IEEE P802.11
Wireless LANs

|  |
| --- |
| Draft Text for 12.15 Security |
| Date: 2020-05-19 |
| Author(s): |
| Name | Affiliation | Address | Phone | email |
| Hitoshi Morioka | SRC Software | Fukuoka, JAPAN |  | hmorioka@src-soft.com |
|  |  |  |  |  |

Abstract

This document describes a TGbc draft text proposal for clause 12 security.

*Yellow marked numbers are temporal and to be assigned by ANA.*

12. Security

*Add the following new subclause in clause 12.*

12.15 Frame authentication for eBCS

12.15.1 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 authentication.

eBCS UL cases use PKFA or HLSA.

* 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)

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

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.
The certificate of the CA(s) may be installed with an application like contents 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 of which the eBCS receiver have the certificate.

12.15.2 eBCS public key frame authentication (PKFA)

12.15.2.1 Signature of the eBCS Info frame

One of the following public key algorithms is used.

* RSA-2048
* ECDSA-P256
* Ed25519

The eBCS transmitter generates an eBCS Info frame when it receives data to be transmitted. 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
* Contents Information
* (Data)
* Signature

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.\* (eBCS Info fragmentation).

If the eBCS Info frame is not fragmented, fill all the fields according to 9.6.7.52 (eBCS Info frame format) except the signature.

generate signature as following:

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

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 of the first fragment))

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) (1)

The output length of SHAKE128 is 256bit.

Then the eBCS transmitter transmits the eBCS Info frame.

12.15.2.2 Authentication of the eBCS Info frame

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

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 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 by the installed certificate of the CA. If the verification is failed 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 by the certificate of the AP in the eBCS Info frame. If the verification is failed, the eBCS Info frame shall be discarded.

If the authentication succeeds,

* The eBCS receiver caches the certificate of the AP and the allowable time difference in the eBCS Info frame.
* If the Data is present in the Contents Information, the eBCS receiver processes the Data in the Contents Information field(s) in accordance with 11.55.\* (eBCS Info frame reception).

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 12.15.2.1(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 11.55.\* (eBCS Info frame defragmentation).

12.15.2.3 Signature of the eBCS Data frame

The eBCS Data frame contains the following items.

* Data
* Timestamp
* Signature

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

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

12.15.2.4 Authentication of the eBCS Data frame

When the eBCS receiver receives the eBCS Data frame, the eBCS receiver shall authenticate it as following:

1. If the difference between the timestamp in the eBCS Data frame and the time of the eBCS receiver is greater than the cached allowable time difference, the eBCS Data frame shall be discarded.
2. Verify the signature in the eBCS Data frame by the cached certificate. If the verification is failed, the eBCS Data frame shall be discarded.

If the authentication succeeds, the eBCS receiver processes the Data in the eBCS Data frame.

12.15.2.5 Signature of the eBCS UL frame

One of the following public key algorithms is used.

* RSA-2048
* ECDSA-P256
* Ed25519

The eBCS transmitter generates an E-BCS UL frame when it receives data to be transmitted. The format of the E-BCS UL frame is described in 9.6.7.X (E-BCS UL frame format).

The signature is generated as following:

Signature = Sign(The eBCS transmitter’s private key, SHAKE128(Transmitter’s MAC address | from the Sequence Number field to the last field before the Frame Signature Length field in the E-BCS UL frame))

Then the eBCS transmitter transmits the E-BCS UL frame.

12.15.2.2 Authentication of the eBCS UL frame

When the eBCS receiver receives the E-BCS UL frame, the eBCS receiver shall authenticate it as following:

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

If the authentication succeeds,

* The eBCS receiver processes the HLP Payload as described in 11.X.Y.2 (E-BCS UL operation at an eBCS AP).

12.15.3 eBCS hash chain frame authentication (HCFA)

12.15.3.1 General

The HCFA uses the digital signature and the modified TESLA (Timed Efficient Stream Loss-Tolerant Authentication, IETF RFC4082).

HCFA is a kind of 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 following:

 *Bs,0* = Random value (256bit length)

 *Bs,n* = SHAKE128(“eBCS HCFA base key” || *Bs,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 following:

 *As,n* = SHAKE128(“eBCS HCFA authentication key” || *Bs,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* = *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-bc1 (HCFA key generation scheme).



**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.

The HCFA keys are generated for each content stream.

The HCFA provides the following two authentication methods to authenticate each eBCS Data frame.

* HCFA authentication
* Instant authentication (optional)

Each authentication method uses separate authenticator. The HCFA authentication uses the HCFA authenticator, and the instant authentication uses the instant authenticator. The instant authenticator is optionally used to filter the malicious eBCS Data frames.

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(*An*, 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)

The HCFA uses both eBCS Info frames (9.6.7.52 eBCS Info frame format) and eBCS Data frames (/\* reference to be added \*/). The frame sequence is shown in Figure 12-bc2 (eBCS HCFA frame sequence).



**Figure 12-bc2 eBCS HCFA frame sequence**

The eBCS Info frames are transmitted periodically in the interval of dot11EBCSInfoInteval (*TI*). *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 “HCFA period”. Each HCFA period is identified by the HCFA sequence number.

Each content stream has its index that is determined by the order, starts with index 0, in the Contents Information field in the eBCS Info frame.

The period that uses the same HCFA authentication key is called “Key period”. Each Key period has its sequence number, Key sequence number, that starts with 0 at the beginning of each HCFA period. Note that the Key sequence number is different from HCFA key indexes.

Each eBCS Data frame has its sequence number that starts from 0 at the beginning of each Key period. The eBCS Data frame is identified by the following identifiers:

* HCFA sequence number
* Content stream index
* Key sequence number
* Data sequence number

eBCSData(*s*, *c*, *k*, *d*) represents the eBCS Data frame that the HCFA sequence number is *s*, the Content stream index 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 that the HCFA sequence number is *s*.

For example, in case of two contents stream, Content A and Content B, the identifiers are shown in Figure 12-bc3 (Identifiers example).



**Figure 12-bc3 Identifiers example**

The index of HCFA base key and 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, *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 opposite order 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 the HCFA authentication key. An example of the HCFA key delivery is shown in Figure 12-bc4 (Example HCFA Key Delivery).

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

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| TESLA base/authentication key index | (N-1) | (N-2) | (N-3) | N-4 | N-5 | N-6 | … | 0 |
| TESLA sequence number | (-3) | (-2) | (-1) | 0 | 1 | 2 | … | N-4 |

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



Figure 12-bc4 Example HCFA Key Delivery

An eBCS Info frame and an eBCS Data frame 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 Distance. 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-bc5 (Example Instant Authenticator Delivery).



Figure 12-bc5 Example Instant Authenticator Delivery

12.15.3.2 eBCS Info frame generation

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

* HCFA sequence number
* Timestamp
* Certificate
* Signature
* HCFA key change interval
* Contents Information
	+ HCFA base key(s)
	+ Instant authenticator(s) of eBCS Data frames 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 *TK*.

The HCFA base keys to be included in the eBCS Info frame of the 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 the instant authentication is used, the instant authenticator(s) with frame identifier (*s, c, k, d*) is present. In this case, the eBCS transmitter requires to buffer data packets to generate instant authenticator(s).

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.\* (eBCS Info fragmentation).

12.15.3.3 eBCS Data frame 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 12.15.3.1 (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*).

12.15.3.4 eBCS Info frame reception

The 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).
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 by the installed certificate of the CA. If the verification is failed 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 by the certificate of the AP in the eBCS Info frame. If the verification is failed, 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 contained and the eBCS Data frames of the previous HCFA period to be authenticated are present, authenticate and process the eBCS Data frames as described in 12.15.3.5 (eBCS Data frame reception).
6. Cache 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.

12.15.3.4 eBCS Data frame reception

The eBCS Data frame shall be discarded until the eBCS Info frame from the BSS is received.

The received eBCS Data frame, 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*, *t*, *d*). If the computed B(*s*, *c*, *k*-3) is different from the cached B(*s*, *c* , *k*-3), The eBCS Data frame shall be discarded.
2. If the 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 shall be discarded.
3. If the 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 may be cached until the HCFA base key for the Key period is received, or the eBCS Data frame may be discarded.
4. If the instant authentication is not used, the eBCS Data frame shall be cached until the HCFA base key for the Key period is received.
5. If the eBCS Data frame(s) using the HCFA authentication key derived from the HCFA base key included in the eBCS Data frame, eBCSData(*s*,*c*,*k*-2,\*), is cached,
6. Derive the HCFA authentication key, A(*s*, *c*, *k*-2), from the HCFA base key, B(*s*, *c*, *k*-2).
7. Compute HCFA authenticator for the cached eBCS Data frame by using the HCFA authentication key.
8. If the computed HCFA authenticator is different from the HCFA authenticator in the cached eBCS Data frame, the cached eBCS Data frame shall be discarded.

Then forward the content in the eBCS Data frame to the higher layer.

Even in case of missing eBCS Data frames, the eBCS receiver recovers HCFA keys. For example, in case of missing all eBCS Data frames the contains B(*s*, *c*, *k*) and the eBCS receiver cached B(*s*, *c*, *k*-1) and receives B(*s*, *c*, *k*+1), the eBCS receiver compute B(*s*, *c*, *k*) and B(*s*, *c*, *k*-1) as following.

 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 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).

12.15.4 No frame authentication with mandatory higher layer source authentication (HLSA)

If neither PKFA nor HCFA is used, the content source authentication mechanism shall be provided by the higher layer. The higher layer source authentication mechanism is out of scope of this standard. In this case, only eBCS Data frames for DL or E-BCS UL frames for UL are used.