IEEE P802.11  
Wireless LANs

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| CC31 Resolutions for Clause 10, 11 and 12 | | | | |
| Date: 2020-09-17 | | | | |
| Author(s): | | | | |
| Name | Affiliation | Address | Phone | email |
| Hitoshi Morioka | SRC Software | Fukuoka, JAPAN |  | [hmorioka@src-soft.com](mailto:hmorioka@src-soft.com) |
|  |  |  |  |  |

Abstract

This document describes proposed changes for clause 10, 11 and 12 in D0.1.

10 MAC sublayer functional description

10.6 Multirate support

10.6.5 Rate selection for Data and Management frame

*Insert the following subclauses:*

10.6.5.3 Rate selection for eBCS frames

The transmission rate for ~~eBCS Info frames and eBCS Data frames~~ eBCS Info frames, eBCS UL frames and Data frames for eBCS [55] ~~are~~ is [207] determined from dot11eBCSInfoTxRate and dot11eBCSDataTxRate respectively, if FMS or GCR is not used, otherwise

follow clause ~~10.\*\*\*~~ 10.6.5.4 (Rate selection for other group addressed Data and Management frames) and 10.6.5.6 (Rate selection for Data frames sent within an FMS stream) [25, 54, 203, 204].

~~Dot11eBCSInfoTxRate and dot11eBCSDataTxRate are configurable item and the values are selected from any available rates other than DSSS.~~ [205]

Renumber 10.6.5.3 to 10.6.5.4:

10.6.5.4 Rate selection for other group addressed Data and Management frames

~~Change the 1st paragraph as follows:~~

Add the following item at the last of the list in the 1st paragraph: [206, 335]

~~This subclause describes the rate selection rules for group addressed Data and Management frames,~~

~~excluding the following:~~

~~— Non-STBC Beacon and non-STBC PSMP frames~~

~~— ER beacon and HE beacon~~

~~— STBC group addressed Data and Management frames~~

~~— Data frames located in an FMS stream (see 11.22.8 (FMS multicast rate processing))~~

~~— Group addressed frames transmitted to the GCR concealment address (see 11.22.16.3.5 (Concealment of GCR transmissions))~~ [206, 335]

~~— Group addressed Data and Management frames transmitted in an HE ER SU PPDU (see 26.15.5 (Additional rules for ER beacons and group addressed frames))~~

~~— Group addressed Data and Management frames transmitted in an HE SU PPDU (see 26.15.6 (Additional rules for HE beacons and group addressed frames))~~

~~— Group addressed Data and Management frames transmitted in an HE MU PPDU (see 26.15.7 (Additional rules for group addressed frames in an HE MU PPDU))~~ [removed in REVmd D4.0]

— ~~eBCS Info, Data and Service Discovery frames~~ eBCS Info frames, eBCS UL frames and Data frames for eBCS [55]

11.bc Enhanced Broadcast Service (eBCS) procedures

Insert the following subclauses:

* **~~11.bc.1 Overview~~**

11.bc.1 Overview [56, 336]

This subclause describes enhanced Broadcast Services (eBCS) procedures that are used for eBCS STAs. eBCS is only supported in a non-DMG non-S1G infrastructure BSS.

eBCS can be used for both directions, downlink (eBCS DL) and uplink (eBCS UL). eBCS DL means broadcast from an eBCS AP to eBCS non-AP STAs. eBCS UL means broadcast from an eBCS non-AP STA to eBCS APs.

11.bc.2 eBCS DL procedures

11.bc.2.1 General

The eBCS DL uses three types of frame authentication mechanism as following.

* PKFA (12.bc.2 Public Key Frame Authentication)
* HCFA (12.bc.3 Hash Chain Frame Authentication)
* HLSA (12.bc.4 No frame authentication with mandatory higher layer source authentication)

eBCS DL uses both eBCS Info frames and eBCS Data frames.

In addition to these frames, Management frames are optionally used.

~~The frame sequence is shown in Figure 11-bc1 (eBCS DL frame sequence).~~

The frame sequence for a non-AP STA without association is shown in Figure 11-bc1A (eBCS DL frame sequence for a non-AP STA without association). The frame sequence for a non-AP STA with association is shown in Figure 11-bc1B (eBCS DL frame sequence for a non-AP STA with association). [338]

~~スクリーンショット が含まれている画像

自動的に生成された説明~~

スクリーンショットの画面

自動的に生成された説明

[214, 215, 338]

Figure 11-bc1A eBCS DL frame sequence for a non-AP STA without association [338]

スクリーンショットの画面

自動的に生成された説明

Figure 11-bc1B eBCS DL frame sequence for a non-AP STA with association [338]

11.bc.2.2 eBCS DL capability indications

The eBCS AP shall include the eBCS Capability element (~~9.4.2.\*~~ 9.4.2.bc [27] eBCS Capability element) in Beacon and Probe Response frames. The eBCS Capability element indicates the next eBCS Info transmission time in units of Beacon Interval. The eBCS Info frame is transmitted immediately after the indicated Beacon. [28] ~~For example, when the next eBCS Info transmission time is 5, the eBCS Info frame will be transmitted between next 5th Beacon and 6th Beacon.~~ [217]

~~When the eBCS non-AP STA that is scanning eBCS APs receives the Beacon frame or the Probe Response frame that include the eBCS Capability element, the eBCS non-AP STA should wait for the next eBCS Info. The eBCS non-AP STA may go into power save mode or move to other channel while waiting the next eBCS Info frame.~~ [219]

11.bc.2.3 eBCS Info frame generation and usage

The eBCS Info frame is transmitted periodically in ~~the interval of dot11BeaconPeriod \* dot11eBCSInfoInterval~~ every dot11eBCSInfoInterval beacon periods [221].

~~The eBCS Info frame contains the following information.~~

* ~~eBCS Info Sequence Number~~
* ~~Timestamp~~
* ~~Fragment Control~~
* ~~Fragment Hash Values (if fragmented)~~
* ~~eBCS Info Interval~~
* ~~Certificate of the AP (optional if HLSA is used for all contents, otherwise mandatory)~~
* ~~Content Information~~
* ~~Signature (optional if HLSA is used for all contents, otherwise mandatory)~~ [220]

The ~~eBCS Info~~ [222] Sequence Number is initialized ~~by a 64bit unsigned integer~~ to a random number [223] at the time of starting eBCS AP and ~~incremented by every eBCS Info frame generation~~ incremented by 1 for every new eBCS Info frame transmission [224]. If the eBCS Info Sequence Number overflows, it is set to 0.

~~The Timestamp is the time in UTC of generating the eBCS Info frame.~~

~~The Fragment Control indicates the total number of the fragmented frames and the index of the eBCS Info frame. The fragmentation procedure is described in 11.bc.2.4 (eBCS Info fragmentation).~~

~~The Fragment Hash Values are present if the eBCS Info frame is fragmented and the fragment sequence number is 0.~~

~~The eBCS Info Interval is the interval of the eBCS Info transmission configured in dot11eBCSInfoInterval.~~

~~The Certificate of the AP is the X.509 certificate of the eBCS AP.~~

~~Details of the Signature is described in 12.bc (Frame authentication for eBCS).~~

~~An eBCS Info frame may contain multiple Content Information.~~

~~Each Content Information contains the following items.~~

* ~~Authentication Algorithm~~
* ~~Allowable Time Difference~~
* ~~Broadcast Service ID~~
* ~~Title~~
* ~~Negotiation method~~
* ~~Higher Layer Destination Address~~
* ~~Time to Termination~~
* ~~Next Schedule~~
* ~~HCFA Base Key (in case of HCFA)~~
* ~~Previous Period HCFA Base Keys (in case of HCFA, optional)~~
* ~~HCFA Key Change Interval (in case of HCFA)~~
* ~~Instant Authenticators (in case of HCFA, optional)~~
* ~~Data (in case of PKFA, optional)~~

~~The Authentication Algorithm is either PKFA or HCFA with the public key algorithm and hash algorithm if required.~~

~~The Allowable Time Difference is the allowable time difference between the eBCS AP and the eBCS non-AP STAs. It depends on the frame authentication algorithm.~~

~~The Broadcast Service ID is the identifier of the content.~~

~~The Title is the human readable title of the content.~~

~~The Negotiation method is used to indicate the negotiation method that should be used to negotiate for the continuation of the broadcast service beyond the time of termination.~~

~~The Higher Layer Destination Address is the higher layer destination address of the content.~~

~~The Time to Termination is used to indicate the remaining time left until the content will terminate unless additional negotiation is conducted by the initiator of the broadcast service or by other users of the content.~~

~~The Next Schedule is used to indicate the next time the content is scheduled to be transmitted.~~

~~The HCFA Base Key, the Previous P 2 eriod HCFA Base Keys and the Instant Authenticators are used only in HCFA. The details are described in 12.bc.3 (eBCS hash chain frame authentication (HCFA)).~~

~~The Data is the piggy-backed data. It is used only in PKFA.~~ [225, 361]

On reception of the eBCS Info frame, the eBCS non-AP STA shall check the integrity of the eBCS Info frame as described in 12.15 (Frame authentication for eBCS) if the Certificate of the AP is included in the eBCS Info frame.

If the integrity of the eBCS Info frame is verified, the eBCS non-AP STA processes each Content Information according to the Authentication Algorithm.

* Common in all authentication algorithms,
  + The non-AP STA shall ~~C~~cache [228] the Title, the Negotiation Method, the Higher Layer Destination Address, the Time to Termination and the Next Schedule.
  + The non-AP STA shall ~~N~~notify [228] the cached information to SME through SME-MLME SAP as described in 6.3.\* (eBCS Content Information).
* In case of PKFA,
  + If the Data is present in the Content Information, the non-AP STA shall [228] forward the MSDU in the Data to higher layer~~. Otherwise~~, and [227] shall cache [228] the Certificate in the eBCS Info frame ~~shall be cached~~ [228] to authenticate PKFA MSDUs [229].
* In case of HCFA,
  + The HCFA Base Key in the Content Information shall be cached.
  + If the Instant Authenticators are present in the Content Information, the Instant Authenticators shall be cached.
  + If the Previous Period HCFA Base Keys are present in the Content Information, the non-AP STA shall [228] authenticate the eBCS Data frames of the previous HCFA period.

11.bc.2.~~3~~4 [60, 230] eBCS Info frame fragmentation ~~(or in 10.4)~~ [29, 59]

An eBCS Info frame may be fragmented into multiple MPDUs. ~~The length of each fragment shall be equal number of octets for all fragments except the last, which may be smaller.~~ [30, 231] The length of each fragment shall be an even number of octets, except for the last fragment, which may have an odd length [232]. The length of a fragment shall ~~never~~ not [233] be larger than dot11FragmentationThreshold [242].

The fragmentation procedure is following.

1. Construct an eBCS Info frame which is not fragmented yet [234] and determine the length of fragments.
2. When the eBCS Info frame is fragmented to N MPDUs, the ~~fragmentation number~~ Number Of Fragments subfield in the eBCS Info Control field [235] is set to N-1.
3. Insert a space for N-1 Fragment Hash Values to the eBCS Info frame.
4. ~~Each fragment contains the eBCS Sequence Number, the Timestamp and the Fragment Control.~~ [236]
5. Divide the eBCS Info frame after the Fragment Hash Values into fragments.

The Fragment Hash Values, the Certificate and the Signature shall be contained in the first fragment.

1. ~~The fragmentation number in the Fragment Control is set to the number of the fragments.~~ [237, 238]
2. The ~~fragmentation index~~ Fragment Index subfield in the ~~Fragment~~ eBCS Info Control field [239] is set to 0 (the first) to N-1 (the last) respectively.
3. Calculate the hash value of each fragment except the first one and put into the Fragment Hash Values.
4. Calculate and fill the signature of the first fragment.
5. Transmit the fragments consecutively in order of the Fragment Index. Data frames for eBCS shall not be transmitted before all of the fragments are transmitted. [240]

The eBCS Info frame fragmentation is shown in Figure 11-bc2 (eBCS Info frame fragmentation).

文字と写真のスクリーンショット

自動的に生成された説明

Figure 11-bc2 eBCS Info frame fragmentation [61, 241]

11.55.2.5 eBCS Info frame defragmentation

When the eBCS non-AP STA receives the eBCS Info frame with the Fragmentation Number in the Fragment Control is not 0 and the Fragmentation Index is 0, the eBCS non-AP STA shall verify the signature. If the verification succeeds, the eBCS non-AP STA shall cache the eBCS Info Sequence Number, Timestamp, the Fragment Control and the Fragment Hash Values.

When the eBCS non-AP STA receives the subsequent fragments of the eBCS Info frame, the eBCS non-AP STA shall check the integrity of the fragments by the following procedure.

1. The eBCS Info Sequence Number, the Timestamp and the Fragmentation Number in the Fragment Control in the received fragment are equal to that in the first fragment. If the values are different, the received fragment shall be discarded.
2. Calculate the hash value of the received fragment and compare it with the hash value in the Fragment Hash Values in the first fragment. If the hash value is different, the received fragment shall be discarded.
3. Cache the Content Information of the received fragment.

After all fragments are received, the eBCS non-AP STA concatenates the fragments and process the eBCS Info frame as described in 11.bc.2.3 (eBCS Info frame generation and usage). [62]

12.bc Frame authentication for eBCS

12.bc.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 ~~frame~~ [269] authentication.

- Public Key Frame Authentication (PKFA)

- Hash Chain Frame Authentication (HCFA) with Instant Authentication

- Hash Chain Frame Authentication (HCFA) without Instant Authentication

- ~~No frame authentication with mandatory higher layer source authentication (HLSA)~~ Higher layer source authentication (HLSA); no frame authentication [269]

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 requires computational cost. HCFA requires latency for buffering. The charactoristics of HLSA depends on the algorithm of higher layer. [270]

The following preparations are required for both PKFA and HCFA before starting eBCS.

* The certificate(s) of the CA(s) (Certificate Authority) shall be installed into the eBCS receivers.

Note: The certificate of the CA(s) ~~may be~~ is [271] 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 depends on higher layer algorithm and are out of scope of this standard. [273]

12.bc.2 eBCS public key frame authentication (PKFA)

12.bc.2.1 Signature of the eBCS Info frame

One of the following public key algorithms is used.

* ~~RSA-2048~~ RSASSA-PSS (FIPS PUB 186-5) [282]
* ECDSA~~-P256~~ (FIPS PUB 186-5) [282]
* Ed25519 (FIPS PUB 186-5) [282]

For RSASSA-PSS, the length of the modulus shall be 2048 bits and SHAKE128 (FIPS 202) shall be used as a mask generation function and a hash function during signature generation. The output length of SHAKE128 shall be 256 bits.

For ECDSA, the domain parameters shall be P-256 (NIST SP 800-186) and SHAKE128 shall be used as a hash function during signature generation. The output length of SHAKE128 shall be 256 bits.

For Ed25519, SHA-512 shall be used as a hash function during signature generation. [289]

The eBCS transmitter generates an eBCS Info frame ~~when it receives data to be transmitted~~ periodically as described in 11.bc.2.3 (eBCS Info frame generation and usage) [277, 278]. ~~The eBCS Info frame contains the following items.~~

* ~~eBCS Info sequence number~~
* ~~Timestamp~~
* ~~Authentication algorithm~~
* ~~Allowable time difference~~
* ~~Length of the Certificate of the AP~~
* ~~Certificate of the AP~~
* ~~Content Information~~
* ~~(Data)~~
* ~~Signature~~ [274]

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.bc.\* (eBCS Info fragmentation)~~ 11.bc.2.4 (eBCS Info frame fragmentation) [66].

If the eBCS Info frame is not fragmented, fill all the fields according to 9.6.7.bc.5 [67] (eBCS Info frame format) except the signature.

~~Generate~~ The eBCS transmitter shall generate the [275] signature as follows:

Signature = Sign(The eBCS transmitter’s private key, ~~SHAKE128(~~Transmitter’s MAC address || from the Sequence Number field to the last Content Information field in the eBCS Info frame~~)~~) [289]

where

Sign(*k*, *m*) indicates a digital signature for the message *m* using the private key *k*. [289]

Otherwise, only the first fragment contains the signature.

Signature = Sign(The eBCS transmitter’s private key, ~~SHAKE128(~~Transmitter’s MAC address | from the Sequence Number field to the ~~last~~ end [276] of the first fragment~~)~~) [289]

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

HashValue = SHAKE128(Transmitter’s MAC address || from the Sequence Number field to the last of the fragment) ~~..........................................................~~ (12-bc1) [280]

~~The output length of SHAKE128 is 256bit.~~ [289]

The eBCS transmitter fills the Signature field and the Fragment Hash Values field in eBCS Info frame with the computed Signature and HashValues respectively as described in 11.bc.2.3 (eBCS Info frame generatin and usage). [279]

~~Then the eBCS transmitter transmits the eBCS Info frame.~~ [283]

12.bc.2.2 Authentication of the eBCS Info frame

When the eBCS receiver receives the eBCS Info frame, the eBCS receiver shall authenticate it as follows:

1. If the eBCS Info frame is fragmented, the following procedures are applied only to the first fragment.
2. If the difference between the timestamp in the eBCS Info frame and the time of the clock [284] of the eBCS receiver is greater than the allowable time difference in the eBCS Info frame, the eBCS Info frame shall be discarded.
3. 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.
4. 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.
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.
6. Compute the hash value of the fragment as described in the formula (12-bc1) in 12.bc.2.1(Signing).
7. If the computed hash value is equal to the cached hash value, the authentication succeeds. Otherwise, the fragment shall be discarded.
8. If the authentication succeeds, the eBCS receiver defragments the eBCS Info frame as described in 11.bc.2.5 (eBCS Info frame defragmentation). [69, 286, 287]

If the authentication succeeds,

* The eBCS receiver caches the certificate of the AP and the allowable time difference in the eBCS Info frame.
* If data is present in the Content Information, the eBCS receiver processes the data in the Content Information field(s) in accordance with ~~11.bc.\* (eBCS Info frame reception)~~ 11.bc.2.3 (eBCS Info frame generatin and usage) [68].

~~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 following.~~

1. ~~Compute the hash value of the fragment as described in the formula (1) in 12.bc.2.1(Signing).~~
2. ~~If the computed hash value is equal to the cached hash value, the authentication succeeds. Otherwise, the fragment shall be discarded.~~

~~If the authentication succeeds, the eBCS receiver defragments the eBCS Info frame as described in 11.bc.\* (eBCS Info frame defragmentation).~~ [287]

~~12.bc.2.3 Signature of the eBCS Data frame~~

~~The eBCS Data frame contains the following items.~~

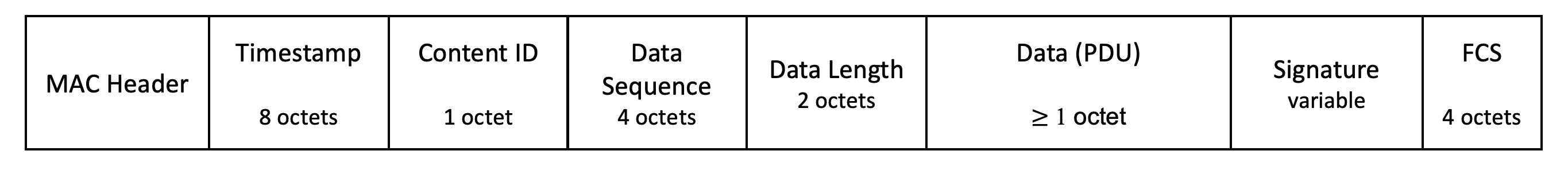
* ~~Data~~
* ~~Timestamp~~
* ~~Signature~~

~~Each eBCS Data frame shall be signed using the certificate of the eBCS transmitter as following.~~

~~Signature = Sign(The eBCS transmitter’s private key, SHAKE128(Transmitter’s MAC address | /\* to be determined \*/))~~[288, 290]

12.bc.2.3 PKFA MPDU format

The MPDU format for PKFA is shown in Figure 12-bc1 (PKFA MPDU format).



**Figure 12-bc1 PKFA MPDU format**

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

The Content ID field contains the content ID of the MPDU.

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 field contains the HLP packet.

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, Transmitter’s MAC address || From Timestamp field to the end of Data field)

where

Sign(*k*, *m*) indicates a digital signature for the message *m* using the private key *k*. [70]

12.bc.2.4 Authentication of an ~~eBCS Data~~ PKFA MPDU [71] 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 ~~eBCS Data frame~~ PKFA MPDU [72] and the time of the clock [284] of the eBCS receiver is greater than the cached allowable time difference, the ~~eBCS Data frame~~ PKFA MPDU [73] shall be discarded.
2. Verify the signature in the ~~eBCS Data frame~~ PKFA MPDU [74] using the cached certificate. If the verification fails, the ~~eBCS Data frame~~ PKFA MPDU [75] ~~shall be discarded.~~ shall be discarded, otherwise decapsulate the PDU. [77]

~~If the authentication succeeds, the eBCS receiver processes the data in the eBCS Data frame.~~ [76, 77]

12.bc.2.5 Signature of the eBCS UL frame

One of the following public key algorithms is used.

* ~~RSA-2048~~RSASSA-PSS
* ECDSA~~-P256~~
* Ed25519

For RSASSA-PSS, the length of the modulus shall be 2048 bits and SHAKE128 shall be used as a mask generation function and a hash function during signature generation. The output length of SHAKE128 shall be 256 bits.

For ECDSA, the domain parameters shall be P-256 and SHAKE128 shall be used as a hash function during signature generation. The output length of SHAKE128 shall be 256 bits.

For Ed25519, SHA-512 shall be used as a hash function during signature generation. [289]

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.bc (eBCS UL frame format).

The signature is generated as following:

Signature = Sign(The eBCS transmitter’s private key, ~~SHAKE128(~~Transmitter’s MAC address || from the eBCS UL Control field to the last field before the Frame Signature field in the eBCS UL frame~~)~~)

where

Sign(*k*, *m*) indicates a digital signature for the message *m* using the private key *k*. [289]

Then the eBCS transmitter transmits the eBCS UL frame.

12.bc.2.~~2~~6 [78, 291] Authentication of an eBCS UL frame

When an eBCS receiver receives an eBCS UL frame, the eBCS receiver shall authenticate it as follows:

1. If the Timestamp is present and the difference between the timestamp in the eBCS UL frame and the time of the eBCS receiver is greater than the configured value, the eBCS UL frame shall be discarded.
2. Verify the certificate of the STA in the eBCS UL frame using the installed certificate of the CA. If the verification fails or the certificate of the CA that signed the certificate of the STA in the eBCS UL frame is not installed, the eBCS UL frame shall be discarded.
3. Verify the signature in the eBCS UL frame using the certificate of the STA in the eBCS UL frame. If the verification fails, the eBCS UL frame shall be discarded.

If the authentication succeeds,

* The eBCS receiver processes the HLP Payload as described in 11.bc.~~1~~3.2 [79] (eBCS UL operation at an eBCS AP).

12.bc.3 eBCS hash chain frame authentication (HCFA)

12.bc.3.1 General

HCFA uses a digital signature and a modified TESLA (Timed Efficient Stream Loss-Tolerant Authentication, IETF RFC 4082 [292]).

~~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 hash function is used for HCFA key generation. The output length of SHAKE128 is 256bit.~~

~~The HCFA base keys (~~*~~B~~~~s,n~~*~~) are generated as follows:~~

*~~B~~~~s,0~~* ~~= Random value (256bit length)~~

*~~B~~~~s,n~~* ~~= SHAKE128(“eBCS HCFA base key” ||~~ *~~B~~~~s,n-1~~*~~) (n >= 1)~~

~~where~~ *~~s~~* ~~is the sequence number of the generating eBCS Info frame.~~

~~The HCFA authentication keys (~~*~~A~~~~s,n~~*~~) are generated as follows:~~

*~~A~~~~s,n~~* ~~= SHAKE128(“eBCS HCFA authentication key” ||~~ *~~B~~~~s,n~~*~~)~~

~~where~~ *~~s~~* ~~is the sequence number of the generating eBCS Info frame.~~

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

*~~N~~* ~~=~~ *~~T~~~~I~~* ~~/~~ *~~T~~~~K~~* ~~+ 3~~

~~where~~ *~~T~~~~I~~* ~~is the eBCS Info frame transmission interval and~~ *~~T~~~~K~~* ~~is the HCFA key change interval.~~

~~The HCFA key generation scheme is shown in Figure 12-bc1 (HCFA key generation scheme).~~

時計 が含まれている画像

自動的に生成された説明

~~Figure 12-bc1 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.~~ [80]

HCFA provides the following two authentication methods to authenticate each ~~eBCS Data frame~~ HCFA MPDU [81].

* HCFA authentication
* Instant authentication (optional)

Each authentication method uses a separate authenticator. HCFA authentication uses the HCFA authenticator, and ~~the~~ [297] instant authentication uses the instant authenticator. The instant authenticator is optionally used to ~~filter~~ avoid buffering [299] ~~the~~ [298] malicious ~~eBCS Data frames~~ HCFA MPDUs [82] on eBCS receivers [299].

~~The HCFA authenticator is the KMAC128 (NIST Special Publication 800-185) value of the eBCS Data frame that contains the HCFA authenticator with HCFA authentication key.~~

~~HCFA Authenticator = KMAC128(~~*~~A~~~~n~~*~~, eBCS Data frame including Instant Authenticator)~~

~~The instant authenticator is the hash value of the eBCS Data frame to be transmitted later that is generated as following:~~

~~Instant Authenticator = SHAKE128(eBCS Data frame to be transmitted later)~~ [83]

The HCFA uses both eBCS Info frames (9.6.7.bc.5 [84] eBCS Info frame format) and ~~eBCS Data frames (/\* reference to be added \*/)~~ Data frames [85]. The frame sequence is shown in Figure 12-bc2 (eBCS HCFA frame sequence).

スクリーンショット が含まれている画像

自動的に生成された説明

Figure 12-bc2 eBCS HCFA frame sequence

The eBCS Info frames are transmitted ~~periodically in the interval of~~ every [301] dot11eBCSInfoInterval (*TI*) [301]. *TK* is the HCFA key change interval configured as dot11eBCSHCFAKeyChangeInterval. *TI* shall be a multiple of *TK*. The period between one eBCS Info frame and the next eBCS Info frame is called the ~~“HCFA period”~~ *HCFA period* [302].

Each HCFA period is identified by the HCFA sequence number.

~~Each content stream has an index that is determined by the order, starting with index 0, in the Content Information field in the eBCS Info frame.~~

Each content has an ID that is indicated in the Content ID subfield in the Content Information field in the eBCS Info frame. [86]

The period that uses the same HCFA authentication key is called a ~~“Key period”~~ *key period* [302, 304]. Each ~~Key period~~ key period [302, 304] has its sequence number, ~~Key sequence number~~ *key sequence number* [302, 303], starting with 0 at the beginning of each HCFA period. Note that

the ~~Key~~ key [304] sequence number is different from HCFA key indexes.

Each ~~eBCS Data frame~~ HCFA MPDU [87] has a sequence number starting from 0 at the beginning of each ~~Key period~~ key period [302, 304]. The ~~eBCS Data frame~~ HCFA MPDU [88] is identified by the following identifiers:

* HCFA sequence number
* Content ~~stream index~~ ID [89]
* Key sequence number
* Data sequence number

eBCSData(*s, c, k, d*) represents the ~~eBCS Data frame~~ HCFA MPDU [90] in which the HCFA sequence number is s, the Content ~~stream index~~ ID [91] is c, the ~~Key~~ key [304] 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-bc3 (Identifiers example).

文字と写真のスクリーンショット

自動的に生成された説明

文字と写真のスクリーンショット

自動的に生成された説明

Figure 12-bc3 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 index~~ Content ID [93], *k* is the ~~Key~~ key [304] sequence number.

The ~~Key~~ key [304] sequence number is different from the HCFA base/authentication key index. The HCFA base/authentication keys are used in the ~~opposite order~~ reverse sequence [305] of the HCFA key generation. The relation between the HCFA base/authentication key index and the HCFA sequence number is shown in Table 12-bc1 (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 shown in Figure 12-bc4

(Example HCFA Key Delivery).

Table 12-bc1 Relation between HCFA base/authentication key index and HCFA sequence number

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| HCFA base/authentication key index | ~~(~~N-1~~)~~ [306] | ~~(~~N-2~~)~~ [306] | ~~(~~N-3~~)~~ [306] | N-4 | N-5 | N-6 | … | 0 |
| HCFA sequence number | ~~(~~-3~~)~~ [306] | ~~(~~-2~~)~~ [306] | ~~(~~-1~~)~~ [306] | 0 | 1 | 2 | … | N-4 |

Note: HCFA sequence number -3, -2 and - 1 1 are used only for key verification.

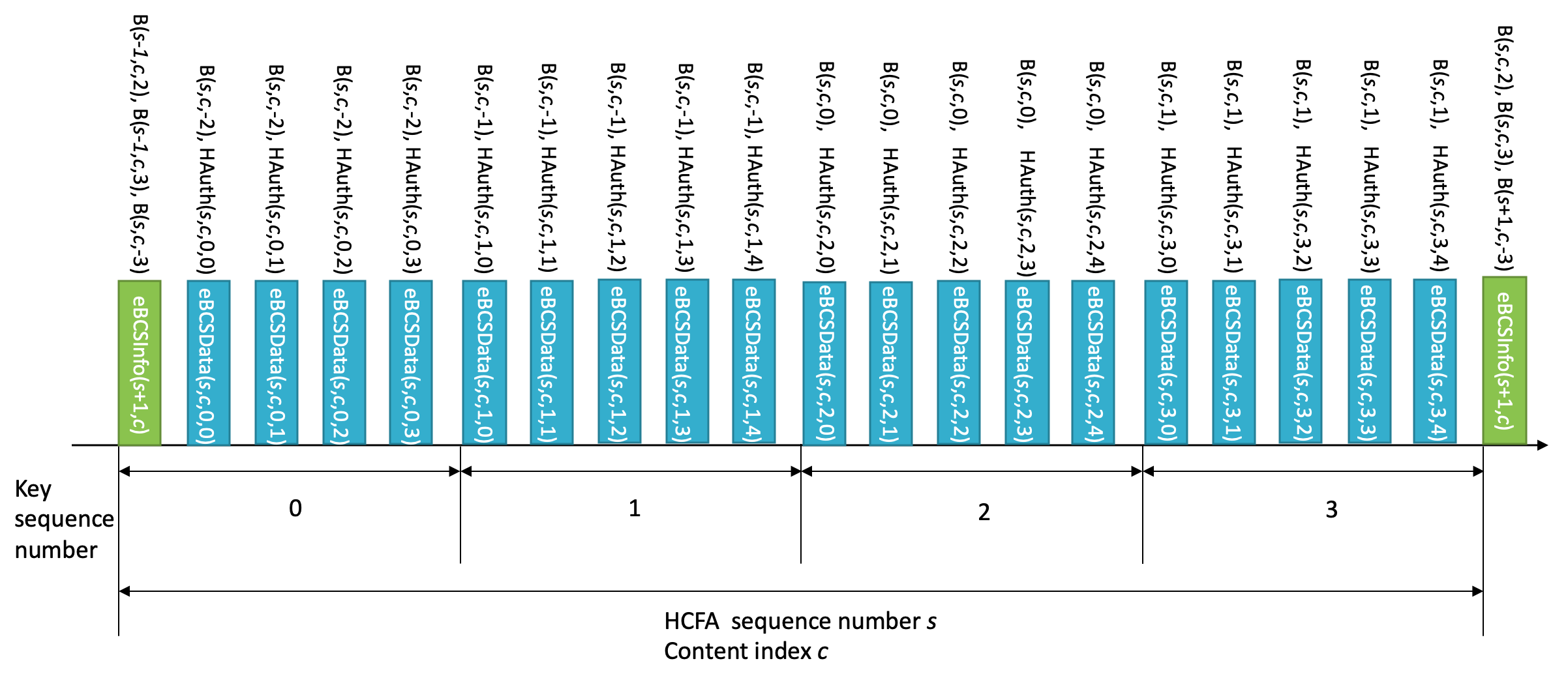


Figure 12-bc4 Example HCFA Key Delivery

An eBCS Info frame and an ~~eBCS Data frame~~ HCFA MPDU [94] 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~~ *hash distances* [302]. The ~~Hash Distance~~ hash distance [302] 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-bc5 (Example Instant Authenticator Delivery).

スクリーンショット が含まれている画像

自動的に生成された説明

Figure 12-bc5 Example Instant Authenticator Delivery

12.bc.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 hash function is used for HCFA key generation. The output length of SHAKE128 is 256bit.

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

*Bs,0* = 256-bit random value [293]

*Bs,n* = SHAKE128(“eBCS HCFA base key” || *Bs,n-1*) (n >= 1)

where

*s* is the HCFA sequence number [294] of the generating eBCS Info frame.

“eBCS HCFA base key” is an ASCII string. [295]

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

*As,n* = SHAKE128(“eBCS HCFA authentication key” || *Bs,n*)

where

*s* is the HCFA sequence number [294] of the generating eBCS Info frame.

“eBCS HCFA authentication key” is an ASCII string. [295]

The number of ~~the~~ [295] 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-bc6 (HCFA key generation scheme).

時計 が含まれている画像

自動的に生成された説明

Figure 12-bc6 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. [80]

~~12.bc.3.2 eBCS Info frame generation~~

~~An eBCS Info frame contains the following items that are related to frame authentication.~~

* ~~HCFA sequence number~~
* ~~Timestamp~~
* ~~Certificate~~
* ~~Signature~~
* ~~HCFA key change interval~~
* ~~Content Information~~
  + ~~HCFA base key(s)~~
  + ~~Instant authenticator(s) of eBCS Data frame to be transmitted (optional)~~

~~The functions of the eBCS Info sequence number, the timestamp, the certificate and the signature are same as those of PKFA.~~

~~The HCFA key change interval, HCFA base key(s) and the instant authenticator(s) are present only in HCFA.~~

~~The HCFA key change interval is~~ *~~T~~~~K~~*~~.~~

~~The HCFA base keys to be included in the eBCS Info frame of sequence number~~ *~~s~~* ~~are B(~~*~~s, c,~~* ~~-3), B(~~*~~s~~*~~-1,~~ *~~c~~*~~, 1) and B(~~*~~s~~*~~-1,~~ *~~c~~*~~, 0) for all content streams, where~~ *~~c~~* ~~is the content index. In case of the first eBCS Info frame, B(~~*~~s~~*~~-1,~~ *~~c~~*~~, 1) and B(~~*~~s~~*~~-1,~~ *~~c~~*~~, 0) are not present.~~

~~If instant authentication is used, the instant authenticator(s) with frame identifier (~~*~~s, c, k, d~~*~~) is present. In this case, the eBCS transmitter must buffer data packets to generate instant authenticators.~~

~~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.bc.\* (eBCS Info fragmentation).~~ [95, 96, 97, 98, 99, 290]

12.bc.3.3 ~~eBCS Data frame~~ HCFA MPDU [100] generation

~~The eBCS Data frame contains the following items.~~

* ~~Content data~~
* ~~HCFA sequence number~~
* ~~Content index~~
* ~~Key sequence number~~
* ~~Data sequence number~~
* ~~HCFA base key~~
* ~~Instant authenticator(s)~~
* ~~HCFA authenticator~~

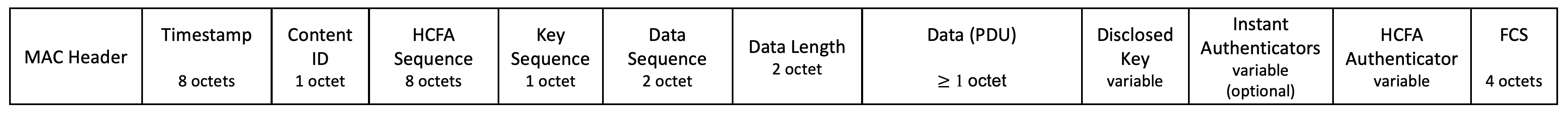
~~The HCFA sequence number, the Content index, the Key sequence number and the Data sequence number are described in 12.bc.3.1 (General).~~

~~The HCFA base key contained in eBCSData(~~*~~s, c, k, d~~*~~) is B(~~*~~s, c, k~~*~~-2).~~

~~The instant authenticator(s) in the eBCSData(~~*~~s, c, k, d~~*~~) depends on the configured Hash Distance.~~

~~The HCFA authenticator in the eBCSData(~~*~~s, c, k, d~~*~~) is HAuth(~~*~~s, c, k, d~~*~~) with A(~~*~~s, c, k~~*~~).~~ [101, 102, 103, 290]

The MPDU format for HCFA is shown in Figure 12-bc7 (HCFA MPDU format).



**Figure 12-bc7 HCFA MPDU format**

The Timestamp field indicates the elapsed time from 2020-01-01 00:00 UTC in milliseconds that indicates the time of the MPDU generation.

The Content ID field indicates the content ID of the MPDU.

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 field contains the HLP packet.

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

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

The HCFA Authenticator field contains the HCFA authenticator of the MPDU. [104]

The HCFA authenticator is the KMAC128 (NIST SP 800-185) value of the MPDU with the HCFA authentication key.

HCFA Authenticator = KMAC128(A(s, c, k), Transmitter’s MAC address || from the Timestamp field to the Instant Authenticator field)

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 that is generated as follows [300]:

Instant Authenticator = SHAKE128(Transmitter’s MAC address || from the Timestamp field to the Disclosed Key field) [83]

12.bc.3.4 eBCS Info frame reception

A received eBCS Info frame, eBCSInfo(s), is processed as following.

1. If the eBCS Info frame is fragmented, defragment it at first as described in ~~11.55.\* (eBCS Info defragmentation)~~ 11.bc.2.5 (eBCS Info frame defragmentation) [105].
2. If the difference betwee 1 n 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 ~~eBCS Data frames~~ HCFA MPDUs [106] of the previous HCFA period to be authenticated are present, authenticate and process the ~~eBCS Data frame~~ HCFA MPDU [106] as described in ~~12.15.3.5~~ 12.bc.3.5 [107] (~~eBCS Data frame~~ HCFA MPDU [106] reception).
6. Cache the HCFA sequence number, *s*, and [108] 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.bc.3.~~4~~5 [80] ~~eBCS Data frame~~ HCFA MPDU [109] reception

~~eBCS Data frames~~ HCFA MPDUs [110] shall be discarded until a eBCS Info frame from the BSS is received.

A received ~~eBCS Data frame~~ HCFA MPDU [111], eBCSData(*s, c, k, d*), is processed as following.

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 ~~eBCS Data frame~~ HCFA MPDU [112] shall be discarded.
2. If instant authentication is used and the instant authenticator of the eBCSData(*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 ~~eBCS Data frame~~ HCFA MPDU [113] 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 ~~eBCS Data frame~~ HCFA MPDU [114] may be cached until the HCFA base key for the ~~Key period~~ key period [302] is received, or the ~~eBCS Data frame~~ HCFA MPDU [115] may be discarded.
4. If instant authentication is not used, the ~~eBCS Data frame~~ HCFA MPDU [116] shall be cached until the HCFA base key for the ~~Key period~~ key period [302, 304] is received.
5. If ~~eBCS Data frame(s)~~ HCFA MPDU(s) [117] using the HCFA authentication key derived from the HCFA base key included in the ~~eBCS Data frame~~ HCFA MPDU [118], eBCSData(*s,c,k*-2,\*), are cached,
   * 1. Derive the HCFA authentication key, A(*s, c, k*-2), from the HCFA base key, B(*s, c, k*-2).
     2. Compute HCFA authenticator for the cached ~~eBCS Data frame~~ HCFA MPDU [119] by using the HCFA authentication key.
     3. If the computed HCFA authenticator is different from the HCFA authenticator in the cached ~~eBCS Data frame~~ HCFA MPDU [120], the cached ~~eBCS Data frame~~ HCFA MPDU [120] shall be discarded.

Then forward the ~~content in the eBCS Data frame~~ PDU [121] to a higher layer.

Even in case of missing ~~eBCS Data frames~~ HCFA MPDUs [122], the eBCS receiver recovers HCFA keys. As described in 12.bc.3.2 (Key generation), HCFA base keys are generated by SHAKE128 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. [307] For example, if the eBCS receiver missed all ~~eBCS Data frames~~ HCFA MPDUs [123] 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(“eBCS base key” || B(*s, c, k*+1))

B(*s, c, k*-1) = SHAKE128(“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 thecached B(*s, c, k*-1). After successful key authentication, the eBCS receiver authenticates eBCSData(*s, c, k,* \*) and eBCSData(*s, c, k*+1, \*).

12.bc.4 No frame authentication with mandatory higher layer source authentication (HLSA)

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.

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 ~~12.15.2.2~~ 12.bc.2.2 [124] (Authentication of the eBCS Info frame).