INTERNATIONAL TELECOMMUNICATION UNION

ITU-T

TELECOMMUNICATION
STANDARDIZATION SECTOR
OF ITU

G.9961

Amendment 4
(06/2018)

SERIES G: TRANSMISSION SYSTEMS AND MEDIA, DIGITAL SYSTEMS AND NETWORKS

Access networks – In premises networks

Unified high-speed wire-line based home networking transceivers - Data link layer specification

Amendment 4

CAUTION !

PREPUBLISHED RECOMMENDATION

This prepublication is an unedited version of a recently approved Recommendation. It will be replaced by the published version after editing. Therefore, there will be differences between this prepublication and the published version.

FOREWORD

The International Telecommunication Union (ITU) is the United Nations specialized agency in the field of telecommunications, information and communication technologies (ICTs). The ITU Telecommunication Standardization Sector (ITU-T) is a permanent organ of ITU. ITU-T is responsible for studying technical, operating and tariff questions and issuing Recommendations on them with a view to standardizing telecommunications on a worldwide basis.

The World Telecommunication Standardization Assembly (WTSA), which meets every four years, establishes the topics for study by the ITU-T study groups which, in turn, produce Recommendations on these topics.

The approval of ITU-T Recommendations is covered by the procedure laid down in WTSA Resolution 1.

In some areas of information technology which fall within ITU-T's purview, the necessary standards are prepared on a collaborative basis with ISO and IEC.

NOTE

In this Recommendation, the expression "Administration" is used for conciseness to indicate both a telecommunication administration and a recognized operating agency.

Compliance with this Recommendation is voluntary. However, the Recommendation may contain certain mandatory provisions (to ensure, e.g., interoperability or applicability) and compliance with the Recommendation is achieved when all of these mandatory provisions are met. The words "shall" or some other obligatory language such as "must" and the negative equivalents are used to express requirements. The use of such words does not suggest that compliance with the Recommendation is required of any party.

INTELLECTUAL PROPERTY RIGHTS

ITU draws attention to the possibility that the practice or implementation of this Recommendation may involve the use of a claimed Intellectual Property Right. ITU takes no position concerning the evidence, validity or applicability of claimed Intellectual Property Rights, whether asserted by ITU members or others outside of the Recommendation development process.

As of the date of approval of this Recommendation, ITU [had/had not] received notice of intellectual property, protected by patents, which may be required to implement this Recommendation. However, implementers are cautioned that this may not represent the latest information and are therefore strongly urged to consult the TSB patent database at http://www.itu.int/ITU-T/ipr/.

Amendment 4 to Recommendation ITU-T G.9961 (2015)

Unified high-speed wire-line based home networking transceivers - Data link layer specification: Amendment 4

Summary

Amendment 4 to Recommendation ITU-T G.9961 (2015) includes a new management message compression mechanism to minimize the length of these messages to reduce overhead. It also includes some additional capabilities for low power modes.

Amendment 4 to Recommendation ITU-T G.9961 (2015)

Unified high-speed wire-line based home networking transceivers - Data link layer specification: Amendment 4

1. **Revise the text of clause 8.3.6.1.1 "Long inactivity scheduling" as follows:**

8.3.6.1.1 Long inactivity scheduling

A node may request the domain master for inactivity scheduling for multiple MAC cycles by sending an IAS_LongInactivity.req message. The node may request two types of long inactivity scheduling: If the node wants this schedule to be effective only once, it indicates the requested duration of the inactive period. If the node wants this schedule to be effective more than once, it indicates the requested duration of the inactive period and the requested duration of the active period that follows the inactive period. The sum of both periods is called a sleep cycle. In this case the specified inactive period followed by the active period will repeat until it is cancelled or changed by the domain master.

A default sleep cycle and an indication of the beginning of the next default sleep cycle shall be published by the DM through the power saving update auxiliary information field of the MAP. Nodes sending a Long Inactivity request should align their requests to this default sleep cycle.

The domain master, if the request is accepted, shall announce the inactivity schedule as proposed by the receiver. The start time and duration of the inactive period and the duration of the following active period (if applicable) for a long inactivity schedule shall be transmitted in the auxiliary information field of the MAP message. The domain master may use the validity counter-based update (AUX_VALID = 3-7 and ModificationFlag = 1) or the immediate update (ModificationFlag = 0) for long inactivity scheduling announcement (see clause 8.8.5).

The domain master may directly force any node in the domain to follow a long inactivity scheduling, defined in the auxiliary information field of the MAP message (see clause 8.8.5.3). The scheduling shall coincide with the low power mode capability of the EP, exchanged in the registration request message (ADM_NodeRegistrRequest.req). The EP, receiving the long inactivity schedule, shall confirm the successful message reception by sending the IAS_LongInactivity.ind message to the domain master. The EP shall conduct the inactivity scheduling in the next MAC cycle after sending out the IAS_LongInactivity.ind message.

All nodes shall track the inactivity scheduling using the domain master transmit clock which is distributed via the MAP message.

A node that is scheduled to enter inactive state shall be able to receive frames transmitted 100 μs before the beginning of the inactive state and shall finish the current frame sequence exchange before entering inactive state.

The node that requested an inactivity schedule may transition into the inactive state as instructed in the MAP. If the MAP does not include the inactivity schedule within 100 ms after the request was sent, the node may repeat the request 200 ms after it transmitted the last request.

During the inactive period, a node is not required to decode the MAP. After the inactive period ends, the node shall transition back into the active state. The duration of any inactive period shall be larger than or equal to a MAC cycle and shall not exceed the re-registration period except for the case of standby mode (L3), where the inactive period is unlimited. After the current schedule expires, a node may request another inactivity schedule.

The domain master, if the request is rejected, shall indicate why the request for inactivity is denied with a reason by sending the IAS_LongInactivity.cnf message. The node that received an inactivity denial shall act based on the ReasonCode.

A node can request to change the current inactivity schedule by sending another inactivity schedule, or cancel the current inactivity schedule by sending IAS_LongInactivity.req with LIS_TYPE = 2 while it is in active period. The domain master may terminate or change the current inactivity scheduling any time by sending different inactivity schedules.

The format of the MMPL of the IAS_LongInactivity.req, ~~and~~ IAS_LongInactivity.cnf and IAS_LongInactivity.ind messages shall be as shown in Table 8-10, ~~and~~ Table 8-11 and Table 8-11.1, respectively.

Table 8-10 – Format of the MMPL of the IAS_LongInactivity.req message

Field	Octet	Bits	Description
LIS_TYPE	0	[2:0]	Proposed type of long inactivity scheduling 0: inactivity schedule is valid only once. In this case LIS_ACT_DUR shall be set to zero. 1: inactivity schedule repeats itself. In this case the inactivity schedule is valid until it is cancelled or changed. 2: inactivity schedule is cancelled by the node. In this case LIST_INACT_DUR and LIST_ACT_DUR shall be set to zero. Other values are reserved by ITU-T.
Reserved		[7:3]	Reserved by ITU-T (Note)
LIS_INACT_DUR	1 and 2	[15:0]	Requested duration of the inactive period, expressed in units of $1/16^{th}$ of the MAC CYCLE duration, represented as a 16-bit unsigned integer. This value shall be larger than or equal to the length of one MAC cycle or set to zero. The reserved value $FFFF_{16}$ is used to indicate an unlimited inactive period. (L3 mode).
LIS_ACT_DUR	3 and 4	[15:0]	Requested duration of the active period that immediately follows the inactivity period specified by LIS_INACT_DUR, expressed in units of $1/16^{th}$ of the MAC CYCLE duration, represented as a 16-bit unsigned integer. This value shall be larger than or equal to the length of one MAC cycle or set to zero.
NOTE – Bits that are reserved by ITU-T shall be set to zero by the transmitter and ignored by the receiver.			

Table 8-11 – Format of the MMPL of the IAS_LongInactivity.cnf message

Field	Octet	Bits	Description
ReasonCode	0	[3:0]	Reason why inactivity is denied (Note 1) 0000 = no reason specified 0001 = proposed inactivity period is too long 0010 = proposed inactivity period is too short 0011 = proposed inactivity rejected due to unsolvable scheduling issues 0100 = proposed inactivity rejected due to scheduling issues being solved. Retry is allowed 0101 = proposed inactivity rejected due to unsolvable topology issues 0110 = proposed inactivity rejected due to topology issues being solved. Retry is allowed.
Reserved		[7:4]	Reserved by ITU-T (Note 2)
NOTE 1 – Definition of other ReasonCodes is for further study. NOTE 2 – Bits that are reserved by ITU-T shall be set to zero by the transmitter and ignored by the receiver.			

Table 8-11.1 – Format of the MMPL of the IAS_LongInactivity.ind message

Field	Octet	Bits	Description
Message confirmation	0	[0]	This bit shall be set to one.
Reserved		[7:1]	Reserved by ITU-T (Note 1)
NOTE 1– Bits that are reserved by ITU-T shall be set to zero by the transmitter and ignored by the receiver.			

A node that needs to transmit to another node of the domain that is in standby mode L3 (Long Inactivity Scheduling) may request the DM to instruct the node in L3 to leave L3 state. For this, the requesting node shall send the message LP_RequestWakeUp.req to the DM (see clause 8.21.4).

The format of the MMPL of the LP_RequestWakeUp.req message shall be as shown in Table 8-11.2~~1~~.

Table 8-11.2~~1~~ – Format of the MMPL of the LP_RequestWakeUp.req message

Field	Octet	Bits	Description
NumNodesWakeUp	0	[7:0]	Indicates the number of nodes (k) that are being requested to leave L3 power saving mode
WakeUpRequest[0]	1	[7:0]	DeviceID of the first node that is requested to leave L3 power saving mode.
…	…	…	…
WakeUpRequest[k-1]	k	[7:0]	DeviceID of the last node that is requested to leave L3 power saving mode.

2. Revise the text of clause 8.3.6.2.1 "Short inactivity scheduling" as follows:

8.3.6.2.1 Short inactivity scheduling

A node may request the domain master for an inactivity scheduling for a fraction of a MAC cycle by sending an IAS_ShortInactivity.req message. This message defines the inactive periods within a MAC cycle. The node may request two types of short inactivity scheduling: valid once or valid until cancelled or changed.

The domain master, if the request is accepted, shall announce the inactivity scheduling as proposed by the receiver. The inactive and active portions of the MAC cycle for a short inactivity schedule shall be transmitted in the auxiliary information field of the MAP message (see clause 8.8.5.4). The domain master may use the validity counter-based update (AUX_VALID = 3-7 and ModificationFlag = 1) or the immediate update (ModificationFlag = 0) for short inactivity scheduling announcement (see clause 8.8.5).

The domain master may directly force any node in the domain to follow a short inactivity scheduling, defined in the auxiliary information field of the MAP message (see clause 8.8.5.4). The scheduling shall coincide with the low power mode capability of the EP, exchanged in the registration request message (ADM_NodeRegistrRequest.req). The EP, receiving the short inactivity schedule, shall confirm the successful message reception by sending the IAS_ShortInactivity.ind message to the domain master. The EP shall conduct the inactivity scheduling in the next MAC cycle after sending out the IAS_ShortInactivity.ind message.

The node that requested inactivity may transition into the inactive state as instructed in the MAP. If the MAP does not include the inactivity schedule within 100 ms after the request was sent, the node may repeat the request 200 ms after it transmitted the last request.

The domain master, if the request is rejected, shall indicate why the request for inactivity is denied with a reason by sending the IAS_ShortInactivity.cnf message. The node that received an inactivity denial shall act based on the ReasonCode.

Nodes in short inactivity scheduling shall be able to decode the MAP at every MAC cycle. The DM shall schedule MAP transmission(s) and inactivity schedules in a way that ensures that each node can receive MAP(s) during its active portion of the MAC cycle.

A node can change the current inactivity schedule by sending another IAS_ShortInactivity.req while it is in active state. The domain master can terminate or change the current inactivity scheduling any time by sending different inactivity schedules.

The format of the MMPL of the IAS_ShortInactivity.req, ~~and~~ IAS_ShortInactivity.cnf and IAS_ShortInactivity.ind messages shall be as shown in Table 8-12, ~~and~~ Table 8-13 and Table 8-13.1, respectively.

Table 8-12 – Format of the MMPL of the IAS_ShortInactivity.req message

Field	Octet	Bits	Description
SIS_TYPE	0	[2:0]	Proposed type of short inactivity scheduling 0: inactivity schedule is valid only once. 1: inactivity schedule repeats itself. In this case the inactivity schedule is valid until it is cancelled or changed. 2: inactivity schedule is cancelled by the node. In this case SIS_IND shall be set to zero. Other values are reserved by ITU-T.
Reserved		[7:3]	Reserved by ITU-T (Note)
SIS_IND	1 and 2	[15:0]	Requested indication of one or more inactive periods within a MAC cycle represented, as a 16-bit unsigned integer. 16-bit map is used to represent inactive periods. The bit0 (LSB) and bit15 (MSB) correspond to the first and last 1/16-th portions of a MAC cycle, respectively. A bit corresponding to each portion shall be set to one if the node is active during that time, and set to zero otherwise.
NOTE – Bits that are reserved by ITU-T shall be set to zero by the transmitter and ignored by the receiver.			

Table 8-13 – Format of the MMPL of the IAS_ShortInactivity.cnf message

Field	Octet	Bits	Description
ReasonCode	0	[3:0]	Reason why inactivity is denied (Note 1) 0000 = no reason specified 0001 = proposed inactivity period is too long 0010 = proposed inactivity period is too short 0011 = proposed inactivity rejected due to unsolvable scheduling issues 0100 = proposed inactivity rejected due to scheduling issues being solved. Retry is allowed.
Reserved		[7:4]	Reserved by ITU-T (Note 2)
NOTE 1 – Definition of other ReasonCodes is for further study. NOTE 2 – Bits that are reserved by ITU-T shall be set to zero by the transmitter and ignored by the receiver.			

Table 8-13.1 – Format of the MMPL of the IAS_ShortInactivity.ind message

Field	Octet	Bits	Description
Message confirmation	0	[0]	This bit shall be set to one.
Reserved		[7:1]	Reserved by ITU-T (Note 1)
NOTE 1– Bits that are reserved by ITU-T shall be set to zero by the transmitter and ignored by the receiver.			

3. Revise the text of clause 8.3.8 "Extended acknowledgements" as follows:

8.3.8 Extended acknowledgements

…

A source node can offer a destination node this choice by sending a BMSG frame with the BTXEF set to one, EXTACKGR bit set to one and BTXGL set to a non-zero value in any of the BMSG frames. In this case, as BTXGL ≠ 0, the destination node may send a BACK frame, a regular ACK frame or an extended ACK frame that fits within the granted time duration by the source node. The BTXGL shall at least include time for the destination node to send an extended ACK. Specifically, the destination node's response to the BMSG shall be as shown in Table 8-13.2~~1~~:

Table 8-13.2~~1~~ – Extended acknowledgement settings

BTXEF	BTXGL	Frame transmitted by the destination node
1	2 symbols for destination node	ACK or EACK
1	>2 symbols for destination node	ACK or EACK or BACK (Note 1)
NOTE 1 – This row applies to bidirectional transmissions.		

…

4. Revise the text of clause 8.6.1.1.4.1 "Registration request message (ADM_NodeRegistrRequest.req)" as follows:

…

Table 8-16.5 – List of Capabilities

Capability Type	Capability name	Capability Length value	Capability Value field
00_{16}	Bandplan Info	4	See Table 8-16.6
01_{16}	DMDefinedSeed	1	~~If set to 01_{16}, it~~ Bit map that indicates the capability of the node to use a DM-generated value as a domain specific seed (see clause 7.2.2.2.3 of [ITU-T G.9960])~~, 00_{16} otherwise~~. Bit 0: If set to one, it indicates the capability to set a new value for the first section of the preamble. Set to zero otherwise. All other values are Reserved by ITU-T
02_{16}	DM-generated initial seed for unloaded sub-carriers LFSR	1	It indicates that the node has the capability of handling a DM-generated initial seed different from the default value for the LFSR generator used when loading the unloaded supported sub-carriers. 0_{16} indicates no support for this capability 1_{16} indicates support for this capability All other values are reserved by ITU-T

Table 8-16.5 – List of Capabilities

Capability Type	Capability name	Capability Length value	Capability Value field
03_{16}	Piggybacked acknowledgements in BMSG	1	If set to 01_{16}, it indicates the capability of the node to use piggybacked delayed acknowledgements in BMSG (see clause 8.3.8). It shall be set to 00_{16} otherwise.
04_{16}	PSD-related Information	1	See Table 8-16.7
05_{16}	Run-Length Compression Support	1	See Table 8-16.8
06_{16}	Supported secure admission methods	3	Secure admission methods supported by the node. See Table 8-16.9.
07_{16}	Power Saving support	1	Bit map indicating the capability of the node to enter each of the power saving modes (see clause 8.21). Bit 0: Support for L0 mode. This bit shall always be set to one. Bit 1: If one, it indicates support for L1 mode. Bit 2: If one, it indicates support for L2 mode. Bit 3: If one, it indicates support for L3 mode. Bit 4: If one, it indicates support for L4 mode. Bits 5 to 7: Reserved by ITU-T (Note 1)
08_{16}	MMPL Compression support	1	See Table 8-16.10
08~~6~~$_{16}$-FF_{16}	Reserved by ITU-T		Reserved by ITU-T
NOTE – Bits that are reserved by ITU-T shall be set to zero by the transmitter and ignored by the receiver.			

…

Table 8-16.10~~9~~ – MMPL compression support capability value field

Field	Octet	Bits	Description
MMPL Compression Coding	0	[7:0]	This bitmap indicates the compression coding supported by the node Bit 0, if set to one, indicates that the node support Huffman code compression for MMPL Bits 1 to 7 are reserved by ITU-T (Note)
NOTE – Bits that are reserved by ITU-T shall be set to zero by the transmitter and ignored by the receiver.			

5. Revise the text of clause 8.10.1 "Management message format" as follows:

8.10.1 Management message format

Internal management messages, intended for communication between nodes of the same domain, shall be mapped into an LCDU payload field (see Figure 8-6). In-band management messages intended for communication with entities that reside locally above the A-interface of a node or above the A-interface of another node in the domain (see clause 8.1.1) shall be mapped into an APDU payload field (see Figure A-1). All management messages shall be formatted as shown in Figure 8-54, including a management message header (MMH) and a management message parameter list (MMPL or CMMPL) or a segment of MMPL or CMMPL (if segmentation is needed, see clause 8.10.1.2). The first byte (octet 0) of the MMH shall be the first byte of the LCDU/APDU payload, as described in clause 8.1.3.3. Encapsulation of the management message into an LLC frame is shown in Figure 8-7.

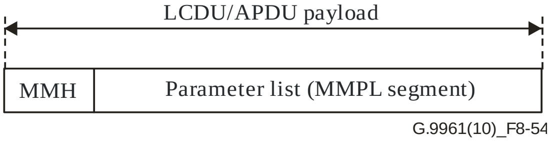

Figure 8-54 – Format of a management message

The MMH defines the length, the type, and other parameters of the message. The type of the message is identified by an OPCODE associated with a particular management function, as presented in Table 8-88. The MMPL (or CCMPL, see clause 8.24) includes a list of management message parameters, depending on the management function. The format of any management message shall be as shown in Table 8-87. An LCDU that contains a MAP message shall be carried only in the MAP or RMAP frame.

The format of MMPLs may be revised in future versions of this Recommendation by appending additional fields. Furthermore, fields may be defined using bits that are currently indicated as reserved for ITU-T. Nodes indicate the node information via NodeInformationTLVs during registration (see Table 8-16) and topology updates (see Table 8-47). Nodes shall be able to parse the MMPL (the length of the MMPL is specified in the MMH) but shall ignore the content of fields that they do not understand, i.e., those associated with later versions of the Recommendation.

Table 8-87 – Format of management messages

	Content	Octet	Bits	Description
MMH	Length	0 to 2	[11:0]	Length (LG) of the MMPL segment in octets, encoded as a 12-bit unsigned integer. The value of LG shall not exceed 1492.
	OPCODE		[23:12]	12-bit OPCODE, indicates message type (Note 1).
	MMPL Compression	3	[1:0]	Indicates whether MMPL data is compressed or not and which type of compression is used. - 00_2 if no compression is used - 01_2 if Huffman Coding is used Other values are reserved by ITU-T
	Reserved		[7:20]	Reserved by ITU-T (Note 4).
	Number of segments	4	[3:0]	Number of segments minus 1, represented as an unsigned integer between 0 and F_{16}. It shall be set to 0_{16} if the message is not segmented (Note 2).

	Segment number		[7:4]	Segment number, represented as an unsigned integer between 0_{16} and F_{16}; set to 0_{16} for the first segment and if message is not segmented (Note 2).
	Sequence number	5 and 6	[15:0]	Sequence number of the segmented message in a format of 16-bit unsigned integer (Notes 2 and 3).
	Repetition number	7	[3:0]	Repetition number of the message formatted as a 4-bit unsigned integer whose initial value is 0. Each time a message is retransmitted (See clause 8.10.1.2) by the originating node this field shall be incremented.
	FSB		[4]	Force Sequence Bit. See clause 8.10.1.2
	Reserved		[7:5]	Reserved by ITU-T (Note 4).
MMPL	Message Parameters	8 to (LG+7)	[(8×LG–1):0]	Depends on the OPCODE, see Table 8-88.
NOTE 1 – The OPCODES are defined in Table 8-88. NOTE 2 – This field is not applicable for a MAP message, and shall be set to zero. NOTE 3 – The meaning of the sequence number depends on the OPCODE. See clause 8.10.1.2. NOTE 4 – Bits that are reserved by ITU-T shall be set to zero by the transmitter and ignored by the receiver.				

6. Revise the text of clause 8.10.1.1 "Management message OPCODEs" as follows:

8.10.1.1 Management message OPCODEs

Management message OPCODEs are formatted as 12-bit unsigned integers. Valid values of OPCODEs are presented in Table 8-88. OPCODEs are categorized (typically by their associated protocol or procedure) according to the value of their eight MSBs.

Table 8-88 – OPCODEs of management messages

Category	Message name	OPCODE (hex)	Description	MMPL Reference
Admission (01X)	ADM_NodeRegistrRequest.req	010	Registration request	Clause 8.6.1.1.4.1
	ADM_DmRegistrResponse.cnf	011	Registration response	Clause 8.6.1.1.4.2
	ADM_NodeResignRequest.req	012	Resignation request	Clause 8.6.1.1.4.3
	ADM_DmResign.cnf	013	Registration announcement	Clause 8.6.1.1.4.4
	ADM_DmForcedResign.req	014	Forced resignation request	Clause 8.6.1.1.4.5
	ADM_NodeReRegistrRequest.req	015	Periodic re-registration request	Clause 8.6.1.1.4.6
	ADM_DmReRegistrResponse.cnf	016	Periodic re-registration response	Clause 8.6.1.1.4.7
	ADM_DmReRegistrInitiate.ind	017	Re-registration initiation request	Clause 8.6.1.1.4.8
	ADM_NodeReportMAPD.ind	018	Report the reception of a MAP-D with matching domain name	Clause 8.6.6.1.4.1
	ADM_NodeReportMAPA.ind	019	Report the reception of a MAP-A with matching DNI	Clause 8.6.6.1.4.2
	Reserved	01A-01F	Reserved by ITU-T	

Table 8-88 – OPCODEs of management messages

Category	Message name	OPCODE (hex)	Description	MMPL Reference
AKM (02X)	AUT_NodeAuthentication.req	020	Request for authentication	Clause 9.2.5.1.1
	AUT_Promp.ind	021	Delivers authentication prompt	Clause 9.2.5.1.2
	AUT_Verification.rsp	022	Authentication prompt verification	Clause 9.2.5.1.3
	AUT_Confirmation.cnf	023	Authentication confirmation message	Clause 9.2.5.1.4
	AKM_KeyRequest.req	024	Request for secure communication with another node(s)	Clause 9.2.5.2.1
	AKM_NewKey.req	025	Message delivers the encryption key to the supplicant node	Clause 9.2.5.2.2
	AKM_KeyConfirmation.req	026	Message delivers the encryption key to the addressee node(s)	Clause 9.2.5.2.4
	AKM_KeyUpdate.req	027	Request for re-authentication and update the keys	Clause 9.2.5.3.1
	AKM_NewKey.cnf	028	Addressee confirmation that encryption key was delivered	Clause 9.2.5.2.3
	SC_DMRes.req	029	Request to resign a node from the domain	Clause 9.2.5.2.5
	SC_DMRes.cnf	02A	Confirmation of resignation from the domain master	Clause 9.2.5.2.6
	AKM_AddClient.req	02B	Request to join a node to a multicast group	Clause 9.2.5.2.1.1
	Reserved	02C	Reserved by ITU-T	
	AKM_NewKey.ind	02D	Indication that the new encryption key is available for use	Clause 9.2.5.2.7
	AKM_DomainKeyUpdate.req	02E	Request to update the domain-wide encryption key, from SC to DM	Clause 9.2.5.3.2
	AKM_DomainKeyUpdate.cnf	02F	Confirmation for the request to update the domain-wide encryption key, from DM to SC	Clause 9.2.5.3.3

Table 8-88 – OPCODEs of management messages

Category	Message name	OPCODE (hex)	Description	MMPL Reference
Topology maintenance (03X)	TM_NodeTopologyChange.ind	030	Topology report from a node	Clause 8.6.4.3.1
	TM_NodeTopologyChange.req	031	Request sent by the domain master to a particular node requesting its topology report	Clause 8.6.4.3.2
	TM_NodeTopologyChange.cnf	032	Topology report from a node in response to the message TM_NodeTopology Change.req	Clause 8.6.4.3.3
	TM_DomainRoutingChange.ind	033	Optimal routing update from the domain master	Clause 8.6.4.3.5
	TM_ReturnDomainRouting.req	034	Request for routing update from the node to the domain master	Clause 8.6.4.3.6
	TM_ReturnDomainRouting.cnf	035	Reply on routing request by the Domain master	Clause 8.6.4.3.7
	TM_DMBackup.ind	036	Topology report from a node sent by backup domain master to a node	Clause 8.6.4.3.4
	Reserved	037-03F	Reserved by ITU-T	
Power-line coexistence with alien networks (04X)	Reserved for use by [ITU-T G.9972]	040-04F		
Multicast binding (05X)	MC_GrpInfoUpdate.ind	050	Multicast binding information update	Clause 8.16.5.1
	MC_GrpInfoUpdate.cnf	051	Multicast binding information update confirmation	Clause 8.16.5.2
	MC_GrpRemove.req	052	Multicast leave request from the transmitter	Clause 8.16.5.3
	MC_GrpRemove.cnf	053	Multicast leave confirmation from the receiver	Clause 8.16.5.4
	DMC_Path.req	054	DLL multicast path establishment request	Clause 8.17.6.1
	DMC_Path.cnf	055	DLL multicast path establishment confirmation	Clause 8.17.6.2
	DMC_PathReject.cnf	056	DLL multicast path establishment rejection	Clause 8.17.6.3
	DMC_EnforcePath.req	057	DLL multicast enforced path establishment request	Clause 8.17.6.4
	DMC_ReleasePath.req	058	A request to release a DLL multicast client node from its MSID	Clause 8.17.6.5

Table 8-88 – OPCODEs of management messages

Category	Message name	OPCODE (hex)	Description	MMPL Reference
	DMC_ReleasePath.cnf	059	Confirmation of the release of a DLL multicast client node from its MSID	Clause 8.17.6.6
	DMC_PathAlive.ind	05A	DLL multicast path alive indication	Clause 8.17.6.7
	DMC_BrokenLink.ind	05B	DLL multicast broken link indication	Clause 8.17.6.8
	Reserved	05C-05F	Reserved by ITU-T	
Domain master selection and backup domain master (06X)	DM_Handover.req	060	Domain master role handover request	Clause 8.6.6.5.1
	DM_Handover.cnf	061	Domain master role handover confirmation	Clause 8.6.6.5.2
	DM_Handover.ind	062	Domain state update	Clause 8.6.6.5.3
	DM_Handover.rsp	063	Domain state update confirmation	Clause 8.6.6.5.4
	DM_BackupAssign.req	064	Backup domain master assignment request	Clause 8.6.5.2
	DM_BackupAssign.cnf	065	Backup domain master assignment confirmation	Clause 8.6.5.2
	DM_BackupData.ind	066	Domain state update	Clause 8.6.5.2
	DM_BackupRelease.req	067	Release of a backup domain master	Clause 8.6.5.2
	DM_BackupRelease.cnf	068	Backup domain master release confirmation	Clause 8.6.5.2
	DM_HandoverRequest.ind	069	Endpoint node indication of need for domain master handover	Clause 8.6.6.5.5
	Reserved	06A-06F	Reserved by ITU-T	
Channel estimation (07X)	CE_ProbeSlotAssign.req	070	Channel estimation bandwidth assignment request	Clause 8.11.7.1
	CE_ProbeSlotRelease.req	071	Channel estimation bandwidth release request	Clause 8.11.7.2
	CE_ParamUpdate.req	072	Channel estimation parameters update request	Clause 8.11.7.3
	CE_ParamUpdateRequest.ind	073	Request for channel estimation parameter update	Clause 8.11.7.4
	CE_PartialBatUpdate.req	074	Partial BAT update request	Clause 8.11.7.5
	CE_ACESymbols.ind	075	Request for an ACE symbol attachment	Clause 8.11.7.6
	CE_ProbeSlotAssign.cnf	076	Channel estimation bandwidth assignment confirmation	Clause 8.11.7.7
	CE_ProbeSlotRelease.cnf	077	Channel estimation bandwidth release confirmation	Clause 8.11.7.8

Table 8-88 – OPCODEs of management messages

Category	Message name	OPCODE (hex)	Description	MMPL Reference
	CE_ParamUpdate.cnf	078	Channel estimation parameters update confirmation	Clause 8.11.7.9
	CE_PartialBatUpdate.cnf	079	Partial BAT update confirmation	Clause 8.11.7.10
	Reserved	07A-07F	Reserved by ITU-T	
Reserved (08X)	Reserved	080-08F	Reserved by ITU-T	
Inactivity scheduling (09X)	IAS_LongInactivity.req	090	Long inactivity scheduling request	Clause 8.3.6.1.1
	IAS_LongInactivity.cnf	091	Long inactivity scheduling confirmation	Clause 8.3.6.1.1
	IAS_ShortInactivity.req	092	Short inactivity scheduling request	Clause 8.3.6.2.1
	IAS_ShortInactivity.cnf	093	Short inactivity scheduling confirmation	Clause 8.3.6.2.1
	IAS_LongInactivity.ind	094	Long inactivity scheduling indication	Clause 8.3.6.1.1
	IAS_ShortInactivity.ind	095	Short inactivity scheduling indication	Clause 8.3.6.2.1
	Reserved	0964-09F	Reserved by ITU-T	
Flow establishment (0AX)	CL_EstablishFlow.req	0A0	Reserved by ITU-T	Clause 8.6.2.3.1
	CL_EstablishFlow.cnf	0A1	Reserved by ITU-T	Clause 8.6.2.3.2
	FL_AdmitFlow.req	0A2	Flow admission request	Clause 8.6.2.3.8
	FL_AdmitFlow.cnf	0A3	Flow admission confirmation	Clause 8.6.2.3.9
	FL_AdmitFlow.ind	0A4	Flow admission indication	Clause 8.6.2.3.10
	FL_AdmitFlow.rsp	0A5	Flow admission acknowledgement	Clause 8.6.2.3.18
	FL_OriginateFlow.req	0A6	Flow origination request	Clause 8.6.2.3.6
	FL_OriginateFlow.cnf	0A7	Flow origination confirmation	Clause 8.6.2.3.7
	Reserved	0A8-0AF	Reserved by ITU-T	
Flow maintenance (0BX)	FL_ModifyFlowParameters.req	0B0	Modification of flow parameters and allocation	Clause 8.6.2.3.11
	FL_ModifyFlowParameters.cnf	0B1		Clause 8.6.2.3.12
	FL_ModifyFlowParameters.ind	0B2		Clause 8.6.2.3.15
	FL_ModifyFlowAllocations.req	0B3	Modification of flow allocation	Clause 8.6.2.3.16
	FL_ModifyFlowAllocations.cnf	0B4		Clause 8.6.2.3.17
	Reserved	0B5-0BF	Reserved by ITU-T	
Flow termination (0CX)	CL_TerminateFlow. req	0C0	Flow termination request and confirmation	Clause 8.6.2.3.3
	CL_TerminateFlow.cnf	0C1		Clause 8.6.2.3.4
	CL_FlowTerminated.ind	0C2		Clause 8.6.2.3.5

Table 8-88 – OPCODEs of management messages

Category	Message name	OPCODE (hex)	Description	MMPL Reference
	FL_TerminateFlow.req	0C3	Request flow termination	Clause 8.6.2.3.13
	FL_TerminateFlow.cnf	0C4	Confirm flow termination	Clause 8.6.2.3.14
	FL_BrokenTunnel.ind	0C5	Indicate broken tunnel	Clause 8.6.2.3.19
	FL_BrokenTunnel.rsp	0C6	Response to indication	Clause 8.6.2.3.20
	FL_ReleaseTunnel.req	0C7	Request Release Tunnel	Clause 8.6.2.3.21
	FL_ReleaseTunnel.cnf	0C8	Confirm Release Tunnel	Clause 8.6.2.3.22
	FL_DM_RenewTunnel.req	0C9	DM renew tunnel request	Clause 8.6.2.3.23
	FL_DM_RenewTunnel.cnf	0CA	Confirm DM renew tunnel	Clause 8.6.2.3.24
	FL_RenewTunnel.req	0CB	Renew tunnel request	Clause 8.6.2.3.25
	FL_RenewTunnel.cnf	0CC	Confirm Renew tunnel	Clause 8.6.2.3.26
	FL_DeleteFlow.req	0CD	Delete Flow request	Clause 8.6.2.3.27
	FL_DeleteFlow.cnf	0CE	Confirm Delete Flow	Clause 8.6.2.3.28
	CL_TerminateFlow.ind	0CF	Indicate flow needs to be terminated	Clause 8.6.2.3.29
Media Access Plan (0DX)	MAP	0D0	MAP message	Clause 8.8
	Reserved	0D1-0DF	Reserved by ITU-T	
Channel Estimation 2 (0EX)	CE_Request.ind	0E0	Channel estimation trigger	Clause 8.11.7.11
	CE_Initiation.req	0E1	Channel estimation initiation request	Clause 8.11.7.12
	CE_Initiation.cnf	0E2	Channel estimation initiation confirmation	Clause 8.11.7.13
	CE_ProbeRequest.ind	0E3	Request for PROBE frame transmission	Clause 8.11.7.14
	CE_Cancellation.req	0E4	Channel estimation cancellation request	Clause 8.11.7.15
	CE_BatIdMaintain.ind	0E5	BAT ID maintenance	Clause 8.11.7.16
	CE_Cancellation.cnf	0E6	Channel estimation cancellation confirmation	Clause 8.11.7.17
	Reserved	0E7-0EF	Reserved by ITU-T	
Transmission Profile (0FX)	TP_TransmitPsd Change.req	0F0	Transmit PSD mask change request	Clause 8.6.9.1
	TP_TransmitPsd Change.cnf	0F1	Transmit PSD mask change confirmation	Clause 8.6.9.2
	Reserved	0F2-0FF	Reserved by ITU-T	
Neighbouring network coordination (10X to 13X)	NDIM_Start Alignment Procedure.ind	100	Request to start a MAC cycle alignment procedure (DM to proxy node)	Clause 8.14.9.1
	NDIM_IDCC Reserve.req	101	Slot reservation request	Clause 8.14.9.2
	NDIM_IDCC Reserve.cnf	102	Slot reservation confirmation	Clause 8.14.9.3
	NDIM_Report Alignment.req	103	Report on MAC cycle alignment	Clause 8.14.9.4

Table 8-88 – OPCODEs of management messages

Category	Message name	OPCODE (hex)	Description	MMPL Reference
	NDIM_ReportAlignment.cnf	104	Confirm receiving NDIM_ReportAlignment.req	Clause 8.14.9.5
	NDIM_Remote Presence.req	105	Request to respond to ID_PresenceRequest	Clause 8.14.9.6
	NDIM_Remote Presence.cnf	106	Permission to respond to ID_PresenceRequest	Clause 8.14.9.7
	NDIM_Transmit.ind	107	DM to proxy node message to be transmitted to neighbouring domain	Clause 8.14.9.8
	NDIM_Receive.ind	108	Proxy node to DM message received from neighbouring domain	Clause 8.14.9.9
	NDIM_InterferenceReport.ind	109	Indication of interference detected	Clause 8.14.9.12
	NDIM_IDCC_ Release.req	10A	Release Slot reservation	Clause 8.14.9.10
	NDIM_IDCC_ Release.cnf	10B	Confirm receiving NDIM_IDCC_Release.req	Clause 8.14.9.11
	Reserved	10C-10F	Reserved by ITU-T	
	IDM_Cluster Alignment.req	120	DM informs other DMs about new cluster alignment	Clause 8.14.10.1
	IDM_Cluster Alignment.cnf	121	DM confirm receiving IDM_ClusterAlignment.req	Clause 8.14.10.2
	IDM_InterfNodes Info.ind	122	Proxy node to neighbouring domains indication of interfering nodes	Clause 8.14.10.3
	IDM_CoordDomainsInfo.ind	123	Proxy node to neighbouring domains indication of coordinating nodes	Clause 8.14.10.5
	IDM_ShareUnalloc Slot.req	124	Request to share unallocated slots	Clause 8.14.10.7
	IDM_ShareUnalloc Slot.cnf	125	Confirmation of request to share unallocated slots	Clause 8.14.10.8
	IDM_ShareUnalloc Slot.ind	126	Indication of status of the request to share unallocated slots	Clause 8.14.10.9
	IDM_Request UnallocSlot.req	127	Request assignment of unallocated slots	Clause 8.14.10.10
	IDM_Request UnallocSlot.cnf	128	Confirmation of request for assignment of unallocated slots	Clause 8.14.10.11
	IDM_Request UnallocSlot.ind	129	Indication of status of the request for assignment of unallocated slots	Clause 8.14.10.12

Table 8-88 – OPCODEs of management messages

Category	Message name	OPCODE (hex)	Description	MMPL Reference
	IDM_SwapAlloc Slot.req	12A	Request to swap allocated slots	Clause 8.14.10.13
	IDM_SwapAlloc Slot.cnf	12B	Confirmation of the request to swap allocated slots	Clause 8.14.10.14
	IDM_SwapAlloc Slot.ind	12C	Indication of status of the request to swap allocated slots	Clause 8.14.10.15
	IDM_CoordPref.ind	12D	Indication of preferred coordination method	Clause 8.14.10.16
	IDM_DmChange.ind	12E	Indication to neighbouring domain masters that the DM of the domain sending the message has changed	Clause 8.14.10.17
	IDM_DniChange.ind	12F	Indication to neighbouring domain masters that the DNI of the domain sending the message has changed	Clause 8.14.10.18
	IDM_InterfNodes Info.rsp	130	A message sent as a confirmation for a received IDM_ InterfNodesInfo.ind	Clause 8.14.10.4
	IDM_CoordDomainsInfo.rsp	131	A message sent as a confirmation for a received IDM_ CoordDomainsInfo. ind	Clause 8.14.10.6
	Reserved	132-13F	Reserved by ITU-T	
AKM 2 (14X)	AUT_NodeAuthenticated.req	140	Indication from the SC to DM that the node has been authentication	Clause 9.2.5.1.5
	AUT_NodeAuthenticated.cnf	141	Confirmation of the AUT_NodeAuthenticated. req message	Clause 9.2.5.1.6
	SC_NewSCDesignation.req	142	New SC designation request for SC re-establishment	Clause 9.2.6.3.1
	SC_NewSCDesignation.cnf	143	confirmation of SC_NewSCDesignation.req	Clause 9.2.6.3.2
	Reserved	144-14F	Reserved by ITU-T	
Power saving modes (15X)	LP_RequestWakeUp.req	150		Clause 8.3.6.1.1
	Reserved	151-15F	Reserved by ITU-T	
L2 configuration and management protocol (16X)	LCMP_Read.req	160	Read request via LCMP protocol	Clause 8.22.3.1
	LCMP_Read.cnf	161	Read confirmation via LCMP protocol	Clause 8.22.3.2
	LCMP_Write.req	162	Write request via LCMP protocol	Clause 8.22.3.3
	LCMP_Write.cnf	163	Write confirmation via LCMP protocol	Clause 8.22.3.4

Table 8-88 – OPCODEs of management messages

Category	Message name	OPCODE (hex)	Description	MMPL Reference
	LCMP_Ctrl.req	164	Control operation request via LCMP protocol	Clause 8.22.3.5
	LCMP_Ctrl.cnf	165	Control operation confirmation via LCMP	Clause 8.22.3.6
	LCMP_Notify.ind	166	Notify operation request via LCMP protocol	Clause 8.22.3.7
	LCMP_Notify.rsp	167	Notify operation confirmation via LCMP	Clause 8.22.3.8
	LCMP.ind	168	For further study	Clause 8.22.3.9
	LCMP.rsp	169	For further study	Clause 8.22.3.10
	Reserved	16A-16F	Reserved by ITU-T	
Flow termination (17X)	FL_TerminateTunnel.req	170	Terminate tunnel request	Clause 8.6.2.3.30
	FL_TerminateTunnel.cnf	171	Terminate tunnel confirmation	Clause 8.6.2.3.31
	Reserved	172-17F	Reserved by ITU-T	
Bandwidth update protocol (18X)	BU_BWUpdate.req	180	Request for bandwidth update for prioritized connections	Clause 8.6.2.4.1.1
	BU_BWUpdate.cnf	181	Confirmation for bandwidth update for prioritized connections	Clause 8.6.2.4.1.2
	Reserved	182-18F	Reserved by ITU-T	
Reserved	Reserved	190-7FF	Reserved by ITU-T	
MIMO (8XX – 9XX)	Reserved for use by [ITU-T G.9963]	800-9FF	Reserved by ITU-T	
HGF (AXX – A3X)	Reserved for use by HomeGrid Forum	A00-A3F	Reserved for use by HomeGrid Forum	
Reserved	Reserved	A40-FFF	Reserved by ITU-T	

7. Revise the text of clause 8.10.1.2 "Management of message sequence numbers and segmentation" as follows:

8.10.1.2 Management of message sequence numbers and segmentation

The sequence number space shall be unique for each {OPCODE, OriginatingNode} tuple. The sequence number shall be incremented for each transmitted message except as follows:

- When the same message is retransmitted (e.g., when a message has been lost), the message sequence number shall be the same as the original transmitted message and the repetition number shall be incremented by 1;
- When a message is relayed, the sequence number and the repetition number fields shall not be modified.

NOTE 1 – The sequence number space used by an originating node for a given OPCODE is the same regardless of the destination (e.g., single counter per OPCODE).

When the field Force Sequence Bit (FSB) of the MMH is set to one, it indicates that the receiver shall process this message without performing any sequence filtering. The receiver shall also consider the sequence number of this message as the latest valid sequence number associated with the transmitter's DeviceID and OPCODE of the message.

NOTE 2 – The increment in the value of the message sequence number is independent of the value of the FSB field.

The following segmentation rules apply to any segmented management messageLCDU:

- The segmentation shall be done in the ascending order of octets;
- All the segments shall have the same sequence number;
- The segmentation shall not be changed if the management messageLCDU is retransmitted, unless a new sequence number is generated;
- The segmentation shall not be changed if the management messageLCDU is relayed (the sequence number shall remain the same).

Segmentation shall only be done for management messagesLCDUs with payload (MMPL or CMMPL) that lead to LCDUs greater than 1500 bytes.

Some management protocols may require knowing if the sequence number of a received LCDU is older, equal or newer than the last correctly received LCDU. The sequence number is a 16-bit unsigned integer used for that purpose and shall be in the range 0 to (SequenceModulus –1), where SequenceModulus is equal to 2^{16}. When it is equal to 2^{16}, it wraps-around to zero. If the FSB field of the MMH is set to one, the received LCDU shall be considered as the newest. If the FSB field of the MMH is set to zero, sequence numbers of LCDUs with the same OPCODE shall be compared according to the following rules:

- The first LCDU received from a node shall be considered as a new message containing new information. The node shall perform the operations required by the protocol that defines that OPCODE;
- If the sequence number of the new received LCDU is the same as the sequence number of the LCDU already kept by the node, the new received LCDU shall be considered to be equal to the LCDU kept by the node;
- If the sequence number of the new received LCDU is higher than the sequence number of the LCDU already kept by the node and the difference between the numbers is, in absolute value, less than half of SequenceModulus, the new received LCDU shall be considered to be newer. Otherwise it shall be considered to be older;
- If the sequence number of the new received LCDU is lower than the sequence number of the LCDU already kept by the node and the difference between the numbers is, in absolute value, lower than half of SequenceModulus, the new received LCDU shall be considered to be older. Otherwise it shall be considered to be newer.

In any of the above cases, the actions to perform by the node that receives the LCDU depend on the protocol that defines that OPCODE.

NOTE 3 – A transmitter may use the FSB bit to force synchronization with the receiver. Once the transmitter gets confirmation that the receiver is synchronized, it should set FSB to zero.

8. Add text for new clause 8.24 "Payload compression in management messages" as follows:

8.24 Payload compression in management messages

When a node detects that the transmitter and receiver in a particular link support the compression of management frames ("MMPL Compression support" capability, see clause 8.6.1.1.4.1), it can compress the payload (MMPL) of a management frame using one of the allowed compression algorithms to create a new compressed version of the MMPL (CMMPL – Compressed MMPL).

The compressed management frame uses the CMMPL instead of the MMPL to generate the MMH information (e.g. Length field) and for segmentation purposes.

8.24.1 Huffman coding of MMPL

When using Huffman coding, the transmitter node shall set the MMPL Compression field of the MMH of management messages to 01_2 (see Table 8-87 in clause 8.10.1).

The procedure to build the Huffman tree shall be as follows:

- Do a pre-order traversal, writing each node visited.
- Differentiate leaf nodes from internal/non-leaf nodes.
- Write a single bit for each node, 1 for leaf and 0 for non-leaf.
- For leaf nodes, write the character stored.
- For non-leaf nodes there's no information that needs to be written, just the bit that indicates there's an internal node.
- Huffman tree and Huffman coded message bits are written from MSB to LSB. The MSB of byte 0 is the first bit of the Huffman tree.

If by the application of the compression algorithm the resulting compressed message length is not byte-aligned (i.e. not a multiple of 8 bits), the compressed message shall include padding bits (set to zero) to complete a whole byte.

NOTE – An implementation of this mechanism can be found at [b-BCL]

9. Revise the text of Annex X "Test vectors" as follows:

Annex X

Test vectors

(This annex forms an integral part of this Recommendation.)

This annex includes test vectors for core operations described in this Recommendation.

X.1 CCM encryption

This clause provides a set of test vectors for parameters involved in CCM encryption described in clause 9.1. Parameters are expressed in a hexadecimal form with the leftmost byte representing the lowest byte within a parameter (i.e., byte 0). Within a byte, the leftmost bit represents the MSB.

X.1.1 CCM test vector 1

X.1.1.1 Input parameters

This clause provides one set of examples for input parameters used in CCM encryption.

Data packet, APDU (75 bytes):

The APDU can be broken into the following parameters:

Destination MAC address, *DA* (6 bytes)

DA =   00 00 5E 10 20 09_{16}

Source MAC address, *SA* (6 bytes)

SA =   00 00 5E 07 20 13_{16}

MAC client length/type, *LT* (2 bytes)

LT =   08 00_{16}

APDU payload, *P* (57 bytes)

P =   52 41 56 49 3B 45 52 45 5A 3B 41 56 4E 45 52 3B 4C 45 53 3B 52
4F 59 3B 4D 41 52 43 4F 53 3B 41 47 55 53 54 49 4E 3B 4A 4F 48
4E 3B 4A 42 3B 54 4F 4E 47 3B 56 5A 45 49 42_{16}

Frame check sequence, *FCS* (4 bytes)

FCS =   A4 55 5A 26_{16}

Note that VLAN TAG does not exist in this example (i.e., *TG* = 0), *Alen* is 21 bytes (see Table 9-5), and *Plen* is 57 bytes.

Parameters for the LLC frame header (LFH) are given as LLCFT = 2 (APDU), TSMPI = 0 (TSMP field not present), CCMPI = 1, LPRI = 0, FLEN = 71, MCSTI = 0, OriginatingNode = 1, DestinationNode = 2, BRCTI = 0, and TTL = 0 (see clause 8.1.3.1.1).

Resulting LFH (6 bytes):

LFH =   12 47 00 01 02 00_{16}

Parameters for the CCMP header (CCMPH) are given as MIC size = 111_2 (16-byte MIC), NN or NMK, Key ID = 0, and FN = 1 (see clause 9.1.2.3).

Resulting CCMP header, CCMPH (6 bytes):

CCMPH = 07 01 00 00 00 00_{16}

Encryption Key K: 47 68 6F 43 65 72 74 66 32 30 31 33 47 68 6E 43_{16}.

X.1.1.2 Parameters generated

This clause provides one set of examples of parameters generated by the node based on given set of input parameters.

Nonce, *N* (13 bytes):

N =   00 00 00 5E 07 20 13 00 00 00 00 00 01_{16}

Nonce block, B_0 (16 bytes):

B_0 =   79 00 00 00 5E 07 20 13 00 00 00 00 00 01 00 39_{16}

Associated data block, B_1 (16 bytes):

B_1 =   00 15 00 00 00 00 5E 10 20 09 00 00 5E 07 20 13_{16}

Associated data block, B_2 (16 bytes):

B_2 =   12 47 00 01 02 08 00 00 00 00 00 00 00 00 00 00_{16}

Payload block, B_3 (16 bytes):

B_3 =   52 41 56 49 3B 45 52 45 5A 3B 41 56 4E 45 52 $3B_{16}$

Payload block, B_4 (16 bytes):

B_4 =   4C 45 53 3B 52 4F 59 3B 4D 41 52 43 4F 53 3B 41_{16}

Payload block, B_5 (16 bytes):

B$_5$ =   47 55 53 54 49 4E 3B 4A 4F 48 4E 3B 4A 42 3B 54$_{16}$

Payload block, B_6 (16 bytes):

B$_6$ =   4F 4E 47 3B 56 5A 45 49 42 00 00 00 00 00 00 00$_{16}$

Counter block 0, Ctr_0 (16 bytes):

Ctr_0 =   01 00 00 00 5E 07 20 13 00 00 00 00 00 01 00 00$_{16}$

Counter block 1, Ctr_1 (16 bytes):

Ctr_1 =   01 00 00 00 5E 07 20 13 00 00 00 00 00 01 00 01$_{16}$

Counter block 2, Ctr_2 (16 bytes):

Ctr_2 =   01 00 00 00 5E 07 20 13 00 00 00 00 00 01 00 02$_{16}$

Counter block 3, Ctr_3 (16 bytes):

Ctr_3 =   01 00 00 00 5E 07 20 13 00 00 00 00 00 01 00 03$_{16}$

Counter block 4, Ctr_4 (16 bytes):

Ctr_4 =   01 00 00 00 5E 07 20 13 00 00 00 00 00 01 00 04$_{16}$

Cyphertext

C =   04 4f 24 21 07 fb 58 68 ba 1a c7 c3 1f 5c e7 20 c1 a2 09 ed a0
29 d5 03 b1 e0 94 43 ed 4f 28 24 62 c8 28 c5 53 50 95 74 86 fc
ea 0e 92 4d 2c 4f 3b 25 cf b5 3a 5e 1f 5e 3f$_{16}$

Message Integrity code

MIC =   8c df 6e 79 03 0f 7e 69 cc 33 b8 29 ef e4 6d e2$_{16}$

X.2 PAK test vectors

This clause provides a set of test vectors for parameters involved in the PAK protocol described in clause 9.2.2. Parameters are expressed in a hexadecimal form with the leftmost bit representing the MSB.

X.2.1 PAK test vector 1

X.2.1.1 Input parameters

This clause provides one set of examples for input parameters known to the supplicant and the authenticator before the initiation of key authentication process.

Node identifier of the supplicant, *A* (48 bits, see clause 9.2.2.2.1):

A =   0019 A717 DD30$_{16}$

Node identifier of the authenticator, *B* (48 bits, see clause 9.2.2.2.1):

B =   0019 A770 8A32$_{16}$

Node password shared by the supplicant and the authenticator, *PW* (96 bits, see clause 9.2.2.2.2):

PW =   5962 A05A B89F C0AA FB14 0EF7$_{16}$

X.2.1.2 Parameters generated or exchanged

This clause provides one set of examples of parameters generated and/or exchanged by the supplicant and the authenticator.

Secret exponent generated by the supplicant, *RA* (384 bits, see clause 9.2.2.2.5):

RA =   89A1 A7B4 F433 9220 2C60 960D 172A 7C45 6B95 C225 26B1 1C7A
9E2E 7712 C43C 9C77 48B6 3936 A62B CF90 3C03 A0E2 0E28 D660$_{16}$

Secret exponent generated by the authenticator, *RB* (384 bits, see clause 9.2.2.2.5):

```
RB =    F052 57CB 1840 6A91 173B 87E4 1F22 9289 7D3E 08A7 BCA0 4EB9
        1A8A CFF3 940C AE00 E15B 302D 7E67 2E81 CCB4 C103 A241 B133₁₆
```

Concatenated input parameters, $P = A \mid B \mid PW$ (192 bits, see [ITU-T X.1035]):

```
P =     0019 A717 DD30 0019 A770 8A32 5962 A05A B89F C0AA FB14 0EF7₁₆
```

Intermediate result, $IR1 = H_1(P)$ (1152 bits, see clause 9.2.2.2.6 and [ITU-T X.1035]):

```
IR1 =   11F9 E6DD 6E7D 48EF 3672 CA0F A2EC 2488 7678 34B9 506C FE86
        5BC0 A847 3051 F6FD 408D 0178 816D 80A7 D8D3 B75C 3176 C8D3
        BD12 2AD1 2AE5 C26C 29F8 3518 BD91 1581 9483 C303 68F3 B137
        3A33 A8E5 6193 83B8 34F1 59B4 E1C3 8259 B3DA D35F 7876 A7FE
        3B0A 9E9A F594 BEA6 B126 77B0 50EC 672E 11F7 3A1E 231E 9ECE
        793A 34AE 154D 4EB0 82BB AC26 1F8E 0B50 735C 01FB C364 9081₁₆
```

Intermediate result, $IR2 = g^{RA} \bmod p$ (1024 bits, see clause 9.2.2.2.3, clause 9.2.2.2.4, and [ITU-T X.1035]):

```
IR2 =   D678 B9D6 E866 FB46 4865 A430 C2BA 0668 722D 236E 7BEA 1C51
        7E4A 4812 1CD4 B42C 7803 2B8C F05F 497B 46EC F894 CB5A 0678
        7104 7E99 448A D384 46A1 15AF 4640 7B9B F13C FFBD 2452 FB69
        3D7C 6445 DE1E 95AF DC13 7B33 01AE 6659 0839 A05E 03A2 2169
        E10C C5F6 D87B 62E5 FF92 B000 4DA9 8058 9F95 5F2E F66A 42D6
        CBC4 E70A A3CA D136₁₆
```

Parameter carried in AUT_NodeAuthentication.req, $X = IR1 \cdot IR2$ (2176 bits, see clause 9.2.5.1.1 and [ITU-T X.1035]):

```
X =     0F0F 612E 0137 3C14 AB36 88FB 07C9 98E6 EBA7 033C E635 4EDA
        54D2 DA67 46D2 43AC FC19 3F9E 7E66 4B5F 1ED8 13D7 7763 0BFF
        DE60 E3D5 397E 901A 1338 99CC 2E52 209E 441F 0DDE 9449 1CDA
        8B36 B454 FF1B 1E9E 784A 07D4 5DF5 85C5 503E 65AD 7E34 EE82
        2E92 99AC B766 EF21 0CEB 7D10 B620 AB10 BA09 7DF7 EEB0 25BE
        E6AD 223B 3049 93F9 FCDB C996 EA09 8BFC 56C7 495E 2E17 BD88
        E201 B2C2 40E9 9F79 B681 9963 3D8F 5F22 7BD8 5373 A868 902D
        93FC 20CB 9F1D 369B 1C54 A143 E416 D7C5 2A59 EAC8 0B49 D013
        575F C302 FA4D AD02 DDF7 BA96 71E9 9B56 DE44 9E57 9DFB 83AD
        B1F1 1A43 0900 2F9C 8EFD A771 0A71 DAA0 176D E5ED C7A8 02F3
        99D8 6E26 0458 3EF1 901F 7C1A 99E8 CBB7 5357 09DA 84F2 5393
        9F2E 3706 79F9 CC36₁₆
```

Intermediate result, $IR3 = H_2(P)$ (1152 bits, see clause 9.2.2.2.6 and [ITU-T X.1035]):

```
IR3 =   2773 D699 51BB 3CC5 D595 F28E 3AAF CCBF C2A3 895D D429 A707
        13EE C1D7 2E08 BCA9 D3C7 AE45 7317 5180 25AE 9B9D 6125 BED6
        EA69 F440 FD1F D309 2404 0AD9 E3DB B2A4 8F1A 49DA 0F14 BD2E
        15B7 2E9D E16E 9E95 EE26 6890 AA45 1ACE A1A7 394C 9BFB 55B8
```

```
            54DE 5CFB 1385 028D 3A58 ED53 C8B1 639C 76D4 F4AF BB51 52D8
            2E7F F099 4210 DA52 CDFD DF2B 973D EC89 DFEB A32C A4B7 4428₁₆
```

Intermediate result, $IR4 = g^{RB}$ *mod p* (1024 bits, see clause 9.2.2.2.3, clause 9.2.2.2.4, and [ITU-T X.1035]):

```
    IR4 =   B503 D0FE AAC7 D9D5 B2C1 ADAC ACB2 F4AC ED7D E0EA 65F2 D88F
            39DA A98C CCA3 C197 40F4 B466 6DC4 310F 6969 482F 2B94 D5A2
            BB64 4E8F 04A7 12D4 81FF 34E0 45F3 E351 E255 3A57 F32F E600
            820A 7C9B 0407 F35C 588D 4C6A 0908 BD7C 9F76 A9BB A478 16BB
            C6F8 73DC B9EF C0CD 54FC E949 F840 2EE6 DD0C D4B9 52F9 96BC
            D529 9885 964C 394D₁₆
```

Parameter carried in AUT_Promp.ind, $Y = IR3 \cdot IR4$ (2176 bits, see clause 9.2.5.1.2 and [ITU-T X.1035]):

```
    Y =     1BE5 7D4B 4832 8ED0 90C6 5623 C4D1 1400 F58E FDEF A37E 2AAC
            0EB6 9A5E 904E F71A 193D 46B2 6113 372A 0517 45CB 1FCB 5200
            2FF9 A00C 9070 72C2 5946 E87D 630E 36A2 AFCE 5FB4 AA35 D2F7
            74DA FED9 11A4 4EF4 698C 4582 9E47 8AA6 F74A 6714 09E0 8CA3
            9654 1D65 9099 DC16 3C40 2E8D 6779 D9CD B182 4EF6 6A83 8A40
            7537 64A9 ABDB 1619 33A7 44E6 8C9A 3D37 2D34 1C46 423F 4679
            B03B 563D 0B02 D397 6171 776F 7FD4 31D6 6B26 4F6A 5AC5 BD89
            434C C914 2698 36A0 DC88 2E31 D3FD B108 69FA 4F86 AFAB CDDC
            7CE6 D753 F7C5 5286 6E12 C2C3 80E4 70A2 6F81 08CD D379 08F5
            AE54 4467 DA86 974D BE27 39A6 5058 E201 1387 AB08 6402 15A3
            E973 5002 8852 6DD8 302B 60F8 28AF 9806 4535 F825 425C 0652
            010F 763A 052C 6808₁₆
```

Intermediate result, $IR5 = P \mid g^{RA} \bmod p \mid g^{RB} \bmod p \mid g^{RA \cdot RB} \bmod p$ (3264 bits, see clause 9.2.2.2.3, clause 9.2.2.2.4, and [ITU-T X.1035]):

```
    IR5 =   0019 A717 DD30 0019 A770 8A32 5962 A05A B89F C0AA FB14 0EF7
            D678 B9D6 E866 FB46 4865 A430 C2BA 0668 722D 236E 7BEA 1C51
            7E4A 4812 1CD4 B42C 7803 2B8C F05F 497B 46EC F894 CB5A 0678
            7104 7E99 448A D384 46A1 15AF 4640 7B9B F13C FFBD 2452 FB69
            3D7C 6445 DE1E 95AF DC13 7B33 01AE 6659 0839 A05E 03A2 2169
            E10C C5F6 D87B 62E5 FF92 B000 4DA9 8058 9F95 5F2E F66A 42D6
            CBC4 E70A A3CA D136 B503 D0FE AAC7 D9D5 B2C1 ADAC ACB2 F4AC
            ED7D E0EA 65F2 D88F 39DA A98C CCA3 C197 40F4 B466 6DC4 310F
            6969 482F 2B94 D5A2 BB64 4E8F 04A7 12D4 81FF 34E0 45F3 E351
            E255 3A57 F32F E600 820A 7C9B 0407 F35C 588D 4C6A 0908 BD7C
            9F76 A9BB A478 16BB C6F8 73DC B9EF C0CD 54FC E949 F840 2EE6
            DD0C D4B9 52F9 96BC D529 9885 964C 394D 9768 81A5 5808 E976
            F569 319A 8764 8539 16E0 1496 6E1F 191A 482B 1838 0E4F 9A77
            99FA C4AF AE0B 9C74 7A57 630C DA71 DF19 5CB2 FE5F B951 52B7
```

EADB C460 8B62 3464 944E 1011 8471 028C 8000 8F8E EC8E B6C7
FC36 30DF 27DD 2D43 3277 2FB4 E1A8 FF9F CA61 6E4E E466 CDA4
B6AD 9B02 F498 39BF 589B C793 2680 8C26 9AA6 B351 9418 $EFEB_{16}$

Parameter carried in AUT_Promp.ind, $S_1 = H_3(IR5)$ (128 bits, see clause 9.2.2.2.6, clause 9.2.5.1.2, and [ITU-T X.1035]):

S_1 = 3BB5 5C57 33CF 1E7F 0711 C525 CD89 3181_{16}

Parameter carried in AUT_Verification.res, $S_2 = H_4(IR5)$ (128 bits, see clause 9.2.2.2.6, clause 9.2.5.1.3, and [ITU-T X.1035]):

S_2 = 3DB1 A72C 64B0 CAE6 57FF D4EA DC31 $F676_{16}$

NSC key generated, $K = H_5(IR5)$ (128 bits, see clause 9.2.2.2.6 and [ITU-T X.1035])

K = ABA6 D8E8 BD2B 705B B4CC 34BD 1107 $E00D_{16}$

X.3 Huffman coding

Huffman coding may be used to compress management frame payloads as described in clause 8.24.

X.3.1 Test vector 1

X.3.1.1 Input (uncompressed) management message payload (MMPL)

MMPL (329 bytes)

Offset	0x0	0x1	0x2	0x3	0x4	0x5	0x6	0x7	--	0x8	0x9	0xa	0xb	0xc	0xd	0xe	0xf
0x000	0x56	0x69	0x63	0x74	0x6f	0x72	0x20	0x5a		0x65	0x69	0x62	0x20	0x42	0x6f	0x72	0x74
0x010	0x0a	0x4d	0x61	0x72	0x63	0x6f	0x73	0x20		0x4d	0x61	0x72	0x74	0x69	0x6e	0x65	0x7a
0x020	0x20	0x56	0x61	0x7a	0x71	0x75	0x65	0x7a		0x0a	0x21	0x5a	0x6e	0x68	0x2c	0x54	0x6e
0x030	0x75	0x67	0x70	0x73	0x69	0x1f	0x6c	0x6d		0x70	0x76	0x74	0x1f	0x75	0x67	0x66	0x1f
0x040	0x68	0x60	0x75	0x64	0x2f	0x1f	0x5a	0x6e		0x68	0x2c	0x54	0x6e	0x75	0x67	0x70	0x73
0x050	0x69	0x1f	0x6a	0x72	0x21	0x73	0x69	0x64		0x21	0x66	0x62	0x73	0x66	0x2d	0x21	0x58
0x060	0x70	0x66	0x2e	0x52	0x70	0x73	0x69	0x6e		0x75	0x67	0x21	0x68	0x74	0x1f	0x75	0x67
0x070	0x66	0x1f	0x6c	0x64	0x7a	0x1f	0x62	0x6d		0x65	0x1f	0x68	0x74	0x62	0x71	0x65	0x68
0x080	0x62	0x6d	0x21	0x6e	0x67	0x1f	0x75	0x67		0x66	0x1f	0x68	0x60	0x75	0x64	0x2f	0x1f
0x090	0x51	0x60	0x74	0x73	0x2d	0x1f	0x71	0x71		0x66	0x72	0x66	0x6d	0x75	0x2b	0x21	0x65
0x0a0	0x76	0x73	0x76	0x71	0x66	0x2b	0x21	0x60		0x6d	0x6b	0x21	0x60	0x73	0x64	0x21	0x6e
0x0b0	0x6f	0x64	0x21	0x68	0x6f	0x1f	0x5a	0x6e		0x68	0x2c	0x54	0x6e	0x75	0x67	0x70	0x73
0x0c0	0x69	0x2d	0x21	0x47	0x66	0x1f	0x6c	0x6d		0x70	0x76	0x74	0x1f	0x78	0x67	0x66	0x71
0x0d0	0x66	0x1f	0x75	0x67	0x66	0x1f	0x50	0x6b		0x65	0x1f	0x50	0x6d	0x66	0x72	0x21	0x61
0x0e0	0x73	0x6e	0x6c	0x64	0x21	0x73	0x69	0x71		0x70	0x74	0x68	0x67	0x21	0x6e	0x67	0x1f
0x0f0	0x70	0x6b	0x65	0x2b	0x21	0x60	0x6f	0x63		0x21	0x76	0x69	0x64	0x73	0x64	0x21	0x53
0x100	0x69	0x64	0x7a	0x1f	0x74	0x67	0x62	0x6b		0x6d	0x1f	0x63	0x71	0x66	0x60	0x6c	0x1f
0x110	0x75	0x67	0x73	0x6e	0x76	0x66	0x69	0x1f		0x62	0x66	0x62	0x68	0x6f	0x2d	0x23	0x09
0x120	0x21	0x1f	0x21	0xe1	0x81	0x92	0x49	0x2d		0x51	0x2d	0x21	0x4b	0x70	0x75	0x66	0x62
0x130	0x73	0x60	0x67	0x73	0x2d	0x1f	0x55	0x67		0x66	0x1f	0x45	0x74	0x6f	0x76	0x6a	0x62
0x140	0x69	0x1f	0x49	0x6e	0x73	0x71	0x70	0x71		0x0b							

X.3.1.2 Compressed management message payload

CMMPL (282 bytes)

Offset	0x0	0x1	0x2	0x3	0x4	0x5	0x6	0x7	--	0x8	0x9	0xa	0xb	0xc	0xd	0xe	0xf
0x000	0x00	0xa9	0xd4	0x95	0x8a	0xa9	0x45	0xa1		0x29	0x74	0x72	0xd5	0x56	0x30	0x37	0x87
0x010	0x87	0x24	0x58	0xec	0x22	0x17	0x09	0x4b		0xa4	0x76	0xb2	0xdb	0x65	0x14	0xda	0x4a
0x020	0xa3	0x50	0xb7	0x96	0x7b	0x99	0x60	0x5e		0xad	0x96	0x60	0xb8	0xd9	0x16	0x2b	0xa2
0x030	0x42	0xb8	0x5a	0x01	0x54	0x96	0x25	0x69		0x7c	0x2a	0xd2	0x5d	0xae	0x57	0x54	0x7d
0x040	0x6e	0x4b	0x55	0xa9	0x00	0x5c	0x89	0x33		0xeb	0xff	0xf1	0x73	0x2f	0xe0	0xa7	0xd6
0x050	0x78	0xe6	0x05	0xd4	0x43	0xaf	0xf3	0x02		0xeb	0x3c	0xf8	0xad	0x7f	0x0a	0x2b	0x50
0x060	0xd9	0x5a	0xc7	0xaf	0xde	0xbe	0x1c	0x1e		0xda	0x59	0x73	0xcd	0x92	0xda	0x4f	0xb6
0x070	0x8f	0xd7	0x66	0xe3	0x8d	0xdf	0xbd	0x7c		0x38	0x3d	0xb4	0xb2	0xe7	0x82	0x6b	0x4b
0x080	0x98	0xd3	0xc9	0x5f	0xea	0x02	0xb3	0x8d		0x00	0xd9	0x73	0xed	0xa5	0x5e	0x7d	0xb4
0x090	0x7e	0x6e	0x2d	0x74	0x88	0x5e	0xbc	0xe5		0x02	0xde	0x44	0xaf	0x27	0x6d	0x1f	0xae
0x0a0	0xcd	0xc7	0x1b	0x8d	0x32	0x6b	0xf7	0x42		0x0f	0xab	0x93	0x78	0xa8	0xba	0x2e	0x90
0x0b0	0x7c	0x54	0xc2	0x0f	0x4c	0x58	0xd7	0x8f		0x1a	0xb9	0xfb	0xf7	0xaf	0x87	0x07	0xb6
0x0c0	0x96	0x5c	0xfd	0x40	0xef	0xcd	0x92	0xda		0x4f	0x83	0x51	0xe0	0xfd	0xb4	0x7e	0x36
0x0d0	0x39	0x78	0xd9	0x1f	0x5a	0x15	0x7c	0xdc		0x69	0x73	0x0b	0x4e	0xe9	0x5e	0x4e	0xb0
0x0e0	0xe5	0xc5	0x4c	0x38	0x95	0xa6	0x62	0xb1		0xa0	0x0c	0xc5	0xae	0x9a	0x48	0x72	0x70
0x0f0	0x90	0x76	0x37	0xdb	0x45	0xf6	0x8f	0x9e		0x93	0xca	0xe7	0xf8	0x6c	0x66	0xba	0x83
0x100	0x83	0x03	0xcc	0xfc	0x6b	0xea	0x06	0xb6		0xde	0x4a	0xc4	0x5f	0xb8	0x0d	0x1f	0x81
0x110	0x26	0x7d	0x02	0x4b	0x3c	0x67	0xcb	0x0b		0x40	0x60						

Huffman Tree (549 bits = 68 bytes + 5bits) as present at CMMPL:

Offset	0x0	0x1	0x2	0x3	0x4	0x5	0x6	0x7	--	0x8	0x9	0xa	0xb	0xc	0xd	0xe	0xf
0x000	0x00	0xa9	0xd4	0x95	0x8a	0xa9	0x45	0xa1		0x29	0x74	0x72	0xd5	0x56	0x30	0x37	0x87
0x010	0x87	0x24	0x58	0xec	0x22	0x17	0x09	0x4b		0xa4	0x76	0xb2	0xdb	0x65	0x14	0xda	0x4a
0x020	0xa3	0x50	0xb7	0x96	0x7b	0x99	0x60	0x5e		0xad	0x96	0x60	0xb8	0xd9	0x16	0x2b	0xa2
0x030	0x42	0xb8	0x5a	0x01	0x54	0x96	0x25	0x69		0x7c	0x2a	0xd2	0x5d	0xae	0x57	0x54	0x7d
0x040	0x6e	0x4b	0x55	0xa9	***0x00***												

The figure shown in bold italic style in the table above indicates that its content is shared between the Huffman Tree and the Compressed payload (5 MSB belongs to Huffman Tree and 3 LSB belongs to Compressed payload).

This Huffman Tree, with its bits ordered from LSB to MSB, is the following:

00000000101010011101010010010101100010101010100101000101101000010010100101110100
01110010110101010101011000110000001101111000011110000111001001000101100011101100
00100010000101110000100101001011101001000111011010110010110110110110010100010100
1101101001001010101000110101000001011011111001011001111011100110010110000001011110
10101101100101100110000001011100011011001000101100010101110100010010000101011 1000
01011010000000010101010010010110001001010110100101111100001010101101001001011101
101011100101011101010100011111010110111001001011010101011010100100000

Which has the following structure:

Code TracePath	Coded Char Bits	Coded Char Hex
00000000		
1	01010011	0x53
1	01010010	0x52
0		
1	01011000	0x58
1	01010101	0x55
00		
1	01000101	0x45
1	01000010	0x42
0		
1	01001011	0x4B
1	01000111	0x47
00		
1	01101010	0x6A
1	01010110	0x56
00		
1	10000001	0x81
1	01111000	0x78
0		
1	11100001	0xE1
1	10010010	0x92
00		
1	01100011	0x63
1	01100001	0x61
000		
1	00001011	0x0B
1	00001001	0x09
0		
1	00101110	0x2E
1	00100011	0x23
1	01101011	0x6B
00		
1	01101101	0x6D
1	01100101	0x65
000		
1	01001101	0x4D
1	01001001	0x49
0		
1	01010001	0x51
1	01010000	0x50
1	01101111	0x6F
00		
1	01100111	0x67
1	01110011	0x73
00		
1	01100000	0x60
0		
1	01111010	0x7A
1	01101100	0x6C
1	01100110	0x66
0000		
1	01110001	0x71
1	01100100	0x64
0		

Code TracePath	Coded Char Bits	Coded Char Hex
1	01100010	0x62
1	01110100	0x74
0		
1	00100001	0x21
0		
1	01110000	0x70
1	01101000	0x68
00000		
1	01010100	0x54
1	00101100	0x2C
0		
1	00101011	0x2B
0		
1	00101111	0x2F
1	00001010	0x0A
1	01101001	0x69
00		
1	01110110	0x76
1	01110010	0x72
1	01110101	0x75
0		
1	00011111	0x1F
0		
1	01101110	0x6E
0		
1	00101101	0x2D
0		
1	01011010	0x5A
1	00100000	0x20

And codes as shown in the following table:

Code	Symbol	Code Bits
0x00	0x53	8
0x01	0x52	8
0x02	0x58	8
0x03	0x55	8
0x04	0x45	8
0x04	0x6a	7
0x04	0x63	6
0x04	0x6d	5
0x04	0x67	4
0x05	0x42	8
0x05	0x56	7
0x05	0x61	6
0x05	0x65	5
0x05	0x73	4
0x06	0x4b	8
0x07	0x47	8
0x07	0x6b	6
0x07	0x6f	5
0x07	0x66	4
0x0a	0x21	4
0x0c	0x81	8
0x0c	0x60	5
0x0d	0x78	8
0x0e	0xe1	8
0x0e	0x1f	4
0x0f	0x92	8
0x10	0x71	5
0x11	0x64	5
0x12	0x62	5
0x13	0x74	5
0x16	0x70	5
0x17	0x68	5
0x18	0x0b	8
0x18	0x4d	7
0x19	0x09	8
0x19	0x49	7
0x19	0x69	5
0x1a	0x2e	8
0x1a	0x51	7
0x1a	0x7a	6
0x1b	0x23	8
0x1b	0x50	7
0x1b	0x6c	6
0x1b	0x75	5
0x1e	0x6e	5
0x34	0x76	6
0x35	0x72	6
0x3e	0x2d	6
0x60	0x54	7
0x61	0x2c	7
0x62	0x2b	7

0x7e	0x5a	7
0x7f	0x20	7
0xc6	0x2f	8
0xc7	0x0a	8

X.3.1.3 Uncompressed received management message payload

Using an intermediate buffer of 726 bytes for uncompressing, it should be as follows:

Offset	0x0	0x1	0x2	0x3	0x4	0x5	0x6	0x7	--	0x8	0x9	0xa	0xb	0xc	0xd	0xe	0xf
0x000	0x56	0x69	0x63	0x74	0x6f	0x72	0x20	0x5a		0x65	0x69	0x62	0x20	0x42	0x6f	0x72	0x74
0x010	0x0a	0x4d	0x61	0x72	0x63	0x6f	0x73	0x20		0x4d	0x61	0x72	0x74	0x69	0x6e	0x65	0x7a
0x020	0x20	0x56	0x61	0x7a	0x71	0x75	0x65	0x7a		0x0a	0x21	0x5a	0x6e	0x68	0x2c	0x54	0x6e
0x030	0x75	0x67	0x70	0x73	0x69	0x1f	0x6c	0x6d		0x70	0x76	0x74	0x1f	0x75	0x67	0x66	0x1f
0x040	0x68	0x60	0x75	0x64	0x2f	0x1f	0x5a	0x6e		0x68	0x2c	0x54	0x6e	0x75	0x67	0x70	0x73
0x050	0x69	0x1f	0x6a	0x72	0x21	0x73	0x69	0x64		0x21	0x66	0x62	0x73	0x66	0x2d	0x21	0x58
0x060	0x70	0x66	0x2e	0x52	0x70	0x73	0x69	0x6e		0x75	0x67	0x21	0x68	0x74	0x1f	0x75	0x67
0x070	0x66	0x1f	0x6c	0x64	0x7a	0x1f	0x62	0x6d		0x65	0x1f	0x68	0x74	0x62	0x71	0x65	0x68
0x080	0x62	0x6d	0x21	0x6e	0x67	0x1f	0x75	0x67		0x66	0x1f	0x68	0x60	0x75	0x64	0x2f	0x1f
0x090	0x51	0x60	0x74	0x73	0x2d	0x1f	0x71	0x71		0x66	0x72	0x66	0x6d	0x75	0x2b	0x21	0x65
0x0a0	0x76	0x73	0x76	0x71	0x66	0x2b	0x21	0x60		0x6d	0x6b	0x21	0x60	0x73	0x64	0x21	0x6e
0x0b0	0x6f	0x64	0x21	0x68	0x6f	0x1f	0x5a	0x6e		0x68	0x2c	0x54	0x6e	0x75	0x67	0x70	0x73
0x0c0	0x69	0x2d	0x21	0x47	0x66	0x1f	0x6c	0x6d		0x70	0x76	0x74	0x1f	0x78	0x67	0x66	0x71
0x0d0	0x66	0x1f	0x75	0x67	0x66	0x1f	0x50	0x6b		0x65	0x1f	0x50	0x6d	0x66	0x72	0x21	0x61
0x0e0	0x73	0x6e	0x6c	0x64	0x21	0x73	0x69	0x71		0x70	0x74	0x68	0x67	0x21	0x6e	0x67	0x1f
0x0f0	0x70	0x6b	0x65	0x2b	0x21	0x60	0x6f	0x63		0x21	0x76	0x69	0x64	0x73	0x64	0x21	0x53
0x100	0x69	0x64	0x7a	0x1f	0x74	0x67	0x62	0x6b		0x6d	0x1f	0x63	0x71	0x66	0x60	0x6c	0x1f
0x110	0x75	0x67	0x73	0x6e	0x76	0x66	0x69	0x1f		0x62	0x66	0x62	0x68	0x6f	0x2d	0x23	0x09
0x120	0x21	0x1f	0x21	0xe1	0x81	0x92	0x49	0x2d		0x51	0x2d	0x21	0x4b	0x70	0x75	0x66	0x62
0x130	0x73	0x60	0x67	0x73	0x2d	0x1f	0x55	0x67		0x66	0x1f	0x45	0x74	0x6f	0x76	0x6a	0x62
0x140	0x69	0x1f	0x49	0x6e	0x73	0x71	0x70	0x71		0x0b	0x53	0x53	0x53	0x53	0x53	0x53	0x53
0x150	0x53	0x53	0x53	0x53	0x53	0x53	0x53	0x53		0x53	0x53	0x53	0x53	0x53	0x53	0x53	0x53
0x160	0x53	0x53	0x53	0x53	0x53	0x53	0x53	0x53		0x53	0x53	0x53	0x53	0x53	0x53	0x53	0x53
0x170	0x53	0x53	0x53	0x53	0x53	0x53	0x53	0x53		0x53	0x53	0x53	0x53	0x53	0x53	0x53	0x53
0x180	0x53	0x53	0x53	0x53	0x53	0x53	0x53	0x53		0x53	0x53	0x53	0x53	0x53	0x53	0x53	0x53
0x190	0x53	0x53	0x53	0x53	0x53	0x53	0x53	0x53		0x53	0x53	0x53	0x53	0x53	0x53	0x53	0x53
0x1a0	0x53	0x53	0x53	0x53	0x53	0x53	0x53	0x53		0x53	0x53	0x53	0x53	0x53	0x53	0x53	0x53
0x1b0	0x53	0x53	0x53	0x53	0x53	0x53	0x53	0x53		0x53	0x53	0x53	0x53	0x53	0x53	0x53	0x53
0x1c0	0x53	0x53	0x53	0x53	0x53	0x53	0x53	0x53		0x53	0x53	0x53	0x53	0x53	0x53	0x53	0x53
0x1d0	0x53	0x53	0x53	0x53	0x53	0x53	0x53	0x53		0x53	0x53	0x53	0x53	0x53	0x53	0x53	0x53
0x1e0	0x53	0x53	0x53	0x53	0x53	0x53	0x53	0x53		0x53	0x53	0x53	0x53	0x53	0x53	0x53	0x53
0x1f0	0x53	0x53	0x53	0x53	0x53	0x53	0x53	0x53		0x53	0x53	0x53	0x53	0x53	0x53	0x53	0x53
0x200	0x53	0x53	0x53	0x53	0x53	0x53	0x53	0x53		0x53	0x53	0x53	0x53	0x53	0x53	0x53	0x53
0x210	0x53	0x53	0x53	0x53	0x53	0x53	0x53	0x53		0x53	0x53	0x53	0x53	0x53	0x53	0x53	0x53
0x220	0x53	0x53	0x53	0x53	0x53	0x53	0x53	0x53		0x53	0x53	0x53	0x53	0x53	0x53	0x53	0x53
0x230	0x53	0x53	0x53	0x53	0x53	0x53	0x53	0x53		0x53	0x53	0x53	0x53	0x53	0x53	0x53	0x53
0x240	0x53	0x53	0x53	0x53	0x53	0x53	0x53	0x53		0x53	0x53	0x53	0x53	0x53	0x53	0x53	0x53
0x250	0x53	0x53	0x53	0x53	0x53	0x53	0x53	0x53		0x53	0x53	0x53	0x53	0x53	0x53	0x53	0x53
0x260	0x53	0x53	0x53	0x53	0x53	0x53	0x53	0x53		0x53	0x53	0x53	0x53	0x53	0x53	0x53	0x53
0x270	0x53	0x53	0x53	0x53	0x53	0x53	0x53	0x53		0x53	0x53	0x53	0x53	0x53	0x53	0x53	0x53

0x280	0x53	0x53	0x53	0x53	0x53	0x53	0x53	0x53		0x53	0x53	0x53	0x53	0x53	0x53	0x53	0x53
0x290	0x53	0x53	0x53	0x53	0x53	0x53	0x53	0x53		0x53	0x53	0x53	0x53	0x53	0x53	0x53	0x53
0x2a0	0x53	0x53	0x53	0x53	0x53	0x53	0x53	0x53		0x53	0x53	0x53	0x53	0x53	0x53	0x53	0x53
0x2b0	0x53	0x53	0x53	0x53	0x53	0x53	0x53	0x53		0x53	0x53	0x53	0x53	0x53	0x53	0x53	0x53
0x2c0	0x53	0x53	0x53	0x53	0x53	0x53	0x53	0x53		0x53	0x53	0x53	0x53	0x53	0x53	0x53	0x53
0x2d0	0x53	0x53	0x53	0x53	0x53	0x53											

X.3.1.4 Output (uncompressed) management message payload (MMPL)

MMPL (329 bytes)

Offset	**0x0**	**0x1**	**0x2**	**0x3**	**0x4**	**0x5**	**0x6**	**0x7**	**--**	**0x8**	**0x9**	**0xa**	**0xb**	**0xc**	**0xd**	**0xe**	**0xf**
0x000	0x56	0x69	0x63	0x74	0x6f	0x72	0x20	0x5a		0x65	0x69	0x62	0x20	0x42	0x6f	0x72	0x74
0x010	0x0a	0x4d	0x61	0x72	0x63	0x6f	0x73	0x20		0x4d	0x61	0x72	0x74	0x69	0x6e	0x65	0x7a
0x020	0x20	0x56	0x61	0x7a	0x71	0x75	0x65	0x7a		0x0a	0x21	0x5a	0x6e	0x68	0x2c	0x54	0x6e
0x030	0x75	0x67	0x70	0x73	0x69	0x1f	0x6c	0x6d		0x70	0x76	0x74	0x1f	0x75	0x67	0x66	0x1f
0x040	0x68	0x60	0x75	0x64	0x2f	0x1f	0x5a	0x6e		0x68	0x2c	0x54	0x6e	0x75	0x67	0x70	0x73
0x050	0x69	0x1f	0x6a	0x72	0x21	0x73	0x69	0x64		0x21	0x66	0x62	0x73	0x66	0x2d	0x21	0x58
0x060	0x70	0x66	0x2e	0x52	0x70	0x73	0x69	0x6e		0x75	0x67	0x21	0x68	0x74	0x1f	0x75	0x67
0x070	0x66	0x1f	0x6c	0x64	0x7a	0x1f	0x62	0x6d		0x65	0x1f	0x68	0x74	0x62	0x71	0x65	0x68
0x080	0x62	0x6d	0x21	0x6e	0x67	0x1f	0x75	0x67		0x66	0x1f	0x68	0x60	0x75	0x64	0x2f	0x1f
0x090	0x51	0x60	0x74	0x73	0x2d	0x1f	0x71	0x71		0x66	0x72	0x66	0x6d	0x75	0x2b	0x21	0x65
0x0a0	0x76	0x73	0x76	0x71	0x66	0x2b	0x21	0x60		0x6d	0x6b	0x21	0x60	0x73	0x64	0x21	0x6e
0x0b0	0x6f	0x64	0x21	0x68	0x6f	0x1f	0x5a	0x6e		0x68	0x2c	0x54	0x6e	0x75	0x67	0x70	0x73
0x0c0	0x69	0x2d	0x21	0x47	0x66	0x1f	0x6c	0x6d		0x70	0x76	0x74	0x1f	0x78	0x67	0x66	0x71
0x0d0	0x66	0x1f	0x75	0x67	0x66	0x1f	0x50	0x6b		0x65	0x1f	0x50	0x6d	0x66	0x72	0x21	0x61
0x0e0	0x73	0x6e	0x6c	0x64	0x21	0x73	0x69	0x71		0x70	0x74	0x68	0x67	0x21	0x6e	0x67	0x1f
0x0f0	0x70	0x6b	0x65	0x2b	0x21	0x60	0x6f	0x63		0x21	0x76	0x69	0x64	0x73	0x64	0x21	0x53
0x100	0x69	0x64	0x7a	0x1f	0x74	0x67	0x62	0x6b		0x6d	0x1f	0x63	0x71	0x66	0x60	0x6c	0x1f
0x110	0x75	0x67	0x73	0x6e	0x76	0x66	0x69	0x1f		0x62	0x66	0x62	0x68	0x6f	0x2d	0x23	0x09
0x120	0x21	0x1f	0x21	0xe1	0x81	0x92	0x49	0x2d		0x51	0x2d	0x21	0x4b	0x70	0x75	0x66	0x62
0x130	0x73	0x60	0x67	0x73	0x2d	0x1f	0x55	0x67		0x66	0x1f	0x45	0x74	0x6f	0x76	0x6a	0x62
0x140	0x69	0x1f	0x49	0x6e	0x73	0x71	0x70	0x71		0x0b							

10. Add text to the Bibliography as follows:

[b-BCL] Basic Compression Library
Available at http://bcl.comli.eu/home-en.html