OPSAWG
Internet-Draft
Updates: 5416 (if approved)
Intended status: Standards Track
Expires: August 18, 2014

Y. Chen
D. Liu
H. Deng
China Mobile
Lei. Zhu
Huawei

February 14, 2014

CAPWAP Extension for 802.11n and Power/channel Autoconfiguration
draft-ietf-opsawg-capwap-extension-02

Abstract

CAPWAP binding for 802.11 is specified by RFC5416 and it was based on IEEE 802-11.2007 standard. After RFC5416 was published in 2009, there were several new amendments of 802.11 have been published. 802.11n is one of those amendments and it has been widely used in real deployment. This document extends the CAPWAP binding for 802.11 to support 802.11n and also defines a power and channel auto configuration extension.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on August 18, 2014.

Copyright Notice

Copyright (c) 2014 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust’s Legal Provisions Relating to IETF Documents (http://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents

carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction 2
2. Conventions used in this document 3
3. Abbreviations 3
4. CAPWAP 802.11n Support 3
4.1. CAPWAP Extension for 802.11n Support 4
4.1.1. 802.11n Radio Capability Information 4
4.1.2. 802.11n Radio Configuration Message Element 4
4.1.3. 802.11n Station Information 6
5. Power and Channel Autoconfiguration 7
5.1. Channel Autoconfiguration When WTP Power On 7
5.2. Power Configuration When WTP Power On 8
5.3. Channel/Power Auto Adjusment 8
5.3.1. Scan Parameter Message Element 9
5.3.2. Channel Bind Message Element 10
5.3.3. Channel Scan Report 11
5.3.4. Neighbor WTP Report 13
6. Security Considerations 13
7. IANA Considerations 13
8. Contributors 14
9. Acknowledgements 14
10. Normative References 14
Authors' Addresses 15

1. Introduction

IEEE 802.11n standard was published in 2009 and it is an amendment to the IEEE 802.11-2007 standard. The maximum data rate increases to 600Mbps. In the physical layer, 802.11n use OFDM and MIMO to achieve the high throughput. 802.11n also use multiple antennas to form antenna array which can be dynamically adjusted to improve the signal strength and extend the coverage.

There are several capabilities of 802.11n need to be supported by CAPWAP control message, such as radio capability, radio configuration and station information etc. This document specifies the 802.11n and power/channel auto-configuration extensions for CAPWAP.

For the AC/WTP that does not support the extensions defined by this document, it can simply ignore the extensions and will not cause any incompatible issue.

2. Conventions used in this document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL","SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

3. Abbreviations

AC: Access Controller
A-MSDU:Aggregate MAC Service Data Unit
A-MPDU:Aggregate MAC Protocol Data Unit
MIMO: Multi-input Multi-output
MSDU: MAC Service Data Unit
MPDU: MAC Protocol Data Unit
MCS: Maximum Modulation and Coding Scheme
OFDM: Orthogonal Frequency-Division Multiplexing
WTP: Wireless Termination Points.

4. CAPWAP 802.11n Support

[IEEE-802.11.2009] standard was published in 2009 and it is an amendment of the IEEE 802.11-2007 standard to improve throughput. The maximum data rate increases to 600Mbps. In the physical layer, 802.11n use OFDM and MIMO to achieve high throughput. 802.11n use multiple antennas to form antenna array which can be dynamically adjusted to improve the signal strength and extend the coverage.

802.11n support three modes of channel usage: 20MHz mode, 40MHz mode and mixed mode. 802.11n has a new feature called channel binding. It can bind two adjacent 20MHz channel to one 40MHz channel to improve the throughput.If using 40MHz channel configuration there will be only one non-overlapping channel in 2.4GHz. In the large scale deployment scenario, operator need to use 20MHz channel configuration in 2.4GHz to allow more non-overlapping channels.

In MAC layer, a new feature of 802.11n is Short Guard Interval(GI). 802.11a/g uses 800ns guard interval between the adjacent information symbols. In 802.11n, the GI can be configured to 400nm under good wireless condition.

Another feature in 802.11 MAC layer is Block ACK. 802.11n can use one ACK frame to acknowledge several MPDU receiving event.

CAPWAP needs to be extended to support the above new 802.11n features. For example, CAPWAP should allow the access controller to know the supported 802.11n features of WTP and the access controller should be able to configure the different channel binding modes for WTP.

4.1. CAPWAP Extension for 802.11n Support

There are three 802.11n features need to be supported by CAPWAP 802.11 binding: 802.11n radio capability, 802.11n radio configuration and station information. This section defines the extension of current CAPWAP 802.11 binding to support 802.11n features.

4.1.1. 802.11n Radio Capability Information

[RFC5416] defines IEEE 802.11 binding for CAPWAP protocol. It defines IEEE 802.11 Information Element (Type 1029) which is used to communicate any IE defined in IEEE 802.11 protocol. The detail definition of IEEE 802.11 Information Element is in section 6.6 of [RFC5416]. The IEEE 802.11 HT information element is defined in section 8.4.2.58 of [IEEE-802.11.2012]. It contains the 802.11n radio capability information. This document specifies use of the IEEE 802.11 Information Element (Type 1029) transporting the IEEE 802.11 HT information element to carry the 802.11n radio capability information. 802.11n radio capability information MAY be included in the CAPWAP Configuration Status Request/Response messages.

4.1.2. 802.11n Radio Configuration Message Element

The 802.11n Radio Configuration Information Element message element is used by the AC to configure a Radio on the WTP and by the WTP to deliver its radio configuration to the AC. The 802.11n Radio Configuration Information Element is defined in figure 1. 802.11n Radio Configuration Message Element MAY be included in the CAPWAP Configuration Update Request/Response message.

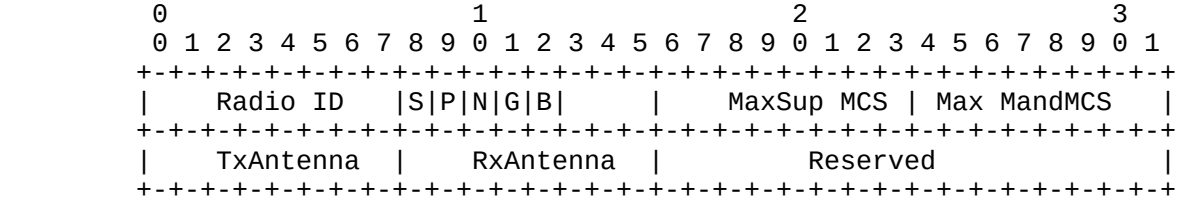


Figure 1: 802.11n Radio Configuration Message Element

Type: TBD for 802.11n Radio Configuration Message Element.

Length: 16.

Radio ID: An 8-bit value representing the radio, whose value is between one (1) and 31.

S bit: A-MSDU Cfg: Enable/disable Aggregate MAC Service Data Unit (A-MSDU). Set to 0 if disabled. Set to 1 if enabled.

P bit: A-MPDU Cfg: Enable/disable Aggregate MAC Protocol Data Unit (A-MPDU). Set to 0 if disabled. Set to 1 if enabled.

N bit: 11n Only Cfg: Whether to allow only 11n user access. Set to 0 if allow non-802.11n user access. Set to 1 if do not allow non-802.11n user access.

G bit: Short GI Cfg: Set to 0 if disabled. Set to 1 if enabled.

B bit: Bandwidth Cfg: Bandwidth binding mode. Set to 0 if 40MHz binding mode. Set to 1 if 20MHz binding mode.

MaxSup MCS: Maximum Modulation and Coding Scheme (MCS) index. It indicates the maximum MCS index that the WTP or the STA can support.

Max Mandatory MCS: Maximum Mandatory Modulation and Coding Scheme (MCS) index. Mandatory rates must be supported by the WTP and the STA that want to associate with the WTP.

TxAntenna: Transmitting antenna configuration. Each TxAntenna bit represent a certain number of antennas. Set to 1 if enabled, set to 0 if disabled.

RxAntenna: Receiving antenna configuration. Each RxAntenna bit represent a certain number of antennas. Set to 1 if enabled, set to 0 if disabled.

The detail definition of TxAntenna/RxAntenna is as follows:

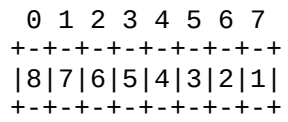


Figure 2: Definition of TxAntenna/RxAntenna

Each bit when enabled will represent the number of antennas correspondent to that bit. For example, when the first bit is enabled,it represents 8 antennas.

4.1.3. 802.11n Station Information

The 802.11n Station Information message element is used to deliver IEEE 802.11n station policy from the AC to the WTP. The definition of the 802.11n Station Information message element is in figure 3. 802.11n Station Information MAY be included in the CAPWAP Station Configuration Request message.

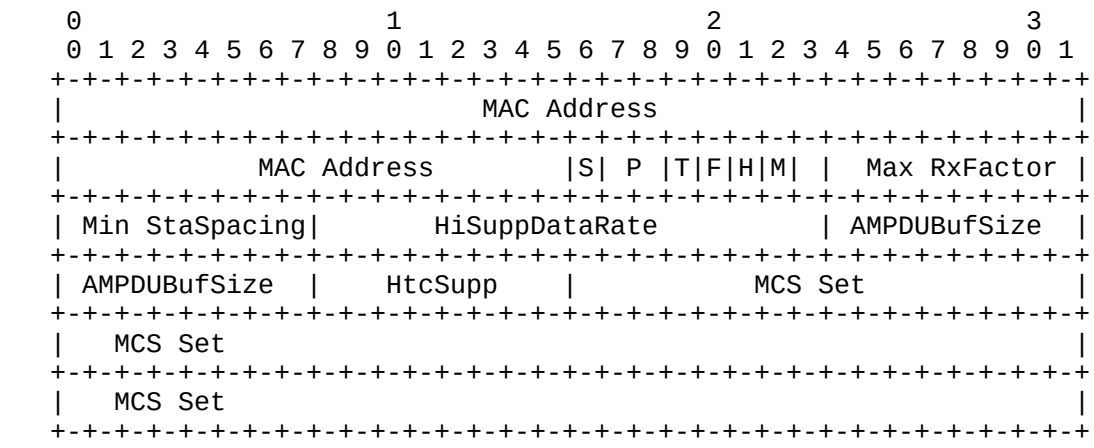


Figure 3: 802.11n Station Information

Type: TBD for 802.11 Station Information.

Length: 24.

S bit: SupChanl width: Supporting bandwidth mode. 0x00: 20MHz bandwidth mode. 0x01: 40MHz bandwidth binding mode.

P flag: Power Save: 0x00: Static power saving mode. 0x01: Dynamic power saving mode. 0x03: Do not support power saving mode.

T bit: ShortGi20: Whether support short GI in 20MHz bandwidth mode. 0x00: Do not support short GI. ox01: Support short GI.

F bit: ShortGi40: Whether support short GI in 40MHz bandwidth mode. 0x00: Do not support short GI. ox01: Support short GI.

H bit: HtDelyBlkack: Whether block Ack support delay mode. 0x00: Do not support delay mode. 0x01: Support delay mode.

M bit: Max Amsdu: The maximal AMSDU length. 0x00: 3839 bytes. 0x01: 7935 bytes.

Max RxFactor: The maximal receiving AMPDU factor.

Min StaSpacing: Minimum MPDU Start Spacing.

HiSuppDataRate: Maximal transmission speed (Mbps).

AMPDUBufSize: AMPDU buffer size.

HtcSupp: Whether the packet have HT header.

MCS Set: The MCS bitmap that the station supports.

5. Power and Channel Autoconfiguration

Power and channel autoconfiguration could avoid potential radio interference and improve the WLAN performance. In general, the auto-configuration of radio power and channel could occur at two stages: when the WTP power on or during the WTP running time.

5.1. Channel Autoconfiguration When WTP Power On

When the WTP is power-on, it is of necessity to configure a proper channel to the WTP in order to achieve best status of radio links. IEEE 802.11 Direct Sequence Control elements or IEEE 802.11 OFDM Control element defined in RFC5416 SHOULD be carried in the Configure Status Response message to offer WTP a channel at this stage. If those information element is zero, the WTP will need to determine its channel by itself, otherwise the WTP SHOULD be configured according to the provided information element.

When the WTP determines its own channel configuration, it should first scan the channel information, then determine which channel it will work on and form a channel quality scan report. The channel quality report will be sent to the AC using WTP Event Request message by the WTP.

AC will determine whether to change the channel configuration based on the received channel quality report. The AC can use IEEE 802.11 Direct Sequence Control or IEEE 802.11 OFDM Control information element carried by the configure Update Request message to configure a new channel for the WTP.

5.2. Power Configuration When WTP Power On

IEEE 802.11 Tx Power information element is used by the AC to control the transmission power of the WTP. The 802.11 Tx Power information element is carried in the Configure Status Response message or in the Configure Update Request message.

5.3. Channel/Power Auto Adjusment

The Channel Scan Procedure is illustrated by the figure 4.

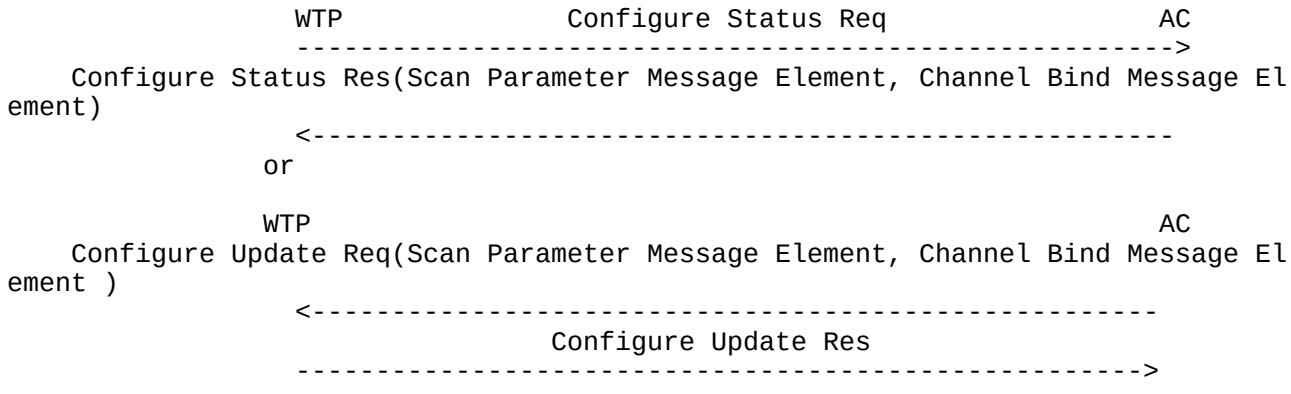


Figure 4: Channel Scan Procedure

WTP has two working modes, the first one is normal working mode. In this mode, the WTP can scan the channel while providing the service to STA. Whether WTP will provide scanning service is determined by the Max Cycles value of Channel Bind Message Element. If this value equls to zero, the WTP will not perform scanning. If this value equls to 255, the WTP will scan the channel continuously until getting notification from AC. Otherwise, the WTP will perform scanning with the number that specified the value of Max Cycles. The second working mode is scan only mode. The WTP will not provide service to STA in this case. In this mode, WTP will scan the channel continuously.

When the WTP work in the scan only mode, there is no difference between the working channel and scan channel. Every channel’s scan duration will be OffChannelScnTime and the PrimeChlSrvTime and OnChannelScanTime is set to 0.

There are two scan types which is determined by the Scan Type value. The first type is passive scan. The WTP will listen the channel passively in this case. The other type is active scan. The WTP will

send probe for the scan. There are three parameters that will determine the working mode of scan: PrimeChlSrvTime, On Channel ScanTime, Off Channel ScanTime. The WTP will provide service for the period of "PrimeChlSrvTime" time then start channel scan for the period of "On Channel ScanTime" time; then continue to provide service for the period of "PrimeChlSrvTime" time; then leave the current working channel and scan next channel for the period of "Off Channel ScanTime" time; then provide service on the next channel for the period of "PrimeChlSrvTime"..until finishing the scan procedure.

5.3.1. Scan Parameter Message Element

The definition of the Scan Para Message Element is as follows:

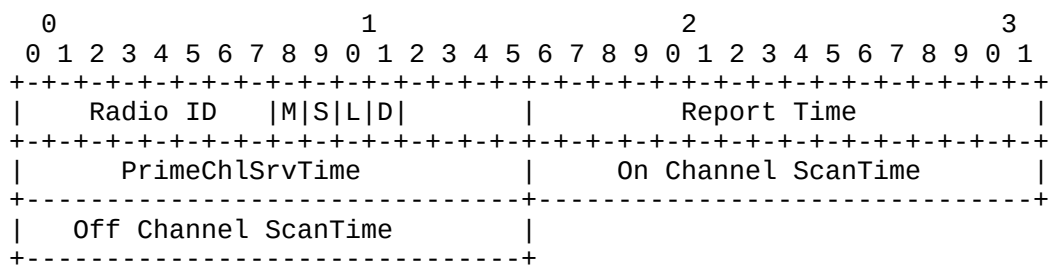


Figure 5: Scan Parameter Message Element

Type: TBD for Scan Parameter Message Element.

Length: 10.

Radio ID: An 8-bit value representing the radio, whose value is between one (1) and 31.

M bit: AP oper mode: the work mode of the WTP. 0x01:normal mode. 0x02: monitor only mode, no service is provided in this mode.ss

S bit: Scan Type: 0x01: active scan; 0x02: passive scan.

L bit: L=1: Open Load Balance Scan. D bit: D=1: Open Rogue WTP detection scan.

Report Time: Channel quality report time (unit: second).

PrimeChlSrvTime: Service time (unit: millisecond) on the working scan channel. This segment is invalid(set to 0) when WTP oper mode is set to 2. The maximum value of this segment is 10000, the minimum value of this segment is 5000, the default value is 5000.

On Channel ScanTime: The scan time (unit: millisecond) of the working channel. When the WTP oper mode is set to 2, this segment is invalid(set to 0). The maximum value of this segment is 120, the minimum value of this segment is 60, the default value is 60.

5.3.2. Channel Bind Message Element

The definition of the Channel Bind Message ELement is as follows:

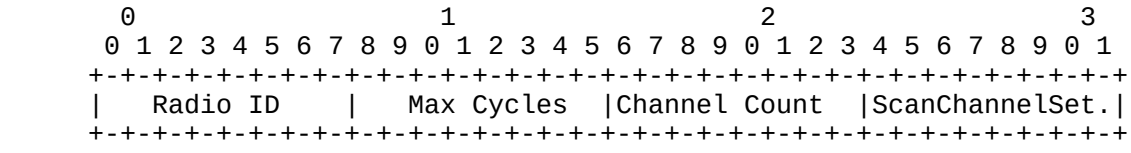


Figure 6: Channel Bind Message Element

Type: TBD for Channel Bind Message Element.

Length: 4.

Radio ID: An 8-bit value representing the radio, whose value is between one (1) and 31.

Flag: bitmap, reserved.

Max Cycles: Scan repeat times. 255 means continuous scan.

Channel Count: The number of channel will be scanned.

Scan Channel Set: The channel information. The format is as follows:

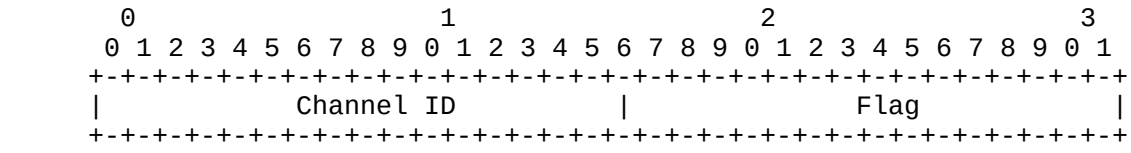


Figure 7: Channel Information Format

Channel ID: the channel ID of the channel which will be scanned.

Flag: Bitmap, reserved for future use.

5.3.3. Channel Scan Report

There are two types of scan report: Channel Scan Reprot and Neighbor STA Reprot. Channel Scan Report is used to channel autoconfiguration while Neighbor WTP Report is used to power autoconfiguration. The WTP send the scan report to the AC through WTP Event Request message. The information element that used to carry the scan report is Channel Scan Report Message Element and Neighbor WTP Report Message Element.

The definition of the Channel Scan Report Message Element is in figure 8.

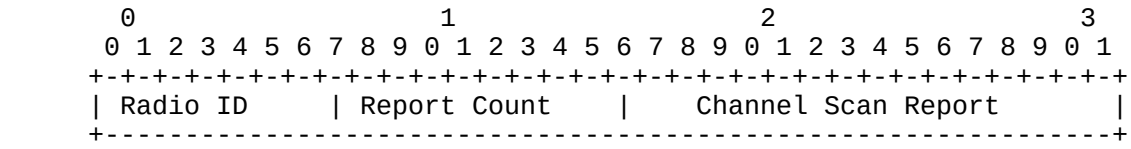


Figure 8: Channel Scan Report Message Element

Type: TBD for Channel Scan Report Message Element.

Length: >=29.

Radio ID: An 8-bit value representing the radio, whose value is between one (1) and 31.

Report Count: The channel number will be reported.

Channel Scan Report: The definition of the Channel Scan Report is in figure 9. It complies with the IEEE 802.11 Beacon report that defined in section 8.4.2.24.7 of [IEEE-802.11.2012].

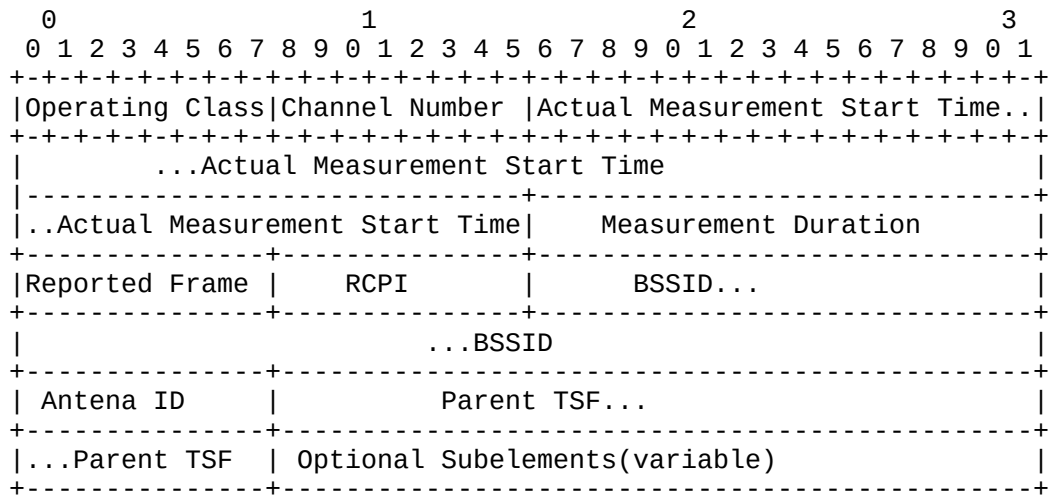


Figure 9: Channel Scan Report

Operating Class: Indicates the channel set for which the measurement request applies. The definition of this field complies with the definition in section 8.4.2.24.7 of [IEEE-802.11.2012].

Channel Number: Indicates the channel number for which the measurement report applies. The definition of this field complies with the definition in section 8.4.2.24.7 of [IEEE-802.11.2012].

Actual Measurement Start Time: Is set to the value of the measuring STA’s TSF timer at the time the measurement started.

Measurement Duration: Is set to the duration over which the Beacon Report was measured. The definition of this field complies with the definition in section 8.4.2.24.7 of [IEEE-802.11.2012].

Reported Frame Information: This field contains two subfields as defined in [IEEE-802.11.2012].

RCPI: Indicates the received channel power of the Beacon, Measurement Pilot, or Probe Response frame.

RSNI:Indicates the received signal to noise indication for the Beacon, Measurement Pilot, or Probe Response frame.

BSSID: This field contains the BSSID from the Beacon,Measurement Pilot, or Probe Response frame being reported.

Antenna ID: This field contains the identifying number for the antennas used for this measurement.

Parent TSF: This field contains the lower 4 octets of the measuring STA's TSF timer value at the start of reception of the first octet of the timestamp field of the reported Beacon, Measurement Pilot, or Probe Response frame at the time the Beacon frame being reported was received.

Optional Subelements: This field contains zero or more subelements.

5.3.4. Neighbor WTP Report

The neighbor WTP report message element is composed of the IEEE 802.11 Information Element that defined in section 6.6 of [RFC5416] and IEEE 802.11 Neighbor Report Element that defined in section 8.4.2.39 of [IEEE-802.11.2012]. The Neighbor Report Element is carried by the IEEE 802.11 Information Element to form the neighbor WTP report message element.

6. Security Considerations

This document is based on RFC5415/RFC5416 and it doesn't increase any security risk. The security considerations of this document aligns with RFC5415/5416.

7. IANA Considerations

The extension defined in this document need to extend CAPWAP IEEE 802.11 binding message element which is defined in section 6 of [RFC5416]. The following IEEE 802.11 specific message element type need to be defined by IANA.

802.11n Radio Configuration Message Element type value described in section 4.1.2.

802.11n Station Message Element type value described in section 4.1.3.

Scan Parameter Message Element type value described in section 5.3.1.

Channel Bind Message Element type value described in section 5.3.2.

Channel Scan Report Message Element type value described in section 5.3.3.

8. Contributors

This draft is a joint effort from the following contributors:

Gang Chen: China Mobile chengang@chinamobile.com

Naibao Zhou: China Mobile zhounaibao@chinamobile.com

Chunju Shao: China Mobile shaochunju@chinamobile.com

Hao Wang: Huawei3Come hwang@h3c.com

Yakun Liu: AUTELAN liuyk@autelan.com

Xiaobo Zhang: GBCOM

Xiaolong Yu: Ruijie Networks

Song zhao: ZhiDaKang Communications

Yiwen Mo: ZhongTai Networks

9. Acknowledgements

The authors would like to thanks Ronald Bonica,Romascanu Dan, Benoit Claise, Melinda Shore and Margaret Wasserman for their useful suggestions. The authors also thanks Dorothy Stanley and Tom Taylor for their review and useful comments.

10. Normative References

[IEEE-802.11.2009]
"IEEE Standard for Information technology - Telecommunications and information exchange between systems Local and metropolitan area networks - Specific requirements Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications", 2009.

[IEEE-802.11.2012]
"IEEE Standard for Information technology - Telecommunications and information exchange between systems Local and metropolitan area networks - Specific requirements Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications", March 2012.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC4564] Govindan, S., Cheng, H., Yao, ZH., Zhou, WH., and L. Yang, "Objectives for Control and Provisioning of Wireless Access Points (CAPWAP)", RFC 4564, July 2006.

[RFC5415] Calhoun, P., Montemurro, M., and D. Stanley, "Control And Provisioning of Wireless Access Points (CAPWAP) Protocol Specification", RFC 5415, March 2009.

[RFC5416] Calhoun, P., Montemurro, M., and D. Stanley, "Control and Provisioning of Wireless Access Points (CAPWAP) Protocol Binding for IEEE 802.11", RFC 5416, March 2009.

Authors' Addresses

Yifan Chen
China Mobile
No.32 Xuanwumen West Street
Beijing 100053
China

Email: chenyifan@chinamobile.com

Dapeng Liu
China Mobile
No.32 Xuanwumen West Street
Beijing 100053
China

Email: liudapeng@chinamobile.com

Hui Deng
China Mobile
No.32 Xuanwumen West Street
Beijing 100053
China

Email: denghui@chinamobile.com

Lei Zhu
Huawei
No. 156, Shi-Chuang-Ke-Ji-Shi-Fan-Yuan Beiqing Road, Haidian District
Beijing 100095
China

Email: lei.zhu@huawei.com

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: June 21, 2015

C. Shao
H. Deng
China Mobile
R. Pazhyannur
Cisco Systems
F. Bari
AT&T
R. Zhang
China Telecom
S. Matsushima
SoftBank Telecom

December 18, 2014

IEEE 802.11 MAC Profile for CAPWAP
draft-ietf-opsawg-capwap-hybridmac-08

Abstract

The CAPWAP protocol binding for IEEE 802.11 defines two MAC (Medium Access Control) modes for IEEE 802.11 WTP (Wireless Transmission Point): Split and Local MAC. In the Split MAC mode, the partitioning of encryption/decryption functions are not clearly defined. In the Split MAC mode description, IEEE 802.11 encryption is specified as located in either the AC (Access Controller) or the WTP, with no clear way for the AC to inform the WTP of where the encryption functionality should be located. This leads to interoperability issues, especially when the AC and WTP come from different vendors. To prevent interoperability issues, this specification defines an IEEE 802.11 MAC profile message element in which each profile specifies an unambiguous division of encryption functionality between the WTP and AC.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on June 21, 2015.

Copyright Notice

Copyright (c) 2014 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (http://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction 2
2. IEEE MAC Profile Descriptions 4
2.1. Split MAC with WTP encryption 4
2.2. Split MAC with AC encryption 5
2.3. IEEE 802.11 MAC Profile Frame Exchange 6
3. MAC Profile Message Element Definitions 7
3.1. IEEE 802.11 Supported MAC Profiles 7
3.2. IEEE 802.11 MAC Profile 8
4. Security Considerations 8
5. IANA Considerations 8
6. Contributors 9
7. Acknowledgments 9
8. Normative References 9
Authors' Addresses 9

1. Introduction

The CAPWAP protocol supports two MAC modes of operation: Split and Local MAC, as described in [RFC5415], [RFC5416]. However, there are MAC functions that have not been clearly defined. For example IEEE 802.11 encryption is specified as located in either in the AC or the WTP with no clear way to negotiate where it should be located. Because different vendors have different definitions of the MAC mode, many MAC layer functions are mapped differently to either the WTP or the AC by different vendors. Therefore, depending upon the vendor, the operators in their deployments have to perform different configurations based on implementation of the two modes by their vendor. If there is no clear specification, then operators will

experience interoperability issues with WTPs and ACs from different vendors.

Figure 1 from [RFC5416], illustrates how some functions are processed in different places in the Local MAC and Split MAC mode. Specifically, note that in the Split MAC mode the IEEE 802.11 encryption/decryption is specified as WTP/AC implying that it could be at either location. This is not an issue with Local MAC because encryption is always at the WTP.

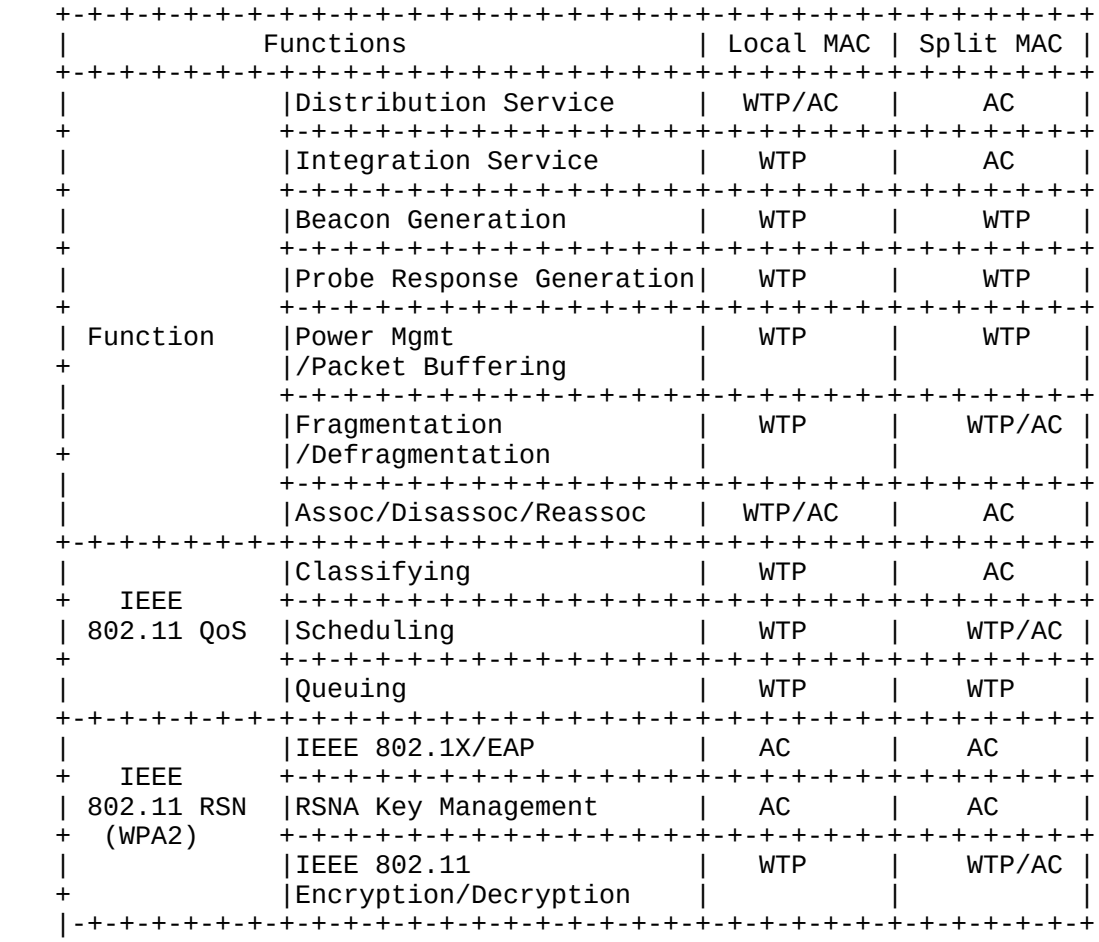


Figure 1: Functions in Local MAC and Split MAC

To solve this problem, this specification introduces IEEE 802.11 MAC profile. The MAC profile unambiguously specifies where the various MAC functionality should be located.

2. IEEE MAC Profile Descriptions

A IEEE MAC Profile refers to a description of how the MAC functionality is split between the WTP and AC shown in Figure 1.

2.1. Split MAC with WTP encryption

The functional split for the Split MAC with WTP encryption is provided in Figure 2. This profile is similar to the Split MAC description in [RFC5416], except that IEEE 802.11 encryption/ decryption is at the WTP. Note that fragmentation is always done at the same entity as the encryption. Consequently, in this profile fragmentation/defragmentation is also done only at the WTP. Note that scheduling functionality is denoted as WTP/AC. As explained in [RFC5416], this means that the admission control component of IEEE 802.11 resides on the AC, the real-time scheduling and queuing functions are on the WTP.

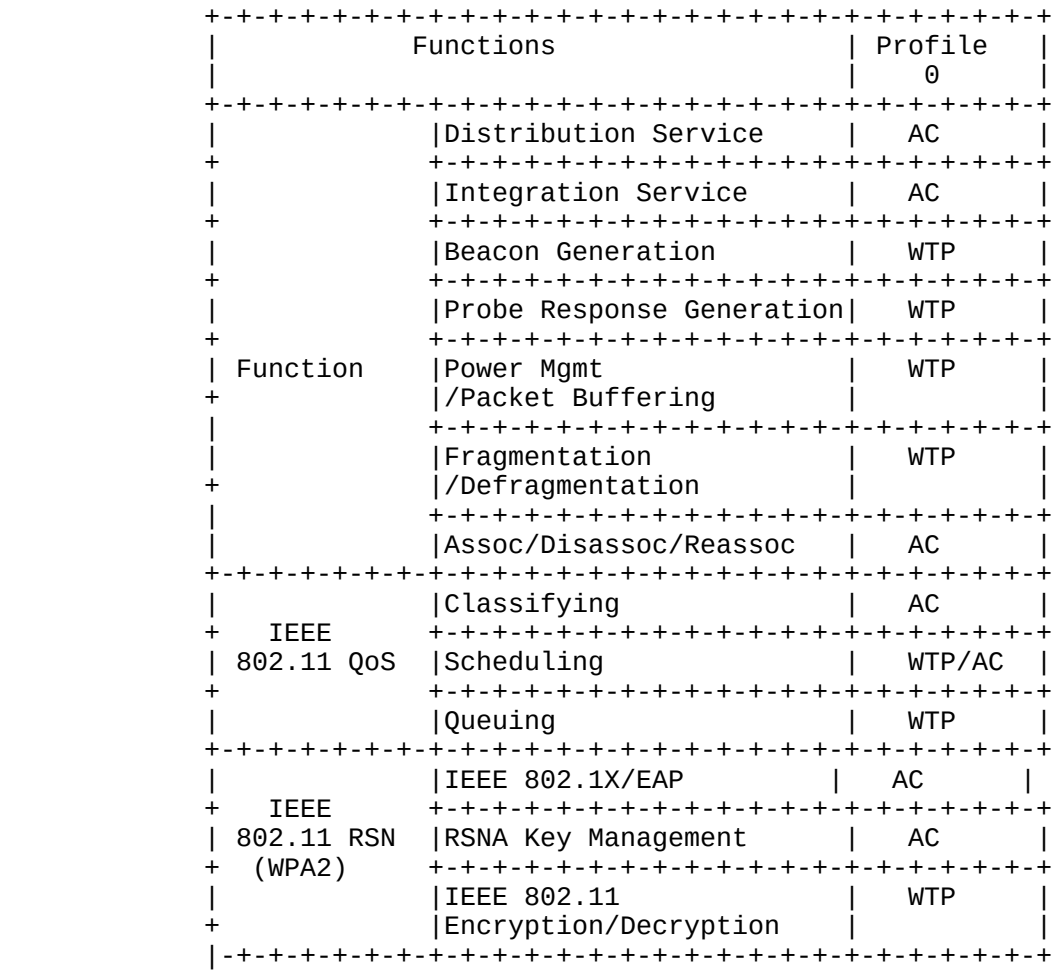


Figure 2: Functions in Split MAC with WTP Encryption

2.2. Split MAC with AC encryption

The functional split for the Split MAC with AC encryption is provided in Figure 3. This profile is similar to the Split MAC in [RFC5416] except that IEEE 802.11 encryption/decryption is at the AC. Since fragmentation is always done at the same entity as the encryption, in this profile, AC does fragmentation/defragmentation.

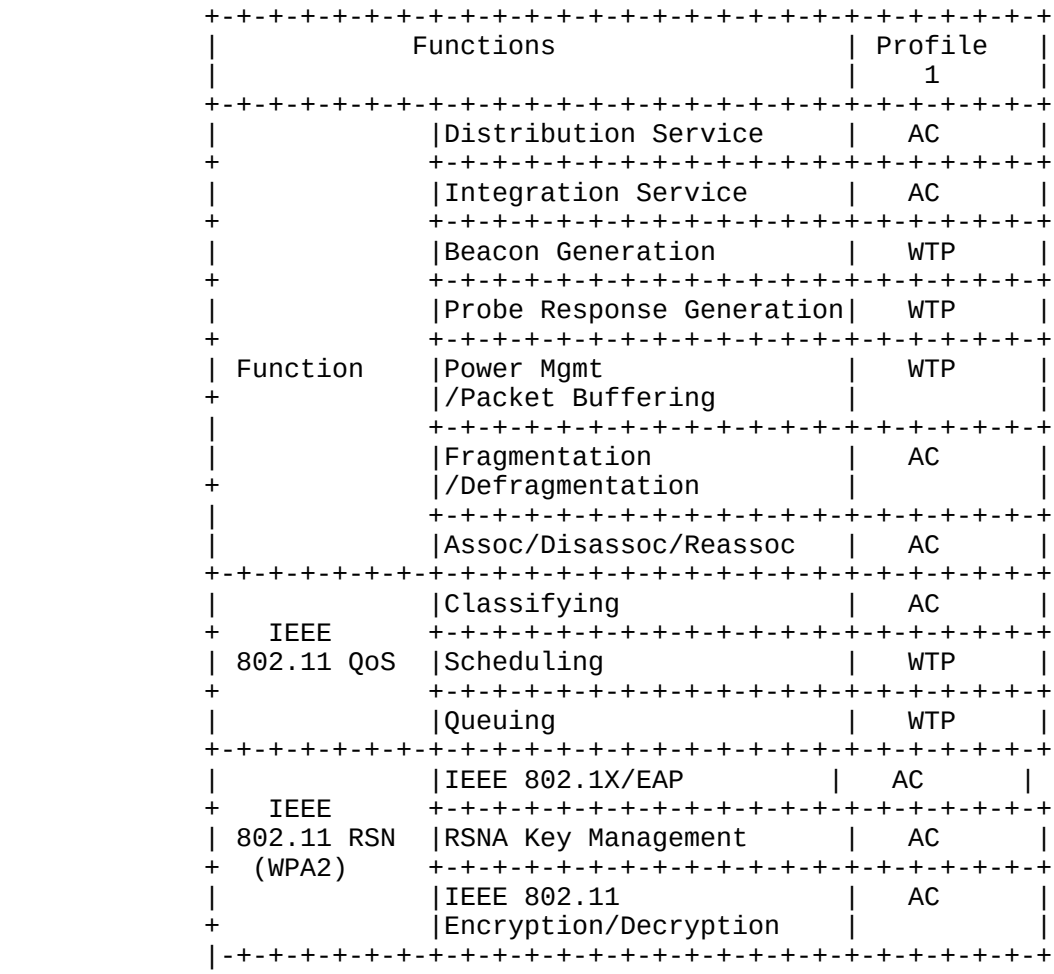


Figure 3: Functions in Split MAC with AC encryption

2.3. IEEE 802.11 MAC Profile Frame Exchange

An example of message exchange using the IEEE 802.11 MAC Profile message element is shown in Figure 4. The WTP informs the AC of the various MAC profiles it supports. This happens either in a Discovery Request message or the Join Request message. The AC determines the appropriate profile and configures the WTP with the profile while configuring the WLAN.

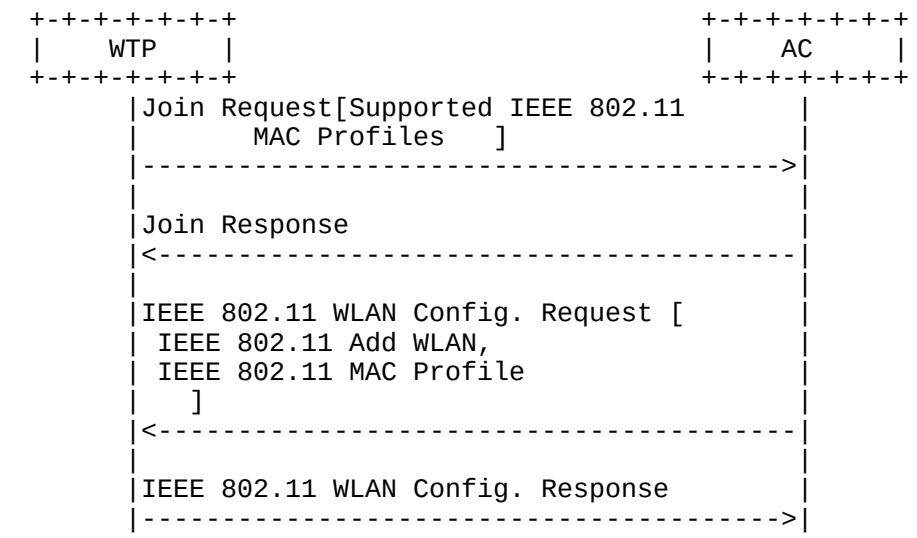


Figure 4: Message Exchange For Negotiating MAC Profile

3. MAC Profile Message Element Definitions

3.1. IEEE 802.11 Supported MAC Profiles

The IEEE 802.11 Supported MAC Profile message element allows the WTP to communicate the profiles it supports. The Discovery Request message, Primary Discovery Request message, and Join Request message may include one such message element.

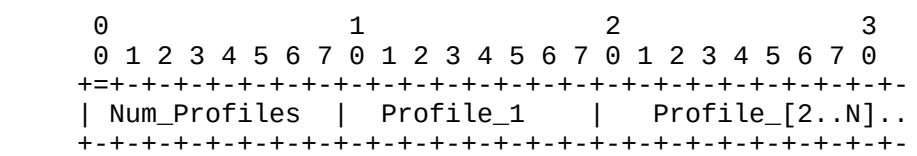


Figure 5: IEEE 802.11 Supported MAC Profiles

- Type: TBD for IEEE 802.11 Supported MAC Profiles
- Num_Profiles >=1: This refers to number of profiles present in this message element. There must be at least one profile.
- Profile: Each profile is identified by a value specified in Section 3.2.

3.2. IEEE 802.11 MAC Profile

The IEEE 802.11 MAC Profile message element allows the AC to select a profile. This message element may be provided along with the IEEE 802.11 ADD WLAN message element while configuring a WLAN on the WTP.

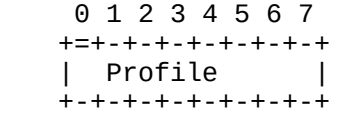


Figure 6: IEEE 802.11 MAC Profile

- Type: TBD for IEEE 802.11 MAC Profile
- Profile: The profile is identified by a value as given below
  - 0: This refers to the Split MAC Profile with WTP encryption
  - 1: This refers to the Split MAC Profile with AC encryption

4. Security Considerations

This document does not introduce any new security risks compared to [RFC5416]. The negotiation messages between the WTP and AC have origin authentication and data integrity. As a result an attacker cannot interfere with the messages to force a less secure mode choice. The security considerations described in [RFC5416] apply here as well.

5. IANA Considerations

This document requires the following IANA actions:

- This specification defines two new message elements, IEEE 802.11 Supported MAC Profiles (described in Section 3.1) and IEEE 802.11 MAC Profile (described in Section 3.2). These elements needs to be registered in the existing CAPWAP Message Element Type registry, defined in [RFC5415]. The values for these elements needs to be between 1024 and 2047 (see Section 15.7 in [RFC5415]).

CAPWAP Protocol Message Element	Type Value
IEEE 802.11 Supported MAC Profiles	TBD1
IEEE 802.11 MAC Profile	TBD2

- The IEEE 802.11 Supported MAC Profiles message element and IEEE 802.11 MAC Profile message element include a Profile Field (as defined in Section 3.2). The Profile field in the IEEE 802.11 Supported MAC Profiles denotes the MAC profiles supported by the WTP. The profile field in the IEEE MAC profile denotes MAC

profile assigned to the WTP. The namespace for the field is 8 bits (0-255). This specification defines two values, zero (0) and one (1) as described below. The remaining values (2-255) are controlled and maintained by IANA and require an Expert Review. IANA needs to create a new sub-registry called IEEE 802.11 Split MAC Profile and add the new sub-registry to the existing registry "Control And Provisioning of Wireless Access Points (CAPWAP) Parameters". The registry format is given below.

Profile	Type Value	Reference
Split MAC with WTP encryption	0	
Split MAC with AC encryption	1	

6. Contributors

Yifan Chen chenyifan@chinamobile.com

Naibao Zhou zhounaibao@chinamobile.com

7. Acknowledgments

The authors are grateful for extremely valuable suggestions from Dorothy Stanley in developing this specification.

Guidance from management team: Melinda Shore, Scott Bradner, Chris Liljenstolpe, Benoit Claise, Joel Jaeggli, Dan Romascanu are highly appreciated.

8. Normative References

[RFC5415] Calhoun, P., Montemurro, M., and D. Stanley, "Control And Provisioning of Wireless Access Points (CAPWAP) Protocol Specification", RFC 5415, March 2009.

[RFC5416] Calhoun, P., Montemurro, M., and D. Stanley, "Control and Provisioning of Wireless Access Points (CAPWAP) Protocol Binding for IEEE 802.11", RFC 5416, March 2009.

Authors' Addresses

Chunju Shao
China Mobile
No.32 Xuanwumen West Street
Beijing 100053
China

Email: shaochunju@chinamobile.com

Hui Deng
China Mobile
No.32 Xuanwumen West Street
Beijing 100053
China

Email: denghui@chinamobile.com

Rajesh S. Pazhyannur
Cisco Systems
170 West Tasman Drive
San Jose, CA 95134
USA

Email: rpazhyan@cisco.com

Farooq Bari
AT&T
7277 164th Ave NE
Redmond WA 98052
USA

Email: farooq.bari@att.com

Rong Zhang
China Telecom
No.109 Zhongshandadao avenue
Guangzhou 510630
China

Email: zhangr@gsta.com

Satoru Matsushima
SoftBank Telecom
1-9-1 Higashi-Shinbashi, Munato-ku
Tokyo
Japan

Email: satoru.matsushima@g.softbank.co.jp

Internet Engineering Task Force
Internet-Draft
Intended status: Informational
Expires: August 18, 2014

M. Ersue, Ed.
Nokia Solutions and Networks
D. Romascanu
Avaya
J. Schoenwaelder
Jacobs University Bremen
February 14, 2014

Management of Networks with Constrained Devices: Problem Statement and Requirements
draft-ietf-opsawg-coman-probstate-reqs-01

Abstract

This document provides a problem statement, deployment and management topology options as well as the requirements for the management of networks where constrained devices are involved.

Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on August 18, 2014.

Copyright Notice

Copyright (c) 2014 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (http://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of

the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction 3
1.1. Overview 3
1.2. Terminology 4
1.3. Networks Types and Characteristics in Focus 5
1.4. Constrained Device Deployment Options 9
1.5. Management Topology Options 9
1.6. Managing the Constrainedness of a Device or Network . . . 10
1.7. Configuration and Monitoring Functionality Levels 13
2. Problem Statement 15
3. Requirements on the Management of Networks with Constrained Devices 17
3.1. Management Architecture/System 17
3.2. Management protocols and data model 22
3.3. Configuration management 25
3.4. Monitoring functionality 27
3.5. Self-management 32
3.6. Security and Access Control 33
3.7. Energy Management 35
3.8. SW Distribution 37
3.9. Traffic management 37
3.10. Transport Layer 39
3.11. Implementation Requirements 41
4. IANA Considerations 43
5. Security Considerations 44
6. Contributors 45
7. Acknowledgments 46
8. References 47
8.1. Normative References 47
8.2. Informative References 47
Appendix A. Change Log 48
A.1. draft-ietf-opsawg-coman-probstate-reqs-00 - draft-ietf-opsawg-coman-probstate-reqs-01 48
A.2. draft-ersue-constrained-mgmt-03 - draft-ietf-opsawg-coman-probstate-reqs-00 48
A.3. draft-ersue-constrained-mgmt-02-03 49
A.4. draft-ersue-constrained-mgmt-01-02 50
A.5. draft-ersue-constrained-mgmt-00-01 50
Authors' Addresses 52

1. Introduction

1.1. Overview

Constrained devices, aka. sensor, smart object, or smart device, with limited CPU, memory, and power resources, can constitute a network. Such a network of constrained devices itself may be constrained or challenged, e.g. with unreliable or lossy channels, wireless technologies with limited bandwidth and a dynamic topology, needing the service of a gateway or proxy to connect to the Internet. In other scenarios, the constrained devices can be connected to a non-constrained network using off-the-shelf protocol stacks.

Constrained devices might be in charge of gathering information in diverse settings including natural ecosystems, buildings, and factories and send the information to one or more server stations. Constrained devices may also work under severe resource constraints such as limited battery and computing power, little memory and insufficient wireless bandwidth, and communication capabilities. A central entity, e.g., a base station or controlling server, might have more computational and communication resources and can act as a gateway between the constrained devices and the application logic in the core network.

Today diverse size of constrained devices with different resources and capabilities are being connected. Mobile personal gadgets, building-automation devices, cellular phones, Machine-to-machine (M2M) devices, etc. benefit from interacting with other "things" in the near or somewhere in the Internet. With this the Internet of Things (IoT) becomes a reality build up of uniquely identifiable objects (things). And over the next decade, this could grow to trillions of constrained devices and will greatly increase the Internet’s size and scope.

Network management is characterized by monitoring network status, detecting faults, and inferring their causes, setting network parameters, and carrying out actions to remove faults, maintain normal operation, and improve network efficiency and application performance. The traditional network management application periodically collects information from a set of elements that are needed to manage, processes the data, and presents them to the network management users. Constrained devices, however, often have limited power, low transmission range, and might be unreliable. They might also need to work in hostile environments with advanced security requirements or need to be used in harsh environments for a long time without supervision. Due to such constraints, the management of a network with constrained devices offers different type of challenges compared to the management of a traditional IP

network.

The IETF has already done substantial standardization work to enable the communication in IP networks and to manage such networks as well as the manifold type of nodes in these networks [RFC6632]. However, the IETF so far has not developed any specific technologies for the management of constrained devices and the networks comprised by constrained devices. IP-based sensors or constrained devices in such an environment, i.e., devices with very limited memory and CPU resources, use today application-layer protocols in an ad-hoc manner to do simple resource management and monitoring.

This document provides a problem statement and lists the requirements for the management of a network with constrained devices. Section 1.3 and Section 1.5 describe different topology options for the networking and management of constrained devices. Section 2 provides a problem statement on the issue of the management of networked constrained devices. Section 3 lists requirements on the management of applications and networks with constrained devices. Note that the requirements in Section 3 need to be seen as standalone, where different implementer may decide to realize a different set of requirements.

The use cases in the context of networks with constrained devices can be found in the companion document [COM-USE].

1.2. Terminology

Concerning constrained devices and networks this document generally builds on the terminology defined in [I-D.ietf-lwig-terminology], where the terms Constrained Device, Constrained Network, etc. are defined.

The following terms are additionally used throughout this documentation:

AMI: (Advanced Metering Infrastructure) A system including hardware, software, and networking technologies that measures, collects, and analyzes energy usage, and communicates with a hierarchically deployed network of metering devices, either on request or on a schedule.

C0: Class 0 constrained device as defined in Section 3. of [I-D.ietf-lwig-terminology].

C1: Class 1 constrained device as defined in Section 3. of [I-D.ietf-lwig-terminology].

C2: Class 2 constrained device as defined in Section 3. of [I-D.ietf-lwig-terminology].

Network of Constrained Devices: A network to which constrained devices are connected that may or may not be a Constrained Network (see [I-D.ietf-lwig-terminology] for the definition of the term Constrained Network).

M2M: (Machine to Machine) stands for the automatic data transfer between devices of different kind. In M2M scenarios a device (such as a sensor or meter) captures an event, which is relayed through a network (wireless, wired or hybrid) to an application.

MANET: Mobile Ad-hoc Networks, a self-configuring and infrastructureless network of mobile devices connected by wireless technologies.

Smart Grid: An electrical grid that uses communication technologies to gather and act on information in an automated fashion to improve the efficiency, reliability and sustainability of the production and distribution of electricity.

Smart Meter: An electrical meter in the context of a Smart Grid.

For a detailed discussion on the constrained networks as well as classes of constrained devices and their capabilities please see [I-D.ietf-lwig-terminology].

1.3. Networks Types and Characteristics in Focus

In this document we differentiate following type of networks concerning their transport and communication technologies:

Note that a network in general can involve constrained and non-constrained devices.

1. Wireline non-constrained networks, e.g. an Ethernet-LAN with constrained and non-constrained devices involved.

2. A combination of wireline and wireless networks, which may or may not be mesh-based but have a multi-hop connectivity between constrained devices, utilizing dynamic routing in both the wireless and wireline portions of the network. Such networks usually support highly distributed applications with many nodes (e.g. environmental monitoring) and tend to deal with large-scale

multipoint-to-point systems with massive data flows. Wireless Mesh Networks (WMN), as a specific variant, use off-the-shelf radio technology such as Wi-Fi, WiMax, and cellular 3G/4G. WMNs are reliable based on the redundancy they offer and have often a more planned deployment to provide dynamic and cost effective connectivity over a certain geographic area.

3. A combination of wireline and wireless networks with point-to-point or point-to-multipoint communication generally with single-hop connectivity to constrained devices, utilizing static routing over the wireless network. Such networks support short-range, point-to-point, low-data-rate, source-to-sink type of applications such as RFID systems, light switches, fire and smoke detectors, and home appliances. This type of networks also support confined short-range spaces such as a home, a factory, a building, or the human body. IEEE 802.15.1 (Bluetooth) and IEEE 802.15.4 are well-known examples of applicable standards for such networks.

4. Self-configuring infrastructureless networks of mobile devices (e.g. Mobile Adhoc networks, MANET) are a particular type of network connected by wireless technologies. Infrastructureless networks are mostly based on point-to-point communications of devices moving independently in any direction and changing the links to other devices frequently. Such devices do act as a router to forward traffic unrelated to their own use.

Wireline non-constrained networks with constrained and non-constrained devices are mainly used for specific applications like Building Automation or Infrastructure Monitoring. Wireline and wireless networks with multi-hop or point-to-multipoint connectivity are used e.g. for environmental monitoring as well as transport and mobile applications.

Furthermore different network characteristics are determined by multiple dimensions: dynamicity of the topology, bandwidth, and loss rate. In the following, each dimension is explained, and networks in scope for this document are outlined:

Network Topology:

The topology of a network can be represented as a graph, with edges (i.e., links) and vertices (routers and hosts). Examples of different topologies include "star" topologies (with one central node and multiple nodes in one hop distance), tree structures (with each node having exactly one parent), directed acyclic graphs (with each node having one or more parents), clustered topologies (where one or more "cluster heads" are responsible for a certain area of the

network), mesh topologies (fully distributed), etc.

Management protocols may take advantage of specific network topologies, for example by distributing large-scale management tasks amongst multiple distributed network management stations (e.g., in case of a mesh topology), or by using a hierarchical management approach (e.g., in case of a tree topology). These different management topology options are described in Section 1.6.

Note that in certain network deployments, such as community ad hoc networks (see the use case "Community Network Applications" in [COM-USE]), the topology is not pre-planned, and thus may be unknown for management purposes. In other use cases, such as industrial applications (see the use case "Industrial Applications" in [COM-USE]), the topology may be designed in advance and therefore taken advantage of when managing the network.

Dynamicity of the network topology:

The dynamicity of the network topology determines the rate of change of the graph per time. Such changes can occur due to different factors, such as mobility of nodes (e.g., in MANETs or cellular networks), duty cycles (for low-power devices enabling their network interface only periodically to transmit or receive packets), or unstable links (in particular wireless links with strongly fluctuating link quality).

Examples of different levels of dynamicity of the topology are Ethernets (with typically a very static topology) on the one side, and low-power and lossy networks (LLNs) on the other side. LLNs nodes often using duty cycles, operate on unreliable wireless links and are potentially mobile (e.g. for sensor networks).

The more the topology is dynamic, the more routing, transport and application layer protocols have to cope with interrupted connectivity and/or longer delays. For example, management protocols (with a given underlying transport protocol) that expect continuous session flows without changes of routes during a communication flow, may fail to operate.

Networks with a very low dynamicity (e.g. Ethernet) with no or infrequent topology changes (e.g. less than once every 30 minutes), are in-scope of this document if they are used with constrained devices (see e.g. the use case "Building Automation" in [COM-USE]).

Traffic flows:

The traffic flow in a network determines from which sources data

traffic is sent to which destinations in the network. Several different traffic flows are defined in [RFC7102], including "point-to-point" (P2P), "multipoint-to-point" (MP2P), and "point-to-multipoint" (P2MP) flows as:

- P2P: Point To Point. This refers to traffic exchanged between two nodes (regardless of the number of hops between the two nodes).

- P2MP: Point-to-Multipoint traffic refers to traffic between one node and a set of nodes. This is similar to the P2MP concept in Multicast or MPLS Traffic Engineering.

- MP2P: Multipoint-to-Point is used to describe a particular traffic pattern (e.g. MP2P flows collecting information from many nodes flowing inwards towards a collecting sink).

If one of these traffic patterns is predominant in a network, protocols (routing, transport, application) may be optimized for the specific traffic flow. For example, in a network with a tree topology and MP2P traffic, collection tree protocols are efficient to send data from the leaves of the tree to the root of the tree, via each node’s parent.

Bandwidth:

The bandwidth of the network is the amount of data that can be sent per time between two communication end-points. It is usually determined by the link with the minimum bandwidth on the path from the source to the destination of data packets. The bandwidth in networks can range from a few Kilobytes per second (such as on some 802.15.4 link layers) to many Gigabytes per second (e.g., on fiber optics).

For management purposes, the management protocol typically requires to send information between the network management station and the clients, for monitoring or control purposes. If the available bandwidth is insufficient for the management protocol, packets will be buffered and eventually dropped, and thus management is not possible with such a protocol.

Networks without bandwidth limitation (e.g. Ethernet) are in-scope of this document if they are used with constrained devices (see the use case "Building Automation" in [COM-USE]).

Loss rate:

The loss rate (or bit error rate) is the number of bit errors divided by the total number of bits transmitted. For wired networks, loss

rates are typically extremely low, e.g. around 10^-12 or 10^-13 for the latest 10Gbit Ethernet. For wireless networks, such as 802.15.4, the bit error rate can be as high as 10^-1 to 10^-0 in case of interferences.Even when using a reliable transport protocol, management operations can fail if the loss rate is too high, unless they are specifically designed to cope with these situations.

1.4. Constrained Device Deployment Options

We differentiate following deployment options for the constrained devices:

- a network of constrained devices, which communicate with each other,
- Constrained devices, which are connected directly to the Internet or an IP network
- A network of constrained devices which communicate with a gateway or proxy with more communication capabilities acting possibly as a representative of the device to entities in the non-constrained network
- Constrained devices, which are connected to the Internet or an IP network via a gateway/proxy
- A hierarchy of constrained devices, e.g., a network of C0 devices connected to one or more C1 devices - connected to one or more C2 devices - connected to one or more gateways - connected to some application servers or NMS system
- The possibility of device grouping (possibly in a dynamic manner) such as that the grouped devices can act as one logical device at the edge of the network and one device in this group can act as the managing entity

1.5. Management Topology Options

We differentiate following options for the management of networks of constrained devices:

- A network of constrained devices managed by one central manager. A logically centralized management might be implemented in a hierarchical fashion for scalability and robustness reasons. The manager and the management application logic might have a gateway/ proxy in between or might be on different nodes in different networks, e.g., management application running on a cloud server.

- Distributed management, where a network of constrained devices is managed by more than one manager. Each manager controls a subnetwork and may communicate directly with other manager stations in a cooperative fashion. The distributed management may be weakly distributed, where functions are broken down and assigned to many managers dynamically, or strongly distributed, where almost all managed things have embedded management functionality and explicit management disappears, which usually comes with the price that the strongly distributed management logic now needs to be managed.

- Hierarchical management, where a hierarchy of networks with constrained devices are managed by the managers at their corresponding hierarchy level. I.e. each manager is responsible for managing the nodes in its sub-network. It passes information from its sub-network to its higher-level manager, and disseminates management functions received from the higher-level manager to its sub-network. Hierarchical management is essentially a scalability mechanism, logically the decision-making may be still centralized.

1.6. Managing the Constrainedness of a Device or Network

The capabilities of a constrained device or network and the constrainedness thereof influence and have an impact on the requirements for the management of such network or devices.

A constrained device:

- might only support an unreliable radio with lossy links, i.e. the client and server of a management protocol need to gracefully ignore incomplete commands or repeat commands as necessary.

- might only be able to go online from time-to-time, where it is reachable, i.e. a command might be necessary to repeat after a longer timeout or the timeout value with which one endpoint waits on a response needs to be sufficiently high.

- might only be able to support a limited operating time (e.g. based on the available battery), or may behave as ’sleepy endpoints’ setting their network links to a disconnected state during long periods of time i.e. the devices need to economize their energy usage with suitable mechanisms and the managing entity needs to monitor and control the energy status of the constrained devices it manages.

- might only be able to support one simple communication protocol, i.e. the management protocol needs to be possible to downscale from constrained (C2) to very constrained (C0) devices with

modular implementation and a very basic version with just a few simple commands.

- might only be able to support limited or no user and/or transport security, i.e. the management system needs to support a less-costly and simple but sufficiently secure authentication mechanism.

- might not be able to support compression and decompression of exchanged data based on limited CPU power, i.e. an intermediary entity which is capable of data compression should be able to communicate with both, devices, which support data compression (e.g. C2) and devices, which do not support data compression (e.g. C1 and C0).

- might only be able to support a simple encryption, i.e. it would be beneficial if the devices use cryptographic algorithms that are supported in hardware and the encryption used is efficient in terms of memeory and CPU usage.

- might only be able to communicate with one single managing entity and cannot support the parallel access of many managing entities.

- might depend on a self-configuration feature, i.e. the managing entity might not know all devices in a network and the device needs to be able to initiate connection setup for the device configuration.

- might depend on self- or neighbor-monitoring feature, i.e. the managing entity might not be able to monitor all devices in a network continuously.

- might only be able to communicate with its neighbors, i.e. the device should be able to get its configuration from a neighbor.

- might only be able to support parsing of data models with limited size, i.e. the device data models need to be compact containing the most necessary data and if possible parsable as a stream.

- might only be able to support a limited or no failure detection, i.e. the managing entity needs to handle the situation, where a failure does not get detected or gets detected late gracefully e.g. with asking repeatedly.

- might only be able to support the reporting of just one or a limited set failure types.

- might only be able to support a limited set of notifications, possible only an "I-am-alive" message.

- might only be able to support a soft-reset from failure recovery.

- might possibly generate a huge amount of redundant reporting data, i.e. the intermediary management entity (see [I-D.ietf-core-coap]) should be able to filter and aggregate redundant data.

A network of constrained devices:

- might only support an unreliable radio with lossy links, i.e. the client and server of a management protocol need to repeat commands as necessary or gracefully ignore incomplete commands.

- might be necessary to manage based on multicast communication, i.e. the managing entity needs to be prepared to configure many devices at once based on the same data model.

- might have a very large topology supporting 10.000 or more nodes for some applications and as such node naming is a specific issue for constrained networks.

- must be able to self-organize, i.e. given the large number of nodes and their potential placement in hostile locations and frequently changing topology, manual configuration is typically not feasible. As such the network must be able to reconfigure itself so that it can continue to operate properly and support reliable connectivity.

- needs a management solution, which is energy-efficient, using as little wireless bandwidth as possible since communication is highly energy demanding.

- needs to support localization schemes to determine the location of devices since the devices might be moving and location information is important for some applications.

- needs a management solution, which is scalable as the network may consist of thousands of nodes and may need to be extended continuously.

- needs to provide fault tolerance. Faults in network operation including hardware and software errors or failures detected by the transport protocol should be handled smoothly enabling. In such a case it should be possible to run the protocol possibly at a reduced level but avoiding to fail completely. E.g. self-monitoring mechanisms or graceful degradation of features can be

used to provide fault tolerance.

- might require new management capabilities: for example, network coverage information and a constrained device power-distribution-map.
- might require a new management function for data management, since the type and amount of data collected in constrained networks is different from those of the traditional networks.
- might also need energy-efficient key management.

1.7. Configuration and Monitoring Functionality Levels

Devices often differ significantly on the level of configuration management support they provide. The configuration management functionality levels can be broadly classified as follows:

CL0: Devices are pre-configured and allow no runtime configuration changes. Configuration parameters are often hard coded and compiled directly into the firmware image.

CL1: Devices have explicit configuration objects. However, changes require a restart of the device to take effect.

CL2: Devices allow management systems to replace the entire configuration (or pre-determined subsets) in bulk. Configuration changes take effect by soft-restarts of the system (or subsystems).

CL3: Devices allow management systems to modify configuration objects without bulk replacements and changes take effect immediately.

CL4: Devices support multiple configuration datastores and they might distinguish between the currently running and the next startup configuration.

CL5: Devices support configuration datastore locking and device-local configuration change transactions, i.e., either all configuration changes are applied or none of them.

CL6: Devices support configuration change transactions across devices.

Devices often also provide different levels of monitoring support:

ML0: Devices push pre-defined monitoring data.

ML1: Devices allow management systems to pull pre-defined monitoring data.

ML2: Devices allow management systems to pull user-defined filtered subsets of monitoring data.

ML3: Devices are able to locally process monitoring data in order to detect threshold crossings or to aggregate data.

Constrained devices often implement a combination of one of FL0-FL2 with one of ML0-ML1.

2. Problem Statement

The terminology for the "Internet of Things" is still nascent, and depending on the network type or layer in focus diverse technologies and terms are in use. Common to all these considerations is the "Things" or "Objects" are supposed to have physical or virtual identities using interfaces to communicate. In this context, we need to differentiate between the Constrained and Smart Devices identified by an IP address compared to virtual entities such as Smart Objects, which can be identified as a resource or a virtual object by using a unique identifier. Furthermore, the smart devices usually have a limited memory and CPU power as well as aim to be self-configuring and easy to deploy.

However, the constraints of the network nodes requires a rethinking of the protocol characteristics concerning power consumption, performance, memory, and CPU usage. As such, there is a demand for protocol simplification, energy-efficient communication, less CPU usage and small memory footprint.

On the application layer the IETF is already developing protocols like the Constrained Application Protocol (CoAP) [I-D.ietf-core-coap] enabling the communication of constrained devices and networks e.g., for smart energy applications or home automation environments. The deployment of such an environment involves in fact many, in some scenarios up to million constrained devices (e.g. smart meters), which produce a huge amount of data. This data needs to be collected, filtered, and pre-processed for further use in diverse services.

Considering the high number of nodes to deploy, one has to think on the manageability aspects of the smart devices and plan for easy deployment, configuration, and management of the networks of constrained devices as well as the devices themselves. Consequently, seamless monitoring and self-configuration of such network nodes becomes more and more imperative. Self-configuration and self-management is already a reality in the standards of some of the bodies such as 3GPP. To introduce self-configuration of smart devices successfully a device-initiated connection establishment is required.

A simple and efficient application layer protocol, such as CoAP, is essential to address the issue of efficient object-to-object communication and information exchange. Such an information exchange should be done based on interoperable data models to enable the exchange and interpretation of diverse application and management related data.

In an ideal world, we would have only one network management protocol for monitoring, configuration, and exchanging management data, independently of the type of the network (e.g., Smart Grid, wireless access, or core network). Furthermore, it would be desirable to derive the basic data models for constrained devices from the core models used today to enable reuse of functionality and end-to-end information exchange. However, the current management protocols seem to be too heavyweight compared to the capabilities the constrained devices have and are not applicable directly for the use in a network of constrained devices. Furthermore, the data models addressing the requirements of such smart devices need yet to be designed.

The IETF so far has not developed any specific technologies for the management of constrained devices and the networks comprised by constrained devices. IP-based sensors or constrained devices in such an environment, i.e., devices with very limited memory and CPU resources, use today, e.g., application-layer protocols to do simple resource management and monitoring. This might be sufficient for some basic cases, however, there is a need to reconsider the network management mechanisms based on the new, changed, as well as reduced requirements coming from smart devices and the network of such constrained devices. Albeit it is questionable whether we can take the same comprehensive approach we use in an IP network also for the management of constrained devices. Hence, the management of a network with constrained devices is necessary to design in a simplified and less complex manner.

As Section 1.6 highlights, there are diverse characterists of constrained devices or networks, which stem from their constrainedness and therefore have an impact on the requirements for the management of such a network with constrained devices. The use cases discussed in [COM-USE] show that the requirements on constrained networks are manifold and need to be analyzed from different angles, e.g. concerning the design of the management architecture, the selection of the appropriate protocol features as well as the specific issues which are new in the context of constrained devices. Examples of such issues are e.g. the careful management of the scarce energy resources, the necessity for self-organization and self-management of such devices but also the implementation considerations to enable the use of common communication technologies on a constrained hardware in an efficient manner. For an exhaustive list of issues and requirements, which need to be addressed for the management of a network with constrained devices please see Section 1.6 and Section 3.

3. Requirements on the Management of Networks with Constrained Devices

This section describes the requirements categorized by management areas listed in subsections.

Note that the requirements in this section need to be seen as standalone requirements. A device might be able to provide only a particular profile of requirements (i.e. selected set of requirements) and might not be capable to provide all requirements in this document. On the other hand a device vendor might select a subset of the requirements to implement. As of today this document does not recommend the realization of a profile of requirements.

Following template is used for the definition of the requirements.

Req-ID: An ID uniquely identified by a three-digit number

Title: The title of the requirement.

Description: The rational and description of the requirement.

Source: The origin of the requirement and the matching use case or application. For the discussion of referred use cases for constrained management please see [COM-USE].

Requirement Type: Functional Requirement, Non-Functional Requirement. A functional requirement is related to a proposed function or component. As such functional requirements may be technical details, or specific functionality that define what a system is supposed to accomplish. Non-functional requirements (also known as design constraints or quality requirements) impose implementation related considerations such as performance requirements, security, or reliability.

Device type: The device types by which this requirement can be supported: C0, C1 and/or C2.

Priority: The priority of the requirement showing it’s importance for a particular type of device: High, Medium, and Low. The priority of a requirement can be High e.g. for a C2 device but Low for a C1 or C0 device as the realization of complex features in a C1 device is in many cases not possible.

3.1. Management Architecture/System

Req-ID: 1.001

Title: Support multiple device classes within a single network.

Description: Larger networks usually are made up of devices belonging to different device classes (e.g., constrained mesh endpoints and less constrained routers) that work together. Hence, the management architecture must be applicable to networks that have a mix of different device classes. See Section 3. of [I-D.ietf-lwig-terminology] for the definition of Constrained Device Classes.

Source: All use cases.

Requirement Type: Non-Functional Requirement

Device type: C1 and/or C2

Priority: High

Req-ID: 1.002

Title: Management scalability.

Description: The management architecture must be able to scale with the number of devices involved and operate efficiently in any network size and topology. This implies that e.g. the managing entity is able to handle huge amount of device monitoring data and the management protocol is not sensitive to the decrease of the time between two client requests. To achieve good scalability, caching techniques, in-network data aggregation techniques, hierarchical management models may be used.

Source: General requirement for all use cases to enable large scale networks.

Requirement Type: Non-Functional Requirement

Device type: C0, C1, and C2

Priority: High

Req-ID: 1.003

Title: Hierarchical management

Description: Provide a means of hierarchical management, i.e. provide intermediary management entities on different levels, which can take over the responsibility for the management of a sub-hierarchy of the network of constraint devices. The intermediary management entity can e.g. support management data aggregation to handle e.g. high-frequent monitoring data or provide a caching mechanism for the uplink and downlink communication. Hierarchical management contributes to management scalability.

Source: Use cases where a huge amount of devices are deployed with a hierarchical topology.

Requirement Type: Non-Functional Requirement

Device type: Managing and intermediary entities.

Priority: Medium

Req-ID: 1.004

Title: Minimize state maintained on constrained devices.

Description: The amount of state that needs to be maintained on constrained devices should be minimized. This is important in order to save memory (especially relevant for C0 and C1 devices) and in order to allow devices to restart for example to apply configuration changes or to recover from extended periods of inactivity.

Note: One way to achieve this is to adopt a RESTful architecture that minimizes the amount of state maintained by managed constrained devices and that makes resources of a device addressable via URIs.

Source: Basic requirement which concerns all use cases.

Requirement Type: Functional Requirement

Device type: C0, C1, and C2

Priority: High

Req-ID: 1.005

Title: Automatic re-synchronization with eventual consistency.

Description: To support large scale networks, where some constrained devices may be offline at any point in time, it is necessary to distribute configuration parameters in a way that allows temporary inconsistencies but eventually converges, after a sufficiently long period of time without further changes, towards global consistency.

Source: Use cases with large scale networks with many devices.

Requirement Type: Functional Requirement

Device type: C0, C1, and C2

Priority: High

Req-ID: 1.006

Title: Support for lossy links and unreachable devices.

Description: Some constrained devices will only be able to support lossy and unreliable links characterized by a limited data rate, a high latency, and a high transmission error rate. Furthermore constrained devices often duty cycle their radio or the whole device in order to save energy. Some classes of devices labelled as ’sleepy endpoints’ set their network links to a disconnected state during long periods of time. In all cases the management system must not assume that constrained devices are always reachable.

Source: Basic requirement for networks of constrained devices with unreliable links and constrained devices which sleep to save energy.

Requirement Type: Non-Functional Requirement

Device type: C0, C1, and C2

Priority: High

Req-ID: 1.007

Title: Network-wide configuration

Description: Provide means by which the behavior of the network can be specified at a level of abstraction (network-wide configuration) higher than a set of configuration information specific to individual devices. It is useful to derive the device specific configuration from the network-wide configuration. Such a repository can be used to configure pre-defined device or protocol parameters for the whole network. Furthermore, such a network-wide view can be used to monitor and manage a group of routers or a whole network. E.g. monitoring the performance of a network requires additional information other than what can be acquired from a single router using a management protocol.

Note: The identification of the relevant subset of the policies to be provisioned is according to the capabilities of each device and can be obtained from a pre-configured data-repository.

Source: In general all use cases, which want to configure the network and its devices based on a network view in a top-down manner.

Requirement Type: Non-Functional Requirement

Device type: C0, C1, and C2

Priority: Medium

Req-ID: 1.008

Title: Distributed Management

Description: Provide a means of simple distributed management, where a network of constrained devices can be managed or monitored by more than one manager. Since the connectivity to a server cannot be guaranteed at all times, a distributed approach may provide a

higher reliability, at the cost of increased complexity. This requirement implies the handling of data consistency in case of concurrent read and write access to the device datastore. It might also happen that no management (configuration) server is accessible and the only reachable node is a peer device. In this case the device should be able to obtain its configuration from peer devices.

Source: Use cases where the count of devices to manage is high.

Requirement Type: Non-Functional Requirement

Device type: C1 and C2

Priority: Medium

3.2. Management protocols and data model

Req-ID: 2.001

Title: Modular implementation of management protocols

Description: Management protocols should be specified to allow for modular implementations, i.e., it should be possible to implement only a basic set of protocol primitives on highly constrained devices while devices with additional resources may provide more support for additional protocol primitives. See Section 1.7 for a discussion on the level of configuration management and monitoring support constrained devices may provide.

Source: Basic requirement interesting for all use cases.

Requirement Type: Non-Functional Requirement

Device type: C0, C1, and C2

Priority: High

Req-ID: 2.002

Title: Compact encoding of management data

Description: The encoding of management data should be compact and space efficient, enabling small message sizes.

Source: General requirement to save memory for the receiver buffer and on-air bandwith.

Requirement Type: Functional Requirement

Device type: C0, C1, and C2

Priority: High

Req-ID: 2.003

Title: Compression of management data or complete messages

Description: Management data exchanges can be further optimized by applying data compression techniques or delta encoding techniques. Compression typically requires additional code size and some additional buffers and/or the maintenance of some additional state information. For C0 devices compression may not be feasible.

Source: Use cases where it is beneficial to reduce transmission time and bandwith, e.g. mobile applications which require to save on-air bandwith.

Requirement Type: Functional Requirement

Device type: C1 and C2

Priority: Medium

Req-ID: 2.004

Title: Mapping of management protocol interactions.

Description: It is desirable to have a loss-less automated mapping between the management protocol used to manage constrained devices and the management protocols used to manage regular devices. In the ideal case, the same core management protocol can be used with certain restrictions taking into account the resource limitations of constrained devices. However, for very resource constrained devices, this goal might not be achievable.

Source: Use cases where high-frequent interaction with the management system of a non-constrained network is required.

Requirement Type: Functional Requirement

Device type: C1 and C2

Priority: Medium

Req-ID: 2.005

Title: Consistency of data models with the underlying information model.

Description: The data models used by the management protocol must be consistent with the information model used to define data models for non-constrained networks. This is essential to facilitate the integration of the management of constrained networks with the management of non-constrained networks. Using an underlying information model for future data model design enables furthermore top-down model design and model reuse as well as data interoperability (i.e. exchange of management information between the constrained and non-constrained networks). This is a strong requirement, even despite the fact that the underlying information models are often not explicitly documented in the IETF.

Source: General requirement to support data interoperability, consistency and model reuse.

Requirement Type: Non-Functional Requirement

Device type: C0, C1, and C2

Priority: High

Req-ID: 2.006

Title: Loss-less mapping of management data models.

Description: It is desirable to have a loss-less automated mapping between the management data models used to manage regular devices and the management data models used for managing constrained devices. In the ideal case, the same core data models can be used with certain restrictions taking into account the resource

limitations of constrained devices. However, for very resource constrained devices, this goal might not be achievable.

Source: Use cases where consistent data exchange with the management system of a non-constrained network is required.

Requirement Type: Functional Requirement

Device type: C2

Priority: Medium

Req-ID: 2.007

Title: Protocol extensibility

Description: Provide means of extensibility for the management protocol, i.e. by adding new protocol messages or mechanisms that can deal with the changing requirements on a supported message and data types effectively, without causing inter-operability problems or having to replace/update large amounts of deployed devices.

Source: Basic requirement useful for all use cases.

Requirement Type: Functional Requirement

Device type: C0, C1, and C2

Priority: High

3.3. Configuration management

Req-ID: 3.001

Title: Self-configuration capability

Description: Automatic configuration and re-configuration of devices without manual intervention. Compared to the traditional management of devices where the management application is the central entity configuring the devices, in the auto-configuration scenario the device is the active part and initiates the configuration process. Self-configuration can be initiated during the initial configuration or for subsequent configurations, where the configuration data needs to be refreshed. Self-configuration should be also supported during the initialization phase or in the event of failures, where prior knowledge of the network topology

is not available or the topology of the network is uncertain.

Source: In general all use cases requiring easy deployment and plug& play behavior as well as easy maintenance of many constrained devices.

Requirement Type: Functional Requirement

Device type: C0, C1, and C2

Priority: High for device categories C0 and C1, Medium for C2.

Req-ID: 3.002

Title: Capability Discovery

Description: Enable the discovery of supported optional management capabilities of a device and their exposure via at least one protocol and/or data model.

Source: Use cases where the device interaction with other devices or applications is a function of the level of support for its capabilities.

Requirement Type: Functional Requirement

Device type: C1 and C2

Priority: Medium

Req-ID: 3.003

Title: Asynchronous Transaction Support

Description: Provide configuration management with asynchronous (event-driven) transaction support. Configuration operations must support a transactional model, with asynchronous indications that the transaction was completed.

Source: Use cases, which require transaction-oriented processing because of reliability or distributed architecture functional requirements.

Requirement Type: Functional Requirement

Device type: C1 and C2

Priority: Medium

Req-ID: 3.004

Title: Network reconfiguration

Description: Provide a means of iterative network reconfiguration in order to recover the network functionality from node and communication faults. The network reconfiguration can be failure-driven and self-initiated (automatic reconfiguration). The network reconfiguration can be also performed on the whole hierarchical structure of a network (network topology).

Source: Practically all use cases, as network connectivity is a basic requirement.

Requirement Type: Functional Requirement

Device type: C0, C1, and C2

Priority: Medium

3.4. Monitoring functionality

Req-ID: 4.001

Title: Device status monitoring

Description: Provide a monitoring function to collect and expose information about device status and exposing it via at least one management interface. The device monitoring might make use of the hierarchical management through the intermediary entities and the caching mechanism. The device monitoring might also make use of neighbor-monitoring (fault detection in local network) to support fast fault detection and recovery, e.g. in a scenario where a managing entity is unreachable and a neighbor can take over the monitoring responsibility.

Source: All use cases

Requirement Type: Functional Requirement

Device type: C0, C1, and C2

Priority: High, Medium for neighbor-monitoring.

Req-ID: 4.002

Title: Energy status monitoring

Description: Provide a monitoring function to collect and expose information about device energy parameters and usage (e.g. battery level and communication power).

Source: Use case Energy Management

Requirement Type: Functional Requirement

Device type: C0, C1, and C2

Priority: High for energy reporting devices, Low for others.

Req-ID: 4.003

Title: Monitoring of current and estimated device availability

Description: Provide a monitoring function to collect and expose information about current device availability (energy, memory, computing power, forwarding plane utilization, queue buffers, etc.) and estimation of remaining available resources.

Source: All use cases. Note that monitoring energy resources (like battery status) may be required on all kinds of devices.

Requirement Type: Functional Requirement

Device type: C0, C1, and C2

Priority: Medium

Req-ID: 4.004

Title: Network status monitoring

Description: Provide a monitoring function to collect, analyse and expose information related to the status of a network or network segments connected to the interface of the device.

Source: All use cases.

Requirement Type: Functional Requirement

Device type: C1 and C2

Priority: Low, based on the realization complexity.

Req-ID: 4.005

Title: Self-monitoring

Description: Provide self-monitoring (local fault detection) feature for fast fault detection and recovery.

Source: Use cases where the devices cannot be monitored centrally in appropriate manner, e.g. self-healing is required.

Requirement Type: Functional Requirement

Device type: C1 and C2

Priority: High for C2, Medium for C1

Req-ID: 4.006

Title: Performance Monitoring

Description: The device will provide a monitoring function to collect and expose information about the basic performance parameter of the device. The performance management functionality might make use of the hierarchical management through the intermediary devices.

Source: Use cases Building automation, and Transport applications

Requirement Type: Functional Requirement

Device type: C1 and C2

Priority: Low

Req-ID: 4.007

Title: Fault detection monitoring

Description: The device will provide fault detection monitoring. The system collects information about network states in order to identify whether faults have occurred. In some cases the detection of the faults might be based on the processing and analysis of the parameters retrieved from the network or other devices. In case of C0 devices the monitoring might be limited to the check whether the device is alive or not.

Source: Use cases Environmental Monitoring, Building Automation, Energy Management, Infrastructure Monitoring

Requirement Type: Functional Requirement

Device type: C0, C1 and C2

Priority: Medium

Req-ID: 4.008

Title: Passive and Reactive Monitoring

Description: The device will provide passive and reactive monitoring capabilities. The system or manager collects information about device components and network states (passive monitoring) and may perform postmortem analysis of collected data. In case events of interest have occurred the system or manager can adaptively react (reactive monitoring), e.g. reconfigure the network. Typically actions (re-actions) will be executed or sent as commands by the management applications.

Source: Diverse use cases relevant for device status and network state monitoring

Requirement Type: Functional Requirement

Device type: C2

Priority: Medium

Req-ID: 4.009

Title: Recovery

Description: Provide local, central and hierarchical recovery mechanisms (recovery is in some cases achieved by recovering the whole network of constrained devices).

Source: Use cases Industrial applications, Home and Building Automation, Mobile Applications that involve different forms of clustering or area managers.

Requirement Type: Functional Requirement

Device type: C2

Priority: Medium

Req-ID: 4.010

Title: Network topology discovery

Description: Provide a network topology discovery capability (e.g. use of topology extraction algorithms to retrieve the network state) and a monitoring function to collect and expose information about the network topology.

Source: Use cases Community Network Applications and Mobile Applications

Requirement Type: Functional Requirement

Device type: C1 and C2

Priority: Low, based on the realization complexity.

Req-ID: 4.011

Title: Notifications

Description: The device will provide the capability of sending notifications on critical events and faults.

Source: All use cases.

Requirement Type: Functional Requirement

Device type: C0, C1, and C2

Priority: Medium for C2, Low for C0 and C1

Req-ID: 4.012

Title: Logging

Description: The device will provide the capability of building, keeping, and allowing retrieval of logs of events (including but not limited to critical faults and alarms).

Source: Use cases Industrial Applications, Building Automation, Infrastructure monitoring

Requirement Type: Functional Requirement

Device type: C2

Priority: High for some medical or industrial applications, Medium otherwise

3.5. Self-management

Req-ID: 5.001

Title: Self-management - Self-healing

Description: Enable event-driven and/or periodic self-management functionality in a device. The device should be able to react in case of a failure e.g. by initiating a fully or partly reset and initiate a self-configuration or management data update as necessary. A device might be further able to check for failures cyclically or schedule-controlled to trigger self-management as necessary. It is a matter of device design and subject for discussion how much self-management a C1 device can support. A minimal failure detection and self-management logic is assumed to be generally useful for the self-healing of a device.

Source: The requirement generally relates to all use cases in this document.

Requirement Type: Functional Requirement

Device type: C1 and C2

Priority: High for C2, Medium for C1

3.6. Security and Access Control

Req-ID: 6.001

Title: Authentication of management system and devices.

Description: Systems having a management role must be properly authenticated to the device such that the device can exercise proper access control and in particular distinguish rightful management systems from rogue systems. On the other hand managed devices must authenticate themselves to systems having a management role such that management systems can protect themselves from rogue devices. In certain application scenarios, it is possible that a large number of devices need to be (re)started at about the same time. Protocols and authentication systems should be designed such that a large number of devices (re)starting simultaneously does not negatively impact the device authentication process.

Source: Basic security requirement for all use cases.

Requirement Type: Functional Requirement

Device type: C0, C1, and C2

Priority: High, Medium for the (re)start of a large number of devices

---

Req-ID: 6.002

Title: Support suitable security bootstrapping mechanisms

Description: Mechanisms should be supported that simplify the bootstrapping of device that is the discovery of newly deployed devices in order to provide them with appropriate access control permissions.

Source: Basic security requirement for all use cases.

Requirement Type: Functional Requirement

Device type: C0, C1, and C2

Priority: High

Req-ID: 6.003

Title: Access control on management system and devices

Description: Systems acting in a management role must provide an access control mechanism that allows the security administrator to restrict which devices can access the managing system (e.g., using an access control white list of known devices). On the other hand managed constrained devices must provide an access control mechanism that allows the security administrator to restrict how systems in a management role can access the device (e.g., no-access, read-only access, and read-write access).

Source: Basic security requirement for use cases where access control is essential.

Requirement Type: Functional Requirement

Device type: C0, C1, and C2

Priority: High

Req-ID: 6.004

Title: Select cryptographic algorithms that are efficient in both code space and execution time.

Description: Cryptographic algorithms have a major impact in terms of both code size and overall execution time. It is therefore necessary to select mandatory to implement cryptographic algorithms that are reasonable to implement with the available code space and that have a small impact at runtime. Furthermore some wireless technologies (e.g., IEEE 802.15.4) require the support of certain cryptographic algorithms. It might be useful to choose algorithms that are likely to be supported in wireless chipsets for certain wireless technologies.

Source: Generic requirement to reduce the footprint and CPU usage of a constrained device.

Requirement Type: Non-Functional Requirement

Device type: C0, C1, and C2

Priority: High, Medium for hardware-supported algorithms.

3.7. Energy Management

Req-ID: 7.001

Title: Management of Energy Resources

Description: Enable managing power resources in the network, e.g. reduce the sampling rate of nodes with critical battery and reduce node transmission power, put nodes to sleep, put single interfaces to sleep, reject a management job based on available energy, criteria e.g. importance levels pre-defined by the management application, etc. (e.g. a task marked as essential can be executed even if the energy level is low). The device may further implement standard data models for energy management and expose it through a management protocol interface, e.g. EMAN MIB modules and extensions. It might be necessary to downscale EMAN MIBs for the use in C1 and C2 devices.

Source: Use case Energy Management

Requirement Type: Functional Requirement

Device type: C0, C1, and C2

Priority: Medium for the use case Energy Management, Low otherwise.

Req-ID: 7.002

Title: Support of energy-optimized communication protocols

Description: Use of an optimized communication protocol to minimize energy usage for the device (radio) receiver/transmitter, on-air bandwidth (protocol efficiency), reduced amount of data communication between nodes (implies data aggregation and filtering but also a compact format for the transferred data).

Source: Use cases Energy Management and Mobile Applications.

Requirement Type: Non-Functional Requirement

Device type: C2

Priority: Medium

Req-ID: 7.003

Title: Support for layer 2 energy-aware protocols

Description: The device will support layer 2 energy management protocols (e.g. energy-efficient Ethernet IEEE 802.3az) and be able to report on these.

Source: Use case Energy Management

Requirement Type: Non-Functional Requirement

Device type: C0, C1, and C2

Priority: Medium

Req-ID: 7.004

Title: Dying gasp

Description: When energy resources draw below the red line level, the device will send a dying gasp notification and perform if still possible a graceful shutdown including conservation of critical device configuration and status information.

Source: Use case Energy Management

Requirement Type: Functional Requirement

Device type: C0, C1, and C2

Priority: Medium

3.8. SW Distribution

Req-ID: 8.001

Title: Group-based provisioning

Description: Support group-based provisioning, i.e. firmware update and configuration management, of a large set of constrained devices with eventual consistency and coordinated reload times. The device should accept group-based configuration management based on bulk commands, which aim similar configurations of a large set of constrained devices of the same type in a given group, and which may share a common data model. Activation of configuration may be based on pre-loaded sets of default values.

Source: All use cases

Requirement Type: Non-Functional Requirement

Device type: C0, C1, and C2

Priority: Medium

3.9. Traffic management

Req-ID: 9.001

Title: Congestion avoidance

Description: Support congestion control principles as defined in [RFC2914], e.g. the ability to avoid congestion by modifying the device's reporting rate for periodical data (which is usually redundant) based on the importance and reliability level of the management data. This functionality is usually controlled by the managing entity, where the managing entity marks the data as important or relevant for reliability. However reducing a device's reporting rate can also be initiated by a device if it is able to detect congestion or has insufficient buffer memory.

Source: Use cases with high reporting rate and traffic e.g. AMI or M2M.

Requirement Type: Non-Functional Requirement

Device type: C1 and C2

Priority: Medium

Req-ID: 9.002

Title: Redirect traffic

Description: Provide the ability for network nodes to redirect traffic from overloaded intermediary nodes in a network to another path in order to prevent congestion on a central server and in the primary network.

Source: Use cases with high reporting rate and traffic e.g. AMI or M2M.

Requirement Type: Non-Functional Requirement

Device type: Intermediary entity in the network.

Priority: Medium

Req-ID: 9.003

Title: Traffic delay schemes.

Description: Provide the ability to apply delay schemes to incoming and outgoing links on an overloaded intermediary node as necessary in order to reduce the amount of traffic in the network.

Source: Use cases with high reporting rate and traffic e.g. AMI or M2M.

Requirement Type: Non-Functional Requirement

Device type: Intermediary entity in the network.

Priority: Medium

3.10. Transport Layer

Req-ID: 10.001

Title: Scalable transport layer

Description: Enable the use of a scalable transport layer, i.e. not sensitive to a high rate of incoming client requests, which is useful for applications requiring frequent access to device data.

Source: Applications with high frequent access to the device data.

Requirement Type: Non-Functional Requirement

Device type: C0, C1 and C2

Priority: Medium

Req-ID: 10.002

Title: Reliable unicast transport of messages

Description: Diverse applications need a reliable transport of messages. The reliability might be achieved based on a transport protocol such as TCP or can be supported based on message repetition if an acknowledgement is missing.

Source: Generally applications benefit from the reliability of the message transport.

Requirement Type: Functional Requirement

Device type: C0, C1, and C2

Priority: High

Req-ID: 10.003

Title: Best-effort multicast

Description: Provide best-effort multicast of messages, which is generally useful when devices need to discover a service provided by a server or many devices need to be configured by a managing entity at once based on the same data model.

Source: Use cases where a device needs to discover services as well as use cases with high amount of devices to manage, which are hierarchically deployed, e.g. AMI or M2M.

Requirement Type: Functional Requirement

Device type: C0, C1, and C2

Priority: Medium

Req-ID: 10.004

Title: Secure message transport.

Description: Enable secure message transport providing authentication, data integrity, confidentiality by using existing transport layer technologies with small footprint such as TLS/ DTLS.

Source: All use cases.

Requirement Type: Non-Functional Requirements

Device type: C1 and C2

Priority: High

3.11. Implementation Requirements

Req-ID: 11.001

Title: Avoid complex application layer transactions requiring large application layer messages.

Description: Complex application layer transactions tend to require large memory buffers that are typically not available on C0 or C1 devices and only by limiting functionality on C2 devices. Furthermore, the failure of a single large transaction requires repeating the whole transaction. On constrained devices, it is often more desirable to a large transaction down into a sequence of smaller transactions, which require less resources and allow to make progress using a sequence of smaller steps.

Source: Basic requirement which concerns all use cases with memory constrained devices.

Requirement Type: Non-Functional Requirement

Device type: C0, C1, and C2

Priority: High

Req-ID: 11.002

Title: Avoid reassembly of messages at multiple layers in the protocol stack.

Description: Reassembly of messages at multiple layers in the protocol stack requires buffers at multiple layers, which leads to inefficient use of memory resources. This can be avoided by making sure the application layer, the security layer, the transport layer, the IPv6 layer and any adaptation layers are aware of the limitations of each other such that unnecessary fragmentation and reassembly can be avoided. In addition, message size constraints must be announced to protocol peers such that they can adapt and avoid sending messages that can’t be processed due to resource constraints on the receiving device.

Source: Basic requirement which concerns all use cases with memory constrained devices.

Requirement Type: Non-Functional Requirement

Device type: C0, C1, and C2

Priority: High

4. IANA Considerations

This document does not introduce any new code-points or namespaces for registration with IANA.

Note to RFC Editor: this section may be removed on publication as an RFC.

5. Security Considerations

This document discusses the problem statement and requirements on networks of constrained devices. Section 1.6 mentions a number of limitations that could prevent the implementation of strong cryptographic algorithms. Requirements for security and access control are listed in Section 3.6.

6. Contributors

Ulrich Herberg (Fujitsu Laboratories of America) contributed to the Section 1.3 on Networks Types and Characteristics in Focus.

7. Acknowledgments

Following persons reviewed and provided valuable comments to different versions of this document:

Dominique Barthel, Andy Bierman, Carsten Bormann, Zhen Cao, Benoit Claise, Hui Deng, Bert Greevenbosch, Ulrich Herberg, James Nguyen, Anuj Sehgal, Zach Shelby, Peter van der Stok and Bert Wijnen.

The editors would like to thank the reviewers and the participants on the Coman and OPSAWG maillists for their valuable contributions and comments.

8. References

8.1. Normative References

8.2. Informative References

[RFC2914] Floyd, S., "Congestion Control Principles", BCP 41, RFC 2914, September 2000.

[RFC6632] Ersue, M. and B. Claise, "An Overview of the IETF Network Management Standards", RFC 6632, June 2012.

[RFC7102] Vasseur, JP., "Terms Used in Routing for Low-Power and Lossy Networks", RFC 7102, January 2014.

[I-D.ietf-lwig-terminology] Bormann, C., Ersue, M., and A. Keranen, "Terminology for Constrained Node Networks", draft-ietf-lwig-terminology-07 (work in progress), February 2014.

[I-D.ietf-core-coap] Shelby, Z., Hartke, K., and C. Bormann, "Constrained Application Protocol (CoAP)", draft-ietf-core-coap-18 (work in progress), June 2013.

[COM-USE] Ersue, M., "Constrained Management: Use Cases", draft-ietf-opsawg-coman-use-cases (work in progress), October 2013.

Appendix A. Change Log

A.1. draft-ietf-opsawg-coman-probstate-reqs-00 - draft-ietf-opsawg-coman-probstate-reqs-01

- General bug fixing.
- Added Section 1.7. on Configuration and Monitoring Functionality Levels.
- Changed diverse occurences of "networks" to "networks with/of constrained devices".
- Introduced the term "Self-configuring infrastructureless networks" instead of MANET as it is a superset.
- Introduced the term ’sleepy endpoints’.
- Changed requirement IDs to be independent of section number.
- Introduced notes for parts of the requirements text if it is focusing on implementation or solution.
- Extended Security Considerations section.
- Deleted Appendix A and B on other SDO’s work and related projects as they provided dynamic information and couldn’t be kept up-to-date.

A.2. draft-ersue-constrained-mgmt-03 - draft-ietf-opsawg-coman-probstate-reqs-00

- Reduced the terminology section for terminology addressed in the LWIG terminology draft. Referenced the LWIG terminology draft.
- Checked and aligned all terminology against the LWIG terminology draft.
- Moved section 1.4. Constrained Device Deployment Options and section 3. Use Cases to the companion document [COM-USE].
- Renamed Section 1.3. Class of Networks in Focus to "Network Types in Focus" and removed abbreviations C0, C1 and C2 for network classes as they have not been used.
- Changed requirement priority classes to be High, Medium and Low.

- Changed requirement types to be Functional and Non-Functional and added text to explain the requirement types.
- Reformulation of some text parts for more clarity.

A.3. draft-ersue-constrained-mgmt-02-03

- Extended the terminology section and removed some of the terminology addressed in the new LWIG terminology draft. Referenced the LWIG terminology draft.
- Moved Section 1.3. on Constrained Device Classes to the new LWIG terminology draft.
- Class of networks considering the different type of radio and communication technologies in use and dimensions extended.
- Extended the Problem Statement in Section 2. following the requirements listed in Section 4.
- Following requirements, which belong together and can be realized with similar or same kind of solutions, have been merged.
  - Distributed Management and Peer Configuration,
  - Device status monitoring and Neighbor-monitoring,
  - Passive Monitoring and Reactive Monitoring,
  - Event-driven self-management - Self-healing and Periodic self-management,
  - Authentication of management systems and Authentication of managed devices,
  - Access control on devices and Access control on management systems,
  - Management of Energy Resources and Data models for energy management,
  - Software distribution (group-based firmware update) and Group-based provisioning.
- Deleted the empty section on the gaps in network management standards, as it will be written in a separate draft.

- Added links to mentioned external pages.
- Added text on OMA M2M Device Classification in appendix.

A.4. draft-ersue-constrained-mgmt-01-02

- Extended the terminology section.
- Added additional text for the use cases concerning deployment type, network topology in use, network size, network capabilities, radio technology, etc.
- Added examples for device classes in a use case.
- Added additional text provided by Cao Zhen (China Mobile) for Mobile Applications and by Peter van der Stok for Building Automation.
- Added the new use cases ’Advanced Metering Infrastructure’ and ’MANET Concept of Operations in Military’.
- Added the section ’Managing the Constrainedness of a Device or Network’ discussing the needs of very constrained devices.
- Added a note that the requirements in Section 3 need to be seen as standalone requirements and the current document does not recommend any profile of requirements.
- Added Section 3 on the detailed requirements on constrained management matched to management tasks like fault, monitoring, configuration management, Security and Access Control, Energy Management, etc.
- Solved nits and added references.
- Added Appendix A on the related development in other bodies.
- Added Appendix B on the work in related research projects.

A.5. draft-ersue-constrained-mgmt-00-01

- Splitted the section on ’Networks of Constrained Devices’ into the sections ’Network Topology Options’ and ’Management Topology Options’.
- Added the use case ’Community Network Applications’ and ’Mobile Applications’.

- Provided a Contributors section.
- Extended the section on 'Medical Applications'.
- Solved nits and added references.

Authors' Addresses

Mehmet Ersue (editor)
Nokia Solutions and Networks

Email: mehmet.ersue@nsn.com

Dan Romascanu
Avaya

Email: dromasca@avaya.com

Juergen Schoenwaelder
Jacobs University Bremen

Email: j.schoenwaelder@jacobs-university.de

Internet Engineering Task Force
Internet-Draft
Intended status: Informational
Expires: August 18, 2014

M. Ersue, Ed.
Nokia Solutions and Networks
D. Romascanu
Avaya
J. Schoenwaelder
A. Sehgal
Jacobs University Bremen

February 14, 2014

Management of Networks with Constrained Devices: Use Cases draft-ietf-opsawg-coman-use-cases-01

Abstract

This document discusses the use cases concerning the management of networks, where constrained devices are involved. A problem statement, deployment options and the requirements on the networks with constrained devices can be found in the companion document on "Management of Networks with Constrained Devices: Problem Statement and Requirements".

Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on August 18, 2014.

Copyright Notice

Copyright (c) 2014 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (http://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents

carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction 3
2. Access Technologies 5
2.1. Constrained Access Technologies 5
2.2. Mobile Access Technologies 5
3. Use Cases 7
3.1. Environmental Monitoring 7
3.2. Infrastructure Monitoring 7
3.3. Industrial Applications 8
3.4. Energy Management 10
3.5. Medical Applications 12
3.6. Building Automation 13
3.7. Home Automation 15
3.8. Transport Applications 15
3.9. Vehicular Networks 17
3.10. Community Network Applications 18
3.11. Military Operations 19
4. IANA Considerations 21
5. Security Considerations 22
6. Contributors 23
7. Acknowledgments 24
8. Informative References 25
Appendix A. Open Issues 26
Appendix B. Change Log 27
B.1. draft-ietf-opsawg-coman-use-cases-00 - draft-ietf-opsawg-coman-use-cases-01 27
B.2. draft-ersue-constrained-mgmt-03 - draft-ersue-opsawg-coman-use-cases-00 27
B.3. draft-ersue-constrained-mgmt-02-03 27
B.4. draft-ersue-constrained-mgmt-01-02 28
B.5. draft-ersue-constrained-mgmt-00-01 29
Authors' Addresses 30

1. Introduction

Small devices with limited CPU, memory, and power resources, so called constrained devices (aka. sensor, smart object, or smart device) can be connected to a network. Such a network of constrained devices itself may be constrained or challenged, e.g., with unreliable or lossy channels, wireless technologies with limited bandwidth and a dynamic topology, needing the service of a gateway or proxy to connect to the Internet. In other scenarios, the constrained devices can be connected to a non-constrained network using off-the-shelf protocol stacks. Constrained devices might be in charge of gathering information in diverse settings including natural ecosystems, buildings, and factories and send the information to one or more server stations.

Network management is characterized by monitoring network status, detecting faults, and inferring their causes, setting network parameters, and carrying out actions to remove faults, maintain normal operation, and improve network efficiency and application performance. The traditional network management application periodically collects information from a set of elements that are needed to manage, processes the data, and presents them to the network management users. Constrained devices, however, often have limited power, low transmission range, and might be unreliable. They might also need to work in hostile environments with advanced security requirements or need to be used in harsh environments for a long time without supervision. Due to such constraints, the management of a network with constrained devices offers different type of challenges compared to the management of a traditional IP network.

This document aims to understand the use cases for the management of a network, where constrained devices are involved. The document lists and discusses diverse use cases for the management from the network as well as from the application point of view. The application scenarios discussed aim to show where networks of constrained devices are expected to be deployed. For each application scenario, we first briefly describe the characteristics followed by a discussion on how network management can be provided, who is likely going to be responsible for it, and on which time-scale management operations are likely to be carried out.

A problem statement, deployment and management topology options as well as the requirements on the networks with constrained devices can be found in the companion document [COM-REQ].

This documents builds on the terminology defined in [I-D.ietf-lwig-terminology] and [COM-REQ].

[I-D.ietf-lwig-terminology] is a base document for the terminology concerning constrained devices and constrained networks. Some use cases specific to IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs) can be found in [RFC6568].

2. Access Technologies

Besides the management requirements imposed by the different use cases, the access technologies used by constrained devices can impose restrictions and requirements upon the Network Management System (NMS) and protocol of choice.

It is possible that some networks of constrained devices might utilize traditional non-constrained access technologies for network access, e.g., local area networks with plenty of capacity. In such scenarios, the constrainedness of the device presents special management restrictions and requirements rather than the access technology utilized.

However, in other situations constrained or mobile access technologies might be used for network access, thereby causing management restrictions and requirements to arise as a result of the underlying access technologies.

2.1. Constrained Access Technologies

Due to resource restrictions, embedded devices deployed as sensors and actuators in the various use cases utilize low-power low data-rate wireless access technologies such as IEEE 802.15.4, DECT ULE or BT-LE for network connectivity.

In such scenarios, it is important for the NMS to be aware of the restrictions imposed by these access technologies to efficiently manage these constrained devices. Specifically, such low-power low data-rate access technologies typically have small frame sizes. So it would be important for the NMS and management protocol of choice to craft packets in a way that avoids fragmentation and reassembly of packets since this can use valuable memory on constrained devices.

Devices using such access technologies might operate via a gateway that translates between these access technologies and more traditional Internet protocols. A hierarchical approach to device management in such a situation might be useful, wherein the gateway device is in-charge of devices connected to it, while the NMS conducts management operations only to the gateway.

2.2. Mobile Access Technologies

Machine to machine (M2M) services are increasingly provided by mobile service providers as numerous devices, home appliances, utility meters, cars, video surveillance cameras, and health monitors, are connected with mobile broadband technologies. Different applications, e.g., in a home appliance or in-car network, use

Bluetooth, Wi-Fi or Zigbee locally and connect to a cellular module acting as a gateway between the constrained environment and the mobile cellular network.

Such a gateway might provide different options for the connectivity of mobile networks and constrained devices:

- a smart phone with 3G/4G and WLAN radio might use BT-LE to connect to the devices in a home area network,
- a femtocell might be combined with home gateway functionality acting as a low-power cellular base station connecting smart devices to the application server of a mobile service provider,
- an embedded cellular module with LTE radio connecting the devices in the car network with the server running the telematics service,
- an M2M gateway connected to the mobile operator network supporting diverse IoT connectivity technologies including ZigBee and CoAP over 6LoWPAN over IEEE 802.15.4.

Common to all scenarios above is that they are embedded in a service and connected to a network provided by a mobile service provider. Usually there is a hierarchical deployment and management topology in place where different parts of the network are managed by different management entities and the count of devices to manage is high (e.g. many thousands). In general, the network is comprised by manifold type and size of devices matching to different device classes. As such, the managing entity needs to be prepared to manage devices with diverse capabilities using different communication or management protocols. In case the devices are directly connected to a gateway they most likely are managed by a management entity integrated with the gateway, which itself is part of the Network Management System (NMS) run by the mobile operator. Smart phones or embedded modules connected to a gateway might be themselves in charge to manage the devices on their level. The initial and subsequent configuration of such a device is mainly based on self-configuration and is triggered by the device itself.

The gateway might be in charge of filtering and aggregating the data received from the device as the information sent by the device might be mostly redundant.

3. Use Cases

3.1. Environmental Monitoring

Environmental monitoring applications are characterized by the deployment of a number of sensors to monitor emissions, water quality, or even the movements and habits of wildlife. Other applications in this category include earthquake or tsunami early-warning systems. The sensors often span a large geographic area, they can be mobile, and they are often difficult to replace. Furthermore, the sensors are usually not protected against tampering.

Management of environmental monitoring applications is largely concerned with the monitoring whether the system is still functional and the roll-out of new constrained devices in case the system looses too much of its structure. The constrained devices themselves need to be able to establish connectivity (auto-configuration) and they need to be able to deal with events such as loosing neighbors or being moved to other locations.

Management responsibility typically rests with the organization running the environmental monitoring application. Since these monitoring applications must be designed to tolerate a number of failures, the time scale for detecting and recording failures is for some of these applications likely measured in hours and repairs might easily take days. However, for certain environmental monitoring applications, much tighter time scales may exist and might be enforced by regulations (e.g., monitoring of nuclear radiation).

3.2. Infrastructure Monitoring

Infrastructure monitoring is concerned with the monitoring of infrastructures such as bridges, railway tracks, or (offshore) windmills. The primary goal is usually to detect any events or changes of the structural conditions that can impact the risk and safety of the infrastructure being monitored. Another secondary goal is to schedule repair and maintenance activities in a cost effective manner.

The infrastructure to monitor might be in a factory or spread over a wider area but difficult to access. As such, the network in use might be based on a combination of fixed and wireless technologies, which use robust networking equipment and support reliable communication. It is likely that constrained devices in such a network are mainly C2 devices and have to be controlled centrally by an application running on a server. In case such a distributed network is widely spread, the wireless devices might use diverse long-distance wireless technologies such as WiMAX, or 3G/LTE, e.g.

based on embedded hardware modules. In cases, where an in-building network is involved, the network can be based on Ethernet or wireless technologies suitable for in-building usage.

The management of infrastructure monitoring applications is primarily concerned with the monitoring of the functioning of the system. Infrastructure monitoring devices are typically rolled out and installed by dedicated experts and changes are rare since the infrastructure itself changes rarely. However, monitoring devices are often deployed in unsupervised environments and hence special attention must be given to protecting the devices from being modified.

Management responsibility typically rests with the organization owning the infrastructure or responsible for its operation. The time scale for detecting and recording failures is likely measured in hours and repairs might easily take days. However, certain events (e.g., natural disasters) may require that status information be obtained much more quickly and that replacements of failed sensors can be rolled out quickly (or redundant sensors are activated quickly). In case the devices are difficult to access, a self-healing feature on the device might become necessary.

3.3. Industrial Applications

Industrial Applications and smart manufacturing refer to tasks such as networked control and monitoring of manufacturing equipment, asset and situation management, or manufacturing process control. For the management of a factory it is becoming essential to implement smart capabilities. From an engineering standpoint, industrial applications are intelligent systems enabling rapid manufacturing of new products, dynamic response to product demands, and real-time optimization of manufacturing production and supply chain networks. Potential industrial applications (e.g., for smart factories and smart manufacturing) are:

- Digital control systems with embedded, automated process controls, operator tools, as well as service information systems optimizing plant operations and safety.

- Asset management using predictive maintenance tools, statistical evaluation, and measurements maximizing plant reliability.

- Smart sensors detecting anomalies to avoid abnormal or catastrophic events.

- Smart systems integrated within the industrial energy management system and externally with the smart grid enabling real-time

energy optimization.

Management of Industrial Applications and smart manufacturing may in some situations involve Building Automation tasks such as control of energy, HVAC (heating, ventilation, and air conditioning), lighting, or access control. Interacting with management systems from other application areas might be important in some cases (e.g., environmental monitoring for electric energy production, energy management for dynamically scaling manufacturing, vehicular networks for mobile asset tracking).

Sensor networks are an essential technology used for smart manufacturing. Measurements, automated controls, plant optimization, health and safety management, and other functions are provided by a large number of networked sectors. Data interoperability and seamless exchange of product, process, and project data are enabled through interoperable data systems used by collaborating divisions or business systems. Intelligent automation and learning systems are vital to smart manufacturing but must be effectively integrated with the decision environment. Wireless sensor networks (WSN) have been developed for machinery Condition-based Maintenance (CBM) as they offer significant cost savings and enable new functionalities. Inaccessible locations, rotating machinery, hazardous areas, and mobile assets can be reached with wireless sensors. WSNs can provide today wireless link reliability, real-time capabilities, and quality-of-service and enable industrial and related wireless sense and control applications.

Management of industrial and factory applications is largely focused on the monitoring whether the system is still functional, real-time continuous performance monitoring, and optimization as necessary. The factory network might be part of a campus network or connected to the Internet. The constrained devices in such a network need to be able to establish configuration themselves (auto-configuration) and might need to deal with error conditions as much as possible locally. Access control has to be provided with multi-level administrative access and security. Support and diagnostics can be provided through remote monitoring access centralized outside of the factory.

Management responsibility is typically owned by the organization running the industrial application. Since the monitoring applications must handle a potentially large number of failures, the time scale for detecting and recording failures is for some of these applications likely measured in minutes. However, for certain industrial applications, much tighter time scales may exist, e.g. in real-time, which might be enforced by the manufacturing process or the use of critical material.

3.4. Energy Management

The EMAN working group developed an energy management framework [I-D.ietf-eman-framework] for devices and device components within or connected to communication networks. This document observes that one of the challenges of energy management is that a power distribution network is responsible for the supply of energy to various devices and components, while a separate communication network is typically used to monitor and control the power distribution network. Devices that have energy management capability are defined as Energy Devices and identified components within a device (Energy Device Components) can be monitored for parameters like Power, Energy, Demand and Power Quality. If a device contains batteries, they can be also monitored and managed.

Energy devices differ in complexity and may include basic sensors or switches, specialized electrical meters, or power distribution units (PDU), and subsystems inside the network devices (routers, network switches) or home or industrial appliances. An Energy Management System is a combination of hardware and software used to administer a network with the primary purpose being Energy Management. The operators of such a system are either the utility providers or customers that aim to control and reduce the energy consumption and the associated costs. The topology in use differs and the deployment can cover areas from small surfaces (individual homes) to large geographical areas. The EMAN requirements document [RFC6988] discusses the requirements for energy management concerning monitoring and control functions.

It is assumed that Energy Management will apply to a large range of devices of all classes and networks topologies. Specific resource monitoring like battery utilization and availability may be specific to devices with lower physical resources (device classes C0 or C1).

Energy Management is especially relevant to the Smart Grid. A Smart Grid is an electrical grid that uses data networks to gather and to act on energy and power-related information in an automated fashion with the goal to improve the efficiency, reliability, economics, and sustainability of the production and distribution of electricity. A Smart Grid provides sustainable and reliable generation, transmission, distribution, storage and consumption of electrical energy based on advanced energy and information technology. Smart Grids enable the following specific application areas: Smart transmission systems, Demand Response/Load Management, Substation Automation, Advanced Distribution Management, Advanced Metering Infrastructure (AMI), Smart Metering, Smart Home and Building Automation, E-mobility, etc.

Smart Metering is a good example of Smart Grid based Energy Management applications. Different types of possibly wireless small meters produce all together a large amount of data, which is collected by a central entity and processed by an application server, which may be located within the customer's residence or off-site in a data-center. The communication infrastructure can be provided by a mobile network operator as the meters in urban areas will have most likely a cellular or WiMAX radio. In case the application server is located within the residence, such meters are more likely to use WiFi protocols to interconnect with an existing network.

An AMI network is another example of the Smart Grid that enables an electric utility to retrieve frequent electric usage data from each electric meter installed at a customer's home or business. This is unlike Smart Metering, in which case the customer or their agents install appliance level meters, because an AMI infrastructure is typically managed by the utility providers. With an AMI network, a utility can also receive immediate notification of power outages when they occur, directly from the electric meters that are experiencing those outages. In addition, if the AMI network is designed to be open and extensible, it could serve as the backbone for communicating with other distribution automation devices besides meters, which could include transformers and reclosers.

Each meter in the AMI network typically contains constrained devices of the C2 type. Each meter uses the constrained devices to connect to mesh networks with a low-bandwidth radio. These radios can be 50, 150, or 200 kbps at raw link speed, but actual network throughput may be significantly lower due to forward error correction, multihop delays, MAC delays, lossy links, and protocol overhead. Usage data and outage notifications can be sent by these meters to the utility's headend systems, typically located in a data center managed by the utility, which include meter data collection systems, meter data management systems, and outage management systems.

Meters in an AMI network, unlike in Smart Metering, act as traffic sources and routers as well. Typically, smaller amounts of traffic (read requests, configuration) flow "downstream" from the headend to the mesh, and larger amounts of traffic flow "upstream" from the mesh to the headend. However, during a firmware update operation for example, larger amounts of traffic might flow downstream while smaller amounts flow upstream. The mesh network is anchored by a collection of higher-end devices that bridge the constrained network with a backhaul link that connects to a less-constrained network via cellular, WiMAX, or Ethernet. These higher-end devices might be installed on utility poles that could be owned and managed by a different entity than the utility company.

While a Smart Metering solution is likely to have a smaller number of devices within a single household, AMI network installations could contain 1000 meters per router, i.e., the higher-end device. Meters in a local network that use a specific router form a Local Meter Network (LMN). When powered on, meters discover nearby LMNs, select the optimal LMN to join, and the meters in that LMN to route through. However, in a Smart Metering application the meters are likely to connect directly to a less-constrained network, thereby not needing to form such local mesh networks.

Encryption key sharing in both types of network is also likely to be important for providing confidentiality for all data traffic. In AMI networks the key may be obtained by a meter only after an end-to-end authentication process based on certificates, ensuring that only authorized and authenticated meters are allowed to join the LMN. Smart Metering solution could adopt a similar approach or the security may be implied due to the encrypted WiFi networks they become part of.

These examples demonstrate that the Smart Grid, and Energy Management in general, is built on a distributed and heterogeneous network and can use a combination of diverse networking technologies, such as wireless Access Technologies (WiMAX, Cellular, etc.), wireline and Internet Technologies (e.g., IP/MPLS, Ethernet, SDH/PDH over Fiber optic) as well as low-power radio technologies enabling the networking of smart meters, home appliances, and constrained devices (e.g., BT-LE, ZigBee, Z-Wave, Wi-Fi). The operational effectiveness of the Smart Grid is highly dependent on a robust, two-way, secure, and reliable communications network with suitable availability.

The management of such a network requires end-to-end management of and information exchange through different types of networks. However, as of today there is no integrated energy management approach and no common information model available. Specific energy management applications or network islands use their own management mechanisms.

3.5. Medical Applications

Constrained devices can be seen as an enabling technology for advanced and possibly remote health monitoring and emergency notification systems, ranging from blood pressure and heart rate monitors to advanced devices capable to monitor implanted technologies, such as pacemakers or advanced hearing aids. Medical sensors may not only be attached to human bodies, they might also exist in the infrastructure used by humans such as bathrooms or kitchens. Medical applications will also be used to ensure treatments are being applied properly and they might guide people

losing orientation. Fitness and wellness applications, such as connected scales or wearable heart monitors, encourage consumers to exercise and empower self-monitoring of key fitness indicators. Different applications use Bluetooth, Wi-Fi or Zigbee connections to access the patient's smartphone or home cellular connection to access the Internet.

Constrained devices that are part of medical applications are managed either by the users of those devices or by an organization providing medical (monitoring) services for physicians. In the first case, management must be automatic and or easy to install and setup by average people. In the second case, it can be expected that devices be controlled by specially trained people. In both cases, however, it is crucial to protect the privacy of the people to which medical devices are attached. Even though the data collected by a heart beat monitor might be protected, the pure fact that someone carries such a device may need protection. As such, certain medical appliances may not want to participate in discovery and self-configuration protocols in order to remain invisible.

Many medical devices are likely to be used (and relied upon) to provide data to physicians in critical situations since the biggest market is likely elderly and handicapped people. As such, fault detection of the communication network or the constrained devices becomes a crucial function that must be carried out with high reliability and, depending on the medical appliance and its application, within seconds.

3.6. Building Automation

Building automation comprises the distributed systems designed and deployed to monitor and control the mechanical, electrical and electronic systems inside buildings with various destinations (e.g., public and private, industrial, institutions, or residential). Advanced Building Automation Systems (BAS) may be deployed concentrating the various functions of safety, environmental control, occupancy, security. More and more the deployment of the various functional systems is connected to the same communication infrastructure (possibly Internet Protocol based), which may involve wired or wireless communications networks inside the building.

Building automation requires the deployment of a large number (10-100.000) of sensors that monitor the status of devices, and parameters inside the building and controllers with different specialized functionality for areas within the building or the totality of the building. Inter-node distances between neighboring nodes vary between 1 to 20 meters. Contrary to home automation, in building management the devices are expected to be managed assets and

known to a set of commissioning tools and a data storage, such that every connected device has a known origin. The management includes verifying the presence of the expected devices and detecting the presence of unwanted devices.

Examples of functions performed by such controllers are regulating the quality, humidity, and temperature of the air inside the building and lighting. Other systems may report the status of the machinery inside the building like elevators, or inside the rooms like projectors in meeting rooms. Security cameras and sensors may be deployed and operated on separate dedicated infrastructures connected to the common backbone. The deployment area of a BAS is typically inside one building (or part of it) or several buildings geographically grouped in a campus. A building network can be composed of subnets, where a subnet covers a floor, an area on the floor, or a given functionality (e.g., security cameras).

Some of the sensors in Building Automation Systems (for example fire alarms or security systems) register, record and transfer critical alarm information and therefore must be resilient to events like loss of power or security attacks. This leads to the need that some components and subsystems operate in constrained conditions and are separately certified. Also in some environments, the malfunctioning of a control system (like temperature control) needs to be reported in the shortest possible time. Complex control systems can misbehave, and their critical status reporting and safety algorithms need to be basic and robust and perform even in critical conditions.

Building Automation solutions are deployed in some cases in newly designed buildings, in other cases it might be over existing infrastructures. In the first case, there is a broader range of possible solutions, which can be planned for the infrastructure of the building. In the second case the solution needs to be deployed over an existing structure taking into account factors like existing wiring, distance limitations, the propagation of radio signals over walls and floors. As a result, some of the existing WLAN solutions (e.g., IEEE 802.11 or IEEE 802.15) may be deployed. In mission-critical or security sensitive environments and in cases where link failures happen often, topologies that allow for reconfiguration of the network and connection continuity may be required. Some of the sensors deployed in building automation may be very simple constrained devices for which class 0 or class 1 may be assumed.

For lighting applications, groups of lights must be defined and managed. Commands to a group of light must arrive within 200 ms at all destinations. The installation and operation of a building network has different requirements. During the installation, many stand-alone networks of a few to 100 nodes co-exist without a

connection to the backbone. During this phase, the nodes are identified with a network identifier related to their physical location. Devices are accessed from an installation tool to connect them to the network in a secure fashion. During installation, the setting of parameters to common values to enable interoperability may occur (e.g., Trickle parameter values). During operation, the networks are connected to the backbone while maintaining the network identifier to physical location relation. Network parameters like address and name are stored in DNS. The names can assist in determining the physical location of the device.

3.7. Home Automation

Home automation includes the control of lighting, heating, ventilation, air conditioning, appliances, entertainment and home security devices to improve convenience, comfort, energy efficiency, and security. It can be seen as a residential extension of building automation. However, unlike a building automation system, the infrastructure in a home is operated in a considerably more ad-hoc manner, with no centralized management system akin to a Building Automation System (BAS) available.

Home automation networks need a certain amount of configuration (associating switches or sensors to actors) that is either provided by electricians deploying home automation solutions, by third party home automation service providers (e.g., small specialized companies or home automation device manufacturers) or by residents by using the application user interface provided by home automation devices to configure (parts of) the home automation solution. Similarly, failures may be reported via suitable interfaces to residents or they might be recorded and made available to services providers in charge of the maintenance of the home automation infrastructure.

The management responsibility lies either with the residents or it may be outsourced to electricians and/or third parties providing management of home automation solutions as a service. A varying combination of electricians, service providers or the residents may be responsible for different aspects of managing the infrastructure. The time scale for failure detection and resolution is in many cases likely counted in hours to days.

3.8. Transport Applications

Transport Application is a generic term for the integrated application of communications, control, and information processing in a transportation system. Transport telematics or vehicle telematics are used as a term for the group of technologies that support transportation systems. Transport applications running on such a

transportation system cover all modes of the transport and consider all elements of the transportation system, i.e. the vehicle, the infrastructure, and the driver or user, interacting together dynamically. The overall aim is to improve decision making, often in real time, by transport network controllers and other users, thereby improving the operation of the entire transport system. As such, transport applications can be seen as one of the important M2M service scenarios with the involvement of manifold small devices.

The definition encompasses a broad array of techniques and approaches that may be achieved through stand-alone technological applications or as enhancements to other transportation communication schemes. Examples for transport applications are inter and intra vehicular communication, smart traffic control, smart parking, electronic toll collection systems, logistic and fleet management, vehicle control, and safety and road assistance.

As a distributed system, transport applications require an end-to-end management of different types of networks. It is likely that constrained devices in a network (e.g. a moving in-car network) have to be controlled by an application running on an application server in the network of a service provider. Such a highly distributed network including mobile devices on vehicles is assumed to include a wireless access network using diverse long distance wireless technologies such as WiMAX, 3G/LTE or satellite communication, e.g. based on an embedded hardware module. As a result, the management of constrained devices in the transport system might be necessary to plan top-down and might need to use data models obliged from and defined on the application layer. The assumed device classes in use are mainly C2 devices. In cases, where an in-vehicle network is involved, C1 devices with limited capabilities and a short-distance constrained radio network, e.g. IEEE 802.15.4 might be used additionally.

Management responsibility typically rests within the organization running the transport application. The constrained devices in a moving transport network might be initially configured in a factory and a reconfiguration might be needed only rarely. New devices might be integrated in an ad-hoc manner based on self-management and -configuration capabilities. Monitoring and data exchange might be necessary to do via a gateway entity connected to the back-end transport infrastructure. The devices and entities in the transport infrastructure need to be monitored more frequently and can be able to communicate with a higher data rate. The connectivity of such entities does not necessarily need to be wireless. The time scale for detecting and recording failures in a moving transport network is likely measured in hours and repairs might easily take days. It is likely that a self-healing feature would be used locally.

3.9. Vehicular Networks

Networks involving mobile nodes, especially transport vehicles, are emerging. Such networks are used to provide inter-vehicle communication services, or even tracking of mobile assets, to develop intelligent transportation systems and drivers and passengers assistance services. Constrained devices are deployed within a larger single entity, the vehicle, and must be individually managed.

Vehicles can be either private, belonging to individuals or private companies, or public transportation. Scenarios consisting of vehicle-to-vehicle ad-hoc networks, a wired backbone with wireless last hops, and hybrid vehicle-to-road communications are expected to be common.

Besides the access control and security, depending on the type of vehicle and service being provided, it would be important for a NMS to be able to function with different architectures, since different manufacturers might have their own proprietary systems.

Unlike some mobile networks, most vehicular networks are expected to have specific patterns in the mobility of the nodes. Such patterns could possibly be exploited, managed and monitored by the NMS.

The challenges in the management of vehicles in a mobile application are manifold. Firstly, the issues caused through the device mobility need to be taken into consideration. The up-to-date position of each node in the network should be reported to the corresponding management entities, since the nodes could be moving within or roaming between different networks. Secondly, a variety of troubleshooting information, including sensitive location information, needs to be reported to the management system in order to provide accurate service to the customer.

The NMS must also be able to handle partitioned networks, which would arise due to the dynamic nature of traffic resulting in large inter-vehicle gaps in sparsely populated scenarios. Constant changes in topology must also be contended with.

Auto-configuration of nodes in a vehicular network remains a challenge since based on location, and access network, the vehicle might have different configurations that must be obtained from its management system. Operating configuration updates, while in remote networks also needs to be considered in the design of a network management system."

3.10. Community Network Applications

Community networks are comprised of constrained routers in a multi-hop mesh topology, communicating over a lossy, and often wireless channel. While the routers are mostly non-mobile, the topology may be very dynamic because of fluctuations in link quality of the (wireless) channel caused by, e.g., obstacles, or other nearby radio transmissions. Depending on the routers that are used in the community network, the resources of the routers (memory, CPU) may be more or less constrained - available resources may range from only a few kilobytes of RAM to several megabytes or more, and CPUs may be small and embedded, or more powerful general-purpose processors. Examples of such community networks are the FunkFeuer network (Vienna, Austria), FreiFunk (Berlin, Germany), Seattle Wireless (Seattle, USA), and AWMN (Athens, Greece). These community networks are public and non-regulated, allowing their users to connect to each other and - through an uplink to an ISP - to the Internet. No fee, other than the initial purchase of a wireless router, is charged for these services. Applications of these community networks can be diverse, e.g., location based services, free Internet access, file sharing between users, distributed chat services, social networking etc, video sharing etc.

As an example of a community network, the FunkFeuer network comprises several hundred routers, many of which have several radio interfaces (with omnidirectional and some directed antennas). The routers of the network are small-sized wireless routers, such as the Linksys WRT54GL, available in 2011 for less than 50 Euros. These routers, with 16 MB of RAM and 264 MHz of CPU power, are mounted on the rooftops of the users. When new users want to connect to the network, they acquire a wireless router, install the appropriate firmware and routing protocol, and mount the router on the rooftop. IP addresses for the router are assigned manually from a list of addresses (because of the lack of autoconfiguration standards for mesh networks in the IETF).

While the routers are non-mobile, fluctuations in link quality require an ad hoc routing protocol that allows for quick convergence to reflect the effective topology of the network (such as NHDP [RFC6130] and OLSRv2 [I-D.ietf-manet-olsrv2] developed in the MANET WG). Usually, no human interaction is required for these protocols, as all variable parameters required by the routing protocol are either negotiated in the control traffic exchange, or are only of local importance to each router (i.e. do not influence interoperability). However, external management and monitoring of an ad hoc routing protocol may be desirable to optimize parameters of the routing protocol. Such an optimization may lead to a more stable perceived topology and to a lower control traffic overhead, and

therefore to a higher delivery success ratio of data packets, a lower end-to-end delay, and less unnecessary bandwidth and energy usage.

Different use cases for the management of community networks are possible:

- One single Network Management Station, e.g. a border gateway providing connectivity to the Internet, requires managing or monitoring routers in the community network, in order to investigate problems (monitoring) or to improve performance by changing parameters (managing). As the topology of the network is dynamic, constant connectivity of each router towards the management station cannot be guaranteed. Current network management protocols, such as SNMP and Netconf, may be used (e.g., using interfaces such as the NHDP-MIB [RFC6779]). However, when routers in the community network are constrained, existing protocols may require too many resources in terms of memory and CPU; and more importantly, the bandwidth requirements may exceed the available channel capacity in wireless mesh networks. Moreover, management and monitoring may be unfeasible if the connection between the network management station and the routers is frequently interrupted.

- A distributed network monitoring, in which more than one management station monitors or manages other routers. Because connectivity to a server cannot be guaranteed at all times, a distributed approach may provide a higher reliability, at the cost of increased complexity. Currently, no IETF standard exists for distributed monitoring and management.

- Monitoring and management of a whole network or a group of routers. Monitoring the performance of a community network may require more information than what can be acquired from a single router using a network management protocol. Statistics, such as topology changes over time, data throughput along certain routing paths, congestion etc., are of interest for a group of routers (or the routing domain) as a whole. As of 2012, no IETF standard allows for monitoring or managing whole networks, instead of single routers.

3.11. Military Operations

The challenges of configuration and monitoring of networks faced by military agencies can be different from the other use cases since the requirements and operating conditions of military networks are quite different.

With technology advancements, military networks nowadays have become

large and consist of varieties of different types of equipment that run different protocols and tools that obviously increase complexity of the tactical networks. In many scenarios, configurations are, most likely, manually performed. Furthermore, some legacy and even modern devices do not even support IP networking. Majority of protocols and tools developed by vendors that are being used are proprietary which makes integration more difficult.

The main reason for this disjoint operation scenario is that most military equipment is developed with specific tasks requirements in mind, rather than interoperability of the varied equipment types. For example, the operating conditions experienced by high altitude equipment is significantly different from that used in desert conditions and interoperation of tactical equipment with telecommunication equipment was not an expected outcome.

Currently, most military networks operate with a fixed Network Operations Center (NOC) that physically manages the configuration and evaluation of all field devices. Once configured, the devices might be deployed in fixed or mobile scenarios. Any configuration changes required would need to be appropriately encrypted and authenticated to prevent unauthorized access.

Hierarchical management of devices is a common requirement of military operations as well since local managers may need to respond to changing conditions within their platoon, regiment, brigade, division or corps. The level of configuration management available at each hierarchy must also be closely governed.

Since most military networks operate in hostile environments, a high failure rate and disconnection rate should be tolerated by the NMS, which must also be able to deal with multiple gateways and disjoint management protocols.

Multi-national military operations are becoming increasingly common, requiring the interoperation of a diverse set of equipment designed with different operating conditions in mind. Furthermore, different militaries are likely to have a different set of standards, best practices, rules and regulation, and implementation approaches that may contradict or conflict with each other. The NMS should be able to detect these and handle them in an acceptable manner, which may require human intervention.

4. IANA Considerations

This document does not introduce any new code-points or namespaces for registration with IANA.

Note to RFC Editor: this section may be removed on publication as an RFC.

5. Security Considerations

In several use cases, constrained devices are deployed in unsafe environments, where attackers can gain physical access to the devices. As a consequence, it is crucial to properly protect any security credentials that may be stored on the device (e.g., by using hardware protection mechanisms). Furthermore, it is important that any credentials leeking from a single device do not simplify the attack on other (similar) devices. In particular, security credentials should never be shared.

Since constrained devices often have limited computational resources, care should be taken in choosing efficient but cryptographically strong crytographic algorithms. Designers of constrained devices that have a long expected lifetime need to ensure that cryptographic algorithms can be updated once devices have been deployed. The ability to perform secure firmware and software updates is an important management requirement.

Several use cases generate sensitive data or require the processing of sensitive data. It is therefore an important requirement to properly protect access to the data in order to protect the privacy of humans using Internet-enabled devices. For certain types of data, protection during the transmission over the network may not be sufficient and methods should be investigated that provide protection of data while it is cached or stored (e.g., when using a store-and-forward transport mechanism).

6. Contributors

Following persons made significant contributions to and reviewed this document:

- Ulrich Herberg (Fujitsu Laboratories of America) contributed the Section 3.10 on Community Network Applications.
- Peter van der Stok contributed to Section 3.6 on Building Automation.
- Zhen Cao contributed to Section 2.2 Mobile Access Technologies.
- Gilman Tolle contributed the Section 3.4 on Automated Metering Infrastructure.
- James Nguyen and Ulrich Herberg contributed to Section 3.11 on Military operations.

7. Acknowledgments

Following persons reviewed and provided valuable comments to different versions of this document:

Dominique Barthel, Carsten Bormann, Zhen Cao, Benoit Claise, Bert Greevenbosch, Ulrich Herberg, James Nguyen, Zach Shelby, and Peter van der Stok.

The editors would like to thank the reviewers and the participants on the Coman maillist for their valuable contributions and comments.

8. Informative References

[RFC6130] Clausen, T., Dearlove, C., and J. Dean, "Mobile Ad Hoc Network (MANET) Neighborhood Discovery Protocol (NHDP)", RFC 6130, April 2011.

[RFC6568] Kim, E., Kaspar, D., and JP. Vasseur, "Design and Application Spaces for IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs)", RFC 6568, April 2012.

[RFC6779] Herberg, U., Cole, R., and I. Chakeres, "Definition of Managed Objects for the Neighborhood Discovery Protocol", RFC 6779, October 2012.

[RFC6988] Quittek, J., Chandramouli, M., Winter, R., Dietz, T., and B. Claise, "Requirements for Energy Management", RFC 6988, September 2013.

[I-D.ietf-lwig-terminology] Bormann, C., Ersue, M., and A. Keranen, "Terminology for Constrained Node Networks", draft-ietf-lwig-terminology-07 (work in progress), February 2014.

[I-D.ietf-eman-framework] Claise, B., Schoening, B., and J. Quittek, "Energy Management Framework", draft-ietf-eman-framework-15 (work in progress), February 2014.

[I-D.ietf-manet-olsrv2] Clausen, T., Dearlove, C., Jacquet, P., and U. Herberg, "The Optimized Link State Routing Protocol version 2", draft-ietf-manet-olsrv2-19 (work in progress), March 2013.

[COM-REQ] Ersue, M., "Constrained Management: Problem statement and Requirements", draft-ietf-opsawg-coman-probstate-reqs (work in progress), January 2014.

Appendix A. Open Issues

- Section 3.11 should be replaced by a different use case motivating similar requirements or perhaps deleted if the IETF prefers to not work on specific requirements coming from military use cases.

- Section 3.8 and Section 3.9 should be merged.

Appendix B. Change Log

B.1. draft-ietf-opsawg-coman-use-cases-00 - draft-ietf-opsawg-coman-use-cases-01

- Reordered some use cases to improve the flow.
- Added "Vehicular Networks".
- Shortened the Military Operations use case.
- Started adding substance to the security considerations section.

B.2. draft-ersue-constrained-mgmt-03 - draft-ersue-opsawg-coman-use-cases-00

- Reduced the terminology section for terminology addressed in the LWIG and Coman Requirements drafts. Referenced the other drafts.
- Checked and aligned all terminology against the LWIG terminology draft.
- Spent some effort to resolve the intersection between the Industrial Application, Home Automation and Building Automation use cases.
- Moved section section 3. Use Cases from the companion document [COM-REQ] to this draft.
- Reformulation of some text parts for more clarity.

B.3. draft-ersue-constrained-mgmt-02-03

- Extended the terminology section and removed some of the terminology addressed in the new LWIG terminology draft. Referenced the LWIG terminology draft.
- Moved Section 1.3. on Constrained Device Classes to the new LWIG terminology draft.
- Class of networks considering the different type of radio and communication technologies in use and dimensions extended.
- Extended the Problem Statement in Section 2. following the requirements listed in Section 4.
- Following requirements, which belong together and can be realized with similar or same kind of solutions, have been merged.

  * Distributed Management and Peer Configuration,

  * Device status monitoring and Neighbor-monitoring,

  * Passive Monitoring and Reactive Monitoring,

  * Event-driven self-management - Self-healing and Periodic self-management,

  * Authentication of management systems and Authentication of managed devices,

  * Access control on devices and Access control on management systems,

  * Management of Energy Resources and Data models for energy management,

  * Software distribution (group-based firmware update) and Group-based provisioning.

- Deleted the empty section on the gaps in network management standards, as it will be written in a separate draft.

- Added links to mentioned external pages.

- Added text on OMA M2M Device Classification in appendix.

B.4. draft-ersue-constrained-mgmt-01-02

- Extended the terminology section.

- Added additional text for the use cases concerning deployment type, network topology in use, network size, network capabilities, radio technology, etc.

- Added examples for device classes in a use case.

- Added additional text provided by Cao Zhen (China Mobile) for Mobile Applications and by Peter van der Stok for Building Automation.

- Added the new use cases ’Advanced Metering Infrastructure’ and ’MANET Concept of Operations in Military’.

- Added the section ’Managing the Constrainedness of a Device or Network’ discussing the needs of very constrained devices.

- Added a note that the requirements in [COM-REQ] need to be seen as standalone requirements and the current document does not recommend any profile of requirements.

- Added a section in [COM-REQ] for the detailed requirements on constrained management matched to management tasks like fault, monitoring, configuration management, Security and Access Control, Energy Management, etc.

- Solved nits and added references.

- Added Appendix A on the related development in other bodies.

- Added Appendix B on the work in related research projects.

B.5. draft-ersue-constrained-mgmt-00-01

- Splitted the section on 'Networks of Constrained Devices' into the sections 'Network Topology Options' and 'Management Topology Options'.

- Added the use case 'Community Network Applications' and 'Mobile Applications'.

- Provided a Contributors section.

- Extended the section on 'Medical Applications'.

- Solved nits and added references.

Authors' Addresses

Mehmet Ersue (editor)
Nokia Solutions and Networks

Email: mehmet.ersue@nsn.com

Dan Romascanu
Avaya

Email: dromasca@avaya.com

Juergen Schoenwaelder
Jacobs University Bremen

Email: j.schoenwaelder@jacobs-university.de

Anuj Sehgal
Jacobs University Bremen

Email: a.sehgal@jacobs-university.de

OPSAWG
Internet-Draft
Intended status: Standards Track
Expires: August 14, 2014

H. Asai
Univ. of Tokyo
M. MacFaden
VMware Inc.
J. Schoenwaelder
Jacobs University
K. Shima
IIJ Innovation Institute Inc.
T. Tsou
Huawei Technologies (USA)

February 10, 2014

Management Information Base for Virtual Machines Controlled by a Hypervisor
draft-ietf-opsawg-vmm-mib-00

Abstract

This document defines a portion of the Management Information Base (MIB) for use with network management protocols in the Internet community. In particular, this specifies objects for managing virtual machines controlled by a hypervisor (a.k.a. virtual machine monitor).

Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on August 14, 2014.

Copyright Notice

Copyright (c) 2014 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust’s Legal

Provisions Relating to IETF Documents
(http://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction 3
2. The Internet-Standard Management Framework 4
3. Overview and Objectives 5
4. Structure of the VM-MIB Module 7
5. Relationship to Other MIB Modules 12
6. Definitions 13
7. IANA Considerations 49
8. Security Considerations 50
9. Acknowledgements 52
10. References 53
10.1. Normative References 53
10.2. Informative References 54
Appendix A. State Transition Table 55
Authors' Addresses 57

1. Introduction

This document defines a portion of the Management Information Base (MIB) for use with network management protocols in the Internet community. In particular, this specifies objects for managing virtual machines controlled by a hypervisor (a.k.a. virtual machine monitor). A hypervisor controls multiple virtual machines on a single physical machine by allocating resources to each virtual machine using virtualization technologies. Therefore, this MIB module contains information on virtual machines and their resources controlled by a hypervisor as well as hypervisor’s hardware and software information.

The design of this MIB module has been derived from enterprise specific MIB modules, namely a MIB module for managing guests of the Xen hypervisor, a MIB module for managing virtual machines controlled by the VMware hypervisor, and a MIB module using the libvirt programming interface to access different hypervisors. However, this MIB module attempts to generalize the managed objects to support other implementations of hypervisors.

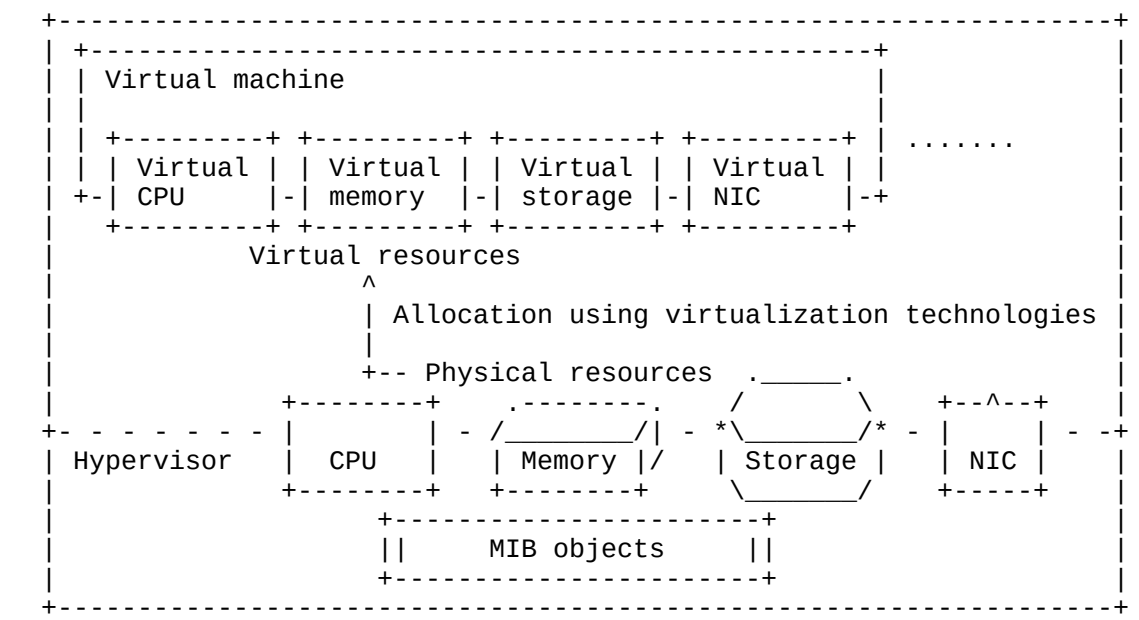
The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

2. The Internet-Standard Management Framework

For a detailed overview of the documents that describe the current Internet-Standard Management Framework, please refer to section 7 of RFC 3410 [RFC3410]. Managed objects are accessed via a virtual information store, termed the Management Information Base or MIB. MIB objects are generally accessed through the Simple Network Management Protocol (SNMP). Objects in the MIB are defined using the mechanisms defined in the Structure of Management Information (SMI). This memo specifies a MIB module that is compliant to the SMIv2, which is described in STD 58, RFC 2578 [RFC2578], STD 58, RFC 2579 [RFC2579] and STD 58, RFC 2580 [RFC2580].

3. Overview and Objectives

This document defines a portion of MIB for the management of virtual machines controlled by a hypervisor. This MIB module consists of the managed objects related to system and software information of a hypervisor, the list of virtual machines controlled by the hypervisor, and information of virtual resources allocated by the hypervisor to virtual machines. This document specifies four specific types of virtual resources that are common to many hypervisors; processors (CPUs), memory, network interfaces (NICs), and storage devices. The objects are independent of the hypervisors or operating systems running on virtual machines.



A hypervisor allocates virtual resources such as virtual CPUs, virtual memory, virtual storage devices, and virtual network interfaces to virtual machines from physical resources.

Figure 1: An example of a virtualization environment

On the common implementations of hypervisors, a hypervisor allocates virtual resources from physical resources; virtual CPUs, virtual memory, virtual storage devices, and virtual network interfaces to virtual machines as shown in Figure 1. Since the virtual resources allocated to virtual machines are managed by the hypervisor, the MIB objects are managed at a hypervisor. If the objects are accessed through the SNMP, an SNMP agent is launched at the hypervisor to

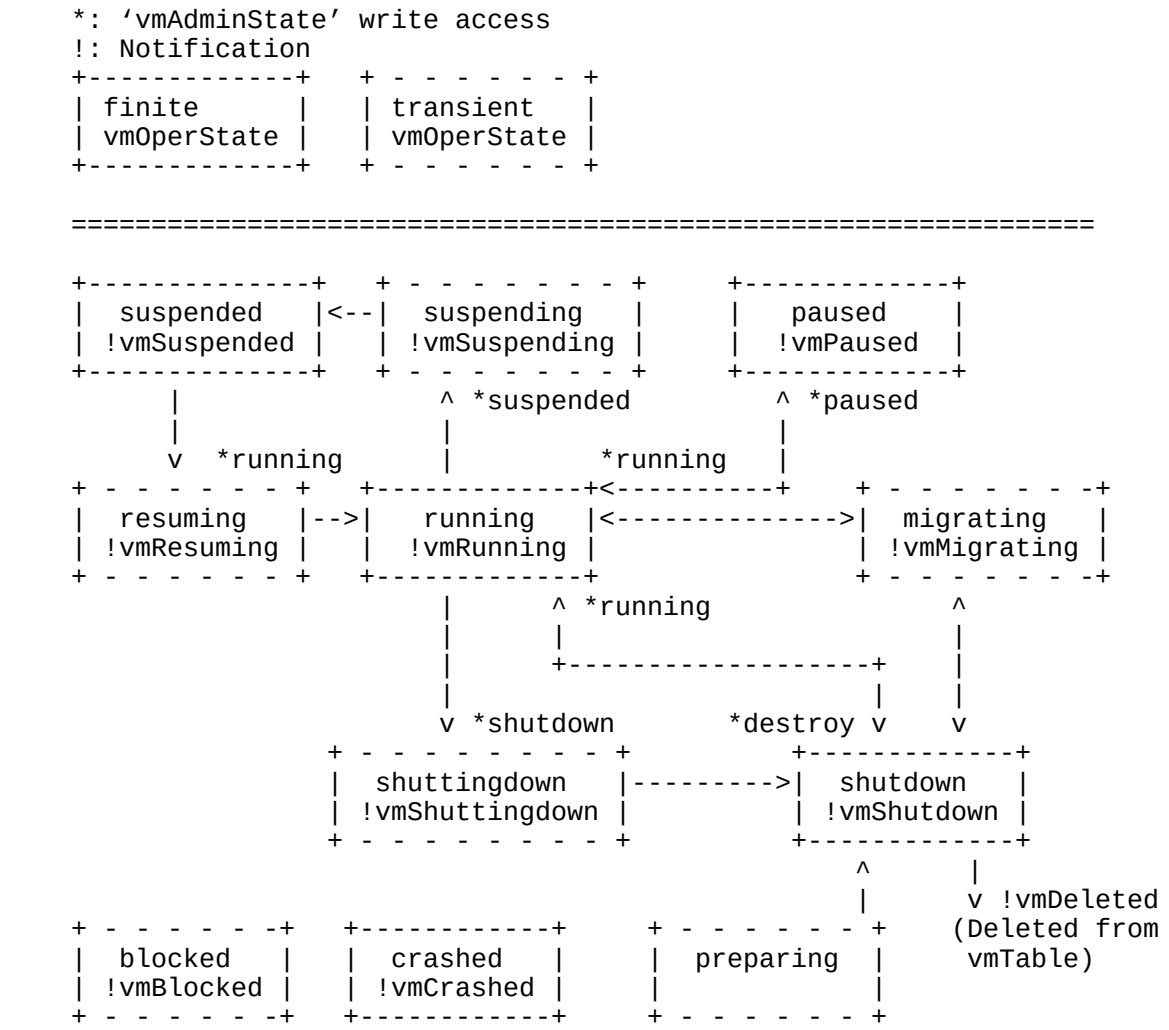
provide access to the objects.

The objects are managed from the viewpoint of the operators of hypervisors, but not the operators of virtual machines; i.e., the objects do not take into account the actual resource utilization on each virtual machine but the resource allocation from the physical resources. For example, vmNetworkIfIndex indicates the virtual interface associated with an interface of a virtual machine at the hypervisor, and consequently, the ‘in’ and ‘out’ directions denote ‘from a virtual machine to the hypervisor’ and ‘from the hypervisor to a virtual machine’, respectively. Moreover, vmStorageAllocatedSize denotes the size allocated by the hypervisor, but not the size actually used by the operating system on the virtual machine. This means that vmStorageDefinedSize and vmStorageAllocatedSize do not take different values when the vmStorageSourceType is ‘block’ or ‘raw’.

The objectives of this document are the followings: 1) This document defines the MIB objects common to many hypervisors for the management of virtual machines controlled by a hypervisor. 2) This document clarifies the relationship between other MIB modules for managing host computers and network devices.

4. Structure of the VM-MIB Module

The MIB module is organized into a group of scalars and tables. The scalars below ‘hypervisor’ provide basic information about the hypervisor. The ‘vmTable’ lists the virtual machines (guests) that are known to the hypervisor. The ‘vmCpuTable’ provides the mapping table of virtual CPUs to virtual machines, including CPU time used by each virtual CPU. The ’vmCpuAffinityTable’ provides the affinity of each virtual CPU to a physical CPU. The ‘vmStorageTable’ provides the list of virtual storage devices and their mapping to virtual machines. In case that an entry in the ‘vmStorageTable’ has a corresponding parent physical storage device managed in ‘vmStorageTable’ of HOST-RESOURCES-MIB [RFC2790], the entry contains a pointer ‘vmStorageParent’ to the physical storage device. The ‘vmNetworkTable’ provides the list of virtual network interfaces and their mapping to virtual machines. Each entry in the ‘vmNetworkTable’ also provides a pointer ‘vmNetworkIfIndex’ to the corresponding entry in the ‘ifTable’ of IF-MIB [RFC2863]. In case that an entry in the ‘vmNetworkTable’ has a corresponding parent physical network interface managed in ‘ifTable’ of IF-MIB, the entry contains a pointer ‘vmNetworkParent’ to the physical network interface.



The state transition of a virtual machine

Figure 2: State transition of a virtual machine

The 'vmAdminState' and 'vmOperState' textual conventions define an administrative state and an operational state model for virtual machines. Events causing transitions between major operational states will cause the generation of notifications. Per virtual machine (per-VM) notifications (vmRunning, vmShutdown, vmPaused, vmSuspended, vmCrashed, vmDeleted) are generated if vmPerVMNotificationsEnabled is true(1). Bulk notifications (vmBulkRunning, vmBulkShutdown, vmBulkPaused, vmBulkSuspended,

vmBulkCrashed, vmBulkDeleted) are generated if vmBulkNotificationsEnabled is true(1). The transition of 'vmOperState' by the write access to 'vmAdminState' and the notifications generated by the operational state changes are summarized in Figure 2. Note that the notifications shown in this figure are per-VM notifications. In the case of Bulk notifications, the prefix 'vm' is replaced with 'vmBulk'.

The bulk notification mechanism is designed to reduce the number of notifications that are trapped by an SNMP manager. This is because the number of virtual machines managed by a bunch of hypervisors in a datacenter possibly becomes several thousands or more, and consequently, many notifications could be trapped if these virtual machines frequently change their administrative state. The per-VM notifications carry more detailed information, but the scalability shall be a problem. An implementation shall support both, either of, or none of per-VM notifications and bulk notifications. The notification filtering mechanism described in section 6 of RFC 3413 [RFC3413] is used by the management applications to control the notifications.

The MIB module provides a few writable objects that can be used to make non-persistent changes, e.g., changing the memory allocation or the CPU allocation. It is not the goal of this MIB module to provide a configuration interface for virtual machines since other protocols and data modeling languages are more suitable for this task.

The OID tree structure of the MIB module is shown below.

```
--vmMIB (1.3.6.1.2.1.yyy)
  +--vmNotifications(0)
  |  +--vmRunning(1) [vmName, vmUUID, vmOperState]
  |  +--vmShuttingdown(2) [vmName, vmUUID, vmOperState]
  |  +--vmShutdown(3) [vmName, vmUUID, vmOperState]
  |  +--vmPaused(4) [vmName, vmUUID, vmOperState]
  |  +--vmSuspending(5) [vmName, vmUUID, vmOperState]
  |  +--vmSuspended(6) [vmName, vmUUID, vmOperState]
  |  +--vmResuming(7) [vmName, vmUUID, vmOperState]
  |  +--vmMigrating(8) [vmName, vmUUID, vmOperState]
  |  +--vmCrashed(9) [vmName, vmUUID, vmOperState]
  |  +--vmBlocked(10) [vmName, vmUUID, vmOperState]
  |  +--vmDeleted(11) [vmName, vmUUID, vmOperState, vmPersistent]
  |  +--vmBulkRunning(12) [vmAffectedVMs]
  |  +--vmBulkShutdown(13) [vmAffectedVMs]
  |  +--vmBulkShuttingdown(14) [vmAffectedVMs]
  |  +--vmBulkPaused(15) [vmAffectedVMs]
  |  +--vmBulkSuspending(16) [vmAffectedVMs]
  |  +--vmBulkSuspended(17) [vmAffectedVMs]
```

```
|  +--vmBulkResuming(18) [vmName, vmUUID, vmOperState]
|  +--vmBulkMigrating(19) [vmAffectedVMs]
|  +--vmBulkCrashed(20) [vmAffectedVMs]
|  +--vmBulkBlocked(21) [vmAffectedVMs]
|  +--vmBulkDeleted(22) [vmAffectedVMs]
+--vmObjects(1)
|  +--vmHypervisor(1)
|  |  +-- r-n SnmpAdminString    vmHvSoftware(1)
|  |  +-- r-n SnmpAdminString    vmHvVersion(2)
|  |  +-- r-n OBJECT IDENTIFIER  vmHvObjectID(3)
|  |  +-- r-n TimeTicks          vmHvUpTime(4)
|  +-- r-n Integer32  vmNumber(2)
|  +-- r-n TimeTicks  vmTableLastChange(3)
|  +--vmTable(4)
|  |  +--vmEntry(1) [vmIndex]
|  |     +-- --- VirtualMachineIndex  vmIndex(1)
|  |     +-- r-n SnmpAdminString      vmName(2)
|  |     +-- r-n UUIDorZero           vmUUID(3)
|  |     +-- r-n SnmpAdminString      vmOSType(4)
|  |     +-- rwn VirtualMachineAdminState
|  |     |                            vmAdminState(5)
|  |     +-- r-n VirtualMachineOperState
|  |     |                            vmOperState(6)
|  |     +-- r-n VirtualMachineAutoStart
|  |     |                            vmAutoStart(7)
|  |     +-- r-n VirtualMachinePersistent
|  |     |                            vmPersistent(8)
|  |     +-- rwn Integer32            vmCurCpuNumber(9)
|  |     +-- rwn Integer32            vmMinCpuNumber(10)
|  |     +-- rwn Integer32            vmMaxCpuNumber(11)
|  |     +-- r-n Integer32            vmMemUnit(12)
|  |     +-- rwn Integer32            vmCurMem(13)
|  |     +-- rwn Integer32            vmMinMem(14)
|  |     +-- rwn Integer32            vmMaxMem(15)
|  |     +-- r-n TimeTicks            vmUpTime(16)
|  |     +-- r-n Counter64            vmCpuTime(17)
|  +--vmCpuTable(5)
|  |  +--vmCpuEntry(1) [vmIndex, vmCpuIndex]
|  |     +-- --- VirtualMachineCpuIndex
|  |     |                            vmCpuIndex(1)
|  |     +-- r-n Counter64            vmCpuCoreTime(2)
|  +--vmCpuAffinityTable(6)
|  |  +--vmCpuAffinityEntry(1) [vmIndex,
|  |     |                      vmCpuIndex,
|  |     |                      vmCpuPhysIndex]
|  |     +-- --- Integer32            vmCpuPhysIndex(1)
|  |     +-- rwn Integer32            vmCpuAffinity(2)
|  +--vmStorageTable(7)
```

```
|  |  +--vmStorageEntry(1) [vmStorageVmIndex, vmStorageIndex]
|  |     +-- --- VirtualMachineIndexOrZero
|  |     |                              vmStorageVmIndex(1)
|  |     +-- --- VirtualMachineStorageIndex
|  |     |                              vmStorageIndex(2)
|  |     +-- r-n Integer32              vmStorageParent(3)
|  |     +-- r-n VirtualMachineStorageSourceType
|  |     |                              vmStorageSourceType(4)
|  |     +-- r-n SnmpAdminString        vmStorageSourceTypeString(5)
|  |     +-- r-n SnmpAdminString        vmStorageResourceID(6)
|  |     +-- r-n VirtualMachineStorageAccess
|  |     |                              vmStorageAccess(7)
|  |     +-- r-n VirtualMachineStorageMediaType
|  |     |                              vmStorageMediaType(8)
|  |     +-- r-n SnmpAdminString        vmStorageMediaTypeString(9)
|  |     +-- r-n Integer32              vmStorageSizeUnit(10)
|  |     +-- r-n Integer32              vmStorageDefinedSize(11)
|  |     +-- r-n Integer32              vmStorageAllocatedSize(12)
|  |     +-- r-n Counter64              vmStorageReadIOs(13)
|  |     +-- r-n Counter64              vmStorageWriteIOs(14)
|  +--vmNetworkTable(8)
|  |  +--vmNetworkEntry(1) [vmIndex, vmNetworkIndex]
|  |     +-- --- VirtualMachineNetworkIndex
|  |     |                              vmNetworkIndex(1)
|  |     +-- r-n InterfaceIndexOrZero   vmNetworkIfIndex(2)
|  |     +-- r-n InterfaceIndexOrZero   vmNetworkParent(3)
|  |     +-- r-n SnmpAdminString        vmNetworkModel(4)
|  |     +-- r-n PhysAddress            vmNetworkPhysAddress(5)
|  +-- rwn TruthValue           vmPerVMNotificationsEnabled(9)
|  +-- rwn TruthValue           vmBulkNotificationsEnabled(10)
|  +-- --n VirtualMachineList   vmAffectedVMs(11)
+--vmConformance(2)
   +--vmCompliances(1)
   |  +--vmFullCompliances(1)
   |  +--vmReadOnlyCompliances(2)
   +--vmGroups(2)
      +--vmHypervisorGroup(1)
      +--vmVirtualMachineGroup(2)
      +--vmCpuGroup(3)
      +--vmCpuAffinityGroup(4)
      +--vmStorageGroup(5)
      +--vmNetworkGroup(6)
      +--vmPerVMNotificationOptionalGroup(7)
      +--vmBulkNotificationsVariablesGroup(8)
      +--vmBulkNotificationOptionalGroup(9)
```

5. Relationship to Other MIB Modules

HOST-RESOURCES-MIB [RFC2790] defines the MIB objects for managing host systems. Hypervisors shall implement HOST-RESOURCES-MIB. On systems implementing HOST-RESOURCES-MIB, the objects of HOST-RESOURCES-MIB indicate resources of a hypervisor. Some objects of HOST-RESOURCES-MIB shall also be used to indicate physical resources through indexes. On systems implementing HOST-RESOURCES-MIB, the 'vmCpuPhysIndex' points to the processor's 'hrDeviceIndex' in the 'hrProcessorTable'. The 'vmStorageParent' also points to the storage device's 'hrStorageIndex' in the 'hrStorageTable'.

HOST-RESOURCES-MIB shall be implemented on systems running on virtual machines. It enables to manage the objects related to the resources of virtual machines from the viewpoint of virtual machine operators. However, from the viewpoint of hypervisor operators, it cannot obtain the list of virtual machines controlled by a hypervisor and the relationship between physical and virtual resources. This document defines the objects of these information.

IF-MIB [RFC2863] defines the MIB objects for managing network interfaces. Both physical and virtual network interfaces are required to be contained in the 'ifTable' of IF-MIB. The virtual network interfaces in the 'ifTable' of IF-MIB are pointed from the 'vmNetworkTable' defined in this document through a pointer 'vmNetworkIfIndex'. In case that an entry in the 'vmNetworkTable' has a corresponding parent physical network interface managed in the 'ifTable' of IF-MIB, the entry contains a pointer 'vmNetworkParent' to the physical network interface.

The objects related to virtual switches are not also included in the MIB module defined in this document though virtual switches shall be placed on a hypervisor. This is because the virtual network interfaces are the lowest abstraction of network resources allocated to a virtual machine. Instead of including the objects related to virtual switches, for example, IEEE8021-BRIDGE-MIB and IEEE8021-Q-BRIDGE-MIB could be used.

The other objects related to virtual machines such as management IP addresses of a virtual machine are not included in this MIB module because this MIB module defines the objects common to general hypervisors but they are specific to some hypervisors. They may be included in the entLogicalTable of ENTITY-MIB [RFC6933].

6. Definitions

```
   VM-MIB DEFINITIONS ::= BEGIN

   IMPORTS
       MODULE-IDENTITY, OBJECT-TYPE, NOTIFICATION-TYPE, TimeTicks,
       Counter64, Integer32, mib-2
           FROM SNMPv2-SMI
       OBJECT-GROUP, MODULE-COMPLIANCE, NOTIFICATION-GROUP
           FROM SNMPv2-CONF
       TEXTUAL-CONVENTION, PhysAddress, TruthValue
           FROM SNMPv2-TC
       SnmpAdminString
           FROM SNMP-FRAMEWORK-MIB
       UUIDorZero
           FROM UUID-TC-MIB
       InterfaceIndexOrZero
           FROM IF-MIB;

   vmMIB MODULE-IDENTITY
       LAST-UPDATED "201402080000Z"        -- 8 February 2014
       ORGANIZATION "IETF Operations and Management Area Working Group"
       CONTACT-INFO
               "
               WG E-mail: opsawg@ietf.org
               Mailing list subscription info:
                 https://www.ietf.org/mailman/listinfo/opsawg

               Hirochika Asai
               The University of Tokyo
               7-3-1 Hongo
               Bunkyo-ku, Tokyo  113-8656
               JP
               Phone: +81 3 5841 6748
               Email: panda@hongo.wide.ad.jp

               Michael MacFaden
               VMware Inc.
               Email: mrm@vmware.com

               Juergen Schoenwaelder
               Jacobs University
               Campus Ring 1
               Bremen 28759
               Germany
               Email: j.schoenwaelder@jacobs-university.de

               Keiichi Shima
```

IIJ Innovation Institute Inc.
3-13 Kanda-Nishikicho
Chiyoda-ku, Tokyo 101-0054
JP
Email: keiichi@iijlab.net

Tina Tsou
Huawei Technologies (USA)
2330 Central Expressway
Santa Clara CA 95050
USA
Email: tina.tsou.zouting@huawei.com
"

```
    DESCRIPTION
            "This MIB module is for use in managing a hypervisor and
            virtual machines controlled by the hypervisor.  The OID
            'yyy' is temporary one, and it must be assigned by IANA
            when this becomes an official document.

            Copyright (c) 2014 IETF Trust and the persons identified
            as authors of the code.  All rights reserved.

            Redistribution and use in source and binary forms, with
            or without modification, is permitted pursuant to, and
            subject to the license terms contained in, the
            Simplified BSD License set forth in Section 4.c of the
            IETF Trust's Legal Provisions Relating to IETF Documents
            (http://trustee.ietf.org/license-info)."

    REVISION "201402080000Z"        -- 8 February 2014
    DESCRIPTION
            "The original version of this MIB, published as
            RFCXXXX."
    ::= { mib-2 yyy }


vmNotifications OBJECT IDENTIFIER ::= { vmMIB 0 }
vmObjects       OBJECT IDENTIFIER ::= { vmMIB 1 }
vmConformance   OBJECT IDENTIFIER ::= { vmMIB 2 }


-- Textual conversion definitions
--
VirtualMachineIndex ::= TEXTUAL-CONVENTION
    DISPLAY-HINT "d"
    STATUS       current
    DESCRIPTION
```

```
                "A unique value, greater than zero, identifying a
                virtual machine.  The value for each virtual machine
                must remain constant at least from one re-initialization
                of the hypervisor to the next re-initialization."
        SYNTAX      Integer32 (1..2147483647)

    VirtualMachineIndexOrZero ::= TEXTUAL-CONVENTION
        DISPLAY-HINT "d"
        STATUS      current
        DESCRIPTION
                "This textual convention is an extension of the
                VirtualMachineIndex convention.  This extension permits
                the additional value of zero.  The meaning of the value
                zero is object-specific and must therefore be defined as
                part of the description of any object which uses this
                syntax.  Examples of the usage of zero might include
                situations where a virtual machine is unknown, or when
                none or all virtual machines need to be referenced."
        SYNTAX      Integer32 (0..2147483647)

    VirtualMachineAdminState ::= TEXTUAL-CONVENTION
        STATUS     current
        DESCRIPTION
                "The administrative state of a virtual machine:

                running(1)    The administrative state of the virtual
                              machine indicating the virtual machine
                              is currently online or should be brought
                              online.

                suspended(2)  The administrative state of the virtual
                              machine where its memory and CPU execution
                              state has been saved to persistent store
                              and will be restored at next running(1).

                paused(3)     The administrative state indicating the
                              virtual machine is resident in memory but
                              is no longer scheduled to execute by the
                              hypervisor.

                shutdown(4)   The administrative state of the virtual
                              machine indicating the virtual machine
                              is currently offline or should be taken
                              shutting down.

                destroy(5)    The administrative state of the virtual
                              machine indicating the virtual machine
                              should be forcibly shutdown.  After the
```

```
                            destroy operation, the administrative
                            state should be automatically changed to
                            shutdown(4)."
        SYNTAX       INTEGER {
                        running(1),
                        suspended(2),
                        paused(3),
                        shutdown(4),
                        destroy(5)
                     }

    VirtualMachineOperState ::= TEXTUAL-CONVENTION
        STATUS       current
        DESCRIPTION
                "The operational state of a virtual machine:

                unknown(1)     The operational state of the virtual
                               machine is unknown, e.g., because the
                               implementation failed to obtain the state
                               from the hypervisor.

                other(2)       The operational state of the virtual
                               machine indicating that an operational
                               state is obtained from the hypervisor but
                               it is not a state defined in this MIB
                               module.

                preparing(3)   The operational state of the virtual
                               machine indicating the virtual machine is
                               currently in the process of preparation,
                               e.g., allocating and initializing virtual
                               storage after creating (defining) virtual
                               machine.

                running(4)     The operational state of the virtual
                               machine indicating the virtual machine is
                               currently executed but it is not in the
                               process of preparing(3), suspending(6),
                               resuming(8), migrating(10), and
                               shuttingdown(11).

                blocked(5)     The operational state of the virtual
                               machine indicating the execution of the
                               virtual machine is currently blocked,
                               e.g., waiting for some action of the
                               hypervisor to finish.  This is a
                               transient state from/to other states.
```

suspending(6) The operational state of the virtual machine indicating the virtual machine is currently in the process of suspending to save its memory and CPU execution state to persistent store. This is a transient state from running(4) to suspended(7).

suspended(7) The operational state of the virtual machine indicating the virtual machine is currently suspended, which means the memory and CPU execution state of the virtual machine are saved to persistent store. During this state, the virtual machine is not scheduled to execute by the hypervisor.

resuming(8) The operational state of the virtual machine indicating the virtual machine is currently in the process of resuming to restore its memory and CPU execution state from persistent store. This is a transient state from suspended(7) to running(4).

paused(9) The operational state of the virtual machine indicating the virtual machine is resident in memory but no longer scheduled to execute by the hypervisor.

migrating(10) The operational state of the virtual machine indicating the virtual machine is currently in the process of migration from/to another hypervisor.

shuttingdown(11)
The operational state of the virtual machine indicating the virtual machine is currently in the process of shutting down. This is a transient state from running(4) to shutdown(12).

shutdown(12) The operational state of the virtual machine indicating the virtual machine is down, and CPU execution is no longer scheduled by the hypervisor and its memory is not resident in the hypervisor.

```
                crashed(13)    The operational state of the virtual
                               machine indicating the virtual machine
                               has crashed."
        SYNTAX       INTEGER {
                        unknown(1),
                        other(2),
                        preparing(3),
                        running(4),
                        blocked(5),
                        suspending(6),
                        suspended(7),
                        resuming(8),
                        paused(9),
                        migrating(10),
                        shuttingdown(11),
                        shutdown(12),
                        crashed(13)
                     }

    VirtualMachineAutoStart ::= TEXTUAL-CONVENTION
        STATUS       current
        DESCRIPTION
                "The autostart configuration of a virtual machine:

                unknown(1)     The autostart configuration is unknown,
                               e.g., because the implementation failed
                               to obtain the autostart configuration
                               from the hypervisor.

                enabled(2)     The autostart configuration of the
                               virtual machine is enabled.  The virtual
                               machine should be automatically brought
                               online at the next re-initialization of
                               the hypervisor.

                disabled(3)    The autostart configuration of the
                               virtual machine is disabled.  The virtual
                               machine should not be automatically
                               brought online at the next
                               re-initialization of the hypervisor."
        SYNTAX      INTEGER {
                        unknown(1),
                        enabled(2),
                        disabled(3)
                    }

    VirtualMachinePersistent ::= TEXTUAL-CONVENTION
        STATUS       current
```

```
        DESCRIPTION
                "This value indicates whether a virtual machine has a
                persistent configuration which means the virtual machine
                will still exist after shutting down:

                unknown(1)     The persistent configuration is unknown,
                               e.g., because the implementation failed
                               to obtain the persistent configuration
                               from the hypervisor. (read-only)

                persistent(2)  The virtual machine is persistent, i.e.,
                               the virtual machine will exist after its
                               shutting down.

                transient(3)   The virtual machine is transient, i.e.,
                               the virtual machine will not exist after
                               its shutting down."
        SYNTAX       INTEGER {
                        unknown(1),
                        persistent(2),
                        transient(3)
                     }

    VirtualMachineCpuIndex ::= TEXTUAL-CONVENTION
        DISPLAY-HINT "d"
        STATUS       current
        DESCRIPTION
                "A unique value for each virtual machine, greater than
                zero, identifying a virtual CPU assigned to a virtual
                machine.  The value for each virtual CPU must remain
                constant at least from one re-initialization of the
                hypervisor to the next re-initialization."
         SYNTAX      Integer32 (1..2147483647)

    VirtualMachineStorageIndex ::= TEXTUAL-CONVENTION
        DISPLAY-HINT "d"
        STATUS       current
        DESCRIPTION
                "A unique value for each virtual machine, greater than
                zero, identifying a virtual storage device allocated to
                a virtual machine.  The value for each virtual storage
                device must remain constant at least from one
                re-initialization of the hypervisor to the next
                re-initialization."
         SYNTAX      Integer32 (1..2147483647)

    VirtualMachineStorageSourceType ::= TEXTUAL-CONVENTION
        STATUS       current
```

```
        DESCRIPTION
                "The source type of a virtual storage device:

                unknown(1)     The source type is unknown, e.g., because
                               the implementation failed to obtain the
                               media type from the hypervisor.

                other(2)       The source type is other than those
                               defined in this conversion.

                block(3)       The source type is a block device.

                raw(4)         The source type is a raw-formatted file.

                sparse(5)      The source type is a sparse file.

                network(6)     The source type is a network device."
        SYNTAX       INTEGER {
                        unknown(1),
                        other(2),
                        block(3),
                        raw(4),
                        sparse(5),
                        network(6)
                     }

    VirtualMachineStorageAccess ::= TEXTUAL-CONVENTION
        STATUS       current
        DESCRIPTION
                "The access permission of a virtual storage:

                readwrite(1)   The virtual storage is a read-write
                               device.

                readonly(2)    The virtual storage is a read-only
                               device."
        SYNTAX       INTEGER {
                        readwrite(1),
                        readonly(2)
                     }

    VirtualMachineStorageMediaType ::= TEXTUAL-CONVENTION
        STATUS       current
        DESCRIPTION
                "The media type of a virtual storage device:

                unknown(1)     The media type is unknown, e.g., because
                               the implementation failed to obtain the
```

```
                              media type from the hypervisor.

                 other(2)     The media type is other than those
                              defined in this conversion.

                 hardDisk(3)  The media type is hard disk.

                 opticalDisk(4) The media type is optical disk."
        SYNTAX       INTEGER {
                        other(1),
                        unknown(2),
                        hardDisk(3),
                        opticalDisk(4)
                     }

    VirtualMachineNetworkIndex ::= TEXTUAL-CONVENTION
        DISPLAY-HINT "d"
        STATUS       current
        DESCRIPTION
                "A unique value for each virtual machine, greater than
                zero, identifying a virtual network interface allocated
                to the virtual machine.  The value for each virtual
                network interface must remain constant at least from one
                re-initialization of the hypervisor to the next
                re-initialization."
         SYNTAX      Integer32 (1..2147483647)

    VirtualMachineList ::= TEXTUAL-CONVENTION
        DISPLAY-HINT "1x"
        STATUS       current
        DESCRIPTION
                "Each octet within this value specifies a set of eight
                virtual machine vmIndex, with the first octet specifying
                virtual machine 1 through 8, the second octet specifying
                virtual machine 9 through 16, etc.  Within each octet,
                the most significant bit represents the lowest numbered
                vmIndex, and the least significant bit represents the
                highest numbered vmIndex.  Thus, each virtual machine of
                the host is represented by a single bit within the value
                of this object.  If that bit has a value of '1', then
                that virtual machine is included in the set of virtual
                machines; the virtual machine is not included if its bit
                has a value of '0'."
        SYNTAX      OCTET STRING

    -- The hypervisor group
    --
    -- A collection of objects common to all hypervisors.
```

```
   --
   vmHypervisor    OBJECT IDENTIFIER ::= { vmObjects 1 }

   vmHvSoftware OBJECT-TYPE
       SYNTAX       SnmpAdminString (SIZE (0..255))
       MAX-ACCESS   read-only
       STATUS       current
       DESCRIPTION
               "A textual description of the hypervisor software.  This
               value should not include its version, and it should be
               included in 'vmHvVersion'."
       ::= { vmHypervisor 1 }

   vmHvVersion OBJECT-TYPE
       SYNTAX       SnmpAdminString (SIZE (0..255))
       MAX-ACCESS   read-only
       STATUS       current
       DESCRIPTION
               "A textual description of the version of the hypervisor
               software."
       ::= { vmHypervisor 2 }

   vmHvObjectID OBJECT-TYPE
       SYNTAX       OBJECT IDENTIFIER
       MAX-ACCESS   read-only
       STATUS       current
       DESCRIPTION
               "The vendor's authoritative identification of the
               hypervisor software contained in the entity.  This value
               is allocated within the SMI enterprises
               subtree (1.3.6.1.4.1).  Note that this is different from
               sysObjectID in the SNMPv2-MIB [RFC3418] because
               sysObjectID is not the identification of the hypervisor
               software but the device, firmware, or management
               operating system."
       ::= { vmHypervisor 3 }

   vmHvUpTime OBJECT-TYPE
       SYNTAX       TimeTicks
       MAX-ACCESS   read-only
       STATUS       current
       DESCRIPTION
               "The time (in centi-seconds) since the hypervisor was
               last re-initialized.  Note that this is different from
               sysUpTime in the SNMPv2-MIB [RFC3418] and hrSystemUptime
               in the HOST-RESOURCES-MIB [RFC2790] because sysUpTime is
               the uptime of the network management portion of the
               system, and hrSystemUptime is the uptime of the
```

```
                management operating system but not the hypervisor
                software."
        ::= { vmHypervisor 4 }


    -- The virtual machine information
    --
    -- A collection of objects common to all virtual machines.
    --
    vmNumber  OBJECT-TYPE
        SYNTAX       Integer32 (0..2147483647)
        MAX-ACCESS   read-only
        STATUS       current
        DESCRIPTION
                "The number of virtual machines (regardless of their
                current state) present on this hypervisor."
        ::= { vmObjects 2 }

    vmTableLastChange  OBJECT-TYPE
        SYNTAX       TimeTicks
        MAX-ACCESS   read-only
        STATUS       current
        DESCRIPTION
                "The value of vmHvUpTime at the time of the last creation
                or deletion of an entry in the vmTable."
        ::= { vmObjects 3 }

    vmTable OBJECT-TYPE
        SYNTAX       SEQUENCE OF VmEntry
        MAX-ACCESS   not-accessible
        STATUS       current
        DESCRIPTION
                "A list of virtual machine entries.  The number of
                entries is given by the value of vmNumber."
        ::= { vmObjects 4 }

    vmEntry OBJECT-TYPE
        SYNTAX       VmEntry
        MAX-ACCESS   not-accessible
        STATUS       current
        DESCRIPTION
                "An entry containing management information applicable
                to a particular virtual machine."
        INDEX   { vmIndex }
        ::= { vmTable 1 }

    VmEntry ::=
        SEQUENCE {
```

```
            vmIndex                 VirtualMachineIndex,
            vmName                  SnmpAdminString,
            vmUUID                  UUIDorZero,
            vmOSType                SnmpAdminString,
            vmAdminState            VirtualMachineAdminState,
            vmOperState             VirtualMachineOperState,
            vmAutoStart             VirtualMachineAutoStart,
            vmPersistent            VirtualMachinePersistent,
            vmCurCpuNumber          Integer32,
            vmMinCpuNumber          Integer32,
            vmMaxCpuNumber          Integer32,
            vmMemUnit               Integer32,
            vmCurMem                Integer32,
            vmMinMem                Integer32,
            vmMaxMem                Integer32,
            vmUpTime                TimeTicks,
            vmCpuTime               Counter64
        }

    vmIndex OBJECT-TYPE
        SYNTAX       VirtualMachineIndex
        MAX-ACCESS   not-accessible
        STATUS       current
        DESCRIPTION
                "A unique value, greater than zero, identifying the
                virtual machine. The value assigned to a given virtual
                machine may not persist across re-initialization of the
                hypervisor.  A command generator must use the vmUUID to
                identify a given virtual machine of interest."
        ::= { vmEntry 1 }

    vmName OBJECT-TYPE
        SYNTAX       SnmpAdminString (SIZE (0..255))
        MAX-ACCESS   read-only
        STATUS       current
        DESCRIPTION
                "A textual name of the virtual machine."
        ::= { vmEntry 2 }

    vmUUID OBJECT-TYPE
        SYNTAX       UUIDorZero
        MAX-ACCESS   read-only
        STATUS       current
        DESCRIPTION
                "The virtual machine's 128-bit UUID or the zero-length
                string when a UUID is not available.  The UUID if set
                must uniquely identify a virtual machine from all other
                virtual machines in an administrative region.  A
```

```
                zero-length octet string is returned if no UUID
                information is known."
        ::= { vmEntry 3 }

    vmOSType OBJECT-TYPE
        SYNTAX       SnmpAdminString (SIZE (0..255))
        MAX-ACCESS   read-only
        STATUS       current
        DESCRIPTION
                "A textual description containing operating system
                information installed on the virtual machine.  This
                value corresponds to the operating system the hypervisor
                assumes to be running when the virtual machine is
                started.  This may differ from the actual operating
                system in case the virtual machine boots into a
                different operating system."
        ::= { vmEntry 4 }

    vmAdminState OBJECT-TYPE
        SYNTAX       VirtualMachineAdminState
        MAX-ACCESS   read-write
        STATUS       current
        DESCRIPTION
                "The administrative power state of the virtual machine.
                Note that a virtual machine is supposed to be resumed
                when vmAdminState of the virtual machine is changed from
                suspended(2) or paused(3) to running(1)."
        ::= { vmEntry 5 }

    vmOperState OBJECT-TYPE
        SYNTAX       VirtualMachineOperState
        MAX-ACCESS   read-only
        STATUS       current
        DESCRIPTION
                "The operational state of the virtual machine."
        ::= { vmEntry 6 }

    vmAutoStart OBJECT-TYPE
        SYNTAX       VirtualMachineAutoStart
        MAX-ACCESS   read-only
        STATUS       current
        DESCRIPTION
                "The autostart configuration of the virtual machine.  If
                this value is enable(2), the virtual machine
                automatically starts at the next initialization of the
                hypervisor."
        ::= { vmEntry 7 }
```

```
   vmPersistent OBJECT-TYPE
       SYNTAX       VirtualMachinePersistent
       MAX-ACCESS   read-only
       STATUS       current
       DESCRIPTION
               "This value indicates whether the virtual machine has a
               persistent configuration which means the virtual machine
               will still exist after its shutdown."
       ::= { vmEntry 8 }

   vmCurCpuNumber OBJECT-TYPE
       SYNTAX       Integer32 (0..2147483647)
       MAX-ACCESS   read-write
       STATUS       current
       DESCRIPTION
               "The number of virtual CPUs currently assigned to the
               virtual machine.  Changes to this object MUST NOT
               persist across re-initialization of the hypervisor."
       ::= { vmEntry 9 }

   vmMinCpuNumber OBJECT-TYPE
       SYNTAX       Integer32 (-1|0..2147483647)
       MAX-ACCESS   read-write
       STATUS       current
       DESCRIPTION
               "The minimum number of virtual CPUs that are assigned to
               the virtual machine when it is in a power-on state.  The
               value -1 indicates that there is no hard boundary for
               the minimum number of virtual CPUs.  Changes to this
               object MUST NOT persist across re-initialization of the
               hypervisor."
       ::= { vmEntry 10 }

   vmMaxCpuNumber OBJECT-TYPE
       SYNTAX       Integer32 (-1|0..2147483647)
       MAX-ACCESS   read-write
       STATUS       current
       DESCRIPTION
               "The maximum number of virtual CPUs that are assigned to
               the virtual machine when it is in a power-on state.  The
               value -1 indicates that there is no limit.  Changes to
               this object MUST NOT persist across re-initialization of
               the hypervisor."
       ::= { vmEntry 11 }

   vmMemUnit OBJECT-TYPE
       SYNTAX       Integer32 (1..2147483647)
       MAX-ACCESS   read-only
```

```
        STATUS       current
        DESCRIPTION
                "The multiplication unit for vmCurMem, vmMinMem, and
                vmMaxMem.  For example, when this value is 1024, the
                memory size unit for vmCurMem, vmMinMem, and vmMaxMem is
                KiB."
        ::= { vmEntry 12 }

    vmCurMem OBJECT-TYPE
        SYNTAX       Integer32 (0..2147483647)
        MAX-ACCESS   read-write
        STATUS       current
        DESCRIPTION
                "The current memory size currently allocated to the
                virtual memory module in the unit designated by
                vmMemUnit.  Changes to this object MUST NOT persist
                across re-initialization of the hypervisor."
        ::= { vmEntry 13 }

    vmMinMem OBJECT-TYPE
        SYNTAX       Integer32 (-1|0..2147483647)
        MAX-ACCESS   read-write
        STATUS       current
        DESCRIPTION
                "The minimum memory size defined to the virtual machine
                in the unit designated by vmMemUnit.  The value -1
                indicates that there is no hard boundary for the minimum
                memory size.  Changes to this object MUST NOT persist
                across re-initialization of the hypervisor."
        ::= { vmEntry 14 }

    vmMaxMem OBJECT-TYPE
        SYNTAX       Integer32 (-1|0..2147483647)
        MAX-ACCESS   read-write
        STATUS       current
        DESCRIPTION
                "The maximum memory size defined to the virtual machine
                in the unit designated by vmMemUnit.  The value -1
                indicates that there is no limit.  Changes to this
                object MUST NOT persist across re-initialization of the
                hypervisor."
        ::= { vmEntry 15 }


    vmUpTime OBJECT-TYPE
        SYNTAX       TimeTicks
        MAX-ACCESS   read-only
        STATUS       current
```

```
        DESCRIPTION
                "The time (in centi-seconds) since the administrative
                state of the virtual machine was last changed from
                shutdown(4) to running(1)."
        ::= { vmEntry 16 }

    vmCpuTime OBJECT-TYPE
        SYNTAX       Counter64
        UNITS        "microsecond"
        MAX-ACCESS   read-only
        STATUS       current
        DESCRIPTION
                "The total CPU time used in microsecond.  If the number
                of virtual CPUs is larger than 1, vmCpuTime may exceed
                real time.

                Discontinuities in the value of this counter can occur
                at re-initialization of the hypervisor, and
                administrative state (vmAdminState) changes of the
                virtual machine."
        ::= { vmEntry 17 }

    -- The virtual CPU on each virtual machines
    vmCpuTable OBJECT-TYPE
        SYNTAX       SEQUENCE OF VmCpuEntry
        MAX-ACCESS   not-accessible
        STATUS       current
        DESCRIPTION
                "The table of virtual CPUs provided by the hypervisor."
        ::= { vmObjects 5 }

    vmCpuEntry OBJECT-TYPE
        SYNTAX       VmCpuEntry
        MAX-ACCESS   not-accessible
        STATUS       current
        DESCRIPTION
                "An entry for one virtual processor assigned to a
                virtual machine."
        INDEX { vmIndex, vmCpuIndex }
        ::= { vmCpuTable 1 }

    VmCpuEntry ::=
        SEQUENCE {
            vmCpuIndex              VirtualMachineCpuIndex,
            vmCpuCoreTime           Counter64
        }

    vmCpuIndex OBJECT-TYPE
```

```
        SYNTAX       VirtualMachineCpuIndex
        MAX-ACCESS   not-accessible
        STATUS       current
        DESCRIPTION
                "A unique value identifying a virtual CPU assigned to
                the virtual machine."
        ::= { vmCpuEntry 1 }

    vmCpuCoreTime OBJECT-TYPE
        SYNTAX       Counter64
        UNITS        "microsecond"
        MAX-ACCESS   read-only
        STATUS       current
        DESCRIPTION
                "The total CPU time used by this virtual CPU in
                microsecond.

                Discontinuities in the value of this counter can occur
                at re-initialization of the hypervisor, and
                administrative state (vmAdminState) changes of the
                virtual machine."
        ::= { vmCpuEntry 2 }

    -- The virtual CPU affinity on each virtual machines
    vmCpuAffinityTable OBJECT-TYPE
        SYNTAX       SEQUENCE OF VmCpuAffinityEntry
        MAX-ACCESS   not-accessible
        STATUS       current
        DESCRIPTION
                "A list of CPU affinity entries of a virtual CPU."
        ::= { vmObjects 6 }

    vmCpuAffinityEntry OBJECT-TYPE
        SYNTAX       VmCpuAffinityEntry
        MAX-ACCESS   not-accessible
        STATUS       current
        DESCRIPTION
                "An entry containing CPU affinity associated with a
                particular virtual machine."
        INDEX   { vmIndex, vmCpuIndex, vmCpuPhysIndex }
        ::= { vmCpuAffinityTable 1 }

    VmCpuAffinityEntry ::=
        SEQUENCE {
            vmCpuPhysIndex          Integer32,
            vmCpuAffinity           Integer32
        }
```

```
   vmCpuPhysIndex OBJECT-TYPE
       SYNTAX       Integer32 (1..2147483647)
       MAX-ACCESS   not-accessible
       STATUS       current
       DESCRIPTION
               "A value identifying a physical CPU on the hypervisor.
               On systems implementing the HOST-RESOURCES-MIB, the
               value must be the same value that is used as the index
               in the hrProcessorTable (hrDeviceIndex)."
       ::= { vmCpuAffinityEntry 2 }

   vmCpuAffinity OBJECT-TYPE
       SYNTAX       INTEGER {
                       unknown(0),   -- unknown
                       enable(1),    -- enabled
                       disable(2)    -- disabled
                    }
       MAX-ACCESS   read-write
       STATUS       current
       DESCRIPTION
               "The CPU affinity of this virtual CPU to the physical
               CPU represented by 'vmCpuPhysIndex'."
       ::= { vmCpuAffinityEntry 3 }


   -- The virtual storage devices on each virtual machine.  This
   -- document defines some overlapped objects with hrStorage in
   -- HOST-RESOURCES-MIB [RFC2790], because virtual resources shall be
   -- allocated from the hypervisor's resources, which is the 'host
   -- resources'
   vmStorageTable OBJECT-TYPE
       SYNTAX       SEQUENCE OF VmStorageEntry
       MAX-ACCESS   not-accessible
       STATUS       current
       DESCRIPTION
               "The conceptual table of virtual storage devices
               attached to the virtual machine."
       ::= { vmObjects 7 }

   vmStorageEntry OBJECT-TYPE
       SYNTAX       VmStorageEntry
       MAX-ACCESS   not-accessible
       STATUS       current
       DESCRIPTION
               "An entry for one virtual storage device attached to the
               virtual machine."
       INDEX { vmStorageVmIndex, vmStorageIndex }
       ::= { vmStorageTable 1 }
```

```
   VmStorageEntry ::=
       SEQUENCE {
           vmStorageVmIndex        VirtualMachineIndexOrZero,
           vmStorageIndex          VirtualMachineStorageIndex,
           vmStorageParent         Integer32,
           vmStorageSourceType     VirtualMachineStorageSourceType,
           vmStorageSourceTypeString
                                   SnmpAdminString,
           vmStorageResourceID     SnmpAdminString,
           vmStorageAccess         VirtualMachineStorageAccess,
           vmStorageMediaType      VirtualMachineStorageMediaType,
           vmStorageMediaTypeString
                                   SnmpAdminString,
           vmStorageSizeUnit       Integer32,
           vmStorageDefinedSize    Integer32,
           vmStorageAllocatedSize  Integer32,
           vmStorageReadIOs        Counter64,
           vmStorageWriteIOs       Counter64
       }

   vmStorageVmIndex OBJECT-TYPE
       SYNTAX       VirtualMachineIndexOrZero
       MAX-ACCESS   not-accessible
       STATUS       current
       DESCRIPTION
               "This value identifies the virtual machine (guest) this
               storage device has been allocated to.  The value zero
               indicates that the storage device is currently not
               allocated to any virtual machines."
       ::= { vmStorageEntry 1 }

   vmStorageIndex OBJECT-TYPE
       SYNTAX       VirtualMachineStorageIndex
       MAX-ACCESS   not-accessible
       STATUS       current
       DESCRIPTION
               "A unique value identifying a virtual storage device
               allocated to the virtual machine."
       ::= { vmStorageEntry 2 }

   vmStorageParent OBJECT-TYPE
       SYNTAX       Integer32 (0..2147483647)
       MAX-ACCESS   read-only
       STATUS       current
       DESCRIPTION
               "The value of hrStorageIndex which is the parent (i.e.,
               physical) device of this virtual device on systems
               implementing the HOST-RESOURCES-MIB.  The value zero
```

```
                denotes this virtual device is not any child represented
                in the hrStorageTable."
        ::= { vmStorageEntry 3 }

    vmStorageSourceType OBJECT-TYPE
        SYNTAX       VirtualMachineStorageSourceType
        MAX-ACCESS   read-only
        STATUS       current
        DESCRIPTION
                "The source type of the virtual storage device."
        ::= { vmStorageEntry 4 }

    vmStorageSourceTypeString OBJECT-TYPE
        SYNTAX       SnmpAdminString (SIZE (0..255))
        MAX-ACCESS   read-only
        STATUS       current
        DESCRIPTION
                "A (detailed) textual string of the source type of the
                virtual storage device.  For example, this represents
                the specific format name of the sparse file."
        ::= { vmStorageEntry 5 }

    vmStorageResourceID OBJECT-TYPE
        SYNTAX       SnmpAdminString (SIZE (0..255))
        MAX-ACCESS   read-only
        STATUS       current
        DESCRIPTION
                "A textual string that represents the resource
                identifier of the virtual storage.  For example, this
                contains the path to the disk image file that
                corresponds to the virtual storage."
        ::= { vmStorageEntry 6 }

    vmStorageAccess OBJECT-TYPE
        SYNTAX       VirtualMachineStorageAccess
        MAX-ACCESS   read-only
        STATUS       current
        DESCRIPTION
                "The access permission of the virtual storage device."
        ::= { vmStorageEntry 7 }

    vmStorageMediaType OBJECT-TYPE
        SYNTAX       VirtualMachineStorageMediaType
        MAX-ACCESS   read-only
        STATUS       current
        DESCRIPTION
                "The media type of the virtual storage device."
        ::= { vmStorageEntry 8 }
```

```
   vmStorageMediaTypeString OBJECT-TYPE
       SYNTAX       SnmpAdminString (SIZE (0..255))
       MAX-ACCESS   read-only
       STATUS       current
       DESCRIPTION
               "A (detailed) textual string of the virtual storage
               media.  For example, this represents the specific driver
               name of the emulated media such as 'IDE' and 'SCSI'."
       ::= { vmStorageEntry 9 }

   vmStorageSizeUnit OBJECT-TYPE
       SYNTAX       Integer32 (1..2147483647)
       MAX-ACCESS   read-only
       STATUS       current
       DESCRIPTION
               "The multiplication unit for vmStorageDefinedSize and
               vmStorageAllocatedSize.  For example, when this value is
               1048576, the storage size unit for vmStorageDefinedSize
               and vmStorageAllocatedSize is MiB."
       ::= { vmStorageEntry 10 }

   vmStorageDefinedSize OBJECT-TYPE
       SYNTAX       Integer32 (-1|0..2147483647)
       MAX-ACCESS   read-only
       STATUS       current
       DESCRIPTION
               "The defined virtual storage size defined in the unit
               designated by vmStorageSizeUnit.  If this information is
               not available, this value shall be -1."
       ::= { vmStorageEntry 11 }

   vmStorageAllocatedSize OBJECT-TYPE
       SYNTAX       Integer32 (-1|0..2147483647)
       MAX-ACCESS   read-only
       STATUS       current
       DESCRIPTION
               "The storage size allocated to the virtual storage from
               a physical storage in the unit designated by
               vmStorageSizeUnit.  When the virtual storage is block
               device or raw file, this value and vmStorageDefinedSize
               are supposed to equal.  This value MUST NOT be different
               from vmStorageDefinedSize when vmStorageSourceType is
               'block' or 'raw'.  If this information is not available,
               this value shall be -1."
       ::= { vmStorageEntry 12 }

   vmStorageReadIOs OBJECT-TYPE
       SYNTAX       Counter64
```

```
        MAX-ACCESS   read-only
        STATUS       current
        DESCRIPTION
                "The number of read I/O requests.

                Discontinuities in the value of this counter can occur
                at re-initialization of the hypervisor, and
                administrative state (vmAdminState) changes of the
                virtual machine."
        ::= { vmStorageEntry 13 }

    vmStorageWriteIOs OBJECT-TYPE
        SYNTAX       Counter64
        MAX-ACCESS   read-only
        STATUS       current
        DESCRIPTION
                "The number of write I/O requests.

                Discontinuities in the value of this counter can occur
                at re-initialization of the hypervisor, and
                administrative state (vmAdminState) changes of the
                virtual machine."
        ::= { vmStorageEntry 14 }

    -- The virtual network interfaces on each virtual machine.
    vmNetworkTable OBJECT-TYPE
        SYNTAX       SEQUENCE OF VmNetworkEntry
        MAX-ACCESS   not-accessible
        STATUS       current
        DESCRIPTION
                "The conceptual table of virtual network interfaces
                attached to the virtual machine."
        ::= { vmObjects 8 }

    vmNetworkEntry OBJECT-TYPE
        SYNTAX       VmNetworkEntry
        MAX-ACCESS   not-accessible
        STATUS       current
        DESCRIPTION
                "An entry for one virtual network interfaces attached to
                the virtual machine."
        INDEX { vmIndex, vmNetworkIndex }
        ::= { vmNetworkTable 1 }

    VmNetworkEntry ::=
        SEQUENCE {
            vmNetworkIndex          VirtualMachineNetworkIndex,
            vmNetworkIfIndex        InterfaceIndexOrZero,
```

```
            vmNetworkParent         InterfaceIndexOrZero,
            vmNetworkModel          SnmpAdminString,
            vmNetworkPhysAddress    PhysAddress
        }

    vmNetworkIndex OBJECT-TYPE
        SYNTAX       VirtualMachineNetworkIndex
        MAX-ACCESS   not-accessible
        STATUS       current
        DESCRIPTION
                "A unique value identifying a virtual network interface
                allocated to the virtual machine."
        ::= { vmNetworkEntry 1 }

    vmNetworkIfIndex OBJECT-TYPE
        SYNTAX       InterfaceIndexOrZero
        MAX-ACCESS   read-only
        STATUS       current
        DESCRIPTION
                "The value of ifIndex which corresponds to this virtual
                network interface.  If this device is not represented in
                the ifTable, then this value shall be zero."
        ::= { vmNetworkEntry 2 }

    vmNetworkParent OBJECT-TYPE
        SYNTAX       InterfaceIndexOrZero
        MAX-ACCESS   read-only
        STATUS       current
        DESCRIPTION
                "The value of ifIndex which corresponds to the parent
                (i.e., physical) device of this virtual device on.  The
                value zero denotes this virtual device is not any child
                represented in the ifTable."
        ::= { vmNetworkEntry 3 }

    vmNetworkModel OBJECT-TYPE
        SYNTAX       SnmpAdminString (SIZE (0..255))
        MAX-ACCESS   read-only
        STATUS       current
        DESCRIPTION
                "A textual string containing the (emulated) model of
                virtual network interface.  For example, this value is
                'virtio' when the emulation driver model is virtio."
        ::= { vmNetworkEntry 4 }

    vmNetworkPhysAddress OBJECT-TYPE
        SYNTAX       PhysAddress
        MAX-ACCESS   read-only
```

```
        STATUS       current
        DESCRIPTION
                "The MAC address of the virtual network interface."
        ::= { vmNetworkEntry 5 }


    -- Notification definitions:

    vmPerVMNotificationsEnabled OBJECT-TYPE
        SYNTAX       TruthValue
        MAX-ACCESS   read-write
        STATUS       current
        DESCRIPTION
                "Indicates if notification generator will send
                notifications per virtual machine."
        ::= { vmObjects 9 }

    vmBulkNotificationsEnabled OBJECT-TYPE
        SYNTAX       TruthValue
        MAX-ACCESS   read-write
        STATUS       current
        DESCRIPTION
                "Indicates if notification generator will send
                notifications per set of virtual machines."
        ::= { vmObjects 10 }

    vmAffectedVMs OBJECT-TYPE
        SYNTAX       VirtualMachineList
        MAX-ACCESS   accessible-for-notify
        STATUS       current
        DESCRIPTION
                "A complete list of virtual machines whose state has
                changed.  This object is the only object sent with bulk
                notifications."
        ::= { vmObjects 11 }

    vmRunning NOTIFICATION-TYPE
        OBJECTS      {
                        vmName,
                        vmUUID,
                        vmOperState
                     }
        STATUS       current
        DESCRIPTION
                "This notification is generated when the operational
                state of a virtual machine has been changed to
                running(4) from some other state.  The other state is
                indicated by the included value of vmOperState."
```

```
        ::= { vmNotifications 1 }

    vmShutdown NOTIFICATION-TYPE
        OBJECTS      {
                        vmName,
                        vmUUID,
                        vmOperState
                     }
        STATUS       current
        DESCRIPTION
                "This notification is generated when the operational
                state of a virtual machine has been changed to
                shutdown(12) from some other state.  The other state is
                indicated by the included value of vmOperState."
        ::= { vmNotifications 2 }

    vmShuttingdown NOTIFICATION-TYPE
        OBJECTS      {
                        vmName,
                        vmUUID,
                        vmOperState
                     }
        STATUS       current
        DESCRIPTION
                "This notification is generated when the operational
                state of a virtual machine has been changed to
                shuttingdown(11) from some other state.  The other state
                is indicated by the included value of vmOperState."
        ::= { vmNotifications 3 }

    vmPaused NOTIFICATION-TYPE
        OBJECTS      {
                        vmName,
                        vmUUID,
                        vmOperState
                     }
        STATUS       current
        DESCRIPTION
                "This notification is generated when the operational
                state of a virtual machine has been changed to
                paused(9) from some other state.  The other state is
                indicated by the included value of vmOperState."
        ::= { vmNotifications 4 }

    vmSuspending NOTIFICATION-TYPE
        OBJECTS      {
                        vmName,
                        vmUUID,
```

```
                    vmOperState
                }
        STATUS      current
        DESCRIPTION
                "This notification is generated when the operational
                state of a virtual machine has been changed to
                suspending(6) from some other state.  The other state is
                indicated by the included value of vmOperState."
        ::= { vmNotifications 5 }

    vmSuspended NOTIFICATION-TYPE
        OBJECTS     {
                    vmName,
                    vmUUID,
                    vmOperState
                }
        STATUS      current
        DESCRIPTION
                "This notification is generated when the operational
                state of a virtual machine has been changed to
                suspended(7) from some other state.  The other state is
                indicated by the included value of vmOperState."
        ::= { vmNotifications 6 }

    vmResuming NOTIFICATION-TYPE
        OBJECTS     {
                    vmName,
                    vmUUID,
                    vmOperState
                }
        STATUS      current
        DESCRIPTION
                "This notification is generated when the operational
                state of a virtual machine has been changed to
                resuming(8) from some other state.  The other state is
                indicated by the included value of vmOperState."
        ::= { vmNotifications 7 }

    vmMigrating NOTIFICATION-TYPE
        OBJECTS     {
                    vmName,
                    vmUUID,
                    vmOperState
                }
        STATUS      current
        DESCRIPTION
                "This notification is generated when the operational
                state of a virtual machine has been changed to
```

```
                migrating(10) from some other state.  The other state is
                indicated by the included value of vmOperState."
        ::= { vmNotifications 8 }

    vmCrashed NOTIFICATION-TYPE
        OBJECTS      {
                        vmName,
                        vmUUID,
                        vmOperState
                     }
        STATUS       current
        DESCRIPTION
                "This notification is generated when a virtual machine
                has been crashed.  The previos state of the virtual
                machine is indicated by the included value of
                vmOperState."
        ::= { vmNotifications 9 }

    vmBlocked NOTIFICATION-TYPE
        OBJECTS      {
                        vmName,
                        vmUUID,
                        vmOperState
                     }
        STATUS       current
        DESCRIPTION
                "This notification is generated when the operational
                state of a virtual machine has been changed to
                blocked(5).  The previos state of the virtual machine is
                indicated by the included value of vmOperState."
        ::= { vmNotifications 10 }

    vmDeleted NOTIFICATION-TYPE
        OBJECTS      {
                        vmName,
                        vmUUID,
                        vmOperState,
                        vmPersistent
                     }
        STATUS       current
        DESCRIPTION
                "This notification is generated when a virtual machine
                has been deleted. The prior state of the virtual
                machine is indicated by the included value of
                vmOperState."
        ::= { vmNotifications 11 }

    vmBulkRunning NOTIFICATION-TYPE
```

```
        OBJECTS      {
                        vmAffectedVMs
                     }
        STATUS       current
        DESCRIPTION
                "This notification is generated when the operational
                state of one or more virtual machine has been changed to
                running(4) from a all prior states except for
                running(4).  Management stations are encouraged to
                subsequently poll the subset of virtual machines of
                interest for vmOperState."
        ::= { vmNotifications 12 }

    vmBulkShuttingdown NOTIFICATION-TYPE
        OBJECTS      {
                       vmAffectedVMs
                     }
        STATUS       current
        DESCRIPTION
                "This notification is generated when the operational
                state of one or more virtual machine has been changed to
                shuttingdown(11) from a state other than
                shuttingdown(11).  Management stations are encouraged to
                subsequently poll the subset of virtual machines of
                interest for vmOperState."
        ::= { vmNotifications 13 }

    vmBulkShutdown NOTIFICATION-TYPE
        OBJECTS      {
                       vmAffectedVMs
                     }
        STATUS       current
        DESCRIPTION
                "This notification is generated when the operational
                state of one or more virtual machine has been changed to
                shutdown(12) from a state other than shutdown(12).
                Management stations are encouraged to subsequently poll
                the subset of virtual machines of interest for
                vmOperState."
        ::= { vmNotifications 14 }

    vmBulkPaused NOTIFICATION-TYPE
        OBJECTS      {
                        vmAffectedVMs
                     }
        STATUS       current
        DESCRIPTION
                "This notification is generated when the operational
```

```
                state of one or more virtual machines have been changed
                to paused(9) from a state other than paused(9).
                Management stations are encouraged to subsequently poll
                the subset of virtual machines of interest for
                vmOperState."
        ::= { vmNotifications 15 }


    vmBulkSuspending NOTIFICATION-TYPE
        OBJECTS      {
                        vmAffectedVMs
                     }
        STATUS       current
        DESCRIPTION
                "This notification is generated when the operational
                state of one or more virtual machines have been changed
                to suspending(6) from a state other than suspending(6).
                Management stations are encouraged to subsequently poll
                the subset of virtual machines of interest for
                vmOperState."
        ::= { vmNotifications 16 }

    vmBulkSuspended NOTIFICATION-TYPE
        OBJECTS      {
                        vmAffectedVMs
                     }
        STATUS       current
        DESCRIPTION
                "This notification is generated when the operational
                state of one or more virtual machines have been changed
                to suspended(7) from a state other than suspended(7).
                Management stations are encouraged to subsequently poll
                the subset of virtual machines of interest for
                vmOperState."
        ::= { vmNotifications 17 }

    vmBulkResuming NOTIFICATION-TYPE
        OBJECTS      {
                        vmAffectedVMs
                     }
        STATUS       current
        DESCRIPTION
                "This notification is generated when the operational
                state of one or more virtual machines have been changed
                to resuming(8) from a state other than resuming(8).
                Management stations are encouraged to subsequently poll
                the subset of virtual machines of interest for
                vmOperState."
```

```
        ::= { vmNotifications 18 }

    vmBulkMigrating NOTIFICATION-TYPE
        OBJECTS      {
                        vmAffectedVMs
                     }
        STATUS       current
        DESCRIPTION
                "This notification is generated when the operational
                state of one or more virtual machines have been changed
                to migrating(10) from a state other than migrating(10).
                Management stations are encouraged to subsequently poll
                the subset of virtual machines of interest for
                vmOperState."
        ::= { vmNotifications 19 }

    vmBulkCrashed NOTIFICATION-TYPE
        OBJECTS      {
                        vmAffectedVMs
                     }
        STATUS       current
        DESCRIPTION
                "This notification is generated when one or more virtual
                machines have been crashed.  Management stations are
                encouraged to subsequently poll the subset of virtual
                machines of interest for vmOperState."
        ::= { vmNotifications 20 }

    vmBulkBlocked NOTIFICATION-TYPE
        OBJECTS      {
                        vmAffectedVMs
                     }
        STATUS       current
        DESCRIPTION
                "This notification is generated when the operational
                state of one or more virtual machines have been changed
                to blocked(5) from a state other than blocked(5).
                Management stations are encouraged to subsequently poll
                the subset of virtual machines of interest for
                vmOperState."
        ::= { vmNotifications 21 }

    vmBulkDeleted NOTIFICATION-TYPE
        OBJECTS      {
                        vmAffectedVMs
                     }
        STATUS       current
        DESCRIPTION
```

```
                "This notification is generated when one or more virtual
                machines have been deleted.  Management stations are
                encouraged to subsequently poll the subset of virtual
                machines of interest for vmOperState."
        ::= { vmNotifications 22 }

    -- Compliance definitions:
    vmCompliances  OBJECT IDENTIFIER ::= { vmConformance 1 }
    vmGroups       OBJECT IDENTIFIER ::= { vmConformance 2 }

    vmFullCompliances MODULE-COMPLIANCE
        STATUS       current
        DESCRIPTION
                "Compliance statement for implementations supporting
                read/write access, according to the object definitions."
        MODULE     -- this module
        MANDATORY-GROUPS {
            vmHypervisorGroup,
            vmVirtualMachineGroup,
            vmCpuGroup,
            vmCpuAffinityGroup,
            vmStorageGroup,
            vmNetworkGroup
        }
        GROUP  vmPerVMNotificationOptionalGroup
        DESCRIPTION
                "Support for per-VM notifications is optional.  If not
                implemented then vmPerVMNotificationsEnabled must report
                false(2)."
        GROUP  vmBulkNotificationsVariablesGroup
        DESCRIPTION
                "Necessary only if vmPerVMNotificationOptionalGroup is
                implemented."
        GROUP  vmBulkNotificationOptionalGroup
        DESCRIPTION
                "Support for bulk notifications is optional.  If not
                implemented then vmBulkNotificationsEnabled must report
                false(2)."

        ::= { vmCompliances 1 }

    vmReadOnlyCompliances MODULE-COMPLIANCE
        STATUS       current
        DESCRIPTION
                "Compliance statement for implementations supporting
                only readonly access."
        MODULE     -- this module
        MANDATORY-GROUPS {
```

```
            vmHypervisorGroup,
            vmVirtualMachineGroup,
            vmCpuGroup,
            vmCpuAffinityGroup,
            vmStorageGroup,
            vmNetworkGroup
        }

        OBJECT vmAdminState
        MIN-ACCESS   read-only
        DESCRIPTION
                "Write access is not required."

        OBJECT vmCurCpuNumber
        MIN-ACCESS   read-only
        DESCRIPTION
                "Write access is not required."

        OBJECT vmMinCpuNumber
        MIN-ACCESS   read-only
        DESCRIPTION
                "Write access is not required."

        OBJECT vmMaxCpuNumber
        MIN-ACCESS   read-only
        DESCRIPTION
                "Write access is not required."

        OBJECT vmCurMem
        MIN-ACCESS   read-only
        DESCRIPTION
                "Write access is not required."

        OBJECT vmMinMem
        MIN-ACCESS   read-only
        DESCRIPTION
                "Write access is not required."

        OBJECT vmMaxMem
        MIN-ACCESS   read-only
        DESCRIPTION
                "Write access is not required."

        OBJECT vmCpuAffinity
        MIN-ACCESS   read-only
        DESCRIPTION
                "Write access is not required."
```

```
        OBJECT vmPerVMNotificationsEnabled
        MIN-ACCESS   read-only
        DESCRIPTION
                "Write access is not required."

        OBJECT vmBulkNotificationsEnabled
        MIN-ACCESS   read-only
        DESCRIPTION
                "Write access is not required."
        ::= { vmCompliances 2 }

    vmHypervisorGroup OBJECT-GROUP
        OBJECTS {
            vmHvSoftware,
            vmHvVersion,
            vmHvObjectID,
            vmHvUpTime,
            vmNumber,
            vmTableLastChange,
            vmPerVMNotificationsEnabled,
            vmBulkNotificationsEnabled
        }
        STATUS       current
        DESCRIPTION
                "A collection of objects providing insight into the
                hypervisor itself."
         ::= { vmGroups 1 }

    vmVirtualMachineGroup OBJECT-GROUP
        OBJECTS {
            -- vmIndex
            vmName,
            vmUUID,
            vmOSType,
            vmAdminState,
            vmOperState,
            vmAutoStart,
            vmPersistent,
            vmCurCpuNumber,
            vmMinCpuNumber,
            vmMaxCpuNumber,
            vmMemUnit,
            vmCurMem,
            vmMinMem,
            vmMaxMem,
            vmUpTime,
            vmCpuTime
        }
```

```
        STATUS       current
        DESCRIPTION
                "A collection of objects providing insight into the
                virtual machines) controlled by a hypervisor."
        ::= { vmGroups 2 }

    vmCpuGroup OBJECT-GROUP
        OBJECTS {
            -- vmCpuIndex,
            vmCpuCoreTime
        }
        STATUS       current
        DESCRIPTION
                "A collection of objects providing insight into the
                virtual machines) controlled by a hypervisor."
        ::= { vmGroups 3 }

    vmCpuAffinityGroup OBJECT-GROUP
        OBJECTS {
            -- vmCpuPhysIndex,
            vmCpuAffinity
        }
        STATUS       current
        DESCRIPTION
                "A collection of objects providing insight into the
                virtual machines) controlled by a hypervisor."
        ::= { vmGroups 4 }

    vmStorageGroup OBJECT-GROUP
        OBJECTS {
            -- vmStorageVmIndex,
            -- vmStorageIndex,
            vmStorageParent,
            vmStorageSourceType,
            vmStorageSourceTypeString,
            vmStorageResourceID,
            vmStorageAccess,
            vmStorageMediaType,
            vmStorageMediaTypeString,
            vmStorageSizeUnit,
            vmStorageDefinedSize,
            vmStorageAllocatedSize,
            vmStorageReadIOs,
            vmStorageWriteIOs
        }
        STATUS       current
        DESCRIPTION
                "A collection of objects providing insight into the
```

```
                virtual storage devices controlled by a hypervisor."
        ::= { vmGroups 5 }

    vmNetworkGroup OBJECT-GROUP
        OBJECTS {
            -- vmNetworkIndex,
            vmNetworkIfIndex,
            vmNetworkParent,
            vmNetworkModel,
            vmNetworkPhysAddress
        }
        STATUS       current
        DESCRIPTION
                "A collection of objects providing insight into the
                virtual network interfaces controlled by a hypervisor."
        ::= { vmGroups 6 }

    vmPerVMNotificationOptionalGroup NOTIFICATION-GROUP
        NOTIFICATIONS {
            vmRunning,
            vmShuttingdown,
            vmShutdown,
            vmPaused,
            vmSuspending,
            vmSuspended,
            vmResuming,
            vmMigrating,
            vmCrashed,
            vmBlocked,
            vmDeleted
        }
        STATUS       current
        DESCRIPTION
                "A collection of notifications for per-VM notification
                of changes to virtual machine state (vmOperState) as
                reported by a hypervisor."
        ::= { vmGroups 7 }

    vmBulkNotificationsVariablesGroup OBJECT-GROUP
        OBJECTS {
            vmAffectedVMs
        }
        STATUS       current
        DESCRIPTION
                "The variables used in vmBulkNotificationOptionalGroup
                virtual network interfaces controlled by a hypervisor."
        ::= { vmGroups 8 }
```

```
   vmBulkNotificationOptionalGroup NOTIFICATION-GROUP
       NOTIFICATIONS {
           vmBulkRunning,
           vmBulkShuttingdown,
           vmBulkShutdown,
           vmBulkPaused,
           vmBulkSuspending,
           vmBulkSuspended,
           vmBulkResuming,
           vmBulkMigrating,
           vmBulkCrashed,
           vmBulkBlocked,
           vmBulkDeleted
       }
       STATUS       current
       DESCRIPTION
               "A collection of notifications for bulk notification of
               changes to virtual machine state (vmOperState) as
               reported by a given hypervisor."
       ::= { vmGroups 9 }

   END
```

7. IANA Considerations

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

Descriptor	OBJECT IDENTIFIER value
vmMIB	{ mib-2 TBD }

8. Security Considerations

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

There are a number of managed objects in this MIB that may contain sensitive information. The objects in the vmHvSoftware and vmHvVersion list information about the hypervisor's software and version. Some may wish not to disclose to others which software they are running. Further, an inventory of the running software and versions may be helpful to an attacker who hopes to exploit software bugs in certain applications. Moreover, the objects in the vmTable, vmCpuTable, vmCpuAffinityTable, vmStorageTable and vmNetworkTable list information about the virtual machines and their virtual resource allocation. Some may wish not to disclose to others how many and what virtual machines they are operating.

It is thus important to control even GET access to these objects and possibly to even encrypt the values of these object when sending them over the network via SNMP. Not all versions of SNMP provide features for such a secure environment.

It is recommended that attention be specifically given to implementing the MAX-ACCESS clause in a number of objects, including vmAdminState, vmMinCpuNumber, vmMaxCpuNumber, vmMinMem, vmMaxMem, and vmCpuAffinity in scenarios that DO NOT use SNMPv3 strong security (i.e. authentication and encryption). Extreme caution must be used to minimize the risk of cascading security vulnerabilities when SNMPv3 strong security is not used. When SNMPv3 strong security is not used, these objects should have access of read-only, not read-create.

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

It is recommended that the implementers consider the security features as provided by the SNMPv3 framework. Specifically, the use of the User-based Security Model [RFC3414] and the View-based Access Control Model [RFC3415] is recommended.

It is then a customer/user responsibility to ensure that the SNMP entity giving access to an instance of this MIB, is properly

configured to give access to the objects only to those principals (users) that have legitimate rights to indeed GET or SET (change/ create/delete) them.

9. Acknowledgements

The authors like to thank Joe Marcus Clarke, Randy Presuhn, and David Black for providing helpful comments during the development of this specification.

Juergen Schoenwaelder was partly funded by Flamingo, a Network of Excellence project (ICT-318488) supported by the European Commission under its Seventh Framework Programme.

10. References

10.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC2578] McCloghrie, K., Ed., Perkins, D., Ed., and J. Schoenwaelder, Ed., "Structure of Management Information Version 2 (SMIv2)", STD 58, RFC 2578, April 1999.

[RFC2579] McCloghrie, K., Ed., Perkins, D., Ed., and J. Schoenwaelder, Ed., "Textual Conventions for SMIv2", STD 58, RFC 2579, April 1999.

[RFC2580] McCloghrie, K., Perkins, D., and J. Schoenwaelder, "Conformance Statements for SMIv2", STD 58, RFC 2580, April 1999.

[RFC2790] Waldbusser, S. and P. Grillo, "Host Resources MIB", RFC 2790, March 2000.

[RFC2863] McCloghrie, K. and F. Kastenholz, "The Interfaces Group MIB", RFC 2863, June 2000.

[RFC3413] Levi, D., Meyer, P., and B. Stewart, "Simple Network Management Protocol (SNMP) Applications", STD 62, RFC 3413, December 2002.

[RFC3414] Blumenthal, U. and B. Wijnen, "User-based Security Model (USM) for version 3 of the Simple Network Management Protocol (SNMPv3)", STD 62, RFC 3414, December 2002.

[RFC3415] Wijnen, B., Presuhn, R., and K. McCloghrie, "View-based Access Control Model (VACM) for the Simple Network Management Protocol (SNMP)", STD 62, RFC 3415, December 2002.

[RFC3418] Presuhn, R., "Management Information Base (MIB) for the Simple Network Management Protocol (SNMP)", STD 62, RFC 3418, December 2002.

[RFC4122] Leach, P., Mealling, M., and R. Salz, "A Universally Unique IDentifier (UUID) URN Namespace", RFC 4122, July 2005.

[RFC6933] Bierman, A., Romascanu, D., Quittek, J., and M. Chandramouli, "Entity MIB (Version 4)", RFC 6933,

May 2013.

10.2. Informative References

[RFC3410] Case, J., Mundy, R., Partain, D., and B. Stewart, "Introduction and Applicability Statements for Internet-Standard Management Framework", RFC 3410, December 2002.

Appendix A. State Transition Table

State	Action or (Event)	Next state	Notification
suspended	running	resuming	vmResuming | vmBulkResuming
suspending	(suspend operation completed)	suspended	vmSuspended | vmBulkSuspended
running	suspended	suspending	vmSuspending | vmBulkSuspending
	shutdown	shuttingdown	vmShuttingdown | vmBulkShuttingdown
	destroy	shutdown	vmShutdown | vmBulkShutdown
	(migration to other hypervisor initiated)	migrating	vmMigrating | vmBulkMingrating
resuming	(resume opeartion completed)	running	vmRunning | vmBulkRunning
paused	running	running	vmRunning | vmBulkRunning
shuttingdown	(shutdown operation completed)	shutdown	vmShutdown | vmBulkShutdown
shutdown	running	running	vmRunning | vmBulkRunning
	(if this state entry is created by a migration operation (*)	migrating	vmMigrating | vmBulkMigrating

	(deletion operation completed)	(no state)	vmDeleted | vmBulkDeleted
migrating	(migration from other hypervisor completed)	running	vmRunning | vmBulkRunning
	(migration to other hypervisor completed)	shutdown	vmShutdown | vmBulkShutdown
preparing	(preparation completed)	shutdown	vmShutdown | vmBulkShutdown
blocked	(blocking operation completed)	(previous state)	-
crashed	-	-	-
(any)	(blocking operation initiated)	blocked	vmBlocked | vmBulkBlocked
	(crashed)	crashed	vmCrashed | vmBulkCrashed
(no state)	(preparation initiated)	preparing	-
	(migrate from other hypervisor initiated)	shutdown (*)	vmShutdown | vmBulkShutdown

State transition table

Authors' Addresses

Hirochika Asai
The University of Tokyo
7-3-1 Hongo
Bunkyo-ku, Tokyo 113-8656
JP

Phone: +81 3 5841 6748
Email: panda@hongo.wide.ad.jp

Michael MacFaden
VMware Inc.

Email: mrm@vmware.com

Juergen Schoenwaelder
Jacobs University
Campus Ring 1
Bremen 28759
Germany

Email: j.schoenwaelder@jacobs-university.de

Yuji Sekiya
The University of Tokyo
2-11-16 Yayoi
Bunkyo-ku, Tokyo 113-8658
JP

Email: sekiya@wide.ad.jp

Keiichi Shima
IIJ Innovation Institute Inc.
3-13 Kanda-Nishikicho
Chiyoda-ku, Tokyo 101-0054
JP

Email: keiichi@iijlab.net

Tina Tsou
Huawei Technologies (USA)
2330 Central Expressway
Santa Clara CA 95050
USA

Email: tina.tsou.zouting@huawei.com

Cathy Zhou
Huawei Technologies
Bantian, Longgang District
Shenzhen 518129
P.R. China

Email: cathyzhou@huawei.com

Hiroshi Esaki
The University of Tokyo
7-3-1 Hongo
Bunkyo-ku, Tokyo 113-8656
JP

Phone: +81 3 5841 6748
Email: hiroshi@wide.ad.jp

Internet Engineering Task Force
Internet-Draft
Intended status: Informational
Expires: November 20, 2014

S. Jiang
Y. Yin
Huawei Technologies Co., Ltd
B. Carpenter
Univ. of Auckland
May 19, 2014

Network Configuration Negotiation Problem Statement and Requirements
draft-jiang-config-negotiation-ps-03

Abstract

This document describes a problem statement and general requirements for distributed autonomic configuration of multiple aspects of networks, in particular carrier networks. The basic model is that network elements need to negotiate configuration settings with each other to meet overall goals. The document describes a generic negotiation behavior model. The document also reviews whether existing management and configuration protocols may be suitable for autonomic networks.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on November 20, 2014.

Copyright Notice

Copyright (c) 2014 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (http://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents

carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction 2
2. Requirements and Application Scenarios for Network Devices Negotiation 3
2.1. Negotiation between downstream and upstream network devices 4
2.2. Negotiation between peer network devices 5
2.3. Negotiation between networks 5
2.4. Information and status queries among devices 6
2.5. Unavoidable configuration 6
3. Existing protocols 6
4. A Behavior Model of a Generic Negotiation Protocol 8
5. Security Considerations 12
6. IANA Considerations 13
7. Acknowledgements 13
8. Change Log [RFC Editor please remove] 13
9. Informative References 13
Authors' Addresses 15

1. Introduction

The success of IP and the Internet has made the network model very complicated, and networks have become larger and larger. The network of a large ISP typically contains more than a hundred thousand network devices which play many roles. The initial setup configuration, dynamic management and maintenance, troubleshooting and recovery of these devices have become a huge outlay for network operators. Particularly, these devices are managed by many different staff requiring very detailed training and skills. The coordination of these staff is also difficult and often inefficient. There are therefore increased requirements for autonomy in the networks. [I-D.boucadair-network-automation-requirements] is one of the attempts to describe such requirements. It listed a "requirement for a protocol to convey configuration information towards the managed entities". However, this document is going further by requiring a configuration negotiation protocol rather than only unidirectional provisioning.

Autonomic operation means network devices could decide configurations by themselves. More background on autonomic networking is given in [I-D.irtf-nmrg-autonomic-network-definitions] and

[I-D.irtf-nmrg-an-gap-analysis]. There are already many existing internal implementations or algorithms for a network device to decide or compute its configuration according to its own status, often referred to as device intelligence. In one particular area, routing protocols, distributed autonomic configuration is a well established mechanism. The question is how to extend autonomy to cover all kinds of configuration, not just routing tables.

However, in order to make right or good decisions, the network devices need to know more information than just routes from the relevant or neighbor devices. There are dependencies between such information and configurations. Currently, most of these configurations currently require centralised manual coordination. The basic model for this document is that in an autonomic network, devices will need to negotiate directly with one another to provide this coordination.

Today, there is no generic negotiation protocol that can be used to control decision processes among distributed devices or between networks. Proprietary network management systems are widely used but tend to be hierarchical systems ultimately relying on a console operator and a central database. An autonomic system needs to be less hierarchical and with less dependence on an operator. This requires network elements to negotiate directly with each other, with an absolute minimum or zero configuration data at the installation stage.

This document analyzes the requirements for a generic negotiation protocol in view of various application use cases, then gives considerations for detailed technical requirements for designing such a protocol. Some existing protocols are also reviewed as part of the analysis. A protocol behavior model, which may be used to define such a negotiation protocol, is also described.

Note in draft: the requirements analysis will need to be reviewed and completed after a number of use cases for autonomic networking have been documented.

2. Requirements and Application Scenarios for Network Devices Negotiation

Routing protocols are a typical autonomic model based on distributed devices. But routing is mainly one-way information announcement (in both directions), rather than bi-directional negotiation. Its only focus is reachability. The future networks need to be able to manage many more dimensions of the network, such as power saving, load balancing, etc. The current routing protocols only show simple link status, as up or down. More information, such as latency,

congestion, capacity, and particularly available throughput, is very helpful to get better path selection and utilization rate.

A negotiation model with no human intervention is needed when the coordination of multiple devices can provide better overall network performance.

A negotiation model provides a possibility for forecasting. A "dry run" becomes possible before the concrete configuration takes place.

Another area is tunnel management, with automatic setup, maintenance, and removal. A related area is ad hoc routes, without encapsulation, to handle specific traffic flows (which might be regarded as a form of software defined networking).

Negotiation of security mechanisms, for example to determine the strongest possible protection for a given link, is another example.

When a new user or device comes online, it might be necessary to set up resources on multiple relevant devices, coordinated and matched to each other so that there is no wasted resource. Security settings might also be needed to allow for the new user/device.

Status information and traffic metrics need to be shared between nodes for dynamic adjustment of resources.

Troubleshooting should be as automatic as possible. Although it is far from trivial, there is a need to detect the "real" breakdown amongst many alerts, and then take action to reconfigure the relevant devices. Again, routing protocols have done this for many years, but in an autonomic network it is not just routing that needs to reconfigure itself after a failure.

2.1. Negotiation between downstream and upstream network devices

The typical scenario is that there is a new access gateway, which could be a wireless base station, WiFi hot spot, Data Center switch, VPN site switch, enterprise CE, home gateway, etc. When it is plugged into the network, bi-direction configuration/control is needed. The upstream network needs to configure the device, its delegated prefix(es), DNS server, etc. For this direction, DHCP might be suitable and sufficient. However, there is another direction: the connection of downstream devices also needs to trigger the upstream devices, for example the provider edge, to create a corresponding configuration, by setting up a new tunnel, service, authentication, etc.

Furthermore, after the communication between gateway and provider has been established, the devices would like to optimize their configurations interactively according to dynamic link status or performance measurements, power consumption, etc. For dynamic management and maintenance, there are many other network events that downstream network devices may need to report to upstream network devices and then initiate some configuration change on these upstream devices. Currently, these kinds of synchronizing operations require the involvement of human operators.

Similar requirements can also appear between other types of downstream and upstream network devices.

2.2. Negotiation between peer network devices

Within a large network, in many segments, there are network devices that are in equivalent positions. They have a peer rather than hierarchical relationship. There may be many horizontal traffic flows or tunnels between them. In order to make these connections efficient, their configurations (for example, quality of service parameters) have to match each other. Any change of a device's configuration may require synchronizing with its peer network devices.

However, in many cases the peer network devices may not be able to make the exact changes as requested. Instead, another slightly different change may be the best choice for optimal performance. In order to decide on this best choice, multiple rounds of information exchange between peers may be necessary. This should be done without requiring the involvement of human operators. To provide this ability, a mechanism for network devices to be able to negotiate with each other is needed.

2.3. Negotiation between networks

A network may announce some information about its internal capabilities to connected peer networks, so that the peer networks can react accordingly. BGP routing information is a simple example.

Beyond reachability, more information may enable better coordination among networks. Examples include traffic engineering among multiple connections between two networks, particularly when these connections are geographically distributed; dynamic capacity adjustment to match changing traffic from a peer network; dynamic establishment and adjustment of differentiated service classes to support Service Level Agreements; and so on.

2.4. Information and status queries among devices

In distributed routers, many data such as status indicators or traffic measurements are dynamically changing. These may be the triggers for subsequent negotiation. For example, assume there are two routers A and B sharing traffic load. Router A may request the traffic situation of router B, then start negotiation, such as requesting router B to handle all the traffic so that router A can enter power-saving mode. Another example is that a device may request its neighbor to send a forecast or dry-run result based on a given potential configuration change. Then, the initiating router can evaluate whether the potential configuration change would meet its original target.

2.5. Unavoidable configuration

Even with autonomic negotiation, some initial configuration data cannot be avoided in some devices. A design goal is to reduce this to an absolute minimum. This information may have to be pre-configured on the device before it has been deployed physically, and is typically static. A preliminary list of unavoidable configuration data is:

- Authentic identity for each device. This may be a public key or a signed certification. This is necessary to protect the infrastructure against unauthorized replacement of equipment.

- The role and function and capability of the device. The role and function may depend on the network planning. The capability is typically decided by the hardware.

- On the network edge, the routers may need to be configured with the identity of each peer provider, and their entitlements to service.

Ideally, everything else (topology, link capacity, address prefixes, shared resources, customer authentication and authority, etc.) will be discovered or negotiated autonomously according to general policy for various negotiated objective.

3. Existing protocols

Routing protocols are mainly one-way information announcements. The receiver makes independent decisions based on the received information and there is no direct feedback information to the announcing peer. This remains true even though the protocol is used in both directions between peer routers; there is no negotiation, and each peer runs its route calculations independently.

Simple Network Management Protocol (SNMP) [RFC3416] uses a command/ response model not well suited for peer negotiation. Network Configuration Protocol (NETCONF) [RFC6241] uses an RPC model that does allow positive or negative responses from the target system, but this is still not adequate for negotiation.

There are various existing protocols that have elementary negotiation abilities, such as Dynamic Host Configure Protocol for IPv6 (DHCPv6) [RFC3315], Neighbor Discovery (ND) [RFC4861], Port Control Protocol (PCP) [RFC6887], Remote Authentication Dial In User Service (RADIUS) [RFC2865], Diameter [RFC6733], etc. Most of them are configuration or management protocols. However, they either provide only a simple request/response model in a master/slave context or very limited negotiation abilities.

There are also signalling protocols with an element of negotiation. For example Resource ReSerVation Protocol (RSVP) [RFC2205] was designed for negotiating quality of service parameters along the path of a unicast or multicast flow. RSVP is a very specialised protocol aimed at end-to-end flows. However, it has some flexibility, having been extended for MPLS label distribution [RFC3209]. A more generic design is General Internet Signalling Transport (GIST) [RFC5971], but it is complex, tries to solve many problems, and is also aimed at per-flow signalling across many hops rather than at device-to-device signalling. However, we cannot yet exclude extended RSVP or GIST as a negotiation protocol.

It is worth noting that some of the above protocols have either an explicit information model describing their messages, or at least a flexible and extensible message format. A negotiation protocol will require such capabilities. One design consideration is whether to adopt an existing information model or to design a new one. Another consideration is whether to be able to carry some or all of the message formats used by the above protocols.

We now consider two protocols that are works in progress at the time of this writing. Firstly, RESTCONF [I-D.ietf-netconf-restconf] is a protocol intended to convey NETCONF information expressed in the YANG language via HTTP, including the ability to transit HTML intermediaries. While this is a powerful approach in the context of centralised configuration of a complex network, it is not well adapted to efficient interactive negotiation between peer devices, especially simple ones that are unlikely to include YANG processing already.

Secondly, we consider HomeNet Control Protocol (HNCP) [I-D.ietf-homenet-hncp]. This is defined as "a minimalist state

synchronization protocol for Homenet routers." Specific features are:

- Every participating node has a unique node identifier.
- "HNCP is designed to operate between directly connected neighbors on a shared link using link-local IPv6 addresses."
- Currency of state is maintained by spontaneous link-local multicast messages.
- HNCP discovers and tracks link-local neighbours.
- HNCP messages are encoded as a sequence of TLV objects, sent over UDP.
- Authentication depends on a signature TLV (assuming public keys are associated with node identifiers).
- The functionality covered initially includes: site border discovery, prefix assignment, DNS namespace discovery, and routing protocol selection.

Clearly HNCP does not completely meet the needs of a general negotiation protocol, especially due to its limitation to link-local messages and its strict dependency on IPv6, but at the minimum it is a very interesting test case for this style of interaction between devices without needing a central authority.

4. A Behavior Model of a Generic Negotiation Protocol

This section describes a behavior model and some considerations for designing a generic negotiation protocol, which would act as a platform for different negotiation objectives.

- A generic platform

  The design of the network device protocol is desired to be a generic platform, which is independent from the negotiation contents. It should only take care of the general intercommunication between negotiation counterparts. The negotiation contents will vary according to the various negotiation objectives and the different pairs of negotiating counterparts.

- Security infrastructure and trust relationship

Because this negotiation protocol may directly cause changes to device configurations and bring significant impacts to a running network, this protocol must be based on a restrictive security infrastructure. It should be carefully managed and monitored so that every device in this negotiation system behaves well and remains well protected.

On the other hand, a limited negotiation model might be deployed based on a limited trust relationship. For example, between two administrative domains, devices might also exchange limited information and negotiate some particular configurations based on a limited conventional or contractual trust relationship.

- A uniform pattern for negotiation contents

  The negotiation contents should be defined according to a uniform pattern. They could be carried either in TLV (Type, Length and Value) format or in payloads described by a flexible language, like XML. A protocol design should choose one of these two. The format must be extensible for unknown future requirements. As noted above, an existing information model and existing message format(s) should be considered.

- A simple initiator/responder model

  Multi-party negotiations are too complicated to be modeled and there may be too many dependencies among the parties to converge efficiently. A simple initiator/responder model is more feasible and could actually complete multiple-party negotiations by indirect steps. Naturally this process must be guaranteed to terminate and must contain tie-breaking rules.

- Organizing of negotiation content

  Naturally, the negotiation content should be organized according to the relevant function or service. The content from different functions or services should be kept independent from each other. They should not be combined into a single option or single session because these contents may be negotiated with different counterparts or may be different in response time.

- Topology neighbor device discovery

Every network device that supports the negotiation protocol is a responder and always listens to a well-known (UDP?) port. A well-known link-local multicast address should be defined for discovery purposes. Upon receiving a discovery or request message, the recipient device should return a message in which it either indicates itself as a proper negotiation counterpart or diverts the initiator towards another more suitable device.

- Self aware network device

  Every network device should be pre-configured with its role and functions and be aware of its own capabilities. The roles may be only distinguished because of network behaviors, which may include forwarding behaviors, aggregation properties, topology location, bandwidth, tunnel or translation properties, etc. The role and functions may depend on the network planning. The capability is typically decided by the hardware or firmware. These parameters are the foundation of the negotiation behavior of a specific device.

- Requests and responses in negotiation procedures

  The initiator should be able to negotiate with its relevant negotiation counterpart devices, which may be different according to the negotiation objective. It may request relevant information from the negotiation counterpart so that it can decide its local configuration to give the most coordinated performance. It may request the negotiation counterpart to make a matching configuration in order to set up a successful communication with it. It may request certain simulation or forecast results by sending some dry run conditions.

  Beyond the traditional yes/no answer, the responder should be able to reply with a suggested alternative if its answer is ’no’. This would start a bi-directional negotiation ending in a compromise between the two devices.

- Convergence of negotiation procedures

  The negotiation procedure should move towards convergent results. It means that when a responder makes a suggestion of a changed condition in a negative reply, it should be as close as possible to the original request or previous suggestion. The suggested value of the third or later negotiation steps should be chosen between the suggested values from the last two negotiation steps.

In any case there must be a mechanism to guarantee rapid convergence in a small number of steps.

- Dependencies of negotiation

  In order to decide a configuration on a device, the device may need information from neighbors. This can be reached through the above negotiation procedure. However, a given item in a neighbor may depend on other information from its own neighbors, which may need another negotiation procedure to obtain or decide. Therefore, there are dependencies among negotiation procedures. There need to be clear boundaries and convergence mechanisms for these negotiation dependencies. Also some mechanisms are needed to avoid loop dependencies.

- End of negotiation

  A single negotiation procedure also needs ending conditions if it does not converge. A limited number of rounds, for example three, should be set on the devices. It may be an implementation choice or a pre-configurable parameter. However, the protocol design needs to clearly specify this, so that the negotiation can be terminated properly. In some cases, a timeout might be needed to end a negotiation.

- Failed negotiation

  There must be a well-defined procedure for concluding that a negotiation cannot succeed, and if so deciding what happens next (deadlock resolution, tie-breaking, or revert to best-effort service).

- Policy constraints

  There must be provision for general policy rules to be applied by all devices in the network (e.g., security rules, prefix length, resource sharing rules). However, policy distribution might not use the negotiation protocol itself.

- Management monitoring, alerts and intervention

  Devices should be able to report to a monitoring system. Some events must be able to generate operator alerts and some provision

for emergency intervention must be possible (e.g. to freeze negotiation in a mis-behaving device). These features may not use the negotiation protocol itself.

5. Security Considerations

This document does not include a detailed threat analysis for autonomic configuration, but it is obvious that a successful attack on autonomic nodes would be extremely harmful, as such nodes might end up with a completely undesirable configuration. A concrete protocol proposal will therefore require a threat analysis, and some form of strong authentication and, if possible, built-in protection against denial of service attacks.

Separation of network devices and user devices may become very helpful in this kind of scenario.

Also, security configuration itself should become autonomic whenever possible. However, in the security area at least, operator override of autonomic configuration must be possible for emergency use.

As noted earlier, a cryptographically authenticated identity for each device is needed in an autonomic network. It is not safe to assume that a large network is physically secured against interference or that all personnel are trustworthy. Each autonomic device should be capable of proving its identity and authenticating its messages. One approach would be to use a private/public key pair and sufficiently strong cryptography. Each device would generate its own private key, which is never exported from the device. The device identity and public key would be recorded in a network-wide database. The alternative of using symmetric keys (shared secrets) is less attractive, since it creates a risk of key leakage as well as a key management problem when devices are installed or removed.

Generally speaking, no personal information is expected to be involved in the negotiation protocol, so there should be no direct impact on personal privacy. Nevertheless, traffic flow paths, VPNs, etc. may be negotiated, which could be of interest for traffic analysis. Also, carriers generally want to conceal details of their network topology and traffic density from outsiders. Therefore, since insider attacks cannot be prevented in a large carrier network, the security mechanism for the negotiation protocol needs to provide message confidentiality.

6. IANA Considerations

This draft does not request any IANA action.

7. Acknowledgements

The authors want to thank Zhenbin Li, Bing Liu for valuable comments.

This document was produced using the xml2rfc tool [RFC2629].

8. Change Log [RFC Editor please remove]

draft-jiang-negotiation-config-ps-03, text improvements, added RESTCONF and HNCP to existing protocols, 2014-05-19.

draft-jiang-negotiation-config-ps-02, text improvements, added extra existing protocols, 2014-01-19.

draft-jiang-negotiation-config-ps-01, add more requirements, and add more considerations for behavior model, 2013-10-11.

draft-jiang-negotiation-config-ps-00, original version, 2013-06-29.

9. Informative References

[I-D.boucadair-network-automation-requirements]
Boucadair, M., Jacquenet, C., and L. Contreras, "Requirements for Automated (Configuration) Management", draft-boucadair-network-automation-requirements-03 (work in progress), February 2014.

[I-D.ietf-homenet-hncp]
Stenberg, M. and S. Barth, "Home Networking Control Protocol", draft-ietf-homenet-hncp-00 (work in progress), April 2014.

[I-D.ietf-netconf-restconf]
Bierman, A., Bjorklund, M., Watsen, K., and R. Fernando, "RESTCONF Protocol", draft-ietf-netconf-restconf-00 (work in progress), March 2014.

[I-D.irtf-nmrg-an-gap-analysis]
Behringer, M., Carpenter, B., and S. Jiang, "Gap Analysis for Autonomic Networking", draft-irtf-nmrg-an-gap-analysis-00 (work in progress), April 2014.

[I-D.irtf-nmrg-autonomic-network-definitions]
Behringer, M., Pritikin, M., Bjarnason, S., Clemm, A., Carpenter, B., Jiang, S., and L. Ciavaglia, "Autonomic Networking - Definitions and Design Goals", draft-irtf-nmrg-autonomic-network-definitions-00 (work in progress), December 2013.

[RFC2205] Braden, B., Zhang, L., Berson, S., Herzog, S., and S. Jamin, "Resource ReSerVation Protocol (RSVP) -- Version 1 Functional Specification", RFC 2205, September 1997.

[RFC2629] Rose, M., "Writing I-Ds and RFCs using XML", RFC 2629, June 1999.

[RFC2865] Rigney, C., Willens, S., Rubens, A., and W. Simpson, "Remote Authentication Dial In User Service (RADIUS)", RFC 2865, June 2000.

[RFC3209] Awduche, D., Berger, L., Gan, D., Li, T., Srinivasan, V., and G. Swallow, "RSVP-TE: Extensions to RSVP for LSP Tunnels", RFC 3209, December 2001.

[RFC3315] Droms, R., Bound, J., Volz, B., Lemon, T., Perkins, C., and M. Carney, "Dynamic Host Configuration Protocol for IPv6 (DHCPv6)", RFC 3315, July 2003.

[RFC3416] Presuhn, R., "Version 2 of the Protocol Operations for the Simple Network Management Protocol (SNMP)", STD 62, RFC 3416, December 2002.

[RFC4861] Narten, T., Nordmark, E., Simpson, W., and H. Soliman, "Neighbor Discovery for IP version 6 (IPv6)", RFC 4861, September 2007.

[RFC5971] Schulzrinne, H. and R. Hancock, "GIST: General Internet Signalling Transport", RFC 5971, October 2010.

[RFC6241] Enns, R., Bjorklund, M., Schoenwaelder, J., and A. Bierman, "Network Configuration Protocol (NETCONF)", RFC 6241, June 2011.

[RFC6733] Fajardo, V., Arkko, J., Loughney, J., and G. Zorn, "Diameter Base Protocol", RFC 6733, October 2012.

[RFC6887] Wing, D., Cheshire, S., Boucadair, M., Penno, R., and P. Selkirk, "Port Control Protocol (PCP)", RFC 6887, April 2013.

Authors' Addresses

Sheng Jiang
Huawei Technologies Co., Ltd
Q14, Huawei Campus, No.156 Beiqing Road
Hai-Dian District, Beijing, 100095
P.R. China

Email: jiangsheng@huawei.com

Yuanbin Yin
Huawei Technologies Co., Ltd
Q15, Huawei Campus, No.156 Beiqing Road
Hai-Dian District, Beijing, 100095
P.R. China

Email: yinyuanbin@huawei.com

Brian Carpenter
Department of Computer Science
University of Auckland
PB 92019
Auckland 1142
New Zealand

Email: brian.e.carpenter@gmail.com

Network Working Group
Internet-Draft
Intended status: Informational
Expires: August 18, 2014

W. Liu
C. Zhou
Huawei Technologies
Q. Sun
China Telecom
G. Leclanche
Viagenie

February 14, 2014

Openv6 Architecture for IPv6 Deployment
draft-liu-openv6-architecture-01

Abstract

IPv6 transition leads to costly end-to-end network upgrades and poses new challenges in terms of device management with a variety of transitional protocols.

This document provides a cost-effective and flexible unified IPv6 deployment by describing an architecture of a standard and programmatic manner for IPv6 deployment.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on August 18, 2014.

Copyright Notice

Copyright (c) 2014 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (http://trustee.ietf.org/license-info) in effect on the date of

publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction 2
2. Terminology 3
3. Motivation for OpenV6 Architecture 3
4. Overview of the OpenV6 Architecture 4
5. OpenV6 Considerations 5
6. Manageability Considerations 5
7. Security Considerations 5
8. IANA Considerations 5
9. Acknowledgements 6
10. References 6
10.1. Normative References 6
10.2. Informative References 6
Authors' Addresses 6

1. Introduction

The exhaustion of the IPv4 address space has been a practical problem that network carriers are facing today. Existing solutions such as IPv4 re-addressing and address reusing fail to fundamentally solve this problem. Instead, IPv6 is regarded as a complete and thorough solution to this problem.

To date, the adoption of IPv6 is progressing slowly. [Google-IPv6-Statistics] shows the statistics of IPv6 adoption. On one hand, IPv6 lacks support from applications. As a result, end users are reluctant to transition to IPv6 due to lack of attractive applications and competitive prices on IPv6. On the other hand, a large-scale IPv6 network as well as a stable and large IPv6 user group are the fundamental driving forces for evolving to IPv6.

The key to the above deadlock is that network carriers should take the initiative in constructing and developing an IPv6-friendly infrastructure, thus providing IPv6-based service access capabilities and actively nurturing the IPv6 adoption. The Openv6 and this document are focused on flexibly unifying the IPv6 transition mechanisms. The Openv6 provides an IPv6-friendly infrastructure to let the users decide for themselves when and how to start the IPv6 transition.

2. Terminology

3. Motivation for OpenV6 Architecture

Several motivations for the Openv6 are listed below. This list is not meant to be exhaustive and is provided for the sake of illustration.

It should be highlighted that the aim of this section is to provide some application examples for which the OpenV6 may be suitable: this also clearly states that such a model does not aim to replace existing IPv6 transition mechanisms but would apply to specific existing or future situations.

The Openv6 does not replace the existing IPv6 transition mechanisms in the network. Instead, it is compatible (or accommodate) existing and future IPv6 transition mechanisms.

Not all networks, servers and users will upgrade to IPv6 at the same pace along IPv6 transition. There will be many different scenarios, among which we can highlight the following ones:

Some regions will stay as IPv4-only networks (whenever transition is too costly or there are compelling technical reasons for not upgrading), and some regions will start as IPv6-only networks.

IPv6 end users accessing the IPv6 Internet via a service provider’s IPv4 network infrastructure.

IPv4 end users accessing the IPv4 Internet via a service provider’s IPv6 network infrastructure.

IPv6 end users accessing the IPv4 Internet.

According to these (and many others) different scenarios, and to the current status of network infrastructure, a number of different IPv6 transition technologies have been defined. For any device it becomes extremely hard to support them all at the same time, so addressing all the potential situations can become extremely costly, both in terms of CAPEX and OPEX.

The Openv6 provides an opportunity to build a unified approach to the different IPv6 transition technologies. With unified devices on the forwarding plane, packets are processed according to flow tables, in a way completely oblivious to the transition technology particular aspects.

4. Overview of the OpenV6 Architecture

This section gives an overview of the architecture of the Openv6.

The figure in Figure 1 shows the basic architecture of Openv6.

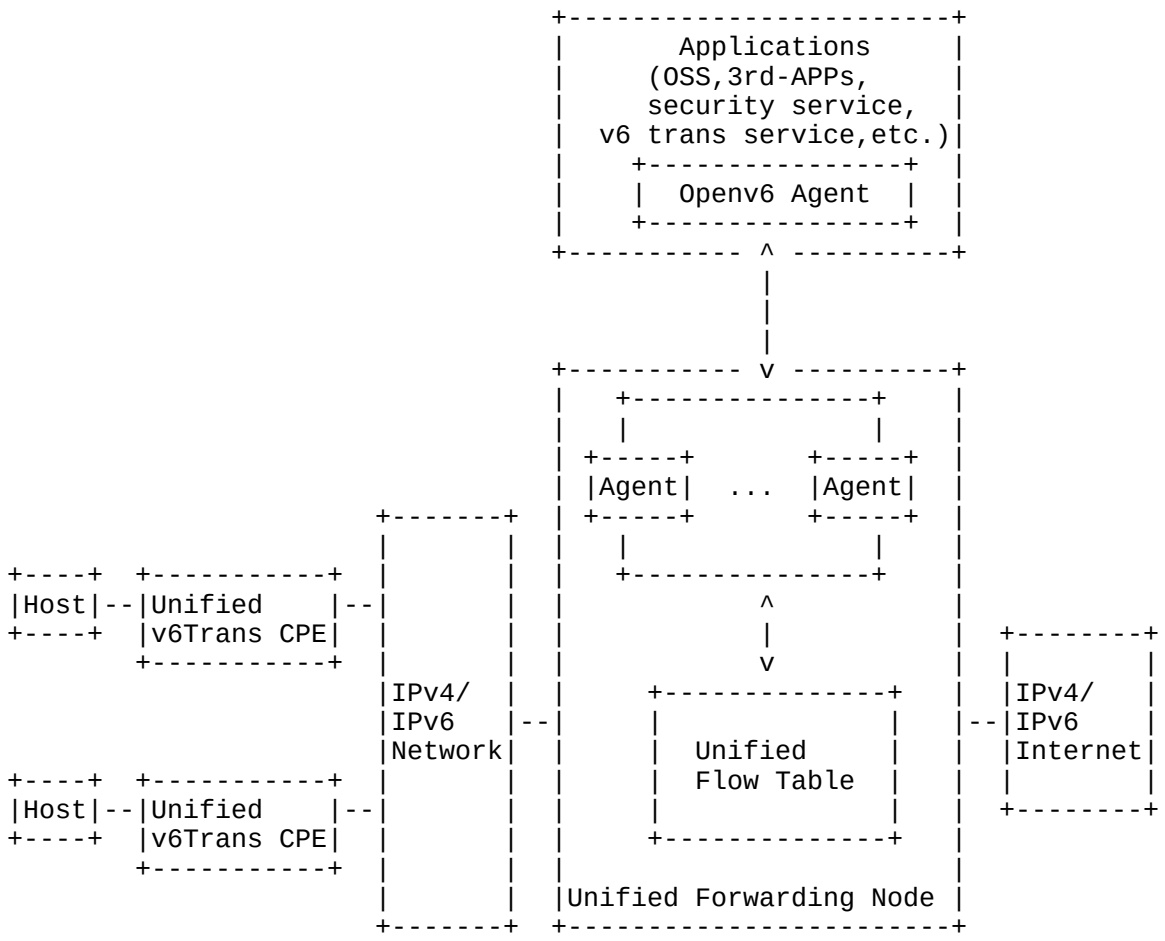


Figure 1: Architecture of OpenV6

Unified Forwarding Node: A forwarding node that handles incoming packets basing on the flow table. Examples of Forwarding Nodes can include:

A router that has an extended function module. The extended module handles incoming packets basing on the flow table of the module.

A server that runs vRouter or vSwitch.

A CGN that runs NAT, Tunnel En/De-capsulation functions.

A Forwarding Node may be locally managed, whether via CLI, SNMP, or NETCONF.

Unified Flow Table: The flow table is used for handling incoming packets of the forwarding node. The flow table can be updated by the application. If an incoming packet does not match any entry of the flow table, the packet will be delivered to the agent for generating new entries.

OpenV6 Agent: The OpenV6 agent interacts with the forwarding node to provide specified behavior for incoming packets via the flow table.

Applications: A network application that needs to manipulate the network to achieve its service requirements. Various IPv6 related services can be enabled by corresponding Applications such as:

OSS: can be considered as an application;

3rd-party APPs: IPv6 based 3rd-party applications;

Network security service: security related services such as savi

IPv6 transition service: transition mechanisms are are considered to be a variety of applications in OpenV6. The application can communicate with multiple forwarding node.

Agent: The agent interacts with the applications and the forwarding nodes. It can be implemented in the forwarding node for policies driven provisioning. There may be multiple agents in an forwarding node. Each agent executes a specific policy(for example, one agent for App-Lw4over6, one agent for NAT64, etc.)

5. OpenV6 Considerations

6. Manageability Considerations

7. Security Considerations

8. IANA Considerations

9. Acknowledgements

N/A.

10. References

10.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

10.2. Informative References

[Google-IPv6-Statistics]
Google, "Google IPv6 Statistics", <http://www.google.com/ipv6/statistics.html#tab=ipv6-adoption>.

[RFC6674] Brockners, F., Gundavelli, S., Speicher, S., and D. Ward, "Gateway-Initiated Dual-Stack Lite Deployment", RFC 6674, July 2012.

[RFC6888] Perreault, S., Yamagata, I., Miyakawa, S., Nakagawa, A., and H. Ashida, "Common Requirements for Carrier-Grade NATs (CGNs)", BCP 127, RFC 6888, April 2013.

Authors' Addresses

Will(Shucheng) Liu
Huawei Technologies
Bantian, Longgang District
Shenzhen 518129
P.R. China

Email: liushucheng@huawei.com

Cathy Zhou
Huawei Technologies
Bantian, Longgang District
Shenzhen 518129
P.R. China

Email: cathy.zhou@huawei.com

Qiong Sun
China Telecom
No.118 Xizhimennei street, Xicheng District
Beijing 100035
P.R. China

Email: sunqiong@ctbri.com.cn

Guillaume Leclanche
Viagenie
246 Aberdeen
Quebec, QC G1R 2E1
Canada

Phone: +1 418 656 9254
Email: guillaume.leclanche@viagenie.ca

Network Working Group
Internet-Draft
Intended status: Informational
Expires: August 17, 2014

Q. Sun
China Telecom
W. Liu
C. Zhou
Huawei Technologies
G. Leclanche
Viagenie
February 13, 2014

Problem Statement for Openv6 Scheme
draft-sun-openv6-problem-statement-01

Abstract

This document assesses the variety and complexity of IPv6 deployments, and proposes a new space of study to simplify the enablement of new IPv6 applications on an existing network. The document evaluates the identified technical gaps as well.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on August 17, 2014.

Copyright Notice

Copyright (c) 2014 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust’s Legal Provisions Relating to IETF Documents (http://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must

include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction 2
2. Terminology 3
3. Problem Extent and Existing Work 3
3.1. Variety of IPv6 deployment technologies 3
3.2. Complexity of IPv6 operation 5
3.2.1. End-to-End Network Management 5
3.2.2. Open Network Business Capabilities 6
3.3. Existing evaluations of the IPv6 Transition Landscape . . 7
4. Alternative Approach to IPv6 applications enablement 8
5. Existing protocols and methods for the alternate approach . . 9
6. Missing protocols and methods for the alternate approach . . 9
6.1. Dynamic devices forwarding table configuration 9
6.2. Address Management 9
7. Security Considerations 9
7.1. Source Address Validation and Traceback with Openv6 . . . 10
8. IANA Considerations 10
9. Authors 10
10. Acknowledgements 10
11. References 10
11.1. Normative References 10
11.2. Informative References 10
Authors' Addresses 12

1. Introduction

The exhaustion of the IPv4 address space has been a practical problem that providers are facing today. Network address migration to IPv6 is ongoing or upcoming throughout the world. However, IPv6 activation requires costly end-to-end network upgrades and different network scenarios will co-exist during IPv6 transition. In addition, the technologies deployed for the transition are suppose to be obsoleted once the transition is completed.

This document proposes a new approach to deploy and operate IPv6 applications on a network, whether related to transition technologies or purely native ones. Such a technology would allow to continue using the same equipments and operational practices for various deployment scenarios.

2. Terminology

3. Problem Extent and Existing Work

3.1. Variety of IPv6 deployment technologies

The IPv6 transition period contains three stages for IP Networks: IPv4-only, dual-stack and IPv6-only. The networks should support both IPv4 services and IPv6 services during each stage.[One-vision-for-IPv6]

There are multiple IPv6 transition technologies for different network scenarios (e.g. IPv4 network for IPv4/IPv6 user access, IPv6 network for IPv4/IPv6 user access, IPv4 servers for IPv6 visitors, etc.). Different network scenarios will co-exist during the IPv6 transition period, which means the devices implementing the IPv6 transition technology should support the array of technologies, or there has to be as many devices as technologies used in a given network. The following scenarios below will happen during the IPv6 transition period :

Scenario 1: An IPv6 host visits IPv6 servers via an IPv4 access network

Scenario 2: An IPv4 host visits IPv4 servers via an IPv4 NAT Dual-stack network

Scenario 3: An IPv6 host visits IPv6 servers via an IPv6 network

Scenario 4: An IPv4 host visits IPv4 servers via an IPv6 access network

Scenario 5: IPv4 host and IPv6 host interaction

Different transition mechanisms may have different impacts on user experience. For example, DS-Lite would have some impact due to address sharing compared to 6rd mechanisms, and NAT64 would have extra impact due to ALG issue. An operator having a diverse customer base might have to deploy different transition technologies for a given scenario depending on the required user experience. This implies that it is useful to support multiple transition mechanisms in the same area, and preferably on the same transition devices.

Another use case is that multiple scenarios may exist in the same stage. For example, if there are both IPv6-only devices and IPv4-only host in the same area with limited public IPv4 address, both NAT64 and NAT44 (or DS-Lite) are required to achieve IPv4 service connectivity.

The current implementations normally use a separate instance for each mechanism, and additional policies need to be applied when running multiple mechanisms in one device. Some have a limitation on the number of policies that can be configured in one device, while some have restrictions regarding the resource occupation (e.g. one transition instance will use a static amount of memory). The major challenges of IPv6 deployment mainly lie in two aspects:

The need to implement different IPv6 transition technologies in the same hardware and the need to support this by upgrading network devices as little as possible.

The need to hop over legacy infrastructures which are not IPv6 enabled, costly or impossible to upgrade.

The issues are:

1. How to support multiple transition mechanisms in a cost-efficient and flexible way ?

2. How to easily identify the transition type of different subscribers ?

A random operator will most likely not go through each scenario one by one. For example, some operators may start from scenario 1, and some may start directly from scenario 2 or scenario 4. However, since the target scenario is the IPv6-only access network, a single operator will be confronted to multiple scenarios on the long term.

In such a case, the operator should either upgrade existing devices to support new features, or replace them with new ones. In particular, when the operator’s network consists of devices from different vendors, it is difficult to guarantee that all the legacy devices can be upgraded at the same time. This is costly and operationally complicated.

We call Transition Data Plane (TDP) the data forwarding plane of the operator network during the whole transition period. Issues that can be identified to improve the situation are:

1. How to manipulate Transition Data Plane with different modes?

2. How to identify the capabilities of different transition devices ?

3. How does the Transition Data Plane identify different modes in the unified platform ?

3.2. Complexity of IPv6 operation

3.2.1. End-to-End Network Management

3.2.1.1. Scattered Address Pool Management

When operators are facing the IPv4 address shortage problem, the remaining IPv4 address pools are usually quite scattered. It is quite complicated for an operator to manage scattered address pools in many transition devices. The situation will become even worse when multiple transition mechanisms in the same device need to be configured with different address pools. Besides, the occupation of the address pools may vary during different transition periods: when there is not many IPv6-enabled services and IPv6-enabled devices, IPv4 traffic will still represent a great portion of the total traffic, while in the later stage of IPv6 transition, IPv4 traffic will decrease and the amount of allocated IPv4 addresses may decrease as well, depending on customer requirements.

A solution could be to manage the address pools centrally. Different transition mechanisms can require the address pools on-demand. For example, when one transition mechanism is running out of the current address pools, it may request a additional address pool. It can also release the address pools that it is not using any longer. In this way, operators do not need to configure the address pools one by one manually and it also helps using the address pools more efficiently.

Fixing this problem implies solving those issues:

1. How to configure the address pools for different mechanisms ?

2. How to collect the current status of address pool usage ?

3.2.1.2. Source Address Validation and Traceback with Openv6

It has been long known the IPv4/IPv6 transition makes the tracking and validating of source IP address challenging. Whenever an IPvX packet is translated into an IPvY packet, a major change happens to the IP packet, which brings new issues:

1. How to track the origin of the IPvY packet which is actually in the IPvX world?

2. How to validate the IPvX packet at the edge of the IPvY world to prevent possible spoofing?

3. How to protect the IPvY address from being spoofed in the IPvY world?

SAVI[RFC7039] defines the source address validation solutions for both IPv4 and IPv6, but doesn't cover the scenario where an IPv4/IPv6 transition technology is used in the network. Currently designing a solution for the transition scenario is not an easy task. There are two main challenges:

1. the diversity of IPv4/IPv6 transition mechanisms. There have been a number of transition mechanism. Moreover, new transition mechanisms may be standardized in the future. It would be complex for a SAVI solution to understand each transition mechanism. An unified abstraction of the transition mechanisms (for example, an abstract Openv6 Transition Data Plan (TDP)) and a set of unified open interfaces should be provided by Openv6 to the SAVI solution for the transition scenario. Then the SAVI solution could know the correspondences between the two IP protocols without having to inspect each packet or keep heavy state locally. The SAVI solution can then generate filtering rules and process tracking.

2. the inflexibility of SAVI. Currently SAVI solutions are tightly associated with address assignment mechanisms. It should be noted that each IPv4/IPv6 transition mechanism actually introduce a new mechanism to assign valid IPv4/IPv6 addresses. Based on the current model of SAVI, the SAVI solution for the transition scenario should be able to track the address translation in all the transition mechanism. Such a SAVI solution is heavy and costly for switches. The SAVI solution should introduce flexibility in rule generation similarly as Openv6, which offloading the complexity from network devices to a controller.

3.2.2. Open Network Business Capabilities

3.2.2.1. Dynamic QoS guarantee in IPv6 transition period

Traditionally, almost all bandwidth on the Internet is shared, or with a pre-configured QoS class. However, since the QoS requirements by different applications are not always the same, the subscribers should either waste money by paying for a higher bandwidth service, or can not get qualified service when needed. Therefore, currently, operators are tending to provide more dynamic QoS guarantee for subscribers so that they may apply for a higher bandwidth on-demand when they needed, or specific QoS guarantee can be applied for a certain amount of applications. In this case, the QoS adjustment platform is needed to pass the QoS adjustment request from subscribers or application servers dynamically.

In IPv6 transition period, the situation will become more complicated. When CGNs are introduced in the network, ip address and port will change during the translation or tunnelling process. For

some solutions, e.g. NAT444, DS-Lite, etc., the mappings might be different for different sessions.

In this case, the QoS adjustment platform should have the ability to pass and acquire QoS requirements for certain mappings in the CGNs. Therefore, more flexibility should be introduced in the network to load the dynamic QoS requests to the forwarding devices, no matter whether it is a tunnelling or translating mapping.

3.2.2.2. Coordinated NAT translation

Traditionally, most peer-to-peer applications would deploy relays by their own to achieve NAT traversal. They may use different kinds of ways e.g. TURN, STUN, or use some private protocols for their own purpose. It would not only cost a lot for applications deploy multiple relays, but also introduces a lot of complexity for newly emerging applications. In addition, in IPv6 transition period, there would be more CGNs than before which might make it more difficult for applications to achieve NAT traversal.

However, when operators have deployed some kinds of CGNs in their network, it is reasonable for operators to provide NAT traversal service for third-party applications so that the applications do not need to deploy the relays by their own. For example, the third-party application may require the CGN with the transport address, reflect address, etc., and then choose the one to use for the specific NAT situation. It can also be applied when IPv6 client communicates with iPv4 client with similar procedure. In this case, a centralized controller is needed to acquire the requests from third-party applications and form the specific mappings for them.

3.3. Existing evaluations of the IPv6 Transition Landscape

This paragraph references work done at the IETF or to describe the complex landscape of transition technologies.

The different network environments (architecture, scale, services deployed, varying IP traffic) cause a variety of IPv6 transition technologies for different operators. This section analyses the current and future coexistence of IPv6 transition technologies situation as well as the issues behind it.

Since IPv6 was proposed, there have been a couple of RFCs and on-going documents in IETF, as listed in the table below.

status	number	documents
RFC	8 or more	[RFC5571], [RFC6333], [RFC6674], [RFC5969], [RFC6219], [RFC6535], [RFC6654], [RFC6145], ...
WG draft	6 or more	[I-D.ietf-softwire-4rd], [I-D.ietf-softwire-map], [I-D.ietf-softwire-map-t], [I-D.ietf-softwire-public-4over6], [I-D.ietf-softwire-lw4over6], [I-D.ietf-v6ops-464xlat], ...
Active draft	several	...

Table 1: A Table of IPv6 Transition Technologies @ IETF

The situation described above depicts the difficulty of selecting appropriate IPv6 transition technologies for the carriers. Moreover, according to [SD-NAT], there are multiple stages during the whole IPv6 transition period, and a variety of technologies and equipments are used during different IPv6 transition stages. To protect the user experience and the early investment, an operator will not upgrade its network directly to the final stage of IPv6 transition. During different IPv6 transition stages, an operator needs different technologies in different stages. Thus, a method that is able to implement different IPv6 transition technologies in the same hardware is crucial, to avoid repeated investments.

4. Alternative Approach to IPv6 applications enablement

Finally an IP Network is simply an interconnection of various IPv4- and IPv6-aware devices over some transport. From a payload point of view, there is no need to wonder how the packet got to the destination (security aspects are reserved). Removing the complexity of the transport from the IP-aware devices, by simply considering it as a hop-by-hop "encapsulation" would simplify some situations and bring more flexibility for new applications.

The alternative approach proposed here is to put the IPv6 forwarding rules into the devices by a dynamic configuration protocol like Netconf, depending on the application requirements. Those forwarding rules could for example require a change of encapsulation (e.g. from IPv6oEthernet to IPv6oIPv4oEthernet), or an IP protocol change (e.g. apply a NAT64 translation). A central management server would be able to coordinate this configuration and push it adequately on the forwarding devices.

Today, the configuration of these encapsulation or translations is done manually and is not controlled in a coordinated and standard way. The goal of the application-based approach is to allow the operator to have both the flexibility and full control on what technologies have to be used and when to help with its IPv6 transition process.

5. Existing protocols and methods for the alternate approach

The proposed approach would have impact on layer 3, and maybe 4. Hence there is no need to change anything to Layer 1-2 protocols and techniques.

Higher layer applications are not impacted either as the network forwarding is transparent to them.

The proposed approach requires a dynamic configuration protocol for network devices, to update the forwarding table accordingly. Protocols like Netconf (add ref) or Openflow (add ref) are already existing to achieve this goal. Thanks to their openness, they can easily be extended to support it.

6. Missing protocols and methods for the alternate approach

The authors have identified some missing pieces to be able to use the technology in a fully standard way.

6.1. Dynamic devices forwarding table configuration

The IETF standard for devices configuration is [RFC6241], the NETCONF Protocol. So it may be suitable for the forwarding table configuration of the openv6 devices and the address management in [section 6.2], with some modifications of the code. However, Netconf is not able to support the packet report from the device to the controller/applications, which may need extensions of the protocol.

6.2. Address Management

Having a centralized way to manage addresses requires an efficient protocol to request and allocate them. Among the possible solutions, Netconf or Radius could be extended.

7. Security Considerations

7.1. Source Address Validation and Traceback with Openv6

A easy-to-use solution for Source Address Validation would increase the safety of networks. If operators have an efficient and low cost unified solution for this problem for both IPv4 and IPv6 and the transition itself, they would be more incline to implement it and therefore the security of networks as a whole would improve.

8. IANA Considerations

This document has no actions for IANA.

9. Authors

Credits and Thanks

10. Acknowledgements

Reference previous work.

11. References

11.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

11.2. Informative References

[I-D.ietf-softwire-4rd]
Despres, R., Jiang, S., Penno, R., Lee, Y., Chen, G., and M. Chen, "IPv4 Residual Deployment via IPv6 - a Stateless Solution (4rd)", draft-ietf-softwire-4rd-07 (work in progress), October 2013.

[I-D.ietf-softwire-lw4over6]
Cui, Y., Qiong, Q., Boucadair, M., Tsou, T., Lee, Y., and I. Farrer, "Lightweight 4over6: An Extension to the DS-Lite Architecture", draft-ietf-softwire-lw4over6-06 (work in progress), February 2014.

[I-D.ietf-softwire-map-t]
Li, X., Bao, C., Dec, W., Troan, O., Matsushima, S., and T. Murakami, "Mapping of Address and Port using Translation (MAP-T)", draft-ietf-softwire-map-t-05 (work in progress), February 2014.

[I-D.ietf-softwire-map]
Troan, O., Dec, W., Li, X., Bao, C., Matsushima, S., Murakami, T., and T. Taylor, "Mapping of Address and Port with Encapsulation (MAP)", draft-ietf-softwire-map-10 (work in progress), January 2014.

[I-D.ietf-softwire-public-4over6]
Cui, Y., Wu, J., Wu, P., Vautrin, O., and Y. Lee, "Public IPv4 over IPv6 Access Network", draft-ietf-softwire-public-4over6-10 (work in progress), July 2013.

[I-D.ietf-v6ops-464xlat]
Mawatari, M., Kawashima, M., and C. Byrne, "464XLAT: Combination of Stateful and Stateless Translation", draft-ietf-v6ops-464xlat-10 (work in progress), February 2013.

[One-vision-for-IPv6]
Mark Townsley, "One vision for IPv6", .

[RFC5571] Storer, B., Pignataro, C., Dos Santos, M., Stevant, B., Toutain, L., and J. Tremblay, "Softwire Hub and Spoke Deployment Framework with Layer Two Tunneling Protocol Version 2 (L2TPv2)", RFC 5571, June 2009.

[RFC5969] Townsley, W. and O. Troan, "IPv6 Rapid Deployment on IPv4 Infrastructures (6rd) -- Protocol Specification", RFC 5969, August 2010.

[RFC6145] Li, X., Bao, C., and F. Baker, "IP/ICMP Translation Algorithm", RFC 6145, April 2011.

[RFC6219] Li, X., Bao, C., Chen, M., Zhang, H., and J. Wu, "The China Education and Research Network (CERNET) IVI Translation Design and Deployment for the IPv4/IPv6 Coexistence and Transition", RFC 6219, May 2011.

[RFC6333] Durand, A., Droms, R., Woodyatt, J., and Y. Lee, "Dual-Stack Lite Broadband Deployments Following IPv4 Exhaustion", RFC 6333, August 2011.

[RFC6535] Huang, B., Deng, H., and T. Savolainen, "Dual-Stack Hosts Using "Bump-in-the-Host" (BIH)", RFC 6535, February 2012.

[RFC6654] Tsou, T., Zhou, C., Taylor, T., and Q. Chen, "Gateway-Initiated IPv6 Rapid Deployment on IPv4 Infrastructures (GI 6rd)", RFC 6654, July 2012.

[RFC6674] Brockners, F., Gundavelli, S., Speicher, S., and D. Ward, "Gateway-Initiated Dual-Stack Lite Deployment", RFC 6674, July 2012.

[RFC6674] Brockners, F., Gundavelli, S., Speicher, S., and D. Ward, "Gateway-Initiated Dual-Stack Lite Deployment", RFC 6674, July 2012.

[RFC7039] Wu, J., Bi, J., Bagnulo, M., Baker, F., and C. Vogt, "Source Address Validation Improvement (SAVI) Framework", RFC 7039, October 2013.

[SD-NAT] Alain Durand, "SD-NAT", <http://www.ietf.org/proceedings/82/slides/behave-10.pdf>.

Authors' Addresses

Qiong Sun
China Telecom
No.118 Xizhimennei street, Xicheng District
Beijing 100035
P.R. China

Email: sunqiong@ctbri.com.cn

Will(Shucheng) Liu
Huawei Technologies
Bantian, Longgang District
Shenzhen 518129
P.R. China

Email: liushucheng@huawei.com

Cathy Zhou
Huawei Technologies
Bantian, Longgang District
Shenzhen 518129
P.R. China

Email: cathy.zhou@huawei.com

Guillaume Leclanche
Viagenie
246 Aberdeen
Quebec, QC G1R 2E1
Canada

Phone: +1 418 656 9254
Email: guillaume.leclanche@viagenie.ca

Operations Area Working Group
Internet-Draft
Intended status: Standards Track
Expires: August 14, 2014

S. Winter
RESTENA

February 10, 2014

A Configuration File Format for Extensible Authentication Protocol (EAP) Deployments
draft-winter-opsawg-eap-metadata-00

Abstract

This document specifies a file format for transfering configuration information of deployments of the Extensible Authentication Protocol (EAP). Such configuration files are meant to be discovered, consumed and used by EAP supplicant software to achieve secure and automatic EAP configuration on the consuming device.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on August 14, 2014.

Copyright Notice

Copyright (c) 2014 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust’s Legal Provisions Relating to IETF Documents (http://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of

the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction 2
1.1. Problem Statement 2
1.2. Other Approaches 4
1.3. Requirements Language 4
1.4. Terminology 4
2. XML Schema for EAP Metadata File Format 4
2.1. Location of XML Schema and Sample XML file 4
2.2. Description of Schema Elements 4
2.2.1. Overall structure 4
2.2.2. <AuthenticationMethods> 5
2.2.3. <ProviderInfo> 9
2.3. Internationalisation / Multi-language support 10
3. Issuer Authentication, Integrity Protection and Encryption of EAP Metadata configuration files 11
4. File Discovery 11
4.1. By MIME-Type: application/eap-config 11
4.2. By filename extension: .eap-config 11
4.3. By network location: SCAD 11
5. Existing Implementations 11
6. Design Decisions 12
6.1. Why XML and not $FOO? 12
6.2. Shallow vs. Deep definition of EAP method properties . . 12
6.3. EAP tunneling inside EAP tunnels 12
6.4. Placement of <OuterIdentity> inside <AuthenticationMethod> 12
7. Security Considerations 13
8. IANA Considerations 13
9. Contributors 14
10. References 14
10.1. Normative References 14
10.2. Informative References 14
Appendix A. Appendix A: MIME Type Registration Template 15

1. Introduction

1.1. Problem Statement

The IETF has produced the Extensible Authentication Protocol (EAP, [RFC3748] and numerous EAP methods (for example EAP-TTLS [RFC5281], EAP-TLS [RFC5216] and [RFC5931]); the methods have many properties which need to be setup on the EAP server and matched as configuration items on the EAP peer for a secure EAP deployment.

Setting up these configuration items is comparatively easy if the end-user devices which implement the EAP peer functionality are under central administrative control, e.g. in closed enterprise environments. Group policies or device provisioning by the IT department can push the settings to user devices.

In other environments, for example "BYOD" scenarios where users bring their own devices which are not under enterprise control, or in EAP-based WISP environments (see e.g. [HS20] and [I-D.wierenga-ietf-eduroam]) where it is not desired neither for the ISP nor for his user that the device control is in the ISPs hands, configuration of EAP is significantly harder as it has to be done by potentially very non-technical end users.

Correct configuration of all EAP deployment parameters is required to make the resulting authentications

- functional (i.e. the end user can authenticate to an EAP server at all)
- secure (i.e. the end user device can unambiguously authenticate the EAP server prior to releasing any sensitive client-side credentials)
- privacy-preserving (i.e. the end user is able to conceal his username from the EAP authenticator)

It would be desirable to be able to convey the EAP configuration information of a deployment in a machine parseable way to the end-user device, so that all the gory details need not be known/ understood by the user. Instead, the EAP peer software on the device could consume the configuration information and set up all EAP authentication details automatically.

However, there is currently no standard way of communicating configuration parameters about an EAP setup to the EAP peer.

This specification defines such a file format for EAP configuration metadata.

The specification allows for unique identification of an EAP identity provider by scoping it into a namespace and giving it a unique name inside that namespace. Using this unique identification, other configuration files (e.g. which detail an Enterprise Wi-Fi setup) can then refer to this particular instance of EAP identity information as authentication source.

1.2. Other Approaches

Device manufacturers sometimes have developed their own proprietary configuration formats, examples include Apple's "mobileconfig" (MIME type application/x-apple-aspen-config), Microsoft's XML schemata for EAP methods for use with the command-line "netsh" tool, or Intel's "PRO/Set Wireless" binary configuration files. The multitude of proprietary file formats and their different levels of richness in expression of EAP details create a very heterogenous and non-interoperable landscape.

New devices which would like to benefit from machine-parseable EAP configuration currently either have to choose to follow a competitor's approach and use that competitor's file format or have to develop their own. This situation is very unsatisfactory.

1.3. Requirements Language

In this document, several words are used to signify the requirements of the specification. The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119. [RFC2119]

1.4. Terminology

2. XML Schema for EAP Metadata File Format

2.1. Location of XML Schema and Sample XML file

The schema files are currently hosted on this preliminary location:

- Schema: http://ticker.eduroam.lu/cat/EAP-metadata/eap-metadata.xsd
- Sample: http://ticker.eduroam.lu/cat/EAP-metadata/eap-metadata.xml

2.2. Description of Schema Elements

2.2.1. Overall structure

The root element is the <EAPIdentityProviderList> tag, which contains a sequence of <EAPIdentityProvider> elements; these carry the actual installer information. In most practical applications, the <EAPIdentityProviderList> will contain only a single element; a longer list can be used for metadata transfers between systems or to allow users to select from a set of providers in one file.

Every <EAPIdentityProvider> has two attributes which make it globally unique: one attribute is the ’namespace’ attribute which defines the namespace inside which this EAPIdentityProvider is unique; the other attribute is the ’ID’ attribute which specifies the unique name inside the namespace. The element contains the following sub-elements:

- zero or one <ValidUntil> timestamp with an indication of possible expiry of the information in the configuration file. EAP peers importing the configuration file can use this information for example to re-assess whether the account is still valid (e.g. if the ValidUnil timestamp has passed, and authentication attempts consistently fail, the supplicant should consider the information stale and ask the user to verify his access authorisation with the EAP identity provider)

- exactly one <AuthenticationMethods> block contains a list of EAP methods which the EAPIdentityProvider supports. This element is described in more detail in section Section 2.2.2

- zero or one <ProviderInfo> blocks provide additional information about the EAPIdentityProvider, e.g. a logo to allow visual identification of the provider to the user in a user interface, or Acceptable Use Policies pertaining to the use of this EAP identity. This element is described in more detail in section Section 2.2.3

- zero or more <VendorSpecific> elements with undefined structure for cases where particular implementations of this specification need to convey additional data which is not covered by the other elements of this specification and does not require cross-vendor interoperability. The attribute "vendor" of the element MUST contain the vendor’s IANA Enterprise Number.

2.2.2. <AuthenticationMethods>

<AuthenticationMethods> is a sequence of <AuthenticationMethod> elements. Each such element specifies the properties of one supported authentication method with various elements. These elements are enumerated in section Section 2.2.2.1 The set of configuration parameters depends on the particular EAP method to be configured.

For instance, EAP-PWD [RFC5931] does not require any server certificate parameters; EAP-FAST and TEAP are the only ones making use of Protected Access Credential (PAC) provisioning. On the other hand, properties such as outer ("anonymous") identity or the need for a trusted root Certification Authority are common to several EAP

methods. The server- and client-side credential types of EAP methods are defined as a flat list of elements to choose from (see <ServerSideCredential> and <ClientSideCredential> below); see section Section 6.2 for a rationale.

Where the sequence of <AuthenticationMethod> elements contains more than one element, the order of appearance in the file indicates the server operator's preference for the supported EAP types; occurences earlier in the file indicate a more preferred authentication method.

When a consuming device receives multiple <AuthenticationMethod> elements, it should attempt to install more preferred methods first. If the configuration information for that method is insufficient (e.g. the <AuthenticationMethod> is EAP-TLS, but the configuration file does not contain the client certificate/private key and the device's credential store is not pre-loaded with the client's certificate), the device should query whether the more preferred method should be used (requiring the user to supplement the missing data) or whether a less-preferred method should be configured. In non-interactive provisioning scenarios, all methods should be tried in order until one method can be installed; if no method can be installed in a fully automated way, provisioning is aborted.

2.2.2.1. Authentication Method Properties

The <AuthenticationMethod> element contains

- exactly one <EAPMethod> element, which is an integer of the EAP method identifier as assigned by IANA
- zero or one <ServerSideCredential> elements which are a complex type containing elements which define means to authenticate the EAP server to the EAP peer (for a list of these elements, see section Section 2.2.2.2)
- zero or one <ClientSideCredential> elements which are a complex type containing elements which define means to authenticate the EAP peer to the EAP server (for a list of these elements, see section Section 2.2.2.3)
- zero or more <InnerAuthenticationMethod> elements. Elements of this type indicate that a tunneled EAP method is in use, and that further server-side and/or client-side credentials are defined inside the tunnel. The presence of more than one InnerAuthenticationMethod indicates that EAP Method Chaining is in use, i.e. that several inner EAP methods are to be executed in sequence inside the tunnel.

The <InnerAuthenticationMethod> element itself contains the same <EAPMethod>, <ServerSideCredentials> and <ClientSideCredentials> as described in the preceding list, but differs in two points:

- It can optionally contain the element <NonEAPAuthMethod> (an enumerated integer of authentication methods not based on EAP) instead of <EAPMethod> because some tunneled EAP types do not necessarily contain EAP inside the tunnel (e.g. TTLS-PAP, TEAP). Note that the XML Schema formally allows to specify both <EAPMethod> and <NonEAPAuthMethod>. This situation MUST NOT occur in configuration files to ensure deterministic interpretability.

- It can NOT contain further <InnerAuthenticationMethod> elements because establishing a secure tunnel inside an already established secure tunnel is considered a pathological case which needs not be considered. See section Section 6.3 for a rationale.

2.2.2.2. <ServerSideCredential> Properties

The server-side authentication of a mutually authenticating EAP method is typically based on X.509 certificates, which requires the EAP peer to be pre-provisioned with one or more trusted root Certification Authority prior to authenticating. A server is uniquely identified by presenting a certificate which is signed by these trusted CAs, and by the EAP peer verifying that the name of the server matches the expected one. Consequently, a (set of) CAs and a (set of) server names make up the ServerSideCredentials block.

Note that different EAP methods use different terminology when referring to trusted CA roots, server certificates, and server name identification. They also differ or have inherent ambiguity in their interpretation on where to extract the server name from (e.g. is the server name the CN part of the DistinguishedName, or is the server name one of the subjectAltName:DNS entries; what to do if there is a mismatch?). This specification introduces one single element for CA trust roots and naming; these notions map into the naming of the particular EAP methods very naturally. This specification can not remove the CN vs. sAN:DNS ambiguity in many EAP methods.

- zero or more <CA> elements: a Certification Authority which is trusted to sign the expected server certificate. The set of <CA> elements SHOULD contain self-signed root certificates to establish trust, and MAY contain additional intermediate CA certificates which ultimately root in these self-signed root CAs. A configuration file can, but SHOULD NOT include only an intermediate CA certificate (i.e. without also including the corresponding self-signed root) because trusting only an

intermediate CA without being able to verify to a self-signed root is an unsupported notion in many EAP peers.

- o zero or more <ServerID> elements: these elements contain the expected server names in incoming X.509 EAP server certificates. For EAP methods not using X.509 certificates for their mutual authentication, these elements contain other string-based handles which identify the server (Example: EAP-pwd).

2.2.2.3. <ClientSideCredential> Properties

There is a variety of means to identify the EAP peer to the EAP server. EAP methods use a subset of these criteria. As with server-side credentials, the terminology for the credential type may differ slightly between EAP types. The naming convention in this specification maps nicely into the method-specific terminology. Not all the criteria make sense in all contexts; for EAP methods which do not support a criterion, configuration files SHOULD NOT contain the corresponding elements, and consumers of the file MUST ignore these elements.

Specifying any one of these elements is optional and they can occur at most once. Consumers of configuration files MUST be able to fall back to user-interactive configuration for these parts if they are not specified (e.g. ask for the username and password for an EAP method during import of the EAP configuration data). Configuration files which do contain sensitive elements such as <Password> MUST be handled with due care after the import on the device (e.g. ensure minimal file permissions, or delete the source file after installing). The <ClientSideCredential> element has an attribute 'allow_save'; if it is set to false, sensitive parts of the client-side credentials MUST NOT be permanently saved on the device. See also section Section 3 for transport security considerations.

<OuterIdentity> is typcially used on the outside of a tunneled EAP method and allows to specify which user identity should be used outside the tunnel. This string is not used for actual user authentication, but may contain routing hints to send the request to the right EAP server.

<UserName> contains the actual username to be used for user authentication. For tunneled EAP methods, this element SHOULD only occur in the <InnerAuthenticationMethod>'s <ClientSideCredentials> - if differing outer identities are not desired in the deployment, the <OuterIdentity> element should be populated for the <AuthenticationMethod> element; but may contain the actual username then.

<ClientCertificate> contains a X.509 certificate and private key; if the key is protected, the <Passphrase> element MAY be used to indicate the passphrase, see below

<Passphrase> contains the passphrase needed to unlock a cryptographic credential internally on the device (i.e. it is not used itself for the actual authentication during the EAP conversation)

<Password> contains the user's password, or an otherwise secret string which the user needs to authenticate to the EAP server

<PAC> contains the Protected Access Credential, typically used in EAP-FAST and TEAP.

<ProvisionPAC> is a boolean which indicates whether a PAC should be provisioned on the first connection. Note that the specification allows to use <ProvisionPAC> without a CA nor ServerID in <ServerSideCredential>. While this allows the operation mode of "Anonymous PAC Provisioning" as used in EAP-FAST, due to the known security vulnerabilities of anonymous PAC provisioning, this combination SHOULD NOT be used.

2.2.3. <ProviderInfo>

This specification needs to consider that user interaction during the installation time may be required; the user at the very least must be empowered to decide whether the configuration file was issued by a provider he has an account with; the provider may have hints for the user (e.g. which password to use for the login), or may want to display links to helpdesk pages in case the user has problems with the setup or use of his identity.

The <ProviderInfo> element allows to specify a range of potentially useful information for display to the user (some of which is relevant only during installation time, other pieces of information could be retained by the EAP peer implementation and displayed e.g. in case of failed authentication):

- <DisplayName> specifies a user-friendly name for the EAP Identity Provider. Consumers of this specification should be aware that this is simple text, and self-asserted by the producer of the configuration file. If more authoritative information about the issuer is available (e.g. if the file is signed with S/MIME and carries an Organisation name (O attribute) in the signing certificate) then the more authoritative information should be displayed with more prominence than the self-asserted one.

- <Description> specifies a generic descriptive text which should be displayed to the user prior to the installation of the configuration data.

- <ProviderLocation> specifies the approximate geograhic location(s) of the EAP Identity Provider and/or his Points of Presence. This can be useful if the configuration file contains multiple <EAPIdentityProvider> elements; the user device can then make an informed guess which of the Identity Providers could be a good match to suggest to the user

- <ProviderLogo> specifies the logo of the EAP Identity Provider. The same self-assertion considerations as for <DisplayName> above apply.

- <TermsOfUse> contains terms of use to be displayed to and acknowledged by the user prior to the installation of the configuration on the user's system

- <Helpdesk> is a complex element with three possible sub-elements: <EmailAddress>, <WebAddress> and <Phone>, all of which can be displayed to the user.

2.3. Internationalisation / Multi-language support

Some elements in this specification contain text to be displayed in User Interfaces; depending on the user's language preferences, it would be desirable to present the information in a local language. Other elements contain contact information, and those contact points may only be able to handle requests in a number of languages; it may be desirable to present only contact points to the user which are compatible with his language capabilities.

All elements which either contain localisable text, or which point to external resources in localised languages, have an optional "lang" attribute. The elements can occur more than once in the specification, which enables an iteration of the element in all applicable languages. If the "lang" attribute is omitted or "lang" is set to "C", the instance of the element is considered a default choice which is to be displayed if no other instance is a better match.

If the entire file content consistently uses only one language set, e.g. all the elements are to be treated as "default" choices, the language can also be set for the entire <EAPIdentityProvider> element in its own "lang" attribute.

3. Issuer Authentication, Integrity Protection and Encryption of EAP Metadata configuration files

S/MIME or underlying transport security. Nuff said :-)

4. File Discovery

4.1. By MIME-Type: application/eap-config

For transports where the categorisation of file types via MIME types is possible (e.g. HTTP, E-Mail), this document assigns the MIME type

application/eap-config

Edge devices can associate this MIME type to incoming files on such transports, and register the application which can consume the EAP Metadata as the default handler for this file type. By doing so, for example a single click or tap on a link to the file in the device's browser will invoke the configuration process.

This method of discovery is analogous to the Apple "mobileconfig" discovery on recent versions of Mac OS and iOS.

4.2. By filename extension: .eap-config

In situations where file types can not be determined by MIME type meta-information (e.g. when the file gets stored on a local filesystem), this document RECOMMENDs that EAP Metadata configuration files be stored with the extension

.eap-config

to identify the file as containing EAP Metadata configuration information. Edge devices can register the application which can consume the EAP Metadata with this file extension. By doing so, for example a single click or tap on the filename in the device's User Interface will invoke the configuration process.

4.3. By network location: SCAD

5. Existing Implementations

Producers of the configuration files

- eduroam Configuration Assistant Tool: this existing tool already produces EAP configuration files in various proprietary formats for hundreds of EAP Identity Providers. The authors of this

specification will add a module which will produce configuration files in the file format as specified in this document.

Consumers of the configuration files

- Android: the authors of this specification are currently developing an App for the Android operating system (compatible with API level 18 of Android, i.e. version 4.3 and above) which can consume the file format as defined in this draft specification and configure EAP via the WifiEnterpriseConfig API.

- Linux: the authors of this specification are currently developing an application for UNIX-like operating systems which configure enterprise networks via the NetworkManager daemon; the application can consume the file format as defined in this draft specification and configure the settings via Networkmanager's D-BUS interface.

6. Design Decisions

6.1. Why XML and not $FOO?

XML is a popular choice for EAP configurations: Microsoft's "netsh" files, Apple's "mobileconfig" files, the Wi-Fi Alliance's "PerProviderSubscription Managed Object", and other vendor/SDO definitions are all using XML.

Other possibilities which will be duly considered if sufficient interest warrants it include, but are not limited to:

- JSON (less rich expressions; no verification of conformity such as with XML Schema - but it doesn't need many resources to parse and may thus be advantageous for constrained devices)

- YANG (very rich feature set, and tools can produce automatic conversions to both XML and JSON - but not as well understood by the author, and unlikely to be natively supported on consumer devices)

6.2. Shallow vs. Deep definition of EAP method properties

6.3. EAP tunneling inside EAP tunnels

6.4. Placement of <OuterIdentity> inside <AuthenticationMethod>

7. Security Considerations

8. IANA Considerations

IANA is requested to allocate the MIME type "application/eap-config" in the MIME Media Types / application registry (see section Section 4.1). The allocation should contain the following values:

- Name: eap-config
- Template: see Appendix A (RFC editor note: remove this appendix prior to publication; replace this line with the URL to the application as posted online)
- Reference: RFCabcd (RFC editor note: replace with the RFC number of this document)

IANA is requested to allocate the location "TBD" in the "well-known URIs" registry. The allocation should contain the following values:

- URI Suffix: TBD
- Change Controller: IETF
- Reference: RFCabcd (RFC editor note: replace with the RFC number of this document)
- Related Information: none

IANA is requested to register the XML namespace "urn:ietf:params:xml:ns:eap-config" in the "IETF XML Registry / ns". The allocation should contain the following values:

- ID: eap-config
- URI: urn:ietf:params:xml:ns:eap-config
- Filename: https://www.iana.org/assignments/xml-registry/ns/eap-config.txt (to be created by IANA)
- Reference: RFCabcd (RFC editor note: replace with the RFC number of this document)

IANA is requested to register the XML schema "urn:ietf:params:xml:schema:eap-config" in the "IETF XML Registry / schema". The allocation should contain the following values:

- ID: eap-config

- URI: urn:ietf:params:xml:schema:eap-config
- Filename: https://www.iana.org/assignments/xml-registry/schema/eap-config.xsd (to be created by IANA; current XSD file is linked to in section Section 2.1)
- Reference: RFCabcd (RFC editor note: replace with the RFC number of this document)

9. Contributors

Tomasz Wolniewicz of Nicolaus Copernicus University in Torun, Poland, provided significant input into this specification.

10. References

10.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

10.2. Informative References

[RFC3748] Aboba, B., Blunk, L., Vollbrecht, J., Carlson, J., and H. Levkowetz, "Extensible Authentication Protocol (EAP)", RFC 3748, June 2004.

[RFC5216] Simon, D., Aboba, B., and R. Hurst, "The EAP-TLS Authentication Protocol", RFC 5216, March 2008.

[RFC5281] Funk, P. and S. Blake-Wilson, "Extensible Authentication Protocol Tunneled Transport Layer Security Authenticated Protocol Version 0 (EAP-TTLSv0)", RFC 5281, August 2008.

[RFC5931] Harkins, D. and G. Zorn, "Extensible Authentication Protocol (EAP) Authentication Using Only a Password", RFC 5931, August 2010.

[I-D.wierenga-ietf-eduroam] Wierenga, K., Winter, S., and T. Wolniewicz, "The eduroam architecture for network roaming", draft-wierenga-ietf-eduroam-02 (work in progress), January 2014.

[HS20] Wi-Fi Alliance, "Hotspot 2.0 Technical Specification", 2012, <https://www.wi-fi.org/hotspot-20-technical-specification-v100>.

Appendix A. Appendix A: MIME Type Registration Template

The following values will be used for the online MIME type registration at https://www.iana.org/form/media-types

Your Name: Stefan Winter

Your Email Address: stefan.winter@restena.lu

Media Type Name: Application

Subtype name: (Standards tree) eap-config

Required parameters: (none)

Optional parameters: (none)

Encoding Considerations: 8-Bit text

Security Considerations: This file type carries configuration information for consumer devices. It has the potential to substantially alter the consumer's device; particularly to install a new trusted Certification Authority. Applications consuming files of this type need to be cautious to explain to the end user what is being altered, so that they understand the consequences. For further explanations, see Section 7 of draft-winter-opsawg-eap-metadata. (Note to IANA: replace this reference with the RFC number of this document once known)

Interoperability Considerations: The file content is XML version 1.0 or later. The encoding SHOULD be UTF-8, but implementations consuming the file SHOULD be prepared to encounter different encodings.

Published Specification: draft-winter-opsawg-eap-metadata (Note to IANA: replace this reference with the RFC number of this document once known)

Applications which use this media type: files of this type are intended for consumption by sortware on edge devices; they consume the information therein to configure authentication parameters (EAP protocol and EAP method payload configurations) which are then applied to network or application authentication scenarios.

Fragment Identifier Considerations: files of this type are expected to be transmitted in their entirety. If a reference to a specific part of the content is to be made, XML XPath expressions

are to be used. I.e. fragment identifier formats are not expected to be used.

Restrictions on Usage: none

Provisional registration: initial submission of this form will be executed after adoption in the IETF; it will be a provisional registration. Final registration will be done after IESG review.

Additional information:

Deprecated alias types for this name: none

Magic numbers: none

File extensions: eap-config

Macintosh File Type Codes: TBD

Object Identifiers or OIDs: none

Intended Usage: Common (no further provisions)

Other Information/General Comment: none

Person to contact for further information:

Name: Stefan Winter

E-Mail: stefan.winter@restena.lu

Author/Change controller: IETF

DATA

Author’s Address

Stefan Winter
Fondation RESTENA
6, rue Richard Coudenhove-Kalergi
Luxembourg 1359
LUXEMBOURG

Phone: +352 424409 1
Fax: +352 422473
EMail: stefan.winter@restena.lu
URI: http://www.restena.lu.

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: April 25, 2014

R. Zhang
China Telecom
Z. Cao
H. Deng
China Mobile
R. Pazhyannur
S. Gundavelli
Cisco

October 22, 2013

Alternate Tunnel Encapsulation for Data Frames in CAPWAP
draft-zhang-opsawg-capwap-cds-01

Abstract

CAPWAP ([RFC5416]) defines two tunneling modes for encapsulating data frames from stations associated with WLAN: 802.3 Tunnel and 802.11 Tunnel modes. This document provides for an alternate tunnel encapsulation. The alternate tunnel encapsulation allows 1) the WTP to tunnel non-management data frames to an endpoint different from the AC and 2) allows the WTP to tunnel using one of many known encapsulation types such as IP-IP, IP-GRE, CAPWAP. The WTP may advertise support for Alternate Tunnel encapsulation during the discovery process and AC may select one of the supported Alternate Tunnel encapsulate types during the WTP configuration. Further, the AC may configure WTP to enable the alternate tunnel encapsulation.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on April 25, 2014.

Copyright Notice

Copyright (c) 2013 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (http://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction 2
1.1. Conventions used in this document 4
1.2. Terminology 4
2. Alternate Tunnel Encapsulation 5
2.1. Supported Alternate Tunnel Encpsulation 5
2.2. Alternate Tunnel Encapsulation 5
3. IANA Considerations 6
4. Security Considerations 6
5. Contributors 6
6. References 6
6.1. Normative References 6
6.2. Informative References 6
Authors' Addresses 7

1. Introduction

CAPWAP ([RFC5415], [RFC5416]) defines a tunnel mode that specifies the frame tunneling type to be used for 802.11 data frames from all stations associated with the WLAN. The following types are supported:

- Local Bridging: All user traffic is to be locally bridged.
- 802.3 Tunnel: All user traffic is to be tunneled to the AC in 802.3 format.
- 802.11 Tunnel: All user traffic is to be tunneled to the AC in 802.11 format.

There are two shortcomings with currently specified tunneled modes: 1) it does not allow the WTP to tunnel data frames to an endpoint different from the AC and 2) it does not allow the WTP to tunnel data frames using any encapsulation other than CAPWAP (as specified in Section 4.4.2 of [RFC5415]). Next, we describe what is driving the above mentioned two requirements.

Some operators deploying large number of Access Points prefer to centralize the management and control of Access Points (AP) while distributing the handling of data traffic to increase scaling. This motivates an architecture as shown in Figure 1 that has the Access Controller in a centralized location and one or more tunnel gateways (or Access Routers) that terminate the data tunnels from the various WTPs. central data center. This split architecture has two benefits over an architecture where data traffic is aggregated at the AC: 1) reduces the scale requriement on data traffic handling capability of the AR and 2) leads to more efficient/optimal routing of data traffic.

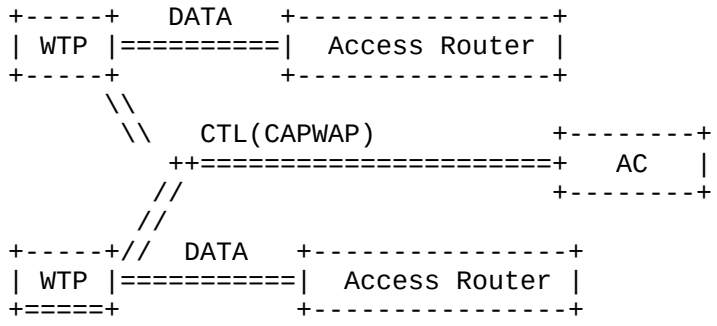
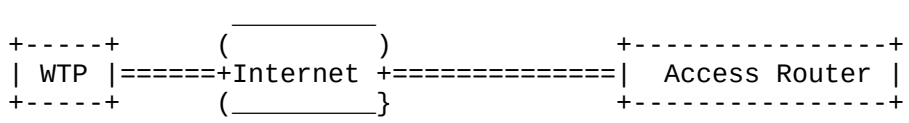


Figure 1: Centralized Control with Distributed Data

The above system (shown in Figure 1) could be achieved by setting the tunnel mode to Local bridging. In such a case the AC would handle control of WTPs as well as handle the management traffic to/from the stations. The data frames (non-management) from the stations would be handled by the local Access Router. However, in many deployments the operator managing the WTPs/AC may be different from the operator providing the internet connectivity to the WTPs. Further, the WTP operator may want (or be required by legal/regulatory requirements) to tunnel the traffic back to an Access Router in its network as shown in Figure 2. The tunneling requirement may be driven by the need to apply policy at the Access Router or a legal requirement to support lawful intercept of user traffic. This motivates the need for the WTP to support an alternate Tunnel encapsulation support where the data tunnels from the WTP are terminated at an AR (and more specifically at an end point different from the AC).



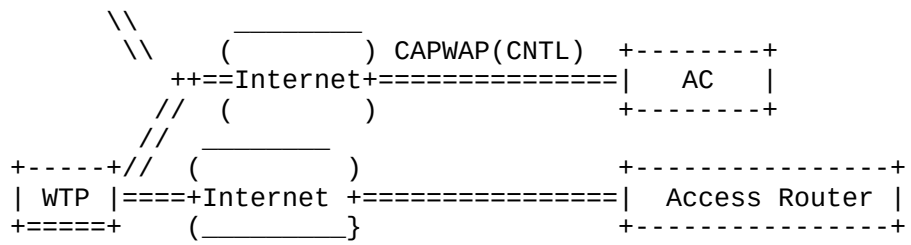


Figure 2: Centralized Control with Distributed Data

In the case where the WTP is tunneling data frames to an AR (and not the AC), the choice of tunnel encapsulation need not be restricted only to CAPWAP (as described in Section 4.4.2 of [RFC5415]). In fact, the WTP may additionally support other widely used encapsulation types such as L2TP, L2TPv3, IP-in-IP, IP/GRE, etc. The WTP may advertise the different alterante tunnel encapsulation types supported and the AC can select one of the supported encapsulation types.

1.1. Conventions used in this document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL","SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119]

1.2. Terminology

Access Controller (AC): The network entity that provides WTP access to the network infrastructure in the data plane, control plane, management plane, or a combination therein.

Access Point (AP): the same with Wireless Termination Point (WTP), The physical or network entity that contains an RF antenna and wireless Physical Layer (PHY) to transmit and receive station traffic for wireless access networks.

CAPWAP Control Plane: A bi-directional flow over which CAPWAP Control packets are sent and received.

CAPWAP Data Plane: A bi-directional flow over which CAPWAP Data frames are sent and received.

EAP: Extensible Authentication Protocol, the EAP framework is specified in [RFC3748].

2. Alternate Tunnel Encapsulation

2.1. Supported Alternate Tunnel Encpsulation

The IEEE 802.11 Supported Alternate Tunnel Encapsulations message element allow the WTP to communicate the supported tunnels. The Discovery Request message, Primary Discovery Request message, and Join Request message may include one such message element

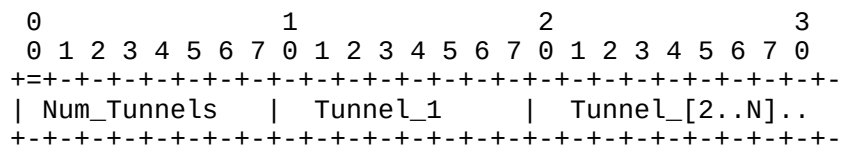


Figure 3: IEEE 802.11 Supported Tunnels Encapsulations

- Type: TBD for IEEE 802.11 Supported MAC Profiles
- Num_Tunnels >=1: This refers to number of profiles presnt in this messaage element. There must be at least one profile.
- Tunnel: Each Tunnel is idnentified by a value given in Section 2.2

2.2. Alternate Tunnel Encapsulation

The IEEE 802.11 Alternate Tunnel Encapsulation message element allows the AC to select the profile. This messsage element may be provided along with the IEEE 802.11 ADD WLAN message element while configuring a WLAN on the WTP.

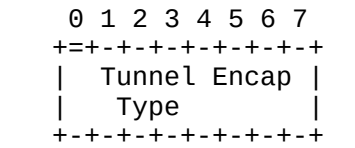


Figure 4: IEEE 802.11 MAC Profile

- Type: TBD for IEEE 802.11 MAC Profile
- Tunnel Encap Type: The profile is idnentified by a value as given below
  - 0: CAPWAP data channel as described in [RFC5415][RFC5416]
  - 1: L2TP
  - 2: L2TPv3
  - 3: IP-in-IP
  - 4: IP/GRE

3. IANA Considerations

To be specified.

4. Security Considerations

Security considerations for the CAPWAP protocol has been analyzed in Section 12 of [RFC5415]. This document does not introduce other security issues besides what has been analyzed in RFC5415.

5. Contributors

This document stems from the joint work of Hong Liu, Yifan Chen, Chunju Shao from China Mobile Research. Thank all the contributors of this document.

6. References

6.1. Normative References

[RFC5415] Calhoun, P., Montemurro, M., and D. Stanley, "Control And Provisioning of Wireless Access Points (CAPWAP) Protocol Specification", RFC 5415, March 2009.

6.2. Informative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC3118] Droms, R. and W. Arbaugh, "Authentication for DHCP Messages", RFC 3118, June 2001.

[RFC3748] Aboba, B., Blunk, L., Vollbrecht, J., Carlson, J., and H. Levkowetz, "Extensible Authentication Protocol (EAP)", RFC 3748, June 2004.

[RFC4564] Govindan, S., Cheng, H., Yao, ZH., Zhou, WH., and L. Yang, "Objectives for Control and Provisioning of Wireless Access Points (CAPWAP)", RFC 4564, July 2006.

[RFC5416] Calhoun, P., Montemurro, M., and D. Stanley, "Control and Provisioning of Wireless Access Points (CAPWAP) Protocol Binding for IEEE 802.11", RFC 5416, March 2009.

[RFC5417] Calhoun, P., "Control And Provisioning of Wireless Access Points (CAPWAP) Access Controller DHCP Option", RFC 5417, March 2009.

Authors' Addresses

Rong Zhang
China Telecom
No.109 Zhongshandadao avenue
Guangzhou 510630
China

Email: zhangr@gsta.com

Zhen Cao
China Mobile
Xuanwumenxi Ave. No. 32
Beijing 100871
China

Phone: +86-10-52686688
Email: zehn.cao@gmail.com, caozhen@chinamobile.com

Hui Deng
China Mobile
No.32 Xuanwumen West Street
Beijing 100053
China

Email: denghui@chinamobile.com

Rajesh S. Pazhyannur
Cisco
170 West Tasman Drive
San Jose, CA 95134
USA

Email: rpazhyan@cisco.com

Sri Gundavelli
Cisco
170 West Tasman Drive
San Jose, CA 95134
USA

Email: sgundave@cisco.com