7. Security suite

7.1 Overview

The MAC sublayer is responsible for providing security services on specified incoming and outgoing frames when requested to do so by the higher layers. This standard supports the following security services:

- Data confidentiality
- Data authenticity
- Replay protection

7.2 Functional description

A device may optionally implement security. A device that does not implement security shall not provide a mechanism for the MAC sublayer to perform any cryptographic transformation on incoming and outgoing frames nor require any PIB attributes associated with security. A device that implements security shall provide a mechanism for the MAC sublayer to provide cryptographic transformations on incoming and outgoing frames using information in the PIB attributes associated with security when the *macSecurityEnabled* attribute is set to TRUE.

If the MAC sublayer is required to transmit a frame or receives an incoming frame, the MAC sublayer shall process the frame as specified in 7.2.1 and 7.2.4, respectively.

7.2.1 Outgoing frame security procedure

The inputs to this procedure are the frame to be secured and the SecurityLevel, KeyIdMode, KeySource, and KeyIndex parameters from the originating primitive or automatic request PIB attributes. The outputs from this procedure are the status of the procedure and, if this status is SUCCESS, the secured frame.

The outgoing frame security procedure involves the following steps as applicable:

a) If the Security Enabled field of the Frame Control field of the frame to be secured is set to zero, the procedure shall set the security level to zero.

b) If the Security Enabled field of the Frame Control field of the frame to be secured is set to one, the procedure shall set the security level to the SecurityLevel parameter. If the resulting security level is zero, the procedure shall return with a status of UNSUPPORTED_SECURITY.

c) If the *macSecurityEnabled* attribute is set to FALSE and the security level is not equal to zero, the procedure shall return with a status of UNSUPPORTED_SECURITY.

d) The procedure shall determine whether the frame to be secured satisfies the constraint on the maximum length of MAC frames, as follows:

   1) The procedure shall set the length *M*, in octets, of the Authentication field to zero if the security level is equal to zero and shall determine this value from the security level and Table 56 otherwise.

   2) The procedure shall determine the length AuxLen, in octets, of the auxiliary security header, as described in 7.4, using KeyIdMode and the security level.

   3) The procedure shall determine the data expansion as AuxLen + *M*.

   4) The procedure shall check whether the length of the frame to be secured, including data expansion and FCS, is less than or equal to *aMaxPHYPacketSize*. If this check fails, the procedure shall return with a status of FRAME_TOO_LONG.

e) If the security level is zero, the procedure shall set the secured frame to be the frame to be secured and return with a status of SUCCESS.

f) The procedure shall set the frame counter to the *macFrameCounter* attribute. If the frame counter has the value 0xffffffff, the procedure shall return with a status of COUNTER_ERROR.

g) The procedure shall obtain the KeyDescriptor using the outgoing frame key retrieval procedure as described in 7.2.2. If that procedure fails, the procedure shall return with a status of UNAVAILABLE_KEY.

h) If the Blacklisted element of the KeyDescriptor is set to TRUE, the procedure shall return with a status of KEY_ERROR.

i) The procedure shall insert the auxiliary security header into the frame, with fields set as follows:

   1) The Security Level field of the Security Control field shall be set to the security level.

   2) The Key Identifier Mode field of the Security Control field shall be set to the KeyIdMode parameter.

   3) The Frame Counter field shall be set to the frame counter.

   4) If the KeyIdMode parameter is set to a value not equal to zero, the Key Source and Key Index fields of the Key Identifier field shall be set to the KeySource and KeyIndex parameters, respectively.

j) The procedure shall then use *macExtendedAddress*, the frame counter, the security level, and the Key element of the KeyDescriptor to produce the secured frame according to the CCM* transformation process defined in 7.3.4.

   1) If the SecurityLevel parameter specifies the use of encryption, as defined in Table 56, the encryption operation shall be applied only to the actual payload field within the MAC payload, i.e., the Beacon Payload field as described in 5.2.2.1.8, Command Payload field, as described in 5.2.2.4.3, or Data Payload field, as described in 5.2.2.2.2, depending on the frame type. The corresponding payload field is passed to the CCM* transformation process described in 7.3.4 as the unsecured payload, as defined in Table 53. The resulting encrypted payload shall substitute the original payload.

   2) The remaining fields in the MAC payload part of the frame shall be passed to the CCM* transformation process described in 7.3.4 as the nonpayload fields, as defined in Table 53.

   3) The ordering and exact manner of performing the encryption and integrity operations and the placement of the resulting encrypted data or integrity code within the MAC Payload field shall be as defined in 7.3.4.

k) The procedure shall increment the frame counter by one and set the *macFrameCounter* attribute to the resulting value.

l) If the *macFrameCounter* element is equal to 0xffffffff, the procedure shall set the Blacklisted element of the KeyDescriptor to TRUE.

m) The procedure shall return with the secured frame and a status of SUCCESS.

7.2.2 Outgoing frame key retrieval procedure

The inputs to this procedure are the frame to be secured and the KeyIdMode, KeySource, and KeyIndex parameters from the originating primitive. The outputs from this procedure are a passed or failed status and, if passed, a KeyDesciptor.

The outgoing frame key retrieval procedure involves the following steps as applicable:

a) If the KeyIdMode parameter is set to 0x00 (implicit key identification), the procedure shall determine the key lookup data and key lookup size as follows:

   1) If the Destination Addressing Mode field of the Frame Control field of the frame is set to 0x00 and the *macPANCoordShortAddress* attribute is set to a value in the range 0x0000–0xfffd (i.e.,

the short address is used), the key lookup data shall be set to the Source PAN Identifier field of the frame right-concatenated, as defined in B.2.1, with the *macPANCoordShortAddress* attribute. The key lookup size shall be set to four.

2) If the Destination Addressing Mode field of the Frame Control field of the frame is set to 0x00 and the *macPANCoordShortAddress* attribute is set to 0xfffe (i.e., the extended address is used), the key lookup data shall be set to the *macPANCoordExtendedAddress* attribute. The key lookup size shall be set to eight.

3) If the Destination Addressing Mode field of the Frame Control field of the frame is set to 0x02, the key lookup data shall be set to the Destination PAN Identifier field of the frame right-concatenated, as defined in B.2.1, with the Destination Address field of the frame. The key lookup size shall be set to four.

4) If the Destination Addressing Mode field of the Frame Control field of the frame is set to 0x03, the key lookup data shall be set to the Destination Address field of the frame. The key lookup size shall be set to eight.

The key index shall be set to the single octet 0x00.

b) If the KeyIdMode parameter is set to a value not equal to 0x00 (explicit key identification), the procedure shall determine the key lookup data and key lookup size as follows:

1) If the KeyIdMode parameter is set to 0x01, the key lookup data shall be set to the *macDefaultKeySource* attribute. The key lookup size shall be set to eight.

2) If the KeyIdMode parameter is set to 0x02, the key lookup data shall be set to the KeySource parameter. The key lookup size shall be set to four.

3) If the KeyIdMode parameter is set to 0x03, the key lookup data shall be set to the KeySource parameter. The key lookup size shall be set to eight.

The key index shall be set to the KeyIndex parameter.

c) The procedure shall obtain the KeySourceDescriptor by passing the key lookup data and the key lookup size to the KeySourceDescriptor lookup procedure as described in 7.2.11. If that procedure returns with a failed status, this procedure shall also return with a failed status.

d) The procedure shall obtain the KeyDescriptor by passing the KeySourceDescriptor and the key index to the KeyDescriptor lookup procedure as described in 7.2.3. If that procedure returns with a failed status, this procedure shall also return with a failed status.

e) The procedure shall return with a passed status, having obtained the KeyDescriptor.

NOTE—For broadcast frames, the outgoing frame key retrieval procedure will result in a failed status if implicit key identification is used. Hence, explicit key identification should be used for broadcast frames.[6]

7.2.3 KeyDescriptor lookup procedure

The inputs to this procedure are the KeySourceDescriptor and the key index. The outputs from this procedure are a passed or failed status and, if passed, a KeyDescriptor.

The KeyDescriptor lookup procedure involves the following steps as applicable:

a) For each KeyDescriptor in the *macKeyTable* attribute, the procedure shall check whether the ExtKeySource element of the KeyDescriptor is equal to the corresponding element of the KeySourceDescriptor and whether the KeyIndex element of the KeyDescriptor is equal to the key index param-

[6] Notes in text, tables, and figures are given for information only and do not contain requirements needed to implement the standard.

eter. If both checks pass (i.e., there is a match), the procedure shall return with this (matching) KeyDescriptor and a passed status.

b) The procedure shall return with a failed status.

7.2.4 Incoming frame security procedure

The input to this procedure is the frame to be unsecured. The outputs from this procedure are the unsecured frame, the security level, the key identifier mode, the key source, the key index, and the status of the procedure.

All outputs of this procedure are assumed to be invalid unless and until explicitly set in this procedure.

It is assumed that the PIB attributes associating KeyDescriptors in *macKeyTable* with a single, unique device or a number of devices will have been established by the next higher layer.

The incoming frame security procedure involves the following steps:

a) If the Security Enabled field of the Frame Control field of the frame to be unsecured is set to zero, the procedure shall set the security level to zero.

b) If the Security Enabled field of the Frame Control field of the frame to be unsecured is set to one and the frame version number of the Frame Control field of the frame to be unsecured is set to zero, the procedure shall set the unsecured frame to be the frame to be unsecured and return with a status of UNSUPPORTED_LEGACY.

c) If the Security Enabled field of the Frame Control field of the frame to be unsecured is set to one, the procedure shall set the security level and the key identifier mode to the corresponding fields of the Security Control field of the auxiliary security header of the frame to be unsecured, and the key source and key index to the corresponding fields of the Key Identifier field of the auxiliary security header of the frame to be unsecured, if present. If the resulting security level is zero, the procedure shall set the unsecured frame to be the frame to be unsecured and return with a status of UNSUPPORTED_SECURITY.

d) If the *macSecurityEnabled* attribute is set to FALSE, the procedure shall set the unsecured frame to be the frame to be unsecured and return with a status of SUCCESS if the security level is equal to zero and with a status of UNSUPPORTED_SECURITY otherwise.

e) The procedure shall obtain the SecurityLevelDescriptor by passing the frame type and, depending on whether the frame is a MAC command frame, the first octet of the MAC payload (i.e., command frame identifier for a MAC command frame) to the SecurityLevelDescriptor lookup procedure described in 7.2.5. If that procedure fails, the procedure shall set the unsecured frame to be the frame to be unsecured and return with a status of UNAVAILABLE_SECURITY_LEVEL.

f) The procedure shall obtain the KeyDescriptor, DeviceDescriptor and KeyDeviceDescriptor using the incoming frame security material retrieval procedure described in 7.2.6. If that procedure fails because the KeyDescriptor lookup procedure failed, the procedure shall set the unsecured frame to be the frame to be unsecured and return with a status of UNAVAILABLE_KEY. If the incomfing frame security material retrieval procedure failed because the DeviceDescriptor lookup failed, then the procedure shall set the unsecured frame to be the frame to be unsecured and return with a status of UNAVAILABLE_DEVICE

g) The procedure shall determine whether the frame to be unsecured conforms to the security level policy by passing the SecurityLevelDescriptor and the security level to the incoming security level checking procedure, as described in 7.2.7. If that procedure returns with a failed status, the procedure shall set the unsecured frame to be the frame to be unsecured and return with a status of IMPROPER_SECURITY_LEVEL; otherwise, if that procedure returns with a passed status and the security level is equal to zero, the procedure shall set the unsecured frame to be the frame to be unsecured and return with a status of SUCCESS.

h) If the incoming security level checking procedure of Step g had as output the ‘conditionally passed’ status, the procedure shall set the unsecured frame to be the frame to be unsecured and return with the unsecured frame, the security level, the key identifier mode, the key source, the key index, and a status of SUCCESS, if the Exempt element of the DeviceDescriptor is set to TRUE, and with a status of IMPROPER_SECURITY_LEVEL otherwise.

i) The procedure shall set the frame counter to the Frame Counter field of the auxiliary security header of the frame to be unsecured. If the frame counter has the value 0xffffffff, the procedure shall set the unsecured frame to be the frame to be unsecured and return with a status of COUNTER_ERROR.

j) The procedure shall determine whether the frame counter is greater than or equal to the FrameCounter element of the DeviceDescriptor. If this check fails, the procedure shall set the unsecured frame to be the frame to be unsecured and return with a status of COUNTER_ERROR.

k) The procedure shall determine whether the frame to be unsecured conforms to the key usage policy by passing the KeyDescriptor, the frame type, and, depending on whether the frame is a MAC command frame, the first octet of the MAC payload (i.e., command frame identifier for a MAC command frame) to the incoming key usage policy checking procedure, as described in 7.2.8. If that procedure fails, the procedure shall set the unsecured frame to be the frame to be unsecured and return with a status of IMPROPER_KEY_TYPE.

l) The procedure shall then use the ExtAddress element of the DeviceDescriptor, the frame counter, the security level, and the Key element of the KeyDescriptor to produce the unsecured frame, according to the CCM* inverse transformation process described in the security operations, as described in 7.3.5.

   1) If the security level specifies the use of encryption, as described in Table 57, the decryption operation shall be applied only to the actual payload field within the MAC payload, i.e., the Beacon Payload field, as defined in 5.2.2.1.8, Command Payload field, as defined in 5.2.2.4.3, or Data Payload field, as defined in 5.2.2.2.2, depending on the frame type. The corresponding payload field shall be passed to the CCM* inverse transformation process described in 7.3.5 as the secure payload.

   2) The remaining fields in the MAC payload part of the frame shall be passed to the CCM* inverse transformation process described in 7.3.5 as the non-payload fields.

   3) The ordering and exact manner of performing the decryption and integrity checking operations and the placement of the resulting decrypted data within the MAC payload field shall be as defined in 7.3.5.

m) If the CCM* inverse transformation process fails, the procedure shall set the unsecured frame to be the frame to be unsecured and return with a status of SECURITY_ERROR.

n) The procedure shall increment the frame counter by one and set the FrameCounter element of the DeviceDescriptor to the resulting value.

o) If the FrameCounter element is equal to 0xffffffff, the procedure shall set the Blacklisted element of the KeyDeviceDescriptor to TRUE.

p) The procedure shall return with the unsecured frame and a status of SUCCESS.

7.2.5 SecurityLevelDescriptor lookup procedure

The inputs to this procedure are the frame type and the command frame identifier. The output from this procedure are a passed or failed status and, if passed, a SecurityLevelDescriptor.

The SecurityLevelDescriptor lookup procedure involves the following steps:

a) For each SecurityLevelDescriptor in the macSecurityLevelTable attribute:

   1) If the frame type is not equal to 0x03 and the frame type is equal to the FrameType element of the SecurityLevelDescriptor (i.e., there is a match), the procedure shall return with the SecurityLevelDescriptor and a passed status.

2) If the frame type is equal to 0x03, the frame type is equal to the FrameType element of the SecurityLevelDescriptor and the command frame identifier is equal to the Command-FrameIdentifier element of the SecurityLevelDescriptor, the procedure shall return with the SecurityLevelDescriptor and a passed status.

b) The procedure shall return with a failed status.

7.2.6 Incoming frame security material retrieval procedure

The input to this procedure is the frame to be unsecured. The outputs from this procedure are a passed or failed status and, if passed, a KeyDescriptor, a DeviceDescriptor, and a KeyDeviceDescriptor.

The incoming frame security material retrieval procedure involves the following steps as applicable:

a) If the Key Identifier Mode field of the Security Control field of the auxiliary security header of the frame is set to 0x00 (implicit key identification), the procedure shall determine the key lookup data and the key lookup size as follows:

   1) If the source address mode of the Frame Control field of the frame is set to 0x00 and the *macPANCoordShortAddress* attribute is set to a value in the range 0x0000–0xfffd (i.e., the short address is used), the key lookup data shall be set to the 2-octet Destination PAN Identifier field of the frame right-concatenated (see C.2.1) with the 2-octet *macPANCoordShortAddress* attribute right-concatenated with the single octet 0x00. The key lookup size shall be set to five.

   2) If the source address mode of the Frame Control field of the frame is set to 0x00 and the *macPANCoordShortAddress* attribute is set to 0xfffe (i.e., the extended address is used), the key lookup data shall be set to the 8-octet *macPANCoordExtendedAddress* attribute right-concatenated (see C.2.1) with the single octet 0x00. The key lookup size shall be set to nine.

   3) If the source address mode of the Frame Control field of the frame is set to 0x02, the key lookup data shall be set to the 2-octet Source PAN Identifier field of the frame, or to the 2-octet Destination PAN Identifier field of the frame if the PAN ID Compression field of the Frame Control field of the frame is set to one, right-concatenated (see C.2.1) with the 2-octet Source Address field of the frame right-concatenated with the single octet 0x00. The key lookup size shall be set to five.

   4) If the source address mode of the Frame Control field of the frame is set to 0x03, the key lookup data shall be set to the 8-octet Source Address field of the frame right-concatenated (see C.2.1) with the single octet 0x00. The key lookup size shall be set to nine.

b) If the Key Identifier Mode field of the Security Control field of the auxiliary security header of the frame is set to a value not equal to 0x00 (explicit key identification), the procedure shall determine the key lookup data and key lookup size as follows:

   1) If the key identifier mode is set to 0x01, the key lookup data shall be set to the 8-octet *macDefaultKeySource* attribute right-concatenated (see C.2.1) with the 1-octet Key Index field of the Key Identifier field of the auxiliary security header. The key lookup size shall be set to nine.

   2) If the key identifier mode is set to 0x02, the key lookup data shall be set to the right-concatenation (see C.2.1) of the 4-octet Key Source field and the 1-octet Key Index field of the Key Identifier field of the auxiliary security header. The key lookup size shall be set to five.

   3) If the key identifier mode is set to 0x03, the key lookup data shall be set to the right-concatenation (see C.2.1) of the 8-octet Key Source field and the 1-octet Key Index field of the Key Identifier field of the auxiliary security header. The key lookup size shall be set to nine.

c) The procedure shall obtain the KeyDescriptor by passing the key lookup data and the key lookup size to the KeyDescriptor lookup procedure as described in 7.2.3. If that procedure returns with a failed status, the procedure shall also return with a failed status.

d) The procedure shall determine the device lookup data and the device lookup size as follows:

1) If the source address mode of the Frame Control field of the frame is set to 0x00 and the *macPANCoordShortAddress* attribute is set to a value in the range 0x0000–0xfffd (i.e., the short address is used), the device lookup data shall be set to the 2-octet Destination PAN Identifier field of the frame right-concatenated, as described in B.2.1, with the 2-octet *macPANCoordShortAddress* attribute. The device lookup size shall be set to four.
2) If the source address mode of the Frame Control field of the frame is set to 0x00 and the *macPANCoordShortAddress* attribute is set to 0xfffe (i.e., the extended address is used), the device lookup data shall be set to the 8-octet *macPANCoordExtendedAddress* attribute. The device lookup size shall be set to eight.
3) If the source address mode of the Frame Control field of the frame is set to 0x02, the device lookup data shall be set to the 2-octet Source PAN Identifier field of the frame, or to the 2-octet Destination PAN Identifier field of the frame if the PAN ID Compression field of the Frame Control field of the frame is set to one, right-concatenated, as defined in B.2.1, with the 2-octet Source Address field of the frame. The device lookup size shall be set to four.
4) If the source address mode of the Frame Control field of the frame is set to 0x03, the device lookup data shall be set to the 8-octet Source Address field of the frame. The device lookup size shall be set to eight.

e) The procedure shall obtain the DeviceDescriptor and the KeyDeviceDescriptor by passing the KeyDescriptor, the device lookup data, and the device lookup size to the blacklist checking procedure as described in 7.2.9. If that procedure returns with a failed status, the procedure shall also return with a failed status.
f) The procedure shall return with a passed status having obtained the KeyDescriptor, the DeviceDescriptor, and the KeyDeviceDescriptor.

7.2.7 Incoming security level checking procedure

The inputs to this procedure are the SecurityLevelDescriptor and the incoming security level. The output from this procedure is a passed, failed, or 'conditionally passed' status.

The incoming security level checking procedure involves the following steps:

a) If the AllowedSecurityLevels is empty, then the procedure shall compare the incoming security level (as SEC1) with the SecurityMinimum element of the SecurityLevelDescriptor (as SEC2) according to the algorithm described in 7.4.1.1. If this comparison evaluates to TRUE, the procedure shall return with a passed status.
b) If the AllowedSecurityLevels is not empty, the procedure shall check whether the incoming security level is equal to any of the elements of the AllowedSecurityLevels. If this check is successful (i.e., there is a match), the procedure shall return with a passed status.
c) If the incoming security level is equal to 0x00 and the DeviceOverrideSecurityMinimum element of the SecurityLevelDescriptor is set to TRUE, the procedure shall return with a 'conditionally passed' status.
d) The procedure shall return with a failed status.

7.2.8 Incoming key usage policy checking procedure

The inputs to this procedure are the KeyDescriptor, the frame type, and the command frame identifier. The output from this procedure is a passed or failed status.

The incoming key usage policy checking procedure involves the following steps as applicable:

a) For each KeyUsageDescriptor in the KeyUsageList of the KeyDescriptor:

1) If the frame type is not equal to 0x03 and the frame type is equal to the FrameType element of the KeyUsageDescriptor, the procedure shall return with a passed status.
2) If the frame type is equal to 0x03, the frame type is equal to the FrameType element of the KeyUsageDescriptor, and the command frame identifier is equal to the CommandFrame-Identifier element of the KeyUsageDescriptor, the procedure shall return with a passed status.

b) The procedure shall return with a failed status.

7.2.9 Blacklist checking procedure

The inputs to this procedure are the KeyDescriptor, the device lookup data, and the device lookup size. The outputs from this procedure are a passed or failed status and, if passed, a DeviceDescriptor and a KeyDeviceDescriptor.

The blacklist checking procedure involves the following steps as applicable:

a) For each KeyDeviceDescriptor in the KeyDeviceList of the KeyDescriptor:
   1) The procedure shall obtain the DeviceDescriptor using the DeviceDescriptorHandle element of the KeyDeviceDescriptor.
   2) If the UniqueDevice element of the KeyDeviceDescriptor is set to TRUE, the procedure shall return with the DeviceDescriptor, the KeyDeviceDescriptor, and a passed status if the BlackListed element of the KeyDeviceDescriptor is set to FALSE, or the procedure shall return with a failed status if this Blacklisted element is set to TRUE.
   3) If the UniqueDevice element of the KeyDeviceDescriptor is set to FALSE, the procedure shall execute the DeviceDescriptor lookup procedure as described in 7.2.10, with the device lookup data and the device lookup size as inputs. If the corresponding output of that procedure is a passed status, the procedure shall return with the DeviceDescriptor, the KeyDevice-Descriptor, and a passed status if the Blacklisted element of the KeyDeviceDescriptor is set to FALSE, or the procedure shall return with a failed status if this Blacklisted element is set to TRUE.

b) The procedure shall return with a failed status.

7.2.10 DeviceDescriptor lookup procedure

The inputs to this procedure are the DeviceDescriptor, the device lookup data, and the device lookup size. The output from this procedure is a passed or failed status.

The DeviceDescriptor lookup procedure involves the following steps as applicable:

a) If the device lookup size is four and the device lookup data is equal to the PAN ID element of the DeviceDescriptor right-concatenated (see C.2.1) with the ShortAddress element of the DeviceDescriptor, this procedure shall return with a passed status.
b) If the device lookup size is eight and the device lookup data is equal to the ExtAddress element of the DeviceDescriptor, this procedure shall return with a passed status.
c) The procedure shall return with a failed status.

7.2.11 KeySourceDescriptor lookup procedure

The inputs to this procedure are the key source lookup data and the key source lookup size. The output from this procedure are a passed or failed status and, if passed, a KeySourceDescriptor.

The KeySourceDescriptor lookup procedure involves the following steps:

a) For each KeySourceDescriptor in the *macKeySourceTable* attribute:

   i) If the key source lookup size is four and the key source lookup data is equal to the ShortKeySource element of the KeySourceDescriptor (i.e., there is a match), this procedure shall return with a passed status.

   ii) If the key source lookup size is eight and the key source lookup data is equal to the ExtKeySource element of the KeySourceDescriptor (i.e., there is a match), this procedure shall return with a passed status.

b) The procedure shall return with a failed status.

7.3 Security operations

This subclause describes the parameters for the CCM* security operations, as specified in B.3.2.

7.3.1 Integer and octet representation

The integer and octet representation conventions specified in B.2 are used throughout 7.3.

7.3.2 CCM* Nonce

The CCM* nonce is a 13-octet string and is used for the advanced encryption standard (AES)-CCM* mode of operation, as described in B.2.2. The nonce shall be formatted as shown in Figure 68, with the leftmost field in the figure defining the first (and leftmost) octets and the rightmost field defining the last (and rightmost) octet of the nonce.

Octets: 8	**4**	**1**
Source address	Frame counter	Security level

Figure 68—CCM* nonce

The source address shall be set to the extended address *macExtendedAddress* of the device originating the frame, the frame counter to the value of the respective field in the auxiliary security header (see 7.7.2), and the security level to the security level identifier corresponding to the Security Level field of the Security Control field of the auxiliary security header as defined in Table 56.

The source address, frame counter, and security level shall be represented as specified in 7.3.1.

7.3.3 CCM* prerequisites

Securing a frame involves the use of the CCM* mode encryption and authentication transformation, as described in B.4.1. Unsecuring a frame involves the use of the CCM* decryption and authentication checking process, as described in B.4.2. The prerequisites for the CCM* forward and inverse transformations are as follows:

- The underlying block cipher shall be the AES encryption algorithm as specified in B.3.1.
- The bit ordering shall be as defined in 7.3.1.
- The length in octets of the Length field *L* shall be 2 octets.
- The length of the Authentication field *M* shall be 0 octets, 4 octets, 8 octets, or 16 octets, as required.

The length of the Authentication field *M* for the CCM* forward transformation and the CCM* inverse transformation is determined from Table 56, using the Security Level field of the Security Control field of the auxiliary security header of the frame.

7.3.4 CCM* transformation data representation

This subclause describes how the inputs and output of the CCM* forward transformation, as described in B.4.1, are formed:

The inputs are

- Key
- Nonce
- *a* data
- *m* data

The output is *c* data.

7.3.4.1 Key and nonce data inputs

The Key data for the CCM* forward transformation is passed by the outgoing frame security procedure described in 7.2.1. The Nonce data for the CCM* transformation is constructed as described in 7.3.2.

7.3.4.2 *a* data and *m* data

In the CCM* transformation process, the data fields shall be applied as in Table 53.

Table 53—*a* data and *m* data for all security levels

Security level identifier	***a* data**	***m* data**
0x00	None	None
0x01	MHR || Auxiliary security header || Nonpayload fields || Unsecured payload fields	None
0x02	MHR || Auxiliary security header || Nonpayload fields || Unsecured payload fields	None
0x03	MHR || Auxiliary security header || Nonpayload fields || Unsecured payload fields	None
0x04	None	Unsecured payload fields
0x05	MHR || Auxiliary security header || Nonpayload fields	Unsecured payload fields
0x06	MHR || Auxiliary security header || Nonpayload fields	Unsecured payload fields
0x07	MHR || Auxiliary security header || Nonpayload fields	Unsecured payload fields

7.3.4.3 *c* data output

In the CCM* transformation process, the data fields that are applied, or right-concatenated and applied, represent octet strings.

The secured payload fields right-concatenated with the authentication tag shall substitute the unsecured payload field in the original unsecured frame to form the secured frame, as defined in Table 54.

Table 54—*c* data for all security levels

Security level identifier	***c* data**
0x00	None
0x01	MIC-32
0x02	MIC-64
0x03	MIC-128
0x04	Secured payload fields
0x05	Secured payload fields || MIC-32
0x06	Secured payload fields || MIC-64
0x07	Secured payload fields || MIC-128

7.3.5 CCM* inverse transformation data representation

This subclause describes how the inputs and output of the CCM* inverse transformation, as described in B.4.2, are formed.

The inputs are

— Key

— Nonce

— *c* data

— *a* data

The output is *m* data.

7.3.5.1 Key and nonce data inputs

The Key data for the CCM* inverse transformation is passed by the incoming frame security procedure described in 7.2.4. The Nonce data for the CCM* transformation is constructed as described in 7.3.2.

7.3.5.2 *c* data and *a* data

In the CCM* inverse transformation process, the data fields shall be applied as in Table 55.

7.3.5.3 *m* data output

The *m* data shall replace the secured payload fields and authentication tag in the original secured frame to form the unsecured frame.

Table 55—*c* data and *a* data for all security levels

Security level identifier	***c* data**	***a* data**
0x00	None	None
0x01	MIC-32	MHR || Auxiliary security header || Non-payload fields ||Secured payload fields
0x02	MIC-64	MHR || Auxiliary security header || Non-payload fields || Secured payload fields
0x03	MIC-128	MHR || Auxiliary security header || Non-payload fields || Secured payload fields
0x04	Secured payload fields	MHR || Auxiliary security header || Non-payload fields
0x05	Secured payload fields || MIC-32	MHR || Auxiliary security header || Non-payload fields
0x06	Secured payload fields || MIC-64	MHR || Auxiliary security header || Non-payload fields
0x07	Secured payload fields || MIC-128	MHR || Auxiliary security header || Non-payload fields

7.4 Auxiliary security header

The Auxiliary Security Header field has a variable length and contains information required for security processing, including a Security Control field, a Frame Counter field, and a Key Identifier field. The Auxiliary Security Header field shall be present only if the Security Enabled field of the Frame Control field is set to one. The Auxiliary Security Header field shall be formatted as illustrated in Figure 69.

Octets: 1	**4**	**0/1/5/9**
Security Control	Frame Counter	Key Identifier

Figure 69—Format of the auxiliary security header

The auxiliary security header uses the representation conventions specified in 5.2.

7.4.1 Security Control field

The Security Control field is used to provide information about what protection is applied to the frame. The Security Control field shall be formatted as shown in Figure 70.

Bit: 0–2	**3–4**	**5–7**
Security Level	Key Identifier Mode	Reserved

Figure 70—Security Control field format

7.4.1.1 Security Level field

The Security Level field indicates the actual frame protection that is provided. This value can be adapted on a frame-by-frame basis and allows for varying levels of data authenticity (to allow minimization of security overhead in transmitted frames where required) and for optional data confidentiality. The cryptographic protection offered by the various security levels is shown in Table 56. When nontrivial protection is required, replay protection is always provided.

Table 56—Security levels available to the MAC sublayer

Security level identifier	**Security Control field (Figure 70) b_2 b_1 b_0**	**Security attributes**	**Data confidentiality**	**Data authenticity (including length *M* of authentication tag, in octets)**
0x00	'000'	None	OFF	NO (M = 0)
0x01	'001'	MIC-32	OFF	YES (M = 4)
0x02	'010'	MIC-64	OFF	YES (M = 8)
0x03	'011'	MIC-128	OFF	YES (M = 16)
0x04	'100'	ENC	ON	NO (M = 0)
0x05	'101'	ENC-MIC-32	ON	YES (M = 4)
0x06	'110'	ENC-MIC-64	ON	YES (M = 8)
0x07	'111'	ENC-MIC-128	ON	YES (M = 16)

Security levels can be ordered according to the corresponding cryptographic protection offered. Here, a first security level SEC1 is greater than or equal to a second security level SEC2 if and only if SEC1 offers at least the protection offered by SEC2, both with respect to data confidentiality and with respect to data authenticity. The statement “SEC1 is greater than or equal to SEC2” shall be evaluated as TRUE if both of the following conditions apply:

a) Bit position b_2 in SEC1 is greater than or equal to bit position b_2 in SEC2 (where Encryption OFF < Encryption ON).

b) The integer value of bit positions b_1 b_0 in SEC1 is greater than or equal to the integer value of bit positions b_1 b_0 in SEC2 (where increasing integer values indicate increasing levels of data authenticity provided, i.e., message integrity code (MIC)-0 < MIC-32 < MIC-64 < MIC-128).

Otherwise, the statement shall be evaluated as FALSE.

For example, ENC-MIC-64 ≥ MIC-64 is TRUE because ENC-MIC-64 offers the same data authenticity protection as MIC-64, plus confidentiality. On the other hand, MIC-128 ≥ ENC-MIC-64 is FALSE because even though MIC-128 offers stronger data authenticity than ENC-MIC-64, it offers no confidentiality.

7.4.1.2 Key Identifier Mode field

The Key Identifier Mode field indicates whether the key that is used to protect the frame can be derived implicitly or explicitly; furthermore, it is used to indicate the particular representations of the Key Identifier field (see 7.7.2.4) if derived explicitly. The Key Identifier Mode field shall be set to one of the values listed in Table 57. The Key Identifier field of the auxiliary security header (see 7.7.2.4) shall be present only if this field has a value that is not equal to 0x00.

Table 57—Values of the key identifier mode

Key identifier mode	**Key Identifier Mode field** **b_1 b_0**	**Description**	**Key Identifier field length (octets)**
0x00	'00'	Key is determined implicitly from the originator and receipient(s) of the frame, as indicated in the frame header.	0
0x01	'01'	Key is determined from the 1-octet Key Index field of the Key Identifier field of the auxiliary security header in conjunction with *macDefaultKeySource.*	1
0x02	'10'	Key is determined explicitly from the 4-octet Key Source field and the 1-octet Key Index field of the Key Identifier field of the auxiliary security header.	5
0x03	'11'	Key is determined explicitly from the 8-octet Key Source field and the 1-octet Key Index field of the Key Identifier field of the auxiliary security header.	9

7.4.2 Frame Counter field

The Frame Counter field represents the *macFrameCounter* attribute of the originator of a protected frame. It is used to provide semantic security of the cryptographic mechanism used to protect a frame and to offer replay protection.

7.4.3 Key Identifier field

The Key Identifier field has a variable length and identifies the key that is used for cryptographic protection of outgoing frames, either explicitly or in conjunction with implicitly defined side information. The Key Identifier field shall be present only if the Key Identifier Mode field of the Security Control field of the auxiliary security header (see 7.7.2.2.2) is set to a value different from 0x00. The Key Identifier field shall be formatted as illustrated in Figure 71.

Octets: 0/4/8	**1**
Key Source	Key Index

Figure 71—Format for the Key Identifier field, if present

7.4.3.1 Key Source field

The Key Source field, when present, indicates the originator of a group key.

7.4.3.2 Key Index field

The Key Index field allows unique identification of different keys with the same originator.

It is the responsibility of each key originator to make sure that actively used keys that it issues have distinct key indices and that the key indices are all different from 0x00.

7.5 Security-related MAC PIB attributes

The security-related MAC PIB attributes contain:

— Key table (*macKeyTable*, *macKeyTableEntries*)
— Device table (*macDeviceTable*, *macDeviceTableEntries*)
— Security level table (*macSecurityLevelTable*, *macSecurityLevelTableEntries*)
— Frame counter (*macFrameCounter*)
— Automatic request attributes (*macAutoRequestSecurityLevel*, *macAutoRequestKeyIdMode*, *macAutoRequestKeySource*, *macAutoRequestKeyIndex*)
— Default key source (*macDefaultKeySource*)
— PAN coordinator address (*macPANCoordExtendedAddress*, *macPANCoordShortAddress*)
— Key source table (*macKeySourceTable*, *macKeySourceTableEntries*)

7.5.1 Security-related MAC PIB attributes

The PIB security-related attributes are defined in Table 58. A MAC implementation may impose additional constraints on read/write operations for the security-related PIB attributes.

Table 58— Security-related MAC PIB attributes

Attribute	Type	Range	Description	Default
macKeyTable	Set of *Key Descriptors*, as defined Table 59	—	KeyDescriptor entries, each containing keys and security related information.	(empty)
macKeyTableEntries	Integer	Implementation specific	The number of entries in *macKeyTable*.	0
macDeviceTable	Set of *Device Descriptors*, as defined in Table 63	—	Device-Descriptors for each remote device with which this device securely communicates.	(empty)
macDeviceTable Entries	Integer	Implementation specific	The number of entries in *macDeviceTable*.	0
macSecurity LevelTable	Set of *SecurityLevel-Descriptors*, as defined in Table 62	—	Provides information about the security level required for each MAC frame type and subtype.	(empty)
macSecurity LevelTableEntries	Integer	Implementation specific	The number of entries in *macSecurityLevelTable*.	0
macFrameCounter	Integer	0x00000000–0xffffffff	The outgoing frame counter for this device.	0x00000000
macAutoRequest SecurityLevel	Integer	As defined in Table 56	The security level used for automatic data requests.	0x06

Table 58— Security-related MAC PIB attributes ***(continued)***

Attribute	Type	Range	Description	Default
macAutoRequest KeyIdMode	Integer	0x00–0x03	The key identifier mode used for automatic data requests. This attribute is invalid if the *macAuto-RequestSecurityLevel* attribute is set to 0x00.	0x00
macAutoRequest KeySource	As specified by the *macAutoRequesKey-IdMode* parameter	—	The originator of the key used for automatic data requests. This attribute is invalid if the *macAutoRequestKeyIdMode* element is invalid or set to 0x00.	All octets 0xff
macAutoRequest KeyIndex	Integer	0x01–0xff	The index of the key used for automatic data requests. This attribute is invalid if the *macAutoRequestKeyIdMode* attribute is invalid or set to 0x00.	All octets 0xff
macDefaultKey Source	Set of 8 octets	—	The originator of the default key used for key identifier mode 0x01.	All octets 0xff
macPANCoord ExtendedAddress	IEEE address	An extended IEEE address	The extended address of the PAN coordinator.	—
macPANCoordShort-Address	Integer	0x0000–0xffff	The short address assigned to the PAN coordinator. A value of 0xfffe indicates that the PAN coordinator is only using its extended address. A value of 0xffff indicates that this value is unknown.	0x0000
macKeySourceTable	0x7f	List of Key-SourceDe-scriptor entries (see Table 64)	–	A table of KeySource-Descriptor entries, each indicating key identifying information required for secured communications.
macKeySource-TableEntries	0x80	Integer	Implementation specific	The number of entries in *macKey-Source-Table*.

Table 59 defines the elements in a KeyDescriptor.

Table 60 defines the elements of a KeyUsageDescriptor.

Table 61 defines the elements of a KeyDeviceDescriptor.

Table 59—Elements of KeyDescriptor

Name	Type	Range	Description
ExtKeySource	Set of 8 octets	–	The 64-bit identifier of the key source (see 7.4.3.1)
KeyIndex	Integer	0x00-0xff	The identifier of the key index (see 7.4.3.2). A value of 0x00 indicates an implicitly identified key; a value in the range 0x01-0xff indicates an explicitly identified key.
Blacklisted	Boolean	TRUE or FALSE	Indicator as to whether the device previously communicated with this key prior to the exhaustion of the frame counter. If TRUE, this indicates that the device shall not use this key further, since it exhausted its use of the frame counter used with this key.
KeyDeviceList	List of KeyDevice-Descriptor entries, as defined in Table 61	—	A list of KeyDeviceDescriptor entries indicating which devices are currently using this key, including their blacklist status.
KeyDeviceListEntries	Integer	Implementation specific	The number of entries in KeyDeviceList.
KeyUsageList	List of KeyUsage-Descriptor entries, as defined in Table 60	—	A list of KeyUsageDescriptor entries indicating which frame types this key may be used with.
KeyUsageListEntries	Integer	—	The number of entries in KeyUsageList.
Key	Set of 16 octets	—	The actual value of the key.

Table 60—Elements of KeyUsageDescriptor

Name	Type	Range	Description
FrameType	Integer	As defined in 5.2.1.1.1	As defined in 5.2.1.1.1.
CommandFrameIdentifier	Integer	As defined in Table 5	As defined in Table 5.

Table 62 defines the elements of a SecurityLevelDescriptor.

Table 63 defines the elements of a DeviceDescriptor.

Table 64 defines the elements of a KeyIdLookupDescriptor.

Table 65 defines the elements of a KeySourceDescriptor.

7.5.2 Key table

The key table holds key descriptors (keys with related key-specific information) that are required for security processing of outgoing and incoming frames. Key-specific information in the key table is identified

Table 61—Elements of KeyDeviceDescriptor

Name	Type	Range	Description
DeviceDescriptorHandle	Integer	Implementation specific	Handle to the DeviceDescriptor corresponding to the device, as defined in Table 63.
UniqueDevice	Boolean	TRUE, FALSE	Indication of whether the device indicated by DeviceDescriptorHandle is uniquely associated with the KeyDescriptor, i.e., it is a link key as opposed to a group key.
Blacklisted	Boolean	TRUE, FALSE	Indication of whether the device indicated by DeviceDescriptorHandle previously communicated with this key prior to the exhaustion of the frame counter. If TRUE, this indicates that the device shall not use this key further because it exhausted its use of the frame counter used with this key.

Table 62—Elements of SecurityLevelDescriptor

Name	Type	Range	Description
FrameType	Integer	As defined in 5.2.1.1.1	As defined in 5.2.1.1.1.
CommandFrameIdentifier	Integer	As defined in Table 5	As defined in Table 5.
SecurityMinimum	Integer	As defined in Table 56	The minimal required/expected security level, as defined in Table 56, for incoming MAC frames with the indicated frame type and, if present, command frame type.
DeviceOverrideSecurity-Minimum	Boolean	TRUE, FALSE	Indication of whether originating devices for which the Exempt flag is set may override the security level indicated by the AllowedSecurityLevels, or if the the or SecurityMinimum. If TRUE, this indicates that for originating devices with Exempt status, the incoming security level zero is acceptable, in addition to the incoming security levels meeting the minimum expected security level indicated by the SecurityMinimum element.
AllowedSecurityLevels	Set of integers	—	A set of allowed security levels, as defined in Table 56, for incoming MAC frames with the indicated frame type, and, if present, command frame identifier. If the set is empty, then the SecurityMinimum parameter applies instead

based on information explicitly contained in the requesting primitive or in the received frame, as described in the outgoing frame key retrieval procedure, as described in 7.2.2, and the incoming frame security material retrieval procedure, as described in 7.2.6, as well as in the KeyDescriptor lookup procedure, as described in 7.2.3.

Table 63—Elements of DeviceDescriptor

Name	Type	Range	Description
PANId	Device PAN ID	0x0000–0xffff	The PAN identifier of the device in this DeviceDescriptor.
ShortAddress	Device short address	0x0000–0xffff	The short address of the device in this DeviceDescriptor. A value of 0xfffe indicates that this device is using only its extended address. A value of 0xffff indicates that this value is unknown.
ExtAddress	IEEE address	Any valid extended IEEE address	The extended IEEE address of the device in this DeviceDescriptor. This element is also used in unsecuring operations on incoming frames.
FrameCounter	Integer	0x00000000–0xffffffff	The incoming frame counter of the device in this DeviceDescriptor. This value is used to ensure sequential freshness of frames.
Exempt	Boolean	TRUE, FALSE	Indication of whether the device may override the minimum security level settings defined in Table 62.

Table 64—Elements of KeyIdLookupDescriptor

Name	Type	Range	Description
LookupData	Set of 5 or 9 octets	—	Data used to identify the key.
LookupDataSize	Integer	0x00–0x01	A value of 0x00 indicates a set of 5 octets; a value of 0x01 indicates a set of 9 octets.

Table 65—Elements of KeySourceDescriptor

Name	Type	Range	Description
ExtKeySource	Set of 8 octets	–	The 64-bit identifier of the key source (see 7.4.3.1).
ShortKeySource	Set of 4 octets	–	The 32-bit identifier of the ExtKeySource in this KeySourceDescriptor. A value of 0xfffffffe indicates that only ExtKeySource is used. A value of 0xffffffff indicates that this value is unknown.

7.5.3 Device table

The device table holds device descriptors (device-specific addressing information and security-related information) that, when combined with key-specific information from the key table, provide all the keying material needed to secure outgoing, as described in 7.2.1, and unsecure incoming frames, as described in 7.2.4. Device-specific information in the device table is identified based on the originator of the frame, as described in the DeviceDescriptor lookup procedure, as described in 7.2.10, and on key-specific information, as described in the blacklist checking procedure, as described in 7.2.9.

7.5.4 Security level table

The security level table holds information regarding the security levels the device expects to have been applied by the originator of a frame, depending on frame type and, if it concerns a MAC command frame, the command frame identifier. Security processing of an incoming frame will fail if the frame is not adequately protected, as described in the incoming frame security procedure, as described in 7.2.4, and in the incoming security level checking procedure, as described in 7.2.10.

7.5.5 Frame counter

The 4-octet frame counter is used to provide replay protection and semantic security of the cryptographic building block used for securing outgoing frames. The frame counter is included in each secured frame and is one of the elements required for the unsecuring operation at the recipient(s). The frame counter is incremented each time an outgoing frame is secured, as described in the outgoing frame security procedure, as described in 7.2.1. When the frame counter reaches its maximum value of 0xffffffff, the associated keying material can no longer be used, thus requiring all keys associated with the device to be updated. This provides a mechanism for ensuring that the keying material for every frame is unique and, thereby, provides for sequential freshness.

7.5.6 Automatic request attributes

Automatic request attributes hold all the information needed to secure outgoing frames generated automatically and not as a result of a higher layer primitive, as is the case with automatic data requests.

7.5.7 Default key source

The default key source is information commonly shared between originator and receipient(s) of a secured frame, which, when combined with additional information explicitly contained in the requesting primitive or in the received frame, allows an originator or a recipient to determine the key required for securing or unsecuring this frame, respectively. This provides a mechanism for significantly reducing the overhead of security information contained in secured frames in particular use cases, as described in 7.2.2 and 7.2.6.

7.5.8 PAN coordinator address

The address of the PAN coordinator is information commonly shared between all devices in a PAN, which, when combined with additional information explicitly contained in the requesting primitive or in the received frame, allows an originator of a frame directed to the PAN coordinator or a recipient of a frame originating from the PAN coordinator to determine the key and security-related information required for securing or unsecuring, respectively, this frame, as described in 7.2.2 and 7.2.6.

7.5.9 Key source table

The key source table holds key source descriptors (key-specific information) that, when combined with additional information explicitly contained in the requesting primitive or in the received frame, allow an originator or a recipient to determine the key required for securing or unsecuring this frame, respectively (see 7.2.2 and 7.2.6). For received frames, this information may be either implicitly derived from the addressing fields of the frame or explicitly indicated in the frame by its originator, as described in the outgoing frame key retrieval procedure (see 7.2.2) and the incoming frame key retrieval procedure (see 7.2.6).