

Multimedia Terminal Adapter (MTA) Management Information Base  
for PacketCable- and IPcablecom-Compliant Devices

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

Copyright (C) The IETF Trust (2006).

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 Simple Network Management Protocol (SNMP)-based management of PacketCable- and IPcablecom-compliant Multimedia Terminal Adapter devices.

Table of Contents

1. The Internet-Standard Management Framework .....	2
2. Terminology .....	2
3. Introduction .....	4
3.1. Structure of the MTA MIB .....	5
3.2. pktcMtaDevBase .....	5
3.3. pktcMtaDevServer .....	6
3.4. pktcMtaDevSecurity .....	6
3.5. Relationship between MIB Objects in the MTA MIB .....	7
3.6. Secure Software Download .....	8
3.7. X.509 Certificates Dependencies .....	8
4. Definitions .....	9
5. Acknowledgements .....	52
6. Security Considerations .....	52
7. IANA Considerations .....	55
8. Normative References .....	55
9. Informative References .....	57

## 1. The Internet-Standard Management Framework

For a detailed overview of the documents that describe the current Internet-Standard Management Framework, please refer to section 7 of RFC 3410 [RFC3410].

Managed objects are accessed via a virtual information store, termed the Management Information Base or MIB. MIB objects are generally accessed through the Simple Network Management Protocol (SNMP). Objects in the MIB are defined using the mechanisms defined in the Structure of Management Information (SMI). This memo specifies a MIB module that is compliant to the SMIV2, which is described in STD 58, RFC 2578 [RFC2578], STD 58, RFC 2579 [RFC2579] and STD 58, RFC 2580 [RFC2580].

## 2. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL", when used in the guidelines in this memo, are to be interpreted as described in RFC 2119 [RFC2119].

The terms "MIB module" and "information module" are used interchangeably in this memo. As used here, both terms refer to any of the three types of information modules defined in Section 3 of RFC 2578 [RFC2578].

Some of the terms used in this memo are defined below. Some additional terms are also defined in the PacketCable MTA Device Provisioning Specification [PKT-SP-PROV] and the PacketCable Security Specification [PKT-SP-SEC].

### DOCSIS

The CableLabs(R) Certified(TM) Cable Modem project, also known as DOCSIS(R) (Data over Cable Service Interface Specification), defines interface requirements for cable modems involved in high-speed data distribution over cable television system networks. DOCSIS also refers to the ITU-T J.112 recommendation, Annex B, for cable modem systems [ITU-T-J112].

### Cable Modem

A Cable Modem (CM) acts as a data transport agent used to transfer call management and voice data packets over a DOCSIS-compliant cable system.

### Multimedia Terminal Adapter

A Multimedia Terminal Adapter (MTA) is a PacketCable- or IP-Cablecom-compliant device providing telephony services over a cable or hybrid

system used to deliver video signals to a community. It contains an interface to endpoints, a network interface, CODECs, and all signaling and encapsulation functions required for Voice over IP transport, call signaling, and Quality of Service signaling. An MTA can be an embedded or a standalone device. An Embedded MTA (E-MTA) is an MTA device containing an embedded DOCSIS Cable Modem. A Standalone MTA (S-MTA) is an MTA device separated from the DOCSIS cable modem by non-DOCSIS Message Access Control (MAC) interface (e.g., Ethernet, USB).

#### Endpoint

An endpoint or MTA endpoint is a standard RJ-11 telephony physical port located on the MTA and used for attaching the telephone device to the MTA.

#### X.509 Certificate

A X.509 certificate is an Internet X.509 Public Key Infrastructure certificate developed as part of the ITU-T X.500 Directory recommendations. It is defined in RFC 3280 [RFC3280] and RFC 4630 [RFC4630].

#### Voice over IP

Voice over IP (VoIP) is a technology providing the means to transfer digitized packets with voice information over IP networks.

#### Public Key Certificate

A Public Key Certificate (also known as a Digital Certificate) is a binding between an entity's public key and one or more attributes relating to its identity.

#### DHCP

The Dynamic Host Configuration Protocol (DHCP) is defined by RFC 2131 [RFC2131]. In addition, commonly used DHCP options are defined in RFC 2132 [RFC2132]. Additional DHCP options used by PacketCable and IPCablecom MTAs can be found in the CableLabs Client Configuration DHCP specifications, RFC 3495 [RFC3495] and RFC 3594 [RFC3594].

#### TFTP

The Trivial File Transfer Protocol (TFTP) is defined by RFC 1350 [RFC1350].

#### HTTP

The Hypertext Transfer Protocol (HTTP/1.1) is defined by RFC 2616 [RFC2616].

#### Call Management Server

A Call Management Server (CMS) is an element of the PacketCable network infrastructure that controls audio connections between MTAs.

#### CODEC, COder-DECoder

A Coder-DECoder is a hardware or software component used in audio/video systems to convert an analog signal to digital, and then (possibly) to compress it so that lower bandwidth telecommunications channels can be used. The signal is decompressed and converted (decoded) back to analog output by a compatible CODEC at the receiving end.

#### Operations Systems Support

An Operations Systems Support system (OSS) is a system of back office software components used for fault, configuration, accounting, performance, and security management working in interaction with each other and providing the operations support in deployed PacketCable systems.

#### Key Distribution Center

A Key Distribution Center (KDC) is an element of the OSS systems functioning as a Kerberos Security Server, providing mutual authentication of the various components of the PacketCable system (e.g., mutual authentication between an MTA and a CMS, or between an MTA and the Provisioning Server).

#### Security Association

A Security Association (SA) is a one-way relationship between a sender and a receiver offering security services on the communication flow.

### 3. Introduction

This MIB module provides a set of objects required for the management of PacketCable, ETSI, and ITU-T IPcablecom compliant MTA devices. The MTA MIB module is intended to supersede various MTA MIB modules from which it is partly derived:

- The PacketCable 1.0 MTA MIB Specification [PKT-SP-MIB-MTA].
- The ITU-T IPcablecom MTA MIB requirements [ITU-T-J168].
- The ETSI MTA MIB [ETSITS101909-8]. The ETSI MTA MIB requirements also refer to various signal characteristics defined in [EN300001], Chapter 3, titled 'Ringing Signal Characteristics', and [EN300659-1].

Several normative and informative references are used to help define MTA MIB objects. As a convention, wherever PacketCable and IPcablecom requirements are equivalent, the PacketCable reference is used in the object REFERENCE clause. IPcablecom-compliant MTA devices MUST use the equivalent IPcablecom references.

### 3.1. Structure of the MTA MIB

The MTA MIB module is identified by `pktcIetfMtaMib` and is structured in three object groups:

- `pktcMtaDevBase` defines the management information pertinent to the MTA device itself.
- `pktcMtaDevServer` defines the management information pertinent to the provisioning back office servers.
- `pktcMtaDevSecurity` defines the management information pertinent to the PacketCable and IPCablecom security mechanisms.

The first two object groups, `pktcMtaDevBase` and `pktcMtaDevServer`, contain only scalar information objects describing the corresponding characteristics of the MTA device and back office servers.

The third group, `pktcMtaDevSecurity`, contains two tables controlling the logical associations between KDC realms and Application Servers (CMS and Provisioning Server). The rows in the various tables of the MTA MIB module can be created automatically (e.g., by the device according to the current state information), or they can be created by the management station, depending on the operational situation. The tables defined in the MTA MIB module may have a mixture of both types of rows.

### 3.2. `pktcMtaDevBase`

This object group contains the management information related to the MTA device itself. It also contains some objects used to control the MTA state. Some highlights are as follows:

- `pktcMtaDevSerialNumber`. This object contains the MTA Serial Number.
- `pktcMtaDevEndPntCount`. This object contains the number of endpoints present in the managed MTA.
- `pktcMtaDevProvisioningState`. This object contains the information describing the completion state of the MTA initialization process.
- `pktcMtaDevEnabled`. This object controls the administrative state of the MTA endpoints and allows operators to enable or disable telephony services on the device.
- `pktcMtaDevResetNow`. This object is used to instruct the MTA to reset.

### 3.3. pktcMtaDevServer

This object group contains the management information describing the back office servers and the parameters related to the communication timers. It also includes some objects controlling the initial MTA interaction with the Provisioning Server.

Some highlights are as follows:

- pktcMtaDevServerDhcp1. This object contains the IP address of the primary DHCP server designated for the MTA provisioning.
- pktcMtaDevServerDhcp2. This object contains the IP address of the secondary DHCP server designated for the MTA provisioning.
- pktcMtaDevServerDns1. This object contains the IP address of the primary DNS used by the managed MTA to resolve the Fully Qualified Domain Name (FQDN) and IP addresses.
- pktcMtaDevServerDns2. This object contains the IP address of the secondary DNS used by the managed MTA to resolve the FQDN and IP addresses.
- pktcMtaDevConfigFile. This object contains the name of the provisioning configuration file the managed MTA must download from the Provisioning Server.
- pktcMtaDevProvConfigHash. This object contains the hash value of the MTA configuration file calculated over its content. When the managed MTA downloads the file, it authenticates the configuration file using the hash value provided in this object.

### 3.4. pktcMtaDevSecurity

This object group contains the management information describing the security-related characteristics of the managed MTA. It contains two tables describing logical dependencies and parameters necessary to establish Security Associations between the MTA and other Application Servers (back office components and CMSes). The CMS table (pktcMtaDevCmsTable) and the realm table (pktcMtaDevRealmTable) are used for managing the MTA signaling security. The realm table defines the CMS domains. The CMS table defines the CMS within the domains. Each MTA endpoint is associated with one CMS at any given time.

The two tables in this object group are as follows:

- pktcMtaDevRealmTable. This table is used in conjunction with any Application Server that communicates securely with the managed MTA (CMS or Provisioning Server).
- pktcMtaDevCmsTable. This table contains the parameters describing the SA establishment between the MTA and CMSes.

### 3.5. Relationship between MIB Objects in the MTA MIB

This section clarifies the relationship between various MTA MIB objects with respect to the role they play in the process of establishing Security Associations.

The process of Security Association establishment between an MTA and Application Servers is described in the PacketCable Security Specification [PKT-SP-SEC]. In particular, an MTA communicates with 2 types of back office Application Servers: Call Management Servers and Provisioning Servers.

The SA establishment process consists of two steps:

- a. Authentication Server Exchange (AS-exchange). This step provides mutual authentication between the parties; i.e., between an MTA and an Authentication Server. The process of AS-exchange is defined by a number of parameters grouped per each realm. These parameters are gathered in the Realm Table (pktcMtaDevRealmTable). The Realm Table is indexed by the Index Counter and contains conceptual column with the Kerberos realm name.
- b. Application server exchange (AP-exchange). This step allows for the establishment of Security Associations between authenticated parties. The CMS table (pktcMtaDevCmsTable) contains the parameters for the AP-exchange process between an MTA and a CMS. The CMS table is indexed by the Index Counter and contains the CMS FQDN (the conceptual column pktcMtaDevCmsFqdn). Each row contains the Kerberos realm name associated with each CMS FQDN. This allows for each CMS to exist in a different Kerberos realm.

The MTA MIB module also contains a group of scalar MIB objects in the server group (pktcMtaDevServer). These objects define various parameters for the AP-exchange process between an MTA and the Provisioning Server. These objects are:

- pktcMtaDevProvUnsolicitedKeyMaxTimeout,
- pktcMtaDevProvUnsolicitedKeyNomTimeout,

- pktcMtaDevProvUnsolicitedKeyMaxRetries, and
- pktcMtaDevProvSolicitedKeyTimeout.

### 3.6. Secure Software Download

E-MTAs are embedded with DOCSIS 1.1 cable modems. E-MTAs have their software upgraded by the Cable Modem according to the DOCSIS requirements.

Although E-MTAs have their software upgraded by the Cable Modem according to the DOCSIS requirements, S-MTAs implement a specific mechanism for Secure Software Download. This provides a means to verify the code upgrade using Code Verification Certificates and is modeled after the DOCSIS mechanism implemented in Cable Modems. This is the reason why the MTA MIB and the S-MTA compliance modules also rely on two MIB object groups:

- docsBpi2CodeDownloadGroup, defined in the IETF BPI Plus MIB module (DOCS-IETF-BPI2-MIB [RFC4131]).
- docsDevSoftwareGroupV2, defined in the IETF Cable Devicev2 MIB module (DOCS-CABLE-DEVICE-MIB [RFC4639]).

### 3.7. X.509 Certificates Dependencies

As specified in the PacketCable Security Specification [PKT-SP-SEC], E-MTAs must use the authentication mechanism based on the X.509 Public Key Infrastructure Certificates, as defined in RFC 3280 [RFC3280] and RFC 4630 [RFC4630].

The value of the pktcMtaDevRealmOrgName MIB object should contain the X.509 organization name attribute of the Telephony Service Provider certificate (OrganizationName). X.509 attributes are defined using UTF-8 string encoding [RFC3629, RFC3280, and RFC4630].

Note that UTF-8 encoded characters can be encoded as sequences of 1 to 6 octets, assuming that code points as high as 0x7fffffff might be used ([RFC3629]). Subsequent versions of Unicode and ISO 10646 have limited the upper bound to 0x10ffff ([RFC3629]). Consequently, the current version of UTF-8, defined in RFC 3629, does not require more than four octets to encode a valid code point.



#### 4. Definitions

The MIB module below makes references and citations to [RFC868], [RFC3280], [RFC4630], and [RFC3617].

PKTC-IETF-MTA-MIB DEFINITIONS ::= BEGIN

##### IMPORTS

```

    MODULE-IDENTITY,
    OBJECT-TYPE,
    OBJECT-IDENTITY,
    Unsigned32,
    Counter32,
    NOTIFICATION-TYPE,
    mib-2
        FROM SNMPv2-SMI                -- [RFC2578]
    TEXTUAL-CONVENTION,
    RowStatus,
    TruthValue
        FROM SNMPv2-TC                -- [RFC2579]
    OBJECT-GROUP,
    MODULE-COMPLIANCE,
    NOTIFICATION-GROUP
        FROM SNMPv2-CONF              -- [RFC2580]
    InetAddressType,
    InetAddress
        FROM INET-ADDRESS-MIB         -- [RFC4001]
    sysDescr
        FROM SNMPv2-MIB                -- [RFC3418]
    SnmpAdminString
        FROM SNMP-FRAMEWORK-MIB        -- [RFC3411]
    docsDevSoftwareGroupV2
        FROM DOCS-CABLE-DEVICE-MIB    -- [RFC4639]
    DocsX509ASN1DEREncodedCertificate,
    docsBpi2CodeDownloadGroup
        FROM DOCS-IETF-BPI2-MIB        -- [RFC4131]
    LongUtf8String
        FROM SYSAPPL-MIB               -- [RFC2287]
    ifPhysAddress
        FROM IF-MIB;                  -- [RFC2863]

pktcIetfMtaMib MODULE-IDENTITY
LAST-UPDATED "200609180000Z" -- September 18, 2006
ORGANIZATION "IETF IP over Cable Data Network Working Group"
CONTACT-INFO
    "Eugene Nechamkin
     Broadcom Corporation,
     200-13711 International Place,
```

Richmond, BC, V6V 2Z8  
 CANADA  
 Phone: +1 604 233 8500  
 Email: enechamkin@broadcom.com

Jean-Francois Mule  
 Cable Television Laboratories, Inc.  
 858 Coal Creek Circle  
 Louisville, CO 80027-9750  
 U.S.A.  
 Phone: +1 303 661 9100  
 Email: jf.mule@cablelabs.com

IETF IPCDN Working Group  
 General Discussion: [ipcdn@ietf.org](mailto:ipcdn@ietf.org)  
 Subscribe: <http://www.ietf.org/mailman/listinfo/ipcdn>  
 Archive: <ftp://ftp.ietf.org/ietf-mail-archive/ipcdn>  
 Co-Chair: Jean-Francois Mule, [jf.mule@cablelabs.com](mailto:jf.mule@cablelabs.com)  
 Co-Chair: Richard Woundy, [Richard\\_Woundy@cable.comcast.com](mailto:Richard_Woundy@cable.comcast.com)

#### DESCRIPTION

"This MIB module defines the basic management object for the Multimedia Terminal Adapter devices compliant with PacketCable and IP\_Cablecom requirements.

Copyright (C) The IETF Trust (2006). This version of this MIB module is part of RFC 4682; see the RFC itself for full legal notices."

REVISION "200609180000Z" -- September 18, 2006

#### DESCRIPTION

"Initial version, published as RFC 4682."

::= { mib-2 140 }

-- Textual Conventions

PktdMtaDevProvEncryptAlg ::= TEXTUAL-CONVENTION

STATUS current

#### DESCRIPTION

" This textual convention defines various types of the encryption algorithms used for the encryption of the MTA configuration file. The description of the encryption algorithm for each enumerated value is as follows:

'none(0)'	no encryption is used,
'des64CbcMode(1)'	DES 64-bit key in CBC mode,

```

        't3Des192CbcMode(2)' 3DES 192-bit key in CBC mode,
        'aes128CbcMode(3)'   AES 128-bit key in CBC mode,
        'aes256CbcMode(4)'   AES 256-bit key in CBC mode."
SYNTAX      INTEGER {
        none                (0),
        des64CbcMode        (1),
        t3Des192CbcMode     (2),
        aes128CbcMode        (3),
        aes256CbcMode        (4)
    }

```

```

-----
-- The MTA MIB module only supports a single Provisioning Server.
-----

```

```

pktcMtaNotification OBJECT IDENTIFIER ::= { pktcIetfMtaMib 0 }
pktcMtaMibObjects    OBJECT IDENTIFIER ::= { pktcIetfMtaMib 1 }
pktcMtaDevBase       OBJECT IDENTIFIER ::= { pktcMtaMibObjects 1 }
pktcMtaDevServer     OBJECT IDENTIFIER ::= { pktcMtaMibObjects 2 }
pktcMtaDevSecurity   OBJECT IDENTIFIER ::= { pktcMtaMibObjects 3 }
pktcMtaDevErrors     OBJECT IDENTIFIER ::= { pktcMtaMibObjects 4 }
pktcMtaConformance   OBJECT IDENTIFIER ::= { pktcIetfMtaMib 2 }

```

```

--
-- The following pktcMtaDevBase group describes the base MTA objects
--

```

pktcMtaDevResetNow OBJECT-TYPE

SYNTAX TruthValue

MAX-ACCESS read-write

STATUS current

DESCRIPTION

" This object controls the MTA software reset.

Reading this object always returns 'false'. Setting this object to 'true' causes the device to reset immediately and the following actions to occur:

1. All connections (if present) are flushed locally.
2. All current actions such as ringing immediately terminate.
3. Requests for signaling notifications, such as notification based on digit map recognition, are flushed.
4. All endpoints are disabled.
5. The provisioning flow is started at step MTA-1.

If a value is written into an instance of pktcMtaDevResetNow, the agent MUST NOT retain the supplied value across MTA re-initializations or reboots."

REFERENCE

```
    " PacketCable MTA Device Provisioning Specification."  
 ::= { pktcMtaDevBase 1 }
```

pktcMtaDevSerialNumber OBJECT-TYPE

SYNTAX SnmpAdminString

MAX-ACCESS read-only

STATUS current

DESCRIPTION

" This object specifies the manufacturer's serial number of this MTA. The value of this object MUST be identical to the value specified in DHCP option 43, sub-option 4. The list of sub-options for DHCP option 43 are defined in the PacketCable MTA Device Provisioning Specification."

REFERENCE

" PacketCable MTA Device Provisioning Specification."

```
 ::= { pktcMtaDevBase 2 }
```

pktcMtaDevSwCurrentVers OBJECT-TYPE

SYNTAX SnmpAdminString

MAX-ACCESS read-only

STATUS current

DESCRIPTION

" This object identifies the software version currently operating in the MTA. The MTA MUST return a string descriptive of the current software load. This object should use the syntax defined by the individual vendor to identify the software version. The data presented in this object MUST be identical to the software version information contained in the 'sysDescr' MIB object of the MTA. The value of this object MUST be identical to the value specified in DHCP option 43, sub-option 6. The list of sub-options for DHCP option 43 are defined in the PacketCable MTA Device Provisioning Specification."

REFERENCE

" PacketCable MTA Device Provisioning Specification."

```
 ::= { pktcMtaDevBase 3 }
```

pktcMtaDevFQDN OBJECT-TYPE

SYNTAX SnmpAdminString

MAX-ACCESS read-only

STATUS current

DESCRIPTION

" This object contains the Fully Qualified Domain Name for this MTA. The MTA FQDN is used to uniquely identify the device to the PacketCable back office elements."

```
::= { pktcMtaDevBase 4 }
```

```
pktcMtaDevEndPntCount      OBJECT-TYPE
```

```
    SYNTAX      Unsigned32 (1..255)
```

```
    MAX-ACCESS  read-only
```

```
    STATUS      current
```

```
    DESCRIPTION
```

```
        " This object contains the number of physical endpoints for
          this MTA."
```

```
::= { pktcMtaDevBase 5 }
```

```
pktcMtaDevEnabled          OBJECT-TYPE
```

```
    SYNTAX      TruthValue
```

```
    MAX-ACCESS  read-write
```

```
    STATUS      current
```

```
    DESCRIPTION
```

```
        " This object contains the MTA Admin Status of this device.
          If this object is set to 'true', the MTA is
          administratively enabled, and the MTA MUST be able to
          interact with the PacketCable entities, such as CMS,
          Provisioning Server, KDC, and other MTAs and MGs on all
          PacketCable interfaces.
```

```
          If this object is set to 'false', the MTA is
          administratively disabled, and the MTA MUST perform the
          following actions for all endpoints:
```

- Shut down all media sessions, if present.
- Shut down Network Control Signaling (NCS)
 signaling by following the Restart in
 Progress procedures in the PacketCable NCS
 specification.

```
          The MTA must execute all actions required to
          enable or disable the telephony services for all
          endpoints immediately upon receipt of an SNMP SET
          operation.
```

```
          Additionally, the MTA MUST maintain the SNMP Interface
          for management and also the SNMP Key management interface.
          Also, the MTA MUST NOT continue Kerberized key management
          with CMSes until this object is set to 'true'.
```

```
          Note: MTAs MUST renew the CMS Kerberos tickets according
          to the PacketCable Security or IPCablecom Specification.
          If a value is written into an instance of
          pktcMtaDevEnabled, the agent MUST NOT retain the supplied
          value across MTA re-initializations or reboots."
```

```
REFERENCE
```

```
    " PacketCable MTA Device Provisioning Specification;
      PacketCable Security Specification;
      PacketCable Network-Based Call Signaling Protocol
```

```

        Specification."
 ::= { pktcMtaDevBase 6 }

pktcMtaDevTypeIdentifier      OBJECT-TYPE
    SYNTAX      SnmpAdminString
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        " This object provides the MTA device type identifier.  The
          value of this object must be a copy of the DHCP option 60
          value exchanged between the MTA and the DHCP server.  The
          DHCP option 60 value contains an ASCII-encoded string
          identifying capabilities of the MTA as defined in the
          PacketCable MTA Device Provisioning Specification."
    REFERENCE
        " RFC 2132, DHCP Options and BOOTP Vendor Extensions;
          PacketCable MTA Device Provisioning Specification."
 ::= { pktcMtaDevBase 7 }

pktcMtaDevProvisioningState    OBJECT-TYPE
    SYNTAX      INTEGER {
        pass                      (1),
        inProgress                (2),
        failConfigFileError       (3),
        passWithWarnings          (4),
        passWithIncompleteParsing (5),
        failureInternalError       (6),
        failureOtherReason        (7)
    }
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        " This object indicates the completion state of the MTA
          device provisioning process.

          pass:
          If the configuration file could be parsed successfully
          and the MTA is able to reflect the same in its
          MIB, the MTA MUST return the value 'pass'.

          inProgress:
          If the MTA is in the process of being provisioned,
          the MTA MUST return the value 'inProgress'.

          failConfigFileError:
          If the configuration file was in error due to incorrect
          values in the mandatory parameters, the MTA MUST reject
          the configuration file, and the MTA MUST return the value

```

'failConfigFileError'.

passWithWarnings:

If the configuration file had proper values for all the mandatory parameters but has errors in any of the optional parameters (this includes any vendor-specific Object Identifiers (OIDs) that are incorrect or not known to the MTA), the MTA MUST return the value 'passWithWarnings'.

passWithIncompleteParsing:

If the configuration file is valid but the MTA cannot reflect the same in its configuration (for example, too many entries caused memory exhaustion), it must accept the CMS configuration entries related, and the MTA MUST return the value 'passWithIncompleteParsing'.

failureInternalError:

If the configuration file cannot be parsed due to an Internal error, the MTA MUST return the value 'failureInternalError'.

failureOtherReason:

If the MTA cannot accept the configuration file for any other reason than the ones stated above, the MTA MUST return the value 'failureOtherReason'.

When a final SNMP INFORM is sent as part of Step 25 of the MTA Provisioning process, this parameter is also included in the final INFORM message."

#### REFERENCE

" PacketCable MTA Device Provisioning Specification."  
 ::= { pktcMtaDevBase 8 }

pktcMtaDevHttpAccess OBJECT-TYPE

SYNTAX TruthValue

MAX-ACCESS read-only

STATUS current

DESCRIPTION

" This object indicates whether the HTTP protocol is supported for the MTA configuration file transfer."

::= { pktcMtaDevBase 9 }

pktcMtaDevProvisioningTimer OBJECT-TYPE

SYNTAX Unsigned32 (0..30)

UNITS "minutes"

MAX-ACCESS read-write

STATUS current

## DESCRIPTION

" This object defines the time interval for the provisioning flow to complete. The MTA MUST finish all provisioning operations starting from the moment when an MTA receives its DHCP ACK and ending at the moment when the MTA downloads its configuration file (e.g., MTA5 to MTA23) within the period of time set by this object. Failure to comply with this condition constitutes a provisioning flow failure. If the object is set to 0, the MTA MUST ignore the provisioning timer condition. If a value is written into an instance of pktcMtaDevProvisioningTimer, the agent MUST NOT retain the supplied value across MTA re-initializations or reboots."

## REFERENCE

" PacketCable MTA Device Provisioning Specification."

DEFVAL {10}

::= {pktcMtaDevBase 10}

pktcMtaDevProvisioningCounter OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

## DESCRIPTION

"This object counts the number of times the provisioning cycle has looped through step MTA-1."

::= {pktcMtaDevBase 11}

pktcMtaDevErrorOidsTable OBJECT-TYPE

SYNTAX SEQUENCE OF PktcMtaDevErrorOidsEntry

MAX-ACCESS not-accessible

STATUS current

## DESCRIPTION

" This table contains the list of configuration errors or warnings the MTA encountered when parsing the configuration file it received from the Provisioning Server.

For each error, an entry is created in this table, containing the configuration parameters the MTA rejected and the associated reason (e.g., wrong or unknown OID, inappropriate object values). If the MTA did not report a provisioning state of 'pass(1)' in the pktcMtaDevProvisioningState object, this table MUST be populated for each error or warning instance. Even if different parameters share the same error type (e.g., all realm name configuration parameters are invalid), all observed errors or warnings must be reported as different instances. Errors are placed into the table in no particular order. The table MUST be cleared each time



the MTA reboots."

#### REFERENCE

" PacketCable MTA Device Provisioning Specification."

::= {pktcMtaDevBase 12 }

#### pktcMtaDevErrorOidsEntry OBJECT-TYPE

SYNTAX PktcMtaDevErrorOidsEntry

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION

" This entry contains the necessary information the MTA MUST attempt to provide in case of configuration file errors or warnings."

INDEX { pktcMtaDevErrorOidIndex }

::= {pktcMtaDevErrorOidsTable 1}

PktcMtaDevErrorOidsEntry ::= SEQUENCE {

pktcMtaDevErrorOidIndex Unsigned32,

pktcMtaDevErrorOid SnmpAdminString,

pktcMtaDevErrorValue SnmpAdminString,

pktcMtaDevErrorReason SnmpAdminString

}

#### pktcMtaDevErrorOidIndex OBJECT-TYPE

SYNTAX Unsigned32 (1..1024)

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION

" This object is the index of the MTA configuration error table. It is an integer value that starts at value '1' and is incremented for each encountered configuration file error or warning.

The maximum number of errors or warnings that can be recorded in the pktcMtaDevErrorOidsTable is set to 1024 as a configuration file is usually validated by operators before deployment. Given the possible number of configuration parameter assignments in the MTA configuration file, 1024 is perceived as a sufficient limit even with future extensions.

If the number of the errors in the configuration file exceeds 1024, all errors beyond the 1024th one MUST be ignored and not be reflected in the pktcMtaDevErrorOidsTable."

::= {pktcMtaDevErrorOidsEntry 1}

**pktcMtaDevErrorOid OBJECT-TYPE**

SYNTAX SnmpAdminString

MAX-ACCESS read-only

STATUS current

**DESCRIPTION**

" This object contains a human readable representation (character string) of the OID corresponding to the configuration file parameter that caused the particular error.

For example, if the value of the pktcMtaDevEnabled object in the configuration file caused an error, then this object instance will contain the human-readable string of '1.3.6.1.2.1.140.1.1.6.0'.

If the MTA generated an error because it was not able to recognize a particular OID, then this object instance would contain an empty value (zero-length string).

For example, if the value of an OID in the configuration file was interpreted by the MTA as being 1.2.3.4.5, and if the MTA was not able to recognize this OID as a valid one, this object instance will contain a zero-length string.

If the number of errors in the configuration file exceeds 1024, then for all subsequent errors, the pktcMtaDevErrorOid of the table's 1024th entry MUST contain a human-readable representation of the pktcMtaDevErrorsTooManyErrors object; i.e., the string '1.3.6.1.2.1.140.1.1.4.1.0'.

Note that the syntax of this object is SnmpAdminString instead of OBJECT IDENTIFIER because the object value may not be a valid OID due to human or configuration tool encoding errors."

::= {pktcMtaDevErrorOidsEntry 2}

**pktcMtaDevErrorValue OBJECT-TYPE**

SYNTAX SnmpAdminString

MAX-ACCESS read-only

STATUS current

**DESCRIPTION**

" This object contains the value of the OID corresponding to the configuration file parameter that caused the error.

If the MTA cannot recognize the OID of the configuration parameter causing the error, then this object instance contains the OID itself as interpreted by the MTA in human-readable representation.

If the MTA can recognize the OID but generate an error due to a wrong value of the parameter, then the object

instance contains the erroneous value of the parameter as read from the configuration file.

In both cases, the value of this object must be represented in human-readable form as a character string. For example, if the value of the pktcMtaDevEnabled object in the configuration file was 3 (invalid value), then the pktcMtaDevErrorValue object instance will contain the human-readable (string) representation of value '3'. Similarly, if the OID in the configuration file has been interpreted by the MTA as being 1.2.3.4.5 and the MTA cannot recognize this OID as a valid one, then this pktcMtaDevErrorValue object instance will contain human readable (string) representation of value '1.2.3.4.5'.

If the number of errors in the configuration file exceeds 1024, then for all subsequent errors, the pktcMtaDevErrorValue of the table's 1024th entry MUST contain a human-readable representation of the pktcMtaDevErrorsTooManyErrors object; i.e., the string '1.3.6.1.2.1.140.1.1.4.1.0'."

```
::= {pktcMtaDevErrorOidsEntry 3}
```

```
pktcMtaDevErrorReason OBJECT-TYPE
```

```
SYNTAX      SnmpAdminString
```

```
MAX-ACCESS  read-only
```

```
STATUS      current
```

```
DESCRIPTION
```

```
    " This object indicates the reason for the error or warning,
      as per the MTA's interpretation, in human-readable form.
      For example:
```

```
    'VALUE NOT IN RANGE', 'VALUE DOES NOT MATCH TYPE',
    'UNSUPPORTED VALUE', 'LAST 4 BITS MUST BE SET TO ZERO',
    'OUT OF MEMORY - CANNOT STORE'.
```

```
    This object may also contain vendor specific errors for
    private vendor OIDs and any proprietary error codes or
    messages that can help diagnose configuration errors.
```

```
    If the number of errors in the configuration file exceeds
    1024, then for all subsequent errors, the
    pktcMtaDevErrorReason of the table's 1024th entry MUST
    contain a human-readable string indicating the reason
    for an error; for example,
    'Too many errors in the configuration file'."
```

```
::= {pktcMtaDevErrorOidsEntry 4}
```

```
--
```

```
-- The following group describes server access and parameters used
```

```
-- for the initial MTA provisioning and bootstrapping phases.
--
```

pktcMtaDevDhcpServerAddressType OBJECT-TYPE

SYNTAX InetAddressType

MAX-ACCESS read-only

STATUS current

DESCRIPTION

" This object contains the Internet address type for the PacketCable DHCP servers specified in MTA MIB."

DEFVAL { ipv4 }

::= { pktcMtaDevServer 1 }

pktcMtaDevServerDhcp1 OBJECT-TYPE

SYNTAX InetAddress

MAX-ACCESS read-only

STATUS current

DESCRIPTION

" This object contains the Internet Address of the primary DHCP server the MTA uses during provisioning. The type of this address is determined by the value of the pktcMtaDevDhcpServerAddressType object. When the latter has the value 'ipv4(1)', this object contains the IP address of the primary DHCP server. It is provided by the CM to the MTA via the DHCP option code 122, sub-option 1, as defined in RFC 3495.

The behavior of this object when the value of pktcMtaDevDhcpServerAddressType is other than 'ipv4(1)' is not presently specified, but it may be specified in future versions of this MIB module.

If this object is of value

0.0.0.0, the MTA MUST stop all provisioning attempts, as well as all other activities.

If this object is of value 255.255.255.255, it means that there was no preference given for the primary DHCP server, and, the MTA must follow the logic of RFC2131, and the value of DHCP option 122, sub-option 2, must be ignored."

REFERENCE

" PacketCable MTA Device Provisioning Specification;  
RFC 2131, Dynamic Host Configuration Protocol;  
RFC 3495, DHCP Option for CableLabs Client Configuration."

::= { pktcMtaDevServer 2 }

pktcMtaDevServerDhcp2 OBJECT-TYPE

SYNTAX InetAddress

MAX-ACCESS read-only

STATUS current

DESCRIPTION

" This object contains the Internet Address of the secondary DHCP server the MTA uses during provisioning. The type of this address is determined by the value of the pktcMtaDevDhcpServerAddressType object. When the latter has the value 'ipv4(1)', this object contains the IP address of the secondary DHCP server. It is provided by the CM to the MTA via the DHCP option code 122, sub-option 2, as defined in RFC 3495.

The behavior of this object when the value of pktcMtaDevDhcpServerAddressType is other than 'ipv4(1)' is not presently specified, but it may be specified in future versions of this MIB module. If there was no secondary DHCP server provided in DHCP Option 122, sub-option 2, this object must return the value 0.0.0.0."

REFERENCE

" PacketCable MTA Device Provisioning Specification; RFC 3495, DHCP Option for CableLabs Client Configuration."  
 ::= { pktcMtaDevServer 3 }

pktcMtaDevDnsServerAddressType OBJECT-TYPE

SYNTAX InetAddressType

MAX-ACCESS read-only

STATUS current

DESCRIPTION

" This object contains the Internet address type for the PacketCable DNS servers specified in MTA MIB."

DEFVAL { ipv4 }

::= { pktcMtaDevServer 4 }

pktcMtaDevServerDns1 OBJECT-TYPE

SYNTAX InetAddress

MAX-ACCESS read-write

STATUS current

DESCRIPTION

" This object contains the IP Address of the primary DNS server to be used by the MTA. The type of this address is determined by the value of the pktcMtaDevDnsServerAddressType object. When the latter has the value 'ipv4(1)', this object contains the IP address of the primary DNS server. As defined in RFC 2132, PacketCable-compliant MTAs receive the IP addresses of the DNS Servers in DHCP option 6. The behavior of this object when the value of pktcMtaDevDnsServerAddressType is other than 'ipv4(1)'

is not presently specified, but it may be specified in future versions of this MIB module.  
 If a value is written into an instance of pktcMtaDevServerDns1, the agent MUST NOT retain the supplied value across MTA re-initializations or reboots."

## REFERENCE

" PacketCable MTA Device Provisioning Specification;  
 RFC 2132, DHCP Options and BOOTP Vendor Extensions."  
 ::= { pktcMtaDevServer 5 }

## pktcMtaDevServerDns2 OBJECT-TYPE

SYNTAX InetAddress

MAX-ACCESS read-write

STATUS current

## DESCRIPTION

" This object contains the IP Address of the secondary DNS server to be used by the MTA. The type of this address is determined by the value of the pktcMtaDevDnsServerAddressType object.  
 When the latter has the value 'ipv4(1)', this object contains the IP address of the secondary DNS server. As defined in RFC 2132, PacketCable-compliant MTAs receive the IP addresses of the DNS Servers in DHCP option 6.  
 The behavior of this object when the value of pktcMtaDevDnsServerAddressType is other than 'ipv4(1)' is not presently specified, but it may be specified in future versions of this MIB module.  
 If a value is written into an instance of pktcMtaDevServerDns2, the agent MUST NOT retain the supplied value across MTA re-initializations or reboots."

## REFERENCE

" PacketCable MTA Device Provisioning Specification;  
 RFC 2132, DHCP Options and BOOTP Vendor Extensions."  
 ::= { pktcMtaDevServer 6 }

## pktcMtaDevTimeServerAddressType OBJECT-TYPE

SYNTAX InetAddressType

MAX-ACCESS read-only

STATUS current

## DESCRIPTION

" This object contains the Internet address type for the PacketCable Time servers specified in MTA MIB."

DEFVAL { ipv4 }

::= { pktcMtaDevServer 7 }

## pktcMtaDevTimeServer OBJECT-TYPE

SYNTAX InetAddress

MAX-ACCESS read-write  
STATUS current  
DESCRIPTION

" This object contains the Internet Address of the Time Server used by an S-MTA for Time Synchronization. The type of this address is determined by the value of the pktcMtaDevTimeServerAddressType object.  
When the latter has the value 'ipv4(1)', this object contains the IP address of the Time Server used for Time Synchronization.  
In the case of an S-MTA, this object must be populated with a value other than 0.0.0.0 as obtained from DHCP option 4. The protocol by which the time of day MUST be retrieved is defined in RFC 868.  
In the case of an E-MTA, this object must contain a value of 0.0.0.0 if the address type is 'ipv4(1)' since an E-MTA does not use the Time Protocol for time synchronization (an E-MTA uses the time retrieved by the DOCSIS cable modem).  
The behavior of this object when the value of pktcMtaDevTimeServerAddressType is other than 'ipv4(1)' is not presently specified, but it may be specified in future versions of this MIB module.  
If a value is written into an instance of pktcMtaDevTimeServer, the agent MUST NOT retain the supplied value across MTA re-initializations or reboots."

REFERENCE

" RFC 868, Time Protocol;  
RFC 2131, Dynamic Host Configuration Protocol;  
RFC 2132, DHCP Options and BOOTP Vendor Extensions."

::= { pktcMtaDevServer 8 }

pktcMtaDevConfigFile OBJECT-TYPE

SYNTAX SnmpAdminString  
MAX-ACCESS read-write  
STATUS current  
DESCRIPTION

" This object specifies the MTA device configuration file information, including the access method, the server name, and the configuration file name. The value of this object is the Uniform Resource Locator (URL) of the configuration file for TFTP or HTTP download.  
If this object value is a TFTP URL, it must be formatted as defined in RFC 3617.  
If this object value is an HTTP URL, it must be formatted as defined in RFC 2616.  
If the MTA SNMP Enrollment mechanism is used, then the MTA must download the file provided by the Provisioning Server

during provisioning via an SNMP SET on this object. If the MTA SNMP Enrollment mechanism is not used, this object MUST contain the URL value corresponding to the 'siaddr' and 'file' fields received in the DHCP ACK to locate the configuration file: the 'siaddr' and 'file' fields represent the host and file of the TFTP URL, respectively. In this case, the MTA MUST return an 'inconsistentValue' error in response to SNMP SET operations.

The MTA MUST return a zero-length string if the server address (host part of the URL) is unknown.

If a value is written into an instance of pktcMtaDevConfigFile, the agent MUST NOT retain the supplied value across MTA re-initializations or reboots."

#### REFERENCE

" PacketCable MTA Device Provisioning Specification;  
RFC 3617, URI Scheme for TFTP; RFC 2616, HTTP 1.1"  
::= { pktcMtaDevServer 9 }

#### pktcMtaDevSnmpEntity OBJECT-TYPE

SYNTAX SnmpAdminString

MAX-ACCESS read-only

STATUS current

#### DESCRIPTION

" This object contains the FQDN of the SNMP entity of the Provisioning Server. When the MTA SNMP Enrollment Mechanism is used, this object represents the server that the MTA communicates with, that it receives the configuration file URL from, and that it sends the enrollment notification to. The SNMP entity is also the destination entity for all the provisioning notifications. It may be used for post-provisioning SNMP operations. During the provisioning phase, this SNMP entity FQDN is supplied to the MTA via DHCP option 122, sub-option 3, as defined in RFC 3495. The MTA must resolve the FQDN value before its very first network interaction with the SNMP entity during the provisioning phase."

#### REFERENCE

" PacketCable MTA Device Provisioning Specification;  
RFC 3495, DHCP Option for CableLabs Client Configuration."  
::= { pktcMtaDevServer 10 }

#### pktcMtaDevProvConfigHash OBJECT-TYPE

SYNTAX OCTET STRING (SIZE(20))

MAX-ACCESS read-write

STATUS current



## DESCRIPTION

" This object contains the hash value of the contents of the configuration file.

The authentication algorithm is Secure Hashing Algorithm 1 (SHA-1), and the length is 160 bits. The hash calculation MUST follow the requirements defined in the PacketCable Security Specification. When the MTA SNMP Enrollment mechanism is used, this hash value is calculated and sent to the MTA prior to sending the config file. This object value is then provided by the Provisioning server via an SNMP SET operation. When the MTA SNMP Enrollment mechanism is not in use, the hash value is provided in the configuration file itself, and it is also calculated by the MTA. This object value MUST represent the hash value calculated by the MTA. When the MTA SNMP Enrollment mechanism is not in use, the MTA must reject all SNMP SET operations on this object and return an 'inconsistentValue' error.

If a value is written into an instance of pktcMtaDevProvConfigHash, the agent MUST NOT retain the supplied value across MTA re-initializations or reboots."

## REFERENCE

" PacketCable MTA Device Provisioning Specification;  
PacketCable Security Specification."

::= { pktcMtaDevServer 11 }

pktcMtaDevProvConfigKey OBJECT-TYPE

SYNTAX OCTET STRING (SIZE(32))

MAX-ACCESS read-write

STATUS current

## DESCRIPTION

" This object contains the key used to encrypt/decrypt the configuration file when secure SNMPv3 provisioning is used.

The value of this object is provided along with the configuration file information (pktcMtaDevConfigFile) and hash (pktcMtaDevProvConfigHash) by the Provisioning Server via SNMP SET once the configuration file has been created, as defined by the PacketCable Security specification.

The privacy algorithm is defined by the pktcMtaDevProvConfigEncryptAlg MIB object. The MTA requirements related to the privacy algorithm are defined in the PacketCable Security Specification.

If this object is set at any other provisioning step than that allowed by the PacketCable MTA Device

Provisioning Specification, the MTA SHOULD return an 'inconsistentValue' error.  
 This object must not be used in non secure provisioning mode. In non-secure provisioning modes, the MTA SHOULD return an 'inconsistentValue' in response to SNMP SET operations, and the MTA SHOULD return a zero-length string in response to SNMP GET operations.  
 If a value is written into an instance of pktcMtaDevProvConfigKey, the agent MUST NOT retain the supplied value across MTA re-initializations or reboots."

## REFERENCE

" PacketCable MTA Device Provisioning Specification;  
 PacketCable Security Specification."

::= { pktcMtaDevServer 12 }

## pktcMtaDevProvConfigEncryptAlg OBJECT-TYPE

SYNTAX PktcMtaDevProvEncryptAlg

MAX-ACCESS read-write

STATUS current

## DESCRIPTION

" This object defines the encryption algorithm used for privacy protection of the MTA Configuration File content."

DEFVAL { des64CbcMode }

::= { pktcMtaDevServer 13 }

## pktcMtaDevProvSolicitedKeyTimeout OBJECT-TYPE

SYNTAX Unsigned32 (0..180)

UNITS "seconds"

MAX-ACCESS read-write

STATUS current

## DESCRIPTION

" This object defines a Kerberos Key Management timer on the MTA. It is the time period during which the MTA saves the nonce and Server Kerberos Principal Identifier to match an AP Request and its associated AP Reply response from the Provisioning Server.

After the timeout has been exceeded, the client discards this (nonce, Server Kerberos Principal Identifier) pair, after which it will no longer accept a matching AP Reply. This timer only applies when the Provisioning Server initiated key management for SNMPv3 (with a Wake Up message).

If this object is set to a zero value, the MTA MUST return an 'inconsistentValue' in response to SNMP SET operations. This object should not be used in non-secure provisioning modes. In non-secure provisioning modes, the MTA MUST return an 'inconsistentValue' in response to SNMP SET operations, and the MTA MUST return a zero value in

```

        response to SNMP GET operations.
        If a value is written into an instance of
        pktcMtaDevProvSolicitedKeyTimeout, the agent MUST NOT
        retain the supplied value across MTA re-initializations
        or reboots."
DEFVAL { 3 }
 ::= { pktcMtaDevServer 14 }

-----
--
-- Unsolicited key updates are retransmitted according to an
-- exponential back-off mechanism using two timers and a maximum
-- retry counter for AS replies.
-- The initial retransmission timer value is the nominal timer
-- value (pktcMtaDevProvUnsolicitedKeyNomTimeout). The
-- retransmissions occur with an exponentially increasing interval
-- that caps at the maximum timeout value
-- (pktcMtaDevProvUnsolicitedKeyMaxTimeout).
-- Retransmissions stop when the maximum retry counter is reached
-- (pktcMtaDevProvUnsolicitedKeyMaxRetries).
-- For example, with values of 3 seconds for the nominal
-- timer, 100 seconds for the maximum timeout, and 8 retries max,
-- and with an exponential value of 2, this results in
-- retransmission intervals will be 3 s, 6 s, 12 s, 24 s, 48 s,
-- 96 s, 100 s, and 100 s;
-- retransmissions then stop because the maximum number of
-- retries (8) has been reached.
--
-----
--
-- Timeouts for unsolicited key management updates are only
-- pertinent before the first SNMPv3 message is sent between the
-- MTA and the Provisioning Server and before the configuration
-- file is loaded.
--
-----

pktcMtaDevProvUnsolicitedKeyMaxTimeout  OBJECT-TYPE
    SYNTAX      Unsigned32 (0..600)
    UNITS        "seconds"
    MAX-ACCESS   read-only
    STATUS       current
    DESCRIPTION
        " This object defines the timeout value that applies to
          an MTA-initiated AP-REQ/REP key management exchange with
          the Provisioning Server in SNMPv3 provisioning.
          It is the maximum timeout value, and it may not be exceeded
          in the exponential back-off algorithm.  If the DHCP option

```

code 122, sub-option 5, is provided to the MTA, it overwrites this value.

In non-secure provisioning modes, the MTA MUST return a zero value in response to SNMP GET operations."

REFERENCE

" PacketCable Security Specification."

DEFVAL {600}

::= { pktcMtaDevServer 15 }

pktcMtaDevProvUnsolicitedKeyNomTimeout OBJECT-TYPE

SYNTAX Unsigned32 (0..600)

UNITS "seconds"

MAX-ACCESS read-only

STATUS current

DESCRIPTION

" This object defines the starting value of the timeout for the AP-REQ/REP Backoff and Retry mechanism with exponential timeout in SNMPv3 provisioning. If the DHCP option code 122, sub-option 5, is provided the MTA, it overwrites this value. In non-secure provisioning modes, the MTA MUST return a zero value in response to SNMP GET operations."

REFERENCE

" PacketCable Security Specification."

DEFVAL {3}

::= { pktcMtaDevServer 16 }

pktcMtaDevProvUnsolicitedKeyMaxRetries OBJECT-TYPE

SYNTAX Unsigned32 (0..32)

MAX-ACCESS read-only

STATUS current

DESCRIPTION

" This object contains a retry counter that applies to an MTA-initiated AP-REQ/REP key management exchange with the Provisioning Server in secure SNMPv3 provisioning. It is the maximum number of retries before the MTA stops attempting to establish a Security Association with Provisioning Server. If the DHCP option code 122, sub-option 5, is provided to the MTA, it overwrites this value. If this object is set to a zero value, the MTA MUST return an 'inconsistentValue' in response to SNMP SET operations. In non-secure provisioning modes, the MTA MUST return a zero value in response to SNMP GET operations."

REFERENCE

```

    " PacketCable Security Specification."
    DEFVAL {8}
    ::= { pktcMtaDevServer 17 }

```

```
pktcMtaDevProvKerbRealmName OBJECT-TYPE
```

```
SYNTAX      SnmpAdminString (SIZE(1..255))
```

```
MAX-ACCESS  read-only
```

```
STATUS      current
```

```
DESCRIPTION
```

```

    " This object contains the name of the associated
    provisioning Kerberos realm acquired during the MTA4
    provisioning step (DHCP Ack) for SNMPv3 provisioning.
    The uppercase ASCII representation of the associated
    Kerberos realm name MUST be used by both the Manager (SNMP
    entity) and the MTA.
    The Kerberos realm name for the Provisioning Server is
    supplied to the MTA via DHCP option code 122, sub-option 6,
    as defined in RFC 3495. In secure SNMP provisioning mode,
    the value of the Kerberos realm name for the Provisioning
    Server supplied in the MTA configuration file must match
    the value supplied in the DHCP option code 122,
    sub-option 6. Otherwise, the value of this object must
    contain the value supplied in DHCP Option 122,
    sub-option 6."

```

```
REFERENCE
```

```

    " PacketCable MTA Device Provisioning Specification;
    RFC 3495, DHCP Option for CableLabs Client Configuration."

```

```
 ::= { pktcMtaDevServer 18 }
```

```
pktcMtaDevProvState OBJECT-TYPE
```

```

SYNTAX      INTEGER {
                operational                (1),
                waitingForSnmpSetInfo      (2),
                waitingForTftpAddrResponse (3),
                waitingForConfigFile       (4)
            }

```

```
MAX-ACCESS  read-only
```

```
STATUS      current
```

```
DESCRIPTION
```

```

    " This object defines the MTA provisioning state.
    If the state is:

```

```

        'operational(1)', the device has completed the loading
        and processing of the initialization parameters.

```

```

        'waitingForSnmpSetInfo(2)', the device is waiting on
        its configuration file download access information.
        Note that this state is only reported when the MTA

```

SNMP enrollment mechanism is used.

'waitingForTftpAddrResponse(3)', the device has sent a DNS request to resolve the server providing the configuration file, and it is awaiting for a response. Note that this state is only reported when the MTA SNMP enrollment mechanism is used.

'waitingForConfigFile(4)', the device has sent a request via TFTP or HTTP for the download of its configuration file, and it is awaiting for a response or the file download is in progress."

#### REFERENCE

" PacketCable MTA Device Provisioning Specification,  
PacketCable Security Specification."

::= { pktcMtaDevServer 19 }

--

-- The following object group describes the security objects.

--

#### pktcMtaDevManufacturerCertificate OBJECT-TYPE

SYNTAX DocsX509ASN1DEREncodedCertificate

MAX-ACCESS read-only

STATUS current

#### DESCRIPTION

" This object contains the MTA Manufacturer Certificate. The object value must be the ASN.1 DER encoding of the MTA manufacturer's X.509 public key certificate. The MTA Manufacturer Certificate is issued to each MTA manufacturer and is installed into each MTA at the time of manufacture or with a secure code download. The specific requirements related to this certificate are defined in the PacketCable or IPCablecom Security specifications."

#### REFERENCE

" PacketCable Security Specification."

::= {pktcMtaDevSecurity 1}

#### pktcMtaDevCertificate OBJECT-TYPE

SYNTAX DocsX509ASN1DEREncodedCertificate

MAX-ACCESS read-only

STATUS current

#### DESCRIPTION

" This object contains the MTA Device Certificate. The object value must be the ASN.1 DER encoding of the MTA's X.509 public-key certificate issued by the manufacturer and installed into the MTA at the time of

manufacture or with a secure code download.  
 This certificate contains the MTA MAC address. The  
 specific requirements related to this certificate are  
 defined in the PacketCable or IPCablecom Security  
 specifications."

## REFERENCE

" PacketCable Security Specification."  
 ::= { pktcMtaDevSecurity 2 }

## pktcMtaDevCorrelationId OBJECT-TYPE

SYNTAX Unsigned32

MAX-ACCESS read-only

STATUS current

## DESCRIPTION

" This object contains a correlation ID, an arbitrary value  
 generated by the MTA that will be exchanged as part of the  
 device capability data to the Provisioning Application.  
 This random value is used as an identifier to correlate  
 related events in the MTA provisioning sequence.  
 This value is intended for use only during the MTA  
 initialization and configuration file download."

## REFERENCE

" PacketCable MTA Device Provisioning Specification."  
 ::= { pktcMtaDevSecurity 3 }

## pktcMtaDevTelephonyRootCertificate OBJECT-TYPE

SYNTAX DocsX509ASN1DEREncodedCertificate

MAX-ACCESS read-only

STATUS current

## DESCRIPTION

" This object contains the telephony Service Provider Root  
 certificate. The object value is the ASN.1 DER encoding of  
 the IP Telephony Service Provider Root X.509 public key  
 certificate. This certification is stored in the MTA  
 non-volatile memory and can be updated with a secure code  
 download. This certificate is used to validate the initial  
 AS Reply received by the MTA from the Key Distribution  
 Center (KDC) during the MTA initialization. The specific  
 requirements related to this certificate are defined in  
 the PacketCable or IPCablecom Security specifications."

## REFERENCE

" PacketCable Security Specification."  
 ::= { pktcMtaDevSecurity 4 }

-----  
 --  
 -- Informative Procedures for Setting up Security Associations  
 --

```
-- A Security Association may be set up either via configuration or
-- via NCS signaling.
--
-- I. Security association setup via configuration.
--
-- The realm must be configured first. Associated with the realm
-- is a KDC. The realm table (pktcMtaDevRealmTable) indicates
-- information about the realm (e.g., name, organization name) and
-- parameters associated with KDC communications (e.g., grace
-- periods, AS Request/AS Reply adaptive back-off parameters).
--
-- Once the realm is established, one or more CMS(es) may be
-- defined in the realm. Associated with each CMS
-- entry in the pktcMtaDevCmsTable is an explicit reference
-- to a Realm via the realm name (pktcMtaDevCmsKerbRealmName),
-- the FQDN of the CMS, and parameters associated with IPsec
-- key management with the CMS (e.g., clock skew, AP Request/
-- AP Reply adaptive back-off parameters).
--
-- II. Security association setup via NCS signaling.
--
-- The procedure of establishing the Security Associations
-- for NCS signaling is described in the PacketCable Security
-- specification.
-- It involves the analysis of the pktcNcsEndPntConfigTable row
-- for the corresponding endpoint number and the correlation of
-- the CMS FQDN from this row with the CMS Table and
-- consequently, with the Realm Table. Both of these tables
-- are defined below. The pktcNcsEndPntConfigTable is defined in
-- the IP over Cable Data Network (IPCDN)
-- NCS Signaling MIB [NCSSIGMIB].
--
-- III. When the MTA receives wake-up or re-key messages from a
-- CMS, it performs key management based on the corresponding
-- entry in the CMS table. If the matching CMS entry does not
-- exist, it must ignore the wake-up or re-key messages.
--
--=====
--=====
--
-- pktcMtaDevRealmTable
--
-- The pktcMtaDevRealmTable shows the KDC realms. The table is
-- indexed with pktcMtaDevRealmIndex. The Realm Table contains the
-- pktcMtaDevRealmName in conjunction with any server that needs
-- a Security Association with the MTA. Uppercase must be used
-- to compare the pktcMtaDevRealmName content.
```



-----

pktcMtaDevRealmAvailSlot OBJECT-TYPE

SYNTAX Unsigned32 (0..64)

MAX-ACCESS read-only

STATUS current

DESCRIPTION

" This object contains the index number of the first available entry in the realm table (pktcMtaDevRealmTable). If all the entries in the realm table have been assigned, this object contains the value of zero. A management station should create new entries in the realm table, using the following procedure:

First, issue a management protocol retrieval operation to determine the value of the first available index in the realm table (pktcMtaDevRealmAvailSlot).

Second, issue a management protocol SET operation to create an instance of the pktcMtaDevRealmStatus object by setting its value to 'createAndWait(5)'.

Third, if the SET operation succeeded, continue modifying the object instances corresponding to the newly created conceptual row, without fear of collision with other management stations. When all necessary conceptual columns of the row are properly populated (via SET operations or default values), the management station may SET the pktcMtaDevRealmStatus object to 'active(1)'."

::= { pktcMtaDevSecurity 5 }

pktcMtaDevRealmTable OBJECT-TYPE

SYNTAX SEQUENCE OF PktcMtaDevRealmEntry

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION

" This object contains the realm table. The CMS table (pktcMtaDevCmsTable) and the realm table (pktcMtaDevRealmTable) are used for managing the MTA-CMS Security Associations. The realm table defines the Kerberos realms for the Application Servers (CMSes and the Provisioning Server)."

::= { pktcMtaDevSecurity 6 }

pktcMtaDevRealmEntry OBJECT-TYPE

SYNTAX PktcMtaDevRealmEntry

MAX-ACCESS not-accessible

STATUS current

## DESCRIPTION

" This table entry object lists the MTA security parameters for a single Kerberos realm. The conceptual rows MUST NOT persist across MTA reboots."

```
INDEX { pktcMtaDevRealmIndex }
 ::= { pktcMtaDevRealmTable 1 }
```

```
PktcMtaDevRealmEntry ::= SEQUENCE {
    pktcMtaDevRealmIndex          Unsigned32,
    pktcMtaDevRealmName          SnmpAdminString,
    pktcMtaDevRealmPkinitGracePeriod Unsigned32,
    pktcMtaDevRealmTgsGracePeriod Unsigned32,
    pktcMtaDevRealmOrgName       LongUtf8String,
    pktcMtaDevRealmUnsolicitedKeyMaxTimeout Unsigned32,
    pktcMtaDevRealmUnsolicitedKeyNomTimeout Unsigned32,
    pktcMtaDevRealmUnsolicitedKeyMaxRetries Unsigned32,
    pktcMtaDevRealmStatus        RowStatus
}
```

```
pktcMtaDevRealmIndex OBJECT-TYPE
    SYNTAX      Unsigned32 (1..64)
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        " This object defines the realm table index."
    ::= { pktcMtaDevRealmEntry 1 }
```

```
pktcMtaDevRealmName OBJECT-TYPE
    SYNTAX      SnmpAdminString (SIZE(1..255))
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
        " This object identifies the Kerberos realm name in all
        capitals. The MTA MUST prohibit the instantiation of any
        two rows with identical Kerberos realm names. The MTA MUST
        also verify that any search operation involving Kerberos
        realm names is done using the uppercase ASCII
        representation of the characters."
    ::= { pktcMtaDevRealmEntry 2 }
```

```
pktcMtaDevRealmPkinitGracePeriod OBJECT-TYPE
    SYNTAX      Unsigned32 (15..600)
    UNITS       "minutes"
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
        " This object contains the PKINIT Grace Period. For the
        purpose of key management with Application Servers (CMSes
```

or the Provisioning Server), the MTA must utilize the PKINIT exchange to obtain Application Server tickets. The MTA may utilize the PKINIT exchange to obtain Ticket Granting Tickets (TGTs), which are then used to obtain Application Server tickets in a TGS exchange. The PKINIT exchange occurs according to the current Ticket Expiration Time (TicketEXP) and on the PKINIT Grace Period (PKINITGP). The MTA MUST initiate the PKINIT exchange at the time: TicketEXP - PKINITGP."

## REFERENCE

" PacketCable Security Specification."

DEFVAL { 15 }

::= { pktcMtaDevRealmEntry 3 }

pktcMtaDevRealmTgsGracePeriod OBJECT-TYPE

SYNTAX Unsigned32 (1..600)

UNITS "minutes"

MAX-ACCESS read-create

STATUS current

## DESCRIPTION

" This object contains the Ticket Granting Server Grace Period (TSGSP). The Ticket Granting Server (TGS) Request/Reply exchange may be performed by the MTA on demand whenever an Application Server ticket is needed to establish security parameters. If the MTA possesses a ticket that corresponds to the Provisioning Server or a CMS that currently exists in the CMS table, the MTA MUST initiate the TGS Request/Reply exchange at the time: TicketEXP - TSGSP."

## REFERENCE

" PacketCable Security Specification."

DEFVAL { 10 }

::= { pktcMtaDevRealmEntry 4 }

pktcMtaDevRealmOrgName OBJECT-TYPE

SYNTAX LongUtf8String

MAX-ACCESS read-create

STATUS current

## DESCRIPTION

" This object contains the X.500 organization name attribute as defined in the subject name of the service provider certificate."

## REFERENCE

" PacketCable Security Specification;  
RFCs 3280 and 4630, Internet X.509 Public Key  
Infrastructure Certificate and Certificate Revocation List  
(CRL) Profile"

::= { pktcMtaDevRealmEntry 5 }

**pktcMtaDevRealmUnsolicitedKeyMaxTimeout OBJECT-TYPE**

SYNTAX Unsigned32 (1..600)

UNITS "seconds"

MAX-ACCESS read-create

STATUS current

**DESCRIPTION**

" This object specifies the maximum time the MTA will attempt to perform the exponential back-off algorithm. This timer only applies when the MTA initiated key management. If the DHCP option code 122, sub-option 4, is provided to the MTA, it overwrites this value.

Unsolicited key updates are retransmitted according to an exponential back-off mechanism using two timers and a maximum retry counter for AS replies.

The initial retransmission timer value is the nominal timer value (pktcMtaDevRealmUnsolicitedKeyNomTimeout). The retransmissions occur with an exponentially increasing interval that caps at the maximum timeout value (pktcMtaDevRealmUnsolicitedKeyMaxTimeout). Retransmissions stop when the maximum retry counter is reached (pktcMatDevRealmUnsolicitedMaxRetries).

For example, with values of 3 seconds for the nominal timer, 20 seconds for the maximum timeout, and 5 retries max, retransmission intervals will be 3 s, 6 s, 12 s, 20 s, and 20 s, and retransmissions then stop because the maximum number of retries has been reached."

**REFERENCE**

" PacketCable Security Specification."

DEFVAL { 100 }

::= { pktcMtaDevRealmEntry 6 }

**pktcMtaDevRealmUnsolicitedKeyNomTimeout OBJECT-TYPE**

SYNTAX Unsigned32 (100..600000)

UNITS "milliseconds"

MAX-ACCESS read-create

STATUS current

**DESCRIPTION**

" This object specifies the initial timeout value for the AS-REQ/AS-REP exponential back-off and retry mechanism. If the DHCP option code 122, sub-option 4, is provided to the MTA, it overwrites this value. This value should account for the average roundtrip time between the MTA and the KDC, as well as the processing delay on the KDC.

Unsolicited key updates are retransmitted according to an exponential back-off mechanism using two timers and a maximum retry counter for AS replies.

The initial retransmission timer value is the nominal timer value (pktcMtaDevRealmUnsolicitedKeyNomTimeout). The retransmissions occur with an exponentially increasing interval that caps at the maximum timeout value (pktcMtaDevRealmUnsolicitedKeyMaxTimeout). Retransmissions stop when the maximum retry counter is reached (pktcMatDevRealmUnsolicitedMaxRetries).

For example, with values of 3 seconds for the nominal timer, 20 seconds for the maximum timeout, and 5 retries max, in retransmission intervals will be 3 s, 6 s, 12 s, 20 s, and 20 s; retransmissions then stop because the maximum number of retries has been reached."

#### REFERENCE

" PacketCable Security Specification."

DEFVAL { 3000 }

::= { pktcMtaDevRealmEntry 7 }

pktcMtaDevRealmUnsolicitedKeyMaxRetries OBJECT-TYPE

SYNTAX Unsigned32 (0..1024)

MAX-ACCESS read-create

STATUS current

#### DESCRIPTION

" This object specifies the maximum number of retries the MTA attempts to obtain a ticket from the KDC.

Unsolicited key updates are retransmitted according to an exponential back-off mechanism using two timers and a maximum retry counter for AS replies.

The initial retransmission timer value is the nominal timer value (pktcMtaDevRealmUnsolicitedKeyNomTimeout). The retransmissions occur with an exponentially increasing interval that caps at the maximum timeout value (pktcMtaDevRealmUnsolicitedKeyMaxTimeout). Retransmissions stop when the maximum retry counter is reached (pktcMatDevRealmUnsolicitedMaxRetries).

For example, with values of 3 seconds for the nominal timer, 20 seconds for the maximum timeout, and 5 retries max, retransmission intervals will be 3 s, 6 s, 12 s, 20 s, and 20 s; retransmissions then stop because the maximum number of retries has been reached."

#### REFERENCE

" PacketCable Security Specification."

DEFVAL { 5 }

```
::= { pktcMtaDevRealmEntry 8 }
```

```
pktcMtaDevRealmStatus      OBJECT-TYPE
```

```
    SYNTAX      RowStatus
```

```
    MAX-ACCESS  read-create
```

```
    STATUS      current
```

```
    DESCRIPTION
```

```
        " This object defines the row status of this realm in the
          realm table (pktcMtaDevRealmTable).
```

```

          An entry in this table is not qualified for activation
          until the object instances of all corresponding columns
          have been initialized, either by default values, or via
          explicit SET operations.  Until all object instances in
          this row are initialized, the status value for this realm
          must be 'notReady(3)'.

```

```

          In particular, two columnar objects must be explicitly
          SET: the realm name (pktcMtaDevRealmName) and the
          organization name (pktcMtaDevRealmOrgName).  Once these 2
          objects have been set and the row status is SET to
          'active(1)', the MTA MUST NOT allow any modification of
          these 2 object values.

```

```

          The value of this object has no effect on whether other
          columnar objects in this row can be modified."

```

```
::= { pktcMtaDevRealmEntry 9 }
```

```
-----
```

```
--
```

```
-- The CMS table, pktcMtaDevCmsTable
```

```
--
```

```
-- The CMS table and the realm table (pktcMtaDevRealmTable) are used
-- for managing the MTA signaling security.  The CMS table defines
-- the CMSes the MTA is allowed to communicate with and contains
-- the parameters describing the SA establishment between the MTA
-- and a CMS.
```

```
-- The CMS table is indexed by pktcMtaDevCmsIndex.  The table
-- contains the CMS FQDN (pktcMtaDevCmsFQDN) and the associated
-- Kerberos realm name (pktcMtaDevCmsKerbRealmName) so that the MTA
-- can find the corresponding Kerberos realm name in the
-- pktcMtaDevRealmTable.
```

```
--
```

```
-----
```

```
pktcMtaDevCmsAvailSlot      OBJECT-TYPE
```

```
    SYNTAX      Unsigned32 (0..128)
```

```
    MAX-ACCESS  read-only
```

```
    STATUS      current
```

```
    DESCRIPTION
```

" This object contains the index number of the first available entry in the CMS table (pktcMtaDevCmsTable). If all the entries in the CMS table have been assigned, this object contains the value of zero. A management station should create new entries in the CMS table, using the following procedure:

First, issue a management protocol retrieval operation to determine the value of the first available index in the CMS table (pktcMtaDevCmsAvailSlot).

Second, issue a management protocol SET operation to create an instance of the pktcMtaDevCmsStatus object by setting its value to 'createAndWait(5)'.

Third, if the SET operation succeeded, continue modifying the object instances corresponding to the newly created conceptual row, without fear of collision with other management stations. When all necessary conceptual columns of the row are properly populated (via SET operations or default values), the management station may SET the pktcMtaDevCmsStatus object to 'active(1)'."

```
::= { pktcMtaDevSecurity 7 }
```

pktcMtaDevCmsTable OBJECT-TYPE

SYNTAX SEQUENCE OF PktcMtaDevCmsEntry

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION

" This object defines the CMS table. The CMS table (pktcMtaDevCmsTable) and the realm table (pktcMtaDevRealmTable) are used for managing security between the MTA and CMSes. Each CMS table entry defines a CMS the managed MTA is allowed to communicate with and contains security parameters for key management with that CMS."

```
::= { pktcMtaDevSecurity 8 }
```

pktcMtaDevCmsEntry OBJECT-TYPE

SYNTAX PktcMtaDevCmsEntry

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION

" This table entry object lists the MTA key management parameters used when establishing Security Associations with a CMS. The conceptual rows MUST NOT persist across MTA reboots."

```
INDEX { pktcMtaDevCmsIndex }
```

```

 ::= { pktcMtaDevCmsTable 1 }

PktcMtaDevCmsEntry ::= SEQUENCE {
    pktcMtaDevCmsIndex                Unsigned32,
    pktcMtaDevCmsFqdn                 SnmpAdminString,
    pktcMtaDevCmsKerbRealmName         SnmpAdminString,
    pktcMtaDevCmsMaxClockSkew          Unsigned32,
    pktcMtaDevCmsSolicitedKeyTimeout   Unsigned32,
    pktcMtaDevCmsUnsolicitedKeyMaxTimeout Unsigned32,
    pktcMtaDevCmsUnsolicitedKeyNomTimeout Unsigned32,
    pktcMtaDevCmsUnsolicitedKeyMaxRetries Unsigned32,
    pktcMtaDevCmsIpsecCtrl              TruthValue,
    pktcMtaDevCmsStatus                 RowStatus
}

pktcMtaDevCmsIndex OBJECT-TYPE
    SYNTAX      Unsigned32 (1..128)
    MAX-ACCESS   not-accessible
    STATUS       current
    DESCRIPTION
        " This object defines the CMS table index."
    ::= { pktcMtaDevCmsEntry 1 }

pktcMtaDevCmsFqdn OBJECT-TYPE
    SYNTAX      SnmpAdminString (SIZE(1..255))
    MAX-ACCESS   read-create
    STATUS       current
    DESCRIPTION
        " This object specifies the CMS FQDN. The MTA must
        prohibit the instantiation of any two rows with identical
        FQDNs. The MTA must also verify that any search and/or
        comparison operation involving a CMS FQDN is case
        insensitive. The MTA must resolve the CMS FQDN as required
        by the corresponding PacketCable Specifications."
    REFERENCE
        " PacketCable MTA Device Provisioning Specification;
        PacketCable Security Specification;
        PacketCable Network-Based Call Signaling Protocol
        Specification."
    ::= { pktcMtaDevCmsEntry 2 }

pktcMtaDevCmsKerbRealmName OBJECT-TYPE
    SYNTAX      SnmpAdminString (SIZE(1..255))
    MAX-ACCESS   read-create
    STATUS       current
    DESCRIPTION
        " This object identifies the Kerberos realm name in uppercase
        characters associated with the CMS defined in this

```



conceptual row. The object value is a reference point to the corresponding Kerberos realm name in the realm table (pktcMtaDevRealmTable)."

```
::= { pktcMtaDevCmsEntry 3 }
```

```
pktcMtaDevCmsMaxClockSkew      OBJECT-TYPE
    SYNTAX      Unsigned32 (1..1800)
    UNITS       "seconds"
    MAX-ACCESS   read-create
    STATUS      current
    DESCRIPTION
        " This object specifies the maximum allowable clock skew
          between the MTA and the CMS defined in this row."
    DEFVAL { 300 }
    ::= { pktcMtaDevCmsEntry 4 }
```

```
pktcMtaDevCmsSolicitedKeyTimeout OBJECT-TYPE
    SYNTAX      Unsigned32 (100..30000)
    UNITS       "milliseconds"
    MAX-ACCESS   read-create
    STATUS      current
    DESCRIPTION
        " This object defines a Kerberos Key Management timer on the
          MTA. It is the time period during which the MTA saves the
          nonce and Server Kerberos Principal Identifier to match an
          AP Request and its associated AP Reply response from the
          CMS. This timer only applies when the CMS initiated key
          management (with a Wake Up message or a Rekey message)."
```

REFERENCE

```
    " PacketCable Security Specification."
    DEFVAL { 1000 }
    ::= { pktcMtaDevCmsEntry 5 }
```

```
-----
--
-- Unsolicited key updates are retransmitted according to an
-- exponential back-off mechanism using two timers and a maximum
-- retry counter for AS replies.
-- The initial retransmission timer value is the nominal timer
-- value (pktcMtaDevCmsUnsolicitedKeyNomTimeout). The
-- retransmissions occur with an exponentially increasing interval
-- that caps at the maximum timeout value
-- (pktcMtaDevCmsUnsolicitedKeyMaxTimeout).
-- Retransmissions stop when the maximum retry counter is reached
-- (pktcMtaDevCmsUnsolicitedMaxRetries).
-- For example, with values of 3 seconds for the nominal
-- timer, 20 seconds for the maximum timeout, and 5 retries max,
-- retransmission intervals will be 3 s, 6 s, 12 s,
```

```
-- 20 s, and 20 s; retransmissions then stop due to the
-- maximum number of retries reached.
```

```
--
```

```
=====
```

```
pktcMtaDevCmsUnsolicitedKeyMaxTimeout OBJECT-TYPE
```

```
SYNTAX      Unsigned32 (1..600)
```

```
UNITS       "seconds"
```

```
MAX-ACCESS  read-create
```

```
STATUS      current
```

```
DESCRIPTION
```

```
    " This object defines the timeout value that only applies
      to an MTA-initiated key management exchange. It is the
      maximum timeout, and it may not be exceeded in the
      exponential back-off algorithm."
```

```
REFERENCE
```

```
    " PacketCable Security Specification."
```

```
DEFVAL { 600 }
```

```
::= { pktcMtaDevCmsEntry 6 }
```

```
pktcMtaDevCmsUnsolicitedKeyNomTimeout OBJECT-TYPE
```

```
SYNTAX      Unsigned32 (100..30000)
```

```
UNITS       "milliseconds"
```

```
MAX-ACCESS  read-create
```

```
STATUS      current
```

```
DESCRIPTION
```

```
    " This object defines the starting value of the timeout
      for an MTA-initiated key management. It should account for
      the average roundtrip time between the MTA and the CMS and
      the processing time on the CMS."
```

```
REFERENCE
```

```
    " PacketCable Security Specification."
```

```
DEFVAL { 500 }
```

```
::= { pktcMtaDevCmsEntry 7 }
```

```
pktcMtaDevCmsUnsolicitedKeyMaxRetries OBJECT-TYPE
```

```
SYNTAX      Unsigned32 (0..1024)
```

```
MAX-ACCESS  read-create
```

```
STATUS      current
```

```
DESCRIPTION
```

```
    " This object contains the maximum number of retries before
      the MTA stops attempting to establish a Security
      Association with the CMS."
```

```
REFERENCE
```

```
    " PacketCable Security Specification."
```

```
DEFVAL { 5 }
```

```
::= { pktcMtaDevCmsEntry 8 }
```

```

pktcMtaDevCmsIpsecCtrl      OBJECT-TYPE
    SYNTAX      TruthValue
    MAX-ACCESS   read-only
    STATUS      current
    DESCRIPTION
        " This object specifies the MTA IPsec control flag.
          If the object value is 'true', the MTA must use Kerberos
          Key Management and IPsec to communicate with this CMS.  If
          it is 'false', IPsec Signaling Security and Kerberos key
          management are disabled for this specific CMS."
    DEFVAL { true }
    ::= { pktcMtaDevCmsEntry 9 }

pktcMtaDevCmsStatus          OBJECT-TYPE
    SYNTAX      RowStatus
    MAX-ACCESS   read-create
    STATUS      current
    DESCRIPTION
        " This object defines the row status associated with this
          particular CMS in the CMS table (pktcMtaDevCmsTable).

          An entry in this table is not qualified for activation
          until the object instances of all corresponding columns
          have been initialized, either by default values or via
          explicit SET operations.  Until all object instances in
          this row are initialized, the status value for this realm
          must be 'notReady(3)'.
          In particular, two columnar objects must be SET: the
          CMS FQDN (pktcMtaDevCmsFqdn) and the Kerberos realm name
          (pktcMtaDevCmsKerbRealmName).  Once these 2 objects have
          been set and the row status is SET to 'active(1)', the MTA
          MUST NOT allow any modification of these 2 object values.

          The value of this object has no effect on
          whether other columnar objects in this row can be
          modified."
    ::= { pktcMtaDevCmsEntry 10 }

pktcMtaDevResetKrbTickets    OBJECT-TYPE
    SYNTAX      BITS {
                                invalidateProvOnReboot (0),
                                invalidateAllCmsOnReboot (1)
                            }
    MAX-ACCESS   read-write
    STATUS      current
    DESCRIPTION
        " This object defines a Kerberos Ticket Control Mask that
          instructs the MTA to invalidate the specific Application

```

Server Kerberos ticket(s) that are stored locally in the MTA NVRAM (non-volatile or persistent memory).

If the MTA does not store Kerberos tickets in NVRAM, it MUST ignore setting of this object and MUST report a BITS value of zero when the object is read.

If the MTA supports Kerberos tickets storage in NVRAM, the object value is encoded as follows:

- Setting the invalidateProvOnReboot bit (bit 0) to 1 means that the MTA MUST invalidate the Kerberos Application Ticket(s) for the Provisioning Application at the next MTA reboot if secure SNMP provisioning mode is used. In non-secure provisioning modes, the MTA MUST return an 'inconsistentValue' in response to SNMP SET operations with a bit 0 set to 1.
- Setting the invalidateAllCmsOnReboot bit (bit 1) to 1 means that the MTA MUST invalidate the Kerberos Application Ticket(s) for all CMSes currently assigned to the MTA endpoints.

If a value is written into an instance of pktcMtaDevResetKrbTickets, the agent MUST retain the supplied value across an MTA re-initialization or reboot."

#### REFERENCE

"PacketCable Security Specification."

DEFVAL { { } }

::= { pktcMtaDevSecurity 9 }

--

-- The following group, pktcMtaDevErrors, defines an OID  
 -- corresponding to error conditions encountered during the MTA  
 -- provisioning.

--

pktcMtaDevErrorsTooManyErrors OBJECT-IDENTITY

STATUS current

DESCRIPTION

"This object defines the OID corresponding to the error condition when too many errors are encountered in the MTA configuration file during provisioning."

::= { pktcMtaDevErrors 1 }

pktcMtaDevProvisioningEnrollment NOTIFICATION-TYPE

OBJECTS {

sysDescr,  
 pktcMtaDevSwCurrentVers,  
 pktcMtaDevTypeIdentifier,  
 ifPhysAddress,  
 pktcMtaDevCorrelationId

```

    }
    STATUS      current
    DESCRIPTION
        " This INFORM notification is issued by the MTA to initiate
        the PacketCable provisioning process when the MTA SNMP
        enrollment mechanism is used.
        It contains the system description, the current software
        version, the MTA device type identifier, the MTA MAC
        address (obtained in the MTA ifTable in the ifPhysAddress
        object that corresponds to the ifIndex 1), and a
        correlation ID."
    ::= { pktcMtaNotification 1 }

pktcMtaDevProvisioningStatus  NOTIFICATION-TYPE
    OBJECTS {
        ifPhysAddress,
        pktcMtaDevCorrelationId,
        pktcMtaDevProvisioningState
    }
    STATUS      current
    DESCRIPTION
        " This INFORM notification may be issued by the MTA to
        confirm the completion of the PacketCable provisioning
        process, and to report its provisioning completion
        status.
        It contains the MTA MAC address (obtained in the MTA
        ifTable in the ifPhysAddress object that corresponds
        to the ifIndex 1), a correlation ID and the MTA
        provisioning state as defined in
        pktcMtaDevProvisioningState."
    ::= { pktcMtaNotification 2 }

--
-- Compliance Statements
--

pktcMtaCompliances  OBJECT IDENTIFIER ::= { pktcMtaConformance 1 }
pktcMtaGroups       OBJECT IDENTIFIER ::= { pktcMtaConformance 2 }

pktcMtaBasicCompliance MODULE-COMPLIANCE
    STATUS      current
    DESCRIPTION
        " The compliance statement for MTA devices that implement
        PacketCable or IPCablecom requirements.

        This compliance statement applies to MTA implementations
        that support PacketCable 1.0 or IPCablecom requirements,
        which are not IPv6-capable at the time of this

```

RFC publication."

MODULE -- Unconditionally mandatory groups for MTAs

```
MANDATORY-GROUPS {
    pktcMtaGroup,
    pktcMtaNotificationGroup
}
```

```
OBJECT pktcMtaDevDhcpServerAddressType
SYNTAX      InetAddressType { ipv4(1) }
DESCRIPTION
    " Support for address types other than 'ipv4(1)'
    is not presently specified and therefore is not
    required. It may be defined in future versions of
    this MIB module."
```

```
OBJECT pktcMtaDevDnsServerAddressType
SYNTAX      InetAddressType { ipv4(1) }
DESCRIPTION
    " Support for address types other than 'ipv4(1)'
    is not presently specified and therefore is not
    required. It may be defined in future versions of
    this MIB module."
```

```
OBJECT pktcMtaDevTimeServerAddressType
SYNTAX      InetAddressType { ipv4(1) }
DESCRIPTION
    " Support for address types other than 'ipv4(1)'
    is not presently specified and therefore is not
    required. It may be defined in future versions of
    this MIB module."
```

```
OBJECT      pktcMtaDevServerDhcp1
SYNTAX      InetAddress (SIZE(4))
DESCRIPTION
    "An implementation is only required to support IPv4
    addresses. Other address types support may be defined in
    future versions of this MIB module."
```

```
OBJECT      pktcMtaDevServerDhcp2
SYNTAX      InetAddress (SIZE(4))
DESCRIPTION
    "An implementation is only required to support IPv4
    addresses. Other address types support may be defined in
    future versions of this MIB module."
```

```
OBJECT      pktcMtaDevServerDns1
```

SYNTAX InetAddress (SIZE(4))  
DESCRIPTION  
"An implementation is only required to support IPv4 addresses. Other address types support may be defined in future versions of this MIB module."

OBJECT pktcMtaDevServerDns2  
SYNTAX InetAddress (SIZE(4))  
DESCRIPTION  
"An implementation is only required to support IPv4 addresses. Other address types support may be defined in future versions of this MIB module."

OBJECT pktcMtaDevTimeServer  
SYNTAX InetAddress (SIZE(4))  
DESCRIPTION  
"An implementation is only required to support IPv4 addresses. Other address types support may be defined in future versions of this MIB module."

OBJECT pktcMtaDevProvConfigEncryptAlg  
SYNTAX PktcMtaDevProvEncryptAlg  
DESCRIPTION  
"An implementation is only required to support values of none(0) and des64Cbcmode(1).  
An IV of zero is used to encrypt in des64Cbcmode, and the length of pktcMtaDevProvConfigKey is 64 bits, as defined in the PacketCable Security specification.  
Other encryption types may be defined in future versions of this MIB module."

OBJECT pktcMtaDevRealmOrgName  
SYNTAX LongUtf8String (SIZE (1..384))  
DESCRIPTION  
"The Organization Name field in X.509 certificates can contain up to 64 UTF-8 encoded characters, as defined in RFCs 3280 and 4630. Therefore, compliant devices are only required to support Organization Name values of up to 64 UTF-8 encoded characters. Given that RFCs 3280 and 4630 define the UTF-8 encoding, compliant devices must support a maximum size of 384 octets for pktcMtaDevRealmOrgName. The calculation of 384 octets comes from the RFC 3629 UTF-8 encoding definition whereby the UTF-8 encoded characters are encoded as sequences of 1 to 6 octets, assuming that code points as high as 0x7fffffff might be used. Subsequent versions of Unicode and ISO 10646 have limited the upper bound to 0x10ffff."

Consequently, the current version of UTF-8, defined in RFC 3629, does not require more than four octets to encode a valid code point."

::= { pktcMtaCompliances 1 }

pktcMtaGroup OBJECT-GROUP

OBJECTS {

pktcMtaDevResetNow,  
pktcMtaDevSerialNumber,  
pktcMtaDevSwCurrentVers,  
pktcMtaDevFQDN,  
pktcMtaDevEndPntCount,  
pktcMtaDevEnabled,  
pktcMtaDevProvisioningCounter,  
pktcMtaDevErrorOid,  
pktcMtaDevErrorValue,  
pktcMtaDevErrorReason,  
pktcMtaDevTypeIdentifier,  
pktcMtaDevProvisioningState,  
pktcMtaDevHttpAccess,  
pktcMtaDevCertificate,  
pktcMtaDevCorrelationId,  
pktcMtaDevManufacturerCertificate,  
pktcMtaDevDhcpServerAddressType,  
pktcMtaDevDnsServerAddressType,  
pktcMtaDevTimeServerAddressType,  
pktcMtaDevProvConfigEncryptAlg,  
pktcMtaDevServerDhcp1,  
pktcMtaDevServerDhcp2,  
pktcMtaDevServerDns1,  
pktcMtaDevServerDns2,  
pktcMtaDevTimeServer,  
pktcMtaDevConfigFile,  
pktcMtaDevSnmpEntity,  
pktcMtaDevRealmPkinitGracePeriod,  
pktcMtaDevRealmTgsGracePeriod,  
pktcMtaDevRealmAvailSlot,  
pktcMtaDevRealmName,  
pktcMtaDevRealmOrgName,  
pktcMtaDevRealmUnsolicitedKeyMaxTimeout,  
pktcMtaDevRealmUnsolicitedKeyNomTimeout,  
pktcMtaDevRealmUnsolicitedKeyMaxRetries,  
pktcMtaDevRealmStatus,  
pktcMtaDevCmsAvailSlot,  
pktcMtaDevCmsFqdn,  
pktcMtaDevCmsKerbRealmName,  
pktcMtaDevCmsUnsolicitedKeyMaxTimeout,



```

    pktcMtaDevCmsUnsolicitedKeyNomTimeout,
    pktcMtaDevCmsUnsolicitedKeyMaxRetries,
    pktcMtaDevCmsSolicitedKeyTimeout,
    pktcMtaDevCmsMaxClockSkew,
    pktcMtaDevCmsIpsecCtrl,
    pktcMtaDevCmsStatus,
    pktcMtaDevResetKrbTickets,
    pktcMtaDevProvUnsolicitedKeyMaxTimeout,
    pktcMtaDevProvUnsolicitedKeyNomTimeout,
    pktcMtaDevProvUnsolicitedKeyMaxRetries,
    pktcMtaDevProvKerbRealmName,
    pktcMtaDevProvSolicitedKeyTimeout,
    pktcMtaDevProvConfigHash,
    pktcMtaDevProvConfigKey,
    pktcMtaDevProvState,
    pktcMtaDevProvisioningTimer,
    pktcMtaDevTelephonyRootCertificate
}
STATUS          current
DESCRIPTION
    " A collection of objects for managing PacketCable or
      IPCablecom MTA implementations."
 ::= { pktcMtaGroups 1 }

pktcMtaNotificationGroup          NOTIFICATION-GROUP
NOTIFICATIONS {
    pktcMtaDevProvisioningStatus,
    pktcMtaDevProvisioningEnrollment
}
STATUS          current
DESCRIPTION
    " A collection of notifications dealing with the change of
      MTA provisioning status."
 ::= { pktcMtaGroups 2 }

pktcMtaBasicSmtaCompliance MODULE-COMPLIANCE
STATUS          current
DESCRIPTION
    " The compliance statement for S-MTA devices
      that implement PacketCable or IPCablecom requirements.

      This compliance statement applies to S-MTA implementations
      that support PacketCable or IPCablecom requirements,
      which are not IPv6-capable at the time of this
      RFC publication."

MODULE -- Unconditionally Mandatory Groups for S-MTA devices
MANDATORY-GROUPS {

```

```
    pktcMtaGroup,  
    pktcMtaNotificationGroup  
}
```

```
OBJECT pktcMtaDevDhcpServerAddressType  
SYNTAX      InetAddressType { ipv4(1) }  
DESCRIPTION  
    " Support for address types other than 'ipv4(1)'  
    is not presently specified and therefore is not  
    required. It may be defined in future versions of  
    this MIB module."
```

```
OBJECT pktcMtaDevDnsServerAddressType  
SYNTAX      InetAddressType { ipv4(1) }  
DESCRIPTION  
    " Support for address types other than 'ipv4(1)'  
    is not presently specified and therefore is not  
    required. It may be defined in future versions of  
    this MIB module."
```

```
OBJECT pktcMtaDevTimeServerAddressType  
SYNTAX      InetAddressType { ipv4(1) }  
DESCRIPTION  
    " Support for address types other than 'ipv4(1)'  
    is not presently specified and therefore is not  
    required. It may be defined in future versions of  
    this MIB module."
```

```
OBJECT pktcMtaDevServerDhcp1  
SYNTAX      InetAddress (SIZE(4))  
DESCRIPTION  
    "An implementation is only required to support IPv4  
    addresses. Other address types support may be defined in  
    future versions of this MIB module."
```

```
OBJECT pktcMtaDevServerDhcp2  
SYNTAX      InetAddress (SIZE(4))  
DESCRIPTION  
    "An implementation is only required to support IPv4  
    addresses. Other address types support may be defined in  
    future versions of this MIB module."
```

```
OBJECT pktcMtaDevServerDns1  
SYNTAX      InetAddress (SIZE(4))  
DESCRIPTION  
    "An implementation is only required to support IPv4  
    addresses. Other address types support may be defined in  
    future versions of this MIB module."
```

OBJECT pktcMtaDevServerDns2  
SYNTAX InetAddress (SIZE(4))  
DESCRIPTION  
"An implementation is only required to support IPv4 addresses. Other address types support may be defined in future versions of this MIB module."

OBJECT pktcMtaDevTimeServer  
SYNTAX InetAddress (SIZE(4))  
DESCRIPTION  
"An implementation is only required to support IPv4 addresses. Other address types support may be defined in future versions of this MIB module."

OBJECT pktcMtaDevProvConfigEncryptAlg  
SYNTAX PktcMtaDevProvEncryptAlg  
DESCRIPTION  
"An implementation is only required to support values of none(0) and des64Cbcmode(1).  
An IV of zero is used to encrypt in des64Cbcmode, and the length of pktcMtaDevProvConfigKey is 64 bits, as defined in the PacketCable Security specification. Other encryption types may be defined in future versions of this MIB module."

OBJECT pktcMtaDevRealmOrgName  
SYNTAX LongUtf8String (SIZE (1..384))  
DESCRIPTION  
"The Organization Name field in X.509 certificates can contain up to 64 UTF-8 encoded characters, as defined in RFCs 3280 and 4630. Therefore, compliant devices are only required to support Organization Name values of up to 64 UTF-8 encoded characters. Given that RFCs 3280 and 4630 define the UTF-8 encoding, compliant devices must support a maximum size of 384 octets for pktcMtaDevRealmOrgName. The calculation of 384 octets comes from the RFC 3629 UTF-8 encoding definition whereby the UTF-8 encoded characters are encoded as sequences of 1 to 6 octets, assuming that code points as high as 0x7fffffff might be used. Subsequent versions of Unicode and ISO 10646 have limited the upper bound to 0x10ffff. Consequently, the current version of UTF-8, defined in RFC 3629 does not require more than four octets to encode a valid code point."

MODULE DOCS-CABLE-DEVICE-MIB  
MANDATORY-GROUPS {

```
        docsDevSoftwareGroupV2
    }

MODULE DOCS-IETF-BPI2-MIB
    MANDATORY-GROUPS {
        docsBpi2CodeDownloadGroup
    }

    ::= { pktcMtaCompliances 2 }

END
```

## 5. Acknowledgements

The current editors would like to thank the members of the IETF IPCDN working group and the CableLabs PacketCable Provisioning and OSS focus team for their comments and suggestions. In particular, we wish to express our gratitude for the contributions made by the following individuals (in no particular order): Angela Lyda, Sumanth Channabasappa, Matt A. Osman, Klaus Hermanns, Paul Duffy, Rick Vetter, Sasha Medvinsky, Roy Spitzer, Itay Sherman, Satish Kumar and Eric Rosenfeld. Finally, special thanks to our area director Bert Wijnen, Rich Woundy, Randy Presuhn, Mike Heard, and Dave Thaler.

## 6. Security Considerations

There are a number of management objects defined in this MIB module with 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. Improper manipulation of the objects defined in this MIB may result in random behavior of MTA devices and may result in service disruption. These are the tables and objects and their sensitivity/vulnerability:

- The following objects, if SET maliciously, would cause the MTA device to reset and/or stop its service:

```
    pktcMtaDevResetNow.
    pktcMtaDevEnabled.
```

- All writable objects in the pktcMtaDevServer group and some in the pktcMtaDevRealmTable share the potential, if SET maliciously, to prevent the MTA from provisioning properly. Thus, they are considered very sensitive for service delivery. The objects in question are:

```
pktcMtaDevProvisioningTimer,  
pktcMtaDevDhcpServerAddressType,  
pktcMtaDevDnsServerAddressType,  
pktcMtaDevTimeServerAddressType,  
pktcMtaDevProvConfigEncryptAlg,  
pktcMtaDevServerDns1,  
pktcMtaDevServerDns2,  
pktcMtaDevTimeServer,  
pktcMtaDevConfigFile,  
pktcMtaDevProvConfigHash,  
pktcMtaDevProvConfigKey,  
pktcMtaDevProvSolicitedKeyTimeout,  
pktcMtaDevRealmName,  
pktcMtaDevRealmOrgName,  
pktcMtaDevRealmUnsolicitedKeyMaxTimeout,  
pktcMtaDevRealmUnsolicitedKeyNomTimeout,  
pktcMtaDevRealmUnsolicitedKeyMaxRetries, and  
pktcMtaDevRealmStatus.
```

Certain of the above objects have additional specific vulnerabilities:

- o pktcMtaDevServerDns1 and pktcMtaDevServerDns2, if SET maliciously, could prevent the MTA from being authenticated and consequently from getting telephony services.
- o pktcMtaDevRealmStatus, if SET maliciously, could cause the whole row of the table to be deleted, which may prevent MTA from getting telephony services.
- All writable objects in the pktcMtaDevCmsTable table share the potential, if SET maliciously, to disrupt the telephony service by altering which Call Management Server the MTA must send signaling registration to; in particular:

```
pktcMtaDevCmsFqdn,  
pktcMtaDevCmsKerbRealmName,  
pktcMtaDevCmsMaxClockSkew,  
pktcMtaDevCmsSolicitedKeyTimeout,  
pktcMtaDevCmsUnsolicitedKeyMaxTimeout,  
pktcMtaDevCmsUnsolicitedKeyNomTimeout,  
pktcMtaDevCmsUnsolicitedKeyMaxRetries (this object, if set to a  
zero value '0', may prevent the MTA from retrying its attempt  
to establish a Security Association with the CMS), and  
pktcMtaDevCmsStatus.
```

- Some writable objects in the `pktcMtaDevRealmTable` table will not have an immediate effect on service, if SET maliciously. However, they may impact the service performance and cause avalanche attacks on provisioning and Kerberos KDC servers, especially after massive device reboots occur. The objects in question are as follows:

`pktcMtaDevResetKrbTickets`: This object, if set to 'true', will cause the MTA to request a new Kerberos ticket at reboot.

`pktcMtaDevRealmPkinitGracePeriod`, `pktcMtaDevRealmTgsGracePeriod`: These 2 objects, if set to short time periods, will cause the MTA to renew its tickets more frequently.

Some of the readable objects in this MIB module (i.e., objects with a MAX-ACCESS other than not-accessible) may be considered sensitive or vulnerable in some network environments. Some of these objects may contain information that may be sensitive from a business or customer perspective. It is thus important to control even GET and/or NOTIFY access to these objects and possibly to even encrypt the values of these objects when sending them over the network via SNMP.

These are the tables and objects and their sensitivity and vulnerability:

- Some readable objects in the `pktcMtaDevBase`, `pktcMtaDevServer`, and `pktcMtaDevSecurity` groups share the potential, if read maliciously, to facilitate Denial-of-Service (DoS) attacks against provisioning or Kerberos servers. The object in question are as follows:

`pktcMtaDevServerDhcp1`, `pktcMtaDevServerDhcp2`, and `pktcMtaDevSnmppEntity`. The values of these objects may be used to launch DoS attacks on the Telephony Service Provider DHCP or Provisioning servers.

`pktcMtaDevProvKerbRealmName`, `pktcMtaDevManufacturerCertificate`, `pktcMtaDevCertificate` and `pktcMtaDevTelephonyRootCertificate`. The values of these objects may be used by attackers to launch DoS attacks against Kerberos servers.

- One additional readable object may expose some security threats: `pktcMtaDevFQDN`. This object may include sensitive information about the domain name, and potentially, the domain topology.

SNMP versions prior to SNMPv3 did not include adequate security. 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 module.

It is RECOMMENDED that implementers consider the security features as provided by the SNMPv3 framework (see Section 8 in [RFC3410]), including full support for the SNMPv3 cryptographic mechanisms (for authentication and privacy).

Further, deployment of SNMP versions prior to SNMPv3 is NOT RECOMMENDED. Instead, it is RECOMMENDED to deploy SNMPv3 and to enable cryptographic security. It is then a customer/operator responsibility to ensure that the SNMP entity giving access to an instance of this MIB module 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.

## 7. IANA Considerations

The MIB module defined in this document uses the following IANA-assigned OBJECT IDENTIFIER values, recorded in the SMI Numbers registry:

Descriptor	OBJECT IDENTIFIER value
-----	-----
pktcIetfMtaMib	{ mib-2 140 }

## 8. Normative References

- [RFC868] Postel, J. and K. Harrenstien, "Time Protocol", STD 26, RFC 868, May 1983.
- [RFC1350] Sollins, K., "The TFTP Protocol (Revision 2)", STD 33, RFC 1350, July 1992.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC2131] Droms, R., "Dynamic Host Configuration Protocol", RFC 2131, March 1997.
- [RFC2132] Alexander, S. and R. Droms, "DHCP Options and BOOTP Vendor Extensions", RFC 2132, March 1997.
- [RFC2287] Krupczak, C. and J. Saperia, "Definitions of System-Level Managed Objects for Applications", RFC 2287, February 1998.

- [RFC2578] McCloghrie, K., Perkins, D., Schoenwaelder J., Case, J. Rose, M. and S. Waldbusser, "Structure of Management Information Version 2 (SMIv2)", STD 58, RFC 2578, April 1999.
- [RFC2579] McCloghrie, K., Perkins, D., Schoenwaelder, J. Case, J. Rose, M. and S. Waldbusser, "Textual Conventions for SMIv2", STD 58, RFC 2579, April 1999.
- [RFC2580] McCloghrie, K., Perkins, D., Schoenwaelder J., Case, J., Rose, M. and S. Waldbusser, "Conformance Statements for SMIv2", STD 58, RFC 2580, April 1999.
- [RFC2616] Fielding, R., Gettys, J., Mogul, J., Frystyk, H., Masinter, L., Leach, P., and T. Berners-Lee, "Hypertext Transfer Protocol -- HTTP/1.1", RFC 2616, June 1999.
- [RFC2863] McCloghrie, K. and F. Kastenholz, "The Interfaces Group MIB", RFC 2863, June 2000.
- [RFC3280] Housley, R., Polk, W., Ford, W., and D. Solo, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 3280, April 2002.
- [RFC3411] Harrington, D., Presuhn, R., and B. Wijnen, "An Architecture for Describing Simple Network Management Protocol (SNMP) Management Frameworks", STD 62, RFC 3411, December 2002.
- [RFC3418] Presuhn, R., "Management Information Base (MIB) for the Simple Network Management Protocol (SNMP)", STD 62, RFC 3418, December 2002.
- [RFC3495] Beser, B. and P. Duffy, "Dynamic Host Configuration Protocol (DHCP) Option for CableLabs Client Configuration", RFC 3495, March 2003.
- [RFC3594] Duffy, P., "PacketCable Security Ticket Control Sub-Option for the DHCP CableLabs Client Configuration (CCC) Option", RFC 3594, September 2003.
- [RFC4001] Daniele, M., Haberman, B., Routhier, S., and J. Schoenwaelder, "Textual Conventions for Internet Network Addresses", RFC 4001, February 2005.



- [RFC4131] Green, S., Ozawa, K., Cardona, E., and A. Katsnelson, "Management Information Base for Data Over Cable Service Interface Specification (DOCSIS) Cable Modems and Cable Modem Termination Systems for Baseline Privacy Plus", RFC 4131, September 2005.
- [RFC4630] Housley, R. and S. Santesson, "Update to DirectoryString Processing in the Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 4630, August 2006.
- [RFC4639] Woundy, R. and K. Marez, "Cable Device Management Information Base for Data-Over-Cable Service Interface Specification (DOCSIS) Compliant Cable Modems and Cable Modem Termination Systems", RFC 4639, December 2006.
- [PKT-SP-PROV] Packetcable MTA Device Provisioning Specification, Issued, PKT-SP-PROV-I11-050812, August 2005.  
<http://www.packetcable.com/specifications/>  
<http://www.cablelabs.com/specifications/archives/>
- [PKT-SP-SEC] PacketCable Security Specification, Issued, PKT-SP-SEC-I12-050812, August 2005.  
<http://www.packetcable.com/specifications/>  
<http://www.cablelabs.com/specifications/archives/>
- [ITU-T-J112] Transmission Systems for Interactive Cable Television Services, Annex B, J.112, ITU-T, March, 1998.
- [ITU-T-J168] IPCablecom Multimedia Terminal Adapter (MTA) MIB requirements, J.168, ITU-T, March, 2001.

## 9. Informative References

- [RFC3410] Case, J., Mundy, R., Partain, D., and B. Stewart, "Introduction and Applicability Statements for Internet-Standard Management Framework", RFC 3410, December 2002.
- [RFC3617] Lear, E., "Uniform Resource Identifier (URI) Scheme and Applicability Statement for the Trivial File Transfer Protocol (TFTP)", RFC 3617, October 2003.

- [RFC3629] Yergeau, F., "UTF-8, a transformation format of ISO 10646", STD 63, RFC 3629, November 2003.
- [PKT-SP-MIB-MTA] PacketCable MTA MIB Specification, Issued, PKT-SP-MIB-MTA-I10-050812, August 2005.  
<http://www.packetcable.com/specifications/>  
<http://www.cablelabs.com/specifications/archives/>
- [ETSITS101909-8] ETSI TS 101 909-8: "Access and Terminals (AT); Digital Broadband Cable Access to the Public Telecommunications Network; IP Multimedia Time Critical Services; Part 8: Media Terminal Adaptor (MTA) Management Information Base (MIB)".
- [EN300001] EN 300 001 V1.5.1 (1998-10): "European Standard (Telecommunications series) Attachments to Public Switched Telephone Network (PSTN); General technical requirements for equipment connected to an analogue subscriber interface in the PSTN".
- [EN300659-1] EN 300 659-1: "Public Switched Telephone Network (PSTN); Subscriber line protocol over the local loop for display (and related) services; Part 1: On hook data transmission".
- [NCSSIGMIB] Beacham G., Kumar S., Channabasappa S., "Network Control Signaling (NCS) Signaling MIB for PacketCable and IPCablecom Multimedia Terminal Adapters (MTAs)", Work in Progress, June 2006.

## Authors' Addresses

Eugene Nechamkin  
Broadcom Corporation,  
200 - 13711 International Place  
Richmond, BC, V6V 2Z8  
CANADA

Phone: +1 604 233 8500  
EMail: enechamkin@broadcom.com

Jean-Francois Mule  
Cable Television Laboratories, Inc.  
858 Coal Creek Circle  
Louisville, Colorado 80027-9750  
U.S.A.

Phone: +1 303 661 9100  
EMail: jf.mule@cablelabs.com

## Full Copyright Statement

Copyright (C) The IETF Trust (2006).

This document is subject to the rights, licenses and restrictions contained in BCP 78, and except as set forth therein, the authors retain all their rights.

This document and the information contained herein are provided on an "AS IS" basis and THE CONTRIBUTOR, THE ORGANIZATION HE/SHE REPRESENTS OR IS SPONSORED BY (IF ANY), THE INTERNET SOCIETY, THE IETF TRUST, AND THE INTERNET ENGINEERING TASK FORCE DISCLAIM ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

## Intellectual Property

The IETF takes no position regarding the validity or scope of any Intellectual Property Rights 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; nor does it represent that it has made any independent effort to identify any such rights. Information on the procedures with respect to rights in RFC documents can be found in BCP 78 and BCP 79.

Copies of IPR disclosures made to the IETF Secretariat 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 implementers or users of this specification can be obtained from the IETF on-line IPR repository at <http://www.ietf.org/ipr>.

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights that may cover technology that may be required to implement this standard. Please address the information to the IETF at [ietf-ipr@ietf.org](mailto:ietf-ipr@ietf.org).

## Acknowledgement

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

