REQUEST		Warn-ing	Bad DS FREQ - Not in use	For CMTS SYSLOG only, ap-pend: MAC Addr: <P1>. P1 = CM MAC ad-dress	I105.1	73010501	docsDevCmtsI nitRegReqFail Trap

Process	Sub-Process	CM Priority	CMTS Priority	Event Messsage	Message Notes And Details	Error Code Set	Event ID	Trap Name
Init	REGIS-TRATION REQUEST		Warn-ing	Bad DS FREQ - Not Multiple of 62500 Hz	For CMTS SYSLOG only, ap-pend: MAC Addr: <P1>. P1 = CM MAC ad-dress	I105.2	73010502	docsDevCmtsI nitRegReqFail Trap
Init	REGIS-TRATION REQUEST		Warn-ing	Bad US CH - In-valid or Unas-signed	For CMTS SYSLOG only, ap-pend: MAC Addr: <P1>. P1 = CM MAC ad-dress	I106.0	73010600	docsDevCmtsI nitRegReqFail Trap
Init	REGIS-TRATION REQUEST		Warn-ing	Bad US CH - Change fol-lowed with (RE-) Registration REQ	For CMTS SYSLOG only, ap-pend: MAC Addr: <P1>. P1 = CM MAC ad-dress	I106.1	73010601	docsDevCmtsI nitRegReqFail Trap
Init	REGIS-TRATION REQUEST		Warn-ing	Bad US CH - Overload	For CMTS SYSLOG only, ap-pend: MAC Addr: <P1>. P1 = CM MAC ad-dress	I107.0	73010700	docsDevCmtsI nitRegReqFail Trap
Init	REGIS-TRATION REQUEST		Warn-ing	Network Access has Invalid Pa-rameter	For CMTS SYSLOG only, ap-pend: MAC Addr: <P1>. P1 = CM MAC ad-dress	I108.0	73010800	docsDevCmtsI nitRegReqFail Trap
Init	REGIS-TRATION REQUEST		Warn-ing	Bad Class of Service - Invalid Configuration	For CMTS SYSLOG only, ap-pend: MAC Addr: <P1>. P1 = CM MAC ad-dress	I109.0	73010900	docsDevCmtsI nitRegReqFail Trap
Init	REGIS-TRATION REQUEST		Warn-ing	Bad Class of Service - Unsup-ported class	For CMTS SYSLOG only, ap-pend: MAC Addr: <P1>. P1 = CM MAC ad-dress	I110.0	73011000	docsDevCmtsI nitRegReqFail Trap

Process	Sub-Process	CM Priority	CMTS Priority	Event Messsage	Message Notes And Details	Error Code Set	Event ID	Trap Name
Init	REGIS-TRATION REQUEST		Warn-ing	Bad Class of Service - Invalid class ID or out of range	For CMTS SYSLOG only, ap-pend: MAC Addr: <P1>. P1 = CM MAC ad-dress	I111.0	73011100	docsDevCmtsI nitRegReqFail Trap
Init	REGIS-TRATION REQUEST		Warn-ing	Bad Max DS Bit Rate - Invalid Format	For CMTS SYSLOG only, ap-pend: MAC Addr: <P1>. P1 = CM MAC ad-dress	I112.0	73011200	docsDevCmtsI nitRegReqFail Trap
Init	REGIS-TRATION REQUEST		Warn-ing	Bad Max DS Bit Rate Unsupport-ed Setting	For CMTS SYSLOG only, ap-pend: MAC Addr: <P1>. P1 = CM MAC ad-dress	I112.1	73011201	docsDevCmtsI nitRegReqFail Trap
Init	REGIS-TRATION REQUEST		Warn-ing	Bad Max US Bit - Invalid Format	For CMTS SYSLOG only, ap-pend: MAC Addr: <P1>. P1 = CM MAC ad-dress	I113.0	73011300	docsDevCmtsI nitRegReqFail Trap
Init	REGIS-TRATION REQUEST		Warn-ing	Bad Max US Bit Rate - Unsup-ported Setting	For CMTS SYSLOG only, ap-pend: MAC Addr: <P1>. P1 = CM MAC ad-dress	I113.1	73011301	docsDevCmtsI nitRegReqFail Trap
Init	REGIS-TRATION REQUEST		Warn-ing	Bad US Priority Configuration - Invalid Format	For CMTS SYSLOG only, ap-pend: MAC Addr: <P1>. P1 = CM MAC ad-dress	I114.0	73011400	docsDevCmtsI nitRegReqFail Trap
Init	REGIS-TRATION REQUEST		Warn-ing	Bad US Priority Configuration - Setting out of Range	For CMTS SYSLOG only, ap-pend: MAC Addr: <P1>. P1 = CM MAC ad-dress	I114.1	73011401	docsDevCmtsI nitRegReqFail Trap

Process	Sub-Process	CM Priority	CMTS Priority	Event Messsage	Message Notes And Details	Error Code Set	Event ID	Trap Name
Init	REGIS-TRATION REQUEST		Warn-ing	Bad Guaran-teed Min US CH Bit rate Configu-ration setting - Invalid Format	For CMTS SYSLOG only, ap-pend: MAC Addr: <P1>. P1 = CM MAC ad-dress	I115.0	73011500	docsDevCmtsI nitRegReqFail Trap
Init	REGIS-TRATION REQUEST		Warn-ing	Bad Guaran-teed Min US CH Bit rate Configu-ration setting - Exceed Max US Bit Rate	For CMTS SYSLOG only, ap-pend: MAC Addr: <P1>. P1 = CM MAC ad-dress	I115.1	73011501	docsDevCmtsI nitRegReqFail Trap
Init	REGIS-TRATION REQUEST		Warn-ing	Bad Guaran-teed Min US CH Bit rate Configu-ration setting - Out of Range	For CMTS SYSLOG only, ap-pend: MAC Addr: <P1>. P1 = CM MAC ad-dress	I115.2	73011502	docsDevCmtsI nitRegReqFail Trap
Init	REGIS-TRATION REQUEST		Warn-ing	Bad Max US CH Transmit Burst configuration setting - Invalid Format	For CMTS SYSLOG only, ap-pend: MAC Addr: <P1>. P1 = CM MAC ad-dress	I116.0	73011600	docsDevCmtsI nitRegReqFail Trap
Init	REGIS-TRATION REQUEST		Warn-ing	Bad Max US CH Transmit Burst configuration setting - Out of Range	For CMTS SYSLOG only, ap-pend: MAC Addr: <P1>. P1 = CM MAC ad-dress	I116.1	73011601	docsDevCmtsI nitRegReqFail Trap
Init	REGIS-TRATION REQUEST		Warn-ing	Invalid Modem Capabilities con-figuration setting	For CMTS SYSLOG only, ap-pend: MAC Addr: <P1>. P1 = CM MAC ad-dress	I117.0	73011700	docsDevCmtsI nitRegReqFail Trap
Init	REGIS-TRATION REQUEST		Warn-ing	Configuration file contains param-eter with the val-ue outside of the range	For CMTS SYSLOG only, ap-pend: MAC Addr: <P1>. P1 = CM MAC ad-dress	I118.0	73011800	docsDevCmtsI nitRegReqFail Trap
Version 1.1 and 2.0 Specific REG-REQ REGISTRATION REQUEST								

Process	Sub-Process	CM Priority	CMTS Priority	Event Messsage	Message Notes And Details	Error Code Set	Event ID	Trap Name
Init	1.1 and 2.0 SPECIFIC REGISTRATION REQUEST		Warning	REG REQ rejected - Unspecified reason	For CMTS SYSLOG only, append: MAC Addr: <P1>. P1 = CM MAC address	I201.0	73020100	docsDevCmtsInitRegReqFailTrap
Init	1.1 and 2.0 SPECIFIC REGISTRATION REQUEST		Warning	REG REQ rejected - Unrecognized configuration setting	For CMTS SYSLOG only, append: MAC Addr: <P1>. P1 = CM MAC address	I201.1	73020101	docsDevCmtsInitRegReqFailTrap
Init	1.1 and 2.0 SPECIFIC REGISTRATION REQUEST		Warning	REG REQ rejected - temporary no resource	For CMTS SYSLOG only, append: MAC Addr: <P1>. P1 = CM MAC address	I201.2	73020102	docsDevCmtsInitRegReqFailTrap
Init	1.1 and 2.0 SPECIFIC REGISTRATION REQUEST		Warning	REG REQ rejected - Permanent administrative	For CMTS SYSLOG only, append: MAC Addr: <P1>. P1 = CM MAC address	I201.3	73020103	docsDevCmtsInitRegReqFailTrap
Init	1.1 and 2.0 SPECIFIC REGISTRATION REQUEST		Warning	REG REQ rejected - Required parameter not present <P1>	P1 = TLV typeIt is up to the vendor to support 1 or manyFor CMTS SYSLOG only, append: MAC Addr: <P2>. P2 = CM MAC address	I201.4	73020104	docsDevCmtsInitRegReqFailTrap
Init	1.1 and 2.0 SPECIFIC REGISTRATION REQUEST		Warning	REG REQ rejected - Header suppression setting not supported	For CMTS SYSLOG only, append: MAC Addr: <P1>. P1 = CM MAC address	I201.5	73020105	docsDevCmtsInitRegReqFailTrap

Process	Sub-Process	CM Priority	CMTS Priority	Event Messsage	Message Notes And Details	Error Code Set	Event ID	Trap Name
Init	1.1 and 2.0 SPECIFIC REGISTRATION REQUEST		Warning	REG REQ rejected - Multiple errors	For CMTS SYSLOG only, append: MAC Addr: <P1>. P1 = CM MAC address	I201.6	73020106	docsDevCmtsInitRegReqFailTrap
Init	1.1 and 2.0 SPECIFIC REGISTRATION REQUEST		Warning	REG REQ rejected - duplicate reference-ID or index in message	For CMTS SYSLOG only, append: MAC Addr: <P1>. P1 = CM MAC address	I201.7	73020107	docsDevCmtsInitRegReqFailTrap
Init	1.1 and 2.0 SPECIFIC REGISTRATION REQUEST		Warning	REG REQ rejected - parameter invalid for context <P1>	P1 = TLV parameter-For CMTS SYSLOG only, append: MAC Addr: <P2>. P2 = CM MAC address	I201.8	73020108	docsDevCmtsInitRegReqFailTrap
Init	1.1 and 2.0 SPECIFIC REGISTRATION REQUEST		Warning	REG REQ rejected - Authorization failure	For CMTS SYSLOG only, append: MAC Addr: <P1>. P1 = CM MAC address	I201.9	73020109	docsDevCmtsInitRegReqFailTrap
Init	1.1 and 2.0 SPECIFIC REGISTRATION REQUEST		Warning	REG REQ rejected - Major service flow error	For CMTS SYSLOG only, append: MAC Addr: <P2>. P2 = CM MAC address	I201.10	73020110	docsDevCmtsInitRegReqFailTrap
Init	1.1 and 2.0 SPECIFIC REGISTRATION REQUEST		Warning	REG REQ rejected - Major classifier error	For CMTS SYSLOG only, append: MAC Addr: <P2>. P2 = CM MAC address	I201.11	73020111	docsDevCmtsInitRegReqFailTrap
Init	1.1 and 2.0 SPECIFIC REGISTRATION REQUEST		Warning	REG REQ rejected - Major PHS rule error	For CMTS SYSLOG only, append: MAC Addr: <P2>. P2 = CM MAC address	I201.12	73020112	docsDevCmtsInitRegReqFailTrap

Process	Sub-Process	CM Priority	CMTS Priority	Event Messsage	Message Notes And Details	Error Code Set	Event ID	Trap Name
Init	1.1 and 2.0 SPECIFIC REGISTRATION REQUEST		Warning	REG REQ rejected - Multiple major errors	For CMTS SYSLOG only, append: MAC Addr: <P1>. P1 = CM MAC address	I201.13	73020113	docsDevCmtsInitRegReqFailTrap
Init	1.1 and 2.0 SPECIFIC REGISTRATION REQUEST		Warning	REG REQ rejected - Message syntax error <P1>	P1 = messageFor CMTS SYSLOG only, append: MAC Addr: <P2>. P2 = CM MAC address	I201.14	73020114	docsDevCmtsInitRegReqFailTrap
Init	1.1 and 2.0 SPECIFIC REGISTRATION REQUEST		Warning	REG REQ rejected - Primary service flow error <P1>	P1 = Service Flow ReferenceFor CMTS SYSLOG only, append: MAC Addr: <P2>. P2 = CM MAC address	I201.15	73020115	docsDevCmtsInitRegReqFailTrap
	1.1 and 2.0 SPECIFIC REGISTRATION REQUEST		Warning	REG REQ rejected - Message too big <P1>	P1 = # of characters-For CMTS SYSLOG only, append: MAC Addr: <P2>. P2 = CM MAC address	I201.16	73020116	docsDevCmtsInitRegReqFailTrap
REG-RSP REGISTRATION RESPONSE								
Init	REGISTRATION RESPONSE	Critical		REG-RSP - invalid format or not recognized		I01.0	73000100	
Init	REGISTRATION RESPONSE	Critical		REG RSP not received		I02.0	73000200	
Init	REGISTRATION RESPONSE	Critical		REG RSP bad SID <P1>		I03.0	73000300	
Version 1.1 and 2.0 Specific REG-RSP								

Process	Sub-Process	CM Priority	CMTS Priority	Event Messsage	Message Notes And Details	Error Code Set	Event ID	Trap Name
Init	1.1 and 2.0 SPECIFIC REGISTRATION RESPONSE	Critical		REG RSP contains service flow parameters that CM cannot support <P1>	P1 = Service Flow ID	I251.0	73025100	
Init	1.1 and 2.0 SPECIFIC REGISTRATION RESPONSE	Critical		REG RSP contains classifier parameters that CM cannot support <P1>	P1 = Service Flow ID	I251.1	73025101	
Init	1.1 and 2.0 SPECIFIC REGISTRATION RESPONSE	Critical		REG RSP contains PHS parameters that CM cannot support <P1>	P1 = Service Flow ID	I251.2	73025102	
Init	1.1 and 2.0 SPECIFIC REGISTRATION RESPONSE	Critical		Registration RSP rejected unspecified reason		I251.3	73025103	
Init	1.1 and 2.0 SPECIFIC REGISTRATION RESPONSE	Critical		Registration RSP rejected message syntax error <P1>	P1 = message	I251.4	73025104	
Init	1.1 and 2.0 SPECIFIC REGISTRATION RESPONSE	Critical		Registration RSP rejected message too big <P1>	P1 = # of characters	I251.5	73025105	
Version 2.0 Specific REG-RSP								
Init	2.0 SPECIFIC REGISTRATION RESPONSE	Warning		REG-RSP received after REG-ACK. Returning to 1.x transmit mode		I261.0	73026100	
REG-ACK REGISTRATION ACKNOWLEDGEMENT								

Process	Sub-Process	CM Priority	CMTS Priority	Event Messsage	Message Notes And Details	Error Code Set	Event ID	Trap Name
Init	REGIS-TRATION AC-KNOWL-EDGE-MENT		Warn-ing	REG aborted no REG-ACK	For CMTS SYSLOG only, ap-pend: MAC Addr: <P1>. P1 = CM MAC ad-dress	I301.0	73030100	docsDevCmtsI nitRegAckFail Trap
Init	REGIS-TRATION Acknowl-edgement		Warn-ing	REG ACK reject-ed unspecified reason	For CMTS SYSLOG only, ap-pend: MAC Addr: <P1>. P1 = CM MAC ad-dress	I302.0	73030200	docsDevCmtsI nitRegAckFail Trap
Init	REGIS-TRATION AC-KNOWL-EDGE-MENT		Warn-ing	REG ACK reject-ed message syntax error	For CMTS SYSLOG only, ap-pend: MAC Addr: <P1>. P1 = CM MAC ad-dress	I303.0	73030300	docsDevCmtsI nitRegAckFail Trap
TLV-11 Failures								
Init	TLV-11 PARSING	Notice		TLV-11 - unrec-ognized OID		I401.0	73040100	docsDevCmIni tTLVUnknown Trap
Init	TLV-11 PARSING	Critical		TLV-11 - Illegal Set operation failed		I402.0	73040200	docsDevCmIni tTLVUnknown Trap
Init	TLV-11 PARSING	Critical		TLV-11 - Failed to set duplicate elements		I403.0	73040300	docsDevCmIni tTLVUnknown Trap
SW UPGRADE INIT								
SW Up-grade	SW UP-GRADE INIT	Notice		SW Download INIT - Via NMS	For SYS-LOG only, append: SW file: <P1> - SW server: < P2>. P1 = SW file name and P2 = Tftp server IP ad-dress	E101.0	69010100	docsDevCmS wUpgradeInitT rap

Process	Sub-Process	CM Priority	CMTS Priority	Event Messsage	Message Notes And Details	Error Code Set	Event ID	Trap Name
SW Up-grade	SW UP-GRADE INIT	Notice		SW Download INIT - Via Con-fig file <P1>	P1 = CM config file nameFor SYSLOG only, ap-pend: SW file: <P2> - SW server: < P3>. P2 = SW file name and P3 = Tftp server IP ad-dress	E102.0	69010200	docsDevCmS wUpgradeInitT rap
SW UPGRADE GENERAL FAILURE								
SW Up-grade	SW UP-GRADE GENER-AL FAIL-URE	Error		SW Upgrade Failed during download - Max retry exceed (3)	For SYS-LOG only, append: SW file: <P1> - SW server: < P2>. P1 = SW file name and P2 = Tftp server IP ad-dress	E103.0	69010300	docsDevCmS wUpgradeFail Trap
SW Up-grade	SW UP-GRADE GENER-AL FAIL-URE	Error		SW Upgrade Failed Before Download - Server not Present	For SYS-LOG only, append: SW file: <P1> - SW server: < P2>. P1 = SW file name and P2 = Tftp server IP ad-dress	E104.0	69010400	docsDevCmS wUpgradeFail Trap
SW Up-grade	SW UP-GRADE GENER-AL FAIL-URE	Error		SW upgrade Failed before download - File not Present	For SYS-LOG only, append: SW file: <P1> - SW server: < P2>. P1 = SW file name and P2 = Tftp server IP ad-dress	E105.0	69010500	docsDevCmS wUpgradeFail Trap

Process	Sub-Process	CM Priority	CMTS Priority	Event Messsage	Message Notes And Details	Error Code Set	Event ID	Trap Name
SW Up-grade	SW UP-GRADE GENER-AL FAIL-URE	Error		SW upgrade Failed before download -TFTP Max Retry Ex-ceeded	For SYS-LOG only, append: SW file: <P1> - SW server: < P2>. P1 = SW file name and P2 = Tftp server IP ad-dress	E106.0	69010600	docsDevCmS wUpgradeFail Trap
SW Up-grade	SW UP-GRADE GENER-AL FAIL-URE	Error		SW upgrade Failed after download -In-compatible SW file	For SYS-LOG only, append: SW file: <P1> - SW server: < P2>. P1 = SW file name and P2 = Tftp server IP ad-dress	E107.0	69010700	docsDevCmS wUpgradeFail Trap
SW Up-grade	SW UP-GRADE GENER-AL FAIL-URE	Error		SW upgrade Failed after download - SW File corruption	For SYS-LOG only, append: SW file: <P1> - SW server: < P2>. P1 = SW file name and P2 = Tftp server IP ad-dress	E108.0	69010800	docsDevCmS wUpgradeFail Trap
SW Up-grade	SW UP-GRADE GENER-AL FAIL-URE	Error		Disruption dur-ing SW down-load - Power Failure	For SYS-LOG only, append: SW file: <P1> - SW server: < P2>. P1 = SW file name and P2 = Tftp server IP ad-dress	E109.0	69010900	docsDevCmS wUpgradeFail Trap

Process	Sub-Process	CM Priority	CMTS Priority	Event Messsage	Message Notes And Details	Error Code Set	Event ID	Trap Name
SW Upgrade	SW UPGRADE GENERAL FAILURE	Error		Disruption during SW download - RF removed	For SYSLOG only, append: SW file: <P1> - SW server: <P2>. P1 = SW file name and P2 = Tftp server IP address	E110.0	69011000	docsDevCmSwUpgradeFailTrap
SW UPGRADE SUCCESS								
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	docsDevCmSwUpgradeSuccessTrap
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	docsDevCmSwUpgradeSuccessTrap
DHCP FAILURE AFTER CM HAS REGISTERED WITH THE CMTS								
DHCP		Error		DHCP RENEW sent - No response		D101.0	68010100	docsDevCmDHCPFailTrap
DHCP		Error		DHCP REBIND sent - No response		D102.0	68010200	docsDevCmDHCPFailTrap
DHCP		Error		DHCP RENEW sent - Invalid DHCP option		D103.0	68010300	docsDevCmDHCPFailTrap
DHCP		Error		DHCP REBIND sent - Invalid DHCP option		D104.0	68010400	docsDevCmDHCPFailTrap
DYNAMIC SERVICE REQUEST								

Process	Sub-Process	CM Priority	CMTS Priority	Event Messsage	Message Notes And Details	Error Code Set	Event ID	Trap Name
DYNAMIC SERVICES	DYNAMIC SERVICE REQUEST	Error	Warning	Service Add rejected - Unspecified reason	For SYSLOG only append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S01.0	83000100	docsDevCmDynServReqFailTrap, docsDevCmtsDynServReqFailTrap
DYNAMIC SERVICES	DYNAMIC SERVICE REQUEST	Error	Warning	Service Add rejected - Unrecognized configuration setting	For SYSLOG only append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S01.1	83000101	docsDevCmDynServReqFailTrap, docsDevCmtsDynServReqFailTrap
DYNAMIC SERVICES	DYNAMIC SERVICE REQUEST	Error	Warning	Service Add rejected - Temporary no resource	For SYSLOG only append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S01.2	83000102	docsDevCmDynServReqFailTrap, docsDevCmtsDynServReqFailTrap
DYNAMIC SERVICES	DYNAMIC SERVICE REQUEST	Error	Warning	Service Add rejected - Permanent administrative	For SYSLOG only append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S01.3	83000103	docsDevCmDynServReqFailTrap, docsDevCmtsDynServReqFailTrap
DYNAMIC SERVICES	DYNAMIC SERVICE REQUEST	Error	Warning	Service Add rejected - Required parameter not present	For SYSLOG only append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S01.4	83000104	docsDevCmDynServReqFailTrap, docsDevCmtsDynServReqFailTrap
DYNAMIC SERVICES	DYNAMIC SERVICE REQUEST	Error	Warning	Service Add rejected - Header suppression setting not supported	For SYSLOG only append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S01.5	83000105	docsDevCmDynServReqFailTrap, docsDevCmtsDynServReqFailTrap

Process	Sub-Process	CM Priority	CMTS Priority	Event Messsage	Message Notes And Details	Error Code Set	Event ID	Trap Name
DYNAMIC SERVICES	DYNAMIC SERVICE REQUEST	Error		Service Add rejected - Service flow exists	For SYSLOG only append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S01.6	83000106	docsDevCmDynServReqFailTrap, docsDevCmtsDynServReqFailTrap
DYNAMIC SERVICES	DYNAMIC SERVICE REQUEST	Error	Warning	Service Add rejected - HMAC Auth failure	For SYSLOG only append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S01.7	83000107	docsDevCmDynServReqFailTrap, docsDevCmtsDynServReqFailTrap
DYNAMIC SERVICES	DYNAMIC SERVICE REQUEST	Error	Warning	Service Add rejected - Add aborted	For SYSLOG only append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S01.8	83000108	docsDevCmDynServReqFailTrap, docsDevCmtsDynServReqFailTrap
DYNAMIC SERVICES	DYNAMIC SERVICE REQUEST	Error	Warning	Service Add rejected - Multiple errors	For SYSLOG only append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S01.9	83000109	docsDevCmDynServReqFailTrap, docsDevCmtsDynServReqFailTrap
DYNAMIC SERVICES	DYNAMIC SERVICE REQUEST	Error	Warning	Service Add rejected - Classifier not found	For SYSLOG only append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S01.10	83000110	docsDevCmDynServReqFailTrap, docsDevCmtsDynServReqFailTrap
DYNAMIC SERVICES	DYNAMIC SERVICE REQUEST	Error		Service Add rejected - Classifier exists	For SYSLOG only append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S01.11	83000111	docsDevCmDynServReqFailTrap, docsDevCmtsDynServReqFailTrap

Process	Sub-Process	CM Priority	CMTS Priority	Event Messsage	Message Notes And Details	Error Code Set	Event ID	Trap Name
DYNAMIC SERVICES	DYNAMIC SERVICE REQUEST	Error	Warning	Service Add rejected - PHS rule exists	For SYSLOG only append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S01.13	83000113	docsDevCmDynServReqFailTrap, docsDevCmtsDynServReqFailTrap
DYNAMIC SERVICES	DYNAMIC SERVICE REQUEST	Error	Warning	Service Add rejected - Duplicated reference-ID or index in message	For SYSLOG only append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S01.14	83000114	docsDevCmDynServReqFailTrap, docsDevCmtsDynServReqFailTrap
DYNAMIC SERVICES	DYNAMIC SERVICE REQUEST	Error	Warning	Service Add rejected - Multiple upstream flows	For SYSLOG only append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S01.15	83000115	docsDevCmDynServReqFailTrap, docsDevCmtsDynServReqFailTrap
DYNAMIC SERVICES	DYNAMIC SERVICE REQUEST	Error	Warning	Service Add rejected - Multiple downstream flows	For SYSLOG only append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S01.16	83000116	docsDevCmDynServReqFailTrap, docsDevCmtsDynServReqFailTrap
DYNAMIC SERVICES	DYNAMIC SERVICE REQUEST	Error	Warning	Service Add rejected - Classifier for another flow	For SYSLOG only append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S01.17	83000117	docsDevCmDynServReqFailTrap, docsDevCmtsDynServReqFailTrap
DYNAMIC SERVICES	DYNAMIC SERVICE REQUEST	Error	Warning	Service Add rejected - PHS rule for another flow	For SYSLOG only append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S01.18	83000118	docsDevCmDynServReqFailTrap, docsDevCmtsDynServReqFailTrap

Process	Sub-Process	CM Priority	CMTS Priority	Event Messsage	Message Notes And Details	Error Code Set	Event ID	Trap Name
DYNAMIC SERVICES	DYNAMIC SERVICE REQUEST	Error	Warning	Service Add rejected - Parameter invalid for context	For SYSLOG only append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S01.19	83000119	docsDevCmDynServReqFailTrap, docsDevCmtsDynServReqFailTrap
DYNAMIC SERVICES	DYNAMIC SERVICE REQUEST	Error	Warning	Service Add rejected - Authorization failure	For SYSLOG only append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S01.20	83000120	docsDevCmDynServReqFailTrap, docsDevCmtsDynServReqFailTrap
DYNAMIC SERVICES	DYNAMIC SERVICE REQUEST	Error	Warning	Service Add rejected - Major service flow error	For SYSLOG only append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S01.21	83000121	docsDevCmDynServReqFailTrap, docsDevCmtsDynServReqFailTrap
DYNAMIC SERVICES	DYNAMIC SERVICE REQUEST	Error	Warning	Service Add rejected - Major classifier error	For SYSLOG only append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S01.22	83000122	docsDevCmDynServReqFailTrap, docsDevCmtsDynServReqFailTrap
DYNAMIC SERVICES	DYNAMIC SERVICE REQUEST	Error	Warning	Service Add rejected - Major PHS rule error	For SYSLOG only append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S01.23	83000123	docsDevCmDynServReqFailTrap, docsDevCmtsDynServReqFailTrap
DYNAMIC SERVICES	DYNAMIC SERVICE REQUEST	Error	Warning	Service Add rejected - Multiple major errors	For SYSLOG only append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S01.24	83000124	docsDevCmDynServReqFailTrap, docsDevCmtsDynServReqFailTrap

Process	Sub-Process	CM Priority	CMTS Priority	Event Messsage	Message Notes And Details	Error Code Set	Event ID	Trap Name
DYNAMIC SERVICES	DYNAMIC SERVICE REQUEST	Error	Warning	Service Add rejected - Message syntax error	For SYSLOG only append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S01.25	83000125	docsDevCmDynServReqFailTrap, docsDevCmtsDynServReqFailTrap
DYNAMIC SERVICES	DYNAMIC SERVICE REQUEST	Error	Warning	Service Add rejected - Message too big	For SYSLOG only append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S01.26	83000126	docsDevCmDynServReqFailTrap, docsDevCmtsDynServReqFailTrap
DYNAMIC SERVICES	DYNAMIC SERVICE REQUEST	Error	Warning	Service Add rejected - Temporary DCC	For SYSLOG only append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S01.27	83000127	docsDevCmDynServReqFailTrap, docsDevCmtsDynServReqFailTrap
DYNAMIC SERVICES	DYNAMIC SERVICE REQUEST	Error	Warning	Service Change rejected - Unspecified reason	For SYSLOG only append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S02.0	83000200	docsDevCmDynServReqFailTrap, docsDevCmtsDynServReqFailTrap
DYNAMIC SERVICES	DYNAMIC SERVICE REQUEST	Error	Warning	Service Change rejected - Unrecognized configuration setting	For SYSLOG only append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S02.1	83000201	docsDevCmDynServReqFailTrap, docsDevCmtsDynServReqFailTrap
DYNAMIC SERVICES	DYNAMIC SERVICE REQUEST	Error	Warning	Service Change rejected - Temporary no resource	For SYSLOG only append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S02.2	83000202	docsDevCmDynServReqFailTrap, docsDevCmtsDynServReqFailTrap

Process	Sub-Process	CM Priority	CMTS Priority	Event Messsage	Message Notes And Details	Error Code Set	Event ID	Trap Name
DYNAMIC SERVICES	DYNAMIC SERVICE REQUEST	Error	Warning	Service Change rejected - Permanent administrative	For SYSLOG only append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S02.3	83000203	docsDevCmDynServReqFailTrap, docsDevCmtsDynServReqFailTrap
DYNAMIC SERVICES	DYNAMIC SERVICE REQUEST	Error	Warning	Service Change rejected - Requester not owner of service flow	For SYSLOG only append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S02.4	83000204	docsDevCmDynServReqFailTrap, docsDevCmtsDynServReqFailTrap
DYNAMIC SERVICES	DYNAMIC SERVICE REQUEST	Error	Warning	Service Change rejected - Service flow not found	For SYSLOG only append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S02.5	83000205	docsDevCmDynServReqFailTrap, docsDevCmtsDynServReqFailTrap
DYNAMIC SERVICES	DYNAMIC SERVICE REQUEST	Error	Warning	Service Change rejected - Required parameter not present	For SYSLOG only append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S02.6	83000206	docsDevCmDynServReqFailTrap, docsDevCmtsDynServReqFailTrap
DYNAMIC SERVICES	DYNAMIC SERVICE REQUEST	Error	Warning	Service Change rejected - Header suppression setting not supported	For SYSLOG only append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S02.7	83000207	docsDevCmDynServReqFailTrap, docsDevCmtsDynServReqFailTrap
DYNAMIC SERVICES	DYNAMIC SERVICE REQUEST	Error	Warning	Service Change rejected - HMAC Auth failure	For SYSLOG only append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S02.8	83000208	docsDevCmDynServReqFailTrap, docsDevCmtsDynServReqFailTrap

Process	Sub-Process	CM Priority	CMTS Priority	Event Messsage	Message Notes And Details	Error Code Set	Event ID	Trap Name
DYNAMIC SERVICES	DYNAMIC SERVICE REQUEST	Error	Warning	Service Change rejected - Multiple errors	For SYSLOG only: append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S02.9	83000209	docsDevCmDynServReqFailTrap, docsDevCmtsDynServReqFailTrap
DYNAMIC SERVICES	DYNAMIC SERVICE REQUEST	Error	Warning	Service Change rejected - Classifier not found	For SYSLOG only: append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S02.10	83000210	docsDevCmDynServReqFailTrap, docsDevCmtsDynServReqFailTrap
DYNAMIC SERVICES	DYNAMIC SERVICE REQUEST	Error		Service Change rejected - Classifier exists	For SYSLOG only: append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S02.11	83000211	docsDevCmDynServReqFailTrap, docsDevCmtsDynServReqFailTrap
DYNAMIC SERVICES	DYNAMIC SERVICE REQUEST	Error	Warning	Service Change rejected - PHS rule not found	For SYSLOG only: append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S02.12	83000212	docsDevCmDynServReqFailTrap, docsDevCmtsDynServReqFailTrap
DYNAMIC SERVICES	DYNAMIC SERVICE REQUEST	Error	Warning	Service Change rejected - PHS rule exists	For SYSLOG only: append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S02.13	83000213	docsDevCmDynServReqFailTrap, docsDevCmtsDynServReqFailTrap
DYNAMIC SERVICES	DYNAMIC SERVICE REQUEST	Error	Warning	Service Change rejected - Duplicated reference-ID or index in message	For SYSLOG only: append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S02.14	83000214	docsDevCmDynServReqFailTrap, docsDevCmtsDynServReqFailTrap

Process	Sub-Process	CM Priority	CMTS Priority	Event Messsage	Message Notes And Details	Error Code Set	Event ID	Trap Name
DYNAMIC SERVICES	DYNAMIC SERVICE REQUEST	Error	Warning	Service Change rejected - Multiple upstream flows	For SYSLOG only: append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S02.15	83000215	docsDevCmDynServReqFailTrap, docsDevCmtsDynServReqFailTrap
DYNAMIC SERVICES	DYNAMIC SERVICE REQUEST	Error	Warning	Service Change rejected - Multiple downstream flows	For SYSLOG only: append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S02.16	83000216	docsDevCmDynServReqFailTrap, docsDevCmtsDynServReqFailTrap
DYNAMIC SERVICES	DYNAMIC SERVICE REQUEST	Error	Warning	Service Change rejected - Classifier for another flow	For SYSLOG only: append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S02.17	83000217	docsDevCmDynServReqFailTrap, docsDevCmtsDynServReqFailTrap
DYNAMIC SERVICES	DYNAMIC SERVICE REQUEST	Error	Warning	Service Change rejected - PHS rule for another flow	For SYSLOG only: append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S02.18	83000218	docsDevCmDynServReqFailTrap, docsDevCmtsDynServReqFailTrap
DYNAMIC SERVICES	DYNAMIC SERVICE REQUEST	Error	Warning	Service Change rejected - Invalid parameter for context	For SYSLOG only: append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S02.19	83000219	docsDevCmDynServReqFailTrap, docsDevCmtsDynServReqFailTrap
DYNAMIC SERVICES	DYNAMIC SERVICE REQUEST	Error	Warning	Service Change rejected - Authorization failure	For SYSLOG only: append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S02.20	83000220	docsDevCmDynServReqFailTrap, docsDevCmtsDynServReqFailTrap

Process	Sub-Process	CM Priority	CMTS Priority	Event Messsage	Message Notes And Details	Error Code Set	Event ID	Trap Name
DYNAMIC SERVICES	DYNAMIC SERVICE REQUEST	Error	Warning	Service Change rejected - Major service flow error	For SYSLOG only: append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S02.21	83000221	docsDevCmDynServReqFailTrap, docsDevCmtsDynServReqFailTrap
DYNAMIC SERVICES	DYNAMIC SERVICE REQUEST	Error	Warning	Service Change rejected -Major classifier error	For SYSLOG only: append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S02.22	83000222	docsDevCmDynServReqFailTrap, docsDevCmtsDynServReqFailTrap
DYNAMIC SERVICES	DYNAMIC SERVICE REQUEST	Error	Warning	Service Change rejected - Major PHS error	For SYSLOG only: append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S02.23	83000223	docsDevCmDynServReqFailTrap, docsDevCmtsDynServReqFailTrap
DYNAMIC SERVICES	DYNAMIC SERVICE REQUEST	Error	Warning	Service Change rejected - Multiple major errors	For SYSLOG only: append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S02.24	83000224	docsDevCmDynServReqFailTrap, docsDevCmtsDynServReqFailTrap
DYNAMIC SERVICES	DYNAMIC SERVICE REQUEST	Error	Warning	Service Change rejected - Message syntax error	For SYSLOG only: append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S02.25	83000225	docsDevCmDynServReqFailTrap, docsDevCmtsDynServReqFailTrap
DYNAMIC SERVICES	DYNAMIC SERVICE REQUEST	Error	Warning	Service Change rejected - Message too big	For SYSLOG only: append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S02.26	83000226	docsDevCmDynServReqFailTrap, docsDevCmtsDynServReqFailTrap

Process	Sub-Process	CM Priority	CMTS Priority	Event Messsage	Message Notes And Details	Error Code Set	Event ID	Trap Name
DYNAMIC SERVICES	DYNAMIC SERVICE REQUEST	Error	Warning	Service Change rejected - Temporary DCC	For SYSLOG only: append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S02.27	83000227	docsDevCmDynServReqFailTrap, docsDevCmtsDynServReqFailTrap
DYNAMIC SERVICES	DYNAMIC SERVICE REQUEST	Error	Warning	Service Delete rejected - Unspecified reason	For SYSLOG only append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S03.0	83000300	docsDevCmDynServReqFailTrap, docsDevCmtsDynServReqFailTrap
DYNAMIC SERVICES	DYNAMIC SERVICE REQUEST	Error	Warning	Service Delete rejected -Requestor not owner of service flow	For SYSLOG only: append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S03.1	83000301	docsDevCmDynServReqFailTrap, docsDevCmtsDynServReqFailTrap
DYNAMIC SERVICES	DYNAMIC SERVICE REQUEST	Error	Warning	Service Delete rejected - Service flow not found	For SYSLOG only: append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S03.2	83000302	docsDevCmDynServReqFailTrap, docsDevCmtsDynServReqFailTrap
DYNAMIC SERVICES	DYNAMIC SERVICE REQUEST	Error	Warning	Service Delete rejected - HMAC Auth failure	For SYSLOG only append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S03.3	83000303	docsDevCmDynServReqFailTrap, docsDevCmtsDynServReqFailTrap
DYNAMIC SERVICES	DYNAMIC SERVICE REQUEST	Error	Warning	Service Delete rejected - Message syntax error	For SYSLOG only: append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S03.4	83000304	docsDevCmDynServReqFailTrap, docsDevCmtsDynServReqFailTrap
DYNAMIC SERVICE RESPONSES								

Process	Sub-Process	CM Priority	CMTS Priority	Event Messsage	Message Notes And Details	Error Code Set	Event ID	Trap Name
DYNAMIC SERVICES	DYNAMIC SERVICE RESPONSE	Error	Warning	Service Add Response rejected -Invalid transaction ID	For SYSLOG only append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S101.0	83010100	docsDevCmDynServRspFailTrap, docsDevCmtsDynServRspFailTrap
DYNAMIC SERVICES	DYNAMIC SERVICE RESPONSE	Error	Warning	Service Add aborted - No RSP	For SYSLOG only append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S101.1	83010101	docsDevCmDynServRspFailTrap, docsDevCmtsDynServRspFailTrap
DYNAMIC SERVICES	DYNAMIC SERVICE RESPONSE	Error	Warning	Service Add Response rejected - HMAC Auth failure	For SYSLOG only append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S101.2	83010102	docsDevCmDynServRspFailTrap, docsDevCmtsDynServRspFailTrap
DYNAMIC SERVICES	DYNAMIC SERVICE RESPONSE	Error	Warning	Service Add Response rejected - Message syntax error	For SYSLOG only append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S101.3	83010103	docsDevCmDynServRspFailTrap, docsDevCmtsDynServRspFailTrap
DYNAMIC SERVICES	DYNAMIC SERVICE RESPONSE	Error	Warning	Service Add Response rejected -Unspecified reason - MAC-addr: <P1	For SYSLOG only append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S101.4	83010104	docsDevCmDynServRspFailTrap, docsDevCmtsDynServRspFailTrap
DYNAMIC SERVICES	DYNAMIC SERVICE RESPONSE	Error	Warning	Service Add Response rejected -Unrecognized configuration setting	For SYSLOG only append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S101.5	83010105	docsDevCmDynServRspFailTrap, docsDevCmtsDynServRspFailTrap

Process	Sub-Process	CM Priority	CMTS Priority	Event Messsage	Message Notes And Details	Error Code Set	Event ID	Trap Name
DYNAMIC SERVICES	DYNAMIC SERVICE RESPONSE	Error	Warning	Service Add Response rejected -Required parameter not present	For SYSLOG only append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S101.6	83010106	docsDevCmDynServRspFailTrap, docsDevCmtsDynServRspFailTrap
DYNAMIC SERVICES	DYNAMIC SERVICE RESPONSE	Error		Service Add Response rejected - Service Flow exists	For SYSLOG only append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S101.7	83010107	docsDevCmDynServRspFailTrap, docsDevCmtsDynServRspFailTrap
DYNAMIC SERVICES	DYNAMIC SERVICE RESPONSE	Error	Warning	Service Add Response rejected - Multiple errors	For SYSLOG only append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S101.8	83010108	docsDevCmDynServRspFailTrap, docsDevCmtsDynServRspFailTrap
DYNAMIC SERVICES	DYNAMIC SERVICE RESPONSE	Error		Service Add Response rejected - Classifier exists	For SYSLOG only append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S101.9	83010109	docsDevCmDynServRspFailTrap, docsDevCmtsDynServRspFailTrap
DYNAMIC SERVICES	DYNAMIC SERVICE RESPONSE	Error	Warning	Service Add Response rejected - PHS rule exists	For SYSLOG only append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S101.10	83010110	docsDevCmDynServRspFailTrap, docsDevCmtsDynServRspFailTrap
DYNAMIC SERVICES	DYNAMIC SERVICE RESPONSE	Error	Warning	Service Add Response rejected -Duplicate reference_ID or index inmessage	For SYSLOG only append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S101.11	83010111	docsDevCmDynServRspFailTrap, docsDevCmtsDynServRspFailTrap

Process	Sub-Process	CM Priority	CMTS Priority	Event Messsage	Message Notes And Details	Error Code Set	Event ID	Trap Name
DYNAMIC SERVICES	DYNAMIC SERVICE RESPONSE	Error	Warning	Service Add Response rejected -Classifier for another flow - MACaddr: <P1>	For SYSLOG only append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S101.12	83010112	docsDevCmDynServRspFailTrap, docsDevCmtsDynServRspFailTrap
DYNAMIC SERVICES	DYNAMIC SERVICE RESPONSE	Error	Warning	Service Add Response rejected -Parameter invalid for context	For SYSLOG only append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S101.13	83010113	docsDevCmDynServRspFailTrap, docsDevCmtsDynServRspFailTrap
DYNAMIC SERVICES	DYNAMIC SERVICE RESPONSE	Error	Warning	Service Add Response rejected -Major service flow error	For SYSLOG only append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S101.14	83010114	docsDevCmDynServRspFailTrap, docsDevCmtsDynServRspFailTrap
DYNAMIC SERVICES	DYNAMIC SERVICE RESPONSE	Error	Warning	Service Add Response rejected -Major classifier error	For SYSLOG only append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S101.15	83010115	docsDevCmDynServRspFailTrap, docsDevCmtsDynServRspFailTrap
DYNAMIC SERVICES	DYNAMIC SERVICE RESPONSE	Error	Warning	Service Add Response rejected -Major PHS Rule error	For SYSLOG only append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S101.16	83010116	docsDevCmDynServRspFailTrap, docsDevCmtsDynServRspFailTrap
DYNAMIC SERVICES	DYNAMIC SERVICE RESPONSE	Error	Warning	Service Add Response rejected -Multiple major errors	For SYSLOG only append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S101.17	83010117	docsDevCmDynServRspFailTrap, docsDevCmtsDynServRspFailTrap

Process	Sub-Process	CM Priority	CMTS Priority	Event Messsage	Message Notes And Details	Error Code Set	Event ID	Trap Name
DYNAMIC SERVICES	DYNAMIC SERVICE RESPONSE	Error	Warning	Service Add Response rejected -Message too big	For SYSLOG only append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S101.18	83010118	docsDevCmDynServRspFailTrap, docsDevCmtsDynServRspFailTrap
DYNAMIC SERVICES	DYNAMIC SERVICE RESPONSE	Error	Warning	Service Change Response rejected - Invalid transaction ID.	For SYSLOG only append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S102.0	83010200	docsDevCmDynServRspFailTrap, docsDevCmtsDynServRspFailTrap
DYNAMIC SERVICES	DYNAMIC SERVICE RESPONSE	Error	Warning	Service Change aborted- No RSP	For SYSLOG only append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S102.1	83010201	docsDevCmDynServRspFailTrap, docsDevCmtsDynServRspFailTrap
DYNAMIC SERVICES	DYNAMIC SERVICE RESPONSE	Error	Warning	Service Change Response rejected - HMAC Auth failure	For SYSLOG only append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S102.2	83010202	docsDevCmDynServRspFailTrap, docsDevCmtsDynServRspFailTrap
DYNAMIC SERVICES	DYNAMIC SERVICE RESPONSE	Error	Warning	Service Change Response rejected - Unspecified reason	For SYSLOG only append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S102.4	83010204	docsDevCmDynServRspFailTrap, docsDevCmtsDynServRspFailTrap
DYNAMIC SERVICES	DYNAMIC SERVICE RESPONSE	Error	Warning	Service Change Response rejected - Unrecognized configuration setting	For SYSLOG only append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S102.5	83010205	docsDevCmDynServRspFailTrap, docsDevCmtsDynServRspFailTrap

Process	Sub-Process	CM Priority	CMTS Priority	Event Messsage	Message Notes And Details	Error Code Set	Event ID	Trap Name
DYNAMIC SERVICES	DYNAMIC SERVICE RESPONSE	Error	Warning	Service Change Response rejected - Required parameter not present	For SYSLOG only append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S102.6	83010206	docsDevCmDynServRspFailTrap, docsDevCmtsDynServRspFailTrap
DYNAMIC SERVICES	DYNAMIC SERVICE RESPONSE	Error	Warning	Service Change Response rejected - Multiple errors	For SYSLOG only append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S102.7	83010207	docsDevCmDynServRspFailTrap, docsDevCmtsDynServRspFailTrap
DYNAMIC SERVICES	DYNAMIC SERVICE RESPONSE	Error		Service Change Response rejected - Classifier exists	For SYSLOG only append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S102.8	83010208	docsDevCmDynServRspFailTrap, docsDevCmtsDynServRspFailTrap
DYNAMIC SERVICES	DYNAMIC SERVICE RESPONSE	Error	Warning	Service Change Response rejected - PHS rule exists	For SYSLOG only append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S102.9	83010209	docsDevCmDynServRspFailTrap, docsDevCmtsDynServRspFailTrap
DYNAMIC SERVICES	DYNAMIC SERVICE RESPONSE	Error	Warning	Service Change Response rejected - Duplicated reference-ID or index in	For SYSLOG only append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S102.10	83010210	docsDevCmDynServRspFailTrap, docsDevCmtsDynServRspFailTrap
DYNAMIC SERVICES	DYNAMIC SERVICE RESPONSE	Error	Warning	Service Change Response rejected - Invalid parameter for context	For SYSLOG only append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S102.11	83010211	docsDevCmDynServRspFailTrap, docsDevCmtsDynServRspFailTrap

Process	Sub-Process	CM Priority	CMTS Priority	Event Messsage	Message Notes And Details	Error Code Set	Event ID	Trap Name
DYNAMIC SERVICES	DYNAMIC SERVICE RESPONSE	Error	Warning	Service Change Response rejected - Major classifier error	For SYSLOG only append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S102.12	83010212	docsDevCmDynServRspFailTrap, docsDevCmtsDynServRspFailTrap
DYNAMIC SERVICES	DYNAMIC SERVICE RESPONSE	Error	Warning	Service Change Response rejected - Major PHS rule error	For SYSLOG only append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S102.13	83010213	docsDevCmDynServRspFailTrap, docsDevCmtsDynServRspFailTrap
DYNAMIC SERVICES	DYNAMIC SERVICE RESPONSE	Error	Warning	Service Change Response rejected - Multiple Major errors	For SYSLOG only append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S102.14	83010214	docsDevCmDynServRspFailTrap, docsDevCmtsDynServRspFailTrap
DYNAMIC SERVICES	DYNAMIC SERVICE RESPONSE	Error	Warning	Service Change Response rejected - Message too big	For SYSLOG only append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S102.15	83010215	docsDevCmDynServRspFailTrap, docsDevCmtsDynServRspFailTrap
DYNAMIC SERVICES	DYNAMIC SERVICE RESPONSE	Error	Warning	Service Change Response rejected - Message syntax error	For SYSLOG only append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S102.3	83010203	docsDevCmDynServRspFailTrap, docsDevCmtsDynServRspFailTrap
DYNAMIC SERVICES	DYNAMIC SERVICE RESPONSE	Error	Warning	Service Delete Response rejected - Invalid transaction ID	For SYSLOG only append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S103.0	83010300	docsDevCmDynServRspFailTrap, docsDevCmtsDynServRspFailTrap
DYNAMIC SERVICE ACKNOWLEDGEMENTS								

Process	Sub-Process	CM Priority	CMTS Priority	Event Messsage	Message Notes And Details	Error Code Set	Event ID	Trap Name
DYNAMIC SERVICES	DYNAMIC SERVICE ACKNOWLEDGEMENT	Error	Warning	Service Add Response rejected - Invalid Transaction ID	For SYSLOG only append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S201.0	83020100	docsDevCmDynServAckFailTrap, docsDevCmtsDynServAckFailTrap
DYNAMIC SERVICES	DYNAMIC SERVICE ACKNOWLEDGEMENT	Error	Warning	Service Add Aborted - No ACK	For SYSLOG only append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S201.1	83020101	docsDevCmDynServAckFailTrap, docsDevCmtsDynServAckFailTrap
DYNAMIC SERVICES	DYNAMIC SERVICE ACKNOWLEDGEMENT	Error	Warning	Service Add ACK rejected - HMAC auth failure	For SYSLOG only append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S201.2	83020102	docsDevCmDynServAckFailTrap, docsDevCmtsDynServAckFailTrap
DYNAMIC SERVICES	DYNAMIC SERVICE ACKNOWLEDGEMENT	Error	Warning	Service Add ACK rejected- Message syntax error	For SYSLOG only append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S201.3	83020103	docsDevCmDynServAckFailTrap, docsDevCmtsDynServAckFailTrap
DYNAMIC SERVICES	DYNAMIC SERVICE ACKNOWLEDGEMENT	Error	Warning	Service Change ACK rejected - Invalid transaction ID	For SYSLOG only append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S202.0	83020200	docsDevCmDynServAckFailTrap, docsDevCmtsDynServAckFailTrap
DYNAMIC SERVICES	DYNAMIC SERVICE ACKNOWLEDGEMENT	Error	Warning	Service Change Aborted - No ACK	For SYSLOG only append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S202.1	83020201	docsDevCmDynServAckFailTrap, docsDevCmtsDynServAckFailTrap

Process	Sub-Process	CM Priority	CMTS Priority	Event Messsage	Message Notes And Details	Error Code Set	Event ID	Trap Name
DYNAMIC SERVICES	DYNAMIC SERVICE ACKNOWLEDGEMENT	Error	Warning	Service Change ACK rejected - HMAC Auth failure	For SYSLOG only append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S202.2	83020202	docsDevCmDynServAckFailTrap, docsDevCmtsDynServAckFailTrap
DYNAMIC SERVICES	DYNAMIC SERVICE ACKNOWLEDGEMENT	Error	Warning	Service Change ACK rejected - Message syntax error	For SYSLOG only append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	S202.3	83020203	docsDevCmDynServAckFailTrap, docsDevCmtsDynServAckFailTrap
CM CONFIGURATION FILE (BPI+)								
Init (BPI+)		Error	Notice	Missing BP Configuration Setting TLV Type: <P1>	P1 = missing required TLV Type	B101.0	66010100	DocsDevCmBpiInitTrap, docsDevCmtsBpiInitTrap
Init (BPI+)		Alert	Notice	Invalid BP Configuration Setting Value: <P1> for Type: <P2>	P1=The TLV Value for P2.P2 = The first Configuration TLV Type that contain invalid value.	B102.0	66010200	docsDevCmBpiInitTrap
AUTH FSM								
BPKM		Warning	Error	Auth Reject - No Information	For SYSLOG only, append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	B301.2	66030102	docsDevCmBPKMTrap, docsDevCmtsBPKMTrap
BPKM		Warning	Error	Auth Reject - Unauthorized CM	For SYSLOG only, append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	B301.3	66030103	docsDevCmBPKMTrap, docsDevCmtsBPKMTrap

Process	Sub-Process	CM Priority	CMTS Priority	Event Messsage	Message Notes And Details	Error Code Set	Event ID	Trap Name
BPKM		Warn-ing	Error	Auth Reject - Unauthorized SAID	For SYS-LOG only, append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	B301.4	66030104	docsDevCmBPKMTrap, docsDevCmtsBPKMTrap
BPKM		Warn-ing	Error	Auth Reject - Permanent Au-thorization Fail-ure	For SYS-LOG only, append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	B301.8	66030108	docsDevCmBPKMTrap, docsDevCmtsBPKMTrap
BPKM		Warn-ing	Error	Auth Reject - Time of Day not acquired	For SYS-LOG only, append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	B301.9	66030109	docsDevCmBPKMTrap, docsDevCmtsBPKMTrap
BPKM		Alert	Error	CM Certificate Error	For SYS-LOG only, append: MAC addr: <P1> P1=Mac Ad-dr of CMTS (for CM) or CM (for CMTS)	B301.11	66030111	DocsDevCmBPKMTrap, docsDevCmtsBPKMTrap
BPKM		Warn-ing	Error	Auth Invalid - No Information	For SYS-LOG only, append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	B302.2	66030202	docsDevCmBPKMTrap, docsDevCmtsBPKMTrap
BPKM		Warn-ing	Error	Auth Invalid - Unauthorized CM	For SYS-LOG only, append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	B302.3	66030203	docsDevCmBPKMTrap, docsDevCmtsBPKMTrap

Process	Sub-Process	CM Priority	CMTS Priority	Event Messsage	Message Notes And Details	Error Code Set	Event ID	Trap Name
BPKM		Warn-ing	Error	Auth Invalid - Unsolicited	For SYS-LOG only, append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	B302.5	66030205	docsDevCmBPKMTrap, docsDevCmtsBPKMTrap
BPKM		Warn-ing	Error	Auth Invalid - Invalid Key Se-quence Number	For SYS-LOG only, append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	B302.6	66030206	docsDevCmBPKMTrap, docsDevCmtsBPKMTrap
BPKM		Warn-ing	Error	Auth Invalid - Message (Key Request) Au-thentication Fail-ure	For SYS-LOG only, append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	B302.7	66030207	docsDevCmBPKMTrap, docsDevCmtsBPKMTrap
BPKM		Warn-ing	Error	Unsupported Crypto Suite	For SYS-LOG only, append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	B303.0	66030300	docsDevCmBPKMTrap, docsDevCmtsBPKMTrap
EVENT BETWEEN AUTH & TEK FSM								
BPKM		Infor-mation-al		Authorized	For CM SYSLOG only ap-pend: MAC addr: <P1> P1 = Mac Addr of CM	B401.0	66040100	docsDevCmBPKMTrap
BPKM		Infor-mation-al		Auto Pend	For CM SYSLOG only ap-pend: MAC addr: <P1> P1 = Mac Addr of CM	B402.0	66040200	docsDevCmBPKMTrap

Process	Sub-Process	CM Priority	CMTS Priority	Event Messsage	Message Notes And Details	Error Code Set	Event ID	Trap Name
BPKM		Infor-mation-al		Auth Comp	For CM SYSLOG only ap-pend: MAC addr: <P1> P1 = Mac Addr of CM	B403.0	66040300	docsDevCmBPKMTrap
BPKM		Infor-mation-al		Stop	For CM SYSLOG only ap-pend: MAC addr: <P1> P1 = Mac Addr of CM	B404.0	66040400	docsDevCmBPKMTrap
TEK FSM								
BPKM		Warn-ing	Error	Key Reject - No Information	For SYS-LOG only, append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	B501.2	66050102	docsDevCmBPKMTrap, docsDevCmtsBPKMTrap
BPKM		Warn-ing	Error	Key Reject - Un-authorized SAID	For SYS-LOG only, append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	B501.3	66050103	docsDevCmBPKMTrap, docsDevCmtsBPKMTrap
BPKM		Warn-ing	Error	TEK Invalid - No Information	For SYS-LOG only, append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	B502.3	66050203	docsDevCmBPKMTrap, docsDevCmtsBPKMTrap
BPKM		Warn-ing	Error	TEK Invalid - In-valid Key Se-quence Number	For SYS-LOG only, append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	B502.6	66050206	docsDevCmBPKMTrap, docsDevCmtsBPKMTrap
SA MAP FSM								

Process	Sub-Process	CM Priority	CMTS Priority	Event Messsage	Message Notes And Details	Error Code Set	Event ID	Trap Name
Dynamic SA		Infor-mation-al		SA Map State Machine Started	For CM SYSLOG only ap-pend: MAC addr: <P1> P1 = Mac Addr of CM	B601.0	66060100	docsDevCmDynamicSATrap
Dynamic SA		Warn-ing	Error	Unsupported Crypto Suite	For SYS-LOG only, append: MAC addr: <P1>. P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	B602.0	66060200	docsDevCmDynamicSATrap, docsDevCmtsDynamicSATrap
Dynamic SA		Error		Map Request Retry Timeout	For CM SYSLOG only append: MAC addr: <P1>. P1 = Mac Addr of CMTS	B603.0	66060300	docsDevCmDynamicSATrap
Dynamic SA		Infor-mation-al		Unmap	For CM SYSLOG only append: MAC addr: <P1>. P1 = Mac Addr of CMTS	B604.0	66060400	docsDevCmDynamicSATrap
Dynamic SA		Warn-ing	Error	Map Reject - Not Authorized for Requested Downstream Traffic Flow (EC=7)	For SYS-LOG only, append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	B605.9	66060509	docsDevCmDynamicSATrap, docsDevCmtsDynamicSATrap
Dynamic SA		Warn-ing	Error	Map Reject - Downstream Traffic Flow Not Mapped to BPI+ SAID (EC=8)	For SYS-LOG only, append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	B605.10	66060510	docsDevCmDynamicSATrap, docsDevCmtsDynamicSATrap
Dynamic SA		Warn-ing	Error	Mapped to Exist-ing SAID	For SYS-LOG only, append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	B606.0	66060600	docsDevCmDynamicSATrap, docsDevCmtsDynamicSATrap

Process	Sub-Process	CM Priority	CMTS Priority	Event Messsage	Message Notes And Details	Error Code Set	Event ID	Trap Name
Dynamic SA		Warn-ing	Error	Mapped to New SAID	For SYS-LOG only, append: MAC addr: <P1> P1 = Mac Addr of CMTS (for CM) or CM (for CMTS)	B607.0	66060700	docsDevCmDynamicSATrap, docsDevCmtsDynamicSATrap
VERIFICAITON OF CODE FILE								
SW Up-grade	SW UP-GRADE GENER-AL FAIL-URE	Error		Improper Code File Controls	For SYS-LOG only, append: Code File: <P1> - Code File Server: <P2>. P1= Code file name, P2 = code file server IP ad-dress	E201.0	69020100	docsDevCmSwUpgradeFailTrap
SW Up-grade	SW UP-GRADE GENER-AL FAIL-URE	Error		Code File Manu-facturer CVC Validation Fail-ure	For SYS-LOG only, append: Code File: <P1> - Code File Server: <P2>. P1= Code file name, P2 = code file server IP ad-dress	E202.0	69020200	docsDevCmSwUpgradeFailTrap
SW Up-grade	SW UP-GRADE GENER-AL FAIL-URE	Error		Code File Manu-facturer CVS Validation Fail-ure	For SYS-LOG only, append: Code File: <P1> - Code File Server: <P2>. P1= Code file name, P2 = code file server IP ad-dress	E203.0	69020300	docsDevCmSwUpgradeFailTrap
SW Up-grade	SW UP-GRADE GENER-AL FAIL-URE	Error		Code File Co-Signer CVC Vali-dation Failure	For SYS-LOG only, append: Code File: <P1> - Code File Server: <P2>. P1= Code file name, P2 = code file server IP ad-dress	E204.0	69020400	docsDevCmSwUpgradeFailTrap

Process	Sub-Process	CM Priority	CMTS Priority	Event Messsage	Message Notes And Details	Error Code Set	Event ID	Trap Name
SW Upgrade	SW UPGRADE GENERAL FAILURE	Error		Code File Co-Signer CVS Validation Failure	For SYSLOG only, append: Code File: <P1> - Code File Server: <P2>. P1= Code file name, P2 = code file server IP address	E205.0	69020500	docsDevCmSwUpgradeFailTrap
VERIFICATION OF CVC								
SW Upgrade	VERIFICATION 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	docsDevCmSwUpgradeCVCFailTrap
SW Upgrade	VERIFICATION OF CVC	Error		Configuration File CVC Validation Failure - TFTP Server: <P1> - Config File: <P2>	P1 = TFTP Server IP AddressP2 = Config File Name	E207.0	69020700	docsDevCmSwUpgradeCVCFailTrap
SW Upgrade	VERIFICATION OF CVC	Error		Improper SNMP CVC Format - Snmp manager: <P1>	P1= IP Address of SNMP Manager	E208.0	69020800	docsDevCmSwUpgradeCVCFailTrap
SW Upgrade	VERIFICATION OF CVC*	Error		SNMP CVC Validation Failure - Snmp manager: <P1>	P1=IP Addr of SNMP manager	E209.0	69020900	docsDevCmSwUpgradeCVCFailTrap
UCC-REQ Upstream Channel Change Request								
UCC	UCC Request	Error	Warning	UCC-REQ received with invalid or out of range US channel ID.		C01.0	67000100	
UCC	UCC Request	Error	Warning	UCC-REQ received unable to send UCC-RSP, no TX opportunity.		C02.0	67000200	
UCC-RSP Upstream Channel Change Response								
UCC	UCC Response		Warning	UCC-RSP not received on previous channel ID.		C101.0	67010100	
UCC	UCC Response		Warning	UCC-RSP received with invalid channel ID.		C102.0	67010200	

Process	Sub-Process	CM Priority	CMTS Priority	Event Messsage	Message Notes And Details	Error Code Set	Event ID	Trap Name
UCC	UCC Response		Warning	UCC-RSP received with invalid channel ID on new channel.		C103.0	67010300	
Dynamic Channel Change Request								
DCC	DCC Request	Error	Warning	DCC rejected already there		C201.0	67020100	DocsDevCmDccReqFailTrap, docsDevCmtsDccReqFailTrap
DCC	DCC Request	Informational	Notice	DCC depart old		C202.0	67020200	DocsDevCmDccReqFailTrap, docsDevCmtsDccReqFailTrap
DCC	DCC Request	Informational	Notice	DCC arrive new		C203.0	67020300	DocsDevCmDccReqFailTrap, docsDevCmtsDccReqFailTrap
DCC	DCC Request	Error	Warning	DCC aborted unable to acquire new downstream channel		C204.0	67020400	DocsDevCmDccReqFailTrap, docsDevCmtsDccReqFailTrap
DCC	DCC Request	Error	Warning	DCC aborted no UCD for new upstream channel		C205.0	67020500	DocsDevCmDccReqFailTrap, docsDevCmtsDccReqFailTrap
DCC	DCC Request	Error	Warning	DCC aborted unable to communicate on new upstream channel		C206.0	67020600	DocsDevCmDccReqFailTrap, docsDevCmtsDccReqFailTrap
DCC	DCC Request	Error	Warning	DCC rejected unspecified reason		C207.0	67020700	DocsDevCmDccReqFailTrap, docsDevCmtsDccReqFailTrap
DCC	DCC Request	Error	Warning	DCC rejected permanent - DCC not supported		C208.0	67020800	DocsDevCmDccReqFailTrap, docsDevCmtsDccReqFailTrap

Process	Sub-Process	CM Priority	CMTS Priority	Event Messsage	Message Notes And Details	Error Code Set	Event ID	Trap Name
DCC	DCC Request	Error	Warning	DCC rejected service flow not found		C209.0	67020900	DocsDevCmDccReqFailTrap, docsDevCmtsDccReqFailTrap
DCC	DCC Request	Error	Warning	DCC rejected required parameter not present		C210.0	67021000	DocsDevCmDccReqFailTrap, docsDevCmtsDccReqFailTrap
DCC	DCC Request	Error	Warning	DCC rejected authentication failure		C211.0	67021100	DocsDevCmDccReqFailTrap, docsDevCmtsDccReqFailTrap
DCC	DCC Request	Error	Warning	DCC rejected multiple errors		C212.0	67021200	DocsDevCmDccReqFailTrap, docsDevCmtsDccReqFailTrap
DCC	DCC Request	Error	Warning	DCC rejected duplicate DCC Request reference-ID or index in message		C215.0	67021500	DocsDevCmDccReqFailTrap, docsDevCmtsDccReqFailTrap
DCC	DCC Request	Error	Warning	DCC rejected parameter invalid for context		C216.0	67021600	DocsDevCmDccReqFailTrap, docsDevCmtsDccReqFailTrap
DCC	DCC Request	Error	Warning	DCC rejected message syntax error		C217.0	67021700	DocsDevCmDccReqFailTrap, docsDevCmtsDccReqFailTrap
DCC	DCC Request	Error	Warning	DCC rejected message too big		C218.0	67021800	DocsDevCmDccReqFailTrap, docsDevCmtsDccReqFailTrap
Dynamic Channel Change Response								
DCC	DCC Response		Warning	DCC-RSP not received on old channel		C301.0	67030100	DocsDevCmDccRspFailTrap, docsDevCmtsDccRspFailTrap

Process	Sub-Process	CM Priority	CMTS Priority	Event Messsage	Message Notes And Details	Error Code Set	Event ID	Trap Name
DCC	DCC Response		Warning	DCC-RSP not received on new channel		C302.0	67030200	DocsDevCmDccRspFailTrap, docsDevCmtsDccRspFailTrap
DCC	DCC Response		Warning	DCC-RSP rejected unspecified reason		C303.0	67030300	DocsDevCmDccRspFailTrap, docsDevCmtsDccRspFailTrap
DCC	DCC Response		Warning	DCC-RSP rejected unknown transaction ID		C304.0	67030400	DocsDevCmDccRspFailTrap, docsDevCmtsDccRspFailTrap
DCC	DCC Response		Warning	DCC-RSP rejected authentication failure		C305.0	67030500	DocsDevCmDccRspFailTrap, docsDevCmtsDccRspFailTrap
DCC	DCC Response		Warning	DCC-RSP rejected message syntax error		C306.0	67030600	DocsDevCmDccRspFailTrap, docsDevCmtsDccRspFailTrap
Dynamic Channel Change Acknowledgement								
DCC	DCC Acknowledgement	Error	Warning	DCC-ACK not received		C401.0	67040100	DocsDevCmDccAckFailTrap, docsDevCmtsDccAckFailTrap
DCC	DCC Acknowledgement	Error	Warning	DCC-ACK rejected unspecified reason		C402.0	67040200	DocsDevCmDccAckFailTrap, docsDevCmtsDccAckFailTrap
DCC	DCC Acknowledgement	Error	Warning	DCC-ACK rejected unknown transaction ID		C403.0	67040300	DocsDevCmDccAckFailTrap, docsDevCmtsDccAckFailTrap

Process	Sub-Process	CM Priority	CMTS Priority	Event Messsage	Message Notes And Details	Error Code Set	Event ID	Trap Name
DCC	DCC Ac-knowl-edgement	Error	Warn-ing	DCC-ACK re-jected authenti-cation failure		C404.0	67040400	DocsDevCmDccAckFailTrap, docsDevCmtsDccAckFailTrap
DCC	DCC Ac-knowl-edgement	Error	Warn-ing	DCC-ACK re-jected message syntax error		C405.0	67040500	DocsDevCmDccAckFailTrap, docsDevCmtsDccAckFailTrap

This page intentionally left blank.

Annex E Application of RFC 2933 to DOCSIS 2.0 Active/Passive IGMP Devices (normative)

E.1 DOCSIS 2.0 IGMP MIBs

DOCSIS 2.0 devices, CM and CMTS, that support IGMP (in active or passive mode), MUST support the IDMR IGMP MIB (RFC 2933). As such, this section describes the application of the IETF IDMR sub-committee IGMP MIB to DOCSIS 2.0 active/passive IGMP devices.

The IDMR IGMP MIB is organized into two distinct tables, the interface and cache tables. The IGMP Interface Table contains entries for each interface that supports IGMP on a device. For DOCSIS 2.0 this includes the NSI and HFC for the CMTS and the HFC and CMCI on the CM. The IGMP Cache Table contains one row for each IP Multicast Group for which there are active members on a given interface. Active membership MUST only exist on the CMCI of a Cable Modem. However, active membership MAY exist on both the NSI and HFC side interfaces of the CMTS. This is because a CMTS may be implemented as a Multicast Router on which other network side devices are actively participating in a multicast session.

Support of the IDMR IGMP MIB by DOCSIS 2.0 devices is presented in terms of IGMP capabilities, the device type (CM or CMTS), and the interface on which IGMP is supported. This is followed by a set of new IGMP MIB conformance, compliance and group statements for DOCSIS 2.0 devices.

E.1.1 IGMP Capabilities: Active and Passive Mode

There are two basic modes of IGMP capability that are applicable to a DOCSIS 2.0 device. The first mode is a passive operation in which the device selectively forwards IGMP based upon the known state of multicast session activity on the subscriber side (an example of this is described in Appendix VI of [DOCSIS 5]). In passive mode, the device derives its IGMP timers based on the rules specified in section 5.3.1 of the RFI specification. The second mode is an active operation in which the device terminates and initiates IGMP based upon the known state of multicast session activity on the subscriber side. One example of the latter, active, mode is commonly referred to as an IGMP-Proxy implementation side (as described in [ID-IGMP]). A more complete example of an active IGMP device is that of a Multicast Router. Although a specific implementation is not imposed by the DOCSIS 2.0 specification, the device MUST meet the requirements stated in section 5.3.1 of [DOCSIS 5] and MUST support the IDMR IGMP MIB as described herein. As presently specified in the DOCSIS 2.0, active CMs are explicitly prohibited from transmitting IGMP Queries upstream onto the HFC. However, active CMTSs may transmit IGMP Queries onto the NSI as mentioned previously.

E.1.2 IGMP Interfaces

A description of the application of the IDMR IGMP MIB to DOCSIS 2.0 devices follows. This description is organized by CM and CMTS device type.

E.2 DOCSIS 2.0 CM Support for the IGMP MIB

There are two types of interfaces applicable to IGMP on the DOCSIS 2.0 CM. These are the HFC-Side and CMCI-Side interfaces, respectively. Application of the IGMP MIB to DOCSIS 2.0 CMs is presented in terms of passive and active CM operation and these two interface types.

E.2.1 igmpInterfaceTable- igmpInterfaceEntry

E.2.1.1 igmpInterfaceIfIndex

The ifIndex value of the interface for which IGMP is enabled.

E.2.1.1.1 All Modes

This is the same for passive and active modes.

HFC-side: not-accessible. ifIndex of docsCableMaclayer(127), CATV MAC Layer

CMCI-side: not-accessible. ifIndex of CMCI-Side interface.

E.2.1.2 igmpInterfaceQueryInterval

The frequency at which IGMP Host-Query packets are transmitted on this interface.

E.2.1.2.1 Passive Mode

HFC-side: n/a, read-only. The CM MUST not transmit queries upstream. Return a value of zero.

CMCI-side: read only . This value is derived based on the interval of queries received from an upstream querier.

E.2.1.2.2 Active Mode

HFC-side: n/a, read-only. The CM MUST not transmit queries upstream. Return a value of zero.

CMCI-side: read-create. Min = 0; Max = (2^32-1); Default = 125

E.2.1.3 igmpInterfaceStatus

The activation of a row enables IGMP on the interface. The destruction of a row disables IGMP on the interface.

E.2.1.3.1 All Modes

MUST be enabled on both interfaces for all DOCSIS 2.0 CM interfaces.

E.2.1.4 igmpInterfaceVersion

The version of IGMP which is running on this interface. MUST be version 2 for all DOCSIS 2.0 CM interfaces.

E.2.1.5 igmpInterfaceQuerier

The address of the IGMP Querier on the IP subnet to which this interface is attached.

E.2.1.5.1 Passive Mode

HFC-side: read-only. MUST be the address of an upstream IGMP Querier device for both active and passive CMs.

CMCI-side: read-only. Same as HFC-side value.

E.2.1.5.2 Active Mode

HFC-side: read-only. MUST be the address of an upstream IGMP Querier device for both active and passive CMs.

CMCI-side: read-only. Active CMs may report it as the HFC-side value. However, active CMs that participate in IGMP Querier negotiation on the CMCI may report it as a different CPE.

E.2.1.6 igmpInterfaceQueryMaxResponseTime

The maximum query response time advertised in IGMPv2 queries on this interface.

E.2.1.6.1 Passive Mode

HFC-side: n/a, read-only. return a value of zero.

CMCI-side: read-only. This value is derived from observation of queries received from an upstream querier

E.2.1.6.2 Active Mode

HFC-side: n/a, read-only. return a value of zero.

CMCI-side: read-create. Min = 0; Max = 255; Default = 100.

E.2.1.7 igmpInterfaceQuerierUpTime

The time since igmpInterfaceQuerier was last changed.

E.2.1.7.1 PassiveMode

HFC-side: read-only.

CMC-side: n/a, read-only. Return a value of zero.

E.2.1.7.2 Active Mode

HFC-side: read-only.

CMCI-side: read-only.

E.2.1.8 igmpInterfaceQuerierExpiryTime

The amount of time remaining before the other querier present timer expires. If the local system is the queier, the value of this object is zero.

E.2.1.8.1 Passive Mode

Both interfaces: n/a, read-only. The CM is never the querier, return 0.

E.2.1.8.2 Active Mode

HFC-side: n/a, read-only. Return 0.

CMCI-side: read-only. The CM may only be the querier on the CMCI.

E.2.1.9 igmpInterfaceVersion1QuerierTimer

The time remaining until the host assumes that there are no IGMPv1 routers present on the interface. While this is non-zero, the host will reply to all queries with version 1 membership reports.

E.2.1.9.1 Passive Mode

HFC-side: n/a read-only. Return a value of zero.

CMCI-side: n/a read-only. Return a value of zero.

E.2.1.9.2 Active Mode

HFC-side: read-only.

CMCI-side: read-only.

E.2.1.10 igmpInterfaceWrongVersionQueries

The number of queries recived whose IGMP version does not match igmpInterfaceVersion, over the lifetime of the row entry. IGMP requires that all routers on a LAN be configured to run the same version of IGMP. Although, DOCSIS 2.0 requires that all CM and CMTS devices support IGMPv2, it is possible for an upstream querier to be an IGMPv1 querier.

E.2.1.10.1 All Modes

All interfaces: read-only. The number of non-v2 queries received on this interface.

E.2.1.11 igmpInterfaceJoins

The number of times a group membership has been added on this interface; that is, the number of times an entry for this interface has been added to the Cache Table. This object gives an indication of the amount of IGMP activity over the lifetime of the row entry.

All HFC-side: n/a, read-only. Always return a value of zero (see CMCI-side).

IAll CMCI-side: read-only. Group membership is defined to only exist on the CMCI.

E.2.1.12 igmpInterfaceProxyIfIndex

Some devices implement a form of IGMP proxying whereby memberships learned on the interface represented by this row, cause IGMP Host Membership Reports to be sent on the interface whose ifIndex value is given by this object. Such a device would implement the igmpV2RouterMIBGroup only on its router interfaces (those interfaces with non-zero igmpInterfaceProxyIfIndex). Typically, the value of this object is 0, indicating that no proxying is being done.

E.2.1.12.1 Passive Mode

All Interfaces: read-only. Always return a value of zero.

E.2.1.12.2 Active Mode

HFC-side: read-only. Always return a value of zero.

CMCI-side: read-only. Always return a ifIndex for HFC-side interface.

E.2.1.13 igmpInterfaceGroups

The current number of entries for this interface in the Cache Table.

E.2.1.13.1 All HFC-side: n/a, read-only. Always return a value of zero (see CMCI-side).

E.2.1.13.2 All CMCI-side: read-only. Group membership is defined to only exist on the CMCI.

Number of active sessions Proxied or Active on this Interface.

E.2.1.14 igmpInterfaceRobustness

The robustness variable allows tuning for the expected packet loss on a subnet. If a subnet is expected to be lossy, the robustness variable may be increased. IGMP is robust to (robustness variable - 1) packet losses.

E.2.1.14.1 Passive Mode

HFC-side: n/a read-only. Return a value of zero.

CMCI-side: n/a read-only. Return a value of zero.

E.2.1.14.2 Active Mode

All interfaces: read-create. Min = 1; Max = (232-1); Default = 2

E.2.1.15 igmpInterfaceLastMemberQueryIntvl

The last member query interval is the max response time inserted into group specific queries sent in response to leave group messages, and is also the amount of time between group specific query messages. This value may be

tuned to modify the leave latency of the network. A reduced value results in reduced time to detect the loss of the last member of a group.

E.2.1.15.1 Passive Mode

HFC-side: n/a, read-only. return a value of zero.

CMCI-side: read-only. This value is derived from observation of queries received from an upstream querier

E.2.1.15.2 Active Mode

HFC-side: n/a, read-only. return a value of zero.

CMCI-side: read-create. Min = 0; Max = 255; Default = 100.

E.2.2 igmpCacheTable - igmpCacheEntry

E.2.2.1 igmpCacheAddress

The IP multicast group address for which this entry contains information.

E.2.2.1.1 All Modes

Not-accessible (index). Report the address of active IP Multicast on the CMCI interface.

E.2.2.2 igmpCacheIfIndex

The interface for which this entry contains information for an IP multicast group address.

E.2.2.2.1 All Modes

MUST only apply to CMCI interface (e.g., membership is only active on subscriber side of CM).

E.2.2.3 igmpCacheSelf

An indication of whether the local system is a member of this group address on this interface.

E.2.2.3.1 Passive Mode

Read-only. MUST be set to FALSE. The CM is not a member of any group.

E.2.2.3.2 Active Mode

Read-create. Implementation specific. If the CM is configured to be a member of the group, then membership reports are sent with the CM's IP Address but MUST ONLY be sent in proxy for active sessions on the CMCI (e.g., the CM MUST NOT be a member of a multicast group that is not active on the CMCI). If the CM is not configured to be a member, then the source IP Address of membership reports MUST be set to the current value of the igmpCacheLastReporter address.

E.2.2.4 igmpCacheLastReporter

The IP address of the source of the last membership report received for this IP Multicast group address on this interface. If no membership report has been received, this object has the value of 0.0.0.0.

E.2.2.4.1 All Modes

MUST only apply to last reporter on CMCI interface (e.g., membership is only active on subscriber side of CM).

E.2.2.5 igmpCacheUpTime

The time elapsed since this entry was created.

E.2.2.5.1 All Modes

read-only. MUST only apply to duration of membership on CMCI interface (e.g., membership is only active on subscriber side of CM).

E.2.2.6 igmpCacheExpiryTime

The minimum amount of time remaining before this entry will be aged out.

E.2.2.6.1 All Modes

read-only. MUST only apply to duration of membership on CMCI interface (e.g., membership is only active on subscriber side of CM).

E.2.2.7 igmpCacheStatus

The status of this entry.

E.2.2.7.1 All Modes

read-create. MUST only apply to membership on CMCI interface (e.g., membership is only active on subscriber side of CM). Deletion of a row results in preventing downstream forwarding to this IP Multicast group address on this interface.

E.2.2.8 igmpCacheVersion1HostTimer

The time remaining until the local querier will assume that there are no longer any IGMP version 1 members on thie IP subnet attached to this interface. Upon hearing any IGMPv1 membership report, this value is reset to the group membership timer. While this time remaining is non-zero, the local querier ignores any IGMPv2 leave messages for this group that it receives on this interface.

E.2.2.8.1 Passive Mode

All interfaces: n/a, read-only. Return a value of zero.

E.2.2.8.2 Active Mode

HFC-side: n/a, read-only. Return a value of zero.

CMCI-side: read-only.

E.3 Docsis 2.0 CMTS support for the IGMP MIB

There are two types of interfaces applicable to IGMP on the DOCSIS 2.0 CMTS. These are the NSI-Side and NSI-Side interfaces, respectively. Application of the IGMP MIB to DOCSIS 2.0 CMTSes is presented in terms of passive and active CMTS operation and these two interface types.

It is important to note that an active IGMP capable CMTS may be implemented as a proxy, router, or hybrid device. As such, the CMTS may be capable of querying on both its NSI and HFC side interfaces and may manage membership for devices on its NSI interfaces (e.g., as a multicast router). This is different than an active CM, which MUST NOT query on its HFC side interface (e.g., it may only query on its CMCI). This capability is accounted for in the application of the IGMP MIB to the CMTS.

E.3.1 igmpInterfaceTable- igmpInterfaceEntry

E.3.1.1 igmpInterfaceIfIndex

The ifIndex value of the interface for which IGMP is enabled.

E.3.1.1.1 All Modes

This is the same for passive and active modes.

NSI-side: not-accessible. ifIndex of applicable network side interface(s).

HFC-side: not-accessible. ifIndex of docsCableMaclayer(127), CATV MAC Layer interface.

E.3.1.2 igmpInterfaceQueryInterval

The frequency at which IGMP Host-Query packets are transmitted on this interface.

E.3.1.2.1 Passive Mode

NSI-side: n/a, read-only. Return a value of zero.

HFC-side: read only . This value is derived based on the interval of queries received from a Network Side querier.

E.3.1.2.2 Active Mode

NSI-side: read-create. Min = 0; Max = (2^32-1); Default = 125

HFC-side: read-create. Min = 0; Max = (2^32-1); Default = 125

E.3.1.3 igmpInterfaceStatus

E.3.1.3.1 All Modes

The activation of a row enables IGMP on the interface. The destruction of a row disables IGMP on the interface.

E.3.1.4 igmpInterfaceVersion

The version of IGMP which is running on this interface. MUST be version 2 for all DOCSIS 2.0 CMTS interfaces.

E.3.1.5 igmpInterfaceQuerier

The address of the IGMP Querier on the IP subnet to which this interface is attached.

E.3.1.5.1 Passive Mode

NSI-side: read-only. This is the address of a network side device.

HFC-side: read-only. Same as NSI-side value.

E.3.1.5.2 Active Mode

NSI-side: read-only.

HFC-side: read-only. Active CMTSs MUST report this as an IP Address assigned to the CMTS's HFC-side interface. That is, queries MUST not originate from CMs or CPE.

E.3.1.6 igmpInterfaceQueryMaxResponseTime

The maximum query response time advertised in IGMPv2 queries on this interface.

E.3.1.6.1 Passive Mode

NSI-side: n/a, read-only. return a value of zero.

HFC-side: read-only. This value is derived from observation of queries received from a network side querier.

E.3.1.6.2 Active Mode

NSI-side: read-create. Min = 0; Max = 255; Default = 100.

HFC-side: read-create. Min = 0; Max = 255; Default = 100.

E.3.1.7 igmpInterfaceQuerierUpTime

The time since igmpInterfaceQuerier was last changed.

E.3.1.7.1 PassiveMode

NSI-side: read-only.

HFC-side: n/a, read-only. Return a value of zero.

E.3.1.7.2 Active Mode

NSI-side: read-only.

HFC-side: read-only.

E.3.1.8 igmpInterfaceQuerierExpiryTime

The amount of time remaining before the other querier present timer expires. If the local system is the queier, the value of this object is zero.

E.3.1.8.1 Passive Mode

Both interfaces: n/a, read-only. The CMTS is not the querier, return 0.

E.3.1.8.2 Active Mode

NSI-side: read-only.

HFC-side: read-only. The CMTS MUST be the only querier on the HFC.

E.3.1.9 igmpInterfaceVersion1QuerierTimer

The time remaining until the host assumes that there are no IGMPv1 routers present on the interface. While this is non-zero, the host will reply to all queries with version 1 membership reports.

E.3.1.9.1 Passive Mode

NSI-side: n/a read-only. Return a value of zero.

HFC-side: n/a read-only. Return a value of zero.

E.3.1.9.2 Active Mode

NSI-side: read-only.

HFC-side: read-only.

E.3.1.10 igmpInterfaceWrongVersionQueries

The number of queries recived whose IGMP version does not match igmpInterfaceVersion, over the lifetime of the row entry. IGMP requires that all routers on a LAN be configured to run the same version of IGMP. Although, DOCSIS 2.0 requires that all CMTS and CMTSTS devices support IGMPv2, it is possible for a network side querier to be an IGMPv1 querier.

E.3.1.10.1 All Modes

All interfaces: read-only. The number of non-v2 queries received on this interface.

E.3.1.11 igmpInterfaceJoins

The number of times a group membership has been added on this interface; that is, the number of times an entry for this interface has been added to the Cache Table. This object gives an indication of the amount of IGMP activity over the lifetime of the row entry.

E.3.1.11.1 Passive Mode

NSI-side: n/a read-only. Return a value of zero.

HFC-side: n/a read-only. Return a value of zero.

E.3.1.11.2 Active Mode

NSI-side: read-only.

HFC-side: read-only.

E.3.1.12 igmpInterfaceProxyIfIndex

Some devices implement a form of IGMP proxying whereby memberships learned on the interface represented by this row, cause IGMP Host Membership Reports to be sent on the interface whose ifIndex value is given by this object. Such a device would implement the igmpV2RouterMIBGroup only on its router interfaces (those interfaces with non-zero igmpInterfaceProxyIfIndex). Typically, the value of this object is 0, indicating that no proxying is being done.

E.3.1.12.1 Passive Mode

All Interfaces: read-only. Always return a value of zero.

E.3.1.12.2 Active Mode

NSI-side: read-only.

HFC-side: read-only. Always return an ifIndex for a NSI-side interface.

E.3.1.13 igmpInterfaceGroups

The current number of entries for this interface in the Cache Table.

E.3.1.13.1 Passive Mode

NSI-side: n/a read-only. Return a value of zero.

HFC-side: n/a read-only. Group membership of HFC-side devices.

E.3.1.13.2 Active Mode

NSI-side: read-only.

HFC-side: read-only.

E.3.1.14 igmpInterfaceRobustness

The robustness variable allows tuning for the expected packet loss on a subnet. If a subnet is expected to be lossy, the robustness variable may be increased. IGMP is robust to (robustness variable - 1) packet losses.

E.3.1.14.1 Passive Mode

NSI-side: n/a read-only. Return a value of zero.

HFC-side: n/a read-only. Return a value of zero.

E.3.1.14.2 Active Mode

All interfaces: read-create. Min = 1; Max = (2^32-1); Default = 2

E.3.1.15 igmpInterfaceLastMemberQueryIntvl

The last member query interval is the max response time inserted into group specific queries sent in response to leave group messages, and is also the amount of time between group specific query messages. This value may be tuned to modify the leave latency of the network. A reduced value results in reduced time to detect the loss of the last member of a group.

E.3.1.15.1 Passive Mode

NSI-side: n/a, read-only. return a value of zero.

HFC-side: read-only. This value is derived from observation of queries received from a network side querier.

E.3.1.15.2 Active Mode

NSI-side: read-create. Min = 0; Max = 255; Default = 100.

HFC-side: read-create. Min = 0; Max = 255; Default = 100.

E.3.2 igmpCacheTable - igmpCacheEntry

E.3.2.1 igmpCacheAddress

The IP multicast group address for which this entry contains information.

E.3.2.1.1 All Modes

Not-accessible (index). Report the address of active IP Multicast on the interface.

E.3.2.2 igmpCacheIfIndex

The interface for which this entry contains information for an IP multicast group address.

E.3.2.2.1 Passive Mode

MUST only apply to HFC side interface (e.g., membership is only active on subscriber side of CMTS).

E.3.2.2.2 Active Mode

NSI-side: not-accessible

HFC-side: not-accessible

E.3.2.3 igmpCacheSelf

An indication of whether the local system is a member of this group address on this interface.

E.3.2.3.1 Passive Mode

read-only. MUST be set to FALSE. The CMTS is not a member of any group.

E.3.2.3.2 Active Mode

NSI-side: read-create. Implementation specific (i.e., may apply to RIPv2 or OSPF)

HFC-side: MUST be set to FALSE. The CMTS is not a member of any group on the HFC.

E.3.2.4 igmpCacheLastReporter

The IP address of the source of the last membership report received for this IP Multicast group address on this interface. If no membership report has been received, this object has the value of 0.0.0.0.

E.3.2.4.1 Passive Mode

MUST only apply to last reporter on HFC-side interface (e.g., membership is only active on subscriber side of CMTS).

E.3.2.4.2 Active Mode

NSI-side: read-only

HFC-side: read-only

E.3.2.5 igmpCacheUpTime

The time elapsed since this entry was created.

E.3.2.5.1 Passive Mode

MUST only apply to duration of membership on HFC-side interface (e.g., membership is only active on subscriber side of CMTS).

E.3.2.5.2 Active Mode

NSI-side: read-only

HFC-side: read-only

E.3.2.6 igmpCacheExpiryTime

The minimum amount of time remaining before this entry will be aged out.

E.3.2.6.1 Passive Mode

MUST only apply to duration of membership on HFC-side interface (e.g., membership is only active on subscriber side of CMTS).

E.3.2.6.2 Active Mode

NSI-side: read-only

HFC-side: read-only

E.3.2.7 igmpCacheStatus

The status of this entry.

E.3.2.7.1 Passive Mode

read-create MUST only apply to membership on HFC-side interface (e.g., membership is only active on subscriber side of CMTS). Deletion of a row results in preventing downstream forwarding to this IP Multicast group address on this interface.

E.3.2.7.2 Active Mode

NSI-side: read-create

HFC-side: read-create

E.3.2.8 igmpCacheVersion1HostTimer

The time remaining until the local querier will assume that there are no longer any IGMP version 1 members on thie IP subnet attached to this interface. Upon hearing any IGMPv1 membership report, this value is reset to the group membership timer. While this time remaining is non-zero, the local querier ignores any IGMPv2 leave messages for this group that it receives on this interface.

E.3.2.8.1 Passive Mode

All interfaces: n/a, read-only. Return a value of zero.

E.3.2.8.2 Active Mode

NSI-side: read-only.

HFC-side: read-only.

E.3.3 IGMP MIB Compliance

E.3.3.1 docsIgmpV2PassiveDeviceCompliance

```
docsIgmpV2PassiveDeviceCompliance MODULE-COMPLIANCE
    STATUS current
    DESCRIPTION
    "The compliance statement for DOCSIS Devices passively running IGMPv2 and
     implementing the IGMP MIB."
    MODULE - this module
    MANDATORY-GROUPS { igmpBaseMIBGroup,
                       igmpRouterMIBGroup,
                       igmpV2RouterMIBGroup
                       }
    OBJECT igmpInterfaceStatus
    MIN-ACCESS read-only
    DESCRIPTION
    "Write access is not required."

    OBJECT igmpCacheStatus
    MIN-ACCESS read-only
    DESCRIPTION
    "Write access is not required."

    ::= {docsIgmpMIBCompliances 1}
```

E.3.3.2 docsIgmpV2ActiveDeviceCompliance

```
docsIgmpV2ActiveCmCompliance MODULE-COMPLIANCE
    STATUS current
    DESCRIPTION
    "The compliance statement for DOCSIS Devices actively running IGMPv2 and
     implementing the IGMP MIB."
    MODULE - this module
    MANDATORY-GROUPS { igmpBaseMIBGroup,
                       igmpV2HostMIBGroup,
                       igmpRouterMIBGroup,
                       igmpV2RouterMIBGroup
                       }
    OBJECT igmpInterfaceStatus
    MIN-ACCESS read-only
    DESCRIPTION
    "Write access is not required."

    OBJECT igmpCacheStatus
    MIN-ACCESS read-only
    DESCRIPTION
    "Write access is not required."
```

```
    ::= {docsIgmpMIBCompliances 2}
```

E.3.4 MIB Groups

See IGMP MIB for a description of the objects included in each group.

E.3.4.1 igmpV2HostMIBGroup

Active Devices only (optional - see notes for igmpCacheSelf).

E.3.4.2 igmpV2RouterMIBGroup

Active and Passive Devices

E.3.4.3 igmpBaseMIBGroup

Active and Passive Devices

E.3.4.4 igmpV2RouterMIBGroup

Active and Passive Devices

E.3.4.5 igmpRouterMIBGroup

Active and Passive Devices

E.3.4.6 igmpV2HostOptMIBGroup

Active and Passive Devices

E.3.4.7 igmpV2ProxyMIBGroup

Active Devices only.

Annex F Expected Behaviors for DOCSIS 2.0 Modem in 1.0, 1.1, and 2.0 Modes in OSS Area (normative)[1]

The following table identifies DOCSIS OSSI 2.0 CM features that MAY and MUST be implemented in 1.1 or 1.0 mode.

Specific requirement	Required behavior, DOCSIS 2.0 Modem in 1.0 Mode	Required behavior, DOCSIS 2.0 Modem in 1.1 Mode	Required behavior, DOCSIS 2.0 Modem in 2.0 Mode
Assignment of event-id	SHOULD support a 32-bit number with the following requirement: 1) Top bit is set to 0 for DOCSIS standard events; 2) top bit is set to 1 for vendor proprietary events.	MUST be a 32-bit number. Top bit is set to 0 for DOCSIS standard events. Top bit is set to 1 for vendor proprietary events.	MUST be a 32-bit number. Top bit is set to 0 for DOCSIS standard events. Top bit is set to 1 for vendor proprietary events.
Event Definitions	CM SHOULD support DOCSIS standard events defined in the OSSI 2.0 specification.	CM MUST support DOCSIS standard events defined in the OSSI 2.0 specification.	CM MUST support DOCSIS standard events defined in the OSSI 2.0 specification.
Default handling of events by priority. (Whether to store locally, send trap, or syslog message)	CM SHOULD behave as follow: Error and notice events are stored locally and sent as traps and syslog messages. Other event levels are stored only to the local log, except for informational and debug which are not stored or sent as traps or syslog messages.	CM MUST behave as follows: Error and notice events are stored locally and send traps and syslog messages. Other event levels store only to the local log, except for informational and debug which are not stored or cause any traps or syslog messages.	CM MUST behave as follows: Error and notice events are stored locally and send traps and syslog messages. Other event levels store only to the local log, except for informational and debug which are not stored or cause any traps or syslog messages.
Meaning of event levels	CM SHOULD support event level definitions specified by the OSSI 2.0 specification.	CM MUST support event level definitions specified by the OSSI 2.0 specification.	CM MUST support event level definitions specified by the OSSI 2.0 specification.

1. table modified per oss2-n-02062, 05/17/02, ab

Specific requirement	**Required behavior, DOCSIS 2.0 Modem in 1.0 Mode**	**Required behavior, DOCSIS 2.0 Modem in 1.1 Mode**	**Required behavior, DOCSIS 2.0 Modem in 2.0 Mode**
Event storage in docsDevEventTable	Each entry in the dosDevEventTable contains an event-ID (identical to the Eventid requirement specified in Section 7.4.2.2.2), event time stamp when the event occurred first time and last time, number of appearances and event description in human-readable English format. Total length of the each event description entry MUST not be longer than 255 characters (max. defined for SNMPadminString). Each event, or group of consecutive events with identical eventIds MUST constitute at least one row in the docsDevEvReporting table. For groups of consecutive events with identical eventIds, the CM MAY choose to store only a single row. In such a case, the event text of that row MUST match that of the most recent event. The event count MUST represent the number of events associated with that row. The first and last time columns MUST contain the time at which the least recent and most recent events associated with the row occurred respectively.	Each entry in the dosDevEventTable contains an event-ID (identical to the Eventid requirement specified in Section 7.4.2.2.2), event time stamp when the event occurred first time and last time, number of appearances and event description in human-readable English format. Total length of the each event description entry MUST not be longer than 255 characters (max. defined for SNMPadminString). Each event, or group of consecutive events with identical eventIds MUST constitute at least one row in the docsDevEvReporting table. For groups of consecutive events with identical eventIds, the CM MAY choose to store only a single row. In such a case, the event text of that row MUST match that of the most recent event. The event count MUST represent the number of events associated with that row. The first and last time columns MUST contain the time at which the least recent and most recent events associated with the row occurred respectively.	Each entry in the dosDevEventTable contains an event-ID (identical to the Eventid requirement specified in Section 7.4.2.2.2), event time stamp when the event occurred first time and last time, number of appearances and event description in human-readable English format. Total length of the each event description entry MUST not be longer than 255 characters (max. defined for SNMPadminString). Each event, or group of consecutive events with identical eventIds MUST constitute at least one row in the docsDevEvReporting table. For groups of consecutive events with identical eventIds, the CM MAY choose to store only a single row. In such a case, the event text of that row MUST match that of the most recent event. The event count MUST represent the number of events associated with that row. The first and last time columns MUST contain the time at which the least recent and most recent events associated with the row occurred respectively.
Number of rows in docsDevEventTable	CM MUST support a minimum of 10 rows of docsDevEventTable.	CM MAY support a minimum of 10 rows of docsDevEventTable.	CM MAY support a minimum of 10 rows of docsDevEventTable.
Event log persistence	Event log MUST persist across reboots	Event log MUST persist across reboots.	Event log MUST persist across reboots.
SNMP Version of Trap Control (when CM is in SNMP v1/ v2c DocsDevNm Access mode)	CM MUST implement docsDevNmAccessTrapVersion, which controls whether SNMP V1 or V2 traps are sent.	CM MUST implement docsDevNmAccessTrapVersion, which controls whether SNMP V1 or V2 traps are sent.	CM MUST implement docsDevNmAccessTrapVersion, which controls whether SNMP V1 or V2 traps are sent.
Syslog message format	CM SHOULD support the syslog message with the format:<level>CABLEMODEM [vendor]: <eventId> textOR<level>Cablemodem [vendor]: text	CM MUST support the syslog message with the format: <level>CABLEMODEM [vendor]: <eventId> text	CM MUST support the syslog message with the format: <level>CABLEMODEM [vendor]: <eventId> text
SNMP Protocol Requirement	CM MUST support SNMP v1/ v2c and SNMPv3 with DH. CM must support SNMP requirements specified in Section 5.2 of the OSSI.	CM MUST support SNMP v1/ v2c and SNMPv3 with DH	CM MUST support SNMP v1/ v2c and SNMPv3 with DH

Specific requirement	Required behavior, DOCSIS 2.0 Modem in 1.0 Mode	Required behavior, DOCSIS 2.0 Modem in 1.1 Mode	Required behavior, DOCSIS 2.0 Modem in 2.0 Mode
MIBs to implement	CM MUST support MIB objects as specified by Annex A.	CM MUST support MIB objects as specified by Annex A.	CM MUST support MIB objects as specified by Annex A.
Deprecated MIB objects	Deprecated object is optional. If supported, the object MUST be implemented correctly. If not supported, the object MUST return appropriate SNMP error notifying that the object does not exist.	Deprecated object is optional. If supported, the object MUST be implemented correctly. If not supported, the object MUST return appropriate SNMP error notifying that the object does not exist.	Deprecated object is optional. If supported, the object MUST be implemented correctly. If not supported, the object MUST return appropriate SNMP error notifying that the object does not exist.
Configuration Management	CM MUST support configuration management requirement as specified by Section 7.2 of the OSSI 2.0 specification.	CM MUST support configuration management requirement as specified by Section 7.2 of the OSSI 2.0 specification.	CM MUST support configuration management requirement as specified by Section 7.2 of the OSSI 2.0 specification.
IP/LLC filters	CM SHOULD support LLC/IP filter requirement as specified by OSSI 2.0 specification.	CM MUST support LLC/IP filter requirement as specified by OSSI 2.0 specification.	CM MUST support LLC/IP filter requirement as specified by OSSI 2.0 specification.
CM interaction with CM configuration file	CM MUST process TLV type 11 entries in a configuration file as specified by Section 6.4 of the OSSI 2.0 specification.	CM MUST process TLV type 11 entries in a configuration file as specified by Section 6.4 of the OSSI 2.0 specification.	CM MUST process TLV type 11 entries in a configuration file as specified by Section 6.4 of the OSSI 2.0 specification.
Additional MIB objects requirement	CM MUST implement additional MIB object requirements (on top of RFCs) as specified in Section 6.3 of the OSSI 2.0 specification.	CM MUST implement additional MIB object requirements (on top of RFCs) as specified in Section 6.3 of the OSSI 2.0 specification.	CM MUST implement additional MIB object requirements (on top of RFCs) as specified in Section 6.3 of the OSSI 2.0 specification.
Performance management	CM MUST support performance management requirements as specified by Section 7.5 of the OSSI 2.0 specification.	CM MUST support performance management requirements as specified by Section 7.5 of the OSSI 2.0 specification.	CM MUST support performance management requirements as specified by Section 7.5 of the OSSI 2.0 specification.
OSS for CMCI	CM MUST support CMCI requirements as specified by Section 9 of the OSSI 2.0 specification.	CM MUST support CMCI requirements as specified by Section 9 of the OSSI 2.0 specification.	CM MUST support CMCI requirements as specified by Section 9 of the OSSI 2.0 specification.

This page intentionally left blank.

Annex G DOCS-IF-EXT-MIB (normative)

All objects included in the DOCS-IF-EXT-MIB have corresponding objects in the MIB specified in RFI-MIB-IPCDN-DRAFT.

A 2.0 CMTS and a 2.0 CM in 2.0 mode MUST implement both DOCS-IF-EXT-MIB and RFI-MIB-IPCDN-DRAFT.

It is intended that an ECN will be released later requiring all CMs in all modes to support RFI-MIB-IPCDN-DRAFT and deprecate DOCS-IF-EXT-MIB.

```
This MIB extends the RFC2670 DOCS-IF-MIB with three new objects defined.

The new object, docsIfDocsisCapability, is used to indicate the DOCSIS capability
of a cable device, that is whether it is DOCSIS1.1 capable or DOCSIS1.0 capable.

The new object, docsIfDocsisOperMode, is used to indicate whether it is registered
as a DOCSIS1.1 device or DOCSIS1.0 device.

The new object, docsIfCmtsCmStatusDocsisMode, which augments the docsIfCmtsCmStatusTable in
DOCS-IF-MIB, is used to indicate whether a CM is registered as DOCSIS1.1 modem or DOCSIS1.0
modem.

DOCS-IF-EXT-MIB DEFINITIONS ::= BEGIN

  IMPORTS
        MODULE-IDENTITY,
        OBJECT-TYPE
                FROM SNMPv2-SMI
        OBJECT-GROUP,
        MODULE-COMPLIANCE
                FROM SNMPv2-CONF
    TEXTUAL-CONVENTION
        FROM SNMPv2-TC
        docsIfMib,
        docsIfCmtsCmStatusEntry
                FROM DOCS-IF-MIB;

docsIfExtMib MODULE-IDENTITY
        LAST-UPDATED    "0011160000Z" -- November 16, 2000
        ORGANIZATION    "IETF IPCDN Working Group"
        CONTACT-INFO
            "  "
        DESCRIPTION
            "This is the extension Module to rfc2670 DOCS-IF-MIB."
        REVISION "0010080000Z"
        DESCRIPTION
            "Initial Version. "
        ::= { docsIfMib 21 }

-- Textual Conventions
DocsisVersion ::= TEXTUAL-CONVENTION
    STATUS          current
    DESCRIPTION     "Indicates the docsis version number."
    SYNTAX          INTEGER {
                        docsis10 (1),
                        docsis11 (2)
                    }

docsIfDocsisCapability OBJECT-TYPE
```

```
        SYNTAX      DocsisVersion
        MAX-ACCESS  read-only
        STATUS      current
        DESCRIPTION
             "Indication of the DOCSIS capability of the device.

             "
        ::= { docsIfExtMib 1 }

docsIfDocsisOperMode OBJECT-TYPE
        SYNTAX      DocsisVersion
        MAX-ACCESS  read-only
        STATUS      current
        DESCRIPTION
             "Indication whether the device has registered as a 1.0 or 1.1.

              For CMTS and unregistered CM, it is always the same as docsDevDocsisCapability.

             "
        ::= { docsIfExtMib 2 }


--
-- CM status table (within CMTS).
-- This table is implemented only at the CMTS.
-- It contains per CM status information available in the CMTS.
--

docsIfCmtsCmStatusExtTable OBJECT-TYPE
        SYNTAX      SEQUENCE OF DocsIfCmtsCmStatusExtEntry
        MAX-ACCESS  not-accessible
        STATUS      current
        DESCRIPTION
            "A set of objects in the CMTS, maintained for each
             Cable Modem connected to this CMTS."
        ::= { docsIfExtMib 3 }

docsIfCmtsCmStatusExtEntry OBJECT-TYPE
        SYNTAX      DocsIfCmtsCmStatusExtEntry
        MAX-ACCESS  not-accessible
        STATUS      current
        DESCRIPTION
            "Status information for a single Cable Modem.
             An entry in this table exists for each Cable Modem
             which is connected to the CMTS."
        AUGMENTS { docsIfCmtsCmStatusEntry }
        ::= { docsIfCmtsCmStatusExtTable 1 }

DocsIfCmtsCmStatusExtEntry ::= SEQUENCE {
            docsIfCmtsCmStatusDocsisMode       DocsisVersion
        }

docsIfCmtsCmStatusDocsisMode OBJECT-TYPE
        SYNTAX      DocsisVersion
        MAX-ACCESS  read-only
        STATUS      current
        DESCRIPTION
            "Indication whether the CM has registered as a 1.0 or 1.1 modem              "
        ::= { docsIfCmtsCmStatusExtEntry 1 }

docsIfExtConformance  OBJECT IDENTIFIER     ::= { docsIfExtMib 4 }
docsIfExtCompliances  OBJECT IDENTIFIER     ::= { docsIfExtConformance 1 }
docsIfExtGroups       OBJECT IDENTIFIER     ::= { docsIfExtConformance 2 }

-- compliance statements
```

```
docsIfExtCmCompliance MODULE-COMPLIANCE
        STATUS      current
        DESCRIPTION
            "The compliance statement."

MODULE  -- docsIfExtMib

-- unconditionally mandatory groups for CM
MANDATORY-GROUPS {
 docsIfDocsisVersionGroup
        }
        ::= { docsIfExtCompliances 1 }


docsIfDocsisVersionGroup OBJECT-GROUP
        OBJECTS {
             docsIfDocsisCapability,
             docsIfDocsisOperMode
        }
        STATUS      current
        DESCRIPTION
            "Object group to indicates DOCSIS version."
        ::= { docsIfExtGroups 1 }

docsIfExtCmtsCompliance MODULE-COMPLIANCE
        STATUS      current
        DESCRIPTION
            "The compliance statement."

MODULE  -- docsIfExtMib

-- unconditionally mandatory groups for CMTS

MANDATORY-GROUPS {
        docsIfExtGroup,
        docsIfDocsisVersionGroup
        }
        ::= { docsIfExtCompliances 2 }
docsIfExtGroup OBJECT-GROUP
        OBJECTS {
            docsIfCmtsCmStatusDocsisMode
        }
        STATUS      current
        DESCRIPTION
            "Mandatory implementation group for CMTS."
        ::= { docsIfExtGroups 2 }

END
```

This page intentionally left blank.

Annex H DOCS-CABLE-DEVICE-TRAP-MIB (normative)[1]

```
DOCS-CABLE-DEVICE-TRAP-MIB DEFINITIONS ::= BEGIN

IMPORTS
MODULE-IDENTITY,
NOTIFICATION-TYPE
FROM SNMPv2-SMI

MODULE-COMPLIANCE,
NOTIFICATION-GROUP
FROM SNMPv2-CONF

docsDev,
--docsDevBase,
docsDevEvLevel,
docsDevEvId,
docsDevEvText,
docsDevSwFilename,
docsDevSwServer,
docsDevServerDhcp,
docsDevServerTime,
docsDevNotification
FROM DOCS-CABLE-DEVICE-MIB --RFC2669

docsIfCmCmtsAddress,
docsIfCmtsCmStatusMacAddress,
docsIfDocsisBaseCapability,
docsIfCmStatusDocsisOperMode,
docsIfCmStatusModulationType,
docsIfCmtsCmStatusDocsisRegMode,
docsIfCmtsCmStatusModulationType
FROM DOCS-IF-MIB -- draft-ietf-ipcdn-docs-rfmibv2-02

docsIfDocsisCapability, -- deprecated
docsIfDocsisOperMode, -- deprecated
docsIfCmtsCmStatusDocsisMode -- deprecated
FROM DOCS-IF-EXT-MIB -- deprecated

ifPhysAddress
FROM IF-MIB;

docsDevTrapMIB MODULE-IDENTITY

LAST-UPDATED "0202250000Z"
ORGANIZATION "Cisco Systems, Inc."
CONTACT-INFO "
Junming Gao
Cisco Systems Inc
<jgao@ cisco. com>
"

DESCRIPTION
"Modified by David Raftus (david.raftus@imedia.com) to deprecate
trap definition objects originating from the docsIfExt MIB.
Corresponding objects from the Docsis 2.0 RF MIB draft were added
to the trap definitions."

REVISION "000926000000Z"
```

1. MIB replaced per ossi2-n-02016, 05/15/02, ab

```
DESCRIPTION
"The CABLE DEVICE TRAP MIB is an extension of the
CABLE DEVICE MIB defined in RFC2669.
It defines various trap objects for both cable
modem and cable modem termination systems.
Two groups of SNMP notification objects are defined.
One group is for notifying cable modem events and one group
for notifying cable modem termination system events.
Common to all CM notification objects (traps) is that
their OBJECTS statements contain information
about the event priority, the event Id, the event message
body, the CM DOCSIS capability, the CM DOCSIS QOS level, the CM
DOCSIS upstream modulation type, the cable interface MAC address
of the cable modem and the cable card MAC address of the CMTS to
which the modem is connected.

These objects are docsDevEvLevel, docsDevId, docsDevEvText,
docsIfDocsisBaseCapability, docsIfCmStatusDocsisOperMode,
docsIfCmStatusModulationType,ifPhysAddress and
docsIfCmCmtsAddress. The values of docsDevEvLevel, docsDevId, and
docsDevEvText are from the entry which logs this event in the
docsDevEventTable, which is defined in
DOCS-CABLE-DEVICE-MIB of RFC2669. The docsIfDocsisBaseCapability,
docsIfCmStatusDocsisOperMode, and docsIfCmStatusModulationType
are defined in the DOCS-IF-MIB.
The ifPhysAddress value is the MAC address of the cable
interface of this cable modem. The docsIfCmCmtsAddress
specifies the MAC address of the CMTS (if there is a cable
card/ interface in the CMTS, then it is actually the
cable interface interface MAC address to which the CM is
connected).
Individual CM trap may contain additional objects to
provide necessary information.

Common to all CMTS notification objects (traps) is that their
OBJECTS statements contain information about the event priority,
the event Id, the event message body, the connected CM DOCSIS QOS
status, the connected CM DOCSIS modulation type, the CM cable
interface MAC address, the CMTS DOCSIS capability, and
the CMTS MAC address.

These objects are docsDevEvLevel, docsDevId, docsDevEvText,
docsIfCmtsCmStatusDocsisRegMode,
docsIfCmtsCmStatusModulationType, docsIfCmtsCmStatusMacAddress,
docsIfDocsisBaseCapability, and ifPhysAddress. The values of
docsDevEvLevel, docsDevId, and docsDevEvText are
similar to those in CM traps. The values of
docsIfCmtsCmStatusDocsisRegMode,
docsIfCmtsCmStatusModulationType, and
docsIfCmtsCmStatusMacAddress are from the
docsIfCmtsCmStatusEntry (defined in DOCS-IF-MIB)
corresponding to a connected CM. The docsIfDocsisBaseCapability
indicates the CMTS DOCSIS capability.
The ifPhysAddress value is the CMTS MAC address (if there is a
cable
card/ interface in the CMTS, then it is actually the MAC address
of the cable
interface which connected to the CM).
"

::= { docsDev 10 }

--
--docsDevNotification OBJECT IDENTIFIER ::= { docsDev 2 }
--
```

```
docsDevTraps OBJECT IDENTIFIER ::= { docsDevNotification 1 }
docsDevTrapControl OBJECT IDENTIFIER ::= { docsDevTraps 1}
docsDevCmTraps OBJECT IDENTIFIER ::= { docsDevTraps 2 0 }
docsDevCmtsTraps OBJECT IDENTIFIER ::= { docsDevTraps 3 0 }

docsDevCmTrapControl OBJECT-TYPE

SYNTAX BITS {
cmInitTLVUnknownTrap( 0),
cmDynServReqFailTrap( 1),
cmDynServRspFailTrap( 2),
cmDynServAckFailTrap( 3),
cmBpiInitTrap( 4),
cmBPKMTrap( 5),
cmDynamicSATrap( 6),
cmDHCPFailTrap( 7),
cmSwUpgradeInitTrap( 8),
cmSwUpgradeFailTrap( 9),
cmSwUpgradeSuccessTrap( 10),
cmSwUpgradeCVCTrap( 11),
cmTODFailTrap( 12),
cmDCCReqFailTrap( 13),
cmDCCRspFailTrap( 14),
cmDCCAckFailTrap( 15)
}

MAX-ACCESS read-write
STATUS current
DESCRIPTION
"The object is used to enable CM traps. From left to right,
the set bit indicates the corresponding CM trap is enabled.
For example, if the first bit is set, then
docsDevCmInitTLVUnknownTrap is enabled. If it is zero,
the trap is disabled.
"

DEFVAL { '00'h }
::= { docsDevTrapControl 1 }

docsDevCmtsTrapControl OBJECT-TYPE

SYNTAX BITS {
cmtsInitRegReqFailTrap( 0),
cmtsInitRegRspFailTrap( 1),
cmtsInitRegAckFailTrap( 2),
cmtsDynServReqFailTrap( 3),
cmtsDynServRspFailTrap( 4),
cmtsDynServAckFailTrap( 5),
cmtsBpiInitTrap( 6),
cmtsBPKMTrap( 7),
cmtsDynamicSATrap( 8),
cmtsDCCReqFailTrap( 9),
cmtsDCCRspFailTrap( 10),
cmtsDCCAckFailTrap( 11)
}

MAX-ACCESS read-write
STATUS current
DESCRIPTION
"The object is used to enable CMTS traps. From left to right,
the set bit indicates the corresponding CMTS trap is enabled.
For example, if the first bit is set, then
docsDevCmtsInitRegRspFailTrap is enabled. If it is zero,
the trap is disabled.
"
```

```
DEFVAL { '00'h }
::= { docsDevTrapControl 2 }

docsDevCmInitTLVUnknownTrap NOTIFICATION-TYPE
OBJECTS { docsDevEvLevel,
docsDevEvId,
docsDevEvText,
docsIfDocsisCapability, -- deprecated
docsIfDocsisOperMode, -- deprecated
ifPhysAddress,
docsIfCmCmtsAddress,
docsIfDocsisBaseCapability,
docsIfCmStatusDocsisOperMode,
docsIfCmStatusModulationType }

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 docsIfDocsisBaseCapability
indicates the DOCSIS version information. The
docsIfCmStatusDocsisOperMode indicates the QOS level of the CM,
while the docsIfCmStatusModulationType indicates the upstream
modulation methodology used by the CM.
The ifPhysAddress value is the MAC address of the cable interface
of this cable modem.
The docsIfCmCmtsAddress specifies the MAC address
of the CMTS to which the CM is connected (if there is a cable
card/ interface in the CMTS, then it is actually the MAC address
of the cable
interface which connected to the CM).
This part of information is uniformed across all CM traps.
"

::= { docsDevCmTraps 1 }

docsDevCmDynServReqFailTrap NOTIFICATION-TYPE
OBJECTS { docsDevEvLevel,
docsDevEvId,
docsDevEvText,
docsIfDocsisCapability, -- deprecated
docsIfDocsisOperMode, -- deprecated
ifPhysAddress,
docsIfCmCmtsAddress,
docsIfDocsisBaseCapability,
docsIfCmStatusDocsisOperMode,
docsIfCmStatusModulationType }

STATUS current
DESCRIPTION
"An event to report the failure of a dynamic service
request happened during the dynamic services process.
"

::= { docsDevCmTraps 2 }

docsDevCmDynServRspFailTrap NOTIFICATION-TYPE
OBJECTS { docsDevEvLevel,
docsDevEvId,
docsDevEvText,
docsIfDocsisCapability, -- deprecated
docsIfDocsisOperMode, -- deprecated
ifPhysAddress,
```

```
docsIfCmCmtsAddress,
docsIfDocsisBaseCapability,
docsIfCmStatusDocsisOperMode,
docsIfCmStatusModulationType }

STATUS current
DESCRIPTION
"An event to report the failure of a dynamic service
response happened during the dynamic services process.
"

::= { docsDevCmTraps 3}

docsDevCmDynServAckFailTrap NOTIFICATION-TYPE
OBJECTS { docsDevEvLevel,
docsDevEvId,
docsDevEvText,
docsIfDocsisCapability, -- deprecated
docsIfDocsisOperMode, -- deprecated
ifPhysAddress,
docsIfCmCmtsAddress,
docsIfDocsisBaseCapability,
docsIfCmStatusDocsisOperMode,
docsIfCmStatusModulationType }

STATUS current
DESCRIPTION
"An event to report the failure of a dynamic service
acknowledgement happened during the dynamic services process.
"

::= { docsDevCmTraps 4}

docsDevCmBpiInitTrap NOTIFICATION-TYPE
OBJECTS { docsDevEvLevel,
docsDevEvId,
docsDevEvText,
docsIfDocsisCapability, -- deprecated
docsIfDocsisOperMode, -- deprecated
ifPhysAddress,
docsIfCmCmtsAddress,
docsIfDocsisBaseCapability,
docsIfCmStatusDocsisOperMode,
docsIfCmStatusModulationType }

STATUS current
DESCRIPTION
"An event to report the failure of a BPI initialization
attempt happened during the registration process.
"

::= { docsDevCmTraps 5 }

docsDevCmBPKMTrap NOTIFICATION-TYPE
OBJECTS { docsDevEvLevel,
docsDevEvId,
docsDevEvText,
docsIfDocsisCapability, -- deprecated
docsIfDocsisOperMode, -- deprecated
ifPhysAddress,
docsIfCmCmtsAddress,
docsIfDocsisBaseCapability,
docsIfCmStatusDocsisOperMode,
docsIfCmStatusModulationType }
```

```
STATUS current
DESCRIPTION
"An event to report the failure of a BPKM operation.
"

::= { docsDevCmTraps 6 }

docsDevCmDynamicSATrap NOTIFICATION-TYPE
OBJECTS { docsDevEvLevel,
docsDevEvId,
docsDevEvText,
docsIfDocsisCapability, -- deprecated
docsIfDocsisOperMode, -- deprecated
ifPhysAddress,
docsIfCmCmtsAddress,
docsIfDocsisBaseCapability,
docsIfCmStatusDocsisOperMode,
docsIfCmStatusModulationType }

STATUS current
DESCRIPTION
"An event to report the failure of a dynamic security
association operation.
"

::= { docsDevCmTraps 7 }

docsDevCmDHCPFailTrap NOTIFICATION-TYPE
OBJECTS { docsDevEvLevel,
docsDevEvId,
docsDevEvText,
docsIfDocsisCapability, -- deprecated
docsIfDocsisOperMode, -- deprecated
ifPhysAddress,
docsIfCmCmtsAddress,
docsDevServerDhcp,
docsIfDocsisBaseCapability,
docsIfCmStatusDocsisOperMode,
docsIfCmStatusModulationType }

STATUS current
DESCRIPTION
"An event to report the failure of a DHCP server.
The value of docsDevServerDhcp is the IP address
of the DHCP server.
"

::= { docsDevCmTraps 8 }

docsDevCmSwUpgradeInitTrap NOTIFICATION-TYPE
OBJECTS { docsDevEvLevel,
docsDevEvId,
docsDevEvText,
docsIfDocsisCapability, -- deprecated
docsIfDocsisOperMode, -- deprecated
ifPhysAddress,
docsIfCmCmtsAddress,
docsDevSwFilename,
docsDevSwServer,
docsIfDocsisBaseCapability,
docsIfCmStatusDocsisOperMode,
docsIfCmStatusModulationType }

STATUS current
DESCRIPTION
```

```
"An event to report a software upgrade initiated
event. The values of docsDevSwFilename, and
docsDevSwServer indicate the software image name
and the server IP address the image is from.
"

::= { docsDevCmTraps 9 }

docsDevCmSwUpgradeFailTrap NOTIFICATION-TYPE
OBJECTS { docsDevEvLevel,
docsDevEvId,
docsDevEvText,
docsIfDocsisCapability, -- deprecated
docsIfDocsisOperMode, -- deprecated
ifPhysAddress,
docsIfCmCmtsAddress,
docsDevSwFilename,
docsDevSwServer,
docsIfDocsisBaseCapability,
docsIfCmStatusDocsisOperMode,
docsIfCmStatusModulationType }

STATUS current
DESCRIPTION
"An event to report the failure of a software upgrade
attempt. The values of docsDevSwFilename, and
docsDevSwServer indicate the software image name
and the server IP address the image is from.
"

::= { docsDevCmTraps 10 }

docsDevCmSwUpgradeSuccessTrap NOTIFICATION-TYPE
OBJECTS { docsDevEvLevel,
docsDevEvId,
docsDevEvText,
docsIfDocsisCapability, -- deprecated
docsIfDocsisOperMode, -- deprecated
ifPhysAddress,
docsIfCmCmtsAddress,
docsDevSwFilename,
docsDevSwServer,
docsIfDocsisBaseCapability,
docsIfCmStatusDocsisOperMode,
docsIfCmStatusModulationType }

STATUS current
DESCRIPTION
"An event to report the Software upgrade success event.
The values of docsDevSwFilename, and
docsDevSwServer indicate the software image name
and the server IP address the image is from.
"

::= { docsDevCmTraps 11 }

docsDevCmSwUpgradeCVCFailTrap NOTIFICATION-TYPE
OBJECTS { docsDevEvLevel,
docsDevEvId,
docsDevEvText,
docsIfDocsisCapability, -- deprecated
docsIfDocsisOperMode, -- deprecated
ifPhysAddress,
docsIfCmCmtsAddress,
docsIfDocsisBaseCapability,
```

```
docsIfCmStatusDocsisOperMode,
docsIfCmStatusModulationType }

STATUS current
DESCRIPTION
"An event to report the failure of the verification
of code file happened during a secure software upgrade
attempt.
"

::= { docsDevCmTraps 12 }

docsDevCmTODFailTrap NOTIFICATION-TYPE
OBJECTS { docsDevEvLevel,
docsDevEvId,
docsDevEvText,
docsIfDocsisCapability, -- deprecated
docsIfDocsisOperMode, -- deprecated
ifPhysAddress,
docsIfCmCmtsAddress,
docsDevServerTime,
docsIfDocsisBaseCapability,
docsIfCmStatusDocsisOperMode,
docsIfCmStatusModulationType }

STATUS current
DESCRIPTION
"An event to report the failure of a time of day server.
The value of docsDevServerTime indicates the server IP
address.
"

::= { docsDevCmTraps 13 }

docsDevCmDCCReqFailTrap NOTIFICATION-TYPE
OBJECTS { docsDevEvLevel,
docsDevEvId,
docsDevEvText,
docsIfDocsisCapability, -- deprecated
docsIfDocsisOperMode, -- deprecated
ifPhysAddress,
docsIfCmCmtsAddress,
docsIfDocsisBaseCapability,
docsIfCmStatusDocsisOperMode,
docsIfCmStatusModulationType }

STATUS current
DESCRIPTION
"An event to report the failure of a dynamic channel
change request happened during the dynamic channel
change process in the CM side.
"

::= { docsDevCmTraps 14 }

docsDevCmDCCRspFailTrap NOTIFICATION-TYPE
OBJECTS { docsDevEvLevel,
docsDevEvId,
docsDevEvText,
docsIfDocsisCapability, -- deprecated
docsIfDocsisOperMode, -- deprecated
ifPhysAddress,
docsIfCmCmtsAddress,
docsIfDocsisBaseCapability,
docsIfCmStatusDocsisOperMode,
```

```
docsIfCmStatusModulationType }

STATUS current
DESCRIPTION
"An event to report the failure of a dynamic channel
change response happened during the dynamic channel
change process in the CM side.
"

::= { docsDevCmTraps 15 }

docsDevCmDCCAckFailTrap NOTIFICATION-TYPE
OBJECTS { docsDevEvLevel,
docsDevEvId,
docsDevEvText,
docsIfDocsisCapability, -- deprecated
docsIfDocsisOperMode, -- deprecated
ifPhysAddress,
docsIfCmCmtsAddress,
docsIfDocsisBaseCapability,
docsIfCmStatusDocsisOperMode,
docsIfCmStatusModulationType }

STATUS current
DESCRIPTION
"An event to report the failure of a dynamic channel
change acknowledgement happened during the dynamic channel
change process in the CM side.
"

::= { docsDevCmTraps 16}

docsDevCmtsInitRegReqFailTrap NOTIFICATION-TYPE
OBJECTS { docsDevEvLevel,
docsDevEvId,
docsDevEvText,
docsIfCmtsCmStatusDocsisMode, -- deprecated
docsIfCmtsCmStatusMacAddress,
docsIfDocsisCapability, -- deprecated
ifPhysAddress,
docsIfCmtsCmStatusDocsisRegMode,
docsIfDocsisBaseCapability,
docsIfCmtsCmStatusModulationType }

STATUS current
DESCRIPTION
"An event to report the failure of a registration
request from CM happening during the CM initialization
process and detected on the CMTS side.
The values of docsDevEvLevel, docsDevId, and
docsDevEvText are from the entry which logs this event
in the docsDevEventTable. The docsIfCmtsCmStatusDocsisRegMode
and docsIfCmtsCmStatusMacAddress indicate the docsis
QOS version and the MAC address of the requesting CM. The
docsIfCmtsCmModulationType indicates the upstream modulation
methodology used by the connected CM.
The docsIfDocsisBaseCapability and ifPhysAddress
indicate the docsis version of the CMTS and the MAC
address of the CMTS (if there is a cable
card/ interface in the CMTS, then it is actually the MAC address
of the cable interface which connected to the CM) cable card
connected to the CM.
This part of information is uniformed across all CMTS traps.
"
```

```
::= { docsDevCmtsTraps 1 }

docsDevCmtsInitRegRspFailTrap NOTIFICATION-TYPE
OBJECTS { docsDevEvLevel,
docsDevEvId,
docsDevEvText,
docsIfCmtsCmStatusDocsisMode, -- deprecated
docsIfCmtsCmStatusMacAddress,
docsIfDocsisCapability, -- deprecated
ifPhysAddress,
docsIfCmtsCmStatusDocsisRegMode,
docsIfDocsisBaseCapability,
docsIfCmtsCmStatusModulationType }

STATUS current
DESCRIPTION
"An event to report the failure of a registration
response happened during the CM initialization
process and detected in the CMTS side.
"

::= { docsDevCmtsTraps 2 }

docsDevCmtsInitRegAckFailTrap NOTIFICATION-TYPE
OBJECTS { docsDevEvLevel,
docsDevEvId,
docsDevEvText,
docsIfCmtsCmStatusDocsisMode, -- deprecated
docsIfCmtsCmStatusMacAddress,
docsIfDocsisCapability, -- deprecated
ifPhysAddress,
docsIfCmtsCmStatusDocsisRegMode,
docsIfDocsisBaseCapability,
docsIfCmtsCmStatusModulationType }

STATUS current
DESCRIPTION
"An event to report the failure of a registration
acknowledgement from CM happened during the CM
initialization process and detected in the CMTS side.
"

::= { docsDevCmtsTraps 3 }

docsDevCmtsDynServReqFailTrap NOTIFICATION-TYPE
OBJECTS { docsDevEvLevel,
docsDevEvId,
docsDevEvText,
docsIfCmtsCmStatusDocsisMode, -- deprecated
docsIfCmtsCmStatusMacAddress,
docsIfDocsisCapability, -- deprecated
ifPhysAddress,
docsIfCmtsCmStatusDocsisRegMode,
docsIfDocsisBaseCapability,
docsIfCmtsCmStatusModulationType }

STATUS current
DESCRIPTION
"An event to report the failure of a dynamic service
request happened during the dynamic services process
and detected in the CMTS side.
"

::= { docsDevCmtsTraps 4 }
```

```
docsDevCmtsDynServRspFailTrap NOTIFICATION-TYPE
OBJECTS { docsDevEvLevel,
docsDevEvId,
docsDevEvText,
docsIfCmtsCmStatusDocsisMode, -- deprecated
docsIfCmtsCmStatusMacAddress,
docsIfDocsisCapability, -- deprecated
ifPhysAddress,
docsIfCmtsCmStatusDocsisRegMode,
docsIfDocsisBaseCapability,
docsIfCmtsCmStatusModulationType }

STATUS current
DESCRIPTION
"An event to report the failure of a dynamic service
response happened during the dynamic services process
and detected in the CMTS side.
"

::= { docsDevCmtsTraps 5 }

docsDevCmtsDynServAckFailTrap NOTIFICATION-TYPE
OBJECTS { docsDevEvLevel,
docsDevEvId,
docsDevEvText,
docsIfCmtsCmStatusDocsisMode, -- deprecated
docsIfCmtsCmStatusMacAddress,
docsIfDocsisCapability, -- deprecated
ifPhysAddress,
docsIfCmtsCmStatusDocsisRegMode,
docsIfDocsisBaseCapability,
docsIfCmtsCmStatusModulationType }

STATUS current
DESCRIPTION
"An event to report the failure of a dynamic service
acknowledgement happened during the dynamic services
process and detected in the CMTS side.
"

::= { docsDevCmtsTraps 6 }

docsDevCmtsBpiInitTrap NOTIFICATION-TYPE
OBJECTS { docsDevEvLevel,
docsDevEvId,
docsDevEvText,
docsIfCmtsCmStatusDocsisMode, -- deprecated
docsIfCmtsCmStatusMacAddress,
docsIfDocsisCapability, -- deprecated
ifPhysAddress,
docsIfCmtsCmStatusDocsisRegMode,
docsIfDocsisBaseCapability,
docsIfCmtsCmStatusModulationType }

STATUS current
DESCRIPTION
"An event to report the failure of a BPI initialization
attempt happened during the CM registration process
and detected in the CMTS side.
"

::= { docsDevCmtsTraps 7 }

docsDevCmtsBPKMTrap NOTIFICATION-TYPE
OBJECTS { docsDevEvLevel,
```

```
docsDevEvId,
docsDevEvText,
docsIfCmtsCmStatusDocsisMode, -- deprecated
docsIfCmtsCmStatusMacAddress,
docsIfDocsisCapability, -- deprecated
ifPhysAddress,
docsIfCmtsCmStatusDocsisRegMode,
docsIfDocsisBaseCapability,
docsIfCmtsCmStatusModulationType }

STATUS current
DESCRIPTION
"An event to report the failure of a BPKM operation
which is detected in the CMTS side.
"

::= { docsDevCmtsTraps 8 }

docsDevCmtsDynamicSATrap NOTIFICATION-TYPE
OBJECTS { docsDevEvLevel,
docsDevEvId,
docsDevEvText,
docsIfCmtsCmStatusDocsisMode, -- deprecated
docsIfCmtsCmStatusMacAddress,
docsIfDocsisCapability, -- deprecated
ifPhysAddress,
docsIfCmtsCmStatusDocsisRegMode,
docsIfDocsisBaseCapability,
docsIfCmtsCmStatusModulationType }

STATUS current
DESCRIPTION
"An event to report the failure of a dynamic security
association operation which is detected in the CMTS side.
"

::= { docsDevCmtsTraps 9 }

docsDevCmtsDCCReqFailTrap NOTIFICATION-TYPE
OBJECTS { docsDevEvLevel,
docsDevEvId,
docsDevEvText,
docsIfCmtsCmStatusDocsisMode, -- deprecated
docsIfCmtsCmStatusMacAddress,
docsIfDocsisCapability, -- deprecated
ifPhysAddress,
docsIfCmtsCmStatusDocsisRegMode,
docsIfDocsisBaseCapability,
docsIfCmtsCmStatusModulationType }

STATUS current
DESCRIPTION
"An event to report the failure of a dynamic channel
change request happened during the dynamic channel
change process in the CM side and detected in the
CMTS side.
"

::= { docsDevCmtsTraps 10 }

docsDevCmtsDCCRspFailTrap NOTIFICATION-TYPE
OBJECTS { docsDevEvLevel,
docsDevEvId,
docsDevEvText,
docsIfCmtsCmStatusDocsisMode, -- deprecated
```

```
docsIfCmtsCmStatusMacAddress,
docsIfDocsisCapability, -- deprecated
ifPhysAddress,
docsIfCmtsCmStatusDocsisRegMode,
docsIfDocsisBaseCapability,
docsIfCmtsCmStatusModulationType }

STATUS current
DESCRIPTION
"An event to report the failure of a dynamic channel
change response happened during the dynamic channel
change process in the CMTS side.
"

::= { docsDevCmtsTraps 11 }

docsDevCmtsDCCAckFailTrap NOTIFICATION-TYPE
OBJECTS { docsDevEvLevel,
docsDevEvId,
docsDevEvText,
docsIfCmtsCmStatusDocsisMode, -- deprecated
docsIfCmtsCmStatusMacAddress,
docsIfDocsisCapability, -- deprecated
ifPhysAddress,
docsIfCmtsCmStatusDocsisRegMode,
docsIfDocsisBaseCapability,
docsIfCmtsCmStatusModulationType }

STATUS current
DESCRIPTION
"An event to report the failure of a dynamic channel
change acknowledgement happened during the dynamic channel
change process in the CMTS side.
"

::= { docsDevCmtsTraps 12}
--
--Conformance definitions
--
docsDevTrapConformance OBJECT IDENTIFIER ::= { docsDevTraps 4 }
docsDevTrapGroups OBJECT IDENTIFIER ::= { docsDevTrapConformance 1 }
docsDevTrapCompliances OBJECT IDENTIFIER ::= {
docsDevTrapConformance 2 }
docsDevCmTrapCompliance MODULE-COMPLIANCE

STATUS current
DESCRIPTION
"The compliance statement for Cable Modem Traps and Control"

MODULE --docsDevTrap
--mandatory groups

GROUP docsDevCmTrapControlGroup
DESCRIPTION
"Mandatory in CM."

GROUP docsDevCmNotificationGroup
DESCRIPTION
"Mandatory in Cable Modem."

::= { docsDevTrapCompliances 1 }

docsDevCmTrapControlGroup OBJECT-GROUP
OBJECTS {
docsDevCmTrapControl
```

```
}
STATUS current
DESCRIPTION
"CM must support docsDevCmTrapControl."
::= { docsDevTrapGroups 1 }

docsDevCmNotificationGroup NOTIFICATION-GROUP

NOTIFICATIONS {
docsDevCmInitTLVUnknownTrap,
docsDevCmDynServReqFailTrap,
docsDevCmDynServRspFailTrap,
docsDevCmDynServAckFailTrap,
docsDevCmBpiInitTrap,
docsDevCmBPKMTrap,
docsDevCmDynamicSATrap,
docsDevCmDHCPFailTrap,
docsDevCmSwUpgradeInitTrap,
docsDevCmSwUpgradeFailTrap,
docsDevCmSwUpgradeSuccessTrap,
docsDevCmSwUpgradeCVCFailTrap,
docsDevCmTODFailTrap,
docsDevCmDCCReqFailTrap,
docsDevCmDCCRspFailTrap,
docsDevCmDCCAckFailTrap
}

STATUS current
DESCRIPTION
"A collection of CM notifications providing device status and
control."

::= { docsDevTrapGroups 2 }

docsDevCmtsTrapCompliance MODULE-COMPLIANCE
STATUS current
DESCRIPTION
"The compliance statement for MCNS Cable Modems and
Cable Modem Termination Systems."
MODULE --docsDevTrap
--mandatory groups

GROUP docsDevCmtsTrapControlGroup
DESCRIPTION
"Mandatory in CMTS."

GROUP docsDevCmtsNotificationGroup
DESCRIPTION
"Mandatory in Cable Modem Termination Systems."

::= { docsDevTrapCompliances 2 }

docsDevCmtsTrapControlGroup OBJECT-GROUP
OBJECTS {
docsDevCmtsTrapControl
}

STATUS current
DESCRIPTION
"CMTS must support docsDevCmtsTrapControl."
::= { docsDevTrapGroups 3 }
docsDevCmtsNotificationGroup NOTIFICATION-GROUP

NOTIFICATIONS {
docsDevCmtsInitRegReqFailTrap,
```

```
docsDevCmtsInitRegRspFailTrap,
docsDevCmtsInitRegAckFailTrap ,
docsDevCmtsDynServReqFailTrap,
docsDevCmtsDynServRspFailTrap,
docsDevCmtsDynServAckFailTrap,
docsDevCmtsBpiInitTrap,
docsDevCmtsBPKMTrap,
docsDevCmtsDynamicSATrap,
docsDevCmtsDCCReqFailTrap,
docsDevCmtsDCCRspFailTrap,
docsDevCmtsDCCAckFailTrap
}

STATUS current
DESCRIPTION
"A collection of CMTS notifications providing device status and
control."
::= { docsDevTrapGroups 4 }
END
```

This page intentionally left blank.

Appendix I Business Process Scenarios For Subscriber Account Management (informative)

In order to develop the DOCS-OSS Subscriber Account Management Specification, it is necessary to consider high-level business processes common to cable operators and the associated operational scenarios. The following definitions represent a generic view of key processes involved. It is understood that business process terminology varies among different cable operators, distinguished by unique operating environments and target market segments

For the purpose of this document, Subscriber Account Management refers to the following business processes and terms:

- Class of Service Provisioning Processes, which are involved in the automatic and dynamic provisioning and enforcement of subscribed class of policy-based service level agreements (SLAs)
- Usage-Based Billing Processes, which are involved in the processing of bills based on services rendered to and consumed by paying subscriber customers

I.1 The old service model: “one class only” and “best-effort” service

The Internet is an egalitarian cyber society in its pure technical form where all Internet Protocol (IP) packets are treated as equals. Given that all IP packets have equal right-of-way over the Internet, it is a “one class fits all”, “first-come, first-served” type of service level arrangement. The response time and quality of delivery service is promised to be on a “best-effort” basis only.

Unfortunately, while all IP packets are theoretically equal, certain classes of IP packets must be processed differently. When transmitting data packets, traffic congestion causes no fatal problems except unpredictable delays and frustrations. However, in a convergent IP world where data packets are mixed with those associated with voice and streaming video, such “one-class” service level and "best-effort only" quality is not workable.

I.2 The old billing model: “flat rate” access

As high-speed data-over-cable service deployment moves to the next stage, serious considerations must be made by all cable operators to abandon old business practices, most notably "flat-rate" fee structure. No service provider can hope to stay in business long by continuing to offer a single, "flat-rate" access service to all subscribers, regardless of actual usage.

Imagine your utility bills were the same month after month, whether you used very little water or electricity every day, or if you ran your water and your air conditioning at full blast 24 hours a day. You would be entitled, just like everyone else, to consume as much or as little as you wished, anytime you wanted it. Chances are you would not accept such a service agreement, not only because it is not a fair arrangement, but also because such wasteful consumption would put enough pressure on the finite supply of water and electricity that most of your normal demands for usage would likely go unfulfilled.

I.3 A Successful New Business Paradigm

The new paradigm for delivering IP-based services over cable networks is forcing all cable operators to adopt a new business paradigm. The retention of customers will require that an operator offer different class-of-service options and associated access rates with guaranteed provisioning and delivery of subscribed services. “Back Office” usage-based accounting and subscriber billing will become an important competitive differentiation in the emergence of high-speed data-over-cable services.

I.3.1 Integrating "front end" processes seamlessly with "back office" functions

A long-standing business axiom states that accountability exists only with the right measurements and that business prospers only with the proper management information. An effective subscriber account management system for data over cable services should meet three (3) major requirements:

Automatic & Dynamic Subscriber Provisioning The first requirement is to integrate service subscription orders and changes automatically and dynamically, with the various processes that invoke the provisioning and delivering of subscribed and/or "on-demand" services.

Guaranteed Class & Quality of Services The second requirement is to offer different class of services with varying rates and guarantee the quality of service level associated with each service class.

Data Collection, Warehousing & Usage Billing The third requirement is to capture a subscriber's actual usage, calculating the bill based on the rate associated with the customer's subscribed service levels.

I.3.2 Designing Classes of Services

While designing different class of service offerings, a cable operator might consider the following framework:

- Class of Service by account type: business vs. residential accounts
- Class of Service by guaranteed service levels
- Class of Service by time of day and/or day of week
- "On Demand" Service by special order

The following is a plausible sample of classes of service:

- "Best Effort" Service Without Minimum Guarantee
  This class of "Best Effort Only" service is the normal practice of today where subscribers of this class of service are allocated only excess channel bandwidth available at the time while each subscriber's access is capped at a maximum bandwidth (for example at 512 kilobit per second).
- Platinum Service for Business and High-Access Residential Accounts
  Business accounts subscribing to this service are guaranteed a minimum data rate of downstream bandwidth - 512 kilobit per second - and if excess bandwidth is available, they are allowed to burst to 10 megabit per second.
- Gold Service for Business Accounts
  This class of service guarantees subscribers a 256 kilobit per second downstream data rate during business hours (for example from 8 a.m. to 6 p.m.) and 128 kilobit per second at other times. If excess bandwidth is available at any time, data is allowed to burst to 5 megabit per second.
- Gold Service for Residential Accounts
  Residential subscribers of this service are guaranteed 128 kilobit per second downstream bandwidth during business hours and 256 kilobit per second at other times (for example from 6 p.m. to 8 a.m.), and a maximum data burst rate of 5 megabit per second with available excess bandwidth.
- Silver Service for Business Accounts
  Business accounts subscribing to this service are guaranteed 128 kilobit per second downstream data rate during business hours and 64 kilobit per second during other times, and a maximum burst rate of 1 megabit per second.
- Silver Service for Residential Accounts
  Subscribers are guaranteed 64 kilobit per second downstream bandwidth during business hours and 128 kilobit per second at other times, with a maximum burst rate of 1 megabit per second.

- “On Demand” Service by Special Order
  This class of "on demand" service allows a subscriber to request additional bandwidth available for a specific period of time. For example, a subscriber can go to operator’s web site and requests for increased guaranteed bandwidth service levels from his registered subscribed class of service from the normal 256 kilobit per second to 1 megabit per second from 2 p.m. to 4 p.m. the following day only, after which his service levels returns to the original subscribed class. The provisioning server will check the bandwidth commitment and utilization history to decide whether such "on demand" service is granted.

I.3.3 Usage-Based Billing

A complete billing solution involves the following processes:

- Design different usage-based billing options
- Capture and manage subscriber account and service subscription information
- Estimate future usage based on past history
- Collect billable event data
- Generate and rate billing records
- Calculate, prepare and deliver bill
- Process and manage bill payment information and records
- Handle customer account inquires
- Manage debt and fraud

This Specification focuses only on various business scenarios on bandwidth-centric usage-based billing options.

I.3.4 Designing Usage-Based Billing Models

In support of the offering of different classes of service is a new set of billing processes, which are based on the accounting of actual usage of subscribed service by each subscriber calculated by the associated fee structures.

There are several alternatives to implementing usage-based billing. The following offers a few examples:

- Billing Based on an Average Bandwidth Usage.
  The average bandwidth usage is defined as the total bytes transmitted divided by the billing period.
- Billing Based on Peak Bandwidth Usage.
  The peak bandwidth usage is the highest bandwidth usage sample during the entire billing period. Each usage sample is defined as the average bandwidth usage over a data collection period (typically 10 minutes).

  Since it is usually the peak usage pattern that creates the highest possibility of access problems for the cable operator, therefore it is reasonable to charge for such usage. One scheme of peak usage billing is called "95 percentile billing". The process is as follows -- at the end of each billing period, the billing software examines the usage records of each subscriber and it "throws away" the top five percent of usage records of that period, then charge the subscriber on the next highest bandwidth usage.
- "Flat Monthly Fee"Plus Usage Billing Based on the Class of Service Subscribed.
  Any usage beyond the minimum guaranteed bandwidth for that particular subscriber service class is subject to an extra charge based on the number of bytes transmitted.
- Billing for "On Demand" Service
  This special billing process is to support the "On Demand" Service offering described above.

This page intentionally left blank.

Appendix II Summary of CM Authentication and Code File Authentication (informative)

The purpose of this appendix is to provide the overview of the two authentication mechanisms defined by the BPI+ specification as well as to provide an example of the responsibility assignment for actual operation but not to add any new requirements for the CMTS or the CM. Please refer to the BPI+ specification regarding the requirement for the CMTS and the CM.

II.1 Authentication of the DOCSIS 2.0-compliant CM

If the CMTS is DOCSIS 2.0/BPI+-compliant and a DOCSIS 2.0 CM is provisioned to run BPI+ by the configuration file, the CMTS authenticates the CM by verifying the CM certificate and the manufacturer certificate. These certificates are contained in the Auth request and Auth Info packets respectively, and are sent to the CMTS by the CM after registration (when provisioned to do so by the configuration file). Only CMs with valid certificates will be authorized by the CMTS. Note that this CM authentication will not be applied if the CMTS and/or the CM is not compliant with BPI+, or the CM is not provisioned to run BPI+.

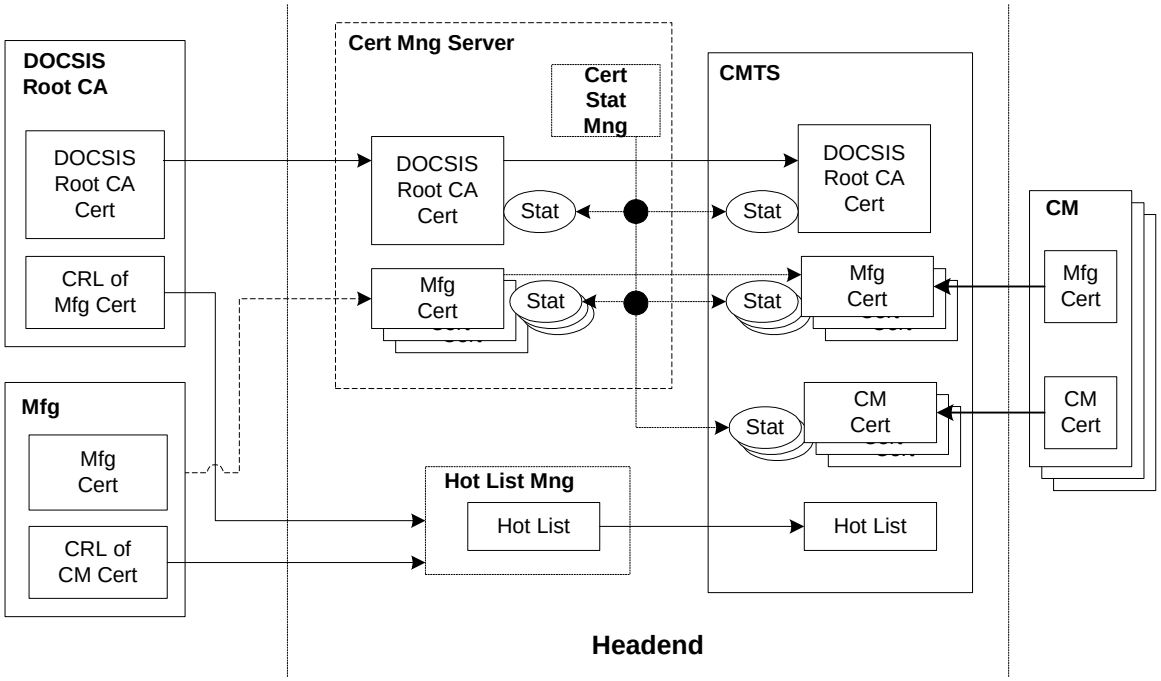


Figure II-1 Authentication of the DOCSIS 2.0-compliant CM

II.1.1 Responsibility of the DOCSIS Root CA

The DOCSIS Root CA is responsible for the following:

- Storing the DOCSIS Root private key in secret
- Maintaining the DOCSIS Root CA certificate
- Issuing the manufacturer CA certificates signed by the DOCSIS Root CA
- Maintaining the CRL of the manufacturer CA
- Providing the operators with the CRL

The DOCSIS Root CA or CableLabs is likely to put the DOCSIS Root CA on their Web or TFTP server in order to let the operators (or the CMTS on behalf of the operator) download it, but this is not yet decided.

II.1.2 Responsibility of the CM manufacturers

The CM manufacturers are responsible for the following:

- Storing the manufacturer CA private key in secret
- Maintaining the manufacturer CA certificate. The manufacturer CA certificate is usually signed by the DOCSIS Root CA but can be self-signed until the DOCSIS Root CA issues it based on the CableLabs policy.
- Issuing the CM certificates
- Putting the manufacturer CA certificate in the CM's software
- Putting each CM certificate in the CM's permanent, write-once memory
- Providing the operators with the hot list of the CM certificates. The hot list may be in CRL format. However, the detail of the format and the way of delivery are TBD.

II.1.3 Responsibility of the operators

The operators are responsible for the following:

- Maintaining that the CMTSes have an accurate date and time. If a CMTS has a wrong date or time, the invalid certificate may be authenticated or the valid certificate may not be authenticated.
- Putting the DOCSIS Root CA certificate in the CMTS during the CMTS provisioning using the BPI+ MIB or the CMTS's proprietary function. The operator may have a server to manage this certificate for one or more CMTS(s).
- Putting the manufacturer CA certificate(s) in the CMTS during the CMTS provisioning using the BPI+ MIB or the CMTS's proprietary function (optional). The operator may have a server to manage this certificate for one or more CMTSes.
- Maintaining the status of the certificates in the CMTSes if desired using the BPI+ MIB or the CMTS's proprietary function (optional). The operator may have a server to manage all the status of the certificates recorded in one or more CMTSes.

  The operator may have a server to manage the DOCSIS Root CA certificate, manufacturer CA certificate(s) and also the status of the certificates recorded in one or more CMTSes.
- Maintaining the hot list for the CMTS based on the CRLs provided by the DOCSIS Root CA and the CM manufacturers (optional). The operator may have a server to manage the hot list based on the CRLs provided by the DOCSIS Root CA and manufacturer CAs. The CMTS may have a function to automatically download the DOCSIS Root CA certificate and the CRLs via the Internet or other method. The DOCSIS Root CA or CableLabs is likely to put the DOCSIS Root CA on their Web or TFTP server in order to let the operators (or the CMTS on behalf of the operator) download it but this is not yet decided.

II.2 Authentication of the code file for the DOCSIS 2.0-compliant CM

When a DOCSIS 2.0/BPI+-compliant CM downloads a code file from a TFTP server, the CM must authenticate the code file as defined in Appendix D of the BPI+ specification regardless of whether the CM was provisioned to use BPI+, BPI, or neither, by the configuration file. The CM installs the new image and restarts using it only if verification of the code image was successful (as defined in Appendix D of the BPI+ specification). If authentication fails, the CM rejects the code file downloaded from the TFTP server and continues to operate using the current code. The CM performs a software download, whether initiated by the configuration file or SNMP, only if it was initialized with a valid CVC received in the CM configuration file. In addition to the code

file authentication by the CM, the operators may authenticate the code file before they put it on the TFTP sever. The following figure shows the summary of these mechanisms.

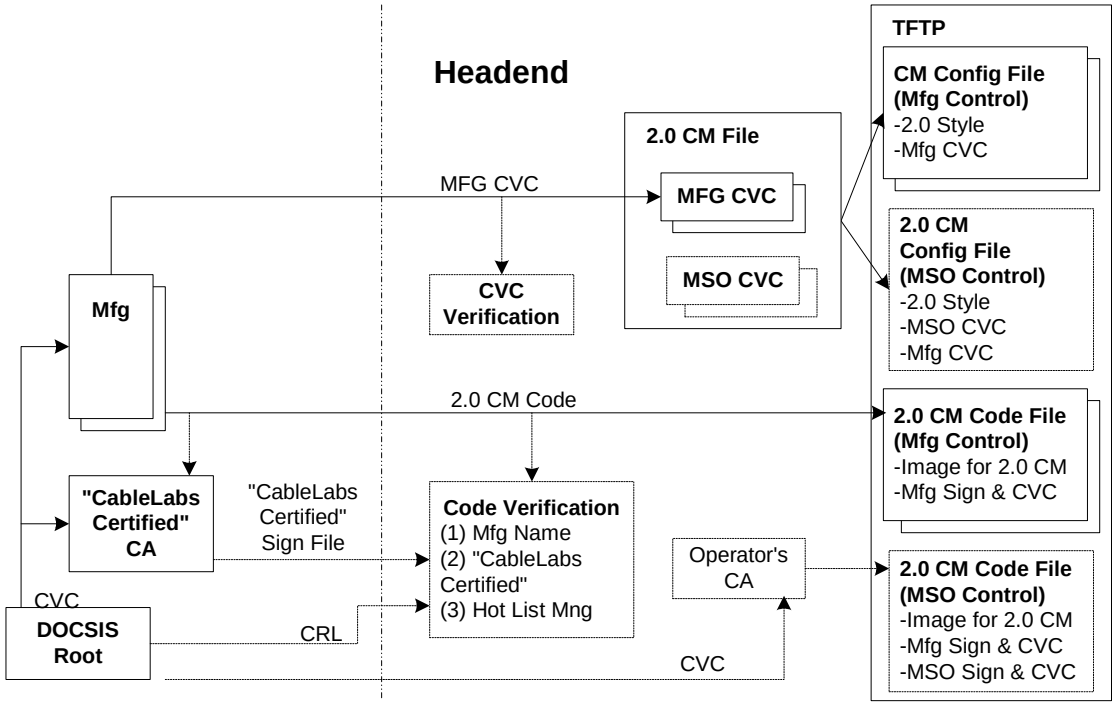


Figure II-2 Authentication of the code file for the DOCSIS 2.0-compliant CM

II.2.1 Responsibility of the DOCSIS Root CA

The DOCSIS Root CA is responsible for the following:

- Storing the DOCSIS Root private key in secret
- Maintaining the DOCSIS Root CA certificate
- Issuing the code verification certificates (CVCs) for the CM manufacturers, for the operators, and for "CableLabs Certified(TM)"
- The Root CA may maintain the CRL of the CVCs and provide it to the operators.

II.2.2 Responsibility of the CM manufacturer

The CM manufacturers are responsible for the following:

- Storing the manufacturer CVC private key in secret
- Putting the DOCSIS Root CA certificate in the CM's software
- Maintaining the manufacturer CVC (the current BPI+ specification only allows CVCs signed by the DOCSIS Root CA and does not accept self-signed CVCs)
- Generating the code file with the manufacturer's CVC and signature
- Providing the operators with the code file and the manufacturer CVC

II.2.3 Responsibility of CableLabs

CableLabs is responsible for the following:

- Storing the "CableLabs Certified(TM)" CVC private key in secret
- Maintaining the "CableLabs Certified(TM)" CVC signed by the DOCSIS Root CA
- Issuing the "CableLabs Certified(TM)" signature file for the DOCSIS 2.0 CM code file certified by CableLabs

II.2.4 Responsibility of the operators

Operators have the following responsibilities and options:

- Verifying the manufacturer CVC and signature in the code file provided by the manufacturer prior to using it (optional). The code file may be rejected (not used to upgrade CMs) if the manufacturer signature or CVC is invalid.
- Checking if the code file provided by the CM manufacturer is "CableLabs Certified(TM)" by verifying the "CableLabs Certified(TM)" CVC and signature in the "CableLabs Certified(TM)" signature file against the code file before the operator loads the code file on the TFTP server (optional).
- Maintaining the operator CA by storing the operator CA private key in secret and maintaining the operator's (co-signer) CVC issued by the DOCSIS Root CA (optional)
- Generating the MSO-controlled code file by adding the operator's CVC and signature to the original code file provided by the CM manufacturer (optional)
- Checking if the CVC provided by the CM manufacturer is valid (optional)
- Putting the appropriate CVC(s) in the CM configuration file. In the case that the original code file is to be downloaded to the CMs, the CM configuration file must contain the valid CVC from the CM's manufacturer. In case that the operator-controlled code file is to be downloaded, the CM configuration file must contain the valid CVC of the operator and may contain the valid CVC from the CM manufacturer. If a CVC is not present the CM configuration file, or the CVCs that are present are invalid, the CM will not initiate a software download if instructed to via SNMP or the CM configuration file. Note that the DOCSIS 2.0-compliant CM may be registered and authorized by the CMTS and become operational regardless of whether the CM configuration file contains valid CVCs.

Appendix III Acknowledgments (informative)

On behalf of CableLabs I would like to thank the following key contributors to DOCSIS 2.0 for their outstanding and superb contributions to this valuable program…

Victor Hou of Juniper Networks (formerly Pacific Broadband) and Yoav Hebron of Conexant led the Physical Layer working groups that rewrote RFI Section 6, and wrote RFI Appendix VII. Ariel Yagil of Texas Instruments, Mike Grimwood of Imedia, Bruce Currivan and Tom Kolze of Broadcom, Hikmet Sari and David Munro of Juniper Networks, David Hull and Shimon Tzukerman of Conexant, Elias Nemer and Hassan Yaghoobi of Intel, and Jack Moran of Motorola participated in those groups.

John Chapman and Dan Crocker of Cisco led the DMPI working group that developed the specification for the CMTS MAC/PHY interface, which became RFI Annex H.

Rich Prodan of Terayon led the OSS working group that developed the new MIB for DOCSIS 2.0 as well as reworking the OSSI specification. Aviv Goren of Terayon, David Raftus of Imedia, Greg Nakanishi of Motorola, Adi Shaliv of Intel, Rich Woundy of Cisco, and Jason Schnitzer of Stargus contributed to that group.

Rusty Cashman of Correlant led the MAC layer working group that reworked much of RFI Sections 8, 9, and 11. Jeff Hoffman of Intel; Lisa Denney of Broadcom; Alon Bernstein of Cisco; Gordon Li of Conexant; Asaf Matatyaou of Terayon; Robert Fanfelle of Imedia; David Doan, Christiaan Prins, Leo Zimmerman and Simon Brand of Philips contributed to that working group.

Clive Holborow of Motorola led the System Capabilities working group which rewrote RFI Section 3 and Annex G, and contributed to RFI Sections 6, 9, and 11. Daniel Howard of Broadcom, Noam Geri of TI, and Doug Jones of YAS contributed to that group.

Clive Holborow of Motorola, Victor Hou of Juniper Networks, Mike Grimwood of Imedia, Bruce Currivan and Daniel Howard of Broadcom, Rich Prodan of Terayon, and Hal Roberts of ADC wrote the informational material in RFI Appendix VIII.

David Hull of Conexant, Luc Martens and Wim De Ketelaere of tComLabs, the engineers at UPC, and the Euro-DOCSIS Certification Board for their contributions to RFI Annex F.

The engineers at Terayon, Imedia, Broadcom, Texas Instruments, and Conexant, as well as the members of the IEEE 802.14a Hi-PHY working group (chaired by Roger Durant of Cabletron (now Riverstone)) developed the technology proposals that became DOCSIS 2.0.

George Hart of Rogers Cable, Oleh Sniezko of AT&T Broadband, Dan Rice of Stargus for their guidance and contributions on behalf of CableLabs member companies.

I would also like to recognize Greg White, Mukta Kar, John Eng, Doug Jones, Eduardo Cardona, Dorothy Raymond, Alex Ball, and Cynthia Metsker from CableLabs for their leadership and first class work.

CableLabs and the cable industry as a whole are grateful to these individuals and organizations for their outstanding, first class contributions.

Rouzbeh Yassini
CEO of YAS ventures, LLC
Exec Consultant to CableLabs

This page intentionally left blank.

Appendix IV Revisions (informative)

IV.1 ECNs included in SP-OSSIv2.0-I02-020617

Table IV-1 Incorporated ECN Table

ECN	Date Accepted	Author	Summary
ossi2-n-02016	02/27/02	David Raftus	Replacement of RF MIB, associated changes.
oss2-n-02024	03/06/02	Efrat Zeharhary, Miron Tzhori	Version 2.0-specific items.
oss2-n-02053	04/10/02	André Lejeune	Delete section 7.3.3.1.
oss2-n-02054	04/03/02	Kaz Ozawa	Replace section 7.3.4.1.
oss2-n-02062	04/10/02	Jon Ulvr	Change section 7.4.2.1 and Annex F.
oss2-n-02069	05/01/02	Steve Cotton	Update to latest IPDR.org specifications.
oss2-n-02079	05/22/02	Lior Levy	Clarify ‘Storage type’ functionality within CMs.
oss2-n-02088	05/29/02	Anik Lacerte	Simplification of the specification.
oss2-n-02107	05/22/02	David Raftus	Update RFI MIB.

This page intentionally left blank.