ETSI TS 103 161-11 V1.1.1 (2011-10)

Access, Terminals, Transmission and Multiplexing (ATTM);
Integrated Broadband Cable and Television Networks;
IPCablecom 1.5;
Part 11: Media terminal adapter (MTA) device provisioning

Reference
DTS/ATTM-003011-11

Keywords
access, broadband, cable, IP, multimedia, PSTN

ETSI

650 Route des Lucioles
F-06921 Sophia Antipolis Cedex - FRANCE

Tel.: +33 4 92 94 42 00 Fax: +33 4 93 65 47 16

Siret N° 348 623 562 00017 - NAF 742 C
Association à but non lucratif enregistrée à la
Sous-Préfecture de Grasse (06) N° 7803/88

Important notice

Individual copies of the present document can be downloaded from:
http://www.etsi.org

The present document may be made available in more than one electronic version or in print. In any case of existing or perceived difference in contents between such versions, the reference version is the Portable Document Format (PDF). In case of dispute, the reference shall be the printing on ETSI printers of the PDF version kept on a specific network drive within ETSI Secretariat.

Users of the present document should be aware that the document may be subject to revision or change of status. Information on the current status of this and other ETSI documents is available at
http://portal.etsi.org/tb/status/status.asp

If you find errors in the present document, please send your comment to one of the following services:
http://portal.etsi.org/chaircor/ETSI_support.asp

Contents

Intellectual Property Rights

IPRs essential or potentially essential to the present document may have been declared to ETSI. The information pertaining to these essential IPRs, if any, is publicly available for **ETSI members and non-members**, and can be found in ETSI SR 000 314: *"Intellectual Property Rights (IPRs); Essential, or potentially Essential, IPRs notified to ETSI in respect of ETSI standards"*, which is available from the ETSI Secretariat. Latest updates are available on the ETSI Web server (http://ipr.etsi.org).

Pursuant to the ETSI IPR Policy, no investigation, including IPR searches, has been carried out by ETSI. No guarantee can be given as to the existence of other IPRs not referenced in ETSI SR 000 314 (or the updates on the ETSI Web server) which are, or may be, or may become, essential to the present document.

Foreword

This Technical Specification (TS) has been produced by ETSI Technical Committee Access, Terminals, Transmission and Multiplexing (ATTM).

The present document is part 11 of a multi-part IPCablecom 1.5 deliverable covering the Digital Broadband Cable Access to the Public Telecommunications Network; IP Multimedia Time Critical Services, as identified below:

Part 1: "Overview";

Part 2: "Architectural framework for the delivery of time critical services over Cable Television Networks using Cable Modems";

Part 3: "Audio Codec Requirements for the Provision of Bi-Directional Audio Service over Cable Television Networks using Cable Modems";

Part 4: "Network Call Signalling Protocol";

Part 5: "Dynamic Quality of Service for the Provision of Real Time Services over Cable Television Networks using Cable Modems";

Part 6: "Event Message Specification";

Part 7: "Media Terminal Adapter (MTA Management Information Base (MIB)";

Part 8: "Network Call Signalling (NCS) MIB Requirements";

Part 9: "Security";

Part 10: "Management Information Base (MIB) Framework";

Part 11: "Media terminal adapter (MTA) device provisioning";

Part 12: "Management Event Mechanism";

Part 13: "Trunking Gateway Control Protocol - MGCP option";

Part 14: "Embedded MTA Analog Interface and Powering Specification"

Part 15: "Analog Trunking for PBX Specification";

Part 16: "Signalling for Call Management Server";

Part 17: "CMS Subscriber Provisioning Specification";

Part 18: "Media Terminal Adapter Extension MIB";

Part 19: "IPCablecom Audio Server Protocol Specification - MGCP option";

Part 20: "Management Event MIB Specification";

Part 21: "Signalling Extension MIB Specification".

NOTE 1: Additional parts may be proposed and will be added to the list in future versions.

NOTE 2: The choice of a multi-part format for this deliverable is to facilitate maintenance and future enhancements.

1 Introduction

The present document describes the IPCablecom 1.5 embedded-MTA device initialization and provisioning. The present document is issued to facilitate design and field-testing leading to manufacturability and interoperability of conforming hardware and software by multiple vendors.

1.1 Scope

The scope of the present document is limited to the provisioning of a IPCablecom 1.5 embedded-MTA device by a single provisioning and network management provider. An attempt has been made to provide enough detail to enable vendors to build an embedded-MTA device that is interoperable in an IPCablecom 1.5 network configuration. The present document defines the provisioning of MTA components of the embedded MTA device (unless stated otherwise).

2 References

References are either specific (identified by date of publication and/or edition number or version number) or non-specific. For specific references, only the cited version applies. For non-specific references, the latest version of the reference document (including any amendments) applies.

Referenced documents which are not found to be publicly available in the expected location might be found at http://docbox.etsi.org/Reference.

NOTE: While any hyperlinks included in this clause were valid at the time of publication, ETSI cannot guarantee their long term validity.

2.1 Normative references

The following referenced documents are necessary for the application of the present document.

[1] IETF RFC 2131: "DHCP: Dynamic Host Configuration Protocol", March 1997.

[2] ETSI TS 103 161-7: "Access, Terminals, Transmission and Multiplexing (ATTM); Integrated Broadband Cable and Television Networks; IPCablecom 1.5; Part 7: Media Terminal Adapter (MTA) Management Information Base (MIB)".

[3] ETSI TS 103 161-8: "Access, Terminals, Transmission and Multiplexing (ATTM); Integrated Broadband Cable and Television Networks; IPCablecom 1.5; Part 8: Network Call Signalling (NCS) MIB Requirements".

[4] ETSI TS 103 161-4: "Access, Terminals, Transmission and Multiplexing (ATTM); Integrated Broadband Cable and Television Networks; IPCablecom 1.5 Part 4: Network Call Signalling Protocol".

[5] ETSI TS 103 161-9: "Access, Terminals, Transmission and Multiplexing (ATTM); Integrated Broadband Cable and Television Networks; IPCablecom 1.5; Part 9: Security".

[6] ETSI ES 201 488: "Access and Terminals (AT); Data Over Cable Systems".

[7] IETF RFC 3413/STD0062: "Simple Network Management Protocol (SNMP) Applications", December 2002.

[8] IETF RFC 3414/STD0062: "User-based Security Model (USM) for version 3 of the Simple Network Management Protocol (SNMPv3)", December 2002.

[9] IETF RFC 3415/STD0062: "View-based Access Control Model (VACM) for the Simple Network Management Protocol (SNMP)", December 2002.

[10] IETF RFC 3396: "Encoding Long Options in the Dynamic Host Configuration Protocol (DHCPv4)", November 2002.

[11] IETF RFC 2132: "DHCP Options and BOOTP Vendor Extensions", March 1997.

[12] ANSI/SCTE 107 2009: "Embedded Cable Modem Devices".

[13] ETSI TS 103 161-20: "Access, Terminals, Transmission and Multiplexing (ATTM); Integrated Broadband Cable and Television Networks; IPCablecom 1.5; Part 20: Management Event MIB Specification".

[14] ETSI TS 103 161-12: "Access, Terminals, Transmission and Multiplexing (ATTM); Integrated Broadband Cable and Television Networks; IPCablecom 1.5 Part 12: Management Event Mechanism".

[15] ETSI TS 103 161-21: "Access, Terminals, Transmission and Multiplexing (ATTM); Integrated Broadband Cable and Television Networks; IPCablecom 1.5; Part 21: Signalling Extension MIB Specification".

[16] ETSI TS 103 161-18: "Access, Terminals, Transmission and Multiplexing (ATTM); Integrated Broadband Cable and Television Networks; IPCablecom 1.5 Part 18: Media Terminal Adapter Extension MIB".

[17] Cable Television Laboratories, Inc. (CL-SP-MIB-BB-I04-100608):"CableLabs® Battery Backup MIB Specification", June 8, 2010.

[18] ETSI EN 302 878-4: "Access, Terminals, Transmission and Multiplexing (ATTM); Third Generation Transmission Systems for Interactive Cable Television Services - IP Cable Modems; Part 4: MAC and Upper Layer Protocols; DOCSIS® 3.0".

[19] Cable Television Laboratories, Inc. (CL-SP-CANN-DHCP-Reg-I06-110210): "CableLabs®' DHCP Options Registry Specification", February 10, 2011,

[20] IETF RFC 3315: "Dynamic Host Configuration Protocol for IPv6 (DHCPv6)", July 2003.

[21] IETF RFC 4682: "Multimedia Terminal Adapter (MTA) Management Information Base for PacketCable- and IPCablecom-Compliant Devices", December 2006.

[22] IETF RFC 5098: "Signaling MIB for PacketCable and IPCablecom Multimedia Terminal Adapters (MTAs)", February 2008.

[23] IETF RFC 5428: "Management Event Management Information Base (MIB) for PacketCable- and IPCablecom-Compliant Devices", April 2009.

[24] IETF RFC 2863: "The Interfaces Group MIB", June 2000.

[25] IETF RFC 2833: "RTP Payload for DTMF Digits, Telephony Tones and Telephony Signals ", May 2000.

[26] IETF RFC 3495: "Dynamic Host Configuration Protocol (DHCP) Option for CableLabs Client Configuration", March 2003.

[27] IETF RFC 3412/STD0062: "Message Processing and Dispatching for the Simple Network Management Protocol (SNMP)", December 2002.

[28] ANSI/SCTE 23-3: 2010, DOCSIS® 1.1 Part 3: "Operations Support System Interface".

[29] ETSI TS 103 161-10: "Access, Terminals, Transmission and Multiplexing (ATTM); Integrated Broadband Cable and Television Networks; IPCablecom 1.5 Part 10: Management Information Base (MIB) Framework"

[30] IETF RFC 2616: "Hypertext Transfer Protocol -- HTTP/1.1".

[31] IETF RFC 3617: "Uniform Resource Identifier (URI) Scheme and Applicability Statement for the Trivial File Transfer Protocol (TFTP)".

[32] IETF RFC 3584: "Coexistence between Version 1, Version 2, and Version 3 of the Internet-standard Network Management Framework", August 2003.

[33] IETF RFC 3611: "RTP Control Protocol Extended Reports (RTCP XR)", November 2003.

2.2 Informative references

The following referenced documents are not necessary for the application of the present document but they assist the user with regard to a particular subject area.

[i.1] Cable Television Laboratories, Inc. (CL-SP-MIB-CLABDEF-I09-110210): "CableLabs® Definition MIB Specification", February 10, 2011.

[i.2] ETSI TS 103 161-2: "Access, Terminals, Transmission and Multiplexing (ATTM); Integrated Broadband Cable and Television Networks; IPCablecom 1.5 Part 2: Architectural framework for the delivery of time critical services over Cable Television Networks using Cable Modems".

[i.3] IETF RFC 3410: "Introduction and Applicability Statements for Internet-Standard Management Framework", December 2002.

[i.4] IETF RFC 3411: "An Architecture for Describing Simple Network Management Protocol (SNMP) Management Frameworks", December 2002.

[i.5] IETF RFC 2475: "An Architecture for Differentiated Services", December 1998.

[i.6] ETSI TS 103 161-3: "Access, Terminals, Transmission and Multiplexing (ATTM); Integrated Broadband Cable and Television Networks; IPCablecom 1.5; Part 3: Audio Codec Requirements for the Provision of Bi-Directional Audio Service over Cable Television Networks using Cable Modems".

[i.7] ETSI TS 103 161-5: "Access, Terminals, Transmission and Multiplexing (ATTM); Integrated Broadband Cable and Television Networks; IPCablecom 1.5; Part 5: Dynamic Quality of Service for the Provision of Real Time Services over Cable Television Networks using Cable Modems".

[i.8] IETF RFC 3594: "PacketCable Security Ticket Control Sub-Option for the DHCP CableLabs Client Configuration (CCC) Option", September 2003.

3 Definitions and abbreviations

3.1 Definitions

For the purposes of the present document, the following terms and definitions apply:

authentication: process of verifying the claimed identity of an entity to another entity

Cable Modem Termination System (CMTS): device at a cable headend which implements the DOCSIS® RFI MAC protocol and connects to CMs over an HFC network

embedded MTA: single node that contains both an MTA and a cable modem

encryption key: key used in a cryptographic algorithm to translate the plaintext to ciphertext

encryption: method used to translate plaintext into ciphertext

endpoint: terminal, Gateway or Multipoint Conference Unit (MCU)

Hybrid Fibre/Coaxial (HFC): broadband bi-directional shared media transmission system using fibre trunks between the headend and the fibre nodes, and coaxial distribution from the fibre nodes to the customer locations

Kerberos: secret-key network authentication protocol that uses a choice of cryptographic algorithms for encryption and a centralized key database for authentication

key: mathematical value input into the selected cryptographic algorithm

Media Access Control (MAC): sublayer of the Data Link Layer that normally runs directly over the physical layer

Message Digest 5 (MD5): one-way hash algorithm that maps variable length plaintext into fixed-length (16 byte) ciphertext

Multimedia Terminal Adapter (MTA): interface between a physical voice device, a network interface, CODECs, and all signalling and encapsulation functions required for VoIP transport, class features signalling, and QoS signalling

network management: functions related to the management of data across the network

Standalone MTA (S-MTA): single node that contains an MTA and a non-DOCSIS® MAC (e.g. ethernet)

Type of Service (TOS): 8-bit field of every IP version 4 packet; in a DiffServ domain, the TOS byte is treated as the DiffServ Code Point, or DSCP

User Datagram Protocol (UDP): connectionless protocol built upon Internet Protocol (IP)

3.2 Abbreviations

For the purposes of the present document, the following abbreviations apply:

CM	DOCSIS® Cable Modem
CMS	Call Management Server
CMTS	Cable Modem Termination System
CODEC	COder-DECoder
CSR	Customer Service Representative
DE	Default
DES	Data Encryption Standard
DHCP	Dynamic Host Configuration Protocol
DNS	Domain Name Service
DOCSIS®	Data-Over-Cable Service Interface Specifications
DSC	Dynamic Service Change
DSCP	DiffServ Code Point
DTMF	Dual-tone Multi Frequency (tones)
E-MTA	Embedded MTA
ETSI	European Telecommunications Standards Institute
FQDN	Fully Qualified Domain Name
HFC	Hybrid Fiber/Coaxial
HTTP	HyperText Transfer Protocol
IETF	Internet Engineering Task Force
IP	Internet Protocol
IPsec	Internet Protocol Security
ISDN	Integrated Services Digital Network
KDC	Key Distribution Centre
MAC	Media Access Control
MD5	Message Digest 5
MIB	Management Information Base
MSB	Most Significant Bit
MTA	Multimedia Terminal Adapter
MWD	Maximum Waiting Delay
NCS	Network Call Signalling
NVRAM	Non-Volatile Random Access Memory
OID	Object Identification
PSTN	Public Switched Telephone Network
RFI	The DOCSIS® Radio Frequency Interface specification
RTO	Retransmission Timeout
S-MTA	Standalone MTA
SNMP	Simple Network Management Protocol
TFTP	Trivial File Transfer Protocol
TGS	Ticket Granting Server A sub-system of the KDC used to grant Kerberos tickets

TLV	Type-Length-Value A tuple within a DOCSIS® configuration file
TOS	Type of Service
UDP	User Datagram Protocol

4 Void

5 Background

5.1 Service Goals

Cable operators are interested in deploying high-speed data communications systems on cable television systems. Cable operators have prepared a series of interface documents that will permit the early definition, design, development, and deployment of packet data over cable systems on an uniform, consistent, open, non-proprietary, multi-vendor interoperable basis. The intended service enables voice communications, video, and data services based on bi-directional transfer of Internet protocol (IP) traffic, between the cable system headend and customer locations, over an all-coaxial or hybrid-fibre/coax (HFC) cable network, defined by the data over cable service interface document (DOCSIS®) standard. This is shown in simplified form in figure 1. It is to be noted that the term "DOCSIS®" in the present document is understood to refer to DOCSIS® version 1.1 or later, unless explicitly specified otherwise. Implementations of eCMs will refer to the corresponding DOCSIS® versions and associated requirements in the present document and DOCSIS® documents for compliance.

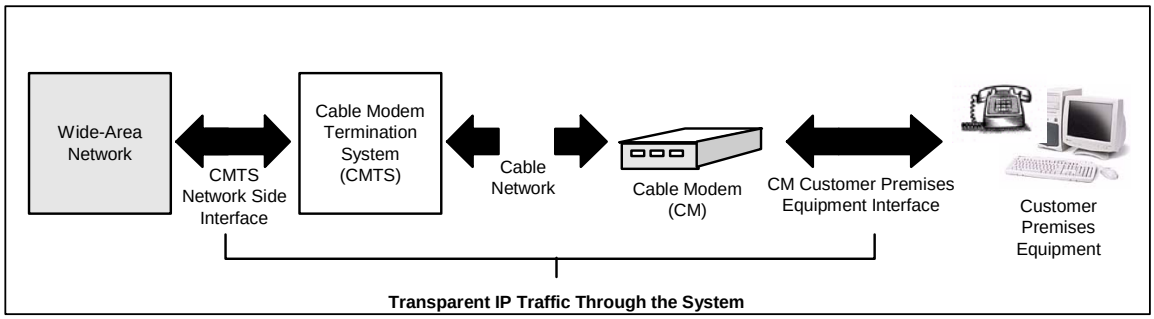

Figure 1: Transparent IP Traffic Through the Data-Over-Cable System

The transmission path over the cable system is realized at the headend by a cable modem termination system (CMTS), and at each customer location by a cable modem (CM). The intent is for operators to transfer IP traffic transparently between these interfaces.

5.2 Document Goals

The goal of the present document is to meet and to satisfy cable member companies, IPCablecom business and technical requirements.

Requirements relevant to device provisioning are:

- A single physical device (e.g. embedded-MTA) will be completely provisioned and managed by a single business entity. This provider may establish business relationships with additional providers for services such as data, voice communications, and other services.
- To provision an E-MTA, both the DOCSIS® and IPCablecom 1.5 provisioning steps must be performed. The eMTA must have its own IP address, different from the IP Address(es) of the eCM. The eMTA must also have its own MAC address, different from the MAC address of the eCM. Furthermore, the E-MTA must in be able to operate in environments where the eMTA IP address may be in the same, or different subnet as the eCM.

- IPCablecom requires a unique FQDN for the MTA-component in the embedded-MTA. This FQDN must be included in the DHCP OFFER and DHCP ACK messages to the MTA-component. IPCablecom makes no additional FQDN requirements on the CM component in the embedded-MTA beyond those required by DOCSIS®. Mapping of the FQDN to IP address must be configured in the network DNS server and be available to the rest of the network.
- IPCablecom 1.5 embedded-MTA provisioning must use DHCP Option-12 and Option-15 to deliver the MTA FQDN to the E-MTA.
- IPCablecom 1.5 embedded-MTA provisioning must support two separate configuration files, a DOCSIS®-specified configuration file for the CM component, and a IPCablecom-specified configuration file for the MTA component.
- The embedded-MTA is outside the IPCablecom network trust boundary as defined in the IPCablecom Architecture Specification [i.2].
- IPCablecom 1.5 E-MTA must support DOCSIS® software download as defined in the corresponding eCM DOCSIS® Specification. A single DOCSIS® software download must be used to upgrade software for both the eCM and the eMTA.
- IPCablecom 1.5 must support use of SNMPv2c co-existence for network management operations for devices provisioned under the Basic Flow or the Hybrid Flow and SNMPv3/v2 co-existence for network management operations when the device is provisioned under the Secure Flow.
- IPCablecom 1.5 embedded-MTA provisioning minimizes the impact to DOCSIS® devices (CM and CMTS) in the network.
- Standard server solutions (TFTP, SNMP, DNS, etc.) are preferable. It is understood that an application layer may be required on top of these protocols to coordinate IPCablecom 1.5 embedded-MTA provisioning.
- Where appropriate, the DOCSIS® management protocols are supported (for example SNMP).

5.3 IPCablecom Reference Architecture

Figure 2 shows the reference architecture for the IPCablecom 1.5 Network. Refer to the IPCablecom Architecture Specification [i.2] for more detailed information on this reference architecture.

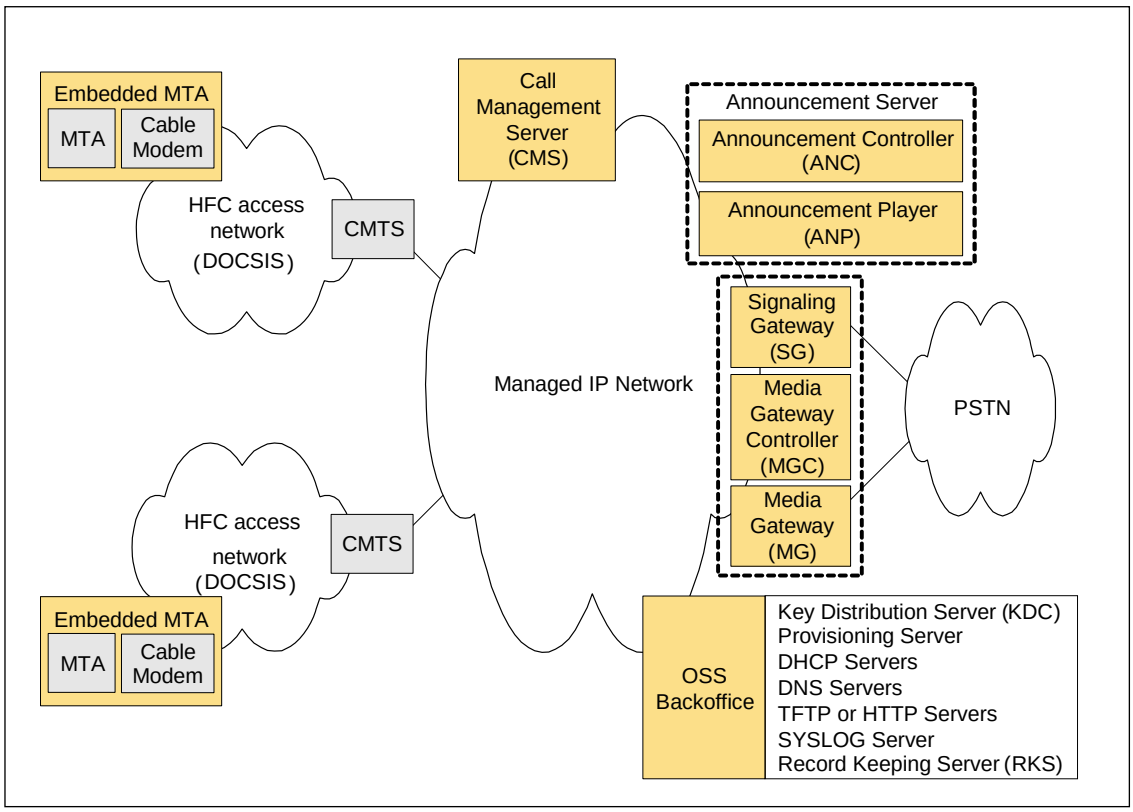

Figure 2: IPCablecom 1.5 Network Component Reference Model

5.4 Components and Interfaces

Figure 3 represents the components and interfaces discussed in the present document.

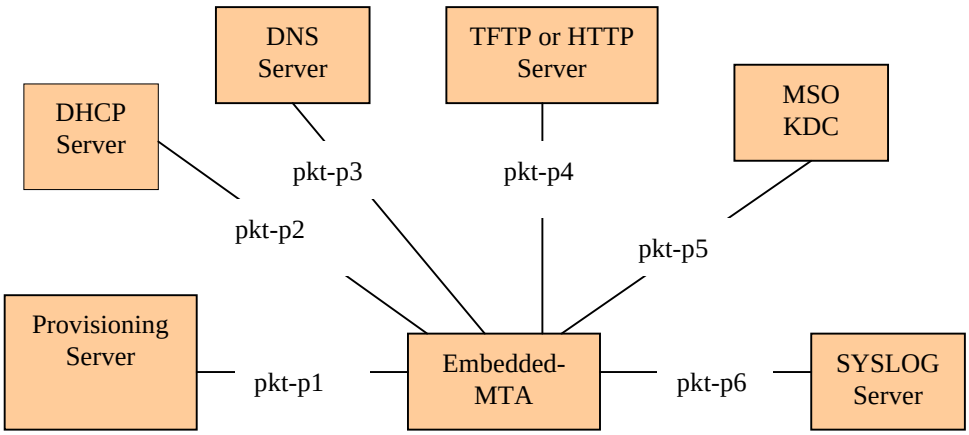

Figure 3: IPCablecom Provisioning Interfaces

5.4.1 MTA

The MTA must conform to the following requirements during the provisioning sequence.

5.4.1.1 MTA Security Requirements

The MTA must conform to the following security requirements during the Secure Flow provisioning sequence:

- The MTA device MIB is structured to represent the assignment of MTA endpoints to a CMS. For more information on the security association between an MTA and a CMS refer to [5].
- CMS Kerberos Principal Name is not explicitly configured in the MTA endpoints. The MTA must be able to determine the CMS Kerberos Principal Name based on the CMS FQDN, as specified in [5].
- For each unique pair of CMS Kerberos principal Name / Kerberos Realm assigned to an endpoint, the MTA must obtain a single Kerberos ticket [5]. If the MTA already has a valid Kerberos ticket for that CMS, the MTA must not request an additional Kerberos ticket for that CMS. (Unless the expiration time of the current Kerberos ticket <= current time + PKINIT Grace Period, in which case the MTA must obtain a fresh ticket for the same CMS.)
- In the case that a CMS FQDN maps to multiple IP addresses, the MTA must initially establish a pair of IPsec Security Associations with one of the IP addresses returned by the DNS server. The MTA may also initially establish IPsec Security Associations with the additional CMS IP addresses. Please refer to [5] for more information.
- If the MTA already has a pair of active Security Associations (inbound and outbound) with a particular CMS IP address, the MTA must not attempt to establish additional Security Associations with the same IP address.

During the provisioning sequence, there are no specific security requirements for the Basic Flow or the Hybrid Flow.

5.4.1.2 MTA SNMP Requirements

The MTA must conform to the following SNMPv3 requirements during the Secure Flow provisioning sequence:

- MTA SNMPv3 security is separate and distinct from DOCSIS® SNMPv3 security. USM security information (authentication and privacy keys, and other USM table entries) is setup separately.
- SNMPv3 initialization must be completed prior to the provisioning enrolment inform.
- In Secure Flow, the MTA must support SNMPv3 and SNMPv2c based device management as defined in [8] and [32].
- The MTA must conform to the following SNMPv2c requirements during the Hybrid Flow or Basic Flow provisioning sequence:
  - SNMPv2c initialization must be completed immediately after the DHCP phase.
  - SNMPv2c based device management as defined in [32].

The MTA must implement the Management Information Base (MIB) modules as described in [29] and specified in [2], [3], [12], [16] and [i.5]. If it also implements the IETF MIB modules as specified in [21], [22] and [23], the MTA must ensure that both sets of MIB modules can co-exist without any conflicting or ambiguous behaviour. MTAs implementing multiple MIB environments must support the DHCP option 'CL_V4_PACKETCABLE_MIB_ENV_OPTION' as specified in clauses 7 and 8.8.

It is understood that any MIB module or MIB Object references in the present document refer to either MIB modules unless explicitly specified otherwise or the MIB Object is specified in only one of them.

5.4.2 Provisioning Server

The Provisioning Server is made up of the following components:

- Provisioning Application - The Provisioning Application is responsible for coordinating the embedded-MTA provisioning process. This application has an associated SNMP Entity.

- Provisioning SNMP Entity - The provisioning SNMP entity must include a trap/inform handler for provisioning enrolment and the provisioning status traps/informs as well as a SNMP engine for retrieving device capabilities and setting the Configuration filename and access method. Refer to the IPCablecom MTA MIB ([2] and [21]) for a description of the MIB accessible MTA attributes.

- Provisioning Server - The Provisioning Server must examine the "Device MIB Support" capability of the MTA (specified in clause 10.23) to choose the MIB modules that can be used to specify the MTA configuration file.

The interface between the Provisioning Application and the associated SNMP Entity is not specified in IPCablecom 1.5 and is left to vendor implementation. The interface between the Provisioning Server and the TFTP Server is not specified in IPCablecom 1.5 and is left to vendor implementation.

5.4.3 MTA to Telephony Syslog Server

The IPCablecom MTAs must implement Management Event Mechanism as per [14], which includes the support for Syslog server.

The IPCablecom MTAs must also implement all the IPCablecom Provisioning Management Events described in Annex A of [14].

5.4.4 MTA to DHCP Server

This interface identifies specific requirements in the DHCP server and the client for IP assignment during the MTA initialization process.

- Both the DHCP server and the embedded-MTA must support DHCP option code 6, 7, 12, 15, 43, 60 and DHCP option code 122 (defined in [11]). Option code 12 (Host Name) and 15 (Domain Name) must form a Fully Qualified Domain Name and must be resolvable by the DNS server.

- The DHCP server must accept and support broadcast and unicast messages per RFC 3396 [10] from the MTA DHCP client.

- The DHCP server must include the MTA's assigned FQDN in the DHCP OFFER and DHCP ACK messages to the MTA-component of the embedded-MTA. Refer to RFC 2131 [1] for details describing the DHCP OFFER message.

5.4.5 MTA to Provisioning Application

This interface identifies specific requirements for the Provisioning Application to satisfy MTA initialization and registration. The Provisioning Application requirements are:

- The MTA must generate a correlation ID - an arbitrary value that will be exchanged as part of the device capability data to the Provisioning Application. This value is used as an identifier to correlate related events in the MTA provisioning sequence.

- The Provisioning Application must provide the MTA with its MTA configuration data file. The MTA configuration file is specific to the MTA-component of the embedded-MTA and separate from the CM-component's configuration data file.

- The configuration data file format is TLV binary data suitable for transport over the specified TFTP or HTTP access method.

- The Provisioning Application must have the capability to configure the MTA with different data and voice service providers.

- The Provisioning Application must use only SNMPv3 to provision devices in the Secure Flow. The support of the Basic and Hybrid Flows is optional for the Provisioning Application. If the Basic and Hybrid Flows are supported, the Provisioning Application must use only SNMPv2c to provision devices in the Hybrid or Basic Flow.

- The Provisioning Application must provide SNMPv3 and SNMPv2c for device management.
- The Provisioning Application must support online incremental device/subscriber provisioning using SNMP.
- MTA must Specify all of its Capabilities in DHCP Option-60 in accordance with clause 10.
- Provisioning Application must not assume any Capabilities, which do not have default values. In case if capabilities supplied by the MTA are not consistent in format and/or in number and/or in values, the Provisioning Application must use the other means to identify the MTA's capabilities (e.g. SNMPv3 if possible).

5.4.6 MTA to CMS

Signalling is the main interface between the MTA and the CMS. Refer to the IPCablecom Signalling Specification [4] for a detailed description of the interface.

The CMS must accept signalling and bearer channel requests from a MTA that has an active security association.

The CMS must not accept signalling and bearer channel requests from a MTA that does not have an active security association unless provisioned to do so with information corresponding to the "pktcMtaDevCmsIpsecCtrl" MIB Object.

5.4.7 MTA to Security Server (KDC)

The interface between the MTA and the Key Distribution Center (KDC) must conform to the IPCablecom Security Specification [5].

AP-REQ/REP exchange back off and retry mechanism of the Kerberized SNMPv3 key negotiation defined in [5] is controlled by the values delivered by DHCP Option 122 sub-option 5 (see clause 8.1.4).

AS-REQ/REP exchange backoff and retry mechanism of the Kerberized SNMPv3 key negotiation defined in [5] is controlled by the values delivered by DHCP Option 122 sub-option 4 (see clause 8.1.3) or by the default values of the corresponding MIB objects in the Realm Table if sub-option 4 is not present in the DHCP Option 122.

5.4.8 MTA and Configuration Data File Access

The present document allows for more than one access method to download the configuration data file to the MTA.

- The MTA must support the TFTP access method for downloading the MTA configuration data file.
- The MTA may support HTTP access method for downloading the MTA configuration data file.
- The Provisioning Server must provide the MTA with the URL-encoded TFTP/HTTP server address and configuration filename via a SNMPv3 SET for the Secure Flow. The Provisioning Server must provide the MTA with the URL-encoded TFTP/HTTP server address via an SNMPv2c SET if it supports the Hybrid Flow provisioning mode. The Basic Flow does not require an SNMP SET to get the configuration file; the Provisioning Server must provide the MTA with the TFTP/HTTP server address in the DHCP "file" and "siaddr" fields if it supports the Basic Flow provisioning mode. For additional information refer to clause 7.3.

5.4.9 DOCSIS® extensions for MTA Provisioning

The present document requires that the following additions to DOCSIS® flows for MTA auto-provisioning be supported:

- eCMs that can obtain IP configuration information from multiple DHCP servers must use the primary DHCP server - designated to provide the eCM configuration information - for obtaining IPCablecom specific options (refer to [i.7] for more information).
- eCMs relying on DHCPv4 for IPCablecom specific information must implement DHCP option code 122 (as specified in [26]) and communicate the Telephony Service Provider's DHCPv4 Server information to the eMTA (as per clause 7).

- eCMs relying on DHCPv6 for IPCablecom specific information must implement the DHCP option code OPTION_VENDOR_OPTS(17) (as specified in [20]) and the "CableLabs Client Configuration" DHCPv6 option code CL_OPTION_CCC (as specified in [19]) to communicate the Telephony Service Provider's DHCPv4 Server information to the eMTA.

6 Provisioning Overview

Provisioning is a subset of configuration management control. The provisioning aspects include, but are not limited to, defining configurable data attributes, managing defined attribute values, resource initialization and registration, managing resource software, and configuration data reporting. The resource (also referred to as the managed resource) always refers to the MTA device. Further, the associated subscriber is also referred to as a managed resource.

6.1 Device Provisioning

Device provisioning is the process by which an embedded-MTA device is configured to support voice communications service.

Device provisioning involves the MTA obtaining its IP configuration required for basic network connectivity, announcing itself to the network, and downloading of its configuration data from its provisioning server.

When the device is provisioned using the "Secure Flow", the MTA device must be able to verify the authenticity of the configuration file it downloads from the server. The "Secure Flow" generated configuration file is "signed" and may be "sealed". Please refer to [5] for further information.

Please refer to clause 5.4.1 for provisioning rules related to security associations.

When the device is provisioned using the Basic Flow or the Hybrid Flow, a content integrity verification check must be conducted on the configuration file by the MTA. For details refer to clause 9.1.

6.2 Endpoint Provisioning

Endpoint provisioning is when a provisioned MTA authenticates itself to the CMS, and establishes a security association with that server. This allows subsequent call signalling to be protected under the established security association.

The MTA must follow the requirements defined in the IPCablecom Security Specification [5] for NCS Kerberized Key Management, independently of the provisioning flow (Secure, Hybrid or Basic Flow) the MTA was provisioned with.

6.3 Secure Flow Provisioning State Transitions

Figure 4 represents logical device states and the possible transitions across these logical states. This representation is for illustrative purposes only, and is not meant to imply a specific implementation. The following MTA state transitions do not specify the number of retry attempts or retry time out values.

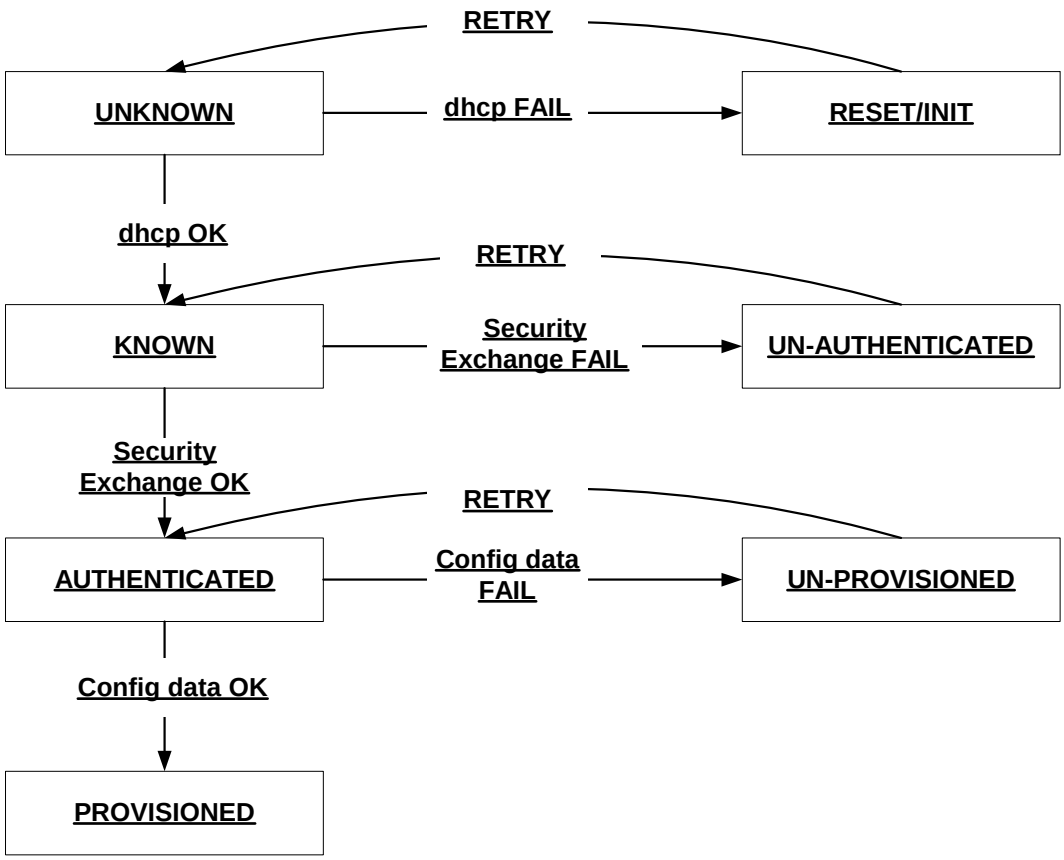

Figure 4: Device States and State Transitions for Secure Flow Provisioning

6.4 Basic and Hybrid Flow Provisioning State Transitions

Figure 5 represents logical device states and the possible transitions across these logical states. This representation is for illustrative purposes only, and is not meant to imply a specific implementation. The following MTA state transitions do not specify the number of retry attempts or retry time out values.

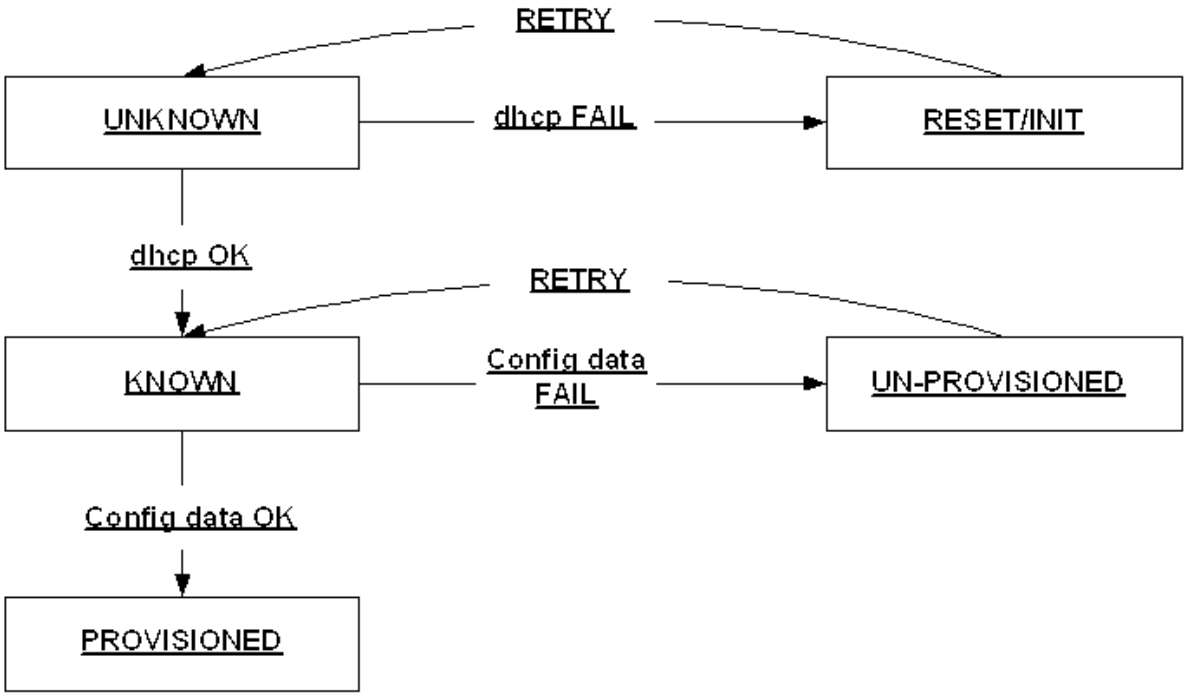

Figure 5: Device States and State Transitions for Basic and Hybrid Flow Provisioning

7 Provisioning Flows

An IPCablecom MTA is provisioned via one of three provisioning flows:

- The Secure Flow supports Kerberos mutual authentication between the MTA and the provisioning system, as well as Kerberized SNMPv3 messaging. The Secure Flow must be supported by IPCablecom MTAs and the Provisioning Applications.
- The Basic Flows are a simplified DOCSIS®-like provisioning flows with no Kerberos or SNMPv3 security and no SNMP enrollment via SNMP INFORM. The Basic Flows should be supported by IPCablecom MTAs and Provisioning Applications.
- The Hybrid Flows are essentially the Secure flow with the Kerberos message exchanges removed, and SNMPv2c substituted for SNMPv3. The Hybrid Flows should be supported by IPCablecom MTAs and Provisioning Applications.

Any mention of SNMP in the present document without a specific reference to the SNMP protocol version must be interpreted as follows:

- For the Secure Flow, the MTA must support 'SNMPv3 only' for Provisioning and SNMPv3/v2c co-existence for Network Management and/or Monitoring operations. The SNMPv3/v2c co-existence must be supported and is configured using the values of TLV-38, or TLV-11 and TLV64 in the MTA configuration file.
- For the Hybrid or Basic Flows, the MTA must support SNMPv2c for Provisioning, Network Management and/or Monitoring operations. The level of SNMPv2c access must be supported according to the values of TLV38 or TLV-11 and TLV64 in the MTA configuration file.

An MTA can also be configured with additional SNMPv2c targets via its configuration file by using TLV38 or TLV11 and TLV64.

An MTA is commanded to execute a specific flow via the contents of DHCP option 122 sub-option 6, as described in clause 8.1.5. Each of these flows begin with a common set of flow steps.

An MTA is required to implement the IPCablecom specified MIBs, and optionally may implement the IETF MIBs. For an MTA supporting both, the operator may wish to provide a preference for usage. An MTA that supports multiple MIB environments must request the DHCPv4 option "CL_V4_PACKETCABLE_MIB_ENV_OPTION (4)", as specified in [19]. The Provisioning Server may indicate an operator preference by responding with the DHCPv4 option "CL_V4_PACKETCABLE_MIB_ENV_OPTION (4)". The MTA requirements associated with this option are specified in clause 8.8.

In all the provisioning flows indicated in this clause, the following requirements apply:

- If an MTA encounters DHCP options that it does not recognize - in the DHCP OFFER or DHCP ACK messages - it must ignore such options and proceed with provisioning as though they were not provided (i.e. MTA operation is unaffected).
- If an IPCablecom Provisioning Server receives SNMP INFORMs from MTAs that contain Object Identifiers that it does not recognize, it should acknowledge them. Further, upon receiving any SNMP notifications containing unrecognized Object Identifiers, the Provisioning Server must comply with the behaviour specified in the present document (i.e. Provisioning Server operation is unaffected).

7.1 Backoff, Retries, and Timeouts

Backoff mechanisms help the network to throttle device registration during a typical or mass registration condition when the MTA client requests are not serviced within the protocol specified timeout values. The details of provisioning behaviour under mass-registration is beyond the scope of IPCablecom 1.5, however this clause provides the following recommendations and requirements.

- The recommendation for the throttling of registration may be based on DOCSIS® CM registration.
- The MTA must follow DHCP [1], and HTTP documents for the timeout and retry mechanisms. It is recommended to follow RFC 3413 [7] for SNMP timeout and retry mechanisms.
- The MTA must use an adaptive timeout for TFTP as specified in the DOCSIS® Specification.
- The MTA must follow backoff and retry recommendations that are defined in the Security Specification [5] for the security message flows.
- In all Provisioning Flows (Secure, Hybrid and Basic) described in clauses 7.2, 7.3 and 7.4:
    - Provisioning timer must start immediately after the receipt of DHCP ACK and must end with the completion of TFTP/HTTP configuration file response.
    - In case the provisioning timer expires before the completion of TFTP/HTTP configuration file response, the MTA must return to MTA-1.
    - MTA must not wait until the Provisioning Timer expires before acting on each Provisioning step's failure condition. For example, in Secure Flow, if step MTA-19 fails, the MTA will not wait until the Provisioning Timer expires but will return to MTA-1 immediately when the failure condition is discovered.
- In the Secure Provisioning Flow - if a failure occurs in any of the steps related to the PROV_SNMP_ENTITY (MTA13, MTA14, MTA15, MTA19) before the MTA obtains the 'Device configuration file - and the MTA resolved multiple IP addresses for the PROV_SNMP_ENTITY (FQDN received in Option 122 Sub option 3), then it must retry the steps with all the resolved IP addresses before returning to MTA1, unless directed otherwise by [5]. However, it is to be noted that once the MTA selects a resolved IP address for use in MTA13, it must use the same IP address in steps MTA15 and MTA25.
- In the Hybrid Provisioning Flow - if a failure occurs in any of the steps related to the PROV_SNMP_ENTITY (H- MTA-15, H-MTA-19) before the MTA obtains the Device configuration file - and the MTA resolved multiple IP addresses for the PROV_SNMP_ENTITY (FQDN received in Option 122 sub-option 3), then it must retry the steps with all the resolved IP addresses before returning to MTA1. However, it is to be noted that once the MTA selects a resolved IP address for use in H-MTA-15, it must use the same IP address in H-MTA-25.

7.2 Embedded-MTA Power-On Initialization Flow (Secure Flow)

Following is the mandatory message flow that the embedded-MTA device must follow during power-on initialization (unless stated explicitly otherwise). It is understood that these flows do not imply implementation or limit functionality.

Although these flows show the MTA configuration file download from a TFTP Server, the descriptive text details the requirements to support the MTA configuration file download from a HTTP Server.

NOTE: In the flow details that certain steps may appear to be a loop in the event of a failure. In other words, the step to proceed to if a given step fails, is to retry that step again. However, it is recommended that if the desired number of backoff and retry attempts does not allow the step to successfully complete, the device detecting the failure should generate a failure event notification.

In the flow details below, the calculation of the Hash and the Encryption/Decryption of the MTA's Configuration File must follow requirements in [5].

The provisioning flow for an eCM utilizing DHCPv4 for IP configuration is shown on figure 6 and explained in clause 7.2.1. The provisioning flow for an eCM utilizing DHCPv6 for IP configuration is shown on figure 7 and explained in clause 7.2.2. The provisioning flows for an eMTA remain the same, irrespective of the eCM provisioning mode.

7.2.1 Embedded-MTA Secure Power-on Initialization Flow (IPv4 eCM)

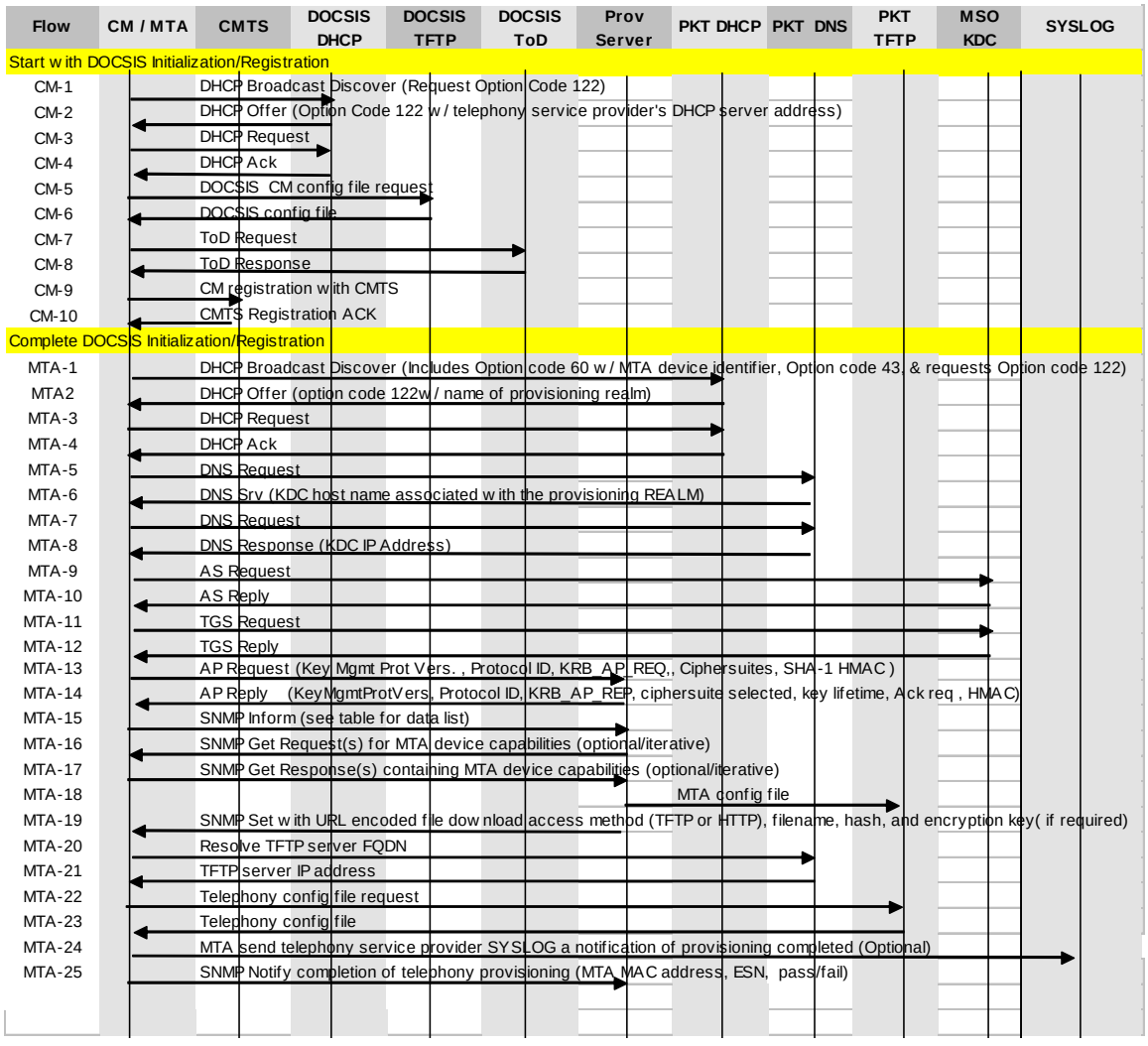

Figure 6: Embedded-MTA Secure Power-on Initialization Flow (IPv4 eCM)

Table 1

Flow	Embedded-MTA Power-On Initialization Flow Description	Normal Flow Sequencing	Must proceed to here if this step fails
See note 1.			
CM1	As defined in the DOCSIS® specified registration sequence, the client device begins device registration by having the cable modem component send a broadcast DHCP discover message. This message includes Option code 60 (Vendor Specific Option) in the format "DOCSIS®y.z::xxxxxxx", where y.z is the DOCSIS® version supported by the eCM". This message must request Option 122 in Option 55, the request parameter list. The remainder of this message must conform to the DHCP discover data as defined in the DOCSIS® Specification.	Initial must Step in Sequence	Per DOCSIS®
CM2	The DOCSIS® DHCP Server, if it has been configured to support MTA devices, must include Option Code 122 with sub-option 1 and, possibly, sub-option 2 as per clause 8.1. If it is configured to prevent the MTA portion of the device from provisioning, then sub-option 1 in Option Code 122 must contain a DHCP server address of value 0.0.0.0. DOCSIS® DHCP Servers without any prior knowledge of MTA devices may respond with DHCP OFFERS without including option 122.	CM2 must occur after CM1 Completion	Per DOCSIS®
CM3	Upon receiving a DHCP OFFER, the CM must check for the requested option 122. If it is not present then it must retry the DHCP DISCOVER process (CM1) exponentially for 3 attempts (e.g. 2, 4, 8 second intervals). Upon failing to receive any DHCP OFFER with option 122 after the exponential retry mechanism it must consider OFFERs without option code 122 and accept one of them as per the DHCP Specification [1]. The client device (CM) must then send a DHCP REQUEST broadcast message to the DHCP server whose OFFER is accepted as specified in the DHCP Specification [1].	CM3 must occur after CM2 Completion	Per DOCSIS®
CM4	The DHCP server sends the client device cable modem component a DHCP ACK message to confirm acceptance of the offered data. Upon receiving the DHCP ACK, the CM must check again for option 122. The absence of option 122 in the DHCP ACK message, that was accepted by the CM, implies that it must not initialize the embedded MTA. The presence of option 122 implies that it must initialize the MTA and pass sub-option 1 and, possibly, sub-option 2. If the option content of this DHCP ACK differs with the preceding DHCP OFFER, the option content of this DHCP ACK must be treated as authoritative (per RFC 2131 [1]).	CM4 must occur after CM3 Completion	Per DOCSIS®
CM5-CM 10	The client device's cable modem component completes the remainder of the DOCSIS® specified registration sequence. This includes downloading the DOCSIS® configuration file, requesting time of day registration, and registering with the CMTS.	CM5 - CM10 must occur after CM4 completion	Per DOCSIS®
MTA1	DHCP Broadcast Discover The MTA must send a broadcast DHCP DISCOVER message. This message must include option code 60 (Vendor Specific Option) in the format "pktc1.5:xxxxxx". The MTA must include the DHCP option code 43 in the DHCP DISCOVER message as defined in clause 8.5. The MTA must request in DHCP option 55 the following: 1, 3, 6, 7, 12, 15, and 122 options. If the CM DHCP option code 122 sub-option 1 (passed by the CM to the MTA) contains a DHCP server of value of 0.0.0.0, then the MTA must not attempt to provision and must remain dormant until it is reinitialized by the CM.	MTA1 must not occur before completion of CM4	If failure per DHCP protocol repeat MTA1

Flow	Embedded-MTA Power-On Initialization Flow Description	Normal Flow Sequencing	Must proceed to here if this step fails
MTA2	DHCP OFFER The MTA may receive multiple DHCP OFFERs (during its wait period as per RFC 2131 [1]. The following requirements apply to the MTA and/or the Provisioning Applications. 1) The MTA must only accept a valid DHCP OFFER message. A valid DHCP OFFER must be sent by the primary or secondary DHCP servers returned in DHCP option code 122 sub-options 1 and 2 as obtained by the E-MTA via the CM provisioning step CM-4. A valid DHCP OFFER must also include the following options: 1, 3, 6, 7, 12, 15, 122 with DHCP option 122 sub-options 3 and 6. DHCP option 122 may contain the additional sub-options 4, 5, 7, 8, and 9. 2) If the DHCP option 122 sub-option 6 returned by a valid DHCP server indicates that the Basic or Hybrid flow must be performed, the MTA must ignore the DHCP option 122 sub-options 4, 5, 7 and 9 if they are present. 3) If the DHCP option 122 sub-option 6 returned by a valid DHCP server indicates that the Basic Flow must be performed, the Provisioning Server must include the configuration file location in the 'siaddr' and 'file' fields in the DHCP responses. 4) If the DHCP option 122 sub-option 6 returned by a valid DHCP server indicates the Secure flow must be performed, the MTA must process the DHCP option 122 sub-options 4, 5, 7, and 9. The MTA next applies the following rules to the set of valid DHCP OFFERs: a) The MTA must check the value of the DHCP option 122 sub-option 3. If all valid OFFERs contain 0.0.0.0 in DHCP option 122 sub-option 3, then the MTA must not further the DHCP process and it must shutdown until it is reinitialized. Otherwise, the MTA must further restrict its set of valid OFFERs to those with a non-zero value in the DHCP option 122 sub-option 3. b) The MTA must check the value of the DHCP option 122 sub-option 6 for indication of the Secure Flow. If no valid DHCP OFFER message directs the MTA to the Secure flow, the MTA must retry the DHCP DISCOVER process (MTA-1) exponentially for 3 attempts (e.g. 2, 4, 8 second intervals). Upon failing to receive any valid DHCP OFFER indicating the Secure flow, the MTA must select, a valid Hybrid Flow DHCP OFFER, or a valid Basic Flow OFFER in that order. If no valid DHCP OFFER is received, the MTA must fail the corresponding provisioning flow step (see note 2).	MTA2 must occur after MTA1 completion	If failure per DHCP protocol return to MTA1
MTA3	DHCP Broadcast REQUEST Once the MTA has selected a valid DHCP OFFER, the MTA must send a DHCP REQUEST broadcast message to accept the DHCP OFFER per [1].	MTA3 must occur after MTA2 completion	If failure per DHCP protocol return to MTA1
MTA4	DHCP ACK The DHCP server sends a DHCP ACK message to the MTA. The DHCP ACK message must include all options and sub-options which had been sent in MTA 2 (DHCP OFFER). If the option and sub-option values of this DHCP ACK differ with the preceding DHCP OFFER (MTA-2), the option and sub-option values of this DHCP ACK must be treated as authoritative (per RFC 2131 [1]). If the DHCP ACK is not valid as per the criteria established in MTA 2, the MTA must fail this step.	MTA4 must occur after MTA3 completion	If failure per DHCP protocol return to MTA1
(see note 3)			

Flow	Embedded-MTA Power-On Initialization Flow Description	Normal Flow Sequencing	Must proceed to here if this step fails
MTA5	DNS Srv Request The MTA requests the cable operator KDC host name for the Kerberos realm.	MTA5 must occur after MTA4 completion	MTA1
MTA6	DNS Srv Reply Returns the cable operator KDC host name associated with the provisioning REALM.	MTA6 must occur after MTA5 completion	MTA1
MTA7	DNS Request The MTA now requests the IP Address of the cable operator KDC.	MTA7 must occur after MTA6 completion	MTA1
MTA8	DNS Reply The DNS Server returns the IP Address of the cable operator KDC.	MTA8 must occur after MTA7 completion	MTA1
MTA9	AS Request The AS Request message is sent to the cable operator KDC to request a Kerberos ticket.	If MTA9 occurs, it must occur after MTA8 completion.	MTA1 The failure conditions are defined by the Security Specification [5].
MTA10	AS Reply The AS Reply Message is received from the cable operator KDC containing the Kerberos ticket. NOTE: The KDC must map the MTA MAC address to the FQDN before send the AS Reply.	MTA10 must occur after MTA9 completion	MTA1
(see notes 4 to 7)			
MTA11	TGS Request If MTA obtained TGT in MTA10, the TGS Request message is sent to the cable operator KDC.	MTA11 occurs, it must occur after MTA10 completion	MTA1
MTA12	TGS Reply The TGS Reply message is received from the cable operator KDC.	MTA12 must occur after MTA11 completion	MTA1
MTA13	AP Request The AP Request message is sent to the Provisioning Server to request the keying information for SNMPv3.	MTA13 must occur after MTA12 or MTA 10 completion	MTA1 The failure conditions are defined by the Security Specification [5].
MTA14	AP Reply The AP Reply message is received from the Provisioning Server containing the keying information for SNMPv3 (see note 8).	MTA14 must occur after MTA13 completion	MTA1

Flow	Embedded-MTA Power-On Initialization Flow Description	Normal Flow Sequencing	Must proceed to here if this step fails
MTA15	SNMP Enrollment INFORM The MTA must send an SNMPv3 Enrollment INFORM to the PROV_SNMP_ENTITY (specified in the DHCP option 122 sub-option 3). The SNMP INFORM must contain a "PktcMtaDevProvisioningEnrollment object as defined in [2]. The PROV_SNMP_ENTITY notifies the Provisioning Application that the MTA has entered the management domain.	MTA15 must occur after MTA14 completion	If failure per SNMP protocol return to MTA1. SNMP server must send response to SNMP-INFORM.
(see note 9)			
MTA16	SNMPv3 GET Request (Optional) If any additional MTA device capabilities are needed by the PROV_APP, the PROV_APP requests these from the MTA via SNMPv3 Get Requests. This is done by having the PROV_APP send the PROV_SNMP_ENTITY a "get request" Iterative: The PROV_SNMP_ENTITY sends the MTA one or more SNMPv3 GET requests to obtain any needed MTA capability information. The Provisioning Application may use a GETBulk request to obtain several pieces of information in a single message.	MTA16 is optional, can occur after MTA15 completion	N/A
MTA17	SNMPv3 GET Response Iterative: MTA sends the PROV_SNMP_ENTITY a response for each GET Request. After all the Gets, or the GetBulk, finish, the PROV_SNMP_ENTITY sends the requested data to the PROV_APP.	MTA17 must occur after MTA16 completion if MTA16 is performed	N/A
MTA18	This Protocol is not defined by IPCablecom. The PROV_APP may use the information from MTA16 and MTA17 to determine the contents of the MTA Configuration Data file. Mechanisms for sending, storing and, possibly, creating the configuration file are outlined in MTA19.	MTA18 should occur after MTA15 completion unless MTA16 is performed, then it should be after MTA17 has completed	N/A
MTA19	SNMPv3 SET The PROV_APP may create the configuration file at this point, or send a predefined one. A hash must be run on the contents of the configuration file. The configuration file may be encrypted The hash and the encryption key (if the configuration file is encrypted) must be sent to the MTA. The PROV_APP must store the configuration file on the appropriate TFTP server. The PROV_APP then instructs the PROV_SNMP_ENTITY to send an SNMP SET message to the MTA containing the following varbindings (defined in [2]): pktcMtaDevConfigFilepktcMtaDevProvConfigHash and pktcMtaDevProvConfigKey (This must not be included if the MTA configuration file is unencrypted) (see notes 10 to 12).	MTA19 must occur after MTA18 completion	If failure per SNMP protocol return to MTA1.

Flow	Embedded-MTA Power-On Initialization Flow Description	Normal Flow Sequencing	Must proceed to here if this step fails
MTA20	DNS Request If the URL-encoded access method contains a FQDN instead of an IPv4 address, the MTA must use the service provider network's DNS server to resolve the FQDN into an IPv4 address of either the TFTP Server or the HTTP Server.	MTA20 must occur after MTA19 completion if FQDN is used	If failure per DNS protocol return to MTA1.
MTA21	DNS Reply DNS Response: DNS server returns the IP address against MTA20 DNS request.	MTA21 must occur after MTA20 completion if FQDN is used	If failure per DNS protocol return to MTA1.
MTA22	TFTP/HTTP Configuration file Request The MTA must perform either the TFTP or HTTP protocol exchange, as specified in step S-MTA-19, to download its configuration file. For specific details of each protocol, see [9] and [27].	MTA22 must occur: After MTA-19 if DNS resolution is not required.· After MTA-21 if DNS resolution is required	If failure per TFTP or HTTP protocols, return to MTA1.
MTA23	TFTP/HTTP Configuration file Request The TFTP/HTTP Server must send the requested configuration file to the MTA. For specific details of each protocol, see [9] and [27]. The hash of the downloaded configuration file is calculated by the MTA and compared to the value received in step MTA-19. If the hashes do not match, the MTA must fail this step. If encrypted, the configuration file must be decrypted. Refer to clause 9.1 for MTA configuration file contents.	MTA23 must occur after MTA22 completion	If the configuration file download failed per TFTP or HTTP protocols, return to MTA1. Otherwise, proceed to MTA24 or MTA25, and send the failed response if the MTA configuration file itself is in error.
MTA24	SYSLOG Notification If a SYSLOG server is configured and enabled as part of the Provisioning Process (Refer to step MTA-2 for DHCP Options and [13], [14], and [23], for configuration using the MEM-MIB), then the MTA must send the voice service provider's SYSLOG a "provisioning complete" event indicating the status of the provisioning operation. This notification will include the pass-fail result of the provisioning operation. The general format of this notification is as defined in clause 5.4.3.	MTA24 must occur after MTA23 completion if SYSLOG is configured	The MTA may retry this step before proceeding to MTA25.
MTA25	SNMP INFORM The MTA must send the PROV_SNMP_ENTITY(specified in DHCP option 122 sub-option 3) an SNMP INFORM containing a "provisioning complete" notification. The receipt of the inform is acknowledged by the response message as defined in RFC 3414 [8]. The SNMP INFORM must contain a "pktcMtaDevProvisioningStatus" MIB object (see notes 13 and 14).	MTA25 must occur after MTA24 if SYSLOG is used, otherwise must occur after MTA23 completion	MTA may generate a Provisioning Failure event notification to the Service Provider's Fault Management server. Provisioning process stops; manual interaction required. SNMP server must send response to SNMP-INFORM.

Flow	Embedded-MTA Power-On Initialization Flow Description	Normal Flow Sequencing	Must proceed to here if this step fails

NOTE 1: Refer to the DOCSIS® Specifications for a complete description of flows CM1- CM10.
NOTE 2: In the case of Secure Flow, if an MTA supports TGTs and receives the DHCP option 122 sub-option 7 set to a FALSE value, it must not request TGTs. If an MTA supports TGTs and receives the DHCP option 122 sub-option 7 set to a TRUE value, it must request TGTs. MTAs that do not support TGTs must ignore the DHCP option 122 sub-option 7.
NOTE 3: The provisioning flow forks into one of three directions as follows:
- If the MTA4 DHCP ACK indicates the Basic Flow, the MTA must proceed to flow step BMTA-22 described in clause 7.3.
- If the MTA4 DHCP ACK indicates the Hybrid Flow, the MTA must proceed to flow step HMTA-15 described in clause 7.4.
- Otherwise, the Secure Flow is indicated and the MTA must proceed to step MTA5 below.

NOTE 4: Flows MTA11- MTA12 are optional in some cases, please reference the Security Specification [5].
NOTE 5: SNMPv3 entity (FQDN) must be resolved to an IP address anywhere during flows MTA-5 to MTA12.
NOTE 6: If an IP address is provided in the Additional information field of the DNS-SRV response (MTA6), MTA may use the same and skip the flows MTA7 and MTA8.
NOTE 7 : If the MTA has valid provisioning application server ticket saved in NVRAM, then it must skip the flows MTA5 to MTA12 in successive MTA resets (flows MTA1 to MTA25).
NOTE 8: The SNMPv3 keys must be established before the next step using the information in the AP Reply.
NOTE 9: The provisioning server can reset the MTA at this point in the flows. The MTA is part of the security domain and must respond to management requests, the SNMP INFORM of MTA15 is the indicator, see clause 5.4.1.2.
NOTE 10: In the case of file download using the HTTP access method, the filename must be URL-encoded with a URL format compliant with RFC 2616 [30] with exception stated below in note 12.
NOTE 11: In the case of file download using the TFTP access method, the filename must be URL-encoded with a URL format compliant with RFC 3617 [31] with exception stated below in note 12.
NOTE 12: MTA must accept IPv4 addresses embedded in URL encoded format with or without square brackets.
NOTE 13: At this stage, the MTA device provisioning data is sufficient to provide any minimal services as determined by the service provider.*
NOTE 14: 2. Depending on the TLV38 configuration, there might be multiple SNMP INFORMs sent to the configured SNMP Management stations.

7.2.2 Embedded-MTA Secure Power-on Initialization Flow (IPv6 eCM)

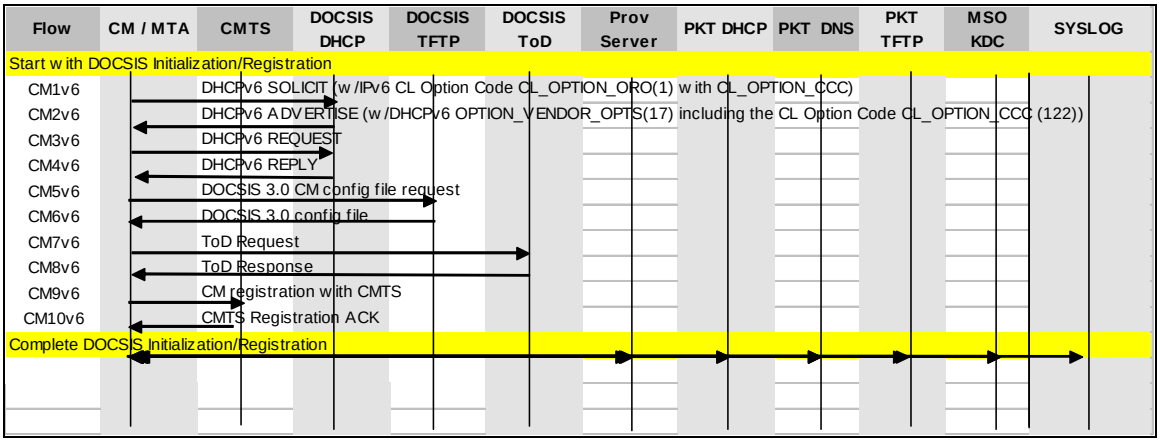

Figure 7: eCM provisioning in IPv6 mode

Table 2

Flow	Embedded-MTA Power-On Initialization Flow Description	Normal Flow Sequencing	must Proceed to here if this step fails
(See note)			
CM1v6	As specified in DOCSIS®, an eCM trying to obtain IP configuration information from a DHCPv6 server transmits a DHCPv6 SOLICIT message. The following requirements apply to the message: • The eCM must include all the DHCPv6 Options required by [18]. • The eCM must request option CL_OPTION_CCC(122) within the "CableLabs® Option Request Option", CL_OPTION_ORO(1), as specified in [19]. • The eCM must include the CableLabs® vendor specific options from CL_OPTION_DEVICE_TYPE(2) until CL_OPTION_VENDOR_NAME(10), inclusive, as specified in [19]. • The eCM must include CL_OPTION_DEVICE_ID(36) as specified in [19].	Initial must Step in Sequence	Per DOCSIS®
CM2v6	The DOCSIS® DHCPv6 Server, if it has been configured to support E-MTA devices, must respond with a DHCPv6 ADVERTISE message. The following requirements apply to the message: • The DOCSIS® DHCPv6 server must include all the DHCPv6 options specified in [18]. • The DOCSIS® DHCPv6 server must include the option OPTION_VENDOR_OPTS(17) containing option CL_OPTION_CCC with sub-option 1 and, possibly, sub-option 2. • A DOCSIS® DHCPv6 server configured to prevent eMTAs from provisioning must include a value of 0.0.0.0 within sub-option 1 of CL_OPTION_CCC. • A DOCSIS® DHCPv6 server without any prior knowledge of eMTA devices may respond with DHCP ADVERTISE messages without the CL_OPTION_CCC option. Refer to clause 8.1 for more information on sub-options 1 and 2.	CM2v6 must occur after CM1v6 completion, unless two-message rapid commit message exchange is used	Per DOCSIS®
CM3v6	Upon receiving a DHCPv6 ADVERTISE by the eCM, the following requirements apply: • The eCM must check for the requested option CL_OPTION_CCC. If it is not present, the eCM must exponentially retransmit the DHCPv6 SOLICIT message (CM1v6) for three attempts (e.g. in two, four, eight second intervals). • If the eCM does not receive any DHCPv6 ADVERTISE message with option CL_OPTION_CCC within the retry attempts, it must select a DHCPv6 ADVERTISE message without the option code CL_OPTION_CCC. • Once a DHCPv6 ADVERTISE message has been selected, the eCM must send a DHCPv6 REQUEST message to indicate acceptance, as specified in [20].	CM3v6 must occur after CM2v6 completion, unless two-message rapid commit message exchange is used	Per DOCSIS®
CM4v6	The DOCSIS® DHCPv6 server, upon receiving a DHCPv6 REQUEST message indicating its presence, must respond with a DHCPv6 REPLY message. Upon receiving this message by an eCM, the following requirements apply: • The eCM must check again for the presence of option CL_OPTION_CCC. If it is not present, the eCM must not initialize the MTA. If it is present and the value of sub-option 1 is not set to a value of 0.0.0.0, the eCM must initialize the MTA and transmit the values pertaining to the Telephony Service Provider's DHCP server information (sub-option 1 and sub-option 2). If it is present and the value of sub-option 1 is set to a value of 0.0.0.0., the eCM must not initialize the MTA. • The eCM must treat the contents of the DHCPv6 REPLY message as authoritative over the DHCPv6 ADVERTISE message.	CM4v6 must occur after CM3v6 or CM1v6 (if rapid commit is used)	Per DOCSIS®

Flow	Embedded-MTA Power-On Initialization Flow Description	Normal Flow Sequencing	must Proceed to here if this step fails
CM5v6 -CM10v6	The eCM must complete the remainder of the DOCSIS® specified registration sequence.	CM5v6 - CM10v6 must occur after CM4v6 completion	Per DOCSIS®
MTA1-MTA25	The eMTA must complete the remainder of the flow as indicated in clause 7.2.1.	MTA1-MTA25 must occur after CM10v6 completion	
NOTE: Refer to the [18] for a complete description of the eCM Provisioning Flows. This only provides IP configuration retrieval using DHCPv6.			

7.3 Embedded-MTA Power-On Initialization Flow (Basic Flow)

The Basic MTA provisioning flow is very similar to the DOCSIS® CM provisioning flow

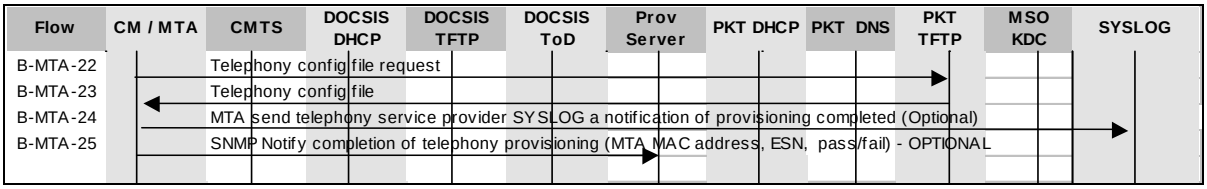

Figure 8: Embedded-MTA Basic Power-on Initialization Flow

Table 3

Flow	Embedded-MTA Power-On Initialization Flow Description	Normal Flow Sequencing	must proceed here if this step fails
See note 1			
B-MTA-22	TFTP Configuration File Request The MTA must perform a TFTP protocol exchange to download its configuration file. The 'siaddr' and 'file' fields of the DHCP ACK are used to locate the configuration file. Specific details of the TFTP protocol can be found in [9].	B-MTA-22 must occur after MTA-4.	If failure per TFTP protocol, return to MTA-1.
B-MTA-23	TFTP Configuration File Response The TFTP server must send the requested configuration file to the MTA. Specific details of the TFTP protocol can be found in [9]. The downloaded configuration file must contain the MIB object 'pktcMtaDevConfigHash'. The MTA must calculate the hash of the downloaded configuration file per clause 9.1 and compare this value to the value contained in the 'pktcMtaDevConfigHash' object. If these values do not match, this step must fail. Refer to clause 9.1 for MTA configuration file contents.	B-MTA-23 must occur after B-MTA-22	If the configuration file download failed per TFTP protocols, return to MTA1. Otherwise, proceed to B- MTA24 and send the failed response if the MTA configuration file itself is in error
B-MTA-24	SYSLOG Notification If a SYSLOG server is configured and enabled as part of the Provisioning Process (Refer to step MTA-2 for DHCP Options and [13], [14], and [23], for configuration using the MEM-MIB), then the MTA must send the voice service provider's SYSLOG a "provisioning complete" event indicating the status of the provisioning operation. The general format of this notification is as defined in clause 5.4.3.	B-MTA-24 must occur after B-MTA-23 completion if SYSLOG is configured.	The MTA may retry this step before proceeding to B-MTA-25.

Flow	Embedded-MTA Power-On Initialization Flow Description	Normal Flow Sequencing	must proceed here if this step fails
B-MTA-25	SNMPv2c Provisioning Status INFORM (optional) If commanded by DHCP option 122 sub-option 6, the MTA must send the PROV_SNMP_ENTITY (specified in DHCP option 122 sub-option 3) an SNMP INFORM containing a "provisioning complete" notification. The receipt of the SNMP INFORM is acknowledged. The SNMP INFORM must contain a "pktcMtaDevProvisioningStatus" MIB object. The SNMPv2c community name used in the status SNMP INFORM must have a value "private" (taken without the quotation marks) (see note 2 and 3).	B-MTA-25 is optional, may occur after B-MTA-24 if SYSLOG is used, otherwise may occur after B-MTA-23 completion	Provisioning process stops; manual interaction required. SNMP server must send response to SNMP-INFORM
NOTE 1: The FQDN provided in the DHCP ACK in DHCP option 122 sub-option 3 (Provisioning Entity Address) must be resolved to an IP address before step B-MTA-22. NOTE 2: At this stage, the MTA device provisioning data is sufficient to provide any minimal services as determined by the service provider. NOTE 3: Depending on the TLV38 configuration value pairs, there might be multiple SNMP INFORMs sent to the configured SNMP Management stations.			

7.4 Embedded-MTA Power-On Initialization Flow (Hybrid Flow).

The Hybrid Provisioning Flow (Hybrid Flow) is essentially the Secure Flow with Kerberos exchanges removed and SNMPv2c substituted for SNMPv3. The SNMPv2c community name, used in the SNMP INFORM messages sent by the MTA in steps H-MTA15 and H-MTA25 below, must have a value "private" (taken without the quotation marks).

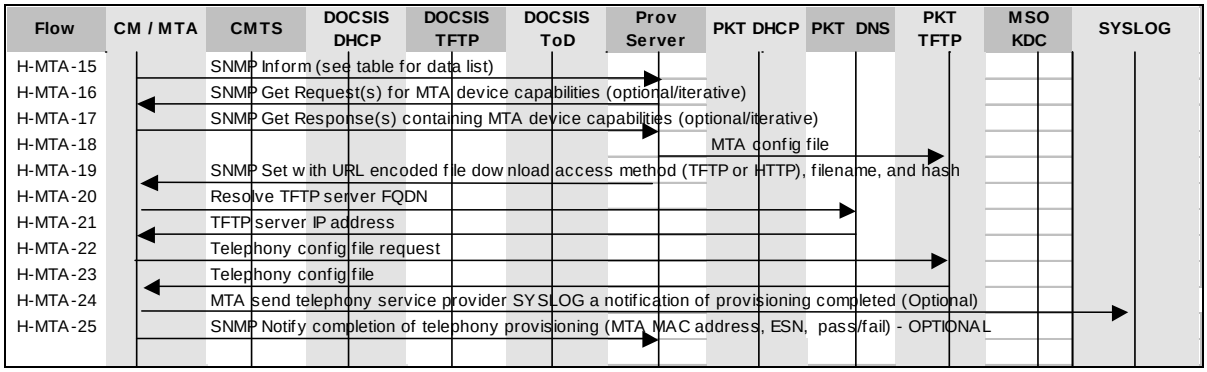

Figure 9: Embedded-MTA Hybrid Power-on Initialization Flow

Table 4

Flow	Embedded-MTA Power-On Initialization Flow Description	Normal Flow Sequencing	must proceed here if this step fails
See note 1			
H-MTA-15	SNMPv2c Enrollment INFORM The MTA must send a SNMPv2c Enrollment INFORM to PROV_SNMP_ENTITY (specified in the DHCP option 122 sub-option 3). The SNMP INFORM must contain a 'PktcMtaDevProvisioningEnrollment' object as defined in [2]. The PROV_SNMP_ENTITY notifies the PROV_APP that the MTA has entered the management domain.	H-MTA-15 must occur after MTA-4 completion	If failure per SNMP protocol return to MTA-1. SNMP server must send response to SNMP-INFORM.
H-MTA-16	SNMPv2c GET Request (optional) The Provisioning Application may request additional MTA device capabilities from the MTA via SNMPv2c GET requests. This is done by having the Provisioning Application send the PROV_SNMP_ENTITY an SNMP GET request. Iterative: The PROV_SNMP_ENTITY sends the MTA one or more SNMPv2c GET requests to obtain any needed MTA capability information. The Provisioning Application may use a GETBulk request to obtain several pieces of information in a single message.	H-MTA-16 is optional, can occur after H-MTA-15 completion	N/A
H-MTA-17	SNMPv2c GET Response (optional) Iterative: MTA sends the PROV_SNMP_ENTITY a Get Response for each Get Request. After all the Gets, or the GetBulk, finish, the PROV_SNMP_ENTITY sends the requested data to the Provisioning Application.	H-MTA-17 must occur after H-MTA-16 completion if H-MTA-16 is performed	N/A
H-MTA-18	This protocol is not defined by IPCablecom. The Provisioning Application may use the information from H-MTA-15, -16, and -17 to determine the contents of the MTA configuration data file. Mechanisms for sending, storing and, possibly, creating the configuration file are outlined in H-MTA-19.	H-MTA-18 should occur after H-MTA-15 completion unless H_MTA-16 is performed, then it should be after H-MTA-17 has completed	N/A
H-MTA-19	SNMPv2c Configuration File Set The Provisioning Application may create the configuration file at this point, or send a predefined one. The Provisioning Application must calculate SHA-1 hash on the contents of the configuration file. The Provisioning Application must store the configuration file on the appropriate TFTP server. The Provisioning Application then instructs the PROV_SNMP_ENTITY to send an SNMPv2c SET message to the MTA, containing the following varbindings (defined in [2]): pktcMtaDevConfigFile pktcMtaDevProvConfigHash Unlike the Secure Flow, the pktcMtaDevProvConfigKey MIB object must not be included. If the pktcMtaDevProvConfigKey MIB object is included, the MTA must return an 'inconsistent value' error (Refer to [7] for more information regarding SNMP SET Responses) (see notes 2 to 4).	H-MTA-19 must occur after H-MTA-18 completion	If failure per SNMP protocol return to MTA-1
H-MTA-20	DNS Request (optional) If the URL-encoded access method contains a FQDN instead of an IPv4 address, the MTA must use the service provider network's DNS server to resolve the FQDN into an IPv4 address of either the TFTP Server or the HTTP Server.	H-MTA-20 must occur after H-MTA-19 completion if FQDN is used	If failure per DNS protocol return to MTA-1

Flow	Embedded-MTA Power-On Initialization Flow Description	Normal Flow Sequencing	must proceed here if this step fails
H-MTA-21	DNS Reply (optional) DNS Response: DNS server returns the IP address against H-MTA-20 DNS request.	H-MTA-21 must occur after H-MTA-20 completion if FQDN is used	If failure per DNS protocol return to MTA-1
H-MTA-22	TFTP/HTTP Configuration file Request The MTA must perform either the TFTP or HTTP protocol exchange, as specified in step H-MTA-19, to download its' configuration file. For specific details of each protocol see [9] and [27].	H-MTA-22 must occur after H-MTA-19 unless FQDN is specified then must be after H-MTA-21	If failure per TFTP or HTTP protocols, return to MTA-1
H-MTA 23	TFTP/HTTP Configuration file Response TFTP/HTTP server must send the requested configuration file to the MTA. For specific details of each protocol see [9] and [27]. The hash of the downloaded configuration file is calculated by the MTA and compared to the value received in step H-MTA-19. If the hashes do not match, this step must fail. Refer to clause 9.1 for MTA configuration file contents	H-MTA-23 must occur after H-MTA-22	If the configuration file download failed per TFTP or HTTP protocols, return to MTA1. Otherwise, proceed to MTA24 or MTA25, and send the failed response if the MTA configuration file itself is in error
H-MTA-24	SYSLOG Notification If a SYSLOG server is configured and enabled as part of the Provisioning Process (Refer to step MTA-2 for DHCP Options and [13], [14], and [23], for configuration using the MEM-MIB), then the MTA must send the voice service provider's SYSLOG a "provisioning complete" event indicating the status of the provisioning operation. This notification will include the pass-fail result of the provisioning operation. The general format of this notification is as defined in clause 5.4.3.	H-MTA-24 must occur after H-MTA-23 completion if SYSLOG is configured	The MTA may retry this step before proceeding to H-MTA-25
H-MTA-25	SNMPv2c Provisioning Status Inform (optional) If commanded by DHCP 122 sub-option 6, the MTA must send the PROV_SNMP_ENTITY (specified in DHCP option 122 sub-option 3) a SNMPv2c Provisioning Status INFORM containing a "provisioning complete" notification. The receipt of the inform is acknowledged. The inform must contain a 'pktcMtaDevProvisioningStatus' MIB object (see notes 5 and 6).	H-MTA-25 is optional. It may occur after H-MTA-24 if SYSLOG is used, otherwise it may occur after H-MTA-23 completion	Provisioning process stops; manual interaction required. SNMP server must send response to SNMP-INFORM

NOTE 1: The FQDN provided in the DHCP ACK in DHCP option 122 sub-option 3 (Provisioning Entity Address) must be resolved to an IP address before step H-MTA-15.
NOTE 2: . In the case of file download using the HTTP access method, the filename must be URL-encoded with a URL format compliant with RFC 2616 [30] with exception stated below in note 4.
NOTE 3: In the case of file download using the TFTP access method, the filename must be URL-encoded with a URL format compliant with RFC 3617 [31] with exception stated below in note 4.
NOTE 4: MTA must accept IPv4 addresses embedded in URL encoded format with or without square brackets.
NOTE 5: At this stage, the MTA device provisioning data is sufficient to provide any minimal services as determined by the service provider.
NOTE 6: Depending on the TLV38 configuration there might be multiple SNMPv2c INFORM sent to the configured SNMP Management stations.

7.5 Endpoint Provisioning Completion Notifications

After the MTA has been provisioned successfully regardless of the selected provisioning flow, the MTA will set up the necessary security association for the related CMS configured realms (KDCs). The MTA NCS signalling software will initiate the establishment of the IPsec security association to the configured CMS clusters. Event notifications are triggered if security associations cannot be established (based on [5]).

With the selected Basic, Hybrid, or Secure flow complete, and after any required security associations are established, the MTA NCS signalling software determines whether a signalling path can be setup with an RSIP message and the associated ACK. Coming from a link down situation, the MTA will send an SNMP Link Up Trap when the RSIP has been properly acknowledged. This indicates that the endpoint is provisioned. If the same CMS is used for multiple endpoints, a SNMP link up message will be sent for each associated endpoint. If not all endpoints use the same CMS, the same process needs to be repeated for each endpoint needing a different configured CMS.

7.6 Post Initialization Incremental Provisioning

This clause describes the flows allowing the Provisioning Application to perform incremental provisioning of individual voice communications endpoints after the MTA has been initialized. Post-Initialization incremental provisioning may involve communication with a Customer Service Representative (CSR).

7.6.1 Synchronization of Provisioning Attributes with Configuration File

Incremental provisioning includes adding, deleting and modifying subscriber services on one or more endpoints of the embedded-MTA. Services on an MTA endpoint must be modified using SNMP via the MTA MIB ([2] and [21]). The back office applications should support a "flow-through" provisioning mechanism that synchronizes all device provisioning information on the embedded-MTA with the appropriate back office databases and servers. Synchronization is required in the event that provisioning information needs to be recovered in order to re-initialize the device. Although the details of the back office synchronization are beyond the scope of the present document, it is expected that, at a minimum, the following information is updated: customer records, and the MTA configuration file on the TFTP or HTTP server.

7.6.2 Adding/Enabling Telephony Services on an MTA Endpoint

The Telephony Services may be added and/or enabled on an MTA endpoint. Telephony Services may be added to MTA endpoints that have not been previously provisioned.

Whenever such an MTA endpoint is added/enabled:

- The MTA must have been provisioned with the 'device level' configuration data via the configuration file (as described in clause 9.1.1).
- The authorized SNMP Management Station must provision all required configuration attributes as described in clauses 9.1.3, 9.1.4 and 9.1.5 using SNMP SET operations to update the provisioning attributes on the device for the specific telephony port being enabled.

Telephony Services may be enabled for MTA endpoints with services provisioned, but disabled (refer to clauses 7.6.3 and 9.1.1 for more details). To enable previously disabled telephony services on the MTA endpoint, an authorized SNMP Management Station must use appropriate SNMP SET operations to achieve both of the following:

- Ensure that the rowstatus MIB Object (pktcNcsEndPntConfigStatus) for the row corresponding to the endpoint is set to a value of "active (1)" (modify it appropriately if it is set to any other value).
- Ensure that the value of "ifAdminStatus" corresponding to the endpoint being enabled has a value of "up(2)" (modify it appropriately if it is set to any other value).

When an endpoint is provisioned or enabled, the MTA must perform the following steps (not necessarily in this order):

- Follow the procedures described in clause 7.1.1.2.5 of the Security Specification [5].
- Modify the "ifOperStatus" MIB Object according to the clause 7.7 of the present document.

If "pktcMtaDevEnabled" MIB Object is set to "true (1)", the MTA must follow the above steps for all configured end-points.

It is to be noted that, given the nature of the MIB Object controlling the absence or presence of IPsec Security Associations with a Call Management Server, Endpoint Provisioning cannot be used to change the IPsec status (refer to [2] and [21] for more information). Thus, enabling new services with a Call Management Server whose status has not been indicated earlier (via the configuration file) will result in IPsec being enabled, upon assignment to an endpoint.

As an example of enabling telephony services on an endpoint, consider the case where a subscriber has requested service on an endpoint that has not been previously provisioned.

NOTE: This example assumes the service provider's account creation process has been completed, and shows only the components critical for the flows. For instance, account creation and billing database creation are assumed to be available and integrated in the back office application suite.

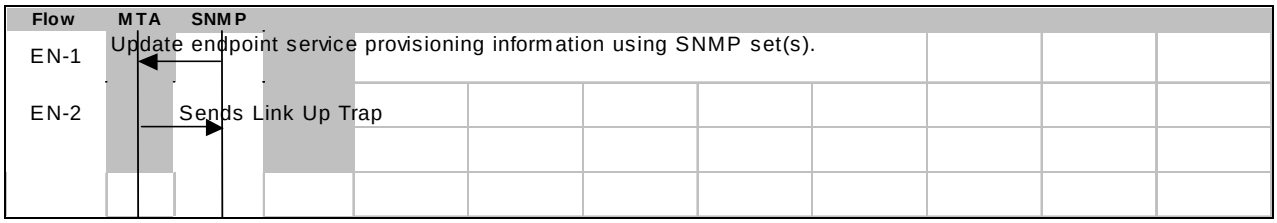

Table 5

Flow	Enabling Telephony Services on an MTA Endpoint Flow Description	Normal Flow Sequencing
EN-1	Authorized SNMP Management Station performs required SNMP SET operations to add services on the MTA Endpoint.	If End Point configuration is desired, EN-1 must occur after successful completion of power on initialization flow.
EN-2	The MTA must send a Link Up trap to the configured SNMP Management Stations. Refer to clause 7.7 and the IF-MIB [24] for more information.	EN-2 must occur after EN-1

7.6.3 Deleting/Disabling Telephony Services on an MTA Endpoint

Provisioned and enabled Telephony Services can be disabled (taken out of service) or deleted if required using SNMP via the MTA MIB ([2] and [21]), and the Signalling MIB ([15] and [22]), on a per-endpoint basis.

Whenever a telephony service is desired to be deleted on an endpoint, the authorized SNMP Management Station must delete appropriate configuration attributes described in the clauses 9.1.3, 9.1.4 and 9.1.5, using SNMP SET operations for the corresponding endpoint.

To disable the services on an MTA endpoint, an authorized SNMP Management Station must use SNMP SET operations to accomplish one or more of the following conditions:

- For the particular endpoint, modify the row status object to a value other than "active (1)" in "pktcNcsEndPntConfigTable.
- Modify the value of "ifAdminStatus" to "down (2)", for the particular endpoint.

If the endpoint is being deleted or disabled while a call in progress, the MTA MUST:

- Shutdown all media sessions if present.
- Shutdown NCS signalling by following the Restart in Progress procedures in the IPCablecom NCS Specification [4].
- Set the pktcNcsEndPntStatusError MIB Object for the particular endpoint to the "disconnected (3)" state.

If "pktcMtaDevEnabled" MIB Object is set to "false (2)", the MTA must follow the above procedure for all configured end-points.

As an example of disabling telephony services on an endpoint, consider the case where a subscriber has requested disabling telephony services on a previously configured endpoint.

NOTE: It is assumed that the service provider's account update process has been completed and shows only the applications critical to MTA operation.

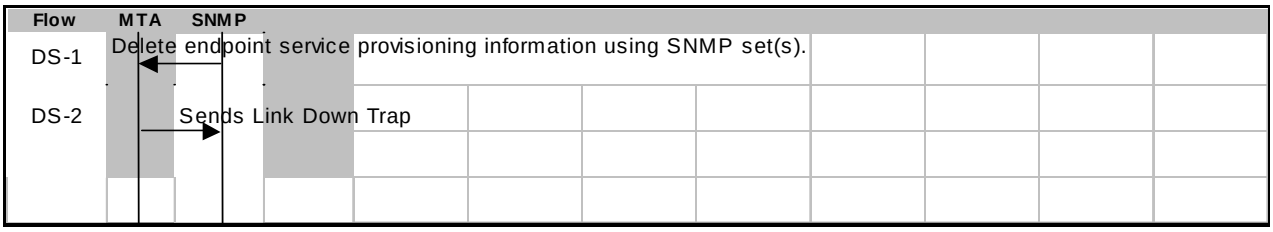

Table 6

Flow	Disabling Telephony Services on an MTA Endpoint Flow Description	Normal Flow Sequencing
DS-1	Authorized SNMP Management Station performs required SNMP SET operations to disable services on the MTA Endpoint.	DS-1 must occur after the endpoint is brought to enabled state either immediately after the initial provisioning or after the per-endpoint incremental provisioning.
DS-2	The MTA must send a Link Down trap to the configured SNMP Management Stations. Refer to clause 7.7 and the IF-MIB [24] for more information.	DS-2 must occur after DS-1

7.6.4 Modifying Telephony Services on an MTA Endpoint

Telephony Services may be modified on a currently provisioned 'MTA Endpoint'. This is accomplished using SNMP via the MTA MIB ([2] and [21]), and the Signalling MIB ([3] and [22]) on a per-endpoint basis. If such a modification to an endpoint changes the CMS association (pktcNcsEndPntConfigCallAgentId) and/or the port (pktcNcsEndPntConfigCallAgentUdpPort), the endpoint is treated as being taken out of service (as per clause 7.6.3), followed by the placing the endpoint back in service (as per clause 7.6.2).

The MTA must also follow the procedures described in clause 7.1.1.2.5 of the Security Specification [5].

It is to be noted that:

- Modification to call service features requires modifications in the CMS, not in the MTA.
- Modification to service level parameters related to the eCM component of the eMTA may require rebooting of the E-MTA.

7.7 Reflecting the State of the Endpoint Interface in the ifTable

The operational state of each 'MTA Endpoint' is reflected in the "ifOperStatus" MIB Object of the MTA. This is influenced by the following conditions:

- The corresponding administrative status for the endpoint, reflected in the "ifAdminStatus" table.
- The state of the telephony service assigned to the corresponding endpoint.
- The presence or absence of the IPsec security associations on the corresponding endpoint, provided IPsec is enabled (i.e. the MIB Object "pktcMtaDevCmsIpsecCtrl" set to a value of "true(1)" for that endpoint).

Whenever an MTA reinitializes (following a reboot or a reset), it must immediately set the "ifAdminStatus" entries corresponding to all available physical endpoints to a value of 'up (1)'. However, entries in the configuration file or the SNMP Management station can change this status. The MTA must further reflect the above conditions in the operational status of each endpoint as explained below.

For each entry corresponding to an endpoint in the "ifTable" MIB, the MTA must set the "ifOperStatus" to a value of:

- "down(2)", if the corresponding endpoint is disabled or deleted, or the corresponding "ifAdminStatus" is set to a value of "down(2)";
- "up(1)", if the corresponding "ifAdminStatus" has a value of "up(1)", the telephony services have been added/enabled for the particular endpoint, and IPsec is disabled with the assigned Call Management Server;

- "up(1)", if the corresponding "ifAdminStatus" has a value of "up(1)", the telephony services have been added/enabled for the particular endpoint, IPsec is enabled for the assigned Call Management Server, and the IPsec Security association has been established;
- "dormant(3)", if the corresponding "ifAdminStatus" has a value of "up(1)", the telephony services have been added/enabled for the particular endpoint, IPsec is enabled for the assigned Call Management Server, but the IPsec Security association has not been established.

Further, the MTA must not set the 'ifOperStatus' to a value of 'dormant(3)' for endpoints on which IPsec is disabled. Refer to [2] for more details on enabling/disabling IPsec, clause 7.6.2 for more details on adding/enabling endpoints, and clause 7.6.3 for more details on deleting/disabling endpoints.

The MTA must be able to enable or disable the 'Link Up Trap' and 'Link Down Trap' by using the "ifLinkUpDownTrapEnable" MIB Object (Refer to the IF-MIB for more details).

7.8 Provisioning of the Signalling Communication Path Between the MTA and CMS

All issues related to the creation and handling of the NCS Service Flows are considered to be resolved by the DOCSIS® means and are out of the scope of IPCablecom 1.5.

7.9 MTA Replacement

IPCablecom 1.5 has no requirement to specify MTA replacement procedures. However, the provisioning sequence flows detailed within the present document provide sufficient coverage and flexibility to support replacement. In fact, the initialization sequence for a replacement MTA could be the same as the original MTA's first time initialization. Back office procedures related to migration of subscriber profiles from one MTA to another are specific to individual service provider's network operations. As a result of this wide variance, discussion of these back office procedures are beyond the scope of IPCablecom 1.5.

7.10 Temporary Signal Loss

If the eCM (in an E-MTA) resets due to any Rf condition (for example Temporary Rf loss), then the associated IPCablecom eMTA must also reset.

NOTE This will impact calls in progress.

7.11 MTA Hard Reboot/Soft Reset scenarios.

Hard Reboot is defined as a 'power cycle' of the entire E-MTA device. Soft Reset is defined as an 'SNMP reset' of the eMTA, an SNMP reset of the eCM, i.e. CM1 or CM1v6 (resulting in the reset of the associated eMTA) or an Rf condition that results in a reset of the eCM (resulting in the reset of the associated eMTA).

The eMTA must not differentiate between a 'Hard Reboot' and a 'Soft Reset', unless explicitly specified otherwise. To be more specific, the eMTA must have the same initialization parameters (e.g. SNMP tables) and follow any requirements regarding persistent information (e.g. NVRAM ticket storage) the same way in either scenario, unless explicitly specified otherwise.

8 DHCP Options

DHCP ([1] and [20]) is used to obtain the IP configuration data for both the eCM and the eMTA. For IPv4 addressing mode, if the total number of octets in any DHCP option exceeds 255 octets, then the MTA must follow [10] to split the DHCP message in to multiple sub messages.

DHCP option code 122 is used to specify the IPCablecom-specific options to the eCM and eMTA components of an E-MTA. This is conveyed as option 122 within DHCPv4. For eCMs obtaining information from DHCPv6 server, this is conveyed as CL_OPTION_CCC(122) within the "CableLabs® Option Request Option", CL_OPTION_ORO(1), as specified in [19].

8.1 DHCP Option 122: CableLabs® Client Configuration Option

DHCP option code 122 is the RFCed replacement for the former option 177 (which was intended as a temporary code). CM and MTA must not request option 177 in their DHCP DISCOVER or REQUEST message in option 55 (parameter request list). In the case that a CM or MTA requests both options 122 and 177:

- The provisioning server must respond with DHCP option 122.
- The provisioning server must not respond with DHCP option 177.
- CM and MTA must treat DHCP option 122 as authoritative.

DHCP option code 122 is used in both the CM and MTA DHCP OFFER/ACK messages to provide the addresses of valid IPCablecom network servers and various device configuration data.

Full details of DHCP option 122 encoding can be found in [26] and [i.8].

The following clauses provide additional semantic details of each sub-option in DHCP option 122.

Table 7

Option	Sub-option	Description and Comments	Sub-option Required or Optional	Default Value
122	1	Service Provider's Primary DHCP Server Address Required by CM only.	Required	N/A
	2	Service Provider's Secondary DHCP Server Address Optional requirement for CM.	Optional	Empty String
	3	Service Provider's Provisioning Entity Address	Required	N/A
	4	AS-REQ/REP Exchange Backoff and Retry for SNMPv3 Key Management	Optional	As per the following MIB Objects: "pktcMtaDevRealmUnsolicitedKeyNomTimeout" "pktcMtaDevRealmUnsolicitedKeyMaxTimeout" "pktcMtaDevRealmUnsolicitedKeyMaxRetries"
	5	AP-REQ/REP Kerberized Provisioning Backoff and Retry	Optional	As per the following MIB Objects: "pktcMtaDevProvUnsolicitedKeyNomTimeout" "pktcMtaDevProvUnsolicitedKeyMaxTimeout" "pktcMtaDevProvUnsolicitedKeyMaxRetries"
	6	Kerberos Realm of SNMP Entity	Required	N/A
	7	Ticket Granting Server Usage	Optional	N/A - if MTA does not implement TGT. 0 - otherwise.
	8	Provisioning Timer	Optional	As per "pktcMtaDevProvisioningTimer" MIB Object (10 minutes)
	9	Security Ticket Invalidation	Optional	0 - apply normal ticket invalidation rules per [5]

MTA must be able to retrieve and process the data from all sub-options in the above table. Provisioning Server must supply to the MTA all "required" sub-options and may supply all "optional" sub-options.

If an "optional" sub-option is not supplied by the Provisioning Server, the MTA must use the default value of the sub-option.

If the "required" sub-option is not supplied by the Provisioning Server, the MTA must reject the corresponding DHCP OFFER/ACK.

If the sub-option contains wrong (invalid) value, the MTA MUST:

- reject the corresponding DHCP OFFER/ACK in case of "required" sub-option;
- use the default value in case of "optional" sub-option For any sub option with multiple parameters (e.g. Option 122 sub option 4 or Option 122 sub option 5) MTA must apply the corresponding default value only to the parameter (or parameters) that contains the wrong value.

An MTA must ignore any other sub-option in Option-122 except those listed in the above table.

8.1.1 Service Provider's DHCP Address (sub-option 1 and 2)

The Service Provider's DHCP Server Addresses identify the DHCP servers that a DHCP OFFER will be accepted from in order to obtain an MTA-unique IP address for a given service provider's network administrative domain. The encoding of these sub-options is defined in [26].

Sub-option 1 must be included in the DHCP OFFER/ACK to the CM and it indicates the Primary DHCP server's IP address. The value contained in sub-option 1 must be a valid IP address, a value of 255.255.255.255 or a value of 0.0.0.0. The value contained in sub-option 2 must be a valid IP address.

The MTA must follow the logic in table 8 when defining its DHCP strategy regardless of the Provisioning Flow used.

Table 8

Value of Sub-option-1	**Value of Sub-option-2**	
	Valid IP - DHCP Server is Responding	Valid IP - DHCP is NOT Responding
255.255.255.255	MTA must select the OFFERs according to the logic of RFC 2131. Value [1] in the sub-option-2 must be ignored	MTA must select the OFFERs according to the logic of RFC 2131 [1]. Value in the sub-option-2 must be ignored
0.0.0.0	MTA must stop all provisioning attempts as well as all other activities.	MTA must stop all provisioning attempts as well as all other activities.
Valid IP - DHCP Server is Responding	MTA must accept DHCP OFFERs coming only from the IP Address in the sub-option-1.	MTA must accept DHCP OFFERs coming only from the IP Address in the sub-option-1.
Valid IP - DHCP is NOT responding	MTA must try exponentially at least three times before accepting the DHCP OFFER coming from the DHCP Server pointed out by Sub-option-2.	MTA must return to MTA-1 step.

8.1.2 Service Provider's Provisioning Entity Address (sub-option 3)

The Service Provider's Provisioning Entity Address is the network address of the provisioning server for a given voice service provider's network administrative domain.

The encoding of this sub-option is defined in [26]. This address must be configured as an FQDN only.

An FQDN value of 0.0.0.0 in sub-option 3 of a valid MTA DHCP OFFER/ACK specifies that the MTA must shutdown and not try to provision unless it is reinitialized by the CM. This is explained in step MTA2 of the provisioning flow process of clause 7.2.

The Service Provider's Provisioning Entity Address component must be capable of accepting SNMP traps. Sub-option 3 must be included in the DHCP OFFER to the MTA.

8.1.3 AS-REQ/REP Exchange Backoff and Retry for SNMPv3 Key Management (sub-option 4)

The MTA must use the DHCP option 122 sub-option 4, if supplied in Secure Flow only. AS-REQ/REP exchange backoff and retry mechanism of the Kerberized SNMPv3 key negotiation defined in [5] is controlled by the values delivered in this sub-option or by the default values of the corresponding MIB objects in the Realm table if this sub-option is not present in the DHCP Option 122.

The encoding of this sub-option is defined in [26].

The sub-option's nominal timeout value corresponds to the pktcMtaDevRealmUnsolicitedKeyNomTimeout MIB object in the pktcMtaDevRealmTable.

The sub-option's maximum timeout value corresponds to the pktcMtaDevRealmUnsolicitedKeyMaxTimeout MIB object in the pktcMtaDevRealmTable.

The sub-option's max retry count corresponds to the pktcMtaDevRealmUnsolicitedKeyMaxRetries MIB object in the pktcMtaDevRealmTable.

An MTA must be able to retrieve the above parameters from this sub-option, if they are supplied by the Provisioning Server.

Provisioning Server may provision an MTA with the above parameters using this sub-option.

If any of the values defined in this sub-option are "FFFFFFFF" (hexadecimal) then the default value of the corresponding column from the Realm Table must be used.

8.1.4 AP-REQ/REP Kerberized Provisioning Backoff and Retry (sub-option 5)

The MTA must use the DHCP option 122 sub-option 5, if supplied in Secure Flow only. AP-REQ/REP backoff and retry mechanism of the Kerberized SNMPv3 key negotiation defined in security [5] is controlled by the values delivered by this sub-option.

The encoding of this sub-option is defined in [26].

The sub-option's nominal timeout value corresponds to the pktcMtaDevProvUnsolicitedKeyNomTimeout MIB object.

The sub-option's maximum timeout value corresponds to the pktcMtaDevProvUnsolicitedKeyMaxTimeout MIB object.

The sub-option's max retry count corresponds to the pktcMtaDevProvUnsolicitedKeyMaxRetries MIB object.

An MTA must be able to retrieve the above parameters from this sub-option, if they are supplied by the Provisioning Server.

Provisioning Server may provision an MTA with the above parameters using this sub-option.

If any of the values defined in this sub-option are "FFFFFFFF" (hexadecimal) then the default value of the corresponding MIB Object must be used.

8.1.5 Kerberos Realm of SNMP Entity (sub-option 6)

In conjunction with the Provisioning Entity Address, the Kerberos Realm is used as a means of contacting a SNMP Entity in the provisioning realm. The realm name is used to perform a DNS SRV lookup for the realm's KDC.

The DHCP option 122 sub-option 6 must be included in the DHCP OFFER to the MTA. For the Secure Flow, the DHCP option 122 sub-option 6 must only contain the realm name in the format of FQDN (type=0 as per [26]).

The MTA must select the corresponding Provisioning Flow as per the following table (the DHCP option 122 sub-option 6 content comparison is case-sensitive and must be in all capital letters).

Table 9: MTA Device Provisioning Flow Selection

Content of the DHCP option 122 sub-option 6	MTA Device Provisioning Flow Selection
BASIC.1	If the DHCP option 122 sub-option 6 value is BASIC.1, the MTA must execute the Basic flow without the provisioning complete SNMP INFORM.
BASIC.2	If the DHCP option 122 sub-option 6 value is BASIC.2, the MTA must execute the Basic flow with the provisioning complete SNMP INFORM.
HYBRID.1	If the DHCP option 122 sub-option 6 value is HYBRID.1, the MTA must execute the Hybrid flow without the provisioning complete SNMP INFORM.
HYBRID.2	If the DHCP option 122 sub-option 6 value is HYBRID.2, the MTA must execute the Hybrid flow with the provisioning complete SNMP INFORM.

The MTA must use the Secure Flow if any other value is provided in the DHCP option 122 sub-option 6. For Secure Flow, the encoding of the DHCP option 122 sub-option 6 is defined in [26].

8.1.5.1 SNMPv3 Key Establishment

The SNMPv3 Key Establishment is applicable for Secure Flow only. The AP Request/AP Reply described in figure 6 the accompanying flow description, and the Security Specification are used by the MTA in the initial provisioning phase to establish keys with the SNMPv3 USM User "MTA-Prov-xx:xx:xx:xx:xx:xx". Where xx:xx:xx:xx:xx:xx represents the MAC address of the MTA and must be uppercase. The MTA must instantiate this user in the USM MIB described in RFC 3414 [8], with the ability to be keyed using the IPCablecom Kerberized key management method described in the Security Specification. SNMPv3 authentication is required and privacy is optional. For the list of allowed SNMPv3 authentication and privacy algorithms see [5].

Additionally, the usmUserSecurityName must be the set to the string "MTA-Prov-xx:xx:xx:xx:xx:xx" (quotation marks not included). Where xx:xx:xx:xx:xx:xx represents the MAC address of the MTA and must be uppercase. This ensures a unique usmUserSecurityName is created for each MTA.

The MTA must first obtain a service ticket for the provisioning realm as described in step MTA9. USM key management is performed over UDP, as specified in [5]. The SNMPv3 keys are established prior to any SNMPv3 communication and therefore SNMPv3 messages must be authenticated at all times (with privacy being optional). The MTA must use the USM user created above in the SNMP Enrollment INFORM.

8.1.6 Ticket Granting Server Usage (sub-option 7)

The MTA must use the DHCP option 122 sub option 7 if supplied for the provisioning kerberized key management in Secure Flow only. This sub-option contains a Boolean, which when true, indicates that the MTA should get its TGT (ticket granting ticket).

Sub-option 7 may be included in the DHCP OFFER/ACK to the MTA.

The encoding of this sub-option is defined in [26].

8.1.7 Provisioning Timer (sub-option 8)

Sub-option 8 defines the value to be used for the provisioning timer. Sub-option 8 may be included in the DHCP OFFER/ACK to the MTA.

The encoding of this sub-option is defined in [26].

8.1.8 Security Ticket Invalidation (sub-option 9)

Sub-option 9 contains a bit mask that directs the MTA to invalidate specific application server security tickets. Sub-option 9 may be included in the DHCP OFFER/ACK to the MTA. The encoding of this sub-option is defined in [i.8].

8.2 DHCP Option 60: Vendor Client Identifier

Option code 60 contains a string identifying Capabilities of the MTA. The MTA must send the following ASCII Coded String in DHCP Option code 60: "pktc1.5:xxxxxx". Where xxxxxx must be an ASCII representation of the hexadecimal encoding of the MTA TLV Encoded Capabilities, as defined in clause 10.

8.3 DHCP Options 12 and 15

MTA FQDN must be sent to the E-MTA in Option-12 and Option-15. Option-12 must contain "Host Name" part of the FQDN, and the Option-15 must contain "Domain Name" part of the FQDN.

For example, if MTA FQDN is "mta1.pclab.com", then Option-12 will contain "mta1" and Option-15 will contain "pclab.com".

8.4 DHCP Option 6

DHCP Option 6 must be used to provide the MTA with its list of DNS server addresses. Option 6 must contain at least one DNS server address. Option 6 may contain a secondary DNS server address. If this option contains more than two DNS servers, the MTA must use the first two addresses.

8.5 DHCP Option 43

The MTA must send the DHCP Option 43 in the DHCP DISCOVER and DHCP REQUEST for the Secure, Hybrid and Basic Flows.

DHCP Option 43 contains the number of sub-options defined to provide the MTA device specific information to the back-office systems. The DHCP option 43 sub-options 1 through 10, 31 and 32 are specified by IPCablecom, sub-options 11 to 30 are reserved for the CableLabs® CableHome project, sub-options 33 through 50 are reserved for IPCablecom, sub-options 51 through 127 are reserved for future CableLabs® use, and sub-options 128 and above are reserved for vendor use. The IPCablecom DHCP option 43 sub-options must be present in the format of "Encapsulated vendor-specific extensions" [11]).

Table 10 contains the sub-options of the DHCP Option-43, which the MTA must use. The MTA must send all required sub-options listed in the table below unless explicitly stated otherwise. If the total number of octets in all DHCP option 43 sub-options exceeds 255 octets, the MTA must follow RFC 3396 [10] to split the option into multiple smaller options.

Table 10: DHCP Option 43 Syntax

MTA DHCP Option 43 Sub-options	Required / Not Used in OPTION-43	Value	Description
Sub-option 1	Not Used		The request sub-option vector is a list of sub-options (within option 43) to be returned to client by the server upon reply to the request. None defined. The DHCP option 43 sub-option 1 must not be used by the MTA, and if present, it must be ignored by the Provisioning Server
Sub-option 2	R	<DevType>	The sub-option 2 contains the device type of the component making the DHCP request. The MTA must send the DHCP option 43 sub-option 2. For IPCablecom MTAs, the allowable device types are: - "EMTA" - for E-MTAs
Sub-option 3	Not Used		The sub-option 3 contains a colon separated list of all components in the eDOCSIS® device. It is used by the eDOCSIS® eCM device. The DHCP option 43 sub-option 3 must not be sent by the MTA, and if present, it must be ignored by the Provisioning Server.
Sub-option 4	R	<device serial number>	The sub-option 4 contains the device serial number represented as an ASCII string. The MTA must send the DHCP option 43 sub-option 4. The DHCP option 43 sub-option 4 value must be identical to the value of the pktcMtaDevSerialNumber MIB Object.
Sub-option 5	R	<Hardware version>	The sub-option 5 contains the hardware version number represented as an ASCII string. The MTA must send the DHCP option 43 sub-option 5. The DHCP option 43 sub-option 5 must be identical to the value of the Hardware version number as in <Hardware version> field in the MIB II object sysDescr.
Sub-option 6	R	<Software version>	The sub-option 6 contains the software version number represented as an ASCII string. The MTA must send the DHCP option 43 sub-option 6. The DHCP option 43 sub-option 6 value must be identical to the value of the pktcMtaDevSwCurrentVers MIB object.
Sub-option 7	R	<Boot ROM Version>	The sub-option 7 contains the Boot ROM Version represented as an ASCII string. The MTA must send the DHCP option 43 sub-option 7. The DHCP option 43 sub-option 7 value must be identical to the <Boot ROM version> field in MIB II object sysDescr.
Sub-option 8	R	<OUI>	The sub-option 8 contains the Organizational Unique Identifier (OUI) represented as a hexadecimal-encoded 3-byte octet string. It may match the OUI in the MTA MAC address. The MTA must send the DHCP option 43 sub-option 8. If omitted, the Provisioning Server should use the MTA MAC address as the MTA OUI.
Sub-option 9	R	<Model Number>	The sub-option 9 contains the MTA Device Model Number represented as an ASCII string. The MTA must send the DHCP option 43 sub-option 9. The DHCP option 43 sub-option 9 value must be identical to <Model Number> field in the MIB-II object sysDescr.
Sub-option 10	R	<Vendor Name>	The sub-option 10 contains the Vendor Name represented as an ASCII string. The MTA must send the DHCP option 43 sub-option 10. The DHCP option 43 sub-option 10 value must be identical to <Vendor Name> field in the MIB-II object sysDescr.
Sub-options 11 -30			Reserved for CableHome.
Sub-option 31	R	<MTA MAC Address>	The sub-option 31 contains the MTA MAC Address encoded as a 6 byte octet string. The MTA must send the DHCP option 43 sub-option 31. The DHCP option 43 sub-option 31 value must be identical to the content of the pktcMtaDevMacAddress MIB object.
Sub-option 32	R	<Correlation ID>	The sub-option 32 contains the Correlation ID number encoded as 4-byte INTEGER in the network order. The MTA must send the DHCP option 43 sub-option 32. The DHCP option 43 sub-option 32 value must be identical to the content of the pktcMtaDevCorrelationId MIB object.

MTA DHCP Option 43 Sub-options	Required / Not Used in OPTION-43	Value	Description
Sub-options 33 to 50			Reserved for IPCablecom.
Sub-options 51 to 127			Reserved for CableLabs.
Sub-options 128 to 254			Reserved for vendors.

8.6 DHCP OPTION 1

DHCP Option 1 is defined in [11].

8.7 DHCP OPTION 3

DHCP Option 3 is defined in [11].

8.8 DHCP OPTION CL_V4_PACKETCABLE_MIB_ENV_OPTION

The Provisioning Server can provide the option 'CL_V4_PACKETCABLE_MIB_ENV_OPTION' within the DHCP OFFER and ACK messages to indicate preference of the MIB module for MTAs implementing both the IPCablecom and IETF MIBs. The following requirements apply:

- If it is not provided the MTA must assume a value of 0x01 (CableLabs).
- If 'CL_V4_PACKETCABLE_MIB_ENV_OPTION' is set to a value of 0x01, the MTA must use the MIB modules specified in clause 10.23.2 for provisioning flows (for example, MTA-1 through MTA-25 for the secure provisioning flow) and default values for all the MIB Objects.
- If 'CL_V4_PACKETCABLE_MIB_ENV_OPTION' is set to a value of 0x02, the MTA must use the IETF issued MIB modules specified in clause 10.23.3, if implemented, for provisioning flows (for example, MTA-1 through MTA-25 for the secure provisioning flow) and default values for all the MIB Objects.
- Irrespective of the preference indicated within 'CL_V4_PACKETCABLE_MIB_ENV_OPTION', the Provisioning Server may use any MIB environment supported by the MTA (as specified in clause 10.23).
- While providing the configuration file information (for example, MTA19), the Provisioning Server must use the MIB environment used by the MTA while requesting the configuration file (for example, MTA15), irrespective of the value provided in 'CL_V4_PACKETCABLE_MIB_ENV_OPTION'.

9 MTA Provisionable Attributes

This clause includes the list of attributes and their associated properties used in device provisioning. All of the provisionable attributes specified in this clause may be updated via the MTA configuration data file, or on a per-attribute basis using SNMP.

IPCablecom 1.5 requires that a MTA configuration data file must be provided to all embedded-MTAs during the registration sequence. Endpoint voice services do not have to be enabled at the time of initialization. MTA device level configuration data must be provisioned during initialization. These items are contained in clause 9.1.1.

The MTA configuration data URL generated by the Provisioning Application must be less than 255 bytes in length and cannot be NULL. Since this filename is provided to the MTA by the Provisioning Application during the registration sequence, it is not necessary to specify a file naming convention.

9.1 MTA Configuration File

This clause explains the format and contents of an MTA configuration file. This file contains a series of "type length and value" (TLV) parameters. Each TLV parameter in the configuration file describes an MTA or endpoint attribute. The configuration data file includes TLVs that have read-write, read only, and no MIB access. Unless specifically indicated, all MIB-accessible configuration file parameters must be defined using the DOCSIS® TLV type 11, the IPCablecom type 64, or IPCablecom TLV type 38. TLV 64 is an IPCablecom defined TLV where the length value is 2 bytes long instead of the 1 byte for DOCSIS® TLV type 11. The TLV type 64 must be used when the length is greater than 254 bytes. If desired, vendor-specific information may be added to the configuration file using the vendor-specific TLV43. This TLV has been specified by the DOCSIS® Specification [6]. Vendors must not provision vendor-specific information using TLV type 11 or 64. TLV 38 is an IPCablecom defined TLV, analogous to TLV-38 used by DOCSIS® and CableHome. The MTA must be able to process the TLVs given in table 11.

Table 11

Type	Length	Value
11	n, where n is 1 byte	variable binding
64	m, where m is 2 bytes	variable binding
38	n, where n is 1 byte	Composite (Contains sub-TLVs)
254	1 byte	0x01 for beginning of the file and 0xFF for the end of the file
NOTE: The use of TLV type 11 rather than TLV 64 is recommended wherever possible.		

In the future, new TLVs introduced in IPCablecom must have a "length field" size 2 bytes.

The VarBind is encoded in ASN.1 Basic Encoding Rules, just as it would be if part of an SNMP Set request.

The MTA configuration file must start with the "telephony configuration file start" tag and must end with the "telephony configuration file end" tag. These tags enable the MTA TLV parameters to be distinguished from DOCSIS® TLV parameters. These tags also provide deterministic indications for start and stop of the MTA configuration file.

The MTA configuration file must contain the attributes identified as "required" in the Device Level Configuration Data table, which appears in clause 9.1.1; failing which, the MTA must reject the configuration file and take the necessary steps as defined in clause 7.2 (failure of step MTA 23 due to 'Configuration file error'). The MTA configuration file may contain any of the non-required attributes which appear in the Device Level Configuration Data table. If the configuration file does not contain required attributes, it must be rejected. The MTA configuration file must be sent to the embedded-MTA every time this device is powered on.

The Device Level Service Data may be sent to the MTA as part of the MTA configuration file or it may be sent to the MTA using SNMP. If included in the configuration file it must contain all of attributes identified as 'required' in the Device Level service data, if any. The MTA configuration file may additionally contain any of the non-required attributes that appear in the Device Level Service Data table.

If voice services are required on the MTA on any endpoint, the following must be done:

1) pktcMtaDevEnabled must be set to TRUE.

2) Per endpoint configuration data must be supplied either through the MTA configuration file (during provisioning) or through end-point provisioning (using SNMP) in the post-provisioning phase.

The End point details, when included must contain the attributes identified as "required" in the Per-Endpoint Configuration Data table, which appears in clause 9.1.3. The MTA configuration file may contain any of the non-required attributes which appear in the Per-Endpoint Configuration Data table in clause 9.1.3. The Per-Endpoint Configuration Data must be sent to the MTA when voice communications service is activated.

It is to be noted that the Device Level Service Data and Per-Endpoint Configuration Data may also be sent to the MTA via incremental provisioning, using SNMP. The MTA must support incremental provisioning.

The MTA must be able to process all TLV11 and TLV64 values with variable bindings containing all MIB objects defined in [29] unless stated otherwise.

The Device Level Configuration data parameter 'pktcMtaDevEnabled' is used to actually enable or disable voice services on an MTA.

Refer to clause 7.6.1 for a discussion concerning synchronization of provisioning attributes with back office systems.

For the Secure and Hybrid Provisioning Flows, the MTA must authenticate the configuration file according to IPCablecom Security Specification [5]; the MTA must reject the configuration file if the configuration file authentication fails and take the necessary steps as defined in clause 7.2 for the Secure Flow and clause 7.4 for the Hybrid Flow. If the configuration file contains the MIB object 'pktcMtaDevProvConfigHash' in the Secure Flow or the Hybrid Flow, the MTA must ignore the value of this MIB object and proceed with further processing of the configuration file and report passWithWarnings and populate the Error OID table (pktcMtaDevErrorOidsTable).

For the Basic Flow, the Provisioning Server and the MTA must support the configuration file data verification process as described below:

1) When the Provisioning Server creates a new MTA Configuration File or modifies an existing one, to be served for an MTA intended to go through the Basic Flow, it must calculate a SHA-1 hash value of the contents of the entire MTA Configuration File including start and end markers, taken as a byte string.

2) The Provisioning Server must add the hash value, calculated in Step 1 to the MTA Configuration File as a TLV-11 triplet corresponding to the 'pktcMtaDevProvConfigHash' MIB Object. The Provisioning Server must insert the TLV11 triplet before the Configuration file end-marker. The Provisioning Server must not change the order of the TLVs in the configuration file after the hash has been calculated. The MTA Configuration File is then made available to the MTA through the appropriate TFTP/HTTP server.

3) Upon receiving the configuration file, the MTA must do the following:

   If the MIB object 'pktcMtaDevProvConfigHash' is absent, MTA must reject the configuration file and must report 'failOtherReason'.

   If the MIB object 'pktcMtaDevProvConfigHash' is present, then MTA MUST:

   a) Calculate SHA-1 over the contents of the file without TLV-11 triplet containing the pktcMtaDevProvConfigHash' and must populate the calculated value into pktcMtaDevProvConfigHash mib object. The MTA must maintain the order of the TLVs for the hash calculation to be correct.

   b) If the computed hash and the value of the 'pktcMtaDevProvConfigHash' MIB object are the same, the MTA Configuration File integrity is verified and the MTA must accept the configuration file for further processing; otherwise, the MTA must reject the Configuration File and the MTA must report 'failOtherReason'.

The MTA must also check for errors in the configuration file. As described above, errors in any of the mandatory parameters must be treated as an error in the configuration file and appropriate steps taken (failure of step MTA 23 due to 'Configuration file error').

If there are errors in the non-required OIDs then the MTA must accept the configuration file, but report the same in the status (MTA-25).

If the Configuration file contains per-cms data and per-endpoint parameters related to CMSs which are not associated to endpoints, an MTA must not establish SAs till and end-point gets associated with that particular CMS (either using SNMP or via NCS redirection).

The MTA must report the state of the configuration file it received in the 'Provisioning complete Inform' (step MTA25 in the provisioning process) as given below:

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

- If the configuration file was in error due to incorrect values in the mandatory parameters, the MTA must reject the configuration file and return: 'failConfigFileError'.

  It must also populate 'pktcMtaDevErrorOidsTable' with the parameter containing the incorrect value and may also populate it with other OID errors/warnings if it parsed the file completely.

- 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 OIDs which are incorrect or not known to the MTA) it must return: 'passWithWarnings'. It must also populate 'pktcMtaDevErrorOidsTable' with a list of all the parameters which were rejected and the reason for the same. The MTA must also use the default values for all such parameters, unless they were overridden by some other means like DHCP, in which case it must use the overridden values.

- If the configuration file is proper, but the MTA cannot reflect the same in its MIB (For ex: Too many entries leading to memory exhaustion), it must accept details related to the CMSs associated with the endpoints and return: 'passWithIncompleteParsing'.

  It must also populate 'pktcMtaDevErrorOidsTable' with a list of all the parameters which cannot be reflected in the MIB.

- If the configuration file cannot be parsed due to an internal error it must return 'failureInternalError'. It should try to populate 'pktcMtaDevErrorOidsTable' for parameters which lead to failure.

- If the configuration file contains overlapping MIB Object references from multiple MIB environments (for example, IETF), the MTA must use the preference provided via DHCP, if such a preference is provided and supported by the MTA (see clause 8.8). In the absence of any preference via DHCP, the MTA must use the last recognized occurrence (vendor specific) of such objects in the configuration file. The MTA must also indicate any rejected configuration file entries as warnings (provisioning state 'passWithWarnings', unless other conditions exist) and populate the error OIDs table (pktcMtaDevErrorOidsTable).

- If the MTA cannot accept the configuration file for any other reason than the ones stated above, it must return 'failureOtherReason'. It should try to populate 'pktcMtaDevErrorOidsTable' for parameters, which lead to the failure.

The MTA Configuration File must contain Per-Realm Configuration Data. In the case of the Secure Provisioning Flow, per-realm Configuration Data must contain at least the data for the Provisioning Realm that is identified in DHCP Option-122, sub-option-6.

In the case of the Secure Provisioning Flow, after receiving the MTA Configuration File, an MTA must validate the following:

- "pktcMtaDevRealmName" MIB Object of the Realm Table must be the same as the Realm Name supplied to the MTA in DHCP Option-122, sub-option-6.

- "pktcMtaDevRealmOrgName" MIB Object of the Realm Table must be the same as the "Organization Name" attribute in the Service Provider Certificate.

- Encryption and Authentication of the MTA Configuration File as per [5].

An MTA must treat any of the above validation failures as failure of the MTA23 Provisioning Flow and the MTA must discard the Configuration File.

If the MTA encounters a vendor-specific TLV43 with a vendor ID that the MTA does not recognize as its own, the MTA must ignore the TLV 43 and the MTA must continue to process the configuration file. If the MTA detects the presence of an unrecognized TLV (TLV type other than TLV 11, TLV 43, TLV 64, TLV 38, or TLV254), the MTA must ignore the TLV assuming the length field of the unrecognized TLV is 2 bytes and proceed with further processing. The MTA must report a provisioning state of passWithWarnings and populate the error OID table (pktcMtaDevErrorOidsTable) if it detects the presence of an unrecognized TLV. If the MTA encounters an unrecognized variable binding in a TLV 11 or TLV 64, it must ignore this binding, must report a provisioning state of passWithWarnings and populate the error OID table (pktcMtaDevErrorOidsTable). It is strongly recommended for the vendors to give serious considerations to backward compatibility issues when modifying existing or introducing new sub-TLVs for TLV 43.

The MTA must attempt to accept configuration file that contains valid set of per-realm and per-CMS configuration data identified in clauses 9.1.4 and 9.1.5 even if the MTA endpoints are not associated with the CMS in the per-CMS configuration data.

IPCablecom MIB objects in MTA-MIB ([2] and [21]), Signalling-MIB ([3] and [22]) and Event-MIB ([13] and [23]) of type 'RowStatus' must not be included in the MTA configuration file. If any IPCablecom MIB objects (MTA MIB, Signalling MIB and Event MIB) of type 'RowStatus' are included in the configuration file, the MTA must ignore the value supplied in any RowStatus object, report a 'passWithWarnings' and populate the MIB table 'pktcMtaDevErrorOidsTable' appropriately. Regardless of the action taken by the MTA, it must properly populate the Error OIDs table with the Row status OID. Non IPCablecom MIB objects type Row status can be present or absent in the MTA configuration file and MTA must process these objects according to the corresponding RFCs for the particular MIB objects (for example SNMPv2c table).

IPCablecom MIB object pktcEnMtaDevMltplGrantsPerInterval if included in the configuration file and is set to enable Multiple Grants per Interval (MGPI) functionality and if the MTA does not support this functionality, then the MTA must ignore the object and report 'PassWithWarnings' and populate the ErrorOidsTable. For more information about the MGPI functionality refer to [6].

9.1.1 Device Level Configuration Data

Refer to the MTA MIB ([2] and [21]) for more detailed information concerning these attributes and their default values.

- The MTA Manufacturer Certificate validates the MTA Device Certificate.

Table 12

Attribute	Syntax	Configuration Access	SNMP Access	MIB File	Object	Comments
Telephony Config File Start	Integer	W, required	None	N/A	N/A	Type ...Length Value 254 1 1 The MTA config file must start with this attribute.
Telephony Config File End	Integer	W, required	None	N/A	N/A	Type Length Value 254 1 255 This must be the last attribute in the MTA config file.
Telephony MTA Admin State	ENUM	W, required	R/W	MTA Device MIB	pktcMtaDevEnabled	Used to enable/disable all telephony ports on the MTA. Applies to the MTA side of the embedded-MTA or the entire stand-alone MTA. Allows blanket management of all telephony ports (external interfaces) on the device. The state of the MTA is controlled by this MIB Object. For more information about this object, refer to the MTA MIB [2].
Realm Organization Name	String	W, required (Secure Provisioning Flow) W, Optional (Basic and Hybrid Provisioning Flows)	R/W	MTA Device MIB	pktcMtaDevRealmOrgName	The value of the X.500 name organization name attribute in the subject name of the service provider certificate.
Solicited Key Timeout	Integer	W, optional	R/W	N/A	pktcMtaDevProvSolicitedKeyTimeout	This timeout applies only when the Provisioning Server initiated key management (with a Wake Up message) for SNMPv3. It is the period during which the MTA will save a nonce (inside the sequence number field) from the sent out AP Request and wait for the matching AP Reply from the Provisioning Server. Since there is a default value, this is optional.
Reset Kerberos ticket information	Integer32	W, optional	R/W	MTA Device MIB	pktcMtaDevResetKrbTickets	Security Specification [5] allows the Kerberos tickets associated with any of the application server (Provisioning Server or CMS) to be stored in the MTA NVRAM until ticket expiry. In order to control the invalidation of the tickets stored in NVRAM, this MIB attribute is used to communicate the required action to the MTA. Upon receiving this attribute in the config file, an MTA must take the specified action. Refer to [2] for more information.

9.1.2 Device Level Service Data

Refer to the MTA MIB ([2] and [21]) the SIGNALLING MIB ([3] and [22]) the NCS Call Signalling spec [4], and RFC 2475 [i.5] for more detailed information concerning these attributes and their default values.

Table 13

Attribute	Syntax	Configuration Access	SNMP Access	MIB File	Object	pktcDevEvSysloComments
NCS Default Call Signalling TOS	Integer	W, optional	R/W	MTA Signalling MIB	pktcSigDefCallSigTos	The default value used in the IP header for setting the TOS value for NCS call signalling.
NCS Default Media Stream TOS	Integer	W, optional	R/W	MTA Signalling MIB	pktcSigDefMediaStreamTos	The default value used in the IP header for setting the TOS value for NCS media stream packets.
MTA UDP receive port used for NCS	Integer (1025..65535)	W, optional	R/O	MTA Signalling MIB	pktcSigDefNcsReceiveUdpPort	This object contains the MTA User Datagram Protocol Receive Port that is used for NCS call signalling. This object should only be changed by the configuration file.
NCS TOS Format Selector	ENUM	W, optional	R/W	MTA Signalling MIB	pktcSigTosFormatSelector	The format of the default NCS signalling and media TOS values. Allowed values are "IPv4 TOS octet" or "DSCP codepoint". Refer to [i.5].
R0 cadence	Bit-field	W, optional	R/W	MTA Signalling MIB	pktcSigDevR0Cadence	User defined field where each bit (least significant bit) represents a duration of 100 milliseconds (6 seconds total) 1 = active ringing, 0 = silence 64 bits are used for representation; MSB 60 bits for ring cadence. Bit 61 is used to represent repeatable (when set to ZERO) and non repeatable (when set to ONE). Other three bits are reserved for future use, and currently set to 000.
R6 cadence	Bit-field	W, optional	R/W	MTA Signalling MIB	pktcSigDevR6Cadence	User defined field where each bit (least significant bit) represents a duration of 100 milliseconds (6 seconds total) 1 = active ringing, 0 = silence 64 bits are used for representation; MSB 60 bits for ring cadence. Bit 61 is used to represent repeatable (when set to ZERO) and non repeatable (when set to ONE). Other three bits are reserved for future use, and currently set to 000.
R7 cadence	Bit-field	W, optional	R/W	MTA Signalling MIB	pktcSigDevR7Cadence	User defined field where each bit (least significant bit) represents a duration of 100 milliseconds (6 seconds total) 1 = active ringing, 0 = silence 64 bits are used for representation; MSB 60 bits for ring cadence. Bit 61 is used to represent repeatable (when set to ZERO) and non repeatable (when set to ONE). Other three bits are reserved for future use, and currently set to 000.

Attribute	Syntax	Configuration Access	SNMP Access	MIB File	Object	pktcDevEvSysloComments
R1 cadence	Bit-field	W, optional	R/W	MTA Signalling MIB	pktcSigDevR1Cadence	User defined field where each bit (least significant bit) represents a duration of 100 milliseconds (6 seconds total) 1 = active ringing, 0 = silence 64 bits are used for representation; MSB 60 bits for ring cadence. Bit 61 is used to represent repeatable (when set to ZERO) and non repeatable (when set to ONE). Other three bits are reserved for future use, and currently set to 000.
R2 cadence	Bit-field	W, optional	R/W	MTA Signalling MIB	pktcSigDevR2Cadence	User defined field where each bit (least significant bit) represents a duration of 100 milliseconds (6 seconds total) 1 = active ringing, 0 = silence 64 bits are used for representation; MSB 60 bits for ring cadence. Bit 61 is used to represent repeatable (when set to ZERO) and non repeatable (when set to ONE). Other three bits are reserved for future use, and currently set to 000.
R3 cadence	Bit-field	W, optional	R/W	MTA Signalling MIB	pktcSigDevR3Cadence	User defined field where each bit (least significant bit) represents a duration of 100 milliseconds (6 seconds total) 1 = active ringing, 0 = silence 64 bits are used for representation; MSB 60 bits for ring cadence. Bit 61 is used to represent repeatable (when set to ZERO) and non repeatable (when set to ONE). Other three bits are reserved for future use, and currently set to 000.
R4 cadence	Bit-field	W, optional	R/W	MTA Signalling MIB	pktcSigDevR4Cadence	User defined field where each bit (least significant bit) represents a duration of 100 milliseconds (6 seconds total) 1 = active ringing, 0 = silence 64 bits are used for representation; MSB 60 bits for ring cadence. Bit 61 is used to represent repeatable (when set to ZERO) and non repeatable (when set to ONE). Other three bits are reserved for future use, and currently set to 000
R5 cadence	Bit-field	W, optional	R/W	MTA Signalling MIB	pktcSigDevR5Cadence	User defined field where each bit (least significant bit) represents a duration of 100 1 = active ringing, 0 = silence 64 bits are used for representation; MSB 60 bits for ring cadence. Bit 61 is used to represent repeatable (when set to ZERO) and non repeatable (when set to ONE). Other three bits are reserved for future use, and currently set to 000.
Rg cadence	Bit-field	W, optional	R/W	MTA Signalling MIB	pktcSigDevRgCadence	User defined field where each bit (least significant bit) represents a duration of 100 milliseconds (6 seconds total) 1 = active ringing, 0 = silence 64 bits are used for representation; MSB 60 bits for ring cadence. Bit 61 is used to represent repeatable (when set to ZERO) and non repeatable (when set to ONE). Other three bits are reserved for future use, and currently set to 000.

Attribute	Syntax	Configuration Access	SNMP Access	MIB File	Object	pktcDevEvSysloComments
Rt cadence	Bit-field	W, optional	R/W	MTA Signalling MIB	pktcSigDevRtCadence	User defined field where each bit (least significant bit) represents a duration of 100 milliseconds (6 seconds total) 1 = active ringing, 0 = silence 64 bits are used for representation; MSB 60 bits for ring cadence. Bit 61 is used to represent repeatable (when set to ZERO) and non repeatable (when set to ONE). Other three bits are reserved for future use, and currently set to 000.
Rs cadence	Bit-field	W, optional	R/W	MTA Signalling MIB	pktcSigDevRsCadence	User defined field where each bit (least significant bit) represents a duration of 100 milliseconds 1 = active ringing, 0 = silence 64 bits are used for representation; MSB 60 bits for ring cadence. Bit 61 is used to represent repeatable (when set to ZERO) and non repeatable (when set to ONE). Other three bits are reserved for future use, and currently set to 000.

9.1.3 Per-Endpoint Configuration Data

Refer to the SIGNALLING MIB ([3] and [22]), the NCS spec [4], the security spec [5] and the MTA MIB ([2] and [21]) for more detailed information concerning these attributes and their default values.

MTA sends KDC the MTA/CMS certificate, MTA's FQDN, CMS-ID. The KDC returns the MTA a "Kerberos Ticket" that says "this MTA is assigned to this CMS".

The Telephony Service Provider Certificate validates the MTA Telephony Certificate.

If two different endpoints share the same Kerberos Realm and same CMS FQDN, then these four attributes must be identical: PKINIT grace period, KDC name list, MTA telephony certificate, telephony service provider certificate.

Table 14

Attribute	Syntax	Access	SNMP Access	MIB File	Object	Comments
Port Admin State	ENUM	W, optional	R/W	IF-MIB [24]	ifAdminStatus	The administrative state of the port the operator can access to either enable or disable service to the port. The administrative state can be used to disable access to the user port without de-provisioning the subscriber. Allowed values for this attribute are: up(1) or down(2). For SNMP access ifAdminStatus is found in the ifTable of MIB-II.
Call Management Server Name	String	W, required	R/W	MTA Signalling MIB	pktcNcsEndPntConfigCallAgentId	This attribute is a string indicating the name of the CMS assigned to the endpoint. The call agent name after the character '@' must be a fully qualified domain name and must have a corresponding conceptual row in the pktcMtaDevCmsTable. DNS support is assumed to support multiple CMS's as described in the NCS spec.

Attribute	Syntax	Access	SNMP Access	MIB File	Object	Comments
Call Management Server UDP Port	Integer	W	R/W	MTA Signalling MIB	pktcNcsEndPntConfigCallAgentUdpPort	UDP port for the CMS.
Partial Dial Timeout	Integer	W	R/W	MTA Signalling MIB	pktcNcsEndPntConfigPartialDialTO	Timeout value in seconds for partial dial timeout.
Critical Dial Timeout	Integer	W	R/W	MTA Signalling MIB	pktcNcsEndPntConfigCriticalDialTO	Timeout value in seconds for critical dial timeout.
Busy Tone Timeout	Integer	W	R/W	MTA Signalling MIB	pktcNcsEndPntConfigBusyToneTO	Timeout value in seconds for busy tone.
Dial tone timeout	Integer	W	R/W	MTA Signalling MIB	pktcNcsEndPntConfigDialToneTO	Timeout value in seconds for dialtone.
Message Waiting timeout	Integer	W	R/W	MTA Signalling MIB	pktcNcsEndPntConfigMessageWaitingTO	Timeout value in seconds for message waiting.
Off Hook Warning timeout	Integer	W	R/W	MTA Signalling MIB	pktcNcsEndPntConfigOffHookWarnToneTO	Timeout value in seconds for off hook warning.
Ringing Timeout	Integer	W	R/W	MTA Signalling MIB	pktcNcsEndPntConfigRingingTO	Timeout value in seconds for ringing.
Ringback Timeout	Integer	W	R/W	MTA Signalling MIB	pktcNcsEndPntConfigRingBackTO	Timeout value in seconds for ringback.
Reorder Tone timeout	Integer	W	R/W	MTA Signalling MIB	pktcNcsEndPntConfigReorderToneTO	Timeout value in seconds for reorder tone.
Stutter dial timeout	Integer	W	R/W	MTA Signalling MIB	pktcNcsEndPntConfigStutterDialToneTO	Timeout value in seconds for stutter dial tone.
TS Max	Integer	W	R/W	MTA Signalling MIB	pktcNcsEndPntConfigTSMax	Contains the maximum time in seconds since the sending of the initial datagram.
Max1	Integer	W	R/W	MTA Signalling MIB	pktcNcsEndPntConfigMax1	The suspicious error threshold for each endpoint retransmission.
Max2	Integer	W	R/W	MTA Signalling MIB	pktcNcsEndPntConfigMax2	The disconnect error threshold per endpoint retransmission.
Max1 Queue Enable	Enum	W	R/W	MTA Signalling MIB	pktcNcsEndPntConfigMax1QEnable	Enables/disables the Max1 DNS query operation when Max1 expires.
Max2 Queue Enable	Enum	W	R/W	MTA Signalling MIB	pktcNcsEndPntConfigMax2QEnable	Enables/disables the Max2 DNS query operation when Max2 expires.
MWD	Integer	W	R/W	MTA Signalling MIB	pktcNcsEndPntConfigMWD	Number of seconds to wait to restart after a restart is received.
Tdinit	Integer	W	R/W	MTA Signalling MIB	pktcNcsEndPntConfigTdinit	Number of seconds to wait after a disconnect.
TDMin	Integer	W	R/W	MTA Signalling MIB	pktcNcsEndPntConfigTdmin	Minimum number of seconds to wait after a disconnect.
TDMax	Integer	W	R/W	MTA Signalling MIB	pktcNcsEndPntConfigTdmax	Maximum number of seconds to wait after a disconnect.
RTO Max	Integer	W	R/W	MTA Signalling MIB	pktcNcsEndPntConfigRtoMax	Maximum number of seconds for the retransmission timer.
RTO Init	Integer	W	R/W	MTA Signalling MIB	pktcNcsEndPntConfigRtoInit	Initial value for the retransmission timer.

Attribute	Syntax	Access	SNMP Access	MIB File	Object	Comments
Long Duration Keepalive	Integer	W	R/W	MTA Signalling MIB	pktcNcsEndPntConfigLong DurationKeepAlive	Timeout in minutes for sending long duration call notification messages.
Thist	Integer	W	R/W	MTA Signalling MIB	pktcNcsEndPntConfigThist	The timeout period in seconds before no response is declared.
Call Waiting Max Reps	Integer	W, optional	R/W	MTA Signalling MIB	pktcNcsEndPntConfigCall WaitingMaxRep	This object contains the maximum number of repetitions of the call waiting that the MTA will play from a single CMS request. A value of zero (0) will be used when the CMS invokes any play repetition
Call Waiting Delay	Integer	W, optional	R/W	IF-MIB [24]	pktcNcsEndPntConfigCall WaitingDelay	This object contains the delay between repetitions of the call waiting that the MTA will play from a single CMS request.

9.1.4 Per-Realm Configuration Data

Refer to the MTA MIB ([2] and [21]) for more detailed information concerning these attributes and their default values. Refer to the Security spec [5] for more information on the use of Kerberos realms. There must be at least one conceptual row in the pktcMtaDevRealmTable to establish service upon completion of configuration. These configuration parameters are optional in the config file, but if included the config file must contain at least one Realm name to permit proper instantiation of the table. There may be more than one set of entries if multiple realms are supported.

Table 15

Attribute	Syntax	Access	SNMP Access	MIB File	Object	Comments
Pkinit Grace Period	Integer	W, optional	R/W	MTA Device MIB	pktcMtaDevRealmPkinit GracePeriod	For the purpose of IPsec key management with a CMS, the MTA must obtain a new Kerberos ticket (with a PKINIT exchange) this many minutes before the old ticket expires. The minimum allowable value is 15 min. The default is 30 min. This parameter may also be used with other Kerberized applications.
TGS Grace Period	Integer	W, optional	R/W	MTA Device MIB	pktcMtaDevRealmTgs GracePeriod	When the MTA implementation uses TGS Request/TGS Reply Kerberos messages for the purpose of IPsec key management with the CMS, the MTA must obtain a new service ticket for the CMS (with a TGS request) this many minutes before the old ticket expires. The minimum allowable value is 1 min. The default is 10 min. This parameter may also be used with other Kerberized applications.
Realm Org Name	Integer	W, required	R/W	MTA Device MIB	pktcMtaDevRealmOrg Name	The value of the X.500 organization name attribute in the subject name of the Service provider certificate.
Unsolicited Keying max Timeout	Integer	W, optional	R/W	MTA Device MIB	PktcMtaDevRealm UnsolicitedKeyMaxTime out	This timeout applies only when the MTA initiated key management. The maximum timeout is the value which may not be exceeded in the exponential backoff algorithm.
Unsolicited Keying Nominal Timeout	Integer	W, optional	R/W	MTA Device MIB	PktcMtaDevRealm UnsolicitedKeyNomTim eout	This timeout applies only when the MTA initiated key management. Typically this is the average roundtrip time between the MTA and the KDC.

Attribute	Syntax	Access	SNMP Access	MIB File	Object	Comments
Unsolicited Keying Max Retries	Integer	W, optional	R/W	MTA Device MIB	PktcMtaDevRealm UnsolicitedKeyMaxRetries	This is the maximum number of retries before the MTA gives up attempting to establish a Security Association.

9.1.5 Per-CMS Configuration Data

Refer to the MTA MIB ([2] and [21]) for more detailed information concerning these attributes and their default values. There must be at least one conceptual row in the pktcDevCmsTable to establish service upon completion of configuration. These configuration parameters are optional in the config file, but if included the config file must identify at least one CMS and its corresponding Kerberos Realm Name. There may be more than one set of entries if multiple CMSs are supported.

As per Security [5], the IPsec signalling security must be controlled by the Operator depending on the deployment and operational conditions. As the IPsec Security Association is established between the MTA and the CMS, the IPsec enabling/disabling control should also be on per CMS basis. Enabling/Disabling of the IPsec Signalling Security must be defined only by the information in the MTA's Configuration File when the file is being downloaded, and the enable/disable toggling must be done only as a result of the MTA Reset.

For more details on the MIB Object controlling the IPsec enabling/disabling, refer to the MTA MIB ([2] and [21]).

Table 16

Attribute	Syntax	Access	SNMP Access	MIB File	Object	Comments
Kerberos Realm Name	String	W, required (see note)	R/W	MTA Device MIB	pktcMtaDevCmsKerbRealmName	The name for the associated Kerberos Realm. This is the corresponding Kerberos Realm Name in the Per Realm Configuration Data.
CMS Maximum Clock Skew	Integer	W, optional	R/W	MTA Device MIB	pktcMtaDevCmsMaxClockSkew	This is the maximum allowable clock skew between the MTA and CMS.
CMS Solicited Key Timeout	Integer	W, optional	R/W	MTA Device MIB	pktcMtaDevCmsSolicitedKeyTimeout	This timeout applies only when the CMS initiated key management (with a Wake Up or Rekey message). It is the period during which the MTA will save a nonce (inside the sequence number field) from the sent out AP Request and wait for the matching AP Reply from the CMS.
Unsolicited Key Max Timeout	Integer	W, optional	R/W	MTA Device MIB	pktcMtaDevCmsUnsolicitedKeyMaxTimeout	This timeout applies only when the MTA initiated key management. The maximum timeout is the value which may not be exceeded in the exponential backoff algorithm.
Unsolicited Key Nominal Timeout	Integer	W. optional	R/W	MTA Device MIB	pktcMtaDevCmsUnsolicitedKeyNomTimeout	This timeout applies only when the MTA initiated key management. Typically this is the average roundtrip time between the MTA and the CMS.
Unsolicited Key Max Retries	Integer	W, optional	R/W	MTA Device MIB	pktcMtaDevCmsUnsolicitedKeyMaxRetries	This is the maximum number of retries before the MTA gives up attempting to establish a security association.
IPsec Control	Integer	W, optional	R/O	MTA Device MIB	pktcMtaDevCmsIpsecCtrl	IPsec Control for each CMS: controls the IPsec establishment and IPsec related Key Management.
NOTE: If any data from the Per-CMS Data Table is included in the config file, this entry must be included.						

9.1.6 Exclusion of MIB objects in configuration File

The following MIB objects must not be sent in the configuration file since the values of these object can be either set only by the MTA or by DHCP options during provisioning. If an MTA receives the following MIB objects in its configuration file, the MTA must ignore the object and report "passWithWarnings" and populate the Error OIDs Table.

- PktcMtaDevSnmpEntity
- PktcMtaDevProvKerbRealmName
- PktcMtaDevFqdn
- PktcMtaDevSerialNumber
- PktcMtaDevMacAddress
- PktcMtaDevEndPntCount
- PktcMtaDevTypeIdentifier
- PktcEnNcsEndPntQuarantineState
- PktcEnNcsEndPntHookState
- pktcEnEndPntInfoTable
- pktcDevEventDescrEnterprise
- pktcDevEventDescrFacility
- pktcDevEventDescrText
- pktcDevEvLogIndex
- pktcDevEvLogTime
- pktcDevEvLogLevel
- pktcDevEvLogId
- pktcDevEvLogText
- pktcDevEvLogEndpointName
- pktcDevEvLogType
- pktcDevEvLogTargetInfo
- pktcDevEvLogCorrelationId
- pktcMtaDevProvConfigKey

NOTE: For Syslog entries, specifically the MIB Objects "pktcDevEvSyslogAddressType" and "pktcDevEvSyslogAddress", the MTA validates the 'type' provided (or stored) with the provided (or stored) 'Syslog Address' - if they are inconsistent, it ignores any such entries in the configuration file, report a 'passWithWarnings' and populate the Error OIDs Table.

10 MTA Device Capabilities

MTA Capabilities string is supplied to the Provisioning Server in Option code 60 (Vendor Class Identifier) to allow the Back-Office to differentiate between MTAs during the Provisioning Process. Use of Capabilities information by the Provisioning Application is optional.

Capabilities string is encoded as an ASCII string containing the Capabilities information in Type/Length/Value (TLV) Format.

For example, the ASCII encoding of the first two TLVs (IPCablecom Version 1.5 and Number of Telephony Endpoints = 2) of an MTA would be 05nn010101020102. Note that many more TLVs are required for IPCablecom MTA, and the field "nn" will contain the length of all the TLVs. This example shows only two TLVs for simplicity.

The "value" field describes the capabilities of a particular modem, i.e. implementation dependent limits on the particular features or number of features, which the modem can support. It is composed from a number of encapsulated TLV fields. The encapsulated sub-types define the specific capabilities for the MTA.

NOTE: The sub-type fields defined are only valid within the encapsulated capabilities configuration setting string.

Type	Length	Value
5	n	

The set of possible encapsulated fields is described below.

MTA must Send Capabilities String in option 60 of the DHCP DISCOVER request.

10.1 IPCablecom Version

This TLV of subtype 5.1 (IPCablecom Version) must be supplied in the Capabilities String.

Type	Length	Values	Comment	Default Value
5.1	1	0	IPCablecom 1.0	NONE
		1	IPCablecom 1.5	

10.2 Number Of Telephony Endpoints

This TLV of subtype 5.2 (Number of telephony Endpoints) must be supplied in the Capabilities String.

Type	Length	Values	Comment	Default
5.2	1	n	Number of endpoints	NONE

10.3 TGT Support

Type	Length	Value	Comment	Default Value
5.3	1	0	0: No	0
		1	1: Yes	

10.4 HTTP Download File Access Method Support

Type	Length	Value	Comment	Default Value
5.4	1	0	0: No	0
		1	1: Yes	

10.5 MTA-24 Event SYSLOG Notification Support

Type	Length	Value	Comment	Default Value
5.5	1	0	0: No	1
		1	1: Yes	

10.6 NCS Service Flow Support

Type	Length	Value	Comment	Default Value
5.6	1	Undefined	Reserved for future use	Undefined

Sub Type 5.6 which was previously used to indicate support for NCS Service Flow functionality, is currently undefined and reserved for future usage.

10.7 Primary Line Support

Type	Length	Value	Comment	Default Value
5.7	1	0	0: No	0
		1	1: Yes	

10.8 Vendor Specific TLV Type(s)

This TLV can be supplied in the Capabilities String if an MTA requires a specific processing of the Vendor Specific TLV Type(s).

Type	Length	Value	Comment	Default Value
5.8	n	{seq-of-bytes}:	One type per byte per byte	43

Sub Type 5.8 which was previously used to indicate vendor specific TLV support by MTAs is currently obsoleted and the sub-type (5.8) reserved for future usage. This must not be used by MTAs in IPCablecom 1.5.

10.9 NVRAM Ticket/Ticket Information Storage Support

Type	Length	Value	Comment	Default Value
5.9	1	0	0: No	1
		1	1: Yes	

10.10 Provisioning Event Reporting Support

Type	Length	Value	Comment	Default Value
5.10	1	0	0: No	1
		1	1: Yes	

10.11 Supported CODEC(s)

This TLV must be supplied in the Capabilities String.

Type	Length	Value	Comment	Default Value
5.11	n	{seq-of-bytes}	one ID per byte	NONE

CODEC ID is the value represented by the Enumerated Type of "PktcCodecType" TEXTUAL CONVENTION in MTA MIB:

1: other,

2: unknown,

3: G.729,

4: reserved,

5: G.729E,

6: PCMU,

7: G.726-32

8: G.728,

9: PCMA,

10: G.726-16,

11: G.726-24,

12: G.726-40,

13: iLBC,

14: BV16,

15: telephone-event

Telephone-event represents RFC2833 DTMF events [25]. For more information refer to [i.6].

10.12 Silence Suppression Support

Type	Length	Value	Comment	Default Value
5.12	1	0	0: No	0
		1	1: Yes	

10.13 Echo Cancellation Support

Type	Length	Value	Comment	Default Value
5.13	1	0	0: No	0
		1	1: Yes	

10.14 RSVP Support

Type	Length	Value	Comment	Default Value
5.14	1	Undefined	Reserved for future Usage	Undefined

Sub Type 5.14 which was previously used to indicate RSVP support is currently undefined and reserved for future usage.

10.15 UGS-AD Support

Type	Length	Value	Comment	Default Value
5.15	1	0	0: No	0
		1	1: Yes	

10.16 MTA's "ifIndex" starting number in "ifTable"

This TLV contains the value of the "ifIndex" for the first MTA Telephony Interface in "ifTable" MIB Table. The TLV must be supplied in the Capabilities String.

Type	Length	Value	Comment	Default Value
5.16	1	n	first MTA Interface	9

10.17 Provisioning Flow Logging Support

This capability is set to a corresponding value depending on the support of the logging capability of the Provisioning Flow (as per clause 5.4.3).

Type	Length	Value	Comment	Default Value
5.17	1	0	0: No	1
		1	1: Yes	

10.18 Supported Provisioning Flows

An MTA must include this TLV of subtype 5.18 (Supported Provisioning flows) in the Capabilities String. This TLV indicates the provisioning flows the MTA supports (Basic, Hybrid and Secure). It contains a bitmask indicating all the provisioning flows supported by the MTA.

Type	Length	Value	Comment	Default Value
5.18	2	{bit-mask}	See below	NONE

The Value field is an unsigned 16 bit integer encoded in network byte order. Each bit represents a specific provisioning flow. If a bit set to 1, the MTA supports the corresponding flow. If a bit is set to 0 (zero), the MTA does not support the flow.

Bit assignments:

Bit 0 - Secure Flow (Full Secure Provisioning Flow)

Bit 1 - Hybrid Flow

Bit 2 - Basic Flow

The MTA must set all unused bits in the bitmask to 0. The MTA must set bit 0 in the TLV to 1 to indicate that it supports the Secure Flow. The MTA must set bits 1 and 2 in the TLV to indicate whether it supports the Basic and Hybrid Flows. An example: if an MTA supports Secure and Basic Provisioning Flows, the integer value of the mask is 0x0005, and the capability will be encoded in Option-60 as the following sequence of octets (in HEX notation): 12 02 00 05.

To provide backward compatibility prior to the introduction of the Basic & Hybrid Flows, the absence of this TLV indicates that the MTA only supports the Secure Flow.

10.19 T38 Version Support

An MTA must include this TLV of subtype 5.19(T38 Version Support) in the Capabilities String. This TLV indicates the version of T.38 the MTA supports. For more details refer [i.6].

Type	Length	Value	Comment	Default Value
5.19	1	0	0: Unsupported:	1
		1	1:Version Zero	
		2	2:Version One	
		3	3: Version Two	
		4	4: Version Three	

10.20 T38 Error Correction Support

An MTA must include this TLV of subtype 5.20 (T38 Error Correction Support) in the Capabilities String. This TLV indicates the type of error correction the MTA supports for T.38. For more details refer [i.6].

Type	Length	Values	Comment	Default Value
5.20	1	0	0: None	1
		1	1: Redundancy	
		2	2: FEC	

If you support FEC, it means you also support Redundancy. For more information refer to [i.6]

10.21 RFC 2833 DTMF Support

An MTA must include this TLV of subtype 5.21 [25] in the Capabilities String. This TLV indicates the support for RFC2833 DTMF relay [25]. For more details refer to [i.6].

Type	Length	Value	Comment	Default Value
5.21	1	0	0: No	1
		1	1: Yes	

10.22 Voice Metrics Support

An MTA must include this TLV of sub type 5.22 (Voice Metrics Support) in the Capabilities String. This TLV indicates the support for voice metrics as defined in [33].

Type	Length	Value	Comment	Default Value
5.22	1	0	0: No	1
		1	1: Yes	

10.23 Device MIB Support

An MTA must include this TLV of subtype 5.23 (Device MIB support) in the Capabilities String. This TLV indicates the various MIBs supported by the MTA.

Type	Length	Values	Comment	Default Value
5.23	n	{seq-of-bytes}	MIB Support encoded as 'length-value' pairs.	NONE

The 'length-value' pairs are defined as follows:

[L1] [OCTET-1] [OCTET-2][OCTET-3] …[OCTET-L1],

[L2] [OCTET-1] [OCTET-2][OCTET-3] …[OCTET-L2]

(And other Length-Value pairs as deemed appropriate)

Where:

'L1' and 'L2' denote lengths

The first OCTET (OCTET-1) always represents the MIB issuing organization (i.e.: CableLabs, IETF etc).

The remaining OCTETS are always placed in network-byte order to form a bit string where each bit represents a particular MIB. Setting a bit (to a value of 1) indicates support for the representative MIB and unsetting a bit (to a value of 0) indicates absence of support for the representative MIB.

MTAs must not use any 'reserved assignments' unless defined by IPCablecom or assigned as 'vendor specific'.

10.23.1 Issuing Organization Assignments

OCTET-1 of the 'length-value' pair indicates the MIB issuing organization and the assignments are as follows:

Assignment	Organization Indicator
0	CableLabs
1	IETF
2	EuroCableLabs
3-9	*reserved*
10-63	*vendor-specific*

NOTE: The higher order two bits of OCTET-1 are reserved allowing for 64 possibilities.

10.23.2 Representing CableLabs® MIBs

For CableLabs® issued MIBs (OCTET-1 = 0) the bit mask is defined as follows:

Bit 0	IPCablecom 1.5 MTA MIB.
Bit 1	IPCablecom 1.5 Signalling MIB.
Bit 2	IPCablecom 1.5 Management Event MIB.
Bit 3	IPCablecom 1.5 MTA Extension MIB.
Bit 4	IPCablecom 1.5 Signalling Extension MIB.
Bit 5	IPCablecom 1.5 MEM Extension MIB.
Bit 6	*reserved*
Bit 7	*reserved*

Where the bits are placed as follows:

7	6	5	4	3	2	1	0

Given only one octet is used currently for the bit mask, the length for this length-value pair must be two (one each for the Organization Indicator and the bit mask, respectively).

10.23.3 Representing IETF MIBs

For MIBs represented as RFCs (OCTET-1 = 1) the bit mask is defined as follows:

Bit 0	MTA MIB.
Bit 1	Signalling MIB.
Bit 2	Management Event MIB.
Bit 3	*reserved*
Bit 4	*reserved*
Bit 5	*reserved*
Bit 6	*reserved*
*Bit 7	*reserved*

Given only one octet is used currently for the bit mask, the length for this length-value pair must be two (One each for the Organization Indicator and the bit mask, respectively).

10.23.4 Example

For an MTA that supports all defined IETF MIBs (MTA, Signalling and MEM) and all defined IPCablecom 1.5 extension MIBs (MTA extension, Signalling extension and MEM extension) this sub-option would be encoded (in Hex) as follows (taken as a snapshot of Option 60):

…	…	17	06	02	00	38	02	01	07	…	…

10.24 Multiple Grants Per Interval Support

An MTA must include this TLV of sub type 5.24 (Multiple Grants Per Interval Support) in the Capabilities String. This TLV indicates the support for Multiple Grants per interval. For more details refer to [i.7].

Type	Length	Value	Comment	Default Value
5.24	1	0	0: No	0
		1	1: Yes	

10.25 V.152 Support

An MTA must include this TLV of sub type 5.25 (V.152 Support) in the Capabilities String. This TLV indicates the support for V.152. For more details refer to [i.6].

Type	Length	Value	Comment	Default Value
5.25	1	0	0: No	1
		1	1: Yes	

11 TLV-38 SNMP Notification Receiver Specification

This IPCablecom TLV 38 specifies one or more Network Management Stations that must receive notifications from the MTA (MTA25 or H-MTA25 or B-MTA25 and post-provisioning, if required). If TLV38 and its sub-TLVs defined in this clause contains incorrect value in 'Length' field, the MTA must reject the configuration file and report a "failConfigFile" error. If TLV38 contains sub-types with wrong Values, then the MTA must follow the requirements specified below in each sub-TLV.

In addition if the MTA encounters unknown sub-TLVs within TLV38, it must

- Assume the length field size of 1 byte for the sub-TLV.
- Ignore the sub-TLV and continue with further processing.
- Report a provisioning state of passWithWarnings and populate Error OID Table.

Type	Length	Value
38	N	Composite (contains sub-TLVs)

Unless specified or configured otherwise, the MTA must send the notifications to the default provisioning system (defined in DHCP option 122 sub-option 3).

11.1 Sub-TLVs of TLV-38

11.1.1 SNMP Notification Receiver IP Address

This sub-TLV specifies the IP address of the notification receiver.

Type	Length	Value
38.1	4	4 bytes of an IPv4 address in network byte order

If TLV 38 is present in the configuration file and the sub-TLV 38.1 is absent, the MTA must ignore TLV-38 and proceed with further processing of the configuration file and must report a provisioning state of passWithWarnings and populate the error OID table (pktcMtaDevErrorOidsTable).

11.1.2 SNMP Notification Receiver UDP Port Number

This sub-TLV specifies the Port number on the notification receiver to receive the notifications.

Type	Length	Value
38.2	2	UDP Port Number

If TLV 38 is present and the sub-TLV 38.2 is absent, then a default value of 162 must be used.

11.1.3 SNMP Notification Receiver Type

This sub-TLV specifies the SNMP Notification Receiver Type; it is the type of SNMP notifications the MTA must send to the associated SNMP Notification Receiver.

Type	Length	Value
38.3	2	1: SNMPv1 trap in an SNMPv1 packet 2: SNMP v2c trap in an SNMP v2c packet 3: SNMP INFORM in an SNMP v2c packet 4: SNMP trap in an SNMPv3 packet 5: SNMP INFORM in a SNMPv3 packet

If TLV 38 is present in the configuration file but sub-TLV 38.3 is absent, the MTA must ignore the entire TLV-38 and proceed with further processing of the configuration file and must report passWithWarnings and populate the Error OID table (pktcMtaDevErrorOidsTable). The MTA and Provisioning Server must support notification type values 2 and 3, and may support notification type values 1,4 or 5 from the above table. If an unsupported or invalid notification type value is received, the MTA must ignore the entire TLV38 that contains this entry and must report passWithWarnings and populate the error OID table (pktcMtaDevErrorOidsTable). If the notification types of 4 or 5 are used in the Basic or Hybrid provisioning flows, SNMPv3 communication is assumed to be implemented as per SNMPv3 recommendations and is outside the scope of the present document.

11.1.4 SNMP Notification Receiver Timeout

This sub-TLV specifies the wait time before a retry is attempted when the sender of an SNMP INFORM fails to receive an acknowledgement. Note that the number of retries is defined in sub-TLV 38.5.

Type	Length	Value
38.4	2	Time in milliseconds

If TLV 38 is present in the configuration file and the sub-TLV 38.4 is absent, the MTA must assume a default value of 15000 milliseconds. This corresponds to the default value of 1500 hundredths of a second defined for the snmpTargetAddrTimeout MIB object ([28] and RFC 3413 [7]).

11.1.5 SNMP Notification Receiver Retries

This sub-TLV specifies the maximum number of times the MTA must retry sending an SNMP INFORM message if an acknowledgement is not received. Note that the wait time before each retry is defined by sub-TLV 38.4.

Type	Length	Value
38.5	2	Number of retries

If not present, the MTA must use a default value of 3. The maximum number of retries that can be specified is 255.

11.1.6 SNMP Notification Receiver Filtering Parameters

This sub-TLV specifies the filtering scheme for notifications and contains the root OID of the MIB sub-tree that defines the notifications to be sent to the Notification Receiver. The MTA must filter notifications being sent to the SNMP manager specified in sub-TLV 38.1 using the information provided. If this sub-TLV is not present, the MTA must use the default OID value for the 'iso' root.

Type	Length	Value
38.6	n	Filter OID (ASN.1 formatted Object Identifier)

The encoding of this TLV value field starts with the ASN.1 Universal type 6 (Object Identifier) followed by the ASN.1 length field and is terminated by the ASN.1 encoded object identifier component.

11.1.7 SNMPv3 Notification Receiver Security Name

This sub-TLV specifies the SNMPv3 Security Name to use when sending an SNMPv3 Notification. This sub-TLV is only being used if MTA supports TLV 38.3 (Notification Receiver Type) types 4 and 5. The MTA must ignore this sub-TLV 38.7 if a Notification Receiver Type (sub-TLV 38.3) other than 4 or 5 is received in the configuration file.

The following requirements apply to MTAs that supports Notification Receiver Type values of 4 or 5 in sub-TLV-38.3:

- If this sub-TLV38.7 is omitted, then the SNMPv3 Notifications must be sent in the noAuthNoPriv security level using the security name "@mtaconfig".
- If this sub-TLV is included, the MTA verifies that the value of the Security Name exists for the MTA local authoritative SNMP engine and creates an entry to further associate with the notification receiver authoritative engine (using the security levels and keys from the existing Security Name). If the Security Name of this sub-TLV does not exist for the local engine, the entire TLV 38 must be ignored and the MTA must reports passWithWarnings and populate the Error OID table (pktcMtaDevErrorOidsTable) for the entire TLV 38 and associated sub-TLVs that are ignored.

Type	Length	Value
38.7	2 to 26	Security Name

11.2 Mapping of TLV fields into SNMP Tables

The following clauses detail the MTA configuration file TLV-38 "IPCablecom SNMP Notification Receiver" mapping onto SNMP functional tables.

Upon receiving each TLV 38 value, the MTA must make entries to the following tables in order to cause the desired SNMP TRAP or INFORM transmission: snmpNotifyTable, snmpTargetAddrTable, snmpTargetAddrExtTable, snmpTargetParamsTable, snmpNotifyFilterProfileTable, snmpNotifyFilterTable, snmpCommunityTable, usmUserTable, vacmSecurityToGroupTable, vacmAccessTable, and vacmViewTreeFamilyTable. An MTA must support a minimum of ten TLV-38 elements in a configuration file.

11.2.1 Mapping of TLV fields into created SNMP Table rows

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

The correspondence between the tags and the sub-TLVs themselves is as shown below:

<IP Address>	TLV 38.1
<Port>	TLV 38.2
<Trap type>	TLV 38.3
<Timeout>	TLV 38.4
<Retries>	TLV 38.5
<Filter OID>	TLV 38.6
<Security Name>	TLV 38.7

The creation of rows with column values or indices containing the suffix "n" in the tables below indicates that these entries are created with the (n-1)th TLV 38 found in the MTA configuration file

11.2.1.1 snmpNotifyTable

If TLV38 elements are present and irrespective of the number of elements the MTA must create two rows with fixed values as described in table 17.

Table 17: snmpNotifyTable

snmpNotifyTable (RFC 3413 [7], SNMP-NOTIFICATION-MIB)	First Row	Second Row
Column Name (* = Part of Index)	Column Value	Column Value
* snmpNotifyName	"@mtaconfig_inform"	"@mtaconfig_trap"
snmpNotifyTag	"@mtaconfig_inform"	"@mtaconfig_trap"
snmpNotifyType	inform (2)	trap (1)
snmpNotifyStorageType	Volatile	Volatile
snmpNotifyRowStatus	active (1)	active (1)

11.2.1.2 snmpTargetAddrTable

For each TLV38 element in the configuration file, the MTA must create one row according to table 18.

Table 18: snmpTargetAddrTable

snmpTargetAddrTable (RFC 3413 [7], SNMP-TARGET-MIB)	New Row
Column Name (* = Part of Index)	Column Value
* snmpTargetAddrName	"@mtaconfig_n" Where n ranges from 0 to m-1 where m is the number of notification receiver TLV elements in the Configuration file
snmpTargetAddrTDomain	snmpUDPDomain = snmpDomains.1
snmpTargetAddrTAddress (IP Address and UDP Port of the Notification Receiver)	OCTET STRING (6) Octets 1-4: <IP Address> Octets 5-6: <Port>
snmpTargetAddrTimeout	<Timeout> from the TLV
snmpTargetAddrRetryCount	<Retries> from the TLV
snmpTargetAddrTagList	If <Trap type> == 2 "@mtaconfig_trap" Else If <Trap type> = 3 "@mtaconfig_inform"
snmpTargetAddrParams	"@mtaconfig_n" (Same as snmpTargetAddrName value)
snmpTargetAddrStorageType	Volatile
snmpTargetAddrRowStatus	active (1)

11.2.1.3 snmpTargetAddrExtTable

For each TLV38 element in the configuration file, the MTA must create one row according to table 19.

Table 19: snmpTargetAddrExtTable

snmpTargetAddrExtTable (RFC 3584 [32], SNMP-COMMUNITY-MIB)	New Row
Column Name (* = Part of Index)	Column Value
* snmpTargetAddrName	"@mtaconfig_n" Where n ranges from 0 to m-1 where m is the number of notification receiver TLV elements in the Configuration file
snmpTargetAddrTMask	<Zero length octet string>
snmpTargetAddrMMS	0

11.2.1.4 snmpTargetParamsTable

For each TLV 38 element in the configuration file MTA must create one row according to table 20.

Table 20: snmpTargetParamsTable

snmpTargetParamsTable (RFC 3413 [7], SNMP-TARGET-MIB)	New Row
Column Name (* = Part of Index)	Column Value
* snmpTargetParamsName	"@mtaconfig_n" Where n ranges from 0 to m-1 where m is the number of notification receiver TLV elements in the Configuration file
snmpTargetParamsMPModel SYNTAX: snmpMessageProcessingModel	SNMPv2c (1)
snmpTargetParamsSecurityModel SYNTAX: snmpSecurityModel	SNMPv2c (2) NOTE: The mapping of SNMP protocol types to value here are different from snmpTargetParamsMPModel
snmpTargetParamsSecurityName	"@mtaconfig"
snmpTargetParamsSecurityLevel	NoAuthNoPriv
snmpTargetParamsStorageType	Volatile
snmpTargetParamsRowStatus	active (1)

11.2.1.5 snmpNotifyFilterProfileTable

For each TLV 38 element in the configuration file with non zero value of TLV38 sub type 6 MTA must create one row according to table 21.

Table 21: snmpNotifyFilterProfileTable

snmpNotifyFilterProfileTable (RFC 3413 [7], SNMP-NOTIFICATION-MIB)	New Row
Column Name (* = Part of Index)	Column Value
* snmpTargetParamsName	"@mtaconfig_n" Where n ranges from 0 to m-1 where m is the number of notification receiver TLV elements in the Configuration file
snmpNotifyFilterProfileName	"@mtaconfig_n" Where n ranges from 0 to m-1 where m is the number of notification receiver TLV elements in the Configuration file
snmpNotifyFilterProfileStorType	volatile
snmpNotifyFilterProfileRowStatus	active (1)

11.2.1.6 snmpNotifyFilterTable

For each TLV 38 element in the configuration file with non zero value of TLV38 sub type 6 MTA must create one row according to table 22.

Table 22: snmpNotifyFilterTable

snmpNotifyFilterTable (RFC 3413 [7], SNMP-NOTIFICATION-MIB)	New Row
Column Name (* = Part of Index)	Column Value
* snmpNotifyFilterProfileName	"@mtaconfig_n" Where n ranges from 0 to m-1 where m is the number of notification receiver TLV elements in the Configuration file
* snmpNotifyFilterSubtree	<Filter OID> from the TLV
snmpNotifyFilterMask	<Zero Length Octet String>
snmpNotifyFilterType	included (1)
snmpNotifyFilterStorageType	Volatile
snmpNotifyFilterRowStatus	active (1)

11.2.1.7 snmpCommunityTable

If TLV38 elements are present and irrespective of the number of elements the MTA must create one row with fixed values as described in table 23.

Table 23: snmpCommunityTable

snmpCommunityTable (RFC 3584 [32], SNMP-COMMUNITY-MIB)	First Row
Column Name (* = Part of Index)	Column Value
* snmpCommunityIndex	"@mtaconfig"
snmpCommunityName	"public"
snmpCommunitySecurityName	"@mtaconfig"
snmpCommunityContextEngineID	<The engineID of the MTA>
snmpCommunityContextName	<Zero length octet string>
snmpCommunityTransportTag	<Zero length octet string>
snmpCommunityStorageType	Volatile
snmpCommunityStatus	active (1)

11.2.1.8 usmUserTable

The usmUserTable is defined in RFC 3414 [8]. The entries in the table specify the user name on the remote notification receiver to which notification is to be sent. Rows in usmUserTable are created in two different ways when <Notification Receiver Type> (TLV-38.3) values 4 and 5 are supported by the MTA and is included in TLV38:

- If <Security Name> (TLV-38.7) is not included, irrespective of the number of TLV 38 elements in the configuration file, the MTA must create one entry row with fixed values as described by the first column ("Static row") in table 10.
- If <Security Name> (TLV38.7) is included then MTA must create additional entry rows as described by the second column ("Other Rows") in table 24. In this case, the creation of additional rows in usmUserTable occurs each time the engine ID of a notification receiver needs to be discovered (see RFC 3414 [8] for more details).

Table 24: usmUserTable

usmUserTable (RFC 3414 [8], SNMP-USER-BASED-SM-MIB)	Static Row Case 1	Other Rows Case 2
Column Name (* = Part of Index)	Column Value	Column Value
* usmUserEngineID	0x00, create a new row on each time the EngineID of the Authoritative Notification Receiver is discovered.	0x00, create a new row on each time the EngineID of the Authoritative Notification Receiver is discovered.
usmUserName	"@mtaconfig".	When other rows are created, this is replaced with the <Security Name> field from the TLV element.
usmUserSecurityName	"@mtaconfig"	When other rows are created, this is replaced with the <Security Name> field from the TLV element
usmUserCloneFrom	<ignore> (zerodotZero) This row is not created by cloning mechanism	<ignore> (zerodotZero) This row is not created by cloning mechanism
usmUserAuthProtocol	None (usmNoAuthProtocol)	When other rows are created, this is replaced with none (usmNoAuthProtocol), or MD5 (usmHMACMD5AuthProtocol),or SHA (usmHMACSHAAuthProtocol) depending of the security level of the SNMPv3 user.
usmUserAuthKeyChange	Empty	Empty
usmUserOwnAuthKeyChange	Empty	Empty
usmUserPrivProtocol	Case 1: none (usmNoPrivProtocol)	When other rows are created this is replaced with none (usmNoPrivProtocol) or DES (usmDESPrivProtocol), depending of the security level of the SNMPv3 user.
usmUserPrivKeyChange	Empty	Empty
usmUserOwnPrivKeyChange	Empty	Empty
usmUserPublic	Empty	Empty
usmUserStorageType	volatile(2)	volatile(2)
usmUserStatus	active (1)	active (1)

11.2.1.9 vacmSecurityToGroupTable

If TLV38 elements are present and irrespective of the number of elements the MTA must create "Second Row" column and may create "First Row" or "Third Row" columns with fixed values as described in table 25 MTA must populate "Second Row" and "Third Row" columns for Secure Provisioning Flow only.

Table 25: vacmSecurityToGroupTable

vacmSecurityToGroupTable (RFC 3415 [9], SNMP-VIEW-BASED-ACM-MIB)	First Row	Second Row	Third Row
Column Name (* = Part of Index)	Column Value	Column Value	Column Value
* vacmSecurityModel	SNMPV1 (1)	SNMPV2c (2)	SNMPUSM (3)
* vacmSecurityName	"@mtaconfig"	"@mtaconfig"	"@mtaconfig"
vacmGroupName	"@mtaconfigV1"	"@mtaconfigV2"	"@mtaconfigUSM"
vacmSecurityToGroupStorageType	volatile(2)	volatile(2)	volatile(2)
vacmSecurityToGroupStatus	active (1)	active (1)	active (1)

11.2.1.10 VacmAccessTable

If TLV38 elements are present and irrespective of the number of elements the MTA must create "Second Row" column and may create "First Row" or "Third Row" columns with fixed values as described in table 26. MTA must populate "Second Row" and "Third Row" columns for Secure Provisioning Flow only.

Table 26: vacmAccessTable

vacmAccessTable (RFC 3415 [9], SNMP-VIEW-BASED-ACM-MIB)	First Row	Second Row	Third Row
Column Name (* = Part of Index)	Column Value	Column Value	Column Value
* vacmGroupName	"@mtaconfigV1"	"@mtaconfigV2"	"@mtaconfigUSM"
* vacmAccessContextPrefix	Empty	Empty	Empty
* vacmAccessSecurityModel	SNMPv1 (1)	SNMPv2c (2)	USM (3)
* vacmAccessSecurityLevel	noAuthNoPriv (1)	noAuthNoPriv (1)	noAuthNoPriv (1)
vacmAccessContextMatch	exact (1)	exact (1)	exact (1)
vacmAccessReadViewName	Empty	Empty	Empty
vacmAccessWriteViewName	Empty	Empty	Empty
vacmAccessNotifyViewName	"@mtaconfig"	"@mtaconfig"	"@mtaconfig"
vacmAccessStorageType	volatile(2)	volatile(2)	volatile(2)
vacmAccessStatus	active (1)	active (1)	active (1)

11.2.1.11 vacmViewTreeFamilyTable

If TLV38 elements are present and irrespective of the number of elements the entry below as defined in table 27 must be created. Note that this entry is already created at MTA initialization.

Table 27: vacmViewTreeFamilyTable

vacmViewTreeFamilyTable (RFC 3415 [9], SNMP-VIEW-BASED-ACM-MIB)	First Row
Column Name (* = Part of Index)	Column Value
* vacmViewTreeFamilyViewName	"@mtaconfig"
* vacmViewTreeFamilySubtree	1.3
vacmViewTreeFamilyMask	<Default from MIB>
vacmViewTreeFamilyType	included (1)
vacmViewTreeFamilyStorageType	volatile
vacmViewTreeFamilyStatus	active (1)

11.3 TLV38 and TLV11 Configuration Example

This clause presents configuration examples for the generation of TLV-38 and TLV-11 for the purpose of SNMP framework configuration based on the framework model and message processing defined in RFC 3410 [i.3], RFC 3411 [i.4], and RFC 3412 [27].

11.3.1 TLV-38 Example

This clause is informational. The example below presents the usability of TLV-38. One of the objectives of this clause is to illustrate the usage of @mtaConfig_n. The following assumptions are made:

- MTA ignores entries with <trap type> 1 and supports <trap type> 2,3, 4 and 5
- MTA already via a configuration process has an entry with usmUserName and usmUserSecurityName which is 'mtaUser' and another entry set for 'superUser'. For simplification, no VACM entries associated to this profile are included

Table 28 contains the Configuration File elements. Empty cells means use default values when applicable.

Table 28: Example Configuration File elements

Sub-TLV					
TLV38 order in the Configuration file	TLV 38 Number 1	TLV 38 Number 2	TLV38 Number 3	TLV38 Number 4	TLV 38 Number 5
SNMP Notification Receiver IP Address	10.0.5.9	10.0.5.9	10.0.4.9	10.0.4.9	10.0.8.9
SNMPv2c Notification Receiver UDP Port Number		162		57 000	
SNMPv2c Notification Receiver Trap Type	2	3	1	4	5
SNMPv2c Notification Receiver Timeout	1 500		2 000		
SNMPv2c Notification Receiver Retries	3	1	2		
Notification Receiver Filtering Parameters	org	pktcMtaDevProvisioningStatus	mib-2	pktcMtaMib	pktcMtaDevProvisioningStatus
Notification Receiver Security Name		not used		SuperUser	mtaUser
@mta@config_n	0	1	2	3	4

11.3.2 Content of the SNMP framework tables after processing of the above example TLV38s

Based on the above assumptions and the contents of TLV38 specified in previous clauses, this clause illustrates the tables the MTA should create. The MTA ignores TLV38 number 1 (notification type = 1), therefore @mtaconfig_2 entries do not exist); the Security Name in TLV n=2 is ignored.

Table 29: snmpCommunityTable

Index	**[@mtaconfig]**
Name	"public"
SecurityName	@mtaconfig
ContextEngineID	<MTA ENGINEID>
ContextName	""
TransportTag	""
StorageType	volatile
Status	active

Table 30: snmpTargetAddrExtTable

Index	[@mtaconfig_0]	[@mtaconfig_1]	[@mtaconfig_2]	[@mtaconfig_3]	[@mtaconfig_4]	[@mtaconfig_5]
TMask	""	""	""	""	""	""
MMS	0	0	0	0	0	0

Table 31: usmUserTable

Index	[0x00][@mtaconfig]	[<local-EngineID>][mtaUser]	[<local-EngineID>][superUser]	[0x00/<Notif-recv-EngineID>][mtaUser]	[0x00/<Notif-recv-EngineID>][superUser]
SecurityName	@mtaconfig	MtaUser	superUser	mtaUser	superUser
CloneFrom	ZeroDotZero	ZeroDotZero	zeroDotZero	zeroDotZero	zeroDotZero
AuthProtocol	usmNoAuthProtocol	usmNoAuthProtocol	usmHMACMD5AuthProtocol	usmNoAuthProtocol	usmHMACMD5AuthProtocol
AuthKeyChange	""	""	""	""	""
OwnAuthKeyChange	""	""	""	""	""
PrivProtocol	usmNoPrivProtocol	usmNoPrivProtocol	usmDESPrivProtocoll	usmNoPrivProtocol	usmDESPrivProtocol
PrivKeyChange	""	""	""	""	""
OwnPrivKeyChange	""	""	""	""	""
Public	""	""	""	""	""
StorageType	Volatile	Volatile	Volatile	Volatile	Volatile
Status	active	active	active	active	active

Table 32: vacmContextTable

Index
VacmContextName

Table 33: vacmSecurityToGroupTable

Index	[1] [@mtaconfig]	[2] [@mtaconfig]	[3] [@mtaconfig]
GroupName	@mtaconfigV1	@mtaconfigV2	@mtaconfigUSM
SecurityToGroupStorageType	Volatile	Volatile	Volatile
SecurityToGroupStatus	active	active	active

Table 34: vacmAccessTable

Index	[@mtaconfigV1][][1][noAuthNoPriv]	[@mtaconfigV2][][2][noAuthNoPriv]	[@mtaconfigUSM][][3][noAuthNoPriv]
ContextMatch	exact	exact	exact
ReadViewName	""	""	""
WriteViewName	""	""	""
NotifyViewName	@mtaconfig	@mtaconfig	@mtaconfig
StorageType	Volatile	Volatile	Volatile
Status	active	active	active

Table 35: vacmViewTreeFamilyTable

Index	[@mtaconfig][org]
Mask	""
Type	included
StorageType	Volatile
Status	active

Table 36: snmpNotifyTable

Index	[@mtaconfig_inform]	[@mtaconfig_trap]
Tag	@mtaconfig_inform	@mtaconfig_trap
Type	inform	Trap
StorageType	Volatile	Volatile
RowStatus	active	active

Table 37: snmpTargetAddrTable

Index	[@mtaconfig_0]	[@mtaconfig_1]	[@mtaconfig_3]	[@mtaconfig_4]
TDomain	snmpUDPDomain	snmpUDPDomain	snmpUDPDomain	snmpUDPDomain
TAddress	"0A 00 05 09 00 82"	"0A 00 05 09 00 82"	"0A 00 04 09 DE A8"	"0A 00 08 09 00 82"
Timeout	1500	1500	1500	1500
RetryCount	3	1	3	3
TagList	@mtaconfig_trap	@mtaconfig_inform	@mtaconfig_trap	@mtaconfig_inform
Params	@mtaconfig_0	@mtaconfig_1	@mtaconfig_3	@mtaconfig_4
StorageType	Volatile	Volatile	Volatile	Volatile
RowStatus	active	active	active	active

Table 38: snmpTargetParamsTable

Index	[@mtaconfig_0]	[@mtaconfig_1]	[@mtaconfig_3]	[@mtaconfig_4]
MPModel	1	1	3	3
SecurityModel	2	2	3	3
SecurityName	'@mtaconfig	'@mtaconfig	'@mtaconfig	'@mtaconfig
SecurityLevel	noAuthNoPriv	noAuthNoPriv	noAuthNoPriv	NoAuthNoPriv
StorageType	Volatile	Volatile	Volatile	Volatile
RowStatus	active	active	active	active

Table 39: snmpNotifyFilterProfileTable

Index	[@mtaconfig_0]	[@mtaconfig_1]	[@mtaconfig_3]	[@mtaconfig_4]
Name	[@mtaconfig_0]	[@mtaconfig_1]	[@mtaconfig_3]	[@mtaconfig_4]
StorType	Volatile	Volatile	Volatile	Volatile
RowStatus	active	active	active	active

Table 40: snmpNotifyFilterTable

Index	[@mtaconfig_0] [org]	[@mtaconfig_1] [pktcMtaProvision-ingStatus]	[@mtaconfig_3] [PktcMtaMib]	[@mtaconfig_4] [pktcMtaProvision-ingStatus]
Mask	""	""	""	""
Type	included	included	included	included
StorageType	Volatile	Volatile	Volatile	Volatile
RowStatus	active	active	active	active

12 SNMPv2c Management Requirements

The management of an MTA device using SNMPv2c can be configured if required by an operator by setting the proper co-existence tables (using TLV11) in the MTA configuration file or via post-provisioning management:

- In the Basic and Hybrid Flows, the MTA must configure the tables described in clause 12.1 and 12.2 after MTA4 to provide SNMPv2c read/write access to the default management system (provisioning entity provided in DHCP option 122 sub-option 3).

- In the Secure Flow, the MTA must configure the tables in clause 12.2 if the configuration file contains TLV11 varbindings with the data of snmpCommunityTable. Additionally in order to restrict SNMP access to the MTA in the inbound direction the configuration file may also contain TLV11 varbindings for snmpTargetAddrTable and/or snmpTargetAddrExtTab.

Annex A provides an example template for operators to enable SNMPv2c management.

12.1 SNMPV2c Co-existence mode tables content created by MTA after MTA-4 for Hybrid and Basic Flows.

Table 41: snmpCommunityTable Content

snmpCommunityTable (RFC 3584 [32], SNMP-COMMUNITY-MIB)	**Read write Access**
Column Name (* = Part of Index)	Column Value
* snmpCommunityIndex	"@mtaprov"
snmpCommunityName	"private"
snmpCommunitySecurityName	"@mtaprov"
snmpCommunityContextEngineID	<The engineID of the MTA>
snmpCommunityContextName	Empty
snmpCommunityTransportTag	"@mtaprovTag"
snmpCommunityStorageType	Volatile(2)
snmpCommunityStatus	active(1)

Table 42: snmpTargetAddrTable Content

snmpTargetAddrTable (RFC 3413 [7], SNMP-TARGET-MIB)	**First Row**
Column Name (* = Part of Index)	Column Value
* snmpTargetAddrName	"@mtaprov"
snmpTargetAddrTDomain	snmpUDPDomain = snmpDomains.1
snmpTargetAddrTAddress (IP Address non-Authoritative SNMP entity)	OCTET STRING (6) Octets 1-4: <IP address of SNMP Entity derived from 122.3> Octets 5-6: any 2 byte port value
snmpTargetAddrTimeout	Ignore, <use default>
snmpTargetAddrRetryCount	ignore, <use default>
snmpTargetAddrTagList	"@mtaprovTag"
snmpTargetAddrParams	"@mtaprov"
snmpTargetAddrStorageType	volatile (2)
snmpTargetAddrRowStatus	active(1)

Table 43: snmpTargetAddrExtTable Content

snmpTargetAddrExtTable (RFC 3584 [32], SNMP-COMMUNITY-MIB)	**First Row**
Column Name (* = Part of Index)	Column Value
* snmpTargetAddrName	"@mtaprov"
snmpTargetAddrTMask	FFFFFFFF:0000
snmpTargetAddrMMS	0

12.2 SNMP Default entries for SNMPv2c Access

The following tables must be created by the MTA during the SNMP agent initialization to configure SNMPv2c access.

Table 44: vacmSecurityToGroupTable Default Entries

vacmSecurityToGroupTable (RFC 3415 [9], SNMP-VIEW-BASED-ACM-MIB)	First Row	Second Row	Third Row
Column Name (* = Part of Index)	Column Value	Column Value	Column Value
* vacmSecurityModel	SNMPv2c (2)	SNMPv2c (2)	SNMPv2c (2)
* vacmSecurityName	"@mtaprov"	"admin"	"operator"
vacmGroupName	"@mtaprov"	"admin"	"operator"
vacmSecurityToGroupStorageType	permanent(4)	permanent(4)	permanent(4)
vacmSecurityToGroupStatus	active(1)	active(1)	active(1)

Table 45: vacmAccessTable Default Entries

vacmAccessTable (RFC 3415 [9], SNMP-VIEW-BASED-ACM-MIB)	First Row	Second Row	Third Row
Column Name (* = Part of Index)	Column Value	Column Value	Column Value
* vacmGroupName	"@mtaprov"	"admin"	"operator"
* vacmAccessContextPrefix	Empty	Empty	Empty
* vacmAccessSecurityModel	SNMPv2c (2)	SNMPv2c (2)	SNMPv2c (2)
* vacmAccessSecurityLevel	noAuthNoPriv (1)	noAuthNoPriv (1)	noAuthNoPriv (1)
vacmAccessContextMatch	exact (1)	exact (1)	exact (1)
VacmAccessReadViewName	"@mtaconfig"	"@mtaconfig"	"@mtaconfig"
VacmAccessWriteViewName	"@mtaconfig"	"@mtaconfig"	Empty
vacmAccessNotifyViewName	"@mtaconfig"	Empty	Empty
vacmAccessStorageType	permanent(4)	permanent(4)	permanent(4)
vacmAccessStatus	active(1)	active(1)	active(1)

Table 46: vacmViewTreeFamilyTable Default Entry

vacmViewTreeFamilyTable (RFC 3415 [9], SNMP-VIEW-BASED-ACM-MIB)	First Row
Column Name (* = Part of Index)	Column Value
* vacmViewTreeFamilyViewName	@mtaconfig
VacmViewTreeFamilySubtree	1.3
vacmViewTreeFamilyMask	Empty <default from MIB>
vacmViewTreeFamilyType	included (1)
vacmViewTreeFamilyStorageType	volatile (2)
VacmViewTreeFamilyStatus	active (1)

NOTE: This entry is also created by default for the purpose of TLV-38 processing, It means only one default entry is needed in the MTA to define SNMPv2c management and TLV-38 configuration

Table 47: snmpTargetParamsTable Default Entry

snmpTargetParamsTable (RFC 3413 [7], SNMP-TARGET-MIB)	First Row
Column Name (* = Part of Index)	Column Value
* snmpTargetParamsName	"@mtaprov"
snmpTargetParamsMPModel	1
snmpTargetParamsSecurityModel	2
snmpTargetParamsSecurityName	"@mtaprov"
snmpTargetParamsSecurityLevel	noAuthNoPriv
snmpTargetParamsStorageType	permanent(4)
snmpTargetParamsRowStatus	active (1)

Table 48: snmpNotifyTable Default Entry

snmpNotifyTable (RFC 3413 [7], SNMP-NOTIFICATION-MIB)	First Row
Column Name (* = Part of Index)	Column Value
* snmpNotifyName	"@mtaprov"
snmpNotifyTag	"@mtaprovTag"
snmpNotifyType	inform (2)
snmpNotifyStorageType	permanent(4)
snmpNotifyRowStatus	active (1)

Table 49: snmpNotifyFilterProfileTable Default Entry

snmpNotifyFilterProfileTable (RFC 3413 [7], SNMP-NOTIFICATION-MIB)	First Row
Column Name (* = Part of Index)	Column Value
* snmpTargetParamsName	"@mtaprov"
snmpNotifyFilterProfileName	"@mtaprov"
snmpNotifyFilterProfileStorType	permanent(4)
snmpNotifyFilterProfileRowStatus	active(1)

Table 50: snmpNotifyFilterTable Default Entry

snmpNotifyFilterTable (RFC 3413 [7], SNMP-NOTIFICATION-MIB)	First Row	Second Row
Column Name (* = Part of Index)	Column Value	Column Value
* snmpNotifyFilterProfileName	"@mtaprov"	"@mtaprov"
* snmpNotifyFilterSubtree	pktcMtaNotification	snmpTraps
snmpNotifyFilterMask	Empty	Empty
snmpNotifyFilterType	included(1)	included(1)
snmpNotifyFilterStorageType	permanent(4)	permanent(4)
snmpNotifyFilterRowStatus	active(1)	active(1)

13 Service interruption impact reporting and other enhanced features support

13.1 eDOCSIS® Requirements support

The IPCablecom eMTA is considered a eSAFE device under eDOCSIS® and must adhere to relevant clauses of the eDOCSIS® Specification defined in [12] In addition to common requirements, the document has certain requirements that are contingent upon the definition in the corresponding eSAFE specification. This clause deals with those additional requirements that are deemed required by the IPCablecom document for implementation.

The requirements can be grouped as:

- Impact Analysis and Reporting requirements.
- eSAFE reboot directives.

13.1.1 Impact Analysis and Reporting Requirements:

As specified in [12], the eCM has the ability to report 'Service Interruption Impact' for each eSAFE device, if in fact the data service was interrupted at the time of the query. This clause deals with the impact levels and the reporting mechanism. It is to be noted that the IPCablecom eMTA is typically associated with multiple services (Voice, Fax) and multiple instances of each service (On each configured endpoint) and hence the eMTA must report the highest possible impact across services/endpoints.

13.1.1.1 Impact Analysis

A service on an endpoint is considered impacted when an endpoint is 'active' and the data service is interrupted. The 'active' condition is defined as the states offHook (3) and onHookPlusNCSActivity (2) as defined in pktcNcsEndPntHookState. (Refer to [12] for more information.)

13.1.1.2 Supported Impact Levels and Reporting

In IPCablecom, any interruption to an 'active' service (even potentially) must be considered as 'High Impact' and everything else considered 'Low Impact'.

Thus, impacts must be reported by the MTA as follows:

- High Impact - If any of the endpoints associated with an MTA are 'Active', then the impact must be reported as 'High Impact'.
- Low Impact - If all of the endpoints associated with an MTA that are capable of providing service are not 'active', then the impact must be reported as 'Low Impact'.

13.2 IPCablecom Extension MIB

IPCablecom extension MIB has been defined for all the new mibs that are part of IPCablecom 1.5. For more information see [i.1]. The extensions are in the areas of MTA MIB and Signalling MIB.

13.2.1 MTA MIB Extension

The IPCablecom MTA MIB Extension is defined in [16]. This provides the additional functionality for controlling new functionality like Multiple Grants Per Interval (MGPPI) on the endpoint.

13.2.2 Signalling MIB Extension

The IPCablecom Signalling MIB Extension is defined in [15]. This provides additional control and reporting functionality for endpoints in the areas of DTMF relay, Quarantine handling, Hookstate, etc.

13.3 Battery Backup MIBS

The E-MTA is a embedded device with the Cable Modem. Since telephony is a high availability service battery backup is very essential. In order to service and maintain the battery modules a set of MIBS have been defined in [17]. E-MTA devices that provide battery backup functionality must support the MIBS defined in [17].

13.4 Syslog MIBS

In order to maintain granularity for the syslog service a set of MIBS have been defined in [13] and [23]. These MIBS help the operator in troubleshooting the syslog service and also obtain a higher level of control over the syslog messages.

13.5 Foreign Potential Detection

Detecting foreign Potential is very important for providing telephony service. A MIB "pktcEnEndPntInfoTable" has been defined in [15] to report any such detection. E-MTA devices should implement this functionality.

Annex A (informative): SNMPv2c co-existence Configuration Example - Template for service providers

The operators can use the template defined in this clause to enable SNMPv2c management (default entries defined in clause 12.2 are reused in the example).

NOTE: Service providers are not restricted to use this template.

Table A.1: snmpCommunityTable Template for Basic and Hybrid Flows Configuration file

snmpCommunityTable (RFC 3584 [32], SNMP-COMMUNITY-MIB)	Read write Access	Read only Access
Column Name (* = Part of Index)	Column Value	Column Value
* snmpCommunityIndex	"admin"	"operator" or <any>
snmpCommunityName	<SNMP Community Name>	<SNMP Community Name>
snmpCommunitySecurityName	"admin"	"operator"
snmpCommunityContextEngineID	<The engineID of the MTA>	<The engineID of the MTA>
snmpCommunityContextName	Empty	Empty
snmpCommunityTransportTag	"adminTag"	"operatorTag"
snmpCommunityStorageType	volatile(2)	Volatile (2)
snmpCommunityStatus	createAndGo(4)	createAndGo(4)

Table A.2: snmpTargetAddrTable Template for Basic and Hybrid Flows Configuration file

snmpTargetAddrTable (RFC-3413 - SNMP-TARGET-MIB	First Row	Second Row
Column Name (* = Part of Index)	Column Value	Column Value
* snmpTargetAddrName	"admin"	"operator"
snmpTargetAddrTDomain	snmpUDPDomain = snmpDomains.1	snmpUDPDomain = snmpDomains.1
snmpTargetAddrTAddress (IP Address non-Authoritative SNMP entity	OCTET STRING (6) Octets 1-4: <SNMP Mgmt Station IPv4 Address> Octets 5-6: <0x0000>	OCTET STRING (6) Octets 1-4: <SNMP Mgmt Station IPv4 Address> Octets 5-6: <0x0000>
snmpTargetAddrTimeout	ignore, <use default>	Ignore, <use default>
snmpTargetAddrRetryCount	Ignore, <use default>	Ignore, <use default>
snmpTargetAddrTagList	"adminTag"	"operatorTag"
snmpTargetAddrParams	Empty	Empty
snmpTargetAddrStorageType	volatile (2)	volatile (2)
snmpTargetAddrRowStatus	createAndGo(4)	createAndGo(4)

Table A.3: snmpTargetAddrExtTable Template for Basic and Hybrid Flows Configuration file

snmpTargetAddrExtTable (RFC 3584 [32], SNMP-COMMUNITY-MIB)	First Row	Second Row
Column Name (* = Part of Index)	Column Value	Column Value
* snmpTargetAddrName	"admin"	"operator"
snmpTargetAddrTMask	OCTET STRING (6) Octets 1-4: <SNMP Mgmt Station Sub Net Mask> Octets 5-6: <0x0000>	OCTET STRING (6) Octets 1-4: <SNMP Mgmt Station Sub Net Mask> Octets 5-6: <0x0000>
snmpTargetAddrMMS	0	0

Annex B (informative): Bibliography

- IETF RFC 3442: "The Classless Static Route Option for Dynamic Host Configuration Protocol (DHCP) version 4", December 2002.
- Protocol Numbers and Assignment Services, Internet Assigned Numbers Authority, http://www.iana.org/numbers.html
- IETF RFC 1350/STD0033: "The TFTP Protocol (Revision 2), MIT", July 1992.
- IETF RFC 1034/STD0013: "Domain Names - Concepts and Facilities", November 1987.
- IETF RFC 1035: "Domain Names - Implementation and Specifications", November 1987.
- IETF RFC 3417/STD0062: "Transport Mappings for the Simple Network Management Protocol (SNMP)", December 2002.
- IETF RFC 2579: "Textual Conventions for SMIv2", April 1999.
- IETF RFC 2821: "Simple Mail Transfer Protocol", April 2001.
- IETF RFC 1123/STD0003: Braden, R.,"Requirements for Internet Hosts -- Application and Support", October 1989.
- IETF RFC 2349: "TFTP Timeout Interval and Transfer Size Options", May 1998.
- IETF RFC 1945 and IETF RFC 2068: "HTTP 1.0 and 1.1", May 1996.
- IETF RFC 2782: "A DNS RR for specifying the location of services (DNS SRV)", February 2000.
- IETF RFC 1591: "Domain Name System Structure and Delegation", March 1994.

History

Document history		
V1.1.1	October 2011	Publication