Project	**IEEE 802.16 Broadband Wireless Access Working Group <http://ieee802.org/16>**	
Title	**AES-CCM Encryption and Authentication Mode for 802.16**	
Date Submitted	**2004-01-10**	
Source(s)	David Johnston Intel Corp 2111 NE 25th Avenue Hillsboro, OR 97124-5961	Voice: 503 264 3855 Fax: mailto:dj.johnston@intel.com
Re:	802.16e	
Abstract	This document describes changes to the 802.16 specification, for incorporation into the 802.16e amendment. These changes introduce a new encryption algorithm, authentication algorithm and key encryption algorithm based on AES operating in CCM mode. This new mode is incorporated into the extensible cipher selector mechanism that exists in the base specification.	
Purpose	These changes are to be incorporated into 802.16e in order to enhance the available cryptographic strength of the link cipher to 128 bits and to introduce data origin authentication and data integrity protection.	
Notice	This document has been prepared to assist IEEE 802.16. It is offered as a basis for discussion and is not binding the contributing individual(s) or organization(s). The material in this document is subject to change in form and content after further study. The contributor(s) reserve(s) the right to add, amend or withdraw material contained herein.	
Release	The contributor grants a free, irrevocable license to the IEEE to incorporate material contained in this contributio and any modifications thereof, in the creation of an IEEE Standards publication; to copyright in the IEEE's name any IEEE Standards publication even though it may include portions of this contribution; and at the IEEE's sole discretion to permit others to reproduce in whole or in part the resulting IEEE Standards publication. The contributor also acknowledges and accepts that this contribution may be made public by IEEE 802.16.	
Patent Policy and Procedures	The contributor is familiar with the IEEE 802.16 Patent Policy and Procedures <http://ieee802.org/16/ipr/patents/policy.html>, including the statement "IEEE standards may include the known use of patent(s), including patent applications, provided the IEEE receives assurance from the patent holder or applicant with respect to patents essential for compliance with both mandatory and optional portions of the standard." Early disclosure to the Working Group of patent information that might be relevant to the standard is essential to reduce the possibility for delays in the development process and increase the likelihood that the draft publication will be approved for publication. Please notify the Chair <mailto:chair@wirelessman.org> as early a possible, in written or electronic form, if patented technology (or technology under patent application) might be incorporated into a draft standard being developed within the IEEE 802.16 Working Group. The Chair will discl this notification via the IEEE 802.16 web site <http://ieee802.org/16/ipr/patents/notices>.	

AES-CCM Encryption and Authentication Mode for 802.16

David Johnston
Intel Corp

[Figure 1, Replace both instances of 'Privacy Sublayer' with 'Security Sublayer].

[1.4, Replace 'Privacy Sublayer' with 'Security Sublayer']

[2. References, Insert the following references]

FIPS 197, Advanced Encryption Standard (AES)

NIST Special Publication 800-38C, Recommendation for Block Cipher Modes of Operation: The CCM Model for Authentication and Confidentiality

[4. Abbreviations and Acronyms, Insert the following abbreviations]

AES Advanced Encryption Standard

CCM CTR mode with CBC-MAC

CBC-MAC Cipher Block Chaining Message Authentication Code

CTR Counter mode encyrption

[Amend 6.4.3.6 as follows]

6.4.3.6 Cryptographic Protection of MAC PDUs

Deleted: Encryption

When transmitting a MAC PDU on a connection that is mapped to an SA, the sender shall perform encryption and data authentication of the MAC PDU payload as specified by that SA. When receiving a MAC PDU on a connection mapped to an SA, the receiver shall perform decryption and data authentication of the MAC PDU payload, as specified by that SA.

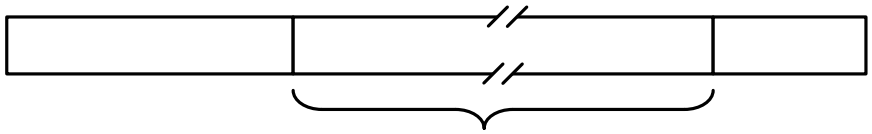
Deleted: NOTE—Data authentication is not currently defined.¶

The generic MAC header shall not be encrypted. The Header contains all the information [EC Field, encryption key sequence (EKS) Field, and CID] needed to decrypt and/or authenticate a Payload at the receiving station. This is illustrated in Figure 32.

Deleted: Encryption

Two bits of a MAC Header contain a key sequence number. Note that the keying material associated with an SA has a limited lifetime, and the BS periodically refreshes an SA's keying material. The BS manages a 2 bit key sequence number independently for each SA and distributes this key sequence number along with the SA's keying material to the client SS. The BS increments the key sequence number with each new generation of keying material. The MAC Header includes this sequence number to identify the specific generation of that SA keying material being used to encrypt the attached payload. Being a 2-bit quantity, the sequence number wraps around to 0 when it reaches 3.

Formatted: Font: (Default) TimesNewRoman, 10 pt



Generic MAC h

Deleted:

Figure 32—MAC PDU encryption

Comparing a received MAC PDU's key sequence number with what it believes to be the "current" key sequence number, an SS or BS can easily recognize a loss of key synchronization with its peer. An SS shall maintain the two most recent generations of keying material for each SA. Keeping on hand the two most recent key generations is necessary for maintaining uninterrupted service during an SA's key transition.

Protection of the payload is indicated by the EC bit field. A value of 1 indicates the payload is cryptographically protected and the EKS field contains meaningful data. A value of 0 indicates the payload is not cryptographically protected. A PDU with no payload shall not be protected. Any unencrypted MAC PDU received on a connection mapped to an SA requiring encryption shall be discarded unless the PDU contains no payload.

Deleted: Encryption
Deleted: encrypted
Deleted: encrypted

[Amend clause 7 as follows]

7. Security sublayer

Deleted: Privacy

The security sublayer provides subscribers with privacy, authentication or confidentiality[15] across the broadband wireless network. It does this by applying cryptographic transforms to MPDUs carried across connections between SS and BS.

In addition, security provides operators with strong protection from theft of service. The BS protects against unauthorized access to these data transport services by securing the associated service flows across the network. Security employs an authenticated client/server key management protocol in which the BS, the server, controls distribution of keying material to client SS. Additionally, the transport connection security mechanisms are strengthened by adding digital-certificate-based SS device-authentication to its key management protocol.

Deleted: Privacy
Deleted: fixed
Deleted: ¶ encrypting
Deleted: Privacy
Deleted: enforcing encryption of
Deleted: Privacy
Deleted: basic privacy

7.1 Architecture

Security has two component protocols as follows:

Deleted: Privacy

a) An encapsulation protocol for securing packet data across the BWA network. This protocol defines (1) a set of supported *cryptographic suites*, i.e., pairings of data encryption and authentication algorithms, and (2) the rules for applying those algorithms to a MAC PDU payload.
b) A key management protocol (PKM) providing the secure distribution of keying data from BS to SS. Through this key management protocol, SS and BS synchronize keying data; in addition, the BS uses the protocol to enforce conditional access to network services.

Deleted: encrypting
Deleted: fixed

7.1.1 Packet data encryption

Encryption services are defined as a set of capabilities within the MAC Security Sublayer. MAC Header information specific to encryption is allocated in the generic MAC header format.

Deleted: Privacy

Encryption is applied to the MAC PDU payload when required by the selected ciphersuite; the generic MAC header is not encrypted. All MAC management messages shall be sent in the clear to facilitate registration, ranging, and normal operation of the MAC.

Deleted: always

The format of MAC PDUs carrying secured packet data payloads is specified in 6.4.3.6.

Deleted: encrypted

[15] In security parlance, confidentiality = privacy + authenticity

7.1.2 Key management protocol

An SS uses the PKM protocol to obtain device authorization and traffic keying material from the BS, and to support periodic reauthorization and key refresh. The key management protocol uses X.509 digital certificates [IETF RFC 3280], the RSA public-key encryption algorithm [PKCS #1], and strong symmetric algorithms to perform key exchanges between SS and BS.

The PKM protocol adheres to a client/server model, where the SS, a PKM "client," requests keying material, and the BS, a PKM "server," responds to those requests, ensuring that individual SS clients receive only keying material for which they are authorized. The PKM protocol uses MAC management messaging, i.e., PKM-REQ and PKM-RSP messages defined in 6.4.2.3.

The PKM protocol uses public-key cryptography to establish a shared secret (i.e., an AK) between SS and BS. The shared secret is then used to secure subsequent PKM exchanges of TEKs. This two-tiered mechanism for key distribution permits refreshing of TEKs without incurring the overhead of computation intensive public-key operations.

A BS authenticates a client SS during the initial authorization exchange. Each SS carries a unique X.509 digital certificate issued by the SS's manufacturer. The digital certificate contains the SS's Public Key and SS MAC address. When requesting an AK, an SS presents its digital certificate to the BS. The BS verifies the digital certificate, and then uses the verified Public Key to encrypt an AK, which the BS then sends back to the requesting SS.

The BS associates an SS's authenticated identity to a paying subscriber, and hence to the data services that subscriber is authorized to access. Thus, with the AK exchange, the BS establishes an authenticated identity of a client SS and the services (i.e., specific TEKs) the SS is authorized to access.

Since the BS authenticates the SS, it can protect against an attacker employing a *cloned* SS, masquerading as a legitimate subscriber's SS. The use of the X.509 certificates prevents cloned SSs from passing fake credentials onto a BS.

All SSs shall have factory-installed RSA private/public key pairs or provide an internal algorithm to generate such key pairs dynamically. If an SS relies on an internal algorithm to generate its RSA key pair, the SS shall generate the key pair prior to its first AK exchange, described in 7.2.1. All SSs with factory-installed RSA key pairs shall also have factory-installed X.509 certificates. All SSs that rely on internal algorithms to generate an RSA key pair shall support a mechanism for installing a manufacturer-issued X.509 certificate following key generation.

The PKM protocol is defined in detail in 7.2.

7.1.3 Security Associations

A *Security Association* (SA) is the set of security information a BS and one or more of its client SSs share in order to support secure communications across the IEEE Std 802.16 network. Three types of SAs are defined: *Primary, Static*, and *Dynamic*. Each SS establishes a Primary Security association during the SS initialization process. Static SAs are provisioned within the BS. Dynamic SAs are established and eliminated, on the fly, in response to the initiation and termination of specific service flows. Both Static and Dynamic SAs can be shared by multiple SSs.

An SA's shared information shall include the Cryptographic Suite employed within the SA. The shared information may include TEKs and Initialization Vectors. The exact content of the SA is dependent on the SA's Cryptographic Suite.

SAs are identified using SAIDs.

Each SS shall establish an exclusive Primary SA with its BS. The SAID of any SS's Primary SA shall be equal to the Basic CID of that SS.

Using the PKM protocol, an SS requests from its BS an SA's keying material. The BS shall ensure that each client SS only has access to the SAs it is authorized to access.

An SA's keying material [e.g., Data Encryption Standard (DES) key and CBC Initialization Vector] has a limited lifetime. When the BS delivers SA keying material to an SS, it also provides the SS with that material's remaining lifetime. It is the responsibility of the SS to request new keying material from the BS before the set of keying material that the SS currently holds expires at the BS. Should the current keying material expire before a new set of keying material is received, the SS shall perform network entry as described in 6.4.9.

In certain ciphersuites, key lifetime may be limited by the exhaustion rate of a number space [e.g. the PN (Packet Number) in AES-CCM mode]. In this case, the key ends either at the expiry of the key lifetime or the exhaustion of the number space, which ever is earliest. Note that in this case, security is not determined by the key lifetime .

The PKM protocol specifies how SS and BS maintain key synchronization.

Deleted:

7.2 PKM protocol

7.2.2 TEK exchange overview

7.2.2.1 TEK exchange overview for PMP topology

Upon achieving authorization, an SS starts a separate TEK state machine for each of the SAIDs identified in the Authorization Reply message. Each TEK state machine operating within the SS is responsible for managing the keying material associated with its respective SAID. TEK state machines periodically send Key Request messages to the BS, requesting a refresh of keying material for their respective SAIDs.

The BS responds to a Key Request with a Key Reply message, containing the BS's active keying material for a specific SAID.

TEKs and KEKs may be either 64 bits or 128 bits long. SAs employing any ciphersuite with a basic block size of 128 bits shall use 128 bit TEKs and KEKs. Otherwise 64 bit TEKs and KEKs shall be used. The name TEK-64 is used to denote a 64 bit TEK and TEK-128 is used to denote a 128 bit TEK. Similarly, KEK-64 is used to denote a 64 bit KEK and KEK-128 is used to denote a 128 bit KEK.

For SAs using a ciphersuite employing DES-CBC the TEK in the Key Reply is triple DES (3-DES) (encrypt-decrypt-encrypt or EDE mode) encrypted, using a two-key, 3-DES KEK derived from the AK.

Deleted: T

For SAs using a ciphersuite employing 128 bits keys, such as AES-CCM mode, the TEK in the key Reply is AES encrypted using a 128 bit key derived from the AK and a 128 bit block size.

Note that at all times the BS maintains two active sets of keying material per SAID. The lifetimes of the two generations overlap such that each generation becomes active halfway through the life of it predecessor and expires halfway through the life of its successor. A BS includes in its Key Replies *both* of an SAID's active generations of keying material.

For SAs using a ciphersuite employing CBC mode encryption the Key Reply provides the requesting SS, in addition to the TEK and CBC initialization vector, the remaining lifetime of each of the two sets of keying material. The receiving SS uses these remaining lifetimes to estimate when the BS will invalidate a particular TEK, and therefore when to schedule future Key Requests such that the SS requests and receives new keying material before the BS expires the keying material the SS currently holds.

Deleted: ¶ T

For SAs using a ciphersuite employing AES-CCM mode, the Key Reply provides the requesting SS, in addition to the TEK, the remaining lifetime of each of the two sets of keying material. The receiving SS uses these remaining lifetimes to estimate when the BS will invalidate a particular TEK, and therefore when to schedule future Key Requests such that the SS requests and receives new keying material before the BS expires the keying material the SS currently holds. When more than half the available PN numbers in the 31 bit PN number space are exhausted, the SS shall schedule a future Key Request in the same fashion as if the key lifetime was approaching expiry.

The operation of the TEK state machine's Key Request scheduling algorithm, combined with the BS's regimen for updating and using an SAID's keying material (see 7.4), ensures that the SS will be able to continually exchange encrypted traffic with the BS.

A TEK state machine remains active as long as

a) the SS is authorized to operate in the BS's security domain, i.e., it has a valid AK, and

b) the SS is authorized to participate in that particular SA, i.e., the BS continues to provide fresh keying material during rekey cycles.

The parent Authorization state machine stops *all* of its child TEK state machines when the SS receives from the BS an Authorization Reject during a reauthorization cycle. Individual TEK state machines can be started or stopped during a reauthorization cycle if an SS's Static SAID authorizations changed between successive re-authorizations.

Communication between Authorization and TEK state machines occurs through the passing of events and protocol messaging. The Authorization state machine generates events (i.e., Stop, Authorized, Authorization Pending, and Authorization Complete events) that are targeted at its child TEK state machines. TEK state machines do not target events at their parent Authorization state machine. The TEK state machine affects the Authorization state machine indirectly through the messaging a BS sends in response to an SS's requests: a BS may respond to a TEK machine's Key Requests with a failure response (i.e., Authorization Invalid message) to be handled by the Authorization state machine.

7.4 Key usage

7.4.1.4 TEK lifetime

The BS shall maintain two sets of active TEKs (and their associated IVs (Initialization Vectors) , or PNs (packet numbers)) per SAID, corresponding to two successive generations of keying material. The two generations of TEKs shall have overlapping lifetimes determined by *TEK Lifetime*, a predefined BS system configuration parameter. The newer TEK shall have a key sequence number one greater (modulo 4) than that of the older TEK. Each TEK becomes active halfway through the lifetime of its predecessor and expires halfway through the lifetime of its successor. Once a TEK's lifetime expires, the TEK becomes inactive and shall no longer be used.

Deleted: IVs

The Key Reply messages sent by a BS contain TEK parameters for the two active TEKs. The TEKs' active lifetimes a BS reports in a Key Reply message shall reflect, as accurately as an implementation permits, the remaining lifetimes of these TEKs at the time the Key Reply message is sent.

7.4.1.5 BS usage of TEK

The BS transitions between the two active TEKs differently, depending on whether the TEK is used for downlink or uplink traffic. For each of its SAIDs, the BS shall transition between active TEKs according to the following rules:

Deleted: for

a) At expiration of the older TEK, the BS shall immediately transition to using the newer TEK for encryption.

b) The uplink transition period begins from the time the BS sends the newer TEK in a Key Reply message and concludes once the older TEK expires.

It is the responsibility of the SS to update its keys in a timely fashion; the BS shall transition to a new downlink encryption key regardless of whether a client SS has retrieved a copy of that TEK.

The BS uses the two active TEKs differently, depending on whether the TEK is used for downlink or uplink traffic. For each of its SAIDs, the BS shall use the two active TEKs according to the following rules:

a) The BS shall use the older of the two active TEKs for encrypting downlink traffic.

b) The BS shall be able to decrypt uplink traffic using either the older or newer TEK.

Note that the BS encrypts with a given TEK for only the second half of that TEK's total lifetime. The BS is able, however, to decrypt with a TEK for the TEK's entire lifetime.

The right-hand side of Figure 129 illustrates the BS's management of an SA's TEKs, where the shaded portion of a TEK's lifetime indicates the time period during which that TEK shall be used to encrypt MAC PDU payloads.

7.4.1.6 Node re-authorization in Mesh Mode during normal operation

When re-authorizing with the network, the re-authorizing node shall tunnel the authorization messages as shown in Figure 126 over UDP.

7.4.2 SS key usage

The SS is responsible for sustaining authorization with its BS and maintaining an active AK. An SS shall be prepared to use its two most recently obtained AKs according to the following manner.

7.4.2.1 SS reauthorization

AKs have a limited lifetime and shall be periodically refreshed. An SS refreshes its AK by reissuing an Auth Request to the BS. The Authorization State Machine (7.2.4) manages the scheduling of Auth Requests for refreshing AKs.

An SS's Authorization state machine schedules the beginning of reauthorization a configurable duration of time, the *Authorization Grace Time*, [see points (x) and (y) in Figure 128], before the SS's latest AK is scheduled to expire. The Authorization Grace Time is configured to provide an SS with an authorization retry period that is sufficiently long to allow for system delays and provide adequate time for the SS to successfully complete an Authorization exchange before the expiration of its most current AK.

Note that the BS does not require knowledge of the Authorization Grace Time. The BS, however, shall track the lifetimes of its AKs and shall deactivate a key once it has expired.

[No Change to Figure 129]

Figure 129—TEK management in BS and SS

7.4.2.2 SS usage of AK

An SS shall use the HMAC_KEY_U derived from the newer of its two most recent AKs when calculating the HMAC-Digests it attaches to Key Request messages.

The SS shall be able to use the HMAC_KEY_D derived from either of its two most recent AKs to authenticate Key Reply, Key Reject, and TEK Reject messages. The SS shall be able to decrypt an encrypted TEK in a Key Reply message with the KEK derived from either of its two most recent AKs. The SS shall use the accompanying AK Key Sequence Number to determine which set of keying material to use.

The left-hand side of Figure 128 illustrates an SS's maintenance and usage of its AKs, where the shaded portion of an AK's lifetime indicates the time period during which that AK shall be used to decrypt TEKs. Even though it is not part of the message exchange, Figure 128 also shows the implicit acknowledgment of the reception of a new AK via the transmission of a Key Request message using the key sequence of the new AK.

An SS shall use the HMAC_KEY_U derived from the newer of its two most recent AKs when calculating the HMAC-Digests of the HMAC Tuple attribute.

7.4.2.3 SS usage of TEK

An SS shall be capable of maintaining two successive sets of traffic keying material per authorized SAID. Through operation of its TEK state machines, an SS shall request a new set of traffic keying material a configurable amount of time, the *TEK Grace Time* [see points (x) and (y) in Figure 129], before the SS's latest TEK is scheduled to expire.

For each of its authorized SAIDs, the SS:

a) shall use the newer of its two TEKs to encrypt uplink traffic, and

b) shall be able to decrypt downlink traffic encrypted with either of the TEKs.

The left-hand side of Figure 129 illustrates the SS's maintenance and usage of an SA's TEKs, where the shaded portion of a TEK's lifetime indicates the time period during which that TEK shall be used to encrypt MAC PDU payloads.

7.4.2.4 TEK usage in Mesh Mode

For each of its SAIDs, the Neighbor shall transition between active TEKs according to the following rules:

a) At expiration of the older TEK, the Neighbor shall immediately transition to using the newer TEK for encryption.

b) The Neighbor that generated the TEK shall use the older of the two active TEKs for encrypting traffic towards the Node that initiated the TEK exchange.

c) The Neighbor that generated the TEK shall be able to decrypt traffic from each Node using either the older or newer TEK.

For each of its authorized SAIDs, the initiator Node:

a) shall use the newer of its two TEKs to encrypt traffic towards its Neighbors with which it initiated a TEK exchange, and

b) shall be able to decrypt traffic from the Neighbor encrypted with either of the TEKs.

7.4.2.5 Node usage of the Operator Shared Secret in Mesh Nodes

Each node shall be capable of maintaining two active Operator Shared Secrets. A Node shall use the Operator Shared Secret to calculate a HMAC-Digest for the Key Request and Key Reply messages when exchanging TEKs with its neighboring nodes.

7.5 Cryptographic methods

This subclause specifies the cryptographic algorithms and key sizes used by the PKM protocol. All SS and BS implementations shall support the method of packet data encryption defined in 7.5.1, encryption of the TEK as specified in 7.5.2, and message digest calculation as specified in 7.5.3.

7.5.1 Data Encryption methods

Formatted: Heading 3, Adjust space between Latin and Asian text, Adjust space between Asian text and numbers

7.5.1.1 Data encryption with DES in CBC mode

Formatted: Heading 4

If the data encryption algorithm identifier in the cryptographic suite of an SA equals 0x01, data on connections associated with that SA shall use the CBC mode of the US Data Encryption Standard (DES) algorithm [FIPS 46-3, FIPS 74, FIPS 81] to encrypt the MAC PDU payloads.

The CBC IV shall be calculated as follows: in the downlink, the CBC shall be initialized with the exclusive-or (XOR) of (1) the IV parameter included in the TEK-64 keying information, and (2) the content of the PHY Synchronization field (right justified) of the latest DL-MAP. In the uplink, the CBC shall be initialized with the XOR of (1) the IV parameter included in the TEK-64 keying information, and (2) the content of the PHY Synchronization field of the DL-MAP that is in effect when the UL-MAP for the uplink transmission is created/received.

Residual termination block processing shall be used to encrypt the final block of plaintext when the final block is less than 64 bits. Given a final block having *n* bits, where *n* is less than 64, the next-to-last ciphertext block shall be DES encrypted a second time, using the electronic code book (ECB) mode, and the most significant *n* bits of the result are XORed with the final *n* bits of the payload to generate the short

final cipher block. In order for the receiver to decrypt the short final cipher block, the receiver DES encrypts the next-to-last ciphertext block, using the ECB mode, and XORs the most significant *n* bits with the short final cipher block in order to recover the short final cleartext block. This encryption procedure is depicted in Figure 9.4 of Schneier [B42].

In the special case when the payload portion of the MAC PDU is less than 64 bits, the IV shall be DES encrypted and the most significant *n* bits of the resulting ciphertext, corresponding to the number of bits of the payload, shall be XORed with the *n* bits of the payload to generate the short cipher block.[16]

7.5.1.2 Data encryption with AES in CCM mode

If the data encryption algorithm identifier in the cryptographic suite of an SA equals 0x02, data on connections associated with that SA shall use the CCM mode of the US Advanced Encryption Standard (AES) algorithm [NIST Special Publication 800-38C, FIPS-197] to encrypt the MAC PDU payloads.

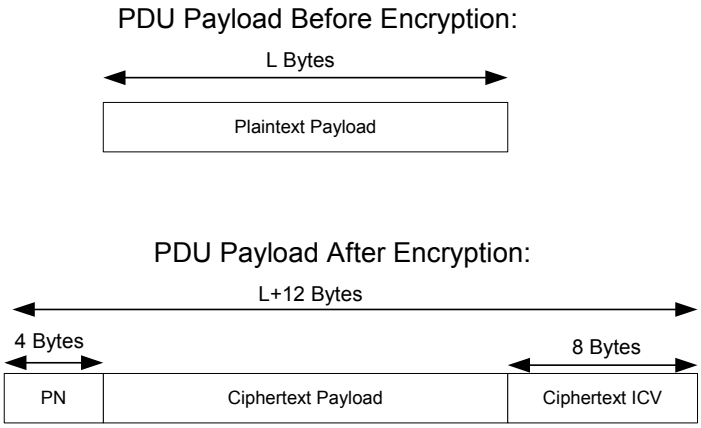
7.5.1.2.1 PDU Payload Format

The PDU payload shall be prepended with a 4 byte PN (Packet Number). The PN shall be transmitted in little endian byte order. The PN shall not be encrypted.

The plaintext PDU shall be encrypted and authenticated using the active TEK, according to the CCM specification. This includes appending an 8 byte ICV (Integrity Check Value) to the end of the payload and encrypting the both the plaintext payload and the appended ICV.

The ciphertext ICV is transmitted in little endian byte order.

The processing yields a payload that is 12 bytes longer than the plaintext payload.



Formatted: Heading 4, Adjust space between Latin and Asian text, Adjust space between Asian text and numbers

Deleted: ¶

Formatted: Heading 5, Adjust space between Latin and Asian text, Adjust space between Asian text and numbers

Formatted: Font: (Default) TimesNewRoman, 10 pt

16 If two or more PDUs with less than 8 byte payloads are transmitted in the same frame using the same SA, the XOR of the payload plaintexts can be found easily. In practice, this situation is very unlikely to occur, as payloads are typically larger than 8 bytes. In the case that multiple payloads of less than 8 bytes are to be transmitted in the same frame on the same SA and service, packing of the short SDUs into a single PDU will eliminate this weakness. If the SDUs are for different services, packing the SDUs with zero-length fictitious SDUs allows the use of the Packing subheader to extend the size of the PDU to at least 8 bytes.

Figure xxx MAC PDU Ciphertext Payload Format

Formatted: Font: Arial,Bold, Bold

Formatted: Centered, Adjust space between Latin and Asian text, Adjust space between Asian text and numbers

Formatted: Heading 5, Adjust space between Latin and Asian text, Adjust space between Asian text and numbers

7.5.1.2.2 PN (Packet Number)

The PN associated with an SA shall be set to 1 when the SA is established and when a new TEK is installed.

The PN shall be transmitted in little endian order in the MAC PDU as described in 7.5.1.2.1. After each PDU transmission, the PN shall be incremented by 1. On uplink connections, the PN shall be XORed with 0x80000000 prior to encryption and transmission. On downlink connections, the PN shall be used without such modification[17].

Any tuple value of {PN, KEY} shall not be used more than once for the purposes of transmitting data[18]. The SS shall ensure that a new TEK is requested and transferred before the PN on either the SS or BS reaches 0x7FFFFFFF. If the PN in either the SS or BS reaches 0x7FFFFFFF without new keys being installed, transport communications on that SA shall be halted until new TEKs are installed.

Formatted: Heading 5, Adjust space between Latin and Asian text, Adjust space between Asian text and numbers

7.5.1.2.3 CCM Algorithm

The NIST CCM specification defines a number of algorithm parameters. These parameters shall be fixed to specific values when used in SAs with a data encryption algorithm identifier of 0x02.

The number of octets in the authentication field M shall be set to 8. Consistent with the CCM specification the 3 bit binary encoding of M shall be 011.

The size of the length field shall be set to 2. Consistent with the CCM specification, the 3 bit binary encoding of the DLEN size field shall be 001.

The length of the additional authenticated data string l(a) shall be set to 0.

The nonce shall be 13 bytes long. Bytes 1 through 5 shall be set to the first five byte of the GMH (thus excluding the HCS). Bytes 6 through 9 are reserved and shall be set to 0x00000000. Bytes 10 through 13 shall be set to the value of the PN. Byte 10 shall take the LSB and byte 13 shall take the MSB.

Consistent with the CCM specification, the initial block B_0 is formatted as shown in figure xxx.

Formatted: Subscript

[17] This achieves the splitting of the PN space. 0x00000001 – 0x7FFFFFFF for the downlink and 0x80000001 – 0xFFFFFFFF on the uplink, preventing a PN collision between the uplink and downlink.

[18] Sending two packets encoded with the same key and PN will eliminate all security guarantees of CCM mode.

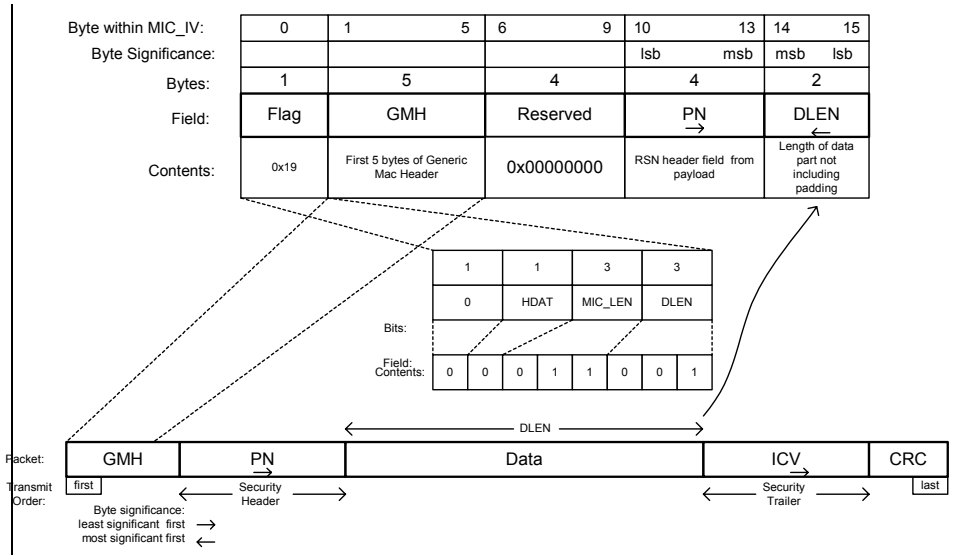


Figure XXX. Initial CCM block B_0

Note the big endian ordering of the DLEN value is opposite that of the normal little endian representation. This is to remain compliant with the letter of the NIST CCM specification.

The 6th byte of the GMH is not included in the nonce since it is redundant.

Consistent with the NIST CCM specification the counter blocks A_i are formatted as shown in figure xxx.

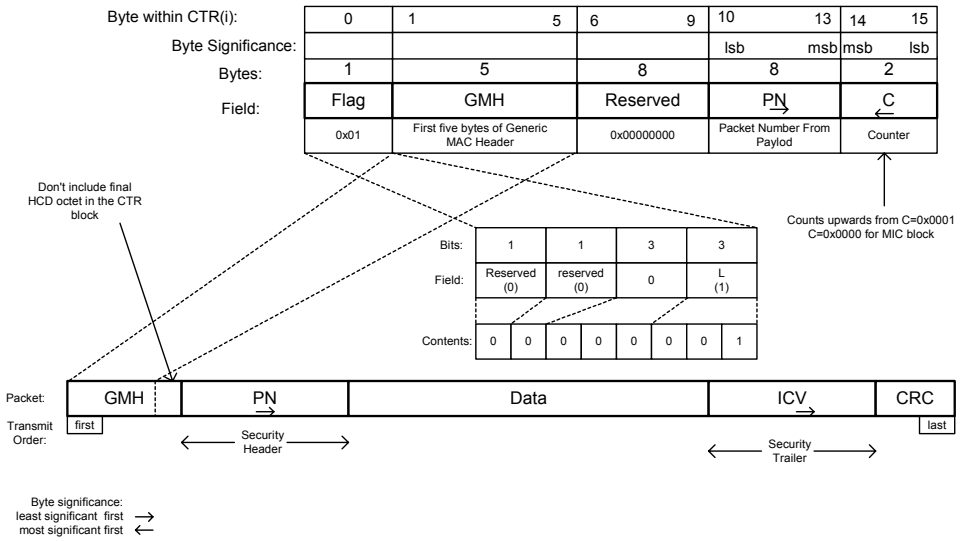


Figure xxx. Construction of A_i

Formatted: Font: Arial,Bold, 10 pt, Bold

Formatted: Font: Arial,Bold, 10 pt, Bold

Formatted: Indent: Left: 0"

Formatted: Font: Arial,Bold, 10 pt, Bold, Subscript

Formatted: Subscript

Formatted: Font: Arial,Bold, 10 pt, Bold

Formatted: Font: Arial,Bold, 10 pt, Bold, Subscript

Formatted: Centered, Adjust space between Latin and Asian text, Adjust space between Asian text and numbers

7.5.1.2.4 Receive Processing Rules

On receipt of a PDU the receiving SS or BS shall decrypt and authenticate the PDU consistent with the NIST CCM specification configured as specified in 7.5.1.2.3.

Packets that are found to be not authentic shall be discarded.

Receiving BS or SSs shall maintain a record of the highest value PN receive for each SA. If a packet is received with a PN that is equal to or less than the recorded maximum for the SA is protected under, then the packet shall be discarded as a replay attempt.

Formatted: Adjust space between Latin and Asian text, Adjust space between Asian text and numbers

Formatted: Heading 5, Adjust space between Latin and Asian text, Adjust space between Asian text and numbers

Formatted: Adjust space between Latin and Asian text, Adjust space between Asian text and numbers

7.5.2 Encryption of TEK

The following options may be used:

Deleted: .

7.5.2.1 Encryption of TEK-64 with 3-DES

This method of encrypting the TEK shall be used for SAs with the TEK encryption algorithm identifier in the cryptographic suite equal to 0x01.

The BS encrypts the value fields of the TEK in the Key Reply messages it sends to client SS. This field is encrypted using two-key 3-DES in the EDE mode [B42]:

encryption: C = Ek1[Dk2[Ek1[P]]]
decryption: P = Dk1[Ek2[Dk1[C]]]
P = Plaintext 64-bit TEK
C = Ciphertext 64-bit TEK
k1 = leftmost 64 bits of the 128-bit KEK
k2 = rightmost 64 bits of the 128-bit KEK
E[ ] = 56-bit DES ECB mode encryption
D[ ] = 56-bit DES ECB decryption

Subclause 7.5.4 below describes how the KEK is derived from the AK.

7.5.2.2 Encryption of TEK with RSA

The RSA method of encrypting the TEK (PKCS #1 v2.0, RSA Cryptography Standard, RSA Laboratories, October 1998) shall be used for SAs with the TEK encryption algorithm identifier in the cryptographic suite equal to 0x02.

7.5.2.3 Encryption of TEK-128 with AES

Formatted: Heading 4, Adjust space between Latin and Asian text, Adjust space between Asian text and numbers

This method of encrypting the TEK-128 shall be used for SAs with the TEK encryption algorithm identifier in the cryptographic suite equal to 0x03.

The BS encrypts the value fields of the TEK-128 in the Key Reply messages it sends to client SS. This field is encrypted using 128 bit AES in ECB mode.

encryption: C = Ek1[P]
decryption: P = Dk1[C]
P = Plaintext 128-bit TEK
C = Ciphertext 128-bit TEK
k1 = the 128-bit KEK
E[ ] = 128-bit AES ECB mode encryption
D[ ] = 128-bit AES ECB decryption

Deleted: ¶
Formatted: English (U.S.)
Formatted: English (U.S.)

Subclause 7.5.4 below describes how the KEK is derived from the AK.

7.5.3 Calculation of HMAC-Digests

The calculation of the keyed hash in the HMAC-Digest attribute and the HMAC Tuple shall use the HMAC (IETF RFC 2104) with the secure hash algorithm SHA-1 (FIPS 180-1). The downlink authentication key HMAC_KEY_D shall be used for authenticating messages in the downlink direction. The uplink authentication key HMAC_KEY_U shall be used for authenticating messages in the uplink direction. Uplink and downlink message authentication keys are derived from the AK (see 7.5.4 below for details). The HMAC Sequence number in the HMAC Tuple shall be equal to the AK Sequence Number of the AK from which the HMAC_KEY_x was derived.

In Mesh Mode HMAC-Digests calculated with the key HMAC_KEY_S shall be supported. When calculating the digest with this key the HMAC sequence Number in the HMAC tuple shall be equal to the Operator Shared Secret Sequence Number.

The digest shall be calculated over the entire MAC Management message with the exception of the HMAC-Digest and HMAC Tuple attributes.

7.5.4 Derivation of TEKs, KEKs, and message authentication keys

The BS generates AKs, TEKs and IVs. A random or pseudo-random number generator shall be used to generate AKs and TEKs. A random or pseudo-random number generator may also be used to generate IVs. Regardless of how they are generated, IVs shall be unpredictable. Recommended practices for generating random numbers for use within cryptographic systems are provided in IETF RFC 1750 [B30].

7.5.4.1 DES Keys

FIPS 81 defines 56-bit DES keys as 8-byte (64-bit) quantities where the seven most significant bits (i.e., seven leftmost bits) of each byte are the independent bits of a DES key, and the least significant bit (i.e., rightmost bit) of each byte is a parity bit computed on the preceding seven independent bits and adjusted so that the byte has odd parity.

PKM does not require odd parity. The PKM protocol generates and distributes 8-byte DES keys of arbitrary parity, and it requires that implementations ignore the value of the least significant bit of each.

7.5.4.2 KEKs

Formatted: Heading 4, Adjust space between Latin and Asian text, Adjust space between Asian text and numbers

7.5.4.2.1 3-DES KEKs

The keying material for two-key 3-DES consists of two distinct (single) DES keys.

The 3-DES KEK used to encrypt the TEK-64 is derived from a common AK. The KEK shall be derived as follows:

KEK=Truncate(SHA(K_PAD_KEK | AK),128)
K_PAD_KEK=0x53 repeated 64 times, i.e., a 512-bit string.

Truncate(x,*n*) denotes the result of truncating x to its leftmost *n* bits.

SHA(*x*|*y*) denotes the result of applying the SHA-1 function to the concatenated bit strings *x* and *y*.
The keying material of 3-DES consists of two distinct DES keys. The 64 most significant bits of the KEK shall be used in the encrypt operation. The 64 least significant bits shall be used in the decrypt operation.

Formatted: Heading 5, Adjust space between Latin and Asian text, Adjust space between Asian text and numbers

7.5.4.2.2 AES KEKs

The construction of the KEK for use with TEK-128 keys shall be the same as for 3-DES KEKs as described in 7.5.4.2.1 except that the full 128 bits of the KEK are used directly as the 128 bit AES key, instead of the KEK being split into two 64 bit DES keys.

[Amend clause 11.2 as follows]

Formatted: Font: 10 pt

11.2.3 TEK

Description: This attribute contains a quantity that is a TEK key, encrypted with a KEK derived from the AK.

Type	Length	Value (string)
8	8 or 16	Encrypted TEK.

When the TEK encryption algorithm identifier in the SA is 0x01, the length shall be 8 and the TEK shall be encrypted with 3DES in EDE mode according to the procedure defined in 7.5.2.1.

When the TEK encryption algorithm identifier in the SA is 0x03, the length shall be 16 and the TEK shall be encrypted with AES in ECB mode according to the procedure in 7.5.2.3

[No Change to 11.2.4 through to 11.2.7]

11.2.8 TEK parameters

Description: This attribute is a compound attribute, consisting of a collection of subattributes. These subattributes represent all security parameters relevant to a particular generation of an SAID's TEK. A summary of the TEK-Parameters attribute format is shown below.

Type	Length	Value (compound)

13	variable	The Compound field contains the subattributes as defined in Table 284

Table 284—TEK-parameters subattributes

Attribute	Contents
TEK	TEK, encrypted with the KEK
Key-Lifetime	TEK Remaining Lifetime
Key- Sequence-Number	TEK Sequence Number
CBC-IV	CBC Initialization Vector

The CBC-IV attribute is required when the data encryption algorithm identifier in the SA ciphersuite is 0x01 (DES in CBC mode).

The CBC-IV attribute is not required when the data encryption algorithm identifier in the SA ciphersuite is 0x02 (AES).

[No change to 11.2.10 through to 11.2.13]

11.2.14 Cryptographic suite

Type	Length	Value (uint8, uint8, uint8)
20	3	A 24-bit integer identifying the cryptographic suite properties. The most significant byte, as defined in Table 287, indicates the encryption algorithm and key length. The middle byte, as defined in Table 288 indicates the data authentication algorithm. The least significant byte, as defined in Table 289, indicates the TEK Encryption Algorithm.

Table 287—Data encryption algorithm identifiers

Value	Description
0	No data encryption
1	CBC-Mode, 56-bit DES
2	CCM-Mode, 128 bit AES
3-255	*Reserved*

Deleted: 2

Table 288—Data authentication algorithm identifiers

Value	Description
0	No data authentication
1	CCM-Mode, 128 bit AES
2-255	*Reserved*

Deleted: 1

Table 289—TEK encryption algorithm identifiers

Value	Description
0	*Reserved*
1	3-DES EDE with 128-bit key

2	RSA with 1024-bit key
3	ECB mode AES with 128-bit key
4-255	*Reserved*

Deleted: 3

The allowed cryptographic suites are itemized in Table 290.

Table 290—Allowed cryptographic suites

Value	Description
0x000001	No data encryption, no data authentication & 3-DES,128
0x010001	CBC-Mode 56-bit DES, no data authentication & 3-DES,128
0x000002	No data encryption, no data authentication & RSA, 1024
0x010002	CBC-Mode 56-bit DES, no data authentication & RSA, 1024
0x020103	CCM-Mode 128 bit AES, CCM-Mode, 128 bit AES, ECB mode AES with 128-bit key
All remaining values	*Reserved*

Formatted: English (U.S.)

[No change to 11.2.15]

[If this document is incorporated into 802.16e, make the following changes to 11.2.16. If this document is incorporated into 802.16d, then do not make the following changes to 11.2.16.]

11.2.16 Version

Type	Length	Value (uint8)
22	1	A 1 byte code identifying a version of PKM security as defined in Table 291.

Table 291—Version attribute values

Value	Description
0	*Reserved*
1	PKM (Initial standard release)
2	PKM (Second standard release)
3-255	*Reserved*

Deleted: 2

[12.3.1.1 ProfM3_PMP, Table 300]

Amend the 'TEK Encryption Algorithms' row

TEK encryption algorithms: 3-DES EDE with 128-bit key (type 1) RSA with 1024-bit key (type 2) ECB mode AES with 128-bit key (type 3)	Yes No No	

Deleted: