IEEE P802.11  
Wireless LANs

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| FILS fixes | | | | |
| Date: 2017-07-11 | | | | |
| Author(s): | | | | |
| Name | Affiliation | Address | Phone | email |
| Jouni Malinen | Qualcomm, Inc. |  |  | jouni.malinen@qca.qualcomm.com |
|  |  |  |  |  |

Abstract

This document proposes changes to IEEE Std 802.11ai-2016 (as merged into IEEE P802.11-REVmd/D0.1) to fix issues found during implementation and review efforts after the amendment publication.

### Discussion

This document addresses the REVmd/D0.1 comment collection CID 102:

“Number of issues in P802.11ai has been discovered during implementation efforts. I ran out of time in filing these individually as comments for this round, but will be working on a contribution to address them. This comment is to get a CID for such discussion..”

Issues addressed in the proposed changes in this document:

* IKM was renamed to FILS-FT, but some places were missed in P802.11ai
* DH\_SS contents for ECC case was not described explicitly to include only the x-coordinate from ECDH which is the common way of implementing such shared secret cases
* RSNE AKM suite selector description for FILS AKMs did not include 802.1X and PMKSA caching options
* PMKID derivation after 4-way handshake for FILS SK case where ERP key hierarchy is established (i.e., FILS authentication is not used) does not list new FILS AKMs in 12.7.1.3; this would result in using the “otherwise” case with HMAC-SHA-1 which does not look appropriate; we have been otherwise consistent on using SHA-256 and SHA-384 with the new AKMs, so this looks like an oversight and a missed edit in P802.11ai
* 9.4.2.178 FILS Request Parameters element: Figure 9-589d (FILS Request Parameters element format) does not show conditional size ("0 or 1") for the FILS Criteria field (it is present only if FILS Criteria Present = 1 just like the other subfields in this element with a matching present-bit)
* Need to add FILS Discovery frame to 11.1.3.7 (Beacon reception) ("A non-AP STA in which dot11MultiBSSIDActivated is true shall support frame filtering for up to two BSSIDs...") exceptions
* 12.12.2.6.2 misses AP requirement on verifying FILS Session matching the one used in Authentication frames
* 9.4.2.171.1 (Reduced Neighbor Report element – Neighbor AP information field) changes in P802.11ai related to the TBTT Information field got lost and misedited number of times during the process.. The end result is incomplete since the text now claims that the value can be either 0 or 1 without specifying what either value means. See below for longer history on this, but the short outcome is that those P802.11ai changes should most likely be reverted so that this field continues to be specified as having value 0.
* P802.11ai did not specify how the new FT+FILS AKMs are used with FTE: FT MIC field not specified (need to use SIV): 13.8.4 FT authentication sequence: contents of third message: how is FTE MIC supposed to be used with new FILS+FT AKMs? Those do not derive KCK..
* FILS PK description of Authentication frame construction and processing uses different style compared to FILS SK and does not describe an option to use PMKSA caching

Additional background information regarding **Reduced Neighbor Report element, Neighbor AP information field, TBTT Information field** (9.4.2.1.71.1):

The current standards says values 2 and 3 are reserved, but does not define when values 1 and 0 are used.

It looks like the actual description of the values was lost in P802.11ai/D3.0. D1.0 and D2.0 did describe these.. The latest version from D2.0:

"Value 0 indicates the presence of the informative Neighbor AP Information that is used to help the STA in AP discovery. Value 1 indicates the presence of the Neighbor AP Information that is used to recommend that the STA switch to another channel, another band, or neighbor AP as specified in the Neighbor AP Information field. Values 2 and 3 are reserved."

It looks like this was actually done on purpose (CID 4521 and 4522):

"Revise. Both value 0 and value 1 for TBTT Information Field Type have similar functionality. So removing the type 1 added by Tgai and keeping the value 0 as in baseline 802.11af.

Revert all changes in last paragraph of page 39 of draft 2.0. Delete the last two paragraphs of Clause 10.43.8."

However, it looks like this was edited incorrectly. That "revert all changes" should have left the baseline "Its value is 0. Values 1, 2, and 3 are reserved." text as-is, but P802.11ai/D3.0 ended up removing it.

Next LB seems to be trying to fix this in CID 6942:

'REVISED

Undo the deletion from baseline of the following sentence at Page 42

Line 21: "Its value is 0. Values 1, 2, and 3 are reserved."'

But this edit is also done incorrectly in D4.0.. Or to be more exact, the group seems to have approved conflicting resolution for CID 6139 and 6140, but it is next to impossible to understand what that resolution is supposed to do.. It is referring to 15/41r1 showing changes, but that is

only a note, not actual part of resolution. The changes from 15/41r1 are clearly not included in full.

So it is a bit difficult to interpret what the group really wanted to do here.. Since the majority of 15/41r1 changes (i.e., text describing how the contents of the TBTT Information subfield depends on the TBTT Information Field Type value) are not included in P802.11ai/D11.0, I'd go with the interpretation that the will of the group is to do what CIDs 4521, 4522, and 6942 did. In other words, the TBTT Information Field Type field shall be set to 0 in all cases defined so far.

Since there are rules in Table 9-258a defining the TBTT Information field contents based on the TBTT Information Length subfield (instead of TBTT Information Field Type subfield), the standard seems to be clear from this view point as well.

As far as legacy STA (11af and/or REVmc, but no 11ai) is concerned, such a STA would likely be mighty confused if it were to receive a Reduced Neighbor Report element with TBTT Information Length set to a value other than 1. I'm not sure how such an implementation would behave here. That said, I would not see setting TBTT Information Field Type subfield to 1 (or any non-zero value) making this behavior any more defined, so all in all, the changes proposed in 15/41r1 do not seem helpful.

### Proposed changes

**9.4.2.25 RSNE**

**9.4.2.25.3 AKM suites**

Table 9-133—AKM suite selectors

*Modify the following rows in the table (REVmd/D0.1 page 946 lines 24-47) as shown with change tracking after this snapshot image from the draft:*



00-0F-AC:14

Key management over FILS using SHA-256 and AES-SIV-256, PMKSA caching, or authentication negotiated over IEEE Std 802.1X

FILS key management defined in 12.12.2.5 (Key establishment with FILS authentication)

Defined in 12.12.2.5 (Key establishment with FILS authentication) using SHA-256.

00-0F-AC:15

Key management over FILS using SHA-384 and AES-SIV-512, PMKSA caching, or authentication negotiated over IEEE Std 802.1X

FILS key management defined in 12.12.2.5 (Key establishment with FILS authentication)

Defined in 12.12.2.5 (Key establishment with FILS authentication) using SHA-384.

00-0F-AC:16

FT authentication over FILS with SHA-256 and AES-SIV-256, PMKSA caching, or authentication negotiated over IEEE Std 802.1X

FT authentication defined in 12.7.1.7.2 (Key derivation function (KDF))

Defined in 12.7.1.7.2 (Key derivation function (KDF)) using SHA-256.

00-0F-AC:17

FT authentication over FILS with SHA-384 and AES-SIV-512, PMKSA caching, or authentication negotiated over IEEE Std 802.1X

FT authentication defined in 12.7.1.7.2 (Key derivation function (KDF))

Defined in 12.7.1.7.2 (Key derivation function (KDF)) using SHA-384.

**12.7.1.7 FT key hierarchy**

**12.7.1.7.1 Overview**

*Replace “IKM” with “FILS-FT” in Figure 12-31 (just below the FILS Authentication box) (REVmd/D0.1 page 2153):*



Replace that IKM with FILS-FT

Figure 12-31—FT key hierarchy at an Authenticator(11ai)

**13.2.2 Authenticator key holders**

*Modify third paragraph as shown (REVmd/D0.1 page 2243 line 39):*

The R0KH derives the PMK-R0 for use in the mobility domain utilizing the MSK (when the AKM negotiated is 00-0F-AC:3), the PSK (when the AKM negotiated is 00-0F-AC:4) or the PMK (when the AKM negotiated is 00-0F-AC:9), or the FILS-FT (when the AKM negotiated is 00-0F-AC:16 or 00-0F-AC:17). The R0KH shall be responsible for deriving a PMK-R1 for each R1KH within the mobility domain.

**13.2.3 Supplicant key holders**

*Modify the following paragraphs as shown (REVmd/D0.1 page 2244 lines 44 and 51):*

The S0KH and S1KH are responsible for the derivation of keys in the FT key hierarchy. The S0KH and S1KH are entities that are assumed to physically reside in the Supplicant.

The S0KH interacts with the IEEE 802.1X functional block (see Figure 4-19 (Portion of the ISO/IEC basic reference model covered in this standard) in 4.9 (Reference model)) to receive the MSK resulting from an EAP authentication or the FILS-FT resulting from a FILS authentication. The S1KH interacts with the IEEE 802.1X entity to open the Controlled Port. Both the S0KH and S1KH interactions with the IEEE 802.1X entity occur within the SME of a STA.

The S0KH derives the PMK-R0 for use in the mobility domain utilizing the MSK (when the AKM negotiated is 00-0F-AC:3), the PSK (when the AKM negotiated is 00-0F-AC:4) or the PMK (when the AKM negotiated is 00-0F-AC:9), or the FILS-FT (when the AKM negotiated is 00-0F-AC:16 or 00-0F-AC:17).

**13.2.4 FT initial mobility domain association over FILS in an RSN**

*Modify the following paragraph as shown (REVmd/D0.1 page 2246 line 1):*

Upon successful completion of FILS authentication request processing, the R0KH on the AP uses the FILS-FT (see 12.12.2.5.3) to establish key hierarchy. If a key hierarchy already exists for this STA belonging to the same mobility domain (i.e., having the same MDID), the R0KH shall delete the existing PMK-R0 security association and PMK-R1 security associations. It then calculates the PMK-R0, PMKR0Name, and PMK-R1 and makes the PMK-R1 available to the R1KH of the AP to which the STA is associated.

*Modify the following paragraph as shown (REVmd/D0.1 page 2246 lines 34-35):*

When FILS authentication is used to establish the FT key hierarchy, TK and KEK for the initial mobility domain association are derived as part of the FILS authentication as defined in 12.12.2.5.3.

NOTE—This means that the PTK derivation as described in 12.7.1.7.5 (PTK) is not used in the case of FT initial mobility domain association over FILS.

**12.12.2.5.2 PMKSA key derivation with FILS authentication**

*Modify the definition of DHss as shown (REVmd/D0.1 page 2236 line 4):*

DHss is the shared secret derived from the Diffie-Hellman exchange, when performed; when ECC is used, only the x-coordinate from ECDH is included

**12.7.1.3 Pairwise key hierarchy**

*Modify the PMKID derivation rules as shown (REVmd/D0.1 page 2149 lines 7-24):*

When the negotiated AKM is 00-0F-AC:5 or 00-0F-AC:6, the PMK identifier is defined as

PMKID = Truncate-128(HMAC-SHA-256(PMK, "PMK Name" || AA || SPA))

When the negotiated AKM is 00-0F-AC:11, the PMK identifier is defined as

PMKID = Truncate-128(HMAC-SHA-256(KCK, "PMK Name" || AA || SPA))

When the negotiated AKM is 00-0F-AC:14 or 00-0F-AC:16, the PMK identifier is defined as

PMKID = Truncate-128(HMAC-SHA-256(KEK, "PMK Name" || AA || SPA))

When the negotiated AKM is 00-0F-AC:12, and the PMK identifier is defined as

PMKID = Truncate-128(HMAC-SHA-384(KCK, "PMK Name" || AA || SPA))

When the negotiated AKM is 00-0F-AC:15,or 00-0F-AC:17, and the PMK identifier is defined as

PMKID = Truncate-128(HMAC-SHA-384(KEK, "PMK Name" || AA || SPA))

Otherwise, the PMK identifier is defined as

PMKID = Truncate-128(HMAC-SHA-1(PMK, "PMK Name" || AA || SPA))

**12.12.2.6.3 (Re)Association Response for FILS key confirmation**

*Modify the following paragraph as shown (REVmd/D0.1 page 2239 line 64-page 2240 line 2):*

The plaintext passed to the AEAD algorithm is the data that would follow the FILS Session element in an

unencrypted frame. The output of the AEAD algorithm becomes the data that follows the FILS Session element in the encrypted and authenticated (Re)Association Response frame. The output of the algorithm is as specified in IETF RFC 5116. The resulting (Re)Association Response frame shall be transmitted to the STA.

**9.4.2.178 FILS Request Parameters element**

*Replace “FILS Criteria” field length in Figure 9-589d from “1” to “0 or 1” (REVmd/D0.1 page 1192) (no changes to Figure 9-594; that figure is included below for discussion context only):*





**11.1.3.7 Beacon reception**

*Modify the following paragraph as shown (REVmd/D0.1 page 1700 line 60):*

A non-AP STA in which dot11MultiBSSIDActivated is true shall support frame filtering for up to two

BSSIDs; one for the transmitted BSSID and one for the nontransmitted BSSID. The STA, when associated

with a BSS corresponding to a nontransmitted BSSID, shall discard all Data and Management frames that use the transmitted BSSID as the transmit address, except for Beacon, FILS Discovery, Probe Response, and TIM broadcast frames.

**12.12.2.6.2 (Re)Association Request for FILS key confirmation**

*Modify as shown (REVmd/D0.1 page 2238 line 6):*

The plaintext passed to the AEAD algorithm is the data that would follow the FILS Session element in an

unencrypted frame. The output of the AEAD algorithm becomes the data that follows the FILS Session

element in the encrypted and authenticated (Re)Association Request frame. The output of the algorithm is as

specified in IETF RFC 5116. The resulting (Re)Association Request frame shall be transmitted to the AP.

The AP compares FILS Session of the received (Re)Association Request frame with the FILS Session that was used to identify the FILS session in the Authentication frames. If they differ, authentication exchange fails.

The AP decrypts and verifies the received (Re)Association Request frame with the AEAD algorithm as

defined in 12.12.2.7 (AEAD cipher mode for FILS) with the KEK as the key. The AAD is reconstructed as

defined above and is passed, along with the ciphertext of the received frame, to the AEAD decryption

operation.

If the output from the AEAD decryption operation returns a failure, the authentication exchange fails. If the

output does not return failure, the output plaintext replaces the ciphertext as portion of the frame that follows the FILS Session element and processing of the received frame continues by checking the value of the FILS Key Confirmation element.

The AP verifies that the RSNE received in the (Re)Association Request frame has identical AKM suite and

cipher suites and RSN capabilities as were included in the RSNE in the Authentication frame from the STA.

If these fields differ, the authentication exchange fails.

**9.4.2.171.2 Neighbor AP Information field**

*Modify the following paragraph as shown (REVmd/D0.1 page 1185 line 15):*

The TBTT Information Field Type subfield is 2 bits in length and defines the structure of the TBTT

Information field. Its value is 0. Values 1, 2, and 3 are reserved.

**9.4.2.48 Fast BSS Transition element (FTE)**

*Modify as shown (REVmd/D0.1 page 995):*

The FTE includes information needed to perform the FT authentication sequence or FILS

authentication(11ai) during a fast BSS transition in an RSN. This element is shown in Figure 9-315 (FTE

format).



The Element ID and Length fields are defined in 9.4.2.1 (General).

The MIC Control field is two octets and is defined in Figure 9-316 (MIC Control field).



The Element Count subfield of the MIC Control field contains the number of elements that are included in

the message integrity code (MIC) calculation. A value of 0 indicates no MIC is present (the MIC field has value 0).

The MIC field contains a MIC that is calculated using the algorithm specified in 13.8.4 (FT authentication

sequence: contents of third message) and 13.8.5 (FT authentication sequence: contents of fourth message). When using an AEAD cipher, the MIC field has value 0.

The ANonce field contains a value chosen by the R1KH. It is encoded following the conventions in 9.2.2

(Conventions).

The SNonce field contains a value chosen by the S1KH. It is encoded following the conventions in 9.2.2

(Conventions).

The format of the Optional Parameter(s) field is shown in Figure 9-317 (Optional Parameter(s) field).



The Subelement ID field is defined in Table 9-159 (Subelement IDs):



R1KH-ID indicates the identity of the R1KH, which is used by the S0KH and the R0KH for deriving the

PMK-R1s. It is encoded following the conventions in 9.2.2 (Conventions).

The GTK subelement contains the group temporal key, which is encrypted (see procedures in 13.8.5 (FT

authentication sequence: contents of fourth message)) and is defined in Figure 9-318 (GTK subelement

format).

*Replace “24-40” with “16-40” as the Wrapped Key field length in Figure 9-318.*



The GTK subelement Key Info subfield is defined in Figure 9-319 (GTK subelement’s Key Info subfield).



Key Length field is the length of the Key field in octets, not including any padding (see 13.8.5 (FT

authentication sequence: contents of fourth message)).

RSC field contains the receive sequence counter (RSC) for the GTK being installed. Delivery of the RSC

field value allows a STA to identify replayed MPDUs. If the RSC field value is less than 8 octets in length,

the remaining octets are set to 0. The least significant octet of the transmit sequence counter (TSC) or packet

number (PN) is in the first octet of the RSC field. See Table 12-5 (Key RSC field).

For WEP, the RSC value is reserved.

The Wrapped Key field contains the encrypted GTK as described in 13.8.5 (FT authentication sequence:

contents of fourth message).

When sent by a non-AP STA, the R0KH-ID indicates the R0KH with which the S0KH negotiated the

PMK-R0 it is using for this transition. When sent by an AP, the R0KH-ID indicates the R0KH that the

S0KH will be using to generate a PMK-R0 security association. It is encoded following the conventions

from 9.2.2 (Conventions).

The IGTK field contains the Integrity GTK, used for protecting robust Management frames. The IGTK

subelement format is shown in Figure 9-320 (IGTK subelement format).

*Replace “24” with “16-40” as the Wrapped Key field length in Figure 9-318.*



The Key ID field indicates the value of the BIP key identifier.

The IPN field indicates the receive sequence counter for the IGTK being installed, to allow a STA to identify

replayed protected group addressed robust Management frames.

The Key Length field is the length of IGTK in octets, not including any padding (see 13.8.5 (FT

authentication sequence: contents of fourth message)).

The Wrapped Key field contains the wrapped IGTK being distributed. The length of the resulting AES-Keywrapped IGTK in the Wrapped Key field is Key Length + 8 octets. When using an AEAD cipher, there is no padding within the Wrapped Key field.

**13.8.4 FT authentication sequence: contents of third message**

*Modify as shown (REVmd/D0.1 page 2267):*

The RSNE shall be present only if dot11RSNAActivated is true. If present, the RSNE shall be set as follows:

— Version field shall be set to 1.

— PMKID Count field shall be set to 1.

— PMKID field shall contain the PMKR1Name.

— All other fields shall be as specified in 9.4.2.25 (RSNE) and 12.6.3 (RSNA policy selection in an

infrastructure BSS).

The MDE shall contain the MDID and FT Capability and Policy fields. This element shall be identical to the

MDE contained in the first message of this sequence.

The FTE shall be present only if dot11RSNAActivated is true. If present, the FTE shall be set as follows:

— ANonce, SNonce, R0KH-ID, and R1KH-ID shall be set to the values contained in the second

message of this sequence.

— The Element Count field of the MIC Control field shall be set to the number of elements protected in

this frame (variable).

— When the negotiated AKM is 00-0F-AC:3, 00-0F-AC:4, or 00-0F-AC:9, the MIC shall be calculated

using the KCK and the AES-128-CMAC algorithm. The output of the AES-128-CMAC shall be 128

bits.

— When the negotiated AKM is 00-0F-AC:16 or 00-0F-AC:17, the MIC field is set to 0.

— When using an AEAD cipher, the AAD used with the AEAD algorithm consists of the following data passed as separate components in the order given here; and when not using an AEAD cipher, the MIC shall be calculated on the concatenation of the following data, in the order given here:

— FTO’s MAC address (6 octets)

— Target AP’s MAC address (6 octets)

— Transaction sequence number (1 octet), which shall be set to the value 5 if this is a Reassociation Request frame and, otherwise, set to the value 3

— RSNE

— MDE

— FTE, with the MIC field of the FTE set to 0

— Contents of the RIC-Request (if present)

— All other fields shall be set to 0.

If resources are being requested by the FTO, then a sequence of elements forming the RIC-Request shall be

included.

**13.8.5 FT authentication sequence: contents of fourth message**

*Modify as shown (REVmd/D0.1 page 2267):*

If the status code is SUCCESS, then the following rules apply.

The RSNE shall be present only if dot11RSNAActivated is true. If present, the RSNE shall be set as follows:

— Version field shall be set to 1.

— PMKID Count field shall be set to 1.

— PMKID field shall contain the PMKR1Name

— All other fields shall be identical to the contents of the RSNE advertised by the target AP in Beacon

and Probe Response frames.

The MDE shall contain the MDID and FT Capability and Policy fields. This element shall be identical to the

MDE contained in the second message of this sequence.

The FTE shall be present only if dot11RSNAActivated is true. If present, the FTE shall be set as follows:

— ANonce, SNonce, R0KH-ID, and R1KH-ID shall be set to the values contained in the second

message of this sequence.

— The Element Count field of the MIC Control field shall be set to the number of elements protected in

this frame (variable).

— When this message of the authentication sequence appears in a Reassociation Response frame, the

Optional Parameter(s) field in the FTE may include the GTK and IGTK subelements. If a GTK or an

IGTK are included, it shall be encrypted. When using an AEAD cipher, the GTK and IGTK subelements shall be encrypted. When not using an AEAD cipher, the Key field of the subelement shall be encrypted using KEK and the NIST AES key wrap algorithm. The Key field shall be padded before encrypting if the key length is less

than 16 octets or if it is not a multiple of 8. The padding consists of appending a single octet 0xdd

followed by zero or more 0x00 octets. When processing a received message, the receiver shall

ignore this trailing padding. Addition of padding does not change the value of the Key Length field.

Note that the length of the encrypted Key field can be determined from the length of the GTK or

IGTK subelement.

— When the negotiated AKM is 00-0F-AC:3, 00-0F-AC:4, or 00-0F-AC:9, the MIC shall be calculated

using the KCK and the AES-128-CMAC algorithm. The output of the AES-128-CMAC algorithm

shall be 128 bits.

— When the negotiated AKM is 00-0F-AC:16 or 00-0F-AC:17, the MIC field is set to 0.

— When using an AEAD cipher, the AAD used with the AEAD algorithm consists of the following data passed as separate components in the order given here; and when not using an AEAD cipher, the MIC shall be calculated on the concatenation of the following data, in the order given here:

— FTO’s MAC address (6 octets)

— Target AP’s MAC address (6 octets)

— Transaction sequence number (1 octet), which shall be set to the value 6 if this is a Reassociation Response frame or, otherwise, set to the value 4

— RSNE

— MDE

— FTE, with the MIC field of the FTE set to 0

— Contents of the RIC-Response (if present)

— All other fields shall be set to 0.

If this message is other than a Reassociation Response frame and dot11RSNAActivated is false, a TIE may

appear. If this message is other than a Reassociation Response frame, includes a RIC-Response, and

dot11RSNAActivated is false, then a timeout interval shall appear. If it appears, it shall be set as follows:

— Timeout Interval Type field shall be set to 1 (reassociation deadline)

— Timeout Interval Value field shall be set to the reassociation deadline time.

If resources were requested by the FTO, then a RIC-Response shall be included.

**12.8.7 Mapping IGTK to BIP keys**

*Modify as shown (REVmd/D0.1 page 2211 line 27):*

See 12.7.1.5 (Integrity group key hierarchy) for the definition of the IGTK. A STA shall use bits 0–127 of

the IGTK as the AES-128-CMAC key, bits 0–127 of the IGTK as the AES-128-GMAC key, bits 0-255 of the IGTK as the AES-256-CMAC key, and bits 0–255 of the IGTK as the AES-256-GMAC key.

**12.12.2.3 Key establishment with FILS Shared Key authentication**

**12.12.2.3.5 Non-AP STA processing of Authentication frame**

*Delete the final paragraph as shown (REVmd/D0.1 page 2233 lines 33-34) (note: the correct next step with reference to 12.12.2.5 and 12.12.2.6 is couple of paragraphs before this incorrect reference to public key authentication):*

**12.12.2.4 Key establishment with FILS Public Key authentication**

**12.12.2.4.1 General**

*Modify as shown (REVmd/D0.1 page 2233):*

This subclause defines the procedure for establishing a shared key between a FILS capable STA and AP

using FILS Public Key authentication.

A STA may initiate FILS Public Key authentication either using a public key or a cached PMKSA.

**12.12.2.4.2 Prior to exchange**

*Modify as shown (REVmd/D0.1 page 2233):*

FILS Public Key authentication performs key establishment with a Diffie-Hellman exchange. Prior to beginning the exchange, the non-AP STA performs the following:

a) Selects a finite cyclic group from the dot11RSNConfigDLCGroup table to perform the Diffie-

Hellman exchange.

b) Generates a random 16-octet nonce, generates an ephemeral private key, and uses the selected group’s scalar-op (see 12.4.4.1 (General)) with its private key to generate its ephemeral public key.

c) Determines whether to attempt PMKSA caching and if so, generates a list of PMKSA identifiers.

d) Constructs an Authentication frame (see 9.3.3.12 (Authentication frame format)) as follows:

1) The Authentication algorithm number shall be set to 6 (FILS Public Key authentication) (see 9.4.1.1 (Authentication Algorithm Number field)) and the Authentication transaction sequence number shall be set to 1.

2) The random nonce shall be encoded in the FILS Nonce element (see 9.4.2.190 (FILS Nonce element)).

3) The chosen finite cyclic group shall be encoded in the Finite Cyclic Group field (see 9.4.1.43 (Finite Cyclic Group field)).

4) The STA’s public key shall be encoded into the FFE field (see 9.4.1.41 (Finite field element (FFE) field)) according to the element to octet-string conversion in 12.4.7.2.4 (Element to octet string conversion).

5) The random FILS Session shall be encoded in the FILS Session element (see 9.4.2.180 (FILS Session element)).

6) If a list of PMKSA identifiers was generated, it shall be used to construct the PMKID List field in RSNE.

The STA then transmits the Authentication frame to the AP.

**12.12.2.4.3 Processing after receipt**

*Modify as shown (REVmd/D0.1 page 2234):*

Upon reception of the Authentication frame with the Authentication algorithm number equal to 6, the AP shall perform the following procedure:

a) If the finite cyclic group indicated by the Finite Cyclic Group field is not acceptable, the AP shall

respond with an Authentication frame with the status code of 77 (“Authentication is rejected

because the offered finite cyclic group is not supported”) and terminate the FILS authentication

protocol.

b) If the finite cyclic group is acceptable, the AP shall verify the validity of the STA’s public key:

1) The public key is converted from an octet string to an element according to the conversion in 12.4.7.2.5 (Octet string to element conversion).

2) The public key, as a group element, is verified in a group-specific fashion as described in 5.6.2.3 of NIST SP 800-56A R2. If verification fails, the AP shall terminate the FILS authentication protocol.

c) The STA’s nonce and validated public key are extracted from the Authentication frame.

d) The AP shall check whether PMKSA caching is being attempted by the presence of the PMKID List

field in RSNE. If the PMKID List field is present in RSNE, the AP checks whether any PMKSA identifier

offered in the PMKID List matches an identifier for a cached PMKSA. If so, the AP selects a

PMKID that matches and continues the FILS Public Key authentication protocol using the

PMK from the identified PMKSA. Otherwise, the AP shall continue with the Diffie-Hellman exchange.

**12.12.2.4.4 Post processing**

*Modify as shown (REVmd/D0.1 page 2234):*

Next, the AP shall

a) Generate a random 16-octet nonce

b) If PMKSA caching is not used, generate a random ephemeral private key, and then uses the agreed-upon group’s scalar-op (see 12.4.4.1 (General)) with its private key to generate its ephemeral public key.

b) Construct an Authentication frame (see 9.3.3.12 (Authentication frame format)) as follows:

1) The Authentication algorithm number is set to 6 (FILS Public Key authentication) (see 9.4.1.1 (Authentication Algorithm Number field)), and the Authentication transaction sequence number is set to 2.

2) The random nonce is encoded in the FILS Nonce element (see 9.4.2.190 (FILS Nonce element)).

3) If PMKSA caching is not used, the finite cyclic group is encoded in the Finite Cyclic Group field (see 9.4.1.43 (Finite Cyclic Group field)).

4) If PMKSA caching is not used, the AP’s public key is encoded in the FFE field (see 9.4.1.41 (Finite field element (FFE) field)) according to the element to octet-string conversion in 12.4.7.2.4 (Element to octet string conversion).

5) The AP copies the FILS Session element from the Authentication frame received from the STA.

6) If PMKSA caching is used, the AP indicates the selected PMKID in the PMKID List.

c) Transmit the Authentication frame to the STA.

d) If PMKSA caching is not used, compute the Diffie-Hellman shared secret, DHss, based on the STA’s ephemeral public key and its own private key with the chosen group’s scalar-op.

e) Perform key derivation (see 12.12.2.5 (Key establishment with FILS authentication)).

**12.12.2.4.5 Upon Receipt**

*Modify as shown (REVmd/D0.1 page 2234):*

The STA processes the received Authentication frame as follows:

a) The STA shall abandon FILS authentication if any of the following conditions occur:

1) The received Authentication frame does not include the Authentication Algorithm Number

equal to 6 (FILS Public Key authentication) (see 9.4.1.1 (Authentication Algorithm Number field)).

2) PMKSA caching was attempted and the received Authentication frame includes a PMKID that

does not match a PMKID in the Authentication frame sent by the STA.

3) The received Authentication frame does not include the FILS Session element.

4) The received FILS Session value does not match the one in the Authentication frame sent by

the STA.

b) If PMKSA caching is not used, verifies that the finite cyclic group in the AP’s response is equal to the group selected by the STA. If these differ, the STA shall terminate the authentication exchange.c) If PMKSA caching is not used, verifies the validity of the AP’s public key:

1) The public key is converted from an octet string to an element according to the conversion in 12.4.7.2.5 (Octet string to element conversion).

2) The public key, as a group element, is verified in a group-specific fashion according to 5.6.2.3 of NIST SP 800-56A R2. If public key validation fails the STA shall terminate the authentication exchange.

c) Extracts the AP’s nonce

d) If PMKSA caching is not used, verified public key from the Authentication frame.

e) If PMKSA caching is not used, compute the Diffie-Hellman shared secret, DHss, based on the AP’s ephemeral public key and its own private key with the chosen group’s scalar-op to derive DHss.

f) Performs key derivation (see 12.12.2.5 (Key establishment with FILS authentication)) and begins

key confirmation (see 12.12.2.6 (Key confirmation with FILS authentication)).

If the STA was attempting PMKSA caching and did not receive an Authentication frame from the AP, the

STA may attempt to use an alternate authentication method.

**9.3.3.12 Authentication frame format**

**Table 9-36—Presence of fields and elements in Authentication frames**

*Modify the rightmost column of the FILS Public Key authentication transaction seq# 2 row of the table as shown (REVmd/D0.1 page 758 line 22):*

The Finite Cyclic Group is present if Status Code field is 0 and PMKSA caching is not used.

The FFE field is present if Status Code field is 0 and PMKSA caching is not used.

The RSNE is present.

**12.12.2.6 Key confirmation with FILS authentication**

**12.12.2.6.2 (Re)Association Request for FILS key confirmation**

*Modify as shown (REVmd/D0.1 page 2237):*

The STA constructs a (Re)Association Request frame for FILS authentication per 9.3.3.6 (Association

Request frame format) and 9.3.3.8 (Reassociation Request frame format). Hash functions are used to

generate the FILS Key Confirmation element and the specific hash function depends on the AKM negotiated

(9.4.2.25.3 (AKM suites)).

For FILS Shared Key authentication and FILS Public Key authentication when using PMKSA caching, the KeyAuth field of the FILS Key Confirmation element is constructed by using the HMAC mode of the negotiated hash function with a key of ICK on a concatenation of the STA’s nonce, the AP’s nonce, the STA’s MAC address, the AP’s BSSID, and conditionally the STA’s public Diffie-Hellman value and the AP’s public Diffie-Hellman value, in that order:

Key-Auth = HMAC-Hash(ICK, SNonce || ANonce || STA-MAC || AP-BSSID [ || gSTA || gAP ])

where

Hash is the hash function specific to the negotiated AKM

SNonce is the STA’s nonce

ANonce is the AP’s nonce

STA-MAC is the MAC address of the STA and AP-BSSID is the BSSID of the AP

gSTA is the STA’s Diffie-Hellman public value

gAP is the AP’s Diffie-Hellman public value

Brackets indicate the inclusion of the Diffie-Hellman public values when doing PFS with FILS Shared Key authentication; there are no Diffie-Hellman public values to include otherwise

For FILS Public Key authentication when PMKSA caching is not used, the KeyAuth field of the FILS Key Confirmation element is a digital signature using the STA’s private key, of the negotiated hash function on a concatenation of the STA’s public Diffie-Hellman value, the AP’s public Diffie-Hellman value, the STA’s nonce, the AP’s nonce, the STA’s MAC address, and the AP’s BSSID, in that order:

Key-Auth = Sig-STA (gSTA || gAP || SNonce || ANonce || STA-MAC || AP-BSSID)

where

Sig-STA ( ) indicates a digital signature using the STA’s private key, analog to the STA’s trusted public key

The form of signature depends on the type of public key used by the STA (IETF RFC 3447 for RSA, FIPS

186-4 for DSA, and ISO/IEC 14888-3 for ECDSA). The data to be signed is first hashed and the hash

algorithm used with the appropriate digital signature algorithm shall be specific to the negotiated AKM.

**12.12.2.6.3 (Re)Association Response for FILS key confirmation**

*Modify as shown (REVmd/D0.1 page 2238):*

The AP constructs a (Re)Association Response frame for FILS authentication per 9.3.3.7 (Association

Response frame format) and 9.3.3.9 (Reassociation Response frame format). As with the (Re)Association

Request frame, hash functions are used to generate the FILS Key Confirmation element and the specific

hash function depends on the AKM negotiated (see9.4.2.25.3 (AKM suites)).

The AP constructs a Key Delivery element indicating the current GTK and Key RSC, the current IGTK and

IPN if management frame protection is enabled. The GTK is carried in a GTK KDE with Tx subfield equal

to 0. The IGTK and IPN are carried in an IGTK KDE. The AP puts this element into the (Re)Association

Response frame.

For FILS Shared Key authentication and FILS Public Key authentication when using PMKSA caching, the KeyAuth field of the FILS Key Confirmation element is constructed by using the HMAC mode of the negotiated hash function with a key of ICK on a concatenation of the AP’s nonce, the STA’s nonce, the AP’s BSSID, the STA’s MAC address, and conditionally the AP’s public Diffie-Hellman value and the STA’s public Diffie-Hellman value, in that order:

Key-Auth = HMAC-Hash(ICK, ANonce || SNonce || AP-BSSID || STA-MAC [ || gAP || gSTA ])

where

Hash is the hash function specific to the negotiated AKM

ANonce is the AP’s nonce

SNonce is the STA’s nonce

AP-BSSID is the BSSID of the AP and STA-MAC is the MAC address of the STA

gAP is the AP’s Diffie-Hellman public value

gSTA is the STA’s Diffie-Hellman public value

Brackets indicate the inclusion of the Diffie-Hellman public values when doing PFS with FILS Shared Key authentication; there are no Diffie-Hellman public values to include otherwise

For FILS Public Key authentication when PMKSA caching is not used, the KeyAuth field of the FILS Key Confirmation element is a digital signature using the AP’s private key of the output from the negotiated hash function on a concatenation of the AP’s public Diffie-Hellman value, the STA’s public Diffie-Hellman value, the AP’s nonce, the STA’s nonce, AP’s BSSID, and the STA’s MAC address, in that order. The specific construction of the digital signature depends on the crypto-system of the public/private keypair:

Key-Auth = Sig-AP (gAP || gSTA || ANonce || SNonce || AP-BSSID || STA-MAC)

where

Sig-AP () indicates a digital signature using the AP’s private key analog to the AP’s trusted public key

The form of signature depends on the type of public key used by the AP (IETF RFC 3447 for RSA, FIPS

186-4 for DSA, and ISO/IEC 14888-3 for ECDSA). The data to be signed is first hashed and the hash

algorithm used with the appropriate digital signature algorithm shall be specific to the negotiated AKM.