IEEE P802.11  
Wireless LANs

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Resolution Document for Clause 12 | | | | |
| Date: 2021-09-15 | | | | |
| Author(s): | | | | |
| Name | Affiliation | Address | Phone | email |
| Hitoshi Morioka | SRC Software | Fukuoka, JAPAN |  | hmorioka@src-soft.com |
|  |  |  |  |  |

Abstract

This document describes the resolutions for LB252.

**The baseline is D1.04.**

# Suggested resolution

***TGbc editor: please change clause 2 as follows:***

## 2. Normative references [1164, 1104]

IETF RFC 4082, Timed Efficient Stream Loss-Tolerant Authentication (TESLA): Multicast Source Authentication Transform Introduction, June 2005.

IETF RFC 6979, Deterministic Usage of the Digital Signature Algorithm (DSA) and Elliptic Curve Digital Signature Algorithm (ECDSA), August 2013.

IETF RFC 8017, PKCS #1: RSA Cryptography Specifications Version 2.2, November 2016.

IETF RFC 8032, Edwards-Curve Digital Signature Algorithm (EdDSA), January 2017.

~~FIPS PUB 202, SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions.~~

~~FIPS PUB 186-5 (Draft), Digital Signature Standard (DSS), October 2019.~~

~~NIST Special Publication 800-185, SHA-3 Derived Functions: cSHAKE, KMAC, TupleHash and ParallelHash.~~

~~Draft NIST Special Publication 800-186, Recommendations for Discrete Logarithm-Based Cryptography: Elliptic Curve Domain Parameters, October 2019.~~

### 5.1.4 MSDU format

***TGbc editor: please change the list as follows:* [1095]**

f) If the transmitting mesh STA peers only with an EPD mesh STA, use EPD.

g) For EBCS, use EPD.

~~g~~h) For all other cases, use LPD.

### 9.6.7.49 EBCS UL frame format

***TGbc editor: please change Table 9-397a as follows:***

Table 9-397a—Encoding of Frame Signature Type subfield [1103, 1520, 1638]

|  |  |  |
| --- | --- | --- |
| **~~Value~~** | **~~Algorithm~~** | **~~Encoding~~** |
| ~~0~~ | ~~HLSA~~ | ~~The authentication of the HLP payload is provided by a higher layer and is~~  ~~included in the HLP Payload field~~ |
| ~~1~~ | ~~RSA-2048~~ | ~~See 12.13.2.6 (Signature of the EBCS UL frame)~~ |
| ~~2~~ | ~~ECDSA-P256~~ |
| ~~3~~ | ~~Ed25519~~ |
| ~~4-7~~ | ~~Reserved~~ |

|  |  |  |  |
| --- | --- | --- | --- |
| **Value** | **Algorithm** | **Frame Signature Length**  **(octets)** | **Encoding** |
| 0 | HLSA | — | The authentication of the HLP payload is provided by a higher layer and is  included in the HLP Payload field |
| 1 | RSASSA-PSS-2048 | 256 | See 12.13.2.6 (Signature of the EBCS UL frame) |
| 2 | RSASSA-PSS-4096 | 512 |
| 3 | ECDSA-P256 | variable |
| 4 | ECDSA-P521 | variable |
| 5 | Ed25519 | 64 |
| 6-7 | Reserved | — | — |

***TGbc editor: please add the following sentence at the end of clause 9.6.7.49:***

The length of the Frame Signature field is determined by the value of Frame Signature Type subfield as shown in Table 9-397a (Encoding of Frame Signature Type subfield). In case of ECDSA, the Signature field contains the value r and s encoded in DER-encoded ASN.1 structure (a SEQUENCE of two INTEGERs, for r and s, in that order) that includes the length. [1520, 1638]

### 9.6.7.50 EBCS Info frame format

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | ~~Category~~ | ~~Public Action~~ | ~~EBCS Info Sequence Number~~ | ~~EBCS Info Timestamp~~ | ~~EBCS Info Control~~ | ~~EBCS Info Interval~~ | ~~Fragment Hash Values~~ | ~~Ceritificate Length~~ |
| ~~Octets:~~ | ~~1~~ | ~~1~~ | ~~4~~ | ~~8~~ | ~~1~~ | ~~1~~ | ~~n x 32~~ | ~~0 or 2~~ |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | ~~Certificate~~ | ~~Content Information Number~~ | ~~Content Information List~~ | ~~Signature~~ |
| ~~Octets:~~ | ~~variable~~ | ~~1~~ | ~~variable~~ | ~~variable~~ |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | Category | Public Action | EBCS Info Sequence Number | EBCS Info Timestamp | EBCS Info Control | EBCS Info Authentication Algorithm | EBCS Info Interval |
| Octets: | 1 | 1 | 4 | 8 | 1 | 1 | 1 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Fragment Hash Values | Certificate Length | Certificate | Content Information Number | Content Information List | Signature |
| Octets: | n x 32 | 0 or 2 | variable | 1 | variable | variable |

Figure 9-909g—EBCS Info Action field format [1103]

***TGbc editor: please change the paragraph at P41L19 as follows:***

The EBCS Info Authentication Algorithm ~~sub~~field indicates the algorithm used to authenticate the EBCS Info frame. Values of this ~~sub~~field are defined in Table 9-397b (EBCS Info Authentication Algorithm ~~sub~~field). [1103]

***TGbc editor: please change Table 9-397b as follows:***

Table 9-397b—EBCS Info Authentication Algorithm ~~sub~~field [1102, 1103, 1520, 1638, 1642]

|  |  |  |
| --- | --- | --- |
| **~~Value~~** | **~~Algorithm~~** | **~~Certificate Present~~** |
| ~~0~~ | ~~None~~ | ~~No~~ |
| ~~1~~ | ~~RSASSA-PSS~~ | ~~Yes~~ |
| ~~2~~ | ~~ECDSA~~ | ~~Yes~~ |
| ~~3~~ | ~~Ed25519~~ | ~~Yes~~ |

|  |  |  |  |
| --- | --- | --- | --- |
| **Value** | **Algorithm** | **Certificate Present** | **Signature Length**  **(octets)** |
| 0 | None | No | — |
| 1 | Pre-negotiated | No | variable |
| 2 | RSASSA-PSS-2048 | Yes | 256 |
| 3 | RSASSA-PSS-4096 | Yes | 512 |
| 4 | ECDSA P-256 | Yes | variable |
| 5 | ECDSA P-521 | Yes | variable |
| 6 | Ed25519 | Yes | 64 |
| 7-127 | Reserved | — | — |

***TGbc editor: please change the last paragraph in clause 9.6.7.50 as follows:***

The Signature field is the digital signature of the EBCS Info frame (see 12.13.2.1 (Signature of the EBCS Info Frame)). The length of the Signature field is determined by the ~~algorithm used to authenticate the EBCS Info frame~~ value of the EBCS Info Authentication Algorithm field as shown in Table 9-937b (EBCS Info Authentication Algorithm subfield). In case of ECDSA, the Signature field contains the value r and s encoded in DER-encoded ASN.1 structure (a SEQUENCE of two INTEGERs, for r and s, in that order) that includes the length. [1520, 1638, 1642] In case of Pre-negotiated, the length of the Signature field is pre-negotiated between the EBCS AP and the EBCS receiver. [1102] The Signature field is not present if the Fragment Index subfield is nonzero.

***TGbc editor: please change clause 12.13 as follows:***

## 12.13 Frame authentication for EBCS

### 12.13.1 General

EBCS provides one-way frame authentication mechanisms that do not require key negotiation between a transmitter and receivers.

EBCS DL cases use one of the following four types of authentication.

- Public Key Frame Authentication (PKFA)

- Hash Chain Frame Authentication (HCFA) with Instant Authentication

- Hash Chain Frame Authentication (HCFA) without Instant Authentication

- Higher layer source authentication (HLSA); no frame authentication

EBCS UL cases use PKFA or HLSA.

While PKFA is suitable for occasional small data transfer or time sensitive data transfer, HCFA is suitable for continuous content distribution such as live streaming or periodic file transfer.

NOTE—PKFA has a higher computational cost than HCFA. HCFA ~~requires~~ generates latency for buffering. The characteristics of HLSA depend on higher layer algorithms. [1398, 1204]

The following preparations are required for both PKFA and HCFA before starting an EBCS content stream.

- The certificate(s) of the CA(s) (Certificate Authority) shall be installed into the EBCS receivers. Note that the certificate of the CA(s) is installed with an application like a content browser. The installation method is out of scope of this standard.

- The EBCS transmitter generates its own private key and public key pair. The public key shall be signed by one of the CAs for which the EBCS receiver(s) have the certificate.

Preparations for HLSA depend on higher layer algorithms and are out of scope of this standard.

### 12.13.2 EBCS public key frame authentication (PKFA)

### 12.13.2.1 Signature of an EBCS Info frame

One of the following public key algorithms is used.

• RSASSA-PSS (~~FIPS PUB 186-5~~ IETF RFC 8017) [1164]

• ECDSA (~~FIPS PUB 186-5~~ IETF RFC 6979) [1164]

• Ed25519 (~~FIPS PUB 186-5~~ IETF RFC 8032) [1164]

For RSASSA-PSS, the length of the modulus shall be 2048 bits or 4096 bits. ~~and SHAKE128 (FIPS 202)~~ SHA-256 for 2048 bits modulus or SHA-512 for 4096 bits modulus shall be used as a mask generation function ~~and~~ as a hash function during signature generation. ~~The output length of SHAKE128 shall be 256 bits.~~ [1103, 1104]

For ECDSA, the domain parameters shall be P-256 ~~(NIST SP 800-186)~~ [1164] or P-521. ~~and SHAKE128~~ SHA-256 for P-256 or SHA-512 for P-521 shall be used as a hash function during signature generation. ~~The output length of SHAKE128 shall be 256 bits.~~ [1103, 1104]

For Ed25519, SHA-512 shall be used as a hash function during signature generation.

The EBCS transmitter generates an EBCS Info frame periodically as described in 11.55.2.4 (EBCS Info frame generation and usage).

If the length of the EBCS Info frame is larger than the maximum MMPDU length (Table 9-25 Maximum data unit sizes (in octets) and durations (in microseconds)), the EBCS Info frame shall be fragmented as described in 11.55.2.5 (EBCS Info frame fragmentation).

If the EBCS Info frame is not fragmented, the EBCS transmitter shall fill all the fields according to9.6.7.100 (EBCS Info frame format) except the signature.

The EBCS transmitter shall generate the signature as follows:

Signature = Sign(~~The EBCS~~ transmitter’s private key, ~~The EBCS~~ transmitter’s MAC address || EBCS Info frame fields from the beginning of the EBCS Info Sequence Number field to the end of the ~~last~~ Content Information List field ~~in the EBCS Info frame~~) [1205, 1311]

where

Sign(k,m) indicates a digital signature for the message m using the private key k.

Otherwise, only the first fragment contains the signature.

Signature = Sign(~~The EBCS~~ transmitter’s private key, ~~T~~transmitter’s MAC address || EBCS Info frame fields from the beginning of the EBCS Info Sequence Number field to the end of the Frame Body field ~~the first fragment~~) [1205, 1311]

And compute the hash value(s) for the following fragment(s).

HashValue = ~~SHAKE128~~ SHA-256 [1104] (~~T~~transmitter’s MAC address || EBCS Info frame fields from the beginning of the EBCS Info Sequence Number field to the ~~last~~ end of the Frame Body field ~~of the fragment~~) [1205, 1311]

The EBCS transmitter fills the Signature field and the Fragment Hash Values field in EBCS Info frame with the computed signature and hash values respectively as described in 11.55.2.4 (EBCS Info frame generation and usage).

### 12.13.2.2 Authentication of an EBCS Info frame

When the EBCS receiver receives an EBCS Info frame of which EBCS Info Authentication Algorithm field does not indicate None nor Pre-negotiated, the EBCS receiver shall authenticate it as follows: [1102]

~~1.~~ If the EBCS Info frame is fragmented, the following procedures are applied only to the first fragment. [1206, 1107]

~~2~~1. If the difference between the timestamp in the EBCS Info frame and the time of the clock of the EBCS receiver is greater than the allowable time difference in the EBCS Info frame, the EBCS Info frame shall be discarded. [1206, 1107]

~~3~~2. Verify the certificate of the AP in the EBCS Info frame with the installed certificate of the CA. If the verification fails or the certificate of the CA that signed the certificate of the AP in the EBCS Info frame is not installed, the EBCS Info frame shall be discarded. [1206, 1107]

~~4~~3. Verify the signature in the EBCS Info frame with the certificate of the AP in the EBCS Info frame. If the verification fails, the EBCS Info frame shall be discarded. [1206, 1107]

When the EBCS receiver receives an EBCS Info frame of which EBCS Info Authentication Algorithm field indicates Pre-negotiated, the EBCS receiver shall authenticate it as follows: [1102]

If the EBCS Info frame is fragmented, the following procedures are applied only to the first fragment. [1102]

1. If the difference between the timestamp in the EBCS Info frame and the time of the clock of the EBCS receiver is greater than the allowable time difference in the EBCS Info frame, the EBCS Info frame shall be discarded. [1102]

2. Verify the signature in the EBCS Info frame with the pre-negotiated public key. If the verification fails, the EBCS Info frame shall be discarded. [1102]

~~5.~~ If the EBCS Info frame is fragmented, the EBCS receiver caches the hash value(s) of the fragment(s) and the EBCS receiver shall authenticate the following fragment(s) as follows: [1206, 1107]

1. Compute the hash value of the fragment as described ~~in the formula (12-bc1)~~ [1310] in 12.13.2.1 (Signature of an EBCS Info Frame).

2. If the computed hash value is equal to the cached hash value, the authentication succeeds. Otherwise, the fragment shall be discarded.

~~3. If the authentication succeeds, the EBCS receiver defragments the EBCS Info frame as described in 11.55.2.6 (EBCS Info frame defragmentation).~~

3. If all the framgments successfully authenticated, the EBCS receiver shall concatenates them and shall process the defragmented EBCS Info frame as described in 11.55.2.4 (EBCS Info frame generation and usage). [1315]

If the authentication succeeds:

• The EBCS receiver caches the certificate of the AP if exists [1102], and the allowable time difference in the EBCS Info frame.

• If data is present in the Content Information field, the EBCS receiver processes the data in the Content Information field(s) in accordance with 11.55.2.4 (EBCS Info frame generation and usage).

### 12.13.2.3 PKFA MPDU format

EBCS Data frames sent under PKFA use the PKFA MPDU format. [1419]

The ~~MPDU~~ format ~~for~~ of a PKFA MPDU is shown in Figure 12-55a (PKFA MPDU format). [1419]

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Mac Header | ~~Timestamp~~  Content ID [w/o CID] | ~~Content ID~~  Timestamp [w/o CID] | Data Sequence | Data Length | Data (PDU) | Signature | FCS |
| Octets | ~~8~~1  [w/o CID] | ~~1~~7  [1095] | ~~4~~2 [1095] | 2 | >= 1 ~~or more~~ [w/o CID] | Variable | 4 |

Figure 12-55a PKFA MPDU format

The Content ID field contains the content ID of the EBCS traffic stream. [w/o CID]

The Timestamp field contains the elapsed time from 2020-01-01 00:00 UTC in ~~milliseconds~~ microseconds [1095] that indicates the time of the MPDU generation.

~~The Content ID field contains the content ID of the MPDU.~~ [w/o CID]

The Data Sequence field indicates the sequence number of the frame that starts from 0 and is incremented by 1 for each MPDU generation. In case of overflow, it is reset to 0.

The Data Length field indicates the length of the Data field.

The Data (PDU) field contains ~~the HLP packet~~ an MSDU . [1095, 1313]

The Signature field contains the signature for the MPDU. The signature is generated by the following formula.

Signature = Sign(~~The EBCS~~ transmitter’s private key, ~~T~~transmitter’s MAC address || PKFA MPDU fields ~~F~~from ~~Timestamp~~ the Content ID field to the end of the Data (PDU) field) [1311]

where

Sign(k, m) indicates a digital signature for the message m using the private key k.

### 12.13.2.4 Authentication of a PKFA MPDU frame

When an EBCS receiver receives an EBCS Data frame, the EBCS receiver shall authenticate it as follows:

1. If the difference between the timestamp in the PKFA MPDU and the time of the clock is greater than the cached allowable time difference, the PKFA MPDU shall be discarded.

2. Verify the signature in the PKFA MPDU using the cached certificate or the pre-negotiated public key [1102]. If 1 the verification fails, the PKFA MPDU shall be discarded, otherwise decapsulate the PDU.

### 12.13.2.5 Signature of the EBCS UL frame

One of the following public key algorithms is used.

• RSASSA-PSS

• ECDSA

• Ed25519

For RSASSA-PSS, the length of the modulus shall be 2048 bits or 4096 bits. ~~and SHAKE128~~ SHA-256 for 2048 bits modulus or SHA-512 for 4096 bits modulus shall be used as a mask generation function ~~and~~ as a hash function during signature generation. ~~The output length of SHAKE128 shall be 256 bits.~~ [1103, 1104]

For ECDSA, the domain parameters shall be P-256 or P-521. ~~and SHAKE128~~ SHA-256 for P-256 or SHA-512 for P-521 shall be used as a hash function during signature generation. ~~The output length of SHAKE128 shall be 256 bits.~~ [1103, 1104]

For Ed25519, SHA-512 shall be used as a hash function during signature generation.

The EBCS transmitter generates an EBCS UL frame when it receives data to be transmitted. The format of the EBCS UL frame is described in 9.6.7.100 (EBCS UL frame format).

The signature is generated as follows:

Signature = Sign(~~The EBCS~~ transmitter’s private key, ~~T~~transmitter’s MAC address || EBCS UL frame fields from the beginning of the EBCS Info Sequence Number field to the last field before the Frame Signature Length field ~~in the EBCS UL frame~~) [1205, 1311]

where

Sign (k, m) indicates a digital signature for the message m using the private key k.

Then the EBCS transmitter transmits the EBCS UL frame.

### 12.13.2.6 Authentication of an EBCS UL frame

An EBCS proxy shall discard an EBCS UL frame if any of the following conditions are met:

1. The STA Certificate subfield is present and any of the following conditions are met:

a. The certificate of the authority (i.e., specified destination or the CA that signed the STA’s certificate) is not installed.

b. The verification of the STA’s certificate using the installed certificate fails.

c. The Frame Signature Type subfield does not indicate HLSA and the verification of the signature of the frame using the STA’s certificate fails.

d. The Frame Signature Type subfield does not indicate HLSA and the verification of the signature of the frame using the STA’s certificate has passed and the Frame Count field is present, and the value is less than or equal to the last seen Frame Count (if any) from the EBCS non-AP STA.

2. The Frame Tx Time field is present and the difference between the value carried in the field and the time when the EBCS UL frame is received is greater than an acceptable value.

NOTE 1 —The acceptable time difference at an EBCS proxy can be selected based on local policies or based on the relationship with the specified destination. The selection of an acceptable time difference is out of scope of this standard.

NOTE 2 —When the Frame Signature Type subfield indicates HLSA, replay protection is 1 performed by a higher layer and is out of scope of this standard.

An EBCS proxy shall save the value in the most recently received Frame Count field of an EBCS UL frame, as last seen Frame Count, for a certain transmitter only if the EBCS UL frame was not discarded based on the conditions above. In addition, an EBCS proxy may have an expiration time after which the last seen Frame Count value for a certain transmitter is discarded.

### 12.13.3 EBCS hash chain frame authentication (HCFA)

### 12.13.3.1 General

HCFA uses a digital signature and ~~a modified~~ the algorithm of [1108] TESLA (Timed Efficient Stream Loss-Tolerant Authentication, IETF RFC 4082).

HCFA provides the following two authentication methods to authenticate each HCFA MPDU.

- HCFA authentication

- Instant authentication (optional)

Each authentication method uses a separate authenticator. HCFA authentication uses the HCFA authenticator, and instant authentication uses the instant authenticator. The instant authenticator is optionally used to avoid buffering malicious HCFA MPDUs on EBCS receivers.

The HCFA method uses both EBCS Info frames (9.6.7.100.5 EBCS Info frame format) and Data frames.

The frame sequence is shown in Figure 12-55b (EBCS HCFA frame sequence).

Figure 12-55b EBCS HCFA frame sequence

EBCS Info frames are transmitted every dot11EBCSInfoInterval (TI). TK is the HCFA key change interval and is configured as dot11EBCSHCFAKeyChangeInterval. TI shall be a multiple of TK. The interval between one EBCS Info frame and the next EBCS Info frame is called the HCFA period. Each HCFA period is identified by the HCFA sequence number that is lower 3 octets of the EBCS Info Sequence Number field value in the EBCS Info frame that is transmitted at the beginning of the HCFA period. [1095]

Each EBCS traffic stream has an ID that is indicated in the Content ID subfield in the Content Information field in the EBCS Info frame.

A period that uses the same HCFA authentication key is called a key period. Each key period has a sequence number, key sequence number, starting with 0 at the beginning of each HCFA period. Note that the key sequence number is different from HCFA key indexes.

Each HCFA MPDU has a sequence number, data sequence number, [1171] starting from 0 at the beginning of each key period. The HCFA MPDU is identified by the following identifiers:

- HCFA sequence number

- Content ID

- Key sequence number

- Data sequence number

EBCSData(s, c, k, d) represents the HCFA MPDU in which the HCFA sequence number is s, the content ID is c, the key sequence number is k and the Data sequence number is d.

IAuth(s, c, k, d) and HAuth(s, c, k, d) represent the instant authenticator and the HCFA authenticator for the EBCSData(s, c, k, d) respectively.

EBCSInfo(s) represents the EBCS Info frame for which the HCFA sequence number is s. For example, in case of two content streams, Content A and Content B, the identifiers are shown in Figure 12-55c (Identifiers example).

Figure 12-55c Identifiers example

The index of the HCFA base key and the HCFA authentication key is defined as B(s, c, k) and A(s, c, k) respectively where s is the HCFA sequence number, c is the content ID, k is the key sequence number.

The key sequence number is different from the HCFA base/authentication key index. The HCFA base/authentication keys are used in the reverse sequence of HCFA key generation. The relation between the HCFA base/authentication key index and the HCFA sequence number is shown in Table 12-10a (Relation between HCFA authentication key index and HCFA sequence number) where N is the number of HCFA authentication keys generated. An example of the HCFA key delivery is 1 shown in Figure 12-55d (Example HCFA Key Delivery).

Table 12-10a Relation between HCFA base/authentication key index and HCFA sequence number

NOTE—HCFA sequence number -3, -2 and -1 are used only for key verification

Figure 12-55d Example HCFA Key Delivery

An EBCS Info frame and an HCFA MPDU may contain multiple instant authenticators.

For example, EBCSData(s, c, k, d) may contain IAuth(s, c, k, d+1) and IAuth(s, c, k, d+3). In this case, the values 1 and 3 are called hash distances. The hash distance is configured in dot11EBCSHCFAHashDistance. Each instant authenticator is delivered with the frame identifier (s, c, k, d). An example of the instant authenticator delivery is shown in Figure 12-55e (Example Instant Authenticator Delivery).

Figure 12-55e Example Instant Authenticator Delivery

### 12.13.3.2 Key generation

HCFA is a one-way key chain authentication mechanism. The EBCS transmitter generates HCFA base keys and HCFA authentication keys for each content stream before each EBCS Info frame generation.

~~SHAKE128~~ SHA-256 hash function is used for HCFA key generation. ~~The output length of SHAKE128 is 256 bits.~~ [1104]

The HCFA base keys (Bs,n) are generated as follows:

Bs,0 = 256-bit random value

Bs,n = ~~SHAKE128~~ SHA-256 [1104] (“EBCS HCFA base key” || Bs,n-1) (n >= 1)

where

s is the HCFA sequence number of the generating EBCS Info frame.

“EBCS HCFA base key” is an ASCII string.

The HCFA authentication keys (As,n) are generated as follows:

As,n = ~~SHAKE128~~ SHA-256 [1104] (“EBCS HCFA authentication key” || Bs,n)

where

s is the HCFA sequence number of the generating EBCS Info frame.

“EBCS HCFA authentication key” is an ASCII string.

The number of keys to be generated (N) is following:

N = TI / TK + 3

where TI is the EBCS Info frame transmission interval and TK is the HCFA key change interval.

The HCFA key generation scheme is shown in Figure 12-55f (HCFA key generation scheme).

Figure 12-55f HCFA key generation scheme

In this figure, HashB is the hash function to generate HCFA base keys and HashA is the hash function to generate HCFA authentication keys.

HCFA keys are generated for each content stream.

### 12.13.3.3 HCFA MPDU generation

EBCS Data frames sent under HCFA use the HCFA MPDU format. [1419]

The ~~MPDU~~ format ~~for~~ of an HCFA MPDU is shown in Figure 12-55g (HCFA MPDU format). [1419]

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | MAC Header | ~~Timestamp~~  Content ID [w/o CID] | ~~Content ID~~  Timestamp [w/o CID] | HCFA Sequence | Key Sequence | Data Sequence |
| Octets |  | ~~8~~1  [w/o CID] | ~~1~~7  [1095] | ~~8~~3  [1095] | 1 | 2 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Data ~~l~~Length [w/o CID] | Data (PDU) | Disclosed Key | Instant Authenticators (optional) | HCFA Authenticator | FCS |
| Octets | 2 | >=1 | variable |  | variable |  |

Figure 12-55g HCFA MPDU format

The Content ID field contains the content ID of the EBCS traffic stream. [w/o CID]

The Timestamp field indicates the elapsed time from 2020-01-01 00:00 UTC in ~~milliseconds~~ microseconds [1095] until the time of the MPDU generation.

~~The Content ID field indicates the content ID of the MPDU.~~ [w/o CID]

The HCFA Sequence field indicates the HCFA sequence number of the MPDU.

The Key Sequence field indicates the key sequence number of the MPDU.

The Data Sequence field indicates the data sequence number of the MPDU.

The Data Length field indicates the length of the Data field.

The Data (PDU) field contains ~~the HLP packet~~ an MSDU. [1095, 1313]

The Disclosed Key field contains the HCFA base key to be disclosed.

The Instant Authenticator field contains instant authenticator(s) for the MPDU(s) that will be transmitted later.

The HCFA Authenticator field contains the HCFA authenticator of the MPDU.

The HCFA authenticator is the ~~KMAC128 (NIST SP 800-185)~~ HMAC-SHA-256 [1104] value of the MPDU with the HCFA authentication key.

HCFA Authenticator = ~~KMAC128~~ HMAC-SHA-256 [1104] (A(s, c, k), ~~T~~transmitter’s MAC address || HCFA MPDU fields from the beginning of the Timestamp field to the end of the Instant Authenticator field) [1207]

where

A(s, c, k) indicates the HCFA authentication key for the HCFA sequence s, the content ID c and the key sequence k.

The instant authenticator is the hash value of the MPDU to be transmitted later, generated as follows:

Instant Authenticator = ~~SHAKE128~~ SHA-256 [1104] (~~T~~transmitter’s MAC address || HCFA MPDU fields from the beginning of the Timestamp field to to the end of the Disclosed Key field) [1207]

### 12.13.3.4 EBCS Info frame reception

A received EBCS Info frame, EBCSInfo(s), is processed as follows:

1. If the EBCS Info frame is fragmented, defragment it as described in 11.55.2.6 (EBCS Info frame defragmentation).

2. If the difference between the timestamp in the EBCS Info frame and the time in the EBCS receiver’s clock is greater than the HCFA key change interval in the EBCS Info frame, the EBCS Info frame shall be discarded.

3. Verify the certificate of the AP in the EBCS Info frame using the installed certificate of the CA. If the verification fails or the certificate of the CA that signed the certificate of the AP in the EBCS Info frame is not installed, the EBCS Info frame shall be discarded.

4. Verify the signature in the EBCS Info frame using the certificate of the AP in the EBCS Info frame. If the verification fails, the EBCS Info frame shall be discarded.

5. If the HCFA base key(s) of the previous HCFA period, B(s-1, c, N-4) and B(s-1, c, N-5), is included and the HCFA MPDUs of the previous HCFA period to be authenticated are present, authenticate and process the HCFA MPDUs as described in 12.13.3.5 (HCFA MPDU reception).

6. Cache the HCFA sequence number, s, and the HCFA base key(s), B(s, c, 0), for the HCFA period of the EBCS Info frame.

7. If the instant authenticator(s) are present, cache the instant authenticators contained in the EBCS Info frame.

### 12.13.3.5 HCFA MPDU reception

HCFA MPDUs shall be discarded until an EBCS Info frame from the BSS is received.

An HCFA MDPU, EBCSData(s, c, k, d), is processed as follows.

1. Compute B(s, c, k-3) from B(s, c, k-2) in the EBCSData(s, c, k, d). If the computed B(s, c, k-3) is different from the cached B(s, c , k-3), the HCFA MPDU shall be discarded.

2. If instant authentication is used and the instant authenticator of the EBCSData(1 s, c, k, d), IAuth(s, c, k, d), is cached, compute the hash value of the EBCSData(s, c, k, d). If the computed hash value is different from the cached instant authenticator, the HCFA MPDU shall be discarded.

3. If instant authentication is used and the instant authenticator of the EBCSData(s, c, k, d), IAuth(s, c, k, d), is not cached, the HCFA MPDU may be cached until the HCFA base key for the key period is received, or the HCFA MPDU may be discarded.

4. If instant authentication is not used, the HCFA MPDU shall be cached until the HCFA base key for the key period is received.

5. If prior HCFA MPDU(s) using the HCFA authentication key derived from the HCFA base key included in the present HCFA MPDU, EBCSData(s,c,k-2,\*), are cached,

a. Derive the HCFA authentication key, A(s, c, k-2), from the HCFA base key, B(s, c, k-2).

b. Compute HCFA authenticator for the cached HCFA MPDU(s) by using the HCFA authentication key.

c. If the computed HCFA authenticator is different from the HCFA authenticator in the cached HCFA MPDU, the cached HCFA MPDU shall be discarded.

Then forward the ~~PDU~~ MSDU [1313] to a higher layer.

Even in case of missing HCFA MPDUs, the EBCS receiver recovers HCFA keys. As described in 12.13.3.2 (Key generation), HCFA base keys are generated by ~~SHAKE128~~SHA-256 [1104] hash function and transmitted in reverse sequence. The EBCS receiver is able to compute the missed HCFA base keys from the last received HCFA base key. For example, if the EBCS receiver missed all ~~e~~HCFA [1051] MPDUs containing B(s, c, k) but still cached B(s, c, k-1) and received B(s, c, k+1), the EBCS receiver computes B(s, c, k) and B(s, c, k-1) as follows:

B(s, c, k) = ~~SHAKE128~~ SHA-256 [1104] (“EBCS base key” || B(s, c, k+1))

B(s, c, k-1) = ~~SHAKE128~~ SHA-256 [1104] (“EBCS base key” || B(s, c, k))

Then the EBCS receiver authenticates the HCFA base keys by comparing the computed B(s, c, k-1) and the cached B(s, c, k-1). After successful key authentication, the EBCS receiver authenticates EBCSData(s, c, k, \*) and EBCSData(s, c, k+1, \*).

### 12.13.4 No frame authentication with mandatory higher layer source authentication (HLSA)

### 12.13.4.1 General [w/o CID]

If neither PKFA nor HCFA is used, a content source authentication mechanism shall be provided by a higher layer. The higher layer source authentication mechanism is out of scope of this standard. ~~In this case, EBCS Info frames and EBCS Data frames for DL or EBCS UL frames for UL are used.~~ [w/o CID]

Authentication of EBCS Info frames is optional if the EBCS Info frames include only HLSA content information. The EBCS AP may decide to use EBCS Info frame authentication or not.

If an EBCS Info frame includes the certificate of the AP, the EBCS receiver shall authenticate the EBCS Info frame as described in 12.13.2.2 (Authentication of the EBCS Info frame).

### 12.13.4.2 HLSA MPDU generation [w/o CID]

EBCS Data frames sent under HLSA use the HLSA MPDU format.

The format of an HLSA MPDU is shown in Figure 12-55h (HLSA MPDU format).

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | MAC Header | Content ID | Reserved | Data  (PDU) | FCS |
| Octets |  | 1 | 3 | 8 | 4 |

Figure 12-55h HLSA MPDU format

The Content ID field contains the content ID of the EBCS traffic stream.

The Data (PDU) field contains an MSDU.