802.16 Security Enhancements

IEEE 802.16 Presentation Submission

Document Number:

IEEE C802.16d-03/60

Date Submitted:

2003-09-03

Source:

David Johnston
Intel
2111 NE 25th
Hillsboro, OR 97124

Voice: 503 264 3855
Fax: 503 202 5047
E-mail: dj.johnston@intel.com, david.johnston@ieee.org

Venue:

September 2003 802.16 Interim, Denver, CO

Base Document:

Purpose:

Description of proposed amendments to the 802.16 privacy layer for improved security.

Notice:

This document has been prepared to assist IEEE 802.16. It is offered as a basis for discussion and is not binding on 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 contribution, 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.

IEEE 802.16 Patent Policy:

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 as 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 disclose this notification via the IEEE 802.16 web site <http://ieee802.org/16/ipr/patents/notices>.

802.16 Security Enhancements

David Johnston, Intel, dj.johnston@intel.com

Jesse Walker, Intel, Jesse.walker@intel.com

Issues & options

Solution

Enhanceable Aspects of 802.16 Security

- Security In 802.16 does not meet the level of security currently demanded for other wireless systems E.G. 802.11i
  - Port Authentication is 1–way. The base authenticates the CPE. The CPE does not authenticate the base.
    - This model only works in service provider networks, where the provider controls all the equipment
  - X.509 Certs are used, derived from DOCSYS. Complex to administer and implement in mobile scenarios where the device is implemented in part in the host
    - Same comment.
  - Key establishment uses RSA (considered too compute intensive for some implementations in 802.11)
    - Either will be TOO SLOW for mobile devices, or will only work for EXPENSIVE mobile devices
  - Key exchange uses 2 key 3-DES (reasonably secure)
  - Data privacy uses DES (not reasonably secure)
  - There is no data authentication
  - There is no data replay protection

Improve the Crypto

- Use AES-128 as the basic block cipher
  - 128 bit strength
    - Common crypto algorithm for key exchange and data encryption
    - Convenient for hardware implementation
    - FIPS approvable
    - Compare DES = 56 bit (41 effective key strength)
    - Compare 2 Key 3DES = 112 bits (84 bits effective key strength)
    - Compare DES Limit of 2^32 blocks that can be sent per key

Add Data Authentication

- *Data Authentication* ***NOT*** *optional in a wireless environment!*
- Currently CBC-DES is used on data
  - Could use Ctr-AES instead
- Data encryption and authentication can be added by using **CCM** mode (Ctr+CBC_MAC)
  - Uses only the AES block cipher
  - Used in 802.11i
  - Stream Cipher, only AES encrypt needed – Fewer gates
  - Much simpler IV (sequential, self synchronizing)
- Could use Ctr-AES+HMAC-SHA1 or similar
  - Need separate keys
  - Needs two crypto algorithms (AES + HMAC-SHA1)
  - Is the more conservative approach

IVs?

7.5.4: “A random or pseudo random number generator shall be used to generate AKs and TEKs. A random or pseudorandom 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 [B10].”

- Unpredictable Unpredicatble IVs?
  - Basic rule is never use a key+IV twice, ever
  - Randomized IV => time to key,IV reuse is $\sqrt{N}$
    - But is best you can do with CBC
    - Must really be computationally indistinguishable from random, or CBC mode breaks
  - Sequential IV => time to key,IV reuse is N
    - You can do this with CCMP

Bi/Unidirectional Keys

- The same TEK is used both in the uplink and the downlink
  - This is bad. It leads to a higher chance of IV reuse in CBC and guarantees it in CCM
- Pass more TEKs
  - TEK_U_0, TEK_D_0, TEK_U_1, TEK_D_1

Authorization – Existing Style

“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.”

- Authorization is only one way
  - Must go both ways to protect user from rogue BS

Authorization – Existing style

7.1.2: “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 Authorization Key (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.”

- I must trust the factory if RSA key pair is factory installed
  - I don’t => no security!
  - This only works if provider controls all the equipment
- Must do local RSA key gen
- Must do RSA crypto every handoff
  - Keep this in mind for BS and CPE compute requirements

Authorization – 802.1X style

- Add new authorization suite
  - Use 802.1X (Implying EAPoL & EAP
  - Mandate mutual authentication (E.G. Archie)
- Maybe make new scheme mandatory for mobile equipment?
- Works with proposed inter system handoff work
  - Advanced EAP_request_indentity messages
  - 802.1x uncontrolled port handoff signalling
- More in common with other 802 standards
  - Good for residential 802.11/802.16 gateway for instance

Authorization
To EAP to not to EAP?

- Existing node authorization based on PKCS #1 and RSA
- Could do EAP
  - Similar to 802.11 and 802.3 methods
  - Is a Client-Server model. OK for fixed. **Broken for Mesh**
  - Good for inter 802 handoff, since 802.1X available
- Or Could amend current method
  - Add network side certs
  - Public keys of networks must be published widely
  - May have to provide special procedures to support inter 802 handoff in place of 802.1x for 802.16e
    - Rules about forwarding certain snap frames based on SA
  - This is preferred method, due to mesh issue

PKM Extensions

- To enable CPE to authenticate BS
  - BS must provide credential to CPE
  - CPE must be able to verify the credential
    - By CPE maintaining local store of credential data
    - Or by BS permitting tunneling of CPE authentication exchanges with a certificate authority or authentication proxy
  - The credential could be a cert, a private key encrypted credential, a shared secret etc.
- Retain master-slave property to avoid race conditions

Hard Credentials

- Public/Private Key SIM/Smartcard
  - Provided to CPE users by network vendor
  - Contains public keys of network vendor and maybe roaming partners
  - Contains CPE public/private key pair
    - Either installed by network provider
    - Or generated in CPE during an enrollment process
- Removes undesirable manufacturing stage
- Limits user by the number of SC slots
- Smart Card could be blank and all data gets programmed during enrollment stage
  - Simplifies provision of multi-provider/multi-identity cards

Soft Credentials

- User free to gather public key information from public sources
  - A healthy PKI would help here
- Enrollment generally required to furnish provider with user credentials
- Implementation left to the manufacturer

Issues & Options

Solution

Some Details

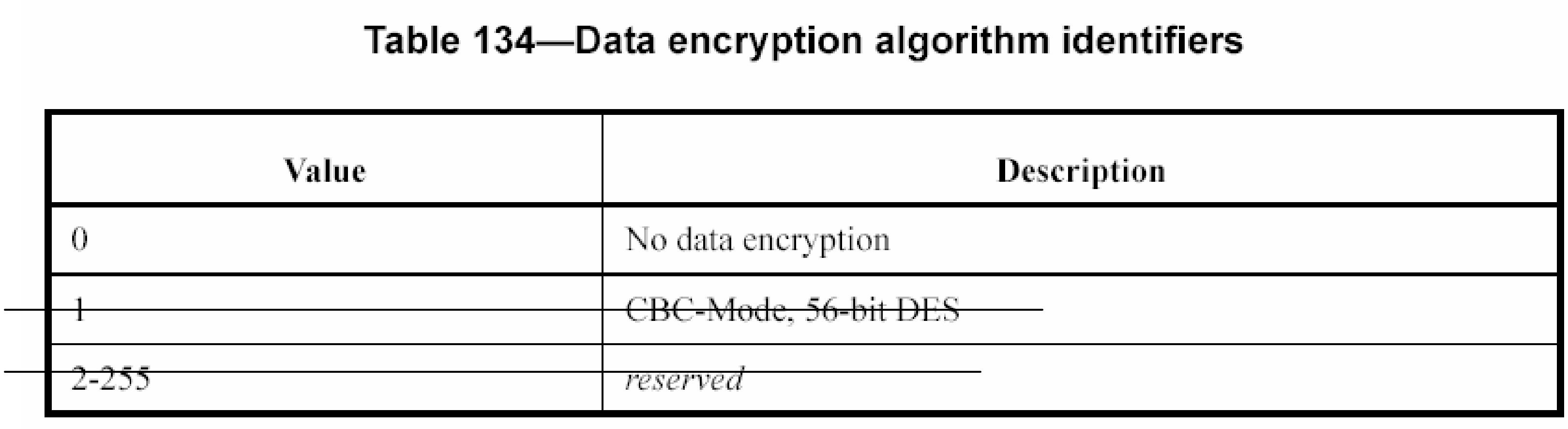


Table 134—Data encryption algorithm identifiers

Value	Description
0	No data encryption
~~1~~	~~CBC-Mode, 56-bit DES~~
~~2-255~~	~~*reserved*~~

2 : CCM Mode, AES-128

3-255 : *reserved*

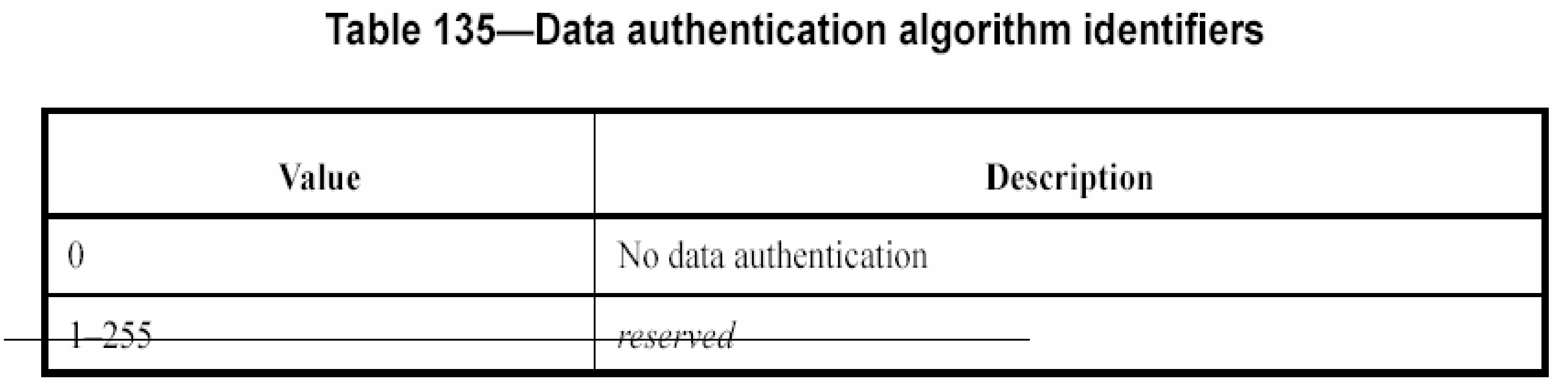


Table 135—Data authentication algorithm identifiers

Value	Description
0	No data authentication
~~1–255~~	~~*reserved*~~

1 : CCM Mode, AES-128

2-255 : *reserved*

Some Details

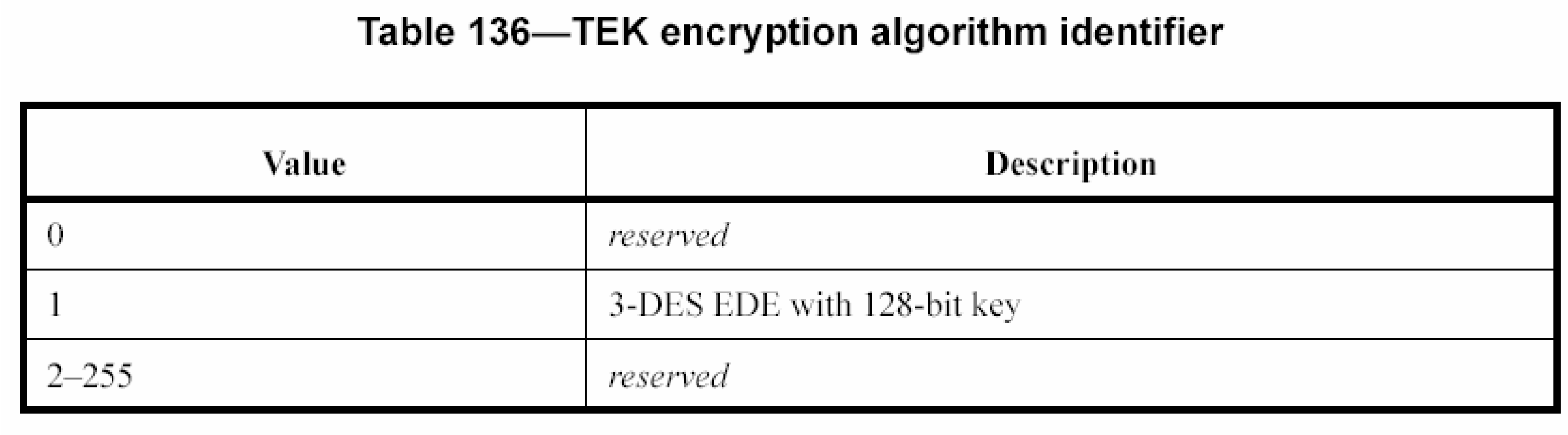


Table 136—TEK encryption algorithm identifier

Value	Description
0	*reserved*
1	3-DES EDE with 128-bit key
2–255	*reserved*

2 : AES-128

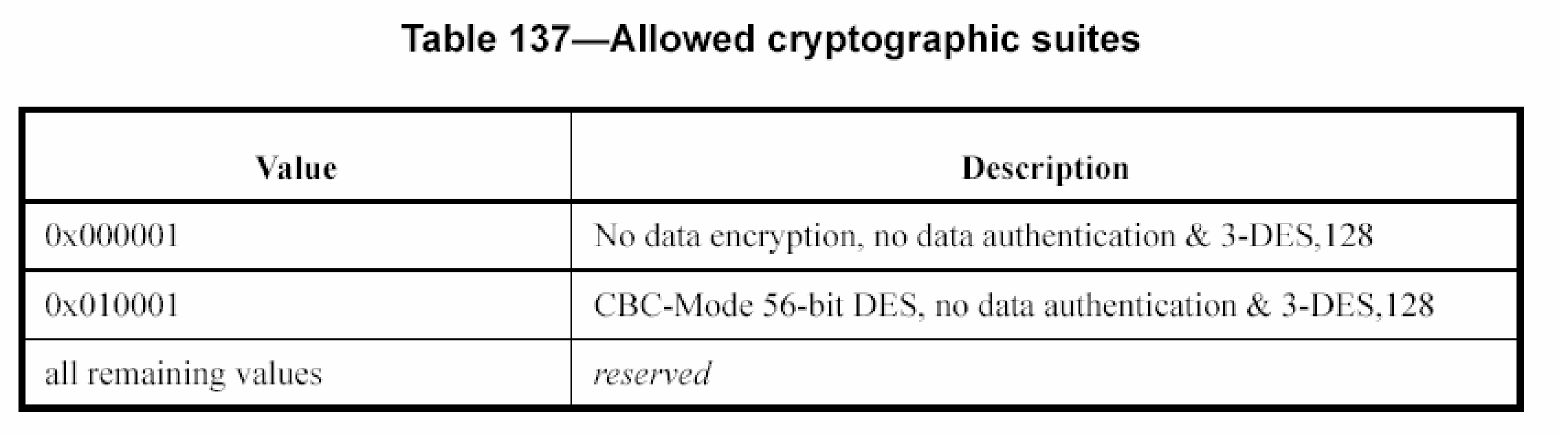


Table 137—Allowed cryptographic suites

Value	Description
0x000001	No data encryption, no data authentication & 3-DES,128
0x010001	CBC-Mode 56-bit DES, no data authentication & 3-DES,128
all remaining values	*reserved*

0x020002 : CCMP-AES-128, CCMP-AES-128, AES-128

Some Details

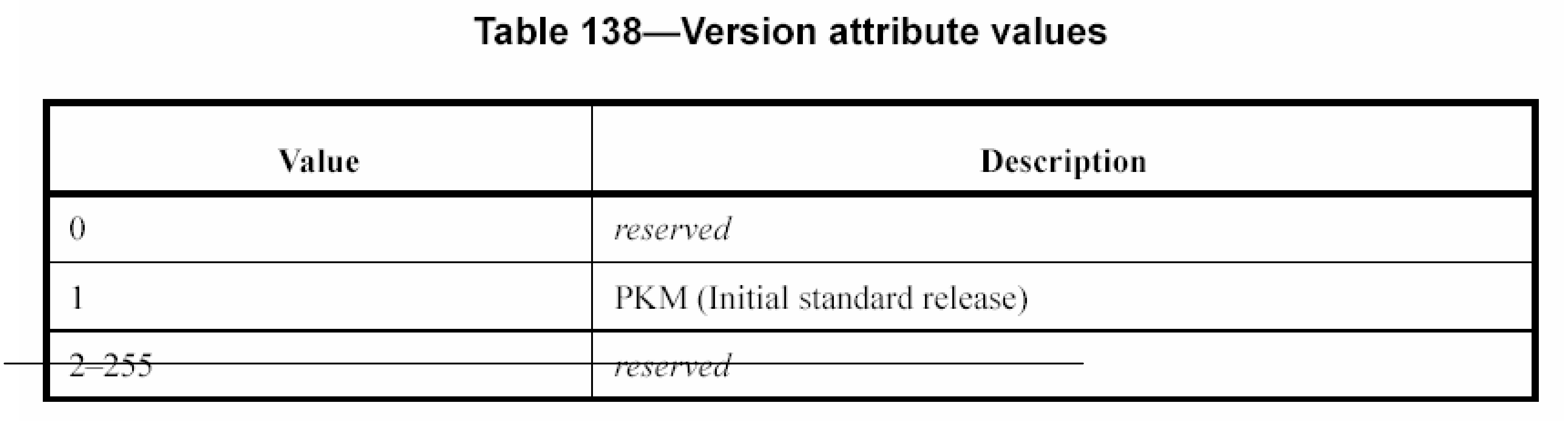


Table 138—Version attribute values

Value	Description
0	*reserved*
1	PKM (Initial standard release)
~~2–255~~	~~*reserved*~~

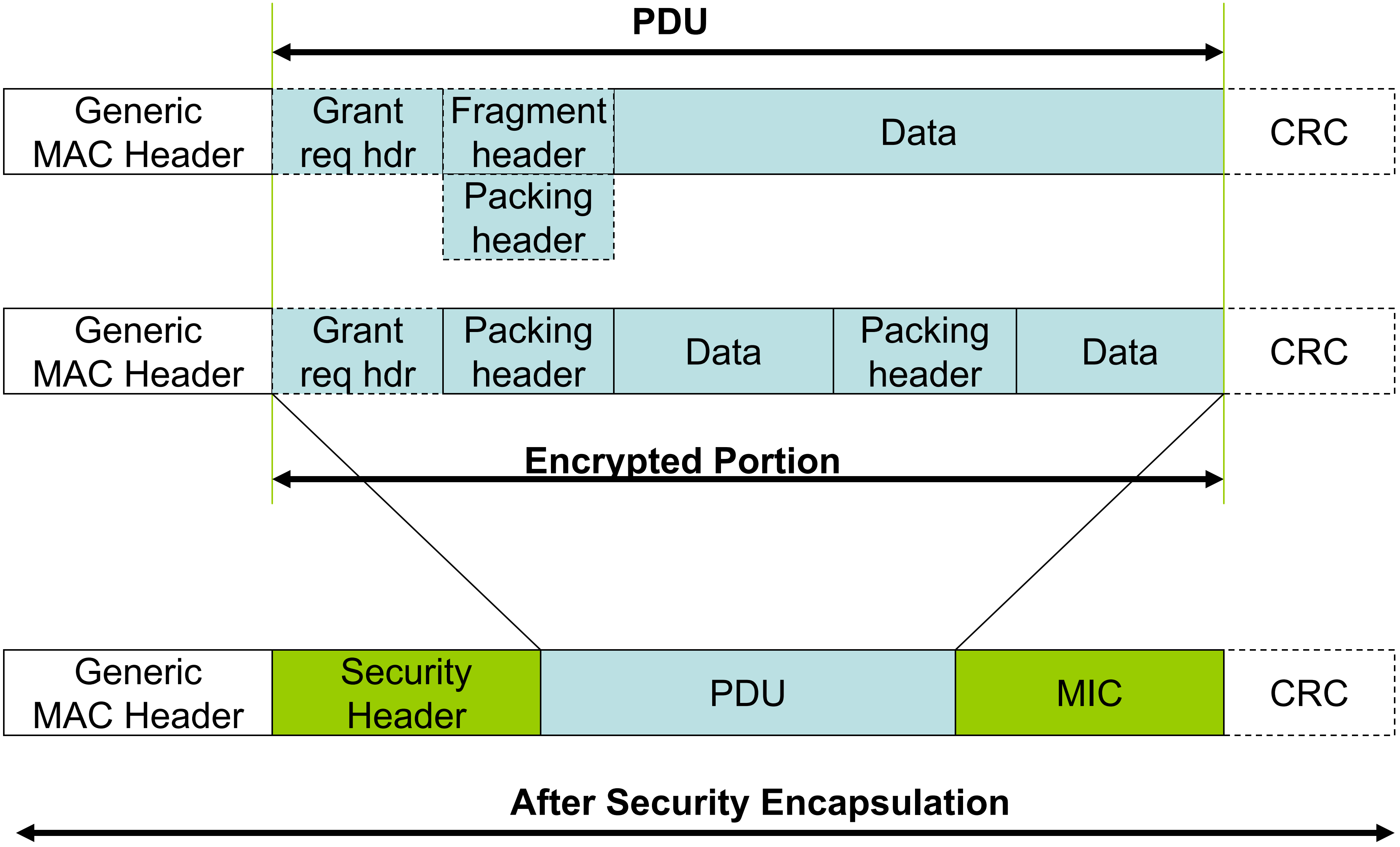
2 : PKM (Revised standard release)

3 : Open Authorization

4-255 : *reserved*

Open Authorization gives implicit 802.16 layer authorization to use the link. This is a suitable selection where 802.1X is used in place of PKM authorization.
Key exchange is not defined in 802.1X and is a work item for 802.1. Defining a ‘key exchange only’ PKM could run into conflict with 802.1 in the future. I suggest leaving ‘Open Authorization’ in as a place holder for 802.1X based authorization until key exchange is defined

CCM Mode (0x20002)



PDU

Generic MAC Header

Grant req hdr

Fragment header

Packing header

Data

CRC

Generic MAC Header

Grant req hdr

Packing header

Data

Packing header

Data

CRC

Encrypted Portion

Generic MAC Header

Security Header

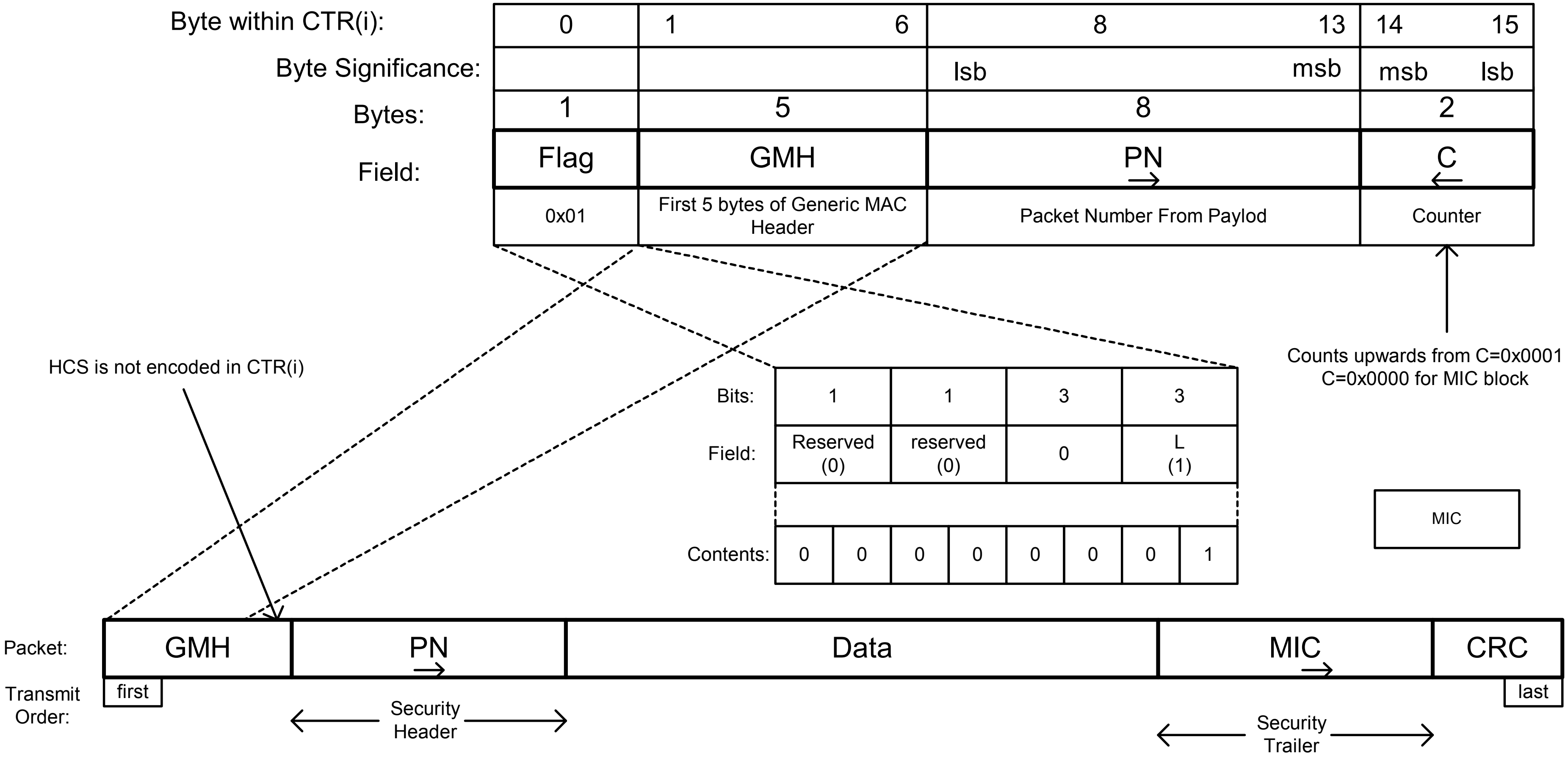
PDU

MIC

CRC

After Security Encapsulation

CCM Mode – CTR(i)



Byte within CTR(i):	0	1 6	8 13	14 15
Byte Significance:			lsb msb	msb lsb
Bytes:	1	5	8	2
Field:	Flag	GMH	PN →	C ←
	0x01	First 5 bytes of Generic MAC Header	Packet Number From Paylod	Counter

Counts upwards from C=0x0001
C=0x0000 for MIC block

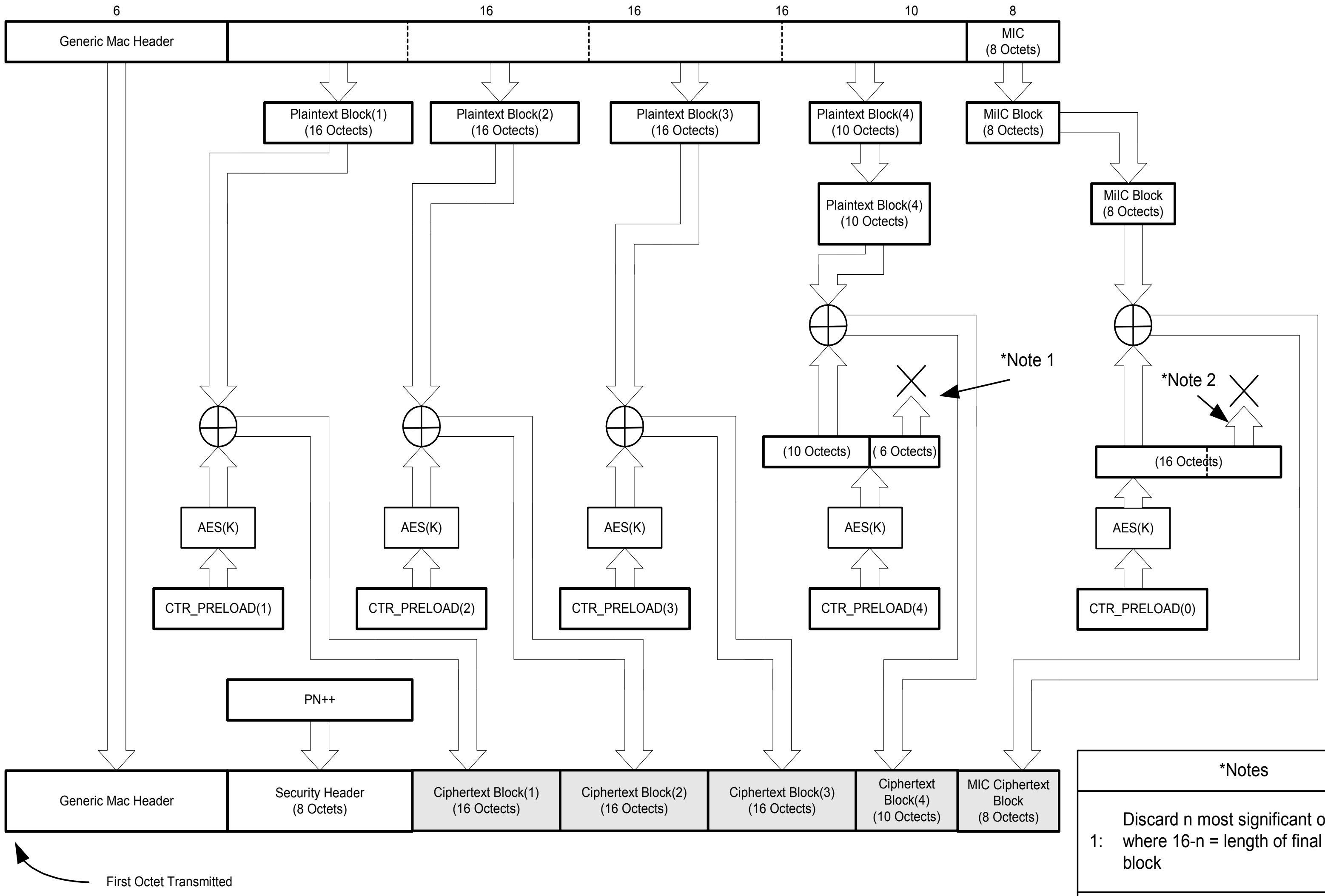
Bits:	1	1	3	3
Field:	Reserved (0)	reserved (0)	0	L (1)

Contents:	0	0	0	0	0	0	0	1

Packet:	GMH	PN →	Data	MIC →	CRC
Transmit Order:	first				last

Byte significance:
least significant first →
most significant first ←

CCM Mode Encrypt (reverse for decrypt)



6
16
16
16
10
8
Generic Mac Header
MIC (8 Octets)
Plaintext Block(1) (16 Octects)
Plaintext Block(2) (16 Octects)
Plaintext Block(3) (16 Octects)
Plaintext Block(4) (10 Octects)
MiIC Block (8 Octects)
Plaintext Block(4) (10 Octects)
MiIC Block (8 Octects)
*Note 1
*Note 2
(10 Octects)
(6 Octects)
(16 Octects)
AES(K)
AES(K)
AES(K)
AES(K)
AES(K)
CTR_PRELOAD(1)
CTR_PRELOAD(2)
CTR_PRELOAD(3)
CTR_PRELOAD(4)
CTR_PRELOAD(0)
PN++
Generic Mac Header
Security Header (8 Octets)
Ciphertext Block(1) (16 Octects)
Ciphertext Block(2) (16 Octects)
Ciphertext Block(3) (16 Octects)
Ciphertext Block(4) (10 Octects)
MIC Ciphertext Block (8 Octects)
First Octet Transmitted

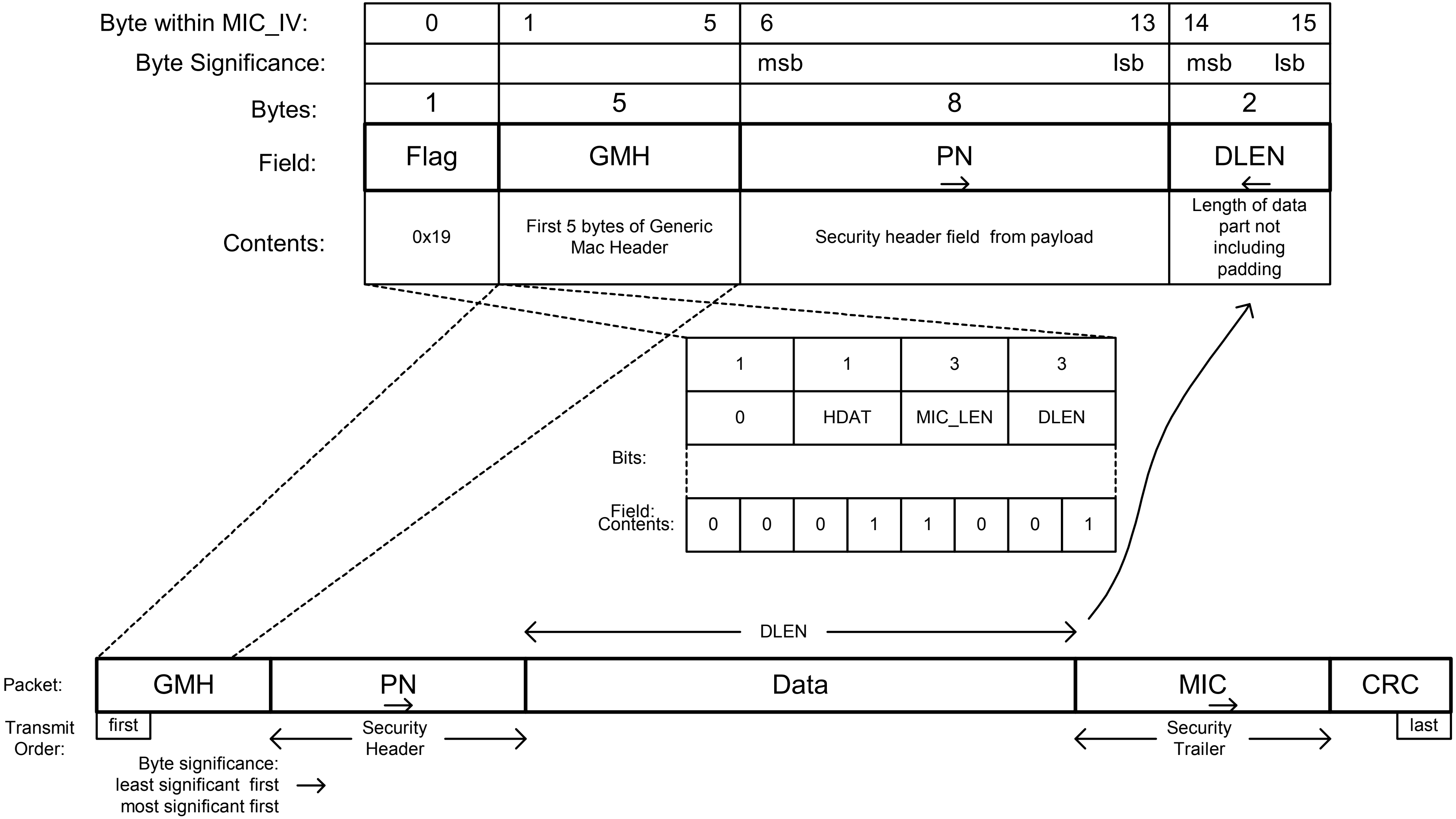
Key: xyz 16 octet (or fewer) data field; AES(K) AES block cipher, using 128 bit key K; ⊕ Bitwise XOR; ▨ xyz Encrypted Field

*Notes

1: Discard n most significant octets where 16-n = length of final plaintext block

2: Discard 8 most significant octets

CCM Mode – MIC IV



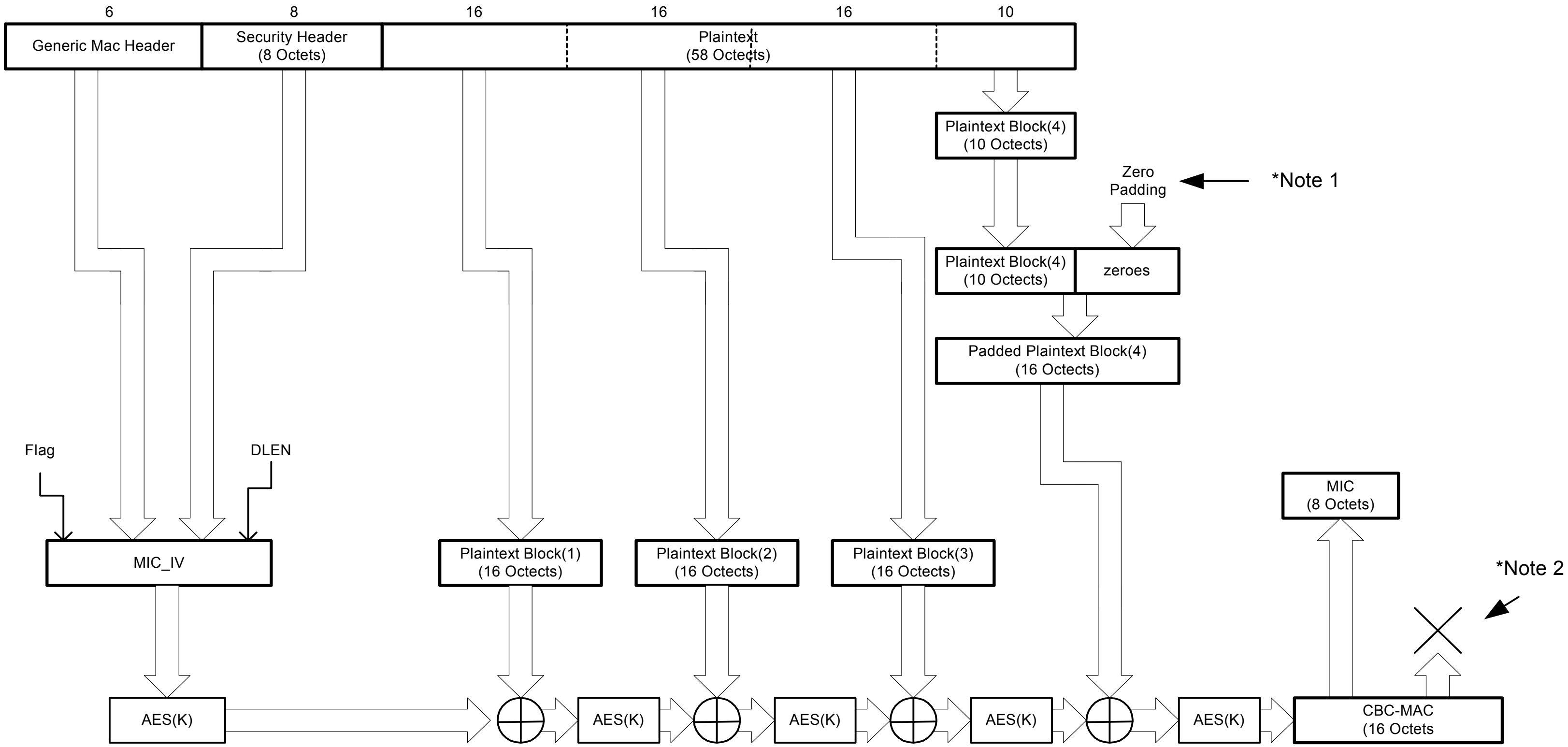
Byte within MIC_IV:	0	1 5	6 13	14 15
Byte Significance:			msb lsb	msb lsb
Bytes:	1	5	8	2
Field:	Flag	GMH	PN →	DLEN ←
Contents:	0x19	First 5 bytes of Generic Mac Header	Security header field from payload	Length of data part not including padding

	1		1		3		3	
	0		HDAT		MIC_LEN		DLEN	
Bits:								
Field: Contents:	0	0	0	1	1	0	0	1

Packet:	GMH	PN →	Data	MIC →	CRC
Transmit Order:	first				last

Byte significance:
least significant first →
most significant first

CCM Mode MIC



Key:

xyz	16 octet (or fewer) data field
AES(K)	AES block cipher, using 128 bit key K
⊕	Bitwise XOR

***Notes**

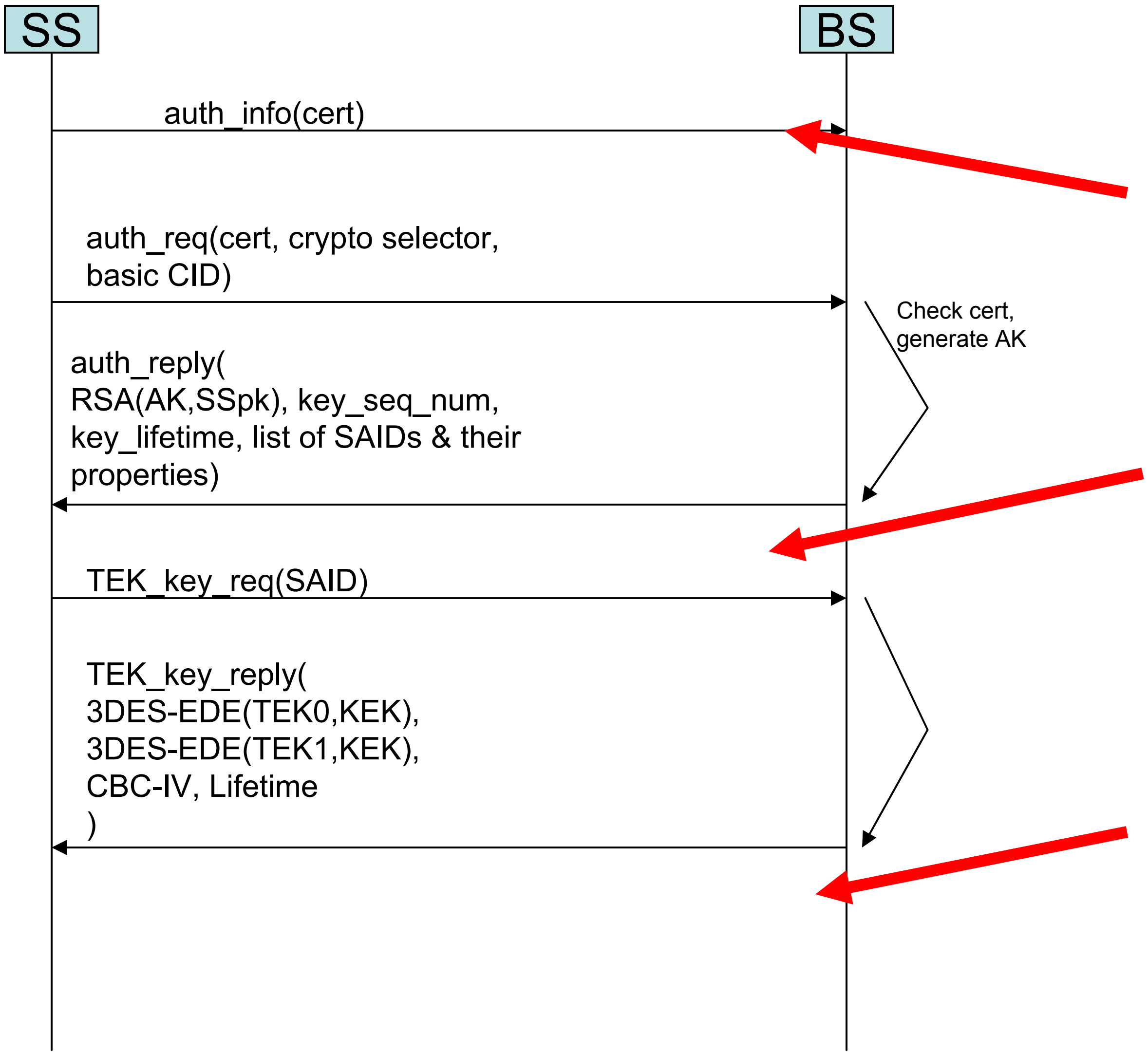
1: Pad n zeroes to most signifiant end of field such that:
(field length + n) = 16

2: Discard most significant 8 octets

PKM

Current: 1 Way Authentication

New Method: 2 Way Authentication



SS

BS

auth_info(cert)

auth_req(cert, crypto selector,
basic CID)

Check cert,
generate AK

auth_reply(
RSA(AK,SSpk), key_seq_num,
key_lifetime, list of SAIDs & their
properties)

TEK_key_req(SAID)

TEK_key_reply(
3DES-EDE(TEK0,KEK),
3DES-EDE(TEK1,KEK),
CBC-IV, Lifetime
)

Enhance startup info?

Introduce new authentication sequence to allow SS to authenticate BS?

Introduce separate uplink and downlink keys

802 Handoff/802.1X

- Provide new messages to substitute for the controlled and uncontrolled port messaging of 802 Handoff
  - Is just a simple container
  - Internal format defined by 802 Handoff
  - Security state must be provided equivalent to the 802.1X port open/closed state
- Beats the alternative
  - Implementing 802.1X

Backup

- Rekeying speed
  - Rekey before 2^[32,48,64] packets
    - 70 Mbps
    - Smallest packet 1 octet + 6 octet GMH + [4,6,8] octet PN + 8 octet MIC (No sub header, no CRC)
      - [19, 21, 23] octets, [152,168,184] bits
      - Ignore other headers, multiple connections, ul/dl, idle time (most pessimistic/fastest rekeying)
    - $(70*10^6)$/[152, 168, 184 = [460520, 416670, 380430] packets/sec
    - 2^{32}/460520 = 9326s = **2 hours, 35 minutes**
    - 2^{48}/416670 = **21.42 Years**
    - 2^{64}/380430 = **1.54 Million Years**
- Is it better to have frequent rekeying?
  - Less time for an attacker to compromise the key
  - More bandwidth & entropy resource used in the signaling