Network Working Group
Request for Comments: 2669
Category: Proposed Standard

M. St. Johns, Ed.
@Home Network

August 1999

DOCSIS Cable Device MIB
Cable Device Management Information Base for DOCSIS compliant Cable Modems and Cable Modem Termination Systems

Status of this Memo

This document specifies an Internet standards track protocol for the Internet community, and requests discussion and suggestions for improvements. Please refer to the current edition of the "Internet Official Protocol Standards" (STD 1) for the standardization state and status of this protocol. Distribution of this memo is unlimited.

Copyright Notice

Abstract

This memo defines a portion of the Management Information Base (MIB) for use with network management protocols in the Internet community. In particular, it defines a basic set of managed objects for SNMP-based management of DOCSIS 1.0 compliant Cable Modems and Cable Modem Termination Systems.

This memo specifies a MIB module in a manner that is compliant to the SNMP SMIv2 [5][6][7]. The set of objects is consistent with the SNMP framework and existing SNMP standards.

This memo is a product of the IPCDN working group within the Internet Engineering Task Force. Comments are solicited and should be addressed to the working group's mailing list at ipcdn@terayon.com and/or the author.

Table of Contents

1. The SNMP Management Framework

The SNMP Management Framework presently consists of five major components:

- An overall architecture, described in RFC 2571 [1].

- Mechanisms for describing and naming objects and events for the purpose of management. The first version of this Structure of Management Information (SMI) is called SMIv1 and described in STD 16, RFC 1155 [2], STD 16, RFC 1212 [3] and RFC 1215 [4]. The second version, called SMIv2, is described in STD 58, RFC 2578 [5], STD 58, RFC 2579 [6] and STD 58, RFC 2580 [7].

- Message protocols for transferring management information. The first version of the SNMP message protocol is called SNMPv1 and described in STD 15, RFC 1157 [8]. A second version of the SNMP message protocol, which is not an Internet standards track protocol, is called SNMPv2c and described in RFC 1901 [9] and RFC

1906 [10]. The third version of the message protocol is called SNMPv3 and described in RFC 1906 [10], RFC 2572 [11] and RFC 2574 [12].

- o Protocol operations for accessing management information. The first set of protocol operations and associated PDU formats is described in STD 15, RFC 1157 [8]. A second set of protocol operations and associated PDU formats is described in RFC 1905 [13].

- o A set of fundamental applications described in RFC 2573 [14] and the view-based access control mechanism described in RFC 2575 [15].

Managed objects are accessed via a virtual information store, termed the Management Information Base or MIB. Objects in the MIB are defined using the mechanisms defined in the SMI.

This memo specifies a MIB module that is compliant to the SMIv2. A MIB conforming to the SMIv1 can be produced through the appropriate translations. The resulting translated MIB must be semantically equivalent, except where objects or events are omitted because no translation is possible (use of Counter64). Some machine readable information in SMIv2 will be converted into textual descriptions in SMIv1 during the translation process. However, this loss of machine readable information is not considered to change the semantics of the MIB.

2. Glossary

The terms in this document are derived either from normal cable system usage, or from the documents associated with the Data Over Cable Service Interface Specification process.

2.1. CATV

Originally "Community Antenna Television", now used to refer to any cable or hybrid fiber and cable system used to deliver video signals to a community.

2.2. CM Cable Modem.

A CM acts as a "slave" station in a DOCSIS compliant cable data system.

2.3. CMTS Cable Modem Termination System.

A generic term covering a cable bridge or cable router in a head-end. A CMTS acts as the master station in a DOCSIS compliant cable data system. It is the only station that transmits downstream, and it controls the scheduling of upstream transmissions by its associated CMs.

2.4. DOCSIS

"Data Over Cable Interface Specification". A term referring to the ITU-T J.112 Annex B standard for cable modem systems [20].

2.5. Downstream

The direction from the head-end towards the subscriber.

2.6. Head-end

The origination point in most cable systems of the subscriber video signals. Generally also the location of the CMTS equipment.

2.7. MAC Packet

A DOCSIS PDU.

2.8. MCNS

"Multimedia Cable Network System". Generally replaced in usage by DOCSIS.

2.9. RF

Radio Frequency.

2.10. Upstream

The direction from the subscriber towards the head-end.

3. Overview

This MIB provides a set of objects required for the management of DOCSIS compliant Cable Modems (CM) and Cable Modem Termination Systems (CMTS). The specification is derived from the DOCSIS Radio Frequency Interface specification [16]. Please note that the DOCSIS 1.0 standard only requires Cable Modems to implement SNMPv1 and to

process IPv4 customer traffic. Design choices in this MIB reflect those requirements. Future versions of the DOCSIS standard are expected to require support for SNMPv3 and IPv6 as well.

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [19].

3.1. Structure of the MIB

This MIB is structured into seven groups:

- The docsDevBase group extends the MIB-II 'system' group with objects needed for cable device system management.
- The docsDevNmAccessGroup provides a minimum level of SNMP access security (see Section 3 of [18]).
- The docsDevSoftware group provides information for network-downloadable software upgrades. See "Handling of Software Upgrades" below..
- The docsDevServer group provides information about the progress of the interaction between the CM or CMTS and various provisioning servers.
- The docsDevEvent group provides control and logging for event reporting.
- The docsDevFilter group configures filters at link layer and IP layer for bridged data traffic. This group consists of a link-layer filter table, docsDevFilterLLCTable, which is used to manage the processing and forwarding of non-IP traffic; an IP packet classifier table, docsDevFilterIpTable, which is used to map classes of packets to specific policy actions; a policy table, docsDevFilterPolicyTable, which maps zero or more policy actions onto a specific packet classification, and one or more policy action tables.

  At this time, this MIB specifies only one policy action table, docsDevFilterTosTable, which allows the manipulation of the type of services bits in an IP packet based on matching some criteria. The working group may add additional policy types and action tables in the future, for example to allow QOS to modem service identifier assignment based on destination.

- The docsDevCpe group provides control over which IP addresses may be used by customer premises equipment (e.g. PCs) serviced by a given cable modem. This provides anti-spoofing control at the point of origin for a large cable modem system. This group is separate from docsDevFilter primarily as this group is only implemented on the Cable Modem (CM) and MUST NOT be implemented on the Cable Modem Termination System (CMTS).

3.2. Management requirements

3.2.1. Handling of Software upgrades

The Cable Modem software upgrade process is documented in [16]. From a network management station, the operator:

- sets docsDevSwServer to the address of the TFTP server for software upgrades
- sets docsDevSwFilename to the file pathname of the software upgrade image
- sets docsDevSwAdminStatus to upgrade-from-mgt

One reason for the SNMP-initiated upgrade is to allow loading of a temporary software image (e.g., special diagnostic software) that differs from the software normally used on that device without changing the provisioning database.

Note that software upgrades should not be accepted blindly by the cable device. The cable device may refuse an upgrade if:

- The download is incomplete.
- The file contents are incomplete or damaged.
- The software is not intended for that hardware device (may include the case of a feature set that has not been purchased for this device).

3.2.2. Events and Traps

This MIB provides control facilities for reporting events through syslog, traps, and non-volatile logging. If events are reported through traps, the specified conventions must be followed. Other means of event reporting are outside the scope of this document.

The definition and coding of events is vendor-specific. In deference to the network operator who must troubleshoot multi-vendor networks, the circumstances and meaning of each event should be reported as human-readable text. Vendors SHOULD provide time-of-day clocks in CMs to provide useful timestamping of events.

For each vendor-specific event that is reportable via TRAP, the vendor must create an enterprise-specific trap definition. Trap definitions MUST include the event reason encoded as DisplayString and should be defined as:

```
trapName NOTIFICATION-TYPE
        OBJECTS {
            ifIndex,
            eventReason,
            other useful objects
        }
        STATUS      current
        DESCRIPTION
            "trap description"
        ::= Object Id
```

Note that ifIndex is only included if the event or trap is interface related.

An example (fake) vendor defined trap might be:

```
xyzVendorModemDropout NOTIFICATION-TYPE
        OBJECTS {
            eventReason,
            xyzModemHighWatermarkCount
        }
        STATUS  current
        DESCRIPTION
           "Sent by a CMTS when a configurable number of modems
        (xyzModemHysteresis) de-register or become unreachable during
        the sampling period (5 minutes).  Used to warn a management
        station about a catastrophic cable plant outage."
        ::= { xyzTraps 23 }
```

In this example eventReason is a DisplayString providing a human readable error message, and xyzModemHighWatermarkCount is a Gauge32 which indicates the maximum number of modems during the epoch.

The last digit of the trap OID for enterprise-specific traps must match docsDevEvId. For SNMPv1-capable Network Management systems, this is necessary to correlate the event type to the trap type. Many Network Management systems are only capable of trap filtering on an enterprise and single-last-digit basis.

3.2.3. Trap Throttling

The CM and CMTS MUST provide support for trap message throttling as described below. The network operator can employ message rate throttling or trap limiting by manipulating the appropriate MIB variables.

3.2.3.1. Trap rate throttling

Network operators may employ either of two rate control methods. In the first method, the device ceases to send traps when the rate exceeds the specified maximum message rate. It resumes sending traps only if reactivated by a network management station request.

In the second method, the device resumes sending traps when the rate falls below the specified maximum message rate.

The network operator configures the specified maximum message rate by setting the measurement interval (in seconds), and the maximum number of traps to be transmitted within the measurement interval. The operator can query the operational throttling state (to determine whether traps are enabled or blocked by throttling) of the device, as well as query and set the administrative throttling state (to manage the rate control method) of the device.

3.2.3.2. Limiting the trap rate

Network operators may wish to limit the number of traps sent by a device over a specified time period. The device ceases to send traps when the number of traps exceeds the specified threshold. It resumes sending traps only when the measurement interval has passed.

The network operator defines the maximum number of traps he is willing to handle and sets the measurement interval to a large number (in hundredths of a second). For this case, the administrative throttling state is set to stop at threshold which is the maximum number of traps.

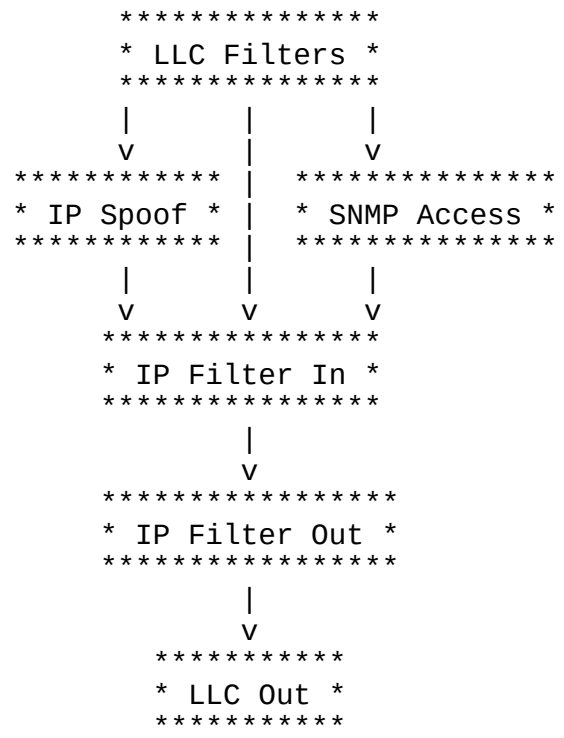
See "Techniques for Managing Asynchronously Generated Alerts" [17] for further information.

3.3. Protocol Filters

The Cable Device MIB provides objects for both LLC 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.

In general, a cable modem applies filters (or more properly, classifiers) in an order appropriate to the 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. Since the cable modem does not generally do any IP processing (other than that specified by the filters) the processing of the IP in filters and IP out filters can usually be combined into a single step.



3.3.1. Inbound LLC Filters - docsDevFilterLLCTable

The inbound LLC (or MAC or level-2) filters are contained in the docsDevFilterLLCTable and are applied to level-2 frames entering the cable modem from either the RF MAC interface or from one of the CPE (ethernet or other) interfaces. These filters are used to prohibit the processing and forwarding of certain types of level-2 traffic that may be disruptive to the network. The filters, as currently specified, can be set to cause the modem to either drop frames which match at least one filter, or to process a frame which matches at least filter. Some examples of possible configurations would be to only permit IP (and ARP) traffic, or to drop NETBUEI traffic.

3.3.2. Special Filters

Special filters are applied after the packet is accepted from the MAC layer by the IP module, but before any other processing is done. They are filters that apply only to a very specific class of traffic.

3.3.2.1. IP Spoofing Filters - docsDevCpeTable

IP spoofing filters are applied to packets entering the modem from one of the CPE interfaces and are intended to prevent a subscriber from stealing or mis-using IP addresses that were not assigned to the subscriber. If the filters are active (enabled), the source address of the IP packet must match at least one IP address in this table or it is discarded without further processing.

The table can be automatically populated where the first N different IP addresses seen from the CPE side of the cable modem are used to automatically populate the table. The spoofing filters are specified in the docsDevCpeTable and the policy for automatically creating filters in that table is controlled by docsDevCpeEnroll and docsDevCpeMax as well as the network management agent.

3.3.2.2. SNMP Access Filters - docsDevNmAccessTable

The SNMP access filters are applied to SNMP packets entering from any interface and destined for the cable modem. If the packets enter from a CPE interface, the SNMP filters are applied after the IP spoofing filters. The filters only apply to SNMPv1 or SNMPv2c traffic, and are not consulted for SNMPv3 traffic (and need not be implemented by a v3 only agent). SNMPv3 access control is specified in the User Security Model MIB in [12].

3.3.3. IP Filtering - docsDevIpFilterTable

The IP Filtering table acts as a classifier table. Each row in the table describes a template against which IP packets are compared. The template includes source and destination addresses (and their associated masks), upper level protocol (e.g. TCP, UDP), source and destination port ranges, TOS and TOS mask. A row also contains interface and traffic direction match values which have to be considered in combination. All columns of a particular row must match the appropriate fields in the packet, and must match the interface and direction items for the packet to result in a match to the packet.

When classifying a packet, the table is scanned beginning with the lowest number filter. If the agent finds a match, it applies the group of policies specified. If the matched filter has the continue bit set, the agent continues the scan possibly matching additional filters and applying additional policies. This allows the agent to take one set of actions for the 24.0.16/255.255.255.0 group and one set of actions for telnet packets to/from 24.0.16.30 and these sets of actions may not be mutually exclusive.

Once a packet is matched, one of three actions happen based on the setting of docsDevFilterIpControl in the row. The packet may be dropped, in which case no further processing is required. The packet may be accepted and processing of the packet continues. Lastly, the packet may have a set of policy actions applied to it. If docsDevFilterIpContinue is set to true, scanning of the table continues and additional matches may result.

When a packet matches, and docsDevFilterIpControl in the filter matched is set to 'policy', the value of docsDevFilterIpPolicyId is used as a selector into the docsDevFilterPolicyTable. The first level of indirection may result in zero or more actions being taken based on the match. The docsDevFilterPolicyTable is scanned in row order and all rows where docsDevFilterPolicyId equals docsDevFilterIpPolicyId have the action specified by docsDevFilterPolicyValue 'executed'. For example, a value pointing to an entry in the docsDevFilterTosTable may result in the re-writing of the TOS bits in the IP packet which was matched. Another possibility may be to assign an output packet to a specific output upstream queue. An even more complex action might be to re-write the TOS value, assign the packet to an upstream service ID, and drop it into a particular IPSEC tunnel.

Example:

docsDevFilterIpTable

```
#  Index,  SrcIP/Mask, DstIP/Mask,ULP, SrcPts,DstPts,Tos/Mask,
# Int/Dir,  Pgroup, [continue]
# drop any netbios traffic
10, 0/0, 0/0, TCP, any, 137-139, 0/0, any/any, drop

# traffic to the proxy gets better service - other matches possible
20, 0/0, proxy/32, TCP, any,any, 0/0, cpe/in, 10, continue

# Traffic from CPE 1 gets 'Gold' service, other matches possible
30, cpe1/32, 0/0, any, any,any, 0/0, cpe/in, 20, continue

# Traffic from CPE2 to work goes, other traffic dropped
40, cpe2/32, workIPs/24, any, 0/0, cpe/in, accept
45, cpe2/32, 0/0, any, any,ayn, 0/0, cpe/in, drop

# Traffic with TOS=4 gets queued on the "silver" queue.
50, 0/0, 0/0, any, any,any, 4/255, cpe/in, 30

# Inbound "server" traffic to low numbered ports gets dropped.
60, 0/0, 0/0, TCP, any,1-1023, 0/0, cpe/out, drop
65, 0/0, 0/0, UDP, any,1-1023, 0/0, cpe/out, drop
```

docsDevFilterIpPolicyTable

```
#
# index, policy group, policy
10, 10, queueEntry.20   -- special queue for traffic to proxy

15, 20, queueEntry.15   -- Gold Service queue
20, 20, docsDevFilterTosStatus.10 -- Mark this packet with TOS 5

25, 30, queueEntry.10   -- Silver service queue
```

This table describes some special processing for packets originating from either the first or second CPE device which results in their queuing on to special upstream traffic queues and for the "gold" service results in having the packets marked with a TOS of 5. The 10, 20, 60 and 65 entries are generic entries that would generally be applied to all traffic to this CM. The 30, 40 and 45 entries are specific to a particular CPE's service assignments. The ordering here is a bit contrived, but is close to what may actually be required by the operator to control various classes of customers.

3.3.4. Outbound LLC Filters

Lastly, any outbound LLC filters are applied to the packet just prior to it being emitted on the appropriate interface. This MIB does not specify any outbound LLC filters, but it is anticipated that the QOS additions to the DOCSIS standard may include some outbound LLC filtering requirements. If so, those filters would be applied as described here.

4. Definitions

```
DOCS-CABLE-DEVICE-MIB DEFINITIONS ::= BEGIN

IMPORTS
        MODULE-IDENTITY,
        OBJECT-TYPE,
-- do not import        BITS,
        IpAddress,
        Unsigned32,
        Counter32,
        Integer32,
        zeroDotZero,
        mib-2
                FROM SNMPv2-SMI
        RowStatus,
        RowPointer,
        DateAndTime,
        TruthValue
                FROM SNMPv2-TC
        OBJECT-GROUP,
        MODULE-COMPLIANCE
                FROM SNMPv2-CONF
        SnmpAdminString
                FROM SNMP-FRAMEWORK-MIB
        InterfaceIndexOrZero
                FROM IF-MIB;  -- RFC2233

docsDev MODULE-IDENTITY
        LAST-UPDATED    "9908190000Z" -- August 19, 1999
        ORGANIZATION    "IETF IPCDN Working Group"
        CONTACT-INFO
            "        Michael StJohns
             Postal: @Home Network
                     425 Broadway
                     Redwood City, CA 94063
                     U.S.A.
             Phone:  +1 650 569 5368
             E-mail: stjohns@corp.home.net"
```

```
        DESCRIPTION
            "This is the MIB Module for MCNS-compliant cable modems and
             cable-modem termination systems."
        REVISION "9908190000Z"
        DESCRIPTION
            "Initial Version, published as RFC 2669.
             Modified by Mike StJohns to add/revise filtering, TOS
             support, software version information objects."
        ::= { mib-2 69 }

docsDevMIBObjects  OBJECT IDENTIFIER ::= { docsDev 1 }
docsDevBase OBJECT IDENTIFIER ::= { docsDevMIBObjects 1 }

--
-- For the following object, there is no concept in the
-- RFI specification corresponding to a backup CMTS. The
-- enumeration is provided here in case someone is able
-- to define such a role or device.
--

docsDevRole OBJECT-TYPE
        SYNTAX INTEGER {
            cm(1),
            cmtsActive(2),
            cmtsBackup(3)
        }
        MAX-ACCESS  read-only
        STATUS      current
        DESCRIPTION
             "Defines the current role of this device.  cm (1) is
             a Cable Modem, cmtsActive(2) is a Cable Modem Termination
             System which is controlling the system of cable modems,
             and cmtsBackup(3) is a CMTS which is currently connected,
             but not controlling the system (not currently used).

             In general, if this device is a 'cm', its role will not
             change during operation or between reboots.  If the
             device is a 'cmts' it may change between cmtsActive and
             cmtsBackup and back again during normal operation.  NB:
             At this time, the DOCSIS standards do not support the
             concept of a backup CMTS, cmtsBackup is included for
             completeness."
        ::= { docsDevBase 1 }

docsDevDateTime OBJECT-TYPE
        SYNTAX      DateAndTime
        MAX-ACCESS  read-write
        STATUS      current
```

```
        DESCRIPTION
            "The date and time, with optional timezone
             information."
        ::= { docsDevBase 2 }

docsDevResetNow OBJECT-TYPE
        SYNTAX      TruthValue
        MAX-ACCESS  read-write
        STATUS      current
        DESCRIPTION
            "Setting this object to true(1) causes the device to reset.
             Reading this object always returns false(2)."
        ::= { docsDevBase 3 }

docsDevSerialNumber OBJECT-TYPE
        SYNTAX      SnmpAdminString
        MAX-ACCESS  read-only
        STATUS      current
        DESCRIPTION
            "The manufacturer's serial number for this device."
        ::= { docsDevBase 4 }

docsDevSTPControl OBJECT-TYPE
        SYNTAX INTEGER {
            stEnabled(1),
            noStFilterBpdu(2),
            noStPassBpdu(3)
        }
        MAX-ACCESS  read-write
        STATUS      current
        DESCRIPTION
            "This object controls operation of the spanning tree
             protocol (as distinguished from transparent bridging).
             If set to stEnabled(1) then the spanning tree protocol
             is enabled, subject to bridging constraints. If
             noStFilterBpdu(2), then spanning tree is not active,
             and Bridge PDUs received are discarded.
             If noStPassBpdu(3) then spanning tree is not active
             and Bridge PDUs are transparently forwarded. Note that
             a device need not implement all of these options,
             but that noStFilterBpdu(2) is required."
        ::= { docsDevBase 5 }

--
-- The following table provides one level of security for access
-- to the device by network management stations.
-- Note that access is also constrained by the
-- community strings and any vendor-specific security.
```

```
--

docsDevNmAccessTable OBJECT-TYPE
        SYNTAX      SEQUENCE OF DocsDevNmAccessEntry
        MAX-ACCESS  not-accessible
        STATUS      current
        DESCRIPTION
            "This table controls access to SNMP objects by network
             management stations. If the table is empty, access
             to SNMP objects is unrestricted.  This table exists only
             on SNMPv1 or v2c agents and does not exist on SNMPv3
             agents. See the conformance section for details.
             Specifically, for v3 agents, the appropriate MIBs and
             security models apply in lieu of this table."
        ::= { docsDevMIBObjects 2 }

docsDevNmAccessEntry OBJECT-TYPE
        SYNTAX      DocsDevNmAccessEntry
        MAX-ACCESS  not-accessible
        STATUS      current
        DESCRIPTION
            "An entry describing  access to SNMP objects by a
             particular network management station. An entry in
             this table is not readable unless the management station
             has read-write permission (either implicit if the table
             is empty, or explicit through an entry in this table.
             Entries are ordered by docsDevNmAccessIndex.  The first
             matching entry (e.g. matching IP address and community
             string) is used to derive access."
        INDEX { docsDevNmAccessIndex  }
        ::= {  docsDevNmAccessTable 1 }

DocsDevNmAccessEntry ::= SEQUENCE {
            docsDevNmAccessIndex         Integer32,
            docsDevNmAccessIp            IpAddress,
            docsDevNmAccessIpMask        IpAddress,
            docsDevNmAccessCommunity     OCTET STRING,
            docsDevNmAccessControl       INTEGER,
            docsDevNmAccessInterfaces    OCTET STRING,
            docsDevNmAccessStatus        RowStatus
        }

docsDevNmAccessIndex OBJECT-TYPE
        SYNTAX      Integer32 (1..2147483647)
        MAX-ACCESS  not-accessible
        STATUS      current
        DESCRIPTION
            "Index used to order the application of access
```

```
             entries."
        ::= { docsDevNmAccessEntry 1 }

docsDevNmAccessIp OBJECT-TYPE
        SYNTAX      IpAddress
        MAX-ACCESS  read-create
        STATUS      current
        DESCRIPTION
            "The IP address (or subnet) of the network management
             station. The address 255.255.255.255 is defined to mean
             any NMS. If traps are enabled for this entry, then the
             value must be the address of a specific device."
        DEFVAL { 'ffffffff'h }
        ::= { docsDevNmAccessEntry 2 }

docsDevNmAccessIpMask OBJECT-TYPE
        SYNTAX      IpAddress
        MAX-ACCESS  read-create
        STATUS      current
        DESCRIPTION
            "The IP subnet mask of the network management stations.
             If traps are enabled for this entry, then the value must
             be 255.255.255.255."
        DEFVAL { 'ffffffff'h }
        ::= { docsDevNmAccessEntry 3 }

docsDevNmAccessCommunity OBJECT-TYPE
        SYNTAX      OCTET STRING
        MAX-ACCESS  read-create
        STATUS      current
        DESCRIPTION
            "The community string to be matched for access by this
             entry. If set to a zero length string then any community
             string will match.  When read, this object SHOULD return
             a zero length string."
        DEFVAL { "public" }
        ::= { docsDevNmAccessEntry 4 }

docsDevNmAccessControl OBJECT-TYPE
        SYNTAX         INTEGER {
            none(1),
            read(2),
            readWrite(3),
            roWithTraps(4),
            rwWithTraps(5),
            trapsOnly(6)
        }
        MAX-ACCESS  read-create
```

```
        STATUS      current
        DESCRIPTION
            "Specifies the type of access allowed to this NMS. Setting
             this object to none(1) causes the table entry to be
             destroyed. Read(2) allows access by 'get' and 'get-next'
             PDUs. ReadWrite(3) allows access by 'set' as well.
             RoWithtraps(4), rwWithTraps(5), and trapsOnly(6)
             control distribution of Trap PDUs transmitted by this
             device."
        DEFVAL { read }
        ::= { docsDevNmAccessEntry 5 }

-- The syntax of the following object was copied from RFC1493,
-- dot1dStaticAllowedToGoTo.

docsDevNmAccessInterfaces OBJECT-TYPE
        SYNTAX      OCTET STRING
        MAX-ACCESS  read-create
        STATUS      current
        DESCRIPTION
            "Specifies the set of interfaces from which requests from
             this NMS will be accepted.
             Each octet within the value of this object specifies a set
             of eight interfaces, with the first octet specifying ports
             1 through 8, the second octet specifying interfaces 9
             through 16, etc.  Within each octet, the most significant
             bit represents the lowest numbered interface, and the least
             significant bit represents the highest numbered interface.
             Thus, each interface is represented by a single bit within
             the value of this object. If that bit has a value of '1'
             then that interface is included in the set.

             Note that entries in this table apply only to link-layer
             interfaces (e.g., Ethernet and CATV MAC). Upstream and
             downstream channel interfaces must not be specified."
--         DEFVAL is the bitmask corresponding to all interfaces
        ::= { docsDevNmAccessEntry 6 }

docsDevNmAccessStatus OBJECT-TYPE
        SYNTAX      RowStatus
        MAX-ACCESS  read-create
        STATUS      current
        DESCRIPTION
            "Controls and reflects the status of rows in this
             table. Rows in this table may be created by either the
             create-and-go or create-and-wait paradigms.  There is no
             restriction on changing values in a row of this table while
             the row is active."
```

```
        ::= { docsDevNmAccessEntry 7 }

--
--  Procedures for using the following group are described in section
--  3.2.1 of the DOCSIS Radio Frequence Interface Specification
--

docsDevSoftware OBJECT IDENTIFIER ::= { docsDevMIBObjects 3 }

docsDevSwServer OBJECT-TYPE
        SYNTAX      IpAddress
        MAX-ACCESS  read-write
        STATUS      current
        DESCRIPTION
            "The address of the TFTP server used for software upgrades.
             If the TFTP server is unknown, return 0.0.0.0."
        ::= { docsDevSoftware 1 }

docsDevSwFilename OBJECT-TYPE
        SYNTAX      SnmpAdminString (SIZE (0..64))
        MAX-ACCESS  read-write
        STATUS      current
        DESCRIPTION
            "The file name of the software image to be loaded into this
             device. Unless set via SNMP, this is the file name
             specified by the provisioning server that corresponds to
             the software version that is desired for this device.
             If unknown, the string '(unknown)' is returned."
        ::= { docsDevSoftware 2 }

docsDevSwAdminStatus OBJECT-TYPE
        SYNTAX INTEGER {
            upgradeFromMgt(1),
            allowProvisioningUpgrade(2),
            ignoreProvisioningUpgrade(3)
        }
        MAX-ACCESS  read-write
        STATUS      current
        DESCRIPTION
            "If set to upgradeFromMgt(1), the device will initiate a
             TFTP software image download using docsDevSwFilename.
             After successfully receiving an image, the device will
             set its state to ignoreProvisioningUpgrade(3) and reboot.
             If the download process is interrupted by a reset or
             power failure, the device will load the previous image
             and, after re-initialization, continue to attempt loading
             the image specified in docsDevSwFilename.
```

```
            If set to allowProvisioningUpgrade(2), the device will
            use the software version information supplied by the
            provisioning server when next rebooting (this does not
            cause a reboot).

            When set to ignoreProvisioningUpgrade(3), the device
            will disregard software image upgrade information from the
            provisioning server.

            Note that reading this object can return upgradeFromMgt(1).
            This indicates that a software download is currently in
            progress, and that the device will reboot after
            successfully receiving an image.

            At initial startup, this object has the default value of
            allowProvisioningUpgrade(2)."
        ::= { docsDevSoftware 3 }

docsDevSwOperStatus OBJECT-TYPE
        SYNTAX INTEGER {
            inProgress(1),
            completeFromProvisioning(2),
            completeFromMgt(3),
            failed(4),
            other(5)
        }
        MAX-ACCESS  read-only
        STATUS      current
        DESCRIPTION
            "InProgress(1) indicates that a TFTP download is underway,
             either as a result of a version mismatch at provisioning
             or as a result of a upgradeFromMgt request.
             CompleteFromProvisioning(2) indicates that the last
             software upgrade was a result of version mismatch at
             provisioning. CompleteFromMgt(3) indicates that the last
             software upgrade was a result of setting
             docsDevSwAdminStatus to upgradeFromMgt.
             Failed(4) indicates that the last attempted download
             failed, ordinarily due to TFTP timeout."
        REFERENCE
             "DOCSIS Radio Frequency Interface Specification, Section
             8.2, Downloading Cable Modem Operating Software."
        ::= { docsDevSoftware 4 }

docsDevSwCurrentVers OBJECT-TYPE
    SYNTAX      SnmpAdminString
    MAX-ACCESS  read-only
    STATUS      current
```

```
    DESCRIPTION
            "The software version currently operating in this device.
             This object should be in the syntax used by the individual
             vendor to identify software versions.  Any CM MUST return a
             string descriptive of the current software load.  For a
             CMTS, this object SHOULD contain either a human readable
             representation of the vendor specific designation of the
             software for the chassis, or of the software for the
             control processor. If neither of these is  applicable,
             this MUST contain an empty string."
    ::= { docsDevSoftware 5 }


--
-- The following group describes server access and parameters used for
-- initial provisioning and bootstrapping.
--

docsDevServer OBJECT IDENTIFIER ::= { docsDevMIBObjects 4 }

docsDevServerBootState OBJECT-TYPE
        SYNTAX INTEGER {
            operational(1),
            disabled(2),
            waitingForDhcpOffer(3),
            waitingForDhcpResponse(4),
            waitingForTimeServer(5),
            waitingForTftp(6),
            refusedByCmts(7),
            forwardingDenied(8),
            other(9),
            unknown(10)
        }
        MAX-ACCESS  read-only
        STATUS      current
        DESCRIPTION
            "If operational(1), the device has completed loading and
             processing of configuration parameters and the CMTS has
             completed the Registration exchange.
             If disabled(2) then the device was administratively
             disabled, possibly by being refused network access in the
             configuration file.
             If waitingForDhcpOffer(3) then a DHCP Discover has been
             transmitted and no offer has yet been received.
             If waitingForDhcpResponse(4) then a DHCP Request has been
             transmitted and no response has yet been received.
             If waitingForTimeServer(5) then a Time Request has been
             transmitted and no response has yet been received.
```

```
             If waitingForTftp(6) then a request to the TFTP parameter
             server has been made and no response received.
             If refusedByCmts(7) then the Registration Request/Response
             exchange with the CMTS failed.
             If forwardingDenied(8) then the registration process
             completed, but the network access option in the received
             configuration file prohibits forwarding. "
         REFERENCE
             "DOCSIS Radio Frequency Interface Specification, Figure
             7-1, CM Initialization Overview."
         ::= { docsDevServer 1 }

docsDevServerDhcp OBJECT-TYPE
         SYNTAX      IpAddress
         MAX-ACCESS  read-only
         STATUS      current
         DESCRIPTION
             "The IP address of the DHCP server that assigned an IP
              address to this device. Returns 0.0.0.0 if DHCP was not
              used for IP address assignment."
         ::= { docsDevServer 2 }

docsDevServerTime OBJECT-TYPE
         SYNTAX      IpAddress
         MAX-ACCESS  read-only
         STATUS      current
         DESCRIPTION
             "The IP address of the Time server (RFC-868). Returns
              0.0.0.0 if the time server IP address is unknown."
         ::= { docsDevServer 3 }

docsDevServerTftp OBJECT-TYPE
         SYNTAX      IpAddress
         MAX-ACCESS  read-only
         STATUS      current
         DESCRIPTION
             "The IP address of the TFTP server responsible for
              downloading provisioning and configuration parameters
              to this device. Returns 0.0.0.0 if the TFTP server
              address is unknown."
         ::= { docsDevServer 4 }

docsDevServerConfigFile OBJECT-TYPE
         SYNTAX      SnmpAdminString
         MAX-ACCESS  read-only
         STATUS      current
         DESCRIPTION
             "The name of the device configuration file read from the
```

```
             TFTP server. Returns an empty string if the configuration
             file name is unknown."
         ::= { docsDevServer 5 }

--
--  Event Reporting
--

docsDevEvent OBJECT IDENTIFIER ::= { docsDevMIBObjects 5 }

docsDevEvControl OBJECT-TYPE
         SYNTAX INTEGER {
             resetLog(1),
             useDefaultReporting(2)
         }
         MAX-ACCESS  read-write
         STATUS      current
         DESCRIPTION
             "Setting this object to resetLog(1) empties the event log.
              All data is deleted. Setting it to useDefaultReporting(2)
              returns all event priorities to their factory-default
              reporting. Reading this object always returns
              useDefaultReporting(2)."
         ::= { docsDevEvent 1 }

docsDevEvSyslog OBJECT-TYPE
         SYNTAX      IpAddress
         MAX-ACCESS  read-write
         STATUS      current
         DESCRIPTION
             "The IP address of the Syslog server. If 0.0.0.0, syslog
              transmission is inhibited."
         ::= { docsDevEvent 2 }

docsDevEvThrottleAdminStatus OBJECT-TYPE
         SYNTAX INTEGER {
             unconstrained(1),
             maintainBelowThreshold(2),
             stopAtThreshold(3),
             inhibited(4)
         }
         MAX-ACCESS  read-write
         STATUS      current
         DESCRIPTION
             "Controls the transmission of traps and syslog messages
              with respect to the trap pacing threshold.
              unconstrained(1) causes traps and syslog messages to be
              transmitted without regard to the threshold settings.
```

```
            maintainBelowThreshold(2) causes trap transmission and
            syslog messages to be suppressed if the number of traps
            would otherwise exceed the threshold.
            stopAtThreshold(3) causes trap transmission to cease
            at the threshold, and not resume until directed to do so.
            inhibited(4) causes all trap transmission and syslog
            messages to be suppressed.

            A single event is always treated as a single event for
            threshold counting. That is, an event causing both a trap
            and a syslog message is still treated as a single event.

            Writing to this object resets the thresholding state.

            At initial startup, this object has a default value of
            unconstrained(1)."
        ::= { docsDevEvent 3 }

docsDevEvThrottleInhibited OBJECT-TYPE
        SYNTAX      TruthValue
        MAX-ACCESS  read-only
        STATUS      current
        DESCRIPTION
            "If true(1), trap and syslog transmission is currently
             inhibited due to thresholds and/or the current setting of
             docsDevEvThrottleAdminStatus. In addition, this is set to
             true(1) if transmission is inhibited due to no
             syslog (docsDevEvSyslog) or trap (docsDevNmAccessEntry)
             destinations having been set."
        ::= { docsDevEvent 4 }

docsDevEvThrottleThreshold OBJECT-TYPE
        SYNTAX      Unsigned32
        MAX-ACCESS  read-write
        STATUS      current
        DESCRIPTION
            "Number of trap/syslog events per docsDevEvThrottleInterval
             to be transmitted before throttling.

             A single event is always treated as a single event for
             threshold counting. That is, an event causing both a trap
             and a syslog message is still treated as a single event.

             At initial startup, this object returns 0."
        ::= { docsDevEvent 5 }

docsDevEvThrottleInterval OBJECT-TYPE
        SYNTAX      Integer32 (1..2147483647)
```

```
        UNITS       "seconds"
        MAX-ACCESS  read-write
        STATUS      current
        DESCRIPTION
            "The interval over which the trap threshold applies.
             At initial startup, this object has a value of 1."

        ::= { docsDevEvent 6 }

--
-- The following table controls the reporting of the various classes of
-- events.
--

docsDevEvControlTable OBJECT-TYPE
        SYNTAX      SEQUENCE OF DocsDevEvControlEntry
        MAX-ACCESS  not-accessible
        STATUS      current
        DESCRIPTION
            "This table allows control of the reporting of event classes.
        For each event priority, a combination of logging and
        reporting mechanisms may be chosen. The mapping of event types
        to priorities is vendor-dependent. Vendors may also choose to
        allow the user to control that mapping through proprietary
        means."
        ::= {  docsDevEvent 7 }


docsDevEvControlEntry OBJECT-TYPE
        SYNTAX      DocsDevEvControlEntry
        MAX-ACCESS  not-accessible
        STATUS      current
        DESCRIPTION
            "Allows configuration of the reporting mechanisms for a
             particular event priority."
        INDEX { docsDevEvPriority }
        ::= { docsDevEvControlTable 1 }

DocsDevEvControlEntry ::= SEQUENCE {
            docsDevEvPriority        INTEGER,
            docsDevEvReporting       BITS
        }

docsDevEvPriority OBJECT-TYPE
        SYNTAX INTEGER {
            emergency(1),
            alert(2),
            critical(3),
```

```
            error(4),
            warning(5),
            notice(6),
            information(7),
            debug(8)
        }
        MAX-ACCESS  not-accessible
        STATUS      current
        DESCRIPTION
            "The priority level that is controlled by this
             entry. These are ordered from most (emergency) to least
             (debug) critical.  Each event with a CM or CMTS has a
             particular priority level associated with it (as defined
             by the vendor). During normal operation no event more
             critical than notice(6) should be generated. Events between
             warning and emergency should be generated at appropriate
             levels of problems (e.g. emergency when the box is about to
             crash)."
        ::= { docsDevEvControlEntry 1 }

docsDevEvReporting OBJECT-TYPE
        SYNTAX BITS {
            local(0),
            traps(1),
            syslog(2)
        }
        MAX-ACCESS  read-write
        STATUS      current
        DESCRIPTION
            "Defines the action to be taken on occurrence of this
             event class. Implementations may not necessarily support
             all options for all event classes, but at minimum must
             allow traps and syslogging to be disabled. If the
             local(0) bit is set, then log to the internal log, if the
             traps(1) bit is set, then generate a trap, if the
             syslog(2) bit is set, then send a syslog message
             (assuming the syslog address is set)."
        ::= { docsDevEvControlEntry 2 }

docsDevEventTable OBJECT-TYPE
        SYNTAX      SEQUENCE OF DocsDevEventEntry
        MAX-ACCESS  not-accessible
        STATUS      current
        DESCRIPTION
            "Contains a log of network and device events that may be
             of interest in fault isolation and troubleshooting."
        ::= {  docsDevEvent 8 }
```

```
docsDevEventEntry OBJECT-TYPE
        SYNTAX      DocsDevEventEntry
        MAX-ACCESS  not-accessible
        STATUS      current
        DESCRIPTION
            "Describes a network or device event that may be of
             interest in fault isolation and troubleshooting. Multiple
             sequential identical events are represented by
             incrementing docsDevEvCounts and setting
             docsDevEvLastTime to the current time rather than creating
             multiple rows.

             Entries are created with the first occurrance of an event.
             docsDevEvControl can be used to clear the table.
             Individual events can not be deleted."
        INDEX { docsDevEvIndex }

        ::= { docsDevEventTable 1 }

DocsDevEventEntry ::= SEQUENCE {
            docsDevEvIndex          Integer32,
            docsDevEvFirstTime      DateAndTime,
            docsDevEvLastTime       DateAndTime,
            docsDevEvCounts         Counter32,
            docsDevEvLevel          INTEGER,
            docsDevEvId             Unsigned32,
            docsDevEvText           SnmpAdminString
        }

docsDevEvIndex OBJECT-TYPE
        SYNTAX      Integer32 (1..2147483647)
        MAX-ACCESS  not-accessible
        STATUS      current
        DESCRIPTION
            "Provides relative ordering of the objects in the event
             log. This object will always increase except when
             (a) the log is reset via docsDevEvControl,
             (b) the device reboots and does not implement non-volatile
             storage for this log, or (c) it reaches the value 2^31.
             The next entry for all the above cases is 1."
        ::= { docsDevEventEntry 1 }

docsDevEvFirstTime OBJECT-TYPE
        SYNTAX      DateAndTime
        MAX-ACCESS  read-only
        STATUS      current
        DESCRIPTION
            "The time that this entry was created."
```

```
        ::= { docsDevEventEntry 2 }

docsDevEvLastTime OBJECT-TYPE
        SYNTAX      DateAndTime
        MAX-ACCESS  read-only
        STATUS      current
        DESCRIPTION
            "If multiple events are reported via the same entry, the
             time that the last event for this entry occurred,
             otherwise this should have the same value as
             docsDevEvFirstTime. "
        ::= { docsDevEventEntry 3 }

-- This object was renamed from docsDevEvCount to meet naming
-- requirements for Counter32
docsDevEvCounts OBJECT-TYPE
        SYNTAX      Counter32
        MAX-ACCESS  read-only
        STATUS      current
        DESCRIPTION
            "The number of consecutive event instances reported by
             this entry.  This starts at 1 with the creation of this
             row and increments by 1 for each subsequent duplicate
             event."
        ::= { docsDevEventEntry 4 }

docsDevEvLevel OBJECT-TYPE
        SYNTAX INTEGER {
            emergency(1),
            alert(2),
            critical(3),
            error(4),
            warning(5),
            notice(6),
            information(7),
            debug(8)
        }
        MAX-ACCESS  read-only
        STATUS      current
        DESCRIPTION
            "The priority level of this event as defined by the
             vendor.  These are ordered from most serious (emergency)
             to least serious (debug)."
        ::= { docsDevEventEntry 5 }

--
-- Vendors will provide their own enumerations for the following.
-- The interpretation of the enumeration is unambiguous for a
```

```
-- particular value of the vendor's enterprise number in sysObjectID.
--

docsDevEvId OBJECT-TYPE
        SYNTAX      Unsigned32
        MAX-ACCESS  read-only
        STATUS      current
        DESCRIPTION
            "For this product, uniquely identifies the type of event
             that is reported by this entry."
        ::= { docsDevEventEntry 6 }

docsDevEvText OBJECT-TYPE
        SYNTAX      SnmpAdminString
        MAX-ACCESS  read-only
        STATUS      current
        DESCRIPTION
            "Provides a human-readable description of the event,
             including all relevant context (interface numbers,
             etc.)."
        ::= { docsDevEventEntry 7 }

docsDevFilter OBJECT IDENTIFIER ::= { docsDevMIBObjects 6 }


--
-- Link Level Control Filtering
--

-- docsDevFilterLLCDefault renamed to docsDevFilterLLCUnmatchedAction

docsDevFilterLLCUnmatchedAction OBJECT-TYPE
        SYNTAX INTEGER {
            discard(1),
            accept(2)
        }
        MAX-ACCESS  read-write
        STATUS      current
        DESCRIPTION
            "LLC (Link Level Control) filters can be defined on an
             inclusive or exclusive basis: CMs can be configured to
             forward only packets matching a set of layer three
             protocols, or to drop packets matching a set of layer
             three protocols.  Typical use of these filters is to
             filter out possibly harmful (given the context of a large
             metropolitan LAN) protocols.

             If set to discard(1), any L2 packet which does not match at
```

```
             least one filter in the docsDevFilterLLCTable will be
             discarded. If set to accept(2), any L2 packet which does not
             match at least one filter in the docsDevFilterLLCTable
             will be accepted for further processing (e.g., bridging).
             At initial system startup, this object returns accept(2)."
         ::= { docsDevFilter 1 }

docsDevFilterLLCTable OBJECT-TYPE
         SYNTAX      SEQUENCE OF DocsDevFilterLLCEntry
         MAX-ACCESS  not-accessible
         STATUS      current
         DESCRIPTION
             "A list of filters to apply to (bridged) LLC
              traffic. The filters in this table are applied to
              incoming traffic on the appropriate interface(s)  prior
              to any further processing (e.g. before handing the packet
              off for level 3 processing, or for bridging).  The
              specific action taken when no filter is matched is
              controlled by docsDevFilterLLCUnmatchedAction."
         ::= { docsDevFilter 2 }

docsDevFilterLLCEntry OBJECT-TYPE
         SYNTAX      DocsDevFilterLLCEntry
         MAX-ACCESS  not-accessible
         STATUS      current
         DESCRIPTION
             "Describes a single filter to apply to (bridged) LLC traffic
              received on a specified interface. "
         INDEX { docsDevFilterLLCIndex }
         ::= { docsDevFilterLLCTable 1 }

DocsDevFilterLLCEntry ::= SEQUENCE {
             docsDevFilterLLCIndex               Integer32,
             docsDevFilterLLCStatus              RowStatus,
             docsDevFilterLLCIfIndex             InterfaceIndexOrZero,
             docsDevFilterLLCProtocolType        INTEGER,
             docsDevFilterLLCProtocol            Integer32,
             docsDevFilterLLCMatches             Counter32
         }

docsDevFilterLLCIndex OBJECT-TYPE
         SYNTAX      Integer32 (1..2147483647)
         MAX-ACCESS  not-accessible
         STATUS      current
         DESCRIPTION
             "Index used for the identification of filters (note that LLC
              filter order is irrelevant)."
         ::= { docsDevFilterLLCEntry 1 }
```

```
docsDevFilterLLCStatus OBJECT-TYPE
        SYNTAX      RowStatus
        MAX-ACCESS  read-create
        STATUS      current
        DESCRIPTION
            "Controls and reflects the status of rows in this
             table. There is no restriction on changing any of the
             associated columns for this row while this object is set
             to active."

        ::= { docsDevFilterLLCEntry 2}

docsDevFilterLLCIfIndex OBJECT-TYPE
        SYNTAX      InterfaceIndexOrZero
        MAX-ACCESS  read-create
        STATUS      current
        DESCRIPTION
            "The entry interface to which this filter applies.
             The value corresponds to ifIndex for either a CATV MAC
             or another network interface. If the value is zero, the
             filter applies to all interfaces. In Cable Modems, the
             default value is the customer side interface. In Cable
             Modem Termination Systems, this object has to be
             specified to create a row in this table."
        ::= { docsDevFilterLLCEntry 3 }

docsDevFilterLLCProtocolType OBJECT-TYPE
        SYNTAX INTEGER {
            ethertype(1),
            dsap(2)
        }
        MAX-ACCESS  read-create
        STATUS      current
        DESCRIPTION
            "The format of the value in docsDevFilterLLCProtocol:
             either a two-byte Ethernet Ethertype, or a one-byte
             802.2 SAP value. EtherType(1) also applies to SNAP-
             encapsulated frames."
        DEFVAL { ethertype }
        ::= { docsDevFilterLLCEntry 4 }

docsDevFilterLLCProtocol OBJECT-TYPE
        SYNTAX      Integer32 (0..65535)
        MAX-ACCESS  read-create
        STATUS      current
        DESCRIPTION
            "The layer three protocol for which this filter applies.
             The protocol value format depends on
```

```
             docsDevFilterLLCProtocolType. Note that for SNAP frames,
             etherType filtering is performed rather than DSAP=0xAA."
        DEFVAL { 0 }
        ::= { docsDevFilterLLCEntry 5 }

docsDevFilterLLCMatches OBJECT-TYPE
        SYNTAX      Counter32
        MAX-ACCESS  read-only
        STATUS      current
        DESCRIPTION
            "Counts the number of times this filter was matched."
        ::= { docsDevFilterLLCEntry 6 }

-- The default behavior for (bridged) packets that do not match IP
-- filters is defined by
-- docsDevFilterIpDefault.

docsDevFilterIpDefault OBJECT-TYPE
        SYNTAX INTEGER {
            discard(1),
            accept(2)
        }
        MAX-ACCESS  read-write
        STATUS      current
        DESCRIPTION
            "If set to discard(1), all packets not matching an IP filter
             will be discarded. If set to accept(2), all packets not
             matching an IP filter will be accepted for further
             processing (e.g., bridging).
             At initial system startup, this object returns accept(2)."
        ::= { docsDevFilter 3 }

docsDevFilterIpTable OBJECT-TYPE
        SYNTAX      SEQUENCE OF DocsDevFilterIpEntry
        MAX-ACCESS  not-accessible
        STATUS      current
        DESCRIPTION
            "An ordered list of filters or classifiers to apply to
             IP traffic. Filter application is ordered by the filter
             index, rather than by a best match algorithm (Note that
             this implies that the filter table may have gaps in the
             index values). Packets which match no filters will have
             policy 0 in the docsDevFilterPolicyTable applied to them if
             it exists. Otherwise, Packets which match no filters
             are discarded or forwarded according to the setting of
             docsDevFilterIpDefault.

             Any IP packet can theoretically match multiple rows of
```

```
            this table.  When considering a packet, the table is
            scanned in row index order (e.g. filter 10 is checked
            before filter 20).  If the packet matches that filter
            (which means that it matches ALL criteria for that row),
            actions appropriate to docsDevFilterIpControl and
            docsDevFilterPolicyId are taken.  If the packet was
            discarded processing is complete.  If
            docsDevFilterIpContinue is set to true, the filter
            comparison continues with the next row in the table
            looking for additional matches.

            If the packet matches no filter in the table, the packet
            is accepted or dropped for further processing based on
            the setting of docsDevFilterIpDefault. If the packet is
            accepted, the actions specified by policy group 0
            (e.g. the rows in docsDevFilterPolicyTable which have a
            value of 0 for docsDevFilterPolicyId) are taken if that
            policy group exists.

            Logically, this table is consulted twice during the
            processing of any IP packet - once upon its acceptance
            from the L2 entity, and once upon its transmission to the
            L2 entity.  In actuality, for cable modems, IP filtering
            is generally the only IP processing done for transit
            traffic.  This means that inbound and outbound filtering
            can generally be done at the same time with one pass
            through the filter table."
        ::= { docsDevFilter 4 }

docsDevFilterIpEntry OBJECT-TYPE
        SYNTAX      DocsDevFilterIpEntry
        MAX-ACCESS  not-accessible
        STATUS      current
        DESCRIPTION
            "Describes a filter to apply to IP traffic received on a
             specified interface.  All identity objects in this table
             (e.g. source and destination address/mask, protocol,
             source/dest port, TOS/mask, interface and direction) must
             match their respective fields in the packet for any given
             filter to match.

             To create an entry in this table, docsDevFilterIpIfIndex
             must be specified."
        INDEX { docsDevFilterIpIndex }
        ::= { docsDevFilterIpTable 1 }

DocsDevFilterIpEntry ::= SEQUENCE {
            docsDevFilterIpIndex            Integer32,
```

```
            docsDevFilterIpStatus           RowStatus,
            docsDevFilterIpControl          INTEGER,
            docsDevFilterIpIfIndex          InterfaceIndexOrZero,
            docsDevFilterIpDirection        INTEGER,
            docsDevFilterIpBroadcast        TruthValue,
            docsDevFilterIpSaddr            IpAddress,
            docsDevFilterIpSmask            IpAddress,
            docsDevFilterIpDaddr            IpAddress,
            docsDevFilterIpDmask            IpAddress,
            docsDevFilterIpProtocol         Integer32,
            docsDevFilterIpSourcePortLow    Integer32,
            docsDevFilterIpSourcePortHigh   Integer32,
            docsDevFilterIpDestPortLow      Integer32,
            docsDevFilterIpDestPortHigh     Integer32,
            docsDevFilterIpMatches          Counter32,
            docsDevFilterIpTos              OCTET STRING,
            docsDevFilterIpTosMask          OCTET STRING,
            docsDevFilterIpContinue         TruthValue,
            docsDevFilterIpPolicyId         Integer32
        }

docsDevFilterIpIndex OBJECT-TYPE
        SYNTAX      Integer32 (1..2147483647)
        MAX-ACCESS  not-accessible
        STATUS      current
        DESCRIPTION
            "Index used to order the application of filters.
             The filter with the lowest index is always applied
             first."
        ::= { docsDevFilterIpEntry 1 }

docsDevFilterIpStatus OBJECT-TYPE
        SYNTAX      RowStatus
        MAX-ACCESS  read-create
        STATUS      current
        DESCRIPTION
            "Controls and reflects the status of rows in this
             table. Specifying only this object (with the appropriate
             index) on a CM is sufficient to create a filter row which
             matches all inbound packets on the ethernet interface,
             and results in the packets being
             discarded. docsDevFilterIpIfIndex (at least) must be
             specified on a CMTS to create a row.  Creation of the
             rows may be done via either create-and-wait or
             create-and-go, but the filter is not applied until this
             object is set to (or changes to) active. There is no
             restriction in changing any object in a row while this
             object is set to active."
```

```
        ::= { docsDevFilterIpEntry 2 }

docsDevFilterIpControl OBJECT-TYPE
        SYNTAX INTEGER {
            discard(1),
            accept(2),
            policy(3)
        }
        MAX-ACCESS  read-create
        STATUS      current
        DESCRIPTION
            "If set to discard(1), all packets matching this filter
             will be discarded and scanning of the remainder of the
             filter list will be aborted. If set to accept(2), all
             packets matching this filter will be accepted for further
             processing (e.g., bridging). If docsDevFilterIpContinue
             is set to true, see if there are other matches, otherwise
             done. If set to policy (3), execute the policy entries
             matched by docsDevIpFilterPolicyId in
             docsDevIpFilterPolicyTable.

             If is docsDevFilterIpContinue is set to true, continue
             scanning the table for other matches, otherwise done."
        DEFVAL { discard }
        ::= { docsDevFilterIpEntry 3 }

docsDevFilterIpIfIndex OBJECT-TYPE
        SYNTAX      InterfaceIndexOrZero
        MAX-ACCESS  read-create
        STATUS      current
        DESCRIPTION
            "The entry interface to which this filter applies. The
             value corresponds to ifIndex for either a CATV MAC or
             another network interface. If the value is zero, the
             filter applies to all interfaces. Default value in Cable
             Modems is the index of the customer-side (e.g. ethernet)
             interface. In Cable Modem Termination Systems, this
             object MUST be specified to create a row in this table."
        ::= { docsDevFilterIpEntry 4 }

docsDevFilterIpDirection OBJECT-TYPE
        SYNTAX INTEGER {
            inbound(1),
            outbound(2),
            both(3)
        }
        MAX-ACCESS  read-create
        STATUS      current
```

```
        DESCRIPTION
            "Determines whether the filter is applied to inbound(1)
             traffic, outbound(2) traffic, or traffic in both(3)
             directions."
        DEFVAL { inbound }
        ::= { docsDevFilterIpEntry 5 }

docsDevFilterIpBroadcast OBJECT-TYPE
        SYNTAX      TruthValue
        MAX-ACCESS  read-create
        STATUS      current
        DESCRIPTION
            "If set to true(1), the filter only applies to multicast
             and broadcast traffic. If set to false(2), the filter
             applies to all traffic."
        DEFVAL { false }
        ::= { docsDevFilterIpEntry 6 }

docsDevFilterIpSaddr OBJECT-TYPE
        SYNTAX      IpAddress
        MAX-ACCESS  read-create
        STATUS      current
        DESCRIPTION
            "The source IP address, or portion thereof, that is to be
             matched for this filter.  The source address is first
             masked (and'ed) against docsDevFilterIpSmask before being
             compared  to this value.  A value of 0 for this object
             and 0 for the mask matches all IP addresses."
        DEFVAL { '00000000'h }
        ::= { docsDevFilterIpEntry 7 }

docsDevFilterIpSmask OBJECT-TYPE
        SYNTAX      IpAddress
        MAX-ACCESS  read-create
        STATUS      current
        DESCRIPTION
            "A bit mask that is to be applied to the source address
             prior to matching. This mask is not necessarily the same
             as a subnet mask, but 1's bits must be leftmost and
             contiguous."
        DEFVAL { '00000000'h }
        ::= { docsDevFilterIpEntry 8 }

docsDevFilterIpDaddr OBJECT-TYPE
        SYNTAX      IpAddress
        MAX-ACCESS  read-create
        STATUS      current
        DESCRIPTION
```

```
            "The destination IP address, or portion thereof, that is
             to be matched for this filter. The destination address is
             first masked (and'ed) against docsDevFilterIpDmask before
             being compared  to this value.  A value of 0 for this
             object and 0 for the mask matches all IP addresses."
        DEFVAL { '00000000'h }
        ::= { docsDevFilterIpEntry 9 }

docsDevFilterIpDmask OBJECT-TYPE
        SYNTAX      IpAddress
        MAX-ACCESS  read-create
        STATUS      current
        DESCRIPTION
            "A bit mask that is to be applied to the destination
             address prior to matching. This mask is not necessarily
             the same as a subnet mask, but 1's bits must be leftmost
             and contiguous."
        DEFVAL { '00000000'h }
        ::= { docsDevFilterIpEntry 10 }

docsDevFilterIpProtocol OBJECT-TYPE
        SYNTAX Integer32 (0..256)
        MAX-ACCESS  read-create
        STATUS      current
        DESCRIPTION
            "The IP protocol value that is to be matched. For example:
             icmp is 1, tcp is 6, udp is 17. A value of 256 matches
             ANY protocol."
        DEFVAL { 256 }
        ::= { docsDevFilterIpEntry 11 }

docsDevFilterIpSourcePortLow OBJECT-TYPE
        SYNTAX      Integer32 (0..65535)
        MAX-ACCESS  read-create
        STATUS      current
        DESCRIPTION
            "If docsDevFilterIpProtocol is udp or tcp, this is the
             inclusive lower bound of the transport-layer source port
             range that is to be matched, otherwise it is ignored
             during matching."
        DEFVAL { 0 }
        ::= { docsDevFilterIpEntry 12 }

docsDevFilterIpSourcePortHigh OBJECT-TYPE
        SYNTAX      Integer32 (0..65535)
        MAX-ACCESS  read-create
        STATUS      current
        DESCRIPTION
```

```
            "If docsDevFilterIpProtocol is udp or tcp, this is the
             inclusive upper bound of the transport-layer source port
             range that is to be matched, otherwise it is ignored
             during matching."
        DEFVAL { 65535 }
        ::= { docsDevFilterIpEntry 13 }

docsDevFilterIpDestPortLow OBJECT-TYPE
        SYNTAX      Integer32 (0..65535)
        MAX-ACCESS  read-create
        STATUS      current
        DESCRIPTION
            "If docsDevFilterIpProtocol is udp or tcp, this is the
             inclusive lower bound of the transport-layer destination
             port range that is to be matched, otherwise it is ignored
             during matching."
        DEFVAL { 0 }
        ::= { docsDevFilterIpEntry 14 }

docsDevFilterIpDestPortHigh OBJECT-TYPE
        SYNTAX      Integer32 (0..65535)
        MAX-ACCESS  read-create
        STATUS      current
        DESCRIPTION
            "If docsDevFilterIpProtocol is udp or tcp, this is the
             inclusive upper bound of the transport-layer destination
             port range that is to be matched, otherwise it is ignored
             during matching."
        DEFVAL { 65535 }
        ::= { docsDevFilterIpEntry 15 }

docsDevFilterIpMatches OBJECT-TYPE
        SYNTAX      Counter32
        MAX-ACCESS  read-only
        STATUS      current
        DESCRIPTION
            "Counts the number of times this filter was matched.
             This object is initialized to 0 at boot, or at row
             creation, and is reset only upon reboot."
        ::= { docsDevFilterIpEntry 16 }

docsDevFilterIpTos  OBJECT-TYPE
        SYNTAX      OCTET STRING ( SIZE (1))
        MAX-ACCESS  read-create
        STATUS      current
        DESCRIPTION
            "This is the value to be matched to the packet's
             TOS (Type of Service) value (after the TOS value
```

```
             is AND'd with docsDevFilterIpTosMask).  A value for this
             object of 0 and a mask of 0 matches all TOS values."
         DEFVAL { '00'h }
         ::= { docsDevFilterIpEntry 17 }

docsDevFilterIpTosMask OBJECT-TYPE
         SYNTAX      OCTET STRING ( SIZE (1) )
         MAX-ACCESS  read-create
         STATUS      current
         DESCRIPTION
             "The mask to be applied to the packet's TOS value before
              matching."
         DEFVAL { '00'h }
         ::= { docsDevFilterIpEntry 18 }

docsDevFilterIpContinue OBJECT-TYPE
         SYNTAX      TruthValue
         MAX-ACCESS  read-create
         STATUS      current
         DESCRIPTION
             "If this value is set to true, and docsDevFilterIpControl
              is anything but discard (1), continue scanning and
              applying policies."
         DEFVAL { false }
         ::= { docsDevFilterIpEntry 19 }

docsDevFilterIpPolicyId OBJECT-TYPE
         SYNTAX      Integer32 (0..2147483647)
         MAX-ACCESS  read-create
         STATUS      current
         DESCRIPTION
             "This object points to an entry in docsDevFilterPolicyTable.
              If docsDevFilterIpControl is set to policy (3), execute
              all matching policies in docsDevFilterPolicyTable.
              If no matching policy exists, treat as if
              docsDevFilterIpControl were set to accept (1).
              If this object is set to the value of 0, there is no
              matching policy, and docsDevFilterPolicyTable MUST NOT be
              consulted."
         DEFVAL { 0 }
         ::= { docsDevFilterIpEntry 20 }

--
--

docsDevFilterPolicyTable OBJECT-TYPE
         SYNTAX      SEQUENCE OF DocsDevFilterPolicyEntry
         MAX-ACCESS  not-accessible
```

```
        STATUS      current
        DESCRIPTION
            "A Table which maps between a policy group ID and a set of
             policies to be applied.  All rows with the same
             docsDevFilterPolicyId are part of the same policy group
             and are applied in the order in which they are in this
             table.

             docsDevFilterPolicyTable exists to allow multiple policy
             actions to be applied to any given classified packet. The
             policy actions are applied in index order For example:

             Index   ID    Type    Action
              1      1      TOS     1
              9      5      TOS     1
              12     1      IPSEC   3

             This says that a packet which matches a filter with
             policy id 1, first has TOS policy 1 applied (which might
             set the TOS bits to enable a higher priority), and next
             has the IPSEC policy 3 applied (which may result in the
             packet being dumped into a secure VPN to a remote
             encryptor).

             Policy ID 0 is reserved for default actions and is
             applied only to packets which match no filters in
             docsDevIpFilterTable."
        ::= { docsDevFilter 5 }

docsDevFilterPolicyEntry OBJECT-TYPE
        SYNTAX      DocsDevFilterPolicyEntry
        MAX-ACCESS  not-accessible
        STATUS      current
        DESCRIPTION
            "An entry in the docsDevFilterPolicyTable. Entries are
             created by Network Management. To create an entry,
             docsDevFilterPolicyId and docsDevFilterPolicyAction
             must be specified."
        INDEX { docsDevFilterPolicyIndex }
        ::= { docsDevFilterPolicyTable 1 }

DocsDevFilterPolicyEntry ::= SEQUENCE {
            docsDevFilterPolicyIndex   Integer32,
            docsDevFilterPolicyId      Integer32,
--            docsDevFilterPolicyType    INTEGER,
--            docsDevFilterPolicyAction  Integer32,
            docsDevFilterPolicyStatus  RowStatus,
            docsDevFilterPolicyPtr     RowPointer
```

```
        }

docsDevFilterPolicyIndex OBJECT-TYPE
        SYNTAX      Integer32 (1..2147483647)
        MAX-ACCESS  not-accessible
        STATUS      current
        DESCRIPTION "Index value for the table."
        ::= { docsDevFilterPolicyEntry 1 }

docsDevFilterPolicyId OBJECT-TYPE
        SYNTAX      Integer32 (0..2147483647)
        MAX-ACCESS  read-create
        STATUS      current
        DESCRIPTION
             "Policy ID for this entry. A policy ID can apply to
              multiple rows of this table, all relevant policies are
              executed. Policy 0 (if populated) is applied to all
              packets which do not match any of the filters. N.B. If
              docsDevFilterIpPolicyId is set to 0, it DOES NOT match
              policy 0 of this table. "
        ::= { docsDevFilterPolicyEntry 2 }

-- docsDevFilterPolicyType ::= { docsDevFilterPolicyEntry 3} Removed
-- docsDevFilterPolicyAction ::= { docsDevFilterPolicyEntry 4 } removed

docsDevFilterPolicyStatus OBJECT-TYPE
        SYNTAX      RowStatus
        MAX-ACCESS  read-create
        STATUS      current
        DESCRIPTION
            "Object used to create an entry in this table."
        ::= { docsDevFilterPolicyEntry 5 }


docsDevFilterPolicyPtr OBJECT-TYPE
        SYNTAX      RowPointer
        MAX-ACCESS  read-create
        STATUS      current
        DESCRIPTION
            "This object points to a row in an applicable filter policy
             table.  Currently, the only standard policy table is
             docsDevFilterTosTable. Per the textual convention, this
             object points to the first accessible object in the row.
             E.g. to point to a row in docsDevFilterTosTable with an
             index of 21, the value of this object would be the object
             identifier docsDevTosStatus.21.

             Vendors must adhere to the same convention when adding
```

```
            vendor specific policy table extensions.

            The default upon row creation is a null pointer which
            results in no policy action being taken."
        DEFVAL { zeroDotZero }
        ::= { docsDevFilterPolicyEntry 6 }

--
-- TOS Policy action table
--

docsDevFilterTosTable OBJECT-TYPE
        SYNTAX      SEQUENCE OF DocsDevFilterTosEntry
        MAX-ACCESS  not-accessible
        STATUS      current
        DESCRIPTION
            "Table used to describe Type of Service (TOS) bits
            processing.

            This table is an adjunct to the docsDevFilterIpTable, and
            the docsDevFilterPolicy table.  Entries in the latter
            table can point to specific rows in this (and other)
            tables and cause specific actions to be taken.  This table
            permits the manipulation of the value of the Type of
            Service bits in the IP header of the matched packet as
            follows:
            Set the tosBits of the packet to
               (tosBits & docsDevFilterTosAndMask) |
                                              docsDevFilterTosOrMask

            This construct allows you to do a clear and set of all
            the TOS bits in a flexible manner."
        ::= { docsDevFilter 6 }

docsDevFilterTosEntry OBJECT-TYPE
        SYNTAX      DocsDevFilterTosEntry
        MAX-ACCESS  not-accessible
        STATUS      current
        DESCRIPTION
            "A TOS policy entry."
        INDEX { docsDevFilterTosIndex }
        ::= { docsDevFilterTosTable 1 }

DocsDevFilterTosEntry ::= SEQUENCE {
            docsDevFilterTosIndex   Integer32,
            docsDevFilterTosStatus  RowStatus,
            docsDevFilterTosAndMask OCTET STRING (SIZE (1)),
            docsDevFilterTosOrMask  OCTET STRING (SIZE (1))
```

```
        }

docsDevFilterTosIndex OBJECT-TYPE
        SYNTAX      Integer32 (1..2147483647)
        MAX-ACCESS  not-accessible
        STATUS      current
        DESCRIPTION
            "The unique index for this row.  There are no ordering
             requirements for this table and any valid index may be
             specified."
        ::= { docsDevFilterTosEntry 1 }

docsDevFilterTosStatus OBJECT-TYPE
        SYNTAX      RowStatus
        MAX-ACCESS  read-create
        STATUS      current
        DESCRIPTION
            "The object used to create and delete entries in this
             table. A row created by specifying just this object
             results in a row which specifies no change to the TOS
             bits.   A row may be created using either the create-and-go
             or create-and-wait paradigms. There is no restriction on
             the ability to change values in this row while the row is
             active."
        ::= { docsDevFilterTosEntry 2 }

docsDevFilterTosAndMask OBJECT-TYPE
        SYNTAX      OCTET STRING (SIZE (1))
        MAX-ACCESS  read-create
        STATUS      current
        DESCRIPTION

            "This value is bitwise AND'd with the matched  packet's
        TOS bits."
        DEFVAL { 'ff'h }
        ::= { docsDevFilterTosEntry 3 }

docsDevFilterTosOrMask OBJECT-TYPE
        SYNTAX      OCTET STRING (SIZE (1))
        MAX-ACCESS  read-create
        STATUS      current
        DESCRIPTION
            "After bitwise AND'ing with the above bits, the packet's
             TOS bits are bitwise OR'd with these bits."
        DEFVAL { '00'h }
        ::= { docsDevFilterTosEntry 4 }
```

```
--
-- CPE IP Management and anti spoofing group.  Only implemented on
-- Cable Modems.
--

docsDevCpe OBJECT IDENTIFIER ::= { docsDevMIBObjects 7}

docsDevCpeEnroll OBJECT-TYPE
        SYNTAX      INTEGER {
            none(1),
            any(2)
        }
        MAX-ACCESS  read-write
        STATUS      current
        DESCRIPTION
            "This object controls the population of docsDevFilterCpeTable.
             If set to none, the filters must be set manually.
             If set to any, the CM wiretaps the packets originating
             from the ethernet and enrolls up to docsDevCpeIpMax
             addresses based on the source IP addresses of those
             packets. At initial system startup, default value for this
             object is any(2)."
        ::= { docsDevCpe 1 }

docsDevCpeIpMax OBJECT-TYPE
        SYNTAX      Integer32 (-1..2147483647)
        MAX-ACCESS  read-write
        STATUS      current
        DESCRIPTION
            "This object controls the maximum number of CPEs allowed to
             connect behind this device. If set to zero, any number of
             CPEs may connect up to the maximum permitted for the device.
             If set to -1, no filtering is done on CPE source addresses,
             and no entries are made in the docsDevFilterCpeTable. If an
             attempt is made to set this to a number greater than that
             permitted for the device, it is set to that maximum.
             At iniitial system startup, default value for this object
             is 1."
        ::= { docsDevCpe 2 }

docsDevCpeTable OBJECT-TYPE
        SYNTAX      SEQUENCE OF DocsDevCpeEntry
        MAX-ACCESS  not-accessible
        STATUS      current
        DESCRIPTION
            "This table lists the IP addresses seen (or permitted)  as
             source addresses in packets originating from the customer
             interface on this device. In addition, this table can be
```

```
             provisioned with the specific addresses permitted for the
             CPEs via the normal row creation mechanisms."
        ::= { docsDevCpe 3 }

docsDevCpeEntry OBJECT-TYPE
        SYNTAX      DocsDevCpeEntry
        MAX-ACCESS  not-accessible
        STATUS      current
        DESCRIPTION
            "An entry in the docsDevFilterCpeTable. There is one entry
             for each IP CPE seen or provisioned. If docsDevCpeIpMax
             is set to -1, this table is ignored, otherwise: Upon receipt
             of an IP  packet from the customer interface of the CM, the
             source IP address is checked against this table. If the
             address is in the table, packet processing continues.
             If the address is not in the table, but docsDevCpeEnroll
             is set to any and the table size is less than
             docsDevCpeIpMax, the address is added to the table and
             packet processing continues. Otherwise, the packet is
             dropped.

             The filtering actions specified by this table occur after
             any LLC filtering (docsDevFilterLLCTable), but prior
             to any IP filtering (docsDevFilterIpTable,
             docsDevNmAccessTable)."
        INDEX   { docsDevCpeIp }
        ::= {docsDevCpeTable 1 }

DocsDevCpeEntry ::= SEQUENCE {
            docsDevCpeIp      IpAddress,
            docsDevCpeSource  INTEGER,
            docsDevCpeStatus  RowStatus
        }

docsDevCpeIp OBJECT-TYPE
        SYNTAX      IpAddress
        MAX-ACCESS  not-accessible
        STATUS      current
        DESCRIPTION
            "The IP address to which this entry applies."
        ::= { docsDevCpeEntry 1 }

docsDevCpeSource OBJECT-TYPE
        SYNTAX      INTEGER {
            other(1),
            manual(2),
            learned(3)
        }
```

```
        MAX-ACCESS  read-only
        STATUS      current
        DESCRIPTION
            "This object describes how this entry was created. If the
             value is manual(2), this row was created by a network
             management action (either configuration, or SNMP set).
             If set to learned(3), then it was found via
             looking at the source IP address of a received packet."
        ::= { docsDevCpeEntry 2 }

docsDevCpeStatus OBJECT-TYPE
        SYNTAX  RowStatus
        MAX-ACCESS  read-create
        STATUS  current
        DESCRIPTION
            "Standard object to manipulate rows. To create a row in this
             table, you only need to specify this object. Management
             stations SHOULD use the create-and-go mechanism for
             creating rows in this table."
        ::= { docsDevCpeEntry 3 }

--
-- Placeholder for notifications/traps.
--
docsDevNotification OBJECT IDENTIFIER   ::= { docsDev 2 }


--
-- Conformance definitions
--
docsDevConformance  OBJECT IDENTIFIER   ::= { docsDev 3 }
docsDevGroups       OBJECT IDENTIFIER   ::= { docsDevConformance 1 }
docsDevCompliances  OBJECT IDENTIFIER   ::= { docsDevConformance 2 }

docsDevBasicCompliance MODULE-COMPLIANCE
        STATUS  current
        DESCRIPTION
            "The compliance statement for MCNS Cable Modems and
             Cable Modem Termination Systems."

MODULE  -- docsDev

-- conditionally mandatory groups

GROUP docsDevBaseGroup
        DESCRIPTION
            "Mandatory in Cable Modems, optional in Cable Modem
             Termination Systems."
```

```
GROUP docsDevEventGroup
        DESCRIPTION
            "Mandatory in Cable Modems, optional in Cable Modem
             Termination Systems."

GROUP docsDevFilterGroup
        DESCRIPTION
            "Mandatory in Cable Modems, optional in Cable Modem
             Termination Systems."

GROUP docsDevNmAccessGroup
        DESCRIPTION
            "This group is only implemented in devices which do not
             implement SNMPv3 User Security Model.  It SHOULD NOT be
             implemented by SNMPv3 conformant devices.

             For devices which do not implement SNMPv3 or later, this
             group is Mandatory in Cable Modems and is optional
             in Cable Modem Termination Systems."

GROUP docsDevServerGroup
        DESCRIPTION
            "This group is implemented only in Cable Modems and is
             not implemented in Cable Modem Termination Systems."

GROUP docsDevSoftwareGroup
        DESCRIPTION
            "This group is Mandatory in Cable Modems and  optional in
             Cable Modem Termination Systems."

GROUP docsDevCpeGroup
        DESCRIPTION
            "This group is Mandatory in Cable Modems, and is
             not implemented in Cable Modem Termination Systems.  A
             similar capability for CMTS devices may be proposed later
             after study."

OBJECT docsDevSTPControl
        MIN-ACCESS read-only
        DESCRIPTION
            "It is compliant to implement this object as read-only.
             Devices need only support noStFilterBpdu(2)."

OBJECT docsDevEvReporting
         MIN-ACCESS read-only
         DESCRIPTION
             "It is compliant to implement this object as read-only.
              Devices need only support local(0)."
```

```
         ::= { docsDevCompliances 1 }

docsDevBaseGroup OBJECT-GROUP
        OBJECTS {
             docsDevRole,
             docsDevDateTime,
             docsDevResetNow,
             docsDevSerialNumber,
             docsDevSTPControl
        }
        STATUS      current
        DESCRIPTION
            "A collection of objects providing device status and
             control."
        ::= { docsDevGroups 1 }

docsDevNmAccessGroup OBJECT-GROUP
        OBJECTS {
             docsDevNmAccessIp,
             docsDevNmAccessIpMask,
             docsDevNmAccessCommunity,
             docsDevNmAccessControl,
             docsDevNmAccessInterfaces,
             docsDevNmAccessStatus
        }
        STATUS      current
        DESCRIPTION
            "A collection of objects for controlling access to SNMP
             objects."
        ::= { docsDevGroups 2 }

docsDevSoftwareGroup OBJECT-GROUP
        OBJECTS {
            docsDevSwServer,
            docsDevSwFilename,
            docsDevSwAdminStatus,
            docsDevSwOperStatus,
            docsDevSwCurrentVers
        }
        STATUS      current
        DESCRIPTION
            "A collection of objects for controlling software
             downloads."
        ::= { docsDevGroups 3 }

docsDevServerGroup OBJECT-GROUP
        OBJECTS {
            docsDevServerBootState,
```

```
            docsDevServerDhcp,
            docsDevServerTime,
            docsDevServerTftp,
            docsDevServerConfigFile
        }
        STATUS      current
        DESCRIPTION
            "A collection of objects providing status about server
             provisioning."
        ::= { docsDevGroups 4 }

docsDevEventGroup OBJECT-GROUP
        OBJECTS {
            docsDevEvControl,
            docsDevEvSyslog,
            docsDevEvThrottleAdminStatus,
            docsDevEvThrottleInhibited,
            docsDevEvThrottleThreshold,
            docsDevEvThrottleInterval,
            docsDevEvReporting,
            docsDevEvFirstTime,
            docsDevEvLastTime,
            docsDevEvCounts,
            docsDevEvLevel,
            docsDevEvId,
            docsDevEvText
        }
        STATUS      current
        DESCRIPTION
            "A collection of objects used to control and monitor
             events."
        ::= { docsDevGroups 5 }

docsDevFilterGroup OBJECT-GROUP
        OBJECTS {
            docsDevFilterLLCUnmatchedAction,
            docsDevFilterIpDefault,
            docsDevFilterLLCStatus,
            docsDevFilterLLCIfIndex,
            docsDevFilterLLCProtocolType,
            docsDevFilterLLCProtocol,
            docsDevFilterLLCMatches,
            docsDevFilterIpControl,
            docsDevFilterIpIfIndex,
            docsDevFilterIpStatus,
            docsDevFilterIpDirection,
            docsDevFilterIpBroadcast,
            docsDevFilterIpSaddr,
```

```
            docsDevFilterIpSmask,
            docsDevFilterIpDaddr,
            docsDevFilterIpDmask,
            docsDevFilterIpProtocol,
            docsDevFilterIpSourcePortLow,
            docsDevFilterIpSourcePortHigh,
            docsDevFilterIpDestPortLow,
            docsDevFilterIpDestPortHigh,
            docsDevFilterIpMatches,
            docsDevFilterIpTos,
            docsDevFilterIpTosMask,
            docsDevFilterIpContinue,
            docsDevFilterIpPolicyId,
            docsDevFilterPolicyId,
            docsDevFilterPolicyStatus,
            docsDevFilterPolicyPtr,
            docsDevFilterTosStatus,
            docsDevFilterTosAndMask,
            docsDevFilterTosOrMask
        }
        STATUS      current
        DESCRIPTION
            "A collection of objects to specify filters at link layer
             and IP layer."
        ::= { docsDevGroups 6 }

docsDevCpeGroup OBJECT-GROUP
        OBJECTS {
           docsDevCpeEnroll,
           docsDevCpeIpMax,
           docsDevCpeSource,
           docsDevCpeStatus
        }
        STATUS      current
        DESCRIPTION
            "A collection of objects used to control the number
             and specific values of IP addresses allowed for
             associated Customer Premises Equipment (CPE)."
        ::= { docsDevGroups 7 }

END
```

5. Acknowledgments

This document was produced by the IPCDN Working Group. It is based on a document written by Pam Anderson from CableLabs, Wilson Sawyer from BayNetworks, and Rich Woundy from Continental Cablevision. The original working group editor, Guenter Roeck of cisco Systems, did much of the grunt work of putting the document into its current form.

Special thanks is also due to Azlina Palmer, who helped a lot reviewing the document.

6. References

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

[2] Rose, M. and K. McCloghrie, "Structure and Identification of Management Information for TCP/IP-based Internets", STD 16, RFC 1155, May 1990.

[3] Rose, M. and K. McCloghrie, "Concise MIB Definitions", STD 16, RFC 1212, March 1991.

[4] Rose, M., "A Convention for Defining Traps for use with the SNMP", RFC 1215, March 1991.

[5] McCloghrie, K., Perkins, D. and J. Schoenwaelder, "Structure of Management Information for Version 2 (SMIv2)", STD 58, RFC 2578, April 1999.

[6] McCloghrie, K., Perkins, D. and J. Schoenwaelder, "Textual Conventions for SMIv2", STD 58, RFC 2579, April 1999.

[7] McCloghrie, K., Perkins, D. and J. Schoenwaelder, "Conformance Statements for SMIv2", STD 58, RFC 2580, April 1999.

[8] Case, J., Fedor, M., Schoffstall, M. and J. Davin, "Simple Network Management Protocol", STD 15, RFC 1157, May 1990.

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

[10] 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.

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

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

[13] 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.

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

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

[16] "Data-Over-Cable Service Interface Specifications: Cable Modem Radio Frequency Interface Specification SP-RFI-I04-980724", DOCSIS, July 1998, http://www.cablemodem.com/public/pubtechspec/SP-RFI-I04-980724.pdf.

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

[18] "Data-Over-Cable Service Interface Specifications: Operations Support System Interface Specification RF Interface SP-OSSI-RF-I02-980410", DOCSIS, April 1998, http://www.cablemodem.com/public/pubtechspec/ossi/sp-ossi.PDF.

[19] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[20] "Data-Over-Cable Service Interface Specifications: Baseline Privacy Interface Specification SP-BPI-I01-970922", DOCSIS, September 1977, http://www.cablemodem.com/public/pubtechspec/ss/SP-BPI-I01-970922.pdf

7. Security Considerations

This MIB relates to a system which will provide metropolitan public internet access. As such, improper manipulation of the objects represented by this MIB may result in denial of service to a large number of end-users. In addition, manipulation of the

docsDevNmAccessTable, docsDevFilterLLCTable, docsDevFilterIpTable and the elements of the docsDevCpe group may allow an end-user to increase their service levels, spoof their IP addresses, change the permitted management stations, or affect other end-users in either a positive or negative manner.

There are a number of management objects defined in this MIB that have a MAX-ACCESS clause of read-write and/or read-create. Such objects may be considered sensitive or vulnerable in some network environments. The support for SET operations in a non-secure environment without proper protection can have a negative effect on network operations. In addition to those mentioned above:

- The use of docsDevNmAccessTable to specify management stations is considered to be only limited protection and does not protect against attacks which spoof the management station's IP address. The use of stronger mechanisms such as SNMPv3 security should be considered where possible. Specifically, SNMPv3 VACM and USM MUST be used with any v3 agent which implements this MIB. Administrators may also wish to consider whether even read access to docsDevNmAccessTable may be undesirable under certain circumstances.

- The CM may have its software changed by the actions of the management system. An improper software load may result in substantial vulnerabilities and the loss of the ability of the management system to control the cable modem.

- The device may be reset by setting docsDevResetNow = true(1). This causes the device to reload its configuration files as well as eliminating all previous non-persistent network management settings. As such, this may provide a vector for attacking the system.

- Setting docsDevEvThrottleAdminStatus = unconstrained(1) (which is also the DEFVAL) may cause flooding of traps, which can disrupt network service.

This MIB does not affect confidentiality of services on a cable modem system. [20] specifies the implementation of the DOCSIS Baseline privacy mechanism. The working group expects to issue a MIB for the management of this mechanism at a later time.

SNMPv1 by itself is not a secure environment. Even if the network itself is secure (for example by using IPSec), even then, there is no control as to who on the secure network is allowed to access and GET/SET (read/change/create/delete) the objects in this MIB.

It is recommended that the implementers consider the security features as provided by the SNMPv3 framework. Specifically, the use of the User-based Security Model [12] and the View-based Access Control Model [15] is recommended.

It is then a customer/user responsibility to ensure that the SNMP entity giving access to an instance of this MIB, is properly configured to give access to the objects only to those principals (users) that have legitimate rights to indeed GET or SET (change/create/delete) them.

8. Intellectual Property

The IETF takes no position regarding the validity or scope of any intellectual property or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; neither does it represent that it has made any effort to identify any such rights. Information on the IETF's procedures with respect to rights in standards-track and standards-related documentation can be found in BCP-11. Copies of claims of rights made available for publication and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementors or users of this specification can be obtained from the IETF Secretariat.

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights which may cover technology that may be required to practice this standard. Please address the information to the IETF Executive Director.

9. Author's Address

Michael StJohns
@Home Network
425 Broadway
Redwood City, CA 94063
U.S.A

Phone: +1 650 569 5368
EMail: stjohns@corp.home.net

10. Full Copyright Statement

Acknowledgement

Funding for the RFC Editor function is currently provided by the Internet Society.