IEEE P802.11  
Wireless LANs

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 11bi D0.4 CR for 1148 | | | | |
| Date: 2024-09-30 | | | | |
| Author(s): | | | | |
| Name | Affiliation | Address | Phone | Email |
| Po-Kai Huang | Intel |  |  | po-kai.huang@intel.com |
| Ido Ouzieli |  |  |  |  |
| Johannes Berg |  |  |  |  |
| Ilan Peer |  |  |  |  |

Abstract

This submission proposes resolutions for the following CIDs:

1148

Revisions:

* Rev 0: Initial version of the document.

Interpretation of a Motion to Adopt

A motion to approve this submission means that the editing instructions and any changed or added material are actioned in the TGbi D0.6 Draft. This introduction is not part of the adopted material.

***Editing instructions formatted like this are intended to be copied into the TGbi D0.6 Draft. (i.e. they are instructions to the 802.11 editor on how to merge the text with the baseline documents). TGbi Editor: Editing instructions preceded by “TGbi Editor” are instructions to the TGbi editor to modify existing material in the TGbi draft. As a result of adopting the changes, the TGbi editor will execute the instructions rather than copy them to the TGbi Draft.***

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **CID** | **Commenter** | **Clause** | **P.L** | **Comment** | **Proposed Change** | **Resolution** |
| 1148 | Po-Kai Huang | 12.14.7 | 80.07 | Need to add procedures for PMKSA caching. There are two options for this. We can use PASN like flows, which requires 3 authentication message exchange. We can also use a new algorithm number and do like FT with only two authentication message exchange. Apparently, 2 message exchagne is better than 3 message exchange. | As in comment | Revised –  Agree in principle with the commenter. For 802.1X, we propose to put PMKID directly in the authentication frame for specific authentication. As a result, if no PMKID is identified, then 802.1X authentication can continue.  This is better than introducing new authentication algorithm number or PASN flow because if no PMKID is identified, then we waste two message exchange.  PMKSA caching for SAE also already exists in the PASN flow which is reused by EDPKE for SAE. Over there it also already has the capability to include PMKID in the first authentication and continue SAE authentication if PMKSA is not identified.  TGbi editor to make the changes shown in 11-24/1678r0 under all headings that include CID 1148 |

**Discussion:**

Currently, PMKID is indicated in (re)association request frame after exchange of open authentication. Given that 11bi needs to encrypt (re)association request/response frame, indicating PMKID in in (re)association request frame is obviously too late and we need to indicate PMKID in authentication frame.

For 802.1X, the most straightforward place is to put PMKID(s) in the first IEEE 802.1X Authentication frame.

* If authentication responder can identify PMKSA with one of the PMKID, then the authentication responder responds PMKID in the second Authentication frame, and the (re)association request/response exchange can continue
* If authentication responder cannot identify PMKSA with any of the indicted PMKID, then the authentication responder will not respond PMKID in the second Authentication frame, and 802.1X authentication can continue

An example for the case that an PMKSA can be identified is shown below.



An example for the case that an PMKSA cannot be identified is shown below.



For SAE, existing PASN flow, which is used by EPDKE, already has similar mechanism for PMKSA caching. It also already has the capability to continue SAE authentication if PMKSA is not identified by PMKID. We simply update the texts to clearly allow PMKSA caching for SAE in EPDKE.

***12.13.3 Key establishment with PASN authentication***

***12.13.3.1 Overview***

*Upon receiving the first PASN frame, the AP:*

* …(existing texts)….
* *Verifies the public key as specified in 5.6.2.3 of NIST SP 800-56A R2. If verification fails, the processing status is set to INVALID\_PUBLIC\_KEY.* *Verifies that a PMKSA named via a PMKID in the RSNE exists for the specified base AKMP, or the base AKMP is set to PASN AKMP* ***or base AKMP data exists in the frame to allow a PMK to be established.*** *If base AKMP is equal to PASN AKMP, verifies that dot11NoAuthPASNActivated is set to true. Otherwise processing status is set to REFUSED.*

**Proposed Texts: (#1148)**

**TGbi Editor: *Instruction: Modify 4.10.7 as shown below***

**4.10.7 PMKSA caching**

***Change the second paragraph as follows (not all lines are shown):***

A STA can supply a list of PMK identifiers in the (Re)Association Request frame or first FILS

Authentication frame or first IEEE 802.1X Authentication frame or first EDPKE Authentication frame. Each PMK identifier names a PMKSA. The Authenticator can specify the selected PMK identifier in message 1 of the 4-way handshake or the second FILS Authentication frame or the second IEEE 802.1X Authentication frame or the second EDPKE Authentication frame. The selection of the (#3493)PMK identifiers to be included by the STA and Authenticator is out of the scope of this standard.

**TGbi Editor: *Instruction: Modify 12.14.7.2 as shown below***

12.14.7 Key derivation with Authentication frame exchange

* IEEE(#Ed) 802.1X

If an authentication originator or an authentication responder defined in 12.14.4 (IEEE 802.1X authentication utilizing Authentication frames) sets the (Re)Association Frame Encryption Support(#1488) field in the RSNXE to 1, then the authentication originator or the authentication responder supports the additional rules defined in this subclause when performing IEEE(#Ed) 802.1X Authentication frame exchange.

An authentication originator that sets the (Re)Association Frame Encryption(#1488) Support field in the RSNXE to 1, has the (#1484)SME to act as the Supplicant, receives the RSNXE from the authentication responder with the (Re)Association Frame Encryption(#1488) Support field(#1474) set to 1, and intends to continue association after authentication shall:

* Include a Nonce element in the first Authentication frame to indicate SNonce.
* Include an RSNE in the first Authentication frame to indicate AKM and pairwise cipher suite. Version field shall be set to 1. Pairwise Cipher Suite Count field shall be set to 1. AKM Suite Count field shall be set to 1. PMKID count and PMKID list set corresponding to PMKSA identifiers if exists. All other fields shall be as specified in 9.4.2.23 (RSNE) and 12.6.3 (RSNA policy selection in an infrastructure BSS).(#1154)
* Not include an AKM Suite Selector element.
* Include an RSNXE in the first Authentication frame.
* Include a Diffie-Hellman Parameter element in the first Authentication frame.
* Select a(#1476) finite cyclic group in the Diffie-Hellman Parameter element from the dot11RSNAConfigDLCGroupTable that is at least of the security strength provided by the AKM and cipher suites.
* With the chosen finite cyclic group, generate an ephemeral (random) private key, use the selected group's scalar operation (see 12.4.4.1 (General)) with the private key to generate its ephemeral public key, and indicate the ephemeral public key in the Diffie-Hellman Parameter element.

Otherwise, an authentication originator shall not include a Diffie-Hellman Parameter element or an RSNE or an RSNXE or a Nonce element in the first Authentication frame for IEEE(#Ed) 802.1X authentication.

For the purpose of interoperability, an authenticator or a supplicant shall support group 19, an ECC group defined over a 256-bit prime order field.

An authentication responder that sets the (Re)Association Frame Encryption(#1488) Support field in the RSNXE to 1, has the (#1484)SME to act as the Authenticator, and receives the first Authentication frame with a Nonce element, RSNE, RSNXE, and a Diffie-Hellman Parameter element shall:

* Verify that the AKM indicated in the RSNE rather than AKM suite selector element as defined in 12.4.4 (IEEE 802.1X authentication utilizing Authentication frames) is supported. Otherwise, the authentication responder shall reject message 1 with status code set to STATUS\_INVALID\_AKMP.
* Verify that the pairwise cipher indicated in the RSNE is supported. Otherwise, the authentication responder shall reject message 1 with status code set to STATUS\_INVALID\_PAIRWISE\_CIPHER.
* Validate that the finite cyclic group indicated in the Diffie-Hellman Parameter element in the first Authentication frame is supported (present in dot11RSNAConfigDLCGroupTable). Otherwise, the authentication responder shall reject message 1 with status code set to UNSUPPORTED\_FINITE\_CYCLIC\_GROUP.
* Verify the public key indicated in the Diffie-Hellman Parameter element in message 1 as specified in 5.6.2.3 of NIST SP 800-56A R2. If verification fails, the authentication responder shall reject the first Authentication frame with status code set to INVALID\_PUBLIC\_KEY.
* Verify that a PMKSA named via a PMKID in the RSNE exists for the specified AKM
* If a PMKSA is identified, use PMKSA caching, does not process the EAPOL PDU in the first Authentication frame, and does not include EAPOL PDU in the second authentication frame.
* If no PMKSA is identified, continue the IEEE 802.1X authentication.
* If the first Authentication frame is not rejected, store the indicated SNonce and generate an ephemeral (random) private key with the chosen finite cyclic group and use the selected group's scalar operation with the private key to generate its ephemeral public key. Perform the group's scalar-op (see 12.4.4.1 (General)) with the authentication originator's ephemeral public key and its own ephemeral private key to produce an ephemeral Diffie-Hellman shared secret, DHss.
* Include an RSNE in the second Authentication frame to indicate the AKM and pairwise cipher indicated in the first Authentication frame.
* If a PMKSA is identified, include the PMKID corresponding to the PMKSA in the RSNE.
* Otherwise, does not include any PMKID in the RSNE.
* Not include an AKM Suite Selector element in the second Authentication frame.
* Include a Diffie-Hellman Parameter element in the second Authentication frame.
* Indicate chosen finite cyclic group in the Diffie-Hellman Parameter element of the second Authentication frame, which is the same as the finite cyclic group in the Diffie-Hellman Parameter element of the first Authentication frame.
* Indicate its ephemeral public key in the Diffie-Hellman Parameter element of the second Authentication frame.
* Include a Nonce element in the second Authentication frame to indicate ANonce.
* If a PMKSA is identified, use PMKSA caching and before sending the second Authentication frame, an authentication responder shall:
* Derive PTK with the identified PMKSA and DHss as defined in 12.7.1.3 (Pairwise key hierarchy).
* Irretrievably delete the shared secret, DHss, upon completion of PTK generation.

Otherwise, an authentication responder shall not include a Diffie-Hellman Parameter element or a Nonce element or an RSNE in the second Authentication frame for IEEE(#Ed) 802.1X authentication.

After receiving the second Authentication frame with the status code set to SUCCESS, an authentication originator shall:

* If the authentication originator includes a Diffie-Hellman Parameter element in the first Authentication frame, validate that there is a Diffie-Hellman Parameter element and an RSNE included in the second Authentication frame and there is no AKM suite selector element in the second Authentication frame. If the validation fails, the authentication originator shall discard the frame and terminate further protocol processing.
* If the authentication originator does not include a Diffie-Hellman Parameter element in the first Authentication frame, validate that there is no Diffie-Hellman Parameter element and no RSNE included in the second Authentication frame. If the validation fails, the authentication originator shall discard the frame and terminate further protocol processing.
* If the authentication originator includes a Diffie-Hellman Parameter element in the first Authentication frame, validate that the finite cyclic group indicated in the Diffie-Hellman Parameter element in the second Authentication frame is the same as the finite cyclic group indicated in the Diffie-Hellman Parameter element in the first Authentication frame, validate that the pairwise cipher suite and the AKM indicated in the second Authentication frame are the same as the pairwise cipher suite and the AKM indicated in the first Authentication frame. The validation of AKM is based on the AKM indication in RSNE rather than AKM suite selector element as defined 12.14.4 (IEEE 802.1X authentication utilizing Authentication frames). If the validation fails, the authentication originator shall discard the frame and terminate further protocol processing.
* Verify the public key indicated in the Diffie-Hellman Parameter element in the second Authentication frame as specified in 5.6.2.3 of NIST SP 800-56A R2. If verification fails, the authentication originator shall discard the frame and terminate further protocol processing.
* If the authentication originator includes one or more PMKID in the first Authentication frame, and the second Authentication frame includes a PMKID, validate that the Encapulation Length field is set to 0 and validate that the PMKID included in the second Authentication frame matches one of the PMKID(s) indicated in the first Authentication frame. If verification succeeds, use PMKSA caching with the PMKSA identified by the PMKID indicated in the second Authentication frame and does not continue the IEEE 802.1X Authentication frame exchange. If verification fails, the authentication originator shall discard the frame and terminate further protocol processing.
* If the authentication originator does not include any PMKID in the first Authentication frame, validate that there is no PMKID included in the second Authentication frame. If verification fails, the authentication originator shall discard the frame and terminate further protocol processing.
* If the second Authentication frame is not discarded, store the indicated ANonce, perform the group's scalar-op (see 12.4.4.1 (General)) with the authentication originator's ephemeral public key and its own ephemeral private key to produce an ephemeral Diffie-Hellman shared secret, DHss.
* If a PMKSA is identified, an authentication originator shall:
* Derive PTK with the identified PMKSA and DHss as defined in 12.7.1.3 (Pairwise key hierarchy).
* Irretrievably delete the shared secret, DHss, upon completion of PTK generation.

If a PMKSA is not identified due to PMKSA caching, before sending the Authentication frame carrying EAP Success, an authentication responder shall:

* Derive PTK with DHss as defined in 12.7.1.3 (Pairwise key hierarchy).
* Irretrievably delete the shared secret, DHss, upon completion of PTK generation.(#1483)

If a PMKSA is not identified due to PMKSA caching, after receiving the Authentication frame carrying EAP Success, an authentication originator shall:

* Derive PTK with DHss as defined in 12.7.1.3 (Pairwise key hierarchy).
* Irretrievably delete the shared secret, DHss, upon completion of PTK generation.(#1483)

The authentication originator and the authentication responder then continue the operation as defined in 12.14.5 ((Re)Association Request/Response Frame Encryption(#1488)) with the following additional rules:

* Authentication responder shall verify that the RSNE other than the PMKID Count field and the PMKID list field in the (Re)Association Request frame is identical to(#1154) the RSNE included in the first Authentication frame. Authentication responder shall also verify that the RSNXE(#1153) in the (Re)Association Request is identical to(#1154) the RSNXE included in the first Authentication frame. If the validation fails, the authentication responder shall reject the association.
* Authentication originator shall verify that the RSNE other than the PMKID Count field and the PMKID list field in the (Re)Association Response frame is the same as the RSNE included in the second Authentication frame. If the validation fails, the authentication originator shall disassociate.

**TGbi Editor: *Instruction: Modify 12.14.4 as shown below***

**12.14.4 IEEE 802.1X authentication utilizing Authentication frames**

(…existing texts…)

The authentication originator then constructs the first Authentication frame of the exchange as follows:

* Authentication Algorithm Number field is set to <ANA> (IEEE 802.1X authentication).
* Authentication Transaction Sequence Number field is set to 1.(#1436)
* The Encapsulation field carries an(#1429) EAPOL PDU.
* Include(#1430) the AKM Suite Selector element indicating the selected IEEE(#1181) 802.1X AKM.

NOTE - The authentication originator sends the EAPOL-Start Authentication frame as the first authentication frame to start the EAP authentication process.

(…existing texts…)

**TGbi Editor: *Instruction: Modify 9.4.2.23.3 as shown below***

**9.4.2.23.3 AKM suites**

***Modify Table 9-190 (AKM suite selectors) as follows:***

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| OUI | Suite type | Meaning | | | Authentication algorithm numbers  (see 9.4.1.1 (Authentication Algorithm Number field)) | Cipher suite selector restriction |
| Authentication  type | Key management  type | Key derivation type |
| 00-0F-AC | 0 | Reserved | Reserved | Reserved | Reserved | Reserved |
| 00-0F-AC | 1 | Authentication negotiated over  IEEE Std 802.1X | RSNA key management as defined in 12.7 (Keys and key distribution) | Defined in 12.7.1.2 (PRF) | 0 (open) or <ANA> (IEEE 802.1X) (#1149) | None |
| 00-0F-AC | 2 | PSK | RSNA key management as defined in 12.7 (Keys and key distribution) | Defined in 12.7.1.2 (PRF) | 0 (open) | None |
| 00-0F-AC | 3 | FT authentication negotiated over  IEEE Std 802.1X | FT key management as defined in 12.7.1.6 (FT key hierarchy) | Defined in 12.7.1.6.2 (Key derivation function (KDF))  using SHA-256 | 2 (FT) for FT protocol reassociation as defined in 13.5 (FT protocol)  0 (open) or <ANA> (IEEE 802.1X) (#1149)  for FT Initial Mobility Domain Association over  IEEE Std 802.1X or  PMKSA caching | None |
| 00-0F-AC | 4 | FT authentication using PSK | FT key management as defined in 12.7.1.6 (FT key hierarchy) | Defined in 12.7.1.6.2 (Key derivation function (KDF))  using SHA-256 | 2 (FT) for FT protocol reassociation as defined in 13.5 (FT protocol)  0 (open) for FT Initial Mobility Domain Association using PSK | None |
| 00-0F-AC | 5 | Authentication negotiated over  IEEE Std 802.1X | RSNA key management as defined in 12.7 (Keys and key distribution) | Defined in 12.7.1.6.2 (Key derivation function (KDF))  using SHA-256 | 0 (open) or <ANA> (IEEE 802.1X) (#1149) | None |
| 00-0F-AC | 6 | PSK | RSNA Key Management as defined in 12.7 (Keys and key distribution) | Defined in 12.7.1.6.2 (Key derivation function (KDF))  using SHA-256 | 0 (open) | None |
| 00-0F-AC | 7 | TDLS | TPK handshake | Defined in 12.7.1.6.2 (Key derivation function (KDF))  using SHA-256 | N/A | None |
| 00-0F-AC | 8 | SAE authentication | RSNA key management as defined in 12.7 (Keys and key distribution), or authenticated mesh peering exchange as defined in 14.6 (Authenticated mesh peering exchange (AMPE)) | Defined in 12.7.1.6.2 (Key derivation function (KDF))  using the hash algorithm specified in 12.4.2 (Assumptions on SAE) | 3 (SAE) for SAE Authentication  0 (open) for PMKSA caching | None |
| 00-0F-AC | 9 | FT authentication over SAE | FT key management defined in 12.7.1.6 (FT key hierarchy) | Defined in 12.7.1.6.2 (Key derivation function (KDF))  using the hash algorithm specified in 12.4.2 (Assumptions on SAE) | 3 (SAE) for FT Initial Mobility Domain Association  2 (FT) for FT protocol reassociation as defined in 13.5 (FT protocol)  0 (open) for FT Initial Mobility Domain Association over PMKSA caching | None |
| 00-0F-AC | 10 | APPeerKey Authentication with SHA-256 | RSNA key management as defined in 12.7 (Keys and key distribution) | Defined in 12.7.1.6.2 (Key derivation function (KDF))  using SHA-256 | N/A | None |
| 00-0F-AC | 11 | Authentication negotiated over  IEEE Std 802.1X using a Suite B compliant EAP method supporting SHA-256 | RSNA key management as defined in 12.7 (Keys and key distribution) | Defined in 12.7.1.6.2 (Key derivation function (KDF))  using SHA-256 | 0 (open) or <ANA> (IEEE 802.1X) (#1149) | Used only with cipher suite selector values 00-0F-AC:8 (GCMP-128) and 00-0F-AC:11 (BIP-GMAC-128) |
| 00-0F-AC | 12 | Authentication negotiated over  IEEE Std 802.1X using a CNSA Suite compliant EAP method | RSNA key management as defined in 12.7 (Keys and key distribution) | Defined in 12.7.1.6.2 (Key derivation function (KDF))  using SHA-384 | 0 (open) or <ANA> (IEEE 802.1X) (#1149) | Used only with cipher suite selector values 00-0F-AC:9 (GCMP-256), 00-0F-AC:10 (CCMP-256), 00-0F-AC:13 (BIP-CMAC-256), and 00-0F-AC:12 (BIP-GMAC-256) |
| 00-0F-AC | 13 | FT authentication negotiated over  IEEE Std 802.1X | FT key management as defined in 12.7.1.6 (FT key hierarchy) | Defined in 12.7.1.6.2 (Key derivation function (KDF))  using SHA-384 | 2 (FT) for FT protocol reassociation as defined in 13.5 (FT protocol)  0 (open) or <ANA> (IEEE 802.1X) (#1149) for FT Initial Mobility Domain Association over  IEEE Std 802.1X or  PMKSA caching | Used only with cipher suite selector values 00-0F-AC:9 (GCMP-256), 00-0F-AC:10 (CCMP-256), 00-0F-AC:13 (BIP-CMAC-256), and 00-0F-AC:12 (BIP-GMAC-256) |
| 00-0F-AC | 14 | Key management over FILS using SHA-256 and  AES-SIV-256, or authentication negotiated over  IEEE Std 802.1X | FILS key management defined in 12.11.2.5 (Key establishment with FILS authentication) | Defined in 12.11.2.5 (Key establishment with FILS authentication)  using SHA-256 | 4, 5 or 6 (FILS) for FILS Authentication  0 (open) or <ANA> (IEEE 802.1X) (#1149) for  IEEE Std 802.1X | None |
| 00-0F-AC | 15 | Key management over FILS using SHA-384 and  AES-SIV-512, or authentication negotiated over  IEEE Std 802.1X | FILS key management defined in 12.11.2.5 (Key establishment with FILS authentication) | Defined in 12.11.2.5 (Key establishment with FILS authentication)  using SHA-384 | 4, 5 or 6 (FILS) for FILS Authentication  0 (open) or <ANA> (IEEE 802.1X) (#1149) for  IEEE Std 802.1X | None |
| 00-0F-AC | 16 | FT authentication over FILS with  SHA-256 and  AES-SIV-256 or authentication negotiated over  IEEE Std 802.1X | FT key management as defined in 12.7.1.6 (FT key hierarchy) | Defined in 12.7.1.6.2 (Key derivation function (KDF))  using SHA-256 | 4, 5 or 6 (FILS) for FT Initial Mobility Domain Association over FILS  2 (FT) for FT protocol reassociation as defined in 13.5 (FT protocol)  0 (open) or <ANA> (IEEE 802.1X) (#1149) for FT Initial Mobility Domain Association over  IEEE Std 802.1X orPMKSA caching | None |
| 00-0F-AC | 17 | FT authentication over FILS with  SHA-384 and  AES-SIV-512, or authentication negotiated over  IEEE Std 802.1X | FT key management as defined in 12.7.1.6 (FT key hierarchy) | Defined in 12.7.1.6.2 (Key derivation function (KDF))  using SHA-384 | 4, 5 or 6 (FILS) for FT Initial Mobility Domain Association over FILS  2 (FT) for FT protocol reassociation as defined in 13.5 (FT protocol)  0 (open) or <ANA> (IEEE 802.1X) (#1149) for FT Initial Mobility Domain Association over  IEEE Std 802.1X or  PMKSA caching | None |
| 00-0F-AC | 18 | None | RSNA key management as defined in 12.7 (Keys and key distribution) | Defined in 12.7.1.6.2 (Key derivation function (KDF)) using the hash algorithm specified in Table 12-1 | 0 (open) | None |
| 00-0F-AC | 19 | FT authentication using PSK | FT key management as defined in 12.7.1.6 (FT key hierarchy) | Defined in 12.7.1.6.2 (Key derivation function (KDF))  using SHA-384 | 2 (FT) for FT protocol reassociation as defined in 13.5 (FT protocol)  0 (open) for FT Initial Mobility Domain Association using PSK | None |
| 00-0F-AC | 20 | PSK | RSNA key management as defined in 12.7 (Keys and key distribution) | Defined in 12.7.1.6.2 (Key derivation function (KDF))  using SHA-384 | 0 (open) | None |
| 00-0F-AC | 21 | PASN | PASN | PASN key management defined in 12.13 (Preassociation security negotiation(11az)) | Defined in 12.13.3 (Key establishment with PASN authentication) | None |
| 00-0F-AC | 22 | FT authentication negotiated over  IEEE Std 802.1X | FT key management as defined in 12.7.1.6 (FT key hierarchy) | Defined in 12.7.1.6.2 (Key derivation function (KDF))  using SHA-384 | 2 (FT) for FT protocol reassociation as defined in 13.5 (FT protocol)  0 (open) or <ANA> (IEEE 802.1X) (#1149) for FT Initial Mobility Domain Association over  IEEE Std 802.1X or  PMKSA caching | None |
| 00-0F-AC | 23 | Authentication negotiated over  IEEE Std 802.1X | RSNA key management as defined in 12.7 (Keys and key distribution) | Defined in 12.7.1.6.2 (Key derivation function (KDF)) using SHA-384 | 0 (open) or <ANA> (IEEE 802.1X) (#1149) | None |
| 00-0F-AC | 24 | SAE authentication | RSNA key management as defined in 12.7 (Keys and key distribution), or authenticated mesh peering exchange as defined in 14.6 (Authenticated mesh peering exchange (AMPE)) | Defined in 12.7.1.6.2 (Key derivation function (KDF))  using the hash algorithm specified in 12.4.2 (Assumptions on SAE) | 3 (SAE) for SAE Authentication  0 (open) for PMKSA caching | None |
| 00-0F-AC | 25 | FT authentication over SAE | FT key management defined in 12.7.1.6 (FT key hierarchy) | Defined in 12.7.1.6.2 (Key derivation function (KDF))  using the hash algorithm specified in 12.4.2 (Assumptions on SAE) | 3 (SAE) for FT Initial Mobility Domain Association  2 (FT) for FT protocol reassociation as defined in 13.5 (FT protocol)  0 (open) for FT Initial Mobility Domain Association over PMKSA caching | None |
| 00-0F-AC | <AN A> | EDPKE | EDPKE | EDPKE key management defined in 12.14.8 (Enhanced Data Privacy Key Exchange) | Defined in 12.14.8.3 (Key establishment with EDPKE authentication) |  |
| 00-0F-AC | 26–255 | Reserved | Reserved | Reserved | Reserved | Reserved |
| Other OUI or CID | Any | Vendor-specific | Vendor-specific | Vendor-specific | Vendor-specific | Vendor-specific |

**TGbi Editor: *Instruction: Modify 12.6.8.3 as shown below***

**12.6.8.3 Cached PMKSAs and RSNA key management**

A STA might cache PMKSAs it establishes as a result of previous authentication. The PMKSA shall not be changed while cached. The PMK in the PMKSA is used with the 4-way handshake or FILS authentication to establish fresh PTKs or PTKSA derivation with IEEE 802.1X Authentication frame exchange as defined in 12.14.7.2 or EDPKE authentication as defined in 12.14.8.

If a STA in an infrastructure BSS has determined it has a valid PMKSA with an AP to which it is about to (re)associate, it performs Open System authentication to the AP, or performs FILS Authentication to the AP or performs PTKSA derivation with IEEE 802.1X Authentication frame exchange or performs EDPKE authentication, or (only in the case of a DMG STA) does not perform IEEE 802.11 authentication. When Open System authentication is used or IEEE 802.11 authentication is not performed, it includes the PMKID for the PMKSA in the RSNE in the (re)association request. When FILS Authentication is performed, it includes the PMKID for the PMKSA in the FILS Authentication frame (see 12.11.2 (FILS authentication protocol)). When PTKSA derivation with IEEE 802.1X Authentication frame exchange and PMKSA caching is performed, it includes the PMKID for the PMKSA in the first 802.1X Authentication frame. When EDPKE authentication with PMKSA caching is performed, it includes the PMKID for the PMKSA in the first EDPKE Authentication frame. When the PMKSA was not created using preauthentication, the AKM indicated in the RSNE by the STA in the (re)association request shall be identical to the AKM used to establish the cached PMKSA in the first place.

(…existing texts….)

If a cached PMKSA is used in FT Initial Mobility Domain Association, the cached MPMK is used to derive the PMK-R0 of a new FT key hierarchy (see 12.7.1.6 (FT key hierarchy)). The PMKID indicated by the STA in the FILS Authentication frame (when FILS authentication is used) or IEEE 802.1X Authentication frame (when PTKSA derivation with IEEE 802.1X Authentication frame exchange is used) or EDPKE Authentication frame (when EDPKE authentication is used) or (Re)Association Request frame and message 1 of the FT 4-way handshake (otherwise) is the PMKID of the cached PMKSA as defined in 12.7.1.6.3 (PMK-R0) (i.e., not the PMKR0Name or PMKR1Name of the FT key hierarchy that was derived when the PMKSA was originally established). The PMKR1Name indicated in (#6299)the RSNE in messages 2 and 3 of the FT 4‑way handshake (when FILS authentication is not used) or in (Re)Association Request and Response frames (when FILS authentication is used) is the PMKR1Name of the newly derived FT key hierarchy (see 13.4 (FT initial mobility domain association)).

(…existing texts….)

Upon receipt of a FILS Authentication frame with one or more PMKIDs, an AP checks whether its Authenticator has cached a PMKSA for the PMKIDs, whether the AKM in the cached PMKSA matches the AKM in the FILS Authentication frame, and whether the PMK is still valid; and if so, it shall assert possession of that PMK by including the PMKID in the FILS Authentication frame sent in response (see 12.11.2.3.4 (AP construction of Authentication frame)). If the Authenticator does not have a PMK for the PMKIDs in the FILS Authentication frame or the AKM does not match, the AP may either reply with (#330)an EAP-Finish/Reauth packet to continue FILS Shared Key authentication option if the non-AP STA included sufficient information for that, or the AP rejects the authentication.

The procedure for the PTKSA derivation with IEEE 802.1X Authentication frame exchange and PMKSA caching is defined in 12.14.7.2.

The procedure for EDPKE authentication exchange and PMKSA caching is defined in 12.14.8. (…existing texts….)

If both sides assert possession of a cached PMKSA, but the 4-way handshake or FILS authentication or encrypted (re)association exchange with 802.1X Authentication frame exchange or EDPKE authentication fails, both sides may delete the cached PMKSA for the selected PMKID.

(…existing texts….)