1 **11. Security**

2

3

# 4 11.4 RSNA confidentiality and integrity protocols

5

6

## 7 11.4.4 Broadcast/Multicast Integrity Protocol (BIP)

8

9

## 10 11.4.4.4 BIP replay protection

11

### 12 *Change as follows:*

13

14

1. When management frame protection is negotiated, the receiver shall maintain a 48-bit replay counter for
2. each IGTK. The receiver shall set the receive replay counter to the value of the IPN in the IGTK key data
3. encapsulation (KDE) (see 11.6.2 (EAPOL-Key frames)) provided by the Authenticator in ~~either~~the 4-Way

18

1. Handshake, FT 4-Way Handshake, FT Handshake, ~~or~~Group Key Handshake, or FILS authentication. The
2. transmitter may reinitialize the sequence counter when the IGTK is refreshed. See 11.4.4.5 (BIP transmis-
3. sion) and 11.4.4.6 (BIP reception) for per packet BIP processing.

22

23

24

25

26

# 27 11.5 RSNA security association management

28

29

## 30 11.5.1 Security associations

31

## 32 11.5.1.1 Security association definitions

33

34

## 35 11.5.1.1.1 General

36

37

### 38 *Change as follows:*

39

40 — PMKSA: A result of a successful IEEE 802.lX exchange, SAE authentication, FILS authentication,

41

1. preshared PMK information, or PMK cached via some other mechanism.
2. — PTKSA: A result of a successful 4-Way Handshake, FT 4-Way Handshake, ~~or~~FT authentication

44

1. sequence, or FILS authentication.
2. — Mesh TKSA: A result of a successful authenticated mesh peering exchange (AMPE).

47

1. — GTKSA: A result of a successful Group Key Handshake, 4-Way Handshake, FT 4-Way Handshake,
2. FT authentication sequence, or FILS authentication.

50

1. — IGTKSA: A result of a successful Group Key Handshake, successful 4-Way Handshake, successful
2. FT 4-Way Handshake, the Reassociation Response message of the fast BSS transition protocol
3. when successful, or successful FILS authentication.

54

55

56

57

58

59

60

61

62

63

64

65

## 1 11.5.1.1.2 PMKSA

2

### 3 *Change as follows:*

4

5

1. When the PMKSA is the result of a successful IEEE 802.1X authentication, it is derived from the EAP
2. authentication and authorization parameters provided by the AS. When the PMKSA is the result of a suc-
3. cessful SAE authentication, it is generated as a result of the successful completion of the SAE exchange.
4. ~~This security~~A PMKSA association is bidirectional. In other words, both parties use the information in the

10

1. security association for both sending and receiving. The PMKSA is created by the Supplicant’s SME when
2. the EAP authentication or FILS authentication completes successfully or the PSK is configured. The
3. PMKSA is created by the Authenticator’s SME when the PMK is created from the keying information trans-
4. ferred from the ~~A~~S, ~~when~~ in an IEEE 802.1X authentication ~~is utilized~~exchange, when the FILS authentica-
5. tion completes successfully, when the SAE exchange successfully completes, or when the PSK is

16

1. configured. The PMKSA is used to create the PTKSA. PMKSAs are cached for up to their lifetimes. The
2. PMKSA consists of the following elements:

19

20 — PMKID, as defined in 11.6.1.3 (Pairwise key hierarchy). The PMKID identifies the security associa-

21

1. tion;
2. — Authenticator’s or peer’s MAC address. For multiband RSNA, the MAC address is associated with
3. the operating band in use when the PMKSA is established;

25

1. — PMK;
2. — Lifetime, as defined in 11.6.1.3 (Pairwise key hierarchy);

28

29 — AKMP;

30

1. — All authorization parameters specified by the AS or local configuration. This might include parame-
2. ters such as the STA’s authorized SSID.

33

34

35

36

## 37 11.5.1.1.6 PTKSA

38

### 39 *Change as follows:*

40

41

1. The PTKSA ~~is a~~ results from a successful ~~of the~~4-Way Handshake, FT 4-Way Handshake, FT Protocol, ~~or~~
2. FT Resource Request Protocol, or FILS authentication. This security association is also bidirectional. PTK-
3. SAs are cached for the life of the PMKSA or PMK-R1 security association. Because the PTKSA is tied to
4. the PMKSA or to a PMK-R1 security association, it only has the additional information from the 4-Way

46

1. Handshake or FILS authentication. For the PTKSA derived as a result of the 4-Way Handshake, there shall
2. be only one PTKSA per band (see 11.5.19 (Protection of robust Management frames)) with the same Suppli-
3. cant and Authenticator MAC addresses. For the PTKSA derived as a result of an initial mobility domain
4. association or fast BSS transition, there shall be only one PTKSA with the same STA's MAC address and
5. BSSID.

52

53

1. During the 4-Way Handshake defined in 11.6.6.5 (4-Way Handshake Message 4) and the FT 4-Way Hand-
2. shake defined in 12.4.2 (FT initial mobility domain association in an RSN), there is state created between
3. Message 1 and Message 3 of the Handshake. This does not create a PTKSA until Message 3 is validated by
4. the Supplicant and Message 4 is validated by the Authenticator.

58

59

1. During the FT authentication sequence defined in 12.8 (FT authentication sequence), the PTKSA is vali-
2. dated when Message 3 is validated by the R1KH and Message 4 is validated by the S1KH.

62

63

1. During the FILS authentication sequence defined in [11.11.2 (FILS authentication protocol)](#_bookmark6), the PTKSA is
2. validated by key confirmation using (Re)Association Request and (Re)Association Response frames.

1 The PTKSA consists of the following elements:

2

1. — PTK
2. — Pairwise cipher suite selector

5

1. — Supplicant MAC address or STA’s MAC address
2. — Authenticator MAC address or BSSID

8

9 — Key ID

10

1. — If FT key hierarchy is used,
2. — R1KH-ID

13

1. — S1KH-ID
2. — PTKName

16

17

18

19

## 20 11.5.1.1.8 GTKSA

21

22

### 23 *Change as follows:*

24

25 The GTKSA results from a successful 4-Way Handshake, FT 4-Way Handshake, FT Protocol, FT Resource

26

1. Request Protocol, ~~or the~~Group Key Handshake, or FILS authentication, and is unidirectional. In an infra-
2. structure BSS, there is one GTKSA, used exclusively for encrypting group addressed MPDUs that are trans-
3. mitted by the AP and for decrypting group addressed transmissions that are received by the STAs. In an
4. IBSS each STA defines its own GTKSA, which is used to encrypt its group addressed transmissions, and
5. stores a separate GTKSA for each peer STA so that encrypted group addressed traffic received from other

32

1. STAs may be decrypted. A GTKSA is created by the Supplicant's SME when Message 3 of the 4-Way
2. Handshake is received or when Message 1 of the Group Key Handshake is received or when a (Re)Associa-
3. tion Response frame of FILS authentication with a status code indicating success is received. The GTKSA is
4. created by the Authenticator's SME when the SME changes the GTK and has sent the GTK to all STAs with
5. which it has a PTKSA. A GTKSA consists of the following elements:

38

39

40

41

## 42 11.5.1.1.9 IGTKSA

43

44

### 45 *Change as follows:*

46

47 When management frame protection is enabled, a non-AP STA's SME creates an IGTKSA when it receives

48

1. a valid Message 3 of the 4-Way Handshake or FT 4-Way Handshake, the Reassociation Response message
2. of the fast BSS transition protocol with a status code indicating success, a Mesh Peering Open Message of
3. the Authenticated Mesh Peering Exchange (AMPE) protocol, ~~or~~ a valid Message 1 of the Group Key Hand-
4. shake, or the (Re)Association Response frame of FILS authentication with a status code indicating success.
5. The Authenticator's SME creates an IGTKSA when it establishes or changes the IGTK with all STAs to

54

55 which it has a valid PTKSA or MTKSA.

56

57

58

59

## 60 11.5.1.3.2 Security association in an ESS

61

### 62 *Change as follows:*

63

64

65 A STA and AP establish an initial security association via the following steps:

1 a) The STA selects an authorized ESS by selecting among APs that advertise an appropriate SSID.

2

1. b) The STA then performs IEEE Std 802.11 authentication followed by association to the chosen AP.
2. Confirmation of security parameters takes place during association. A STA performing IEEE
3. 802.1X authentication uses Open System authentication. A STA performing secure password-based,
4. or PSK, authentication uses SAE authentication. A STA performing FILS uses FILS authentication.

7

1. NOTE 1—It is possible for more than one PMKSA to exist. As an example, a second PMKSA might come into exis-
2. tence through PMKSA caching. A STA might leave the ESS and flush its cache. Before its PMKSA expires in the AP’s
3. cache, the STA returns to the ESS and establishes a second PMKSA from the AP’s perspective. 11
4. NOTE 2—An attack altering the security parameters is detected by the key derivation procedure.
5. NOTE 3—IEEE Std 802.11 Open System authentication provides no security, but is included to maintain backward
6. compatibility with the IEEE 802.11 state machine (see 10.3 (STA authentication and association)). 15
7. c) SAE authentication and FILS authentication provide mutual authentication and derivation of a
8. PMK. If Open System authentication is chosen instead, the Authenticator or the Supplicant initiates

18

1. IEEE 802.1X authentication. The EAP method used by IEEE Std 802.1X-2010 needs to support
2. mutual authentication, as the STA needs assurance that the AP is a legitimate AP.
3. NOTE 4—Prior to the completion of IEEE 802.1X authentication and the installation of keys, the IEEE 802.1X Con-
4. trolled Port in the AP blocks all data frames. The IEEE 802.1X Controlled Port returns to the unauthorized state and
5. blocks all data frames before invocation of an MLME-DELETEKEYS.request primitive. The IEEE 802.1X Uncon-
6. trolled Port allows IEEE 802.1X frames to pass between the Supplicant and Authenticator. Although IEEE Std 802.1X-
7. 20104 does not require a Supplicant Controlled Port, this standard assumes that the Supplicant has a Controlled Port in
8. order to provide the needed level of security. Supplicants without a Controlled Port compromise RSN security and are
9. not used. 28
10. NOTE 5—Any secure network cannot support promiscuous association, e.g., an unsecured operation of IEEE Std
11. 802.11. A trust relationship is needed between the STA and the AS of the targeted SSID prior to association and secure
12. operation, in order for the association to be trustworthy. The reason is that an attacker can deploy a rogue AP just as eas-
13. ily as a legitimate network provider can deploy a legitimate AP, so some sort of prior relationship is necessary to estab-
14. lish credentials between the ESS and the STA. 34
15. d) The last step is key management. The authentication process, whether SAE authentication or FILS
16. authentication utilizing Authentication frames or IEEE 802.1X authentication utilizing Data frames

37

1. post association, creates cryptographic keys shared between the cryptographic endpoints-the AP and
2. STA, or the IEEE 802.1X AS and the STA, when using SAE/FILS or IEEE 802.1X, respectively.
3. When using IEEE 802.1X the AS transfers these keys to the AP, and the AP and STA use one of the
4. key confirmation handshakes, e.g., the 4-Way Handshake or FT 4-Way Handshake, to complete
5. security association establishment. When using SAE authentication there is no AS and therefore no

43

1. key transfer; the 4-way Handshake is performed directly between the AP and STA. The key confir-
2. mation handshake indicates when the link has been secured by the keys and is ready to allow normal
3. data traffic and protected robust management frames. When FILS authentication is performed, the
4. key confirmation is performed as part of the FILS exchange using association frames. Hence, no
5. additional handshake is necessary.

49

50

1. When FT is not enabled, a STA roaming within an ESS establishes a new PMKSA by one of the five
2. schemes:

53

54

1. — In the case of (re)association followed by IEEE 802.1X or PSK authentication, the STA repeats the
2. same actions as for an initial contact association, but its Supplicant also deletes the PTKSA when it
3. roams from the associated AP. The Supplicant also deletes the PTKSA when it disassociates/deau-
4. thenticates from all BSSIDs in the ESS.

59

1. — In the case of SAE authentication followed by (re)association, the STA repeats the same actions as
2. for initial contact association, but the non-AP STA also deletes the PTKSA when it roams from the
3. associated AP. Note that a STA can take advantage of the fact that it can perform SAE authentication

63

1. to multiple APs while maintaining a single association with one AP, and then use any of the PMK-
2. SAs created during authentication to effect a fast BSS transition.
	1. — A STA can ~~retain~~ cache PMKSAs for STAs in the ESS to which it has previously performed a full
	2. IEEE Std 802.1X authentication or SAE authentication. If a STA wishes to roam to an AP for which
	3. it has cached one or more PMKSAs, it can include one or more PMKIDs in the RSNE of its

4

1. (Re)Association Request frame. An AP that has ~~retained~~ cached the PMKSA for one or more of the
2. PMKIDs can proceed with the 4-Way Handshake. The AP shall include the PMKID of the selected
3. PMKSA in Message 1 of the 4-Way Handshake. If none of the PMKIDs of the cached PMKSAs
4. matches any of the supplied PMKIDs, or if the AKM of the cached PMKSA differs from that offered
5. in the (Re)Association Request frame, or if the PMK in the cached PMKSA is no longer valid, then

10

1. the Authenticator, in the case of Open System authentication, shall perform another IEEE Std
2. 802.1X authentication and, in the case of SAE authentication, shall transmit a Deauthentication
3. frame to the STA. Similarly, if the STA fails to send a PMKID, the STA and AP need to perform a
4. full IEEE Std 802.1X authentication.

15

1. — A STA already associated with the ESS can request its IEEE Std 802.1X Supplicant to authenticate
2. with a target AP before associating to that target AP. The normal operation of the DS via the associ-
3. ated AP provides the communication between the STA and the target AP. The SME delays reassoci-

19

1. ation with the target AP until IEEE Std 802.1X authentication completes via the DS. If IEEE Std
2. 802.1X authentication completes successfully, then PMKSAs shared between the target AP and the
3. STA are cached, thereby enabling the possible usage of reassociation without requiring a subsequent
4. full IEEE Std 802.1X authentication procedure.

24

1. — In the case of FILS authentication, the STA may repeat the same actions as an initial contact and
2. authentication. The STA may also use a cached PMKSA to authenticate. A STA already associated

27

28 with the ESS can initiate FILS authentication to multiple other APs while associated.

29

30 The MLME-DELETEKEYS.request primitive destroys the temporal keys established for the security asso-

31

1. ciation so that they cannot be used to protect subsequent IEEE Std 802.11 traffic. An SME uses this primi-
2. tive when it deletes a PTKSA, GTKSA, or IGTKSA.

34

35

36

37

38

## 39 11.5.3 RSNA policy selection in an ESS

40

### 41 *Change as follows:*

42

43

1. An RSNA-enabled AP shall use Table 11-2 (Robust management frame selection in an ESS) and the values
2. of the Management Frame Protection Capable (MFPC) and Management Frame Protection Required

46

1. (MFPR) bits advertised in the RSNEs to determine if it may associate with a non-AP STA. An RSNA
2. enabled non-AP STA shall use Table 11-2 (Robust management frame selection in an ESS) and the values
3. of the Management Frame Protection Capable and Management Frame Protection Required bits advertised
4. in the RSNEs to determine if it may associate with an AP. Management frame protection is enabled when
5. dot11RSNAProtectedManagementFramesActivated is set to 1. Management frame protection is negotiated

52

1. when an AP and non-AP STA set the Management Frame Protection Capable field to 1 in their respective
2. RSNEs in the (re)association procedure, and both parties confirm the Management Frame Protection Capa-
3. ble bit set to 1 in the 4-Way Handshake, FT 4-Way Handshake, ~~or the~~ FT fast BSS transition protocol, or the
4. (Re)Association Request and (Re)Association Response frames of FILS authentication.

57

58

## 59 11.5.10 RSNA authentication in an ESS

60

61

## 62 11.5.10.1 General

63

64

### 65 *Change as follows:*

1. When establishing an RSNA in a non-FT environment or during an FT initial mobility domain association, a
2. STA shall use IEEE Std 802.11 SAE authentication, FILS authentication, or Open System authentication
3. prior to (re)association.

4

5

1. SAE authentication is initiated when a STA’s MLME-SCAN.confirm primitive finds another AP within the
2. current ESS that advertises support for SAE in its RSNE.

8

9 FILS authentication is initiated when a STA’s MLME-SCAN.confirm primitive finds an AP that advertises

10

11 support for FILS authentication in its RSNE.

12

13 IEEE 802.1X authentication is initiated by any one of the following mechanisms:

14

15 — If a STA negotiates to use IEEE 802.1X authentication during (re)association, the STA’s manage-

16

1. ment entity may respond to the MLME-ASSOCIATE.confirm (or indication) primitive by request-
2. ing the Supplicant (or Authenticator) to initiate IEEE 802.1X authentication. Thus, in this case,
3. authentication is driven by the STA’s decision to associate and the AP’s decision to accept the asso-
4. ciation.

21

1. If a STA’s MLME-SCAN.confirm primitive finds another AP within the current ESS, a STA may signal its Supplicant
2. to use IEEE Std 802.1X-2010 to preauthenticate with that AP.

24

1. NOTE—A roaming STA’s IEEE 802.1X Supplicant can initiate preauthentication by sending an EAPOL-Start message
2. via its associated AP, through the DS, to a target AP.
3. — If a STA receives an IEEE 802.1X message, it delivers this to its Supplicant or Authenticator, which

28

29 may initiate a new IEEE 802.1X authentication.

30

31

32

## 33 11.5.10.3 Cached PMKSAs and RSNA key management

34

35

### 36 *Change as follows:*

37

1. In a non-FT environment, a STA might ~~retain~~cache PMKSAs it establishes as a result of previous
2. authentication. The PMKSA cannot be changed while cached. The PMK in the PMKSA is used with the 4-

40

41 Way Handshake or FILS authentication to establish fresh PTKs.

42

1. If a STA in an ESS has determined it has a valid PMKSA with an AP to which it is about to (re)associate, it
2. performs Open System authentication to the AP, and then it includes the PMKID for the PMKSA in the
3. RSNE in the (Re)Association Request. When the PMKSA was not created using pre-authentication, the

46

1. AKM indicated in the RSNE by the STA in the (Re)Association Request shall be identical to the AKM used
2. to establish the cached PMKSA in the first place.

49

50 Upon receipt of a (Re)Association Request frame with one or more PMKIDs, an AP checks whether its

51

1. Authenticator has ~~retained~~cached a PMKSA for the PMKIDs, whether the AKM in the cached PMKSA
2. matches the AKM in the (Re)Association Request, and whether the PMK is still valid; and if so, it shall
3. assert possession of that PMK by beginning the 4-Way Handshake after association has completed. If the
4. Authenticator does not have a PMK for the PMKIDs in the (Re)Association Request, its behavior depends
5. on how the PMKSA was established. If SAE authentication was used to establish the PMKSA, then the AP

57

1. STA shall reject (re)association by sending a (Re)Association Response frame with status code
2. STATUS\_INVALID\_PMKID. Note that his allows the non-AP STA to fall back to full SAE authentication
3. to establish another PMKSA. If IEEE Std 802.1X authentication was used to establish the PMKSA, the AP
4. begins a full IEEE Std 802.1X authentication after association has completed.

62

63

1. Upon receipt of a FILS Authentication frame with one or more PMKIDs, an AP checks whether its
2. Authenticator has cached a PMKSA for the PMKIDs, whether the AKM in the cached PMKSA matches the
3. AKM in the FILS Authentication frame, and whether the PMK is still valid; and if so, it shall assert
4. possession of that PMK by indicating it in the FILS Authentication frame it responds with. If the
5. Authenticator does not have a PMK for the PMKIDs in the FILS Authentication frame, the AP may either

4

1. reply with EAP-Finish/Re-auth to continue FILS shared key authentication option if the non-AP STA
2. included sufficient information for that, or the AP rejects the authentication.

7

1. If both sides assert possession of a cached PMKSA, but the 4-Way Handshake or FILS authentication fails,
2. both sides may delete the cached PMKSA for the selected PMKID.

10

11

1. If a STA roams to an AP with which it is preauthenticating and the STA does not have a PMKSA for that
2. AP, the STA needs to initiate a full IEEE Std 802.1X EAP authentication.

14

15

16

17

18

19

20

21

22

23

24

## 25 11.5.14 RSNA key management in an ESS

26

### 27 *Change as follows:*

28

29

1. When the IEEE 802.1X authentication completes successfully, this standard assumes that the STA’s IEEE
2. 802.1X Supplicant and the IEEE 802.1X AS share a secret, called a PMK. In a non-FT environment, the AS
3. transfers the PMK, within the MSK, to the AP, using a technique that is outside the scope of this standard;
4. the derivation of the PMK from the MSK is EAP-method-specific. With the PMK in place, the AP initiates a

34

1. key confirmation handshake with the STA. The key confirmation handshake sets the IEEE 802.1X state
2. variable port Valid (as described in IEEE Std 802.1X-2010) to true.

37

1. When SAE authentication completes, both STAs share a PMK. With this PMK in place, the AP initiates the
2. key confirmation handshake with the STA.

40

41

1. Key confirmation is part of the FILS authentication exchange and no further handshakes are needed to sat-
2. isfy key management requirements.

44

45 When FILS authentication is not used, t~~T~~he key confirmation handshake is implemented by the 4-Way

46

47 Handshake. The purposes of the 4-Way Handshake are as follows:

48

49 a) Confirm the existence of the PMK at the peer.

50

1. b) Ensure that the security association keys are fresh.
2. c) Synchronize the installation of temporal keys into the MAC.

53

1. d) Transfer the GTK from the Authenticator to the Supplicant.
2. e) Confirm the selection of cipher suites.

56

57

1. NOTE —It is possible to forge message 1 of the 4-Way Handshake. However, the forgery attempt is
2. detected in the -failure of the 4-Way Handshake.

60

1. NOTE 2—Neither the AP nor the STA can use the PMK for any purpose but the one specified
2. herein without possibly compromising the key. If the AP uses it for another purpose, then the STA

63

1. can masquerade as the AP; similarly if the STA reuses the PMK in another context, then the AP can
2. masquerade as the STA.

1

2

## 3 11.5.21 RSNA rekeying

4

5

### 6 *Change as follows:*

7

1. When a PTKSA is deleted, a non-AP and non-PCP STA may reassociate with the same AP or PCP and/or
2. establish a new RSNA with the AP or PCP. If the non-AP and non-PCP STA has cached one or more

10

1. PMKSAs, it may skip the PMKSA establishment and proceed with the creation of a new PTKSA by using 4-
2. Way Handshake or FILS authentication.

13

14

15

16

17

18

19

# 20 11.6 Keys and key distribution

21

22

23

24

## 25 11.6.1.3 Pairwise key hierarchy

26

### 27 *Change as follows:*

28

29

1. Except when preauthentication or FILS authentication is used, the pairwise key hierarchy utilizes PRF-384,
2. PRF-512 or PRF-704to derive session-specific keys from a PMK, as depicted in Figure 11-28 (Pairwise key
3. hierarchy). When using AKM suite selector 00-0F-AC:12, the length of the PMK, PMK\_bits, shall be 384
4. bits. With all other AKM suite selectors, the length of the PMK, PMK\_bits, shall be 256 bits. The pairwise
5. key hierarchy takes a PMK and generates a PTK. The PTK is partitioned into KCK, KEK, and temporal

35

1. keys, which are used by the MAC to protect individually addressed communication between the
2. Authenticator’s and Supplicant’s respective STAs. PTKs are used between a single Supplicant and a single
3. Authenticator.

39

40

41

### 42 *Change as follows:*

43

1. NOTE 4—The Authenticator and Supplicant normally derive a PTK only once per association. A Supplicant or
2. an Authenticator use the 4-Way Handshake or FILS authentication to derive a new PTK. Both the
3. Authenticator and Supplicant create a new nonce value for each 4-Way Handshake or FILS authentication
4. instance.

48

49

50

## 51 11.6.1.7 FT key hierarchy

52

53

### 54 *Change as follows:*

55

## 56 11.6.1.7.1 Overview

57

58

1. This subclause describes the FT key hierarchy and its supporting architecture. The FT key hierarchy is
2. designed to allow a STA to make fast BSS transitions between APs without the need to perform an SAE or
3. IEEE Std 802.1X authentication at every AP within the mobility domain.

62

63

1. The FT key hierarchy can be used with SAE, IEEE Std 802.1X authentication, ~~or~~ PSK authentication, or
2. FILS authentication.
3. A three-level key hierarchy provides key separation between the key holders. The FT key hierarchy for the
4. Authenticator is shown in [Figure 11-30 (FT key hierarchy at an Authenticator](#_bookmark1)). An identical key hierarchy
5. exists for the Supplicant, and identical functions are performed by the corresponding S0KH and S1KH.

4

5

1. The FT key hierarchy shown in [Figure 11-30 (FT key hierarchy at an Authenticator)](#_bookmark1) consists of three levels
2. whose keys are derived using the key derivation function (KDF) described in [11.6.1.7.2 (Key derivation](#_bookmark2)
3. [function (KDF))](#_bookmark2) as follows:

9

1. a) PMK-R0 – the first-level key of the FT key hierarchy. This key is derived as a function of the master
2. session key (MSK) or PSK. It is stored by the PMK-R0 key holders, R0KH and S0KH.
3. b) PMK-R1 – the second-level key of the FT key hierarchy. This key is mutually derived by the S0KH

13

1. and R0KH.
2. c) PTK – the third-level key of the FT key hierarchy that defines the IEEE Std 802.11 and IEEE Std
3. 802.1X protection keys. The PTK is mutually derived by the PMK-R1 key holders, R1KH and
4. S1KH.

18

19

1. As shown in [Figure 11-30 (FT key hierarchy at an Authenticator),](#_bookmark1) the R0KH computes the PMK-R0 from
2. the key obtained from SAE authentication (for the purposes of FT this key is identified as the Master PMK,
3. or MPMK), from the PSK, ~~or~~ from the MSK resulting (per IETF RFC 3748-2004 [B38]) from a successful
4. IEEE Std 802.1X authentication between the AS and the Supplicant, or from the PMK (see [11.11.2.5.2](#_bookmark11)

24

1. [(PMKSA key derivation with FILS authentication](#_bookmark11))) resulting from a successful FILS authentication. Upon a
2. successful authentication, the R0KH shall delete any prior PMK-R0 security association for this mobility
3. domain pertaining to this S0KH. The R0KH shall also delete all PMK-R1 security associations derived from
4. that prior PMK-R0 security association. The PMK-R1s are generated by the R0KH and are assumed to be
5. delivered from the R0KH to the R1KHs within the same mobility domain. The PMK-R1s are used for PTK

30

1. generation. Upon receiving a new PMK-R1 for an S0KH, an R1KH deletes the prior PMK-R1 security
2. association and PTKSAs derived from the prior PMK-R1.

33

### 34 *Change figure as follows:*

35

36

37

38

39

40

41

42

43

44

45

46

47

48

49

50

51

52

53

54

55

56

57

58

59

60

61

62

63

64

65

1

2

3

|  |
| --- |
| Authentication Server (IEEE 802.1XAuthentication only) |
| MSK |  |

4

5

6

7

8

9

10

11

12

13

R0 Key Holder (R0KH) R0KH-ID

Derives PMK-R0 Derives PMK-R1s

PTK Key Holder BSSIDA

PTK Key Holder BSSIDB

R1 Key Holder (R1KH) R1KH-IDA

Derives PTKA

R1 Key Holder (R1KH) R1KH-IDB

Derives PTKB

PMK-R1A

PMK-R1B

14

15

16

17

18

19

20

21

22

23

24

25

27

28

29

30

31

32

33

34

35

36

37

38

39

40

41

42

43

44

45

46

47

48

49

50

51

52

53

54

### 55 *Change as follows:*

56

57

|  |
| --- |
| SAE Authentication |
| PSK | MPMK |

|  |
| --- |
| FILS Authentication |
|  | PMK |

## Figure 11-30—FT key hierarchy at an Authenticator

58 **11.6.1.7.2 Key derivation function (KDF)**

59

1. The KDF for the FT key hierarchy, and for AKMs 00-0F-AC:11, ~~and~~ 00-0F-AC:12, 00-0F-AC:14, 00-0F-
2. AC:15, 00-0F-AC:16, and 00-0F-AC:17 is a variant of the pseudo random function (PRF) defined in
3. 11.6.1.2 (PRF) and is defined as follows:

63

64

65

### 1 *Change as follows:*

2

## 3 11.6.1.7.3 PMK-R0

4

5

1. The first-level key in the FT key hierarchy, PMK-R0, is derived using the KDF defined in [11.6.1.7.2 (Key](#_bookmark2)
2. [derivation function (KDF)).](#_bookmark2) The PMK-R0 is the first level 256-bit keying material used to derive the next
3. level keys (PMK-R1s):

9

10

1. R0-Key-Data = KDF-Hash-Z(XXKey, "FT-R0", SSIDlength || SSID || MDID || R0KHlength ||
2. R0KH-ID || S0KH-ID)

13

14 PMK-R0 = L(R0-Key-Data, 0, L)

15

16

17 PMK-R0Name-Salt = L(R0-Key-Data, L, 128)

18

19 where

20

21

1. — KDF-Hash-Z is the KDF as defined in [11.6.1.7.2 (Key derivation function (KDF))](#_bookmark2) used to generate
2. a key of length 384 bits.
3. — If the AKM negotiated is 00-0F-AC:3, then Hash shall be SHA256, Z shall be 384, L shall be 256,
4. and XXKey shall be the second 256 bits of the MSK (which is derived from the IEEE Std 802.1X

26

1. authentication), i.e., XXKey = L(MSK, 256, 256). If the AKM negotiated is 00-0F-AC:4, then Hash
2. shall be SHA256, Z shall be 384, L shall be 256, and XXKey shall be the PSK. If the AKM
3. negotiated is 00-0F-AC:9, then Hash shall be SHA256, Z shall be 384, L shall be 256, and XXKey
4. shall be the MPMK generated as the result of SAE authentication. If the AKM negotiated is 00-0F-
5. AC:13, then Hash shall be SHA384, Z shall be 512, L shall be 384, and XXKey shall be the first 384

32

1. bits of the MSK (which is derived from the IEEE 802.1X authentication), i.e., XXKey = L(MSK, 0,
2. 384). ). If the AKM negotiated is 00-0F-AC:16, then Hash shall be SHA256, Z shall be 384, Q shall
3. be 256, and XXKey shall be the FILS-FT described in [11.11.2.5.3 (PTKSA key derivation with](#_bookmark12)
4. [FILS authentication)](#_bookmark12). If the AKM negotiated is 00-0F-AC:17, then Hash shall be SHA384, Z shall
5. be 512, Q shall be 384, and XXKey shall be the FILS-FT described in [11.11.2.4 (Key establishment](#_bookmark9)

38

39 [with FILS public key authentication)](#_bookmark9).

40

41

42

### 43 *Change as follows:*

44

45

## 46 11.6.1.7.4 PMK-R1

47

48 The second-level key in the FT key hierarchy, PMK-R1, is a 256-bit key used to derive the PTK. The PMK-

49

50 R1 is derived using the KDF defined in [11.6.1.7.2 (Key derivation function (KDF](#_bookmark2))):

51

52 PMK-R1 = KDF-Hash-Z(PMK-R0, "FT-R1", R1KH-ID || S1KH-ID)

53

54 where

55

1. — KDF-Hash-Z is the KDF as defined in [11.6.1.7.2 (Key derivation function (KDF)).](#_bookmark2)
2. — If the AKM negotiated is 00-0F-AC:3, 00-0F-AC:4, ~~or~~ 00-0F-AC:9, or 00-0F-AC:16, then Hash

58

1. shall be SHA256, and Z shall be 256. If the AKM negotiated is 00-0F-AC:13 or 00-0F-AC:17, then
2. Hash shall be SHA384, and Z shall be 384.

61

62

63

64

### 65 *Change as follows:*

1 **11.6.1.7.5 PTK**

2

3 The third-level key in the FT key hierarchy is the PTK. When FILS authentication is used to establish the FT

4

1. key hierarchy, PTK for the initial mobility domain association is derived as part of the FILS authentication
2. as defined in [11.11.2.5.3 (PTKSA key derivation with FILS authentication](#_bookmark12)). Otherwise, ~~T~~this key is
3. mutually derived by the S1KH and the R1KH used by the target AP, with the key length being a function of
4. the negotiated cipher suite as defined by Table 11-4 (Cipher suite key lengths) in 11.6.2 (EAPOL-Key
5. frames).

10

11

12

13

14

15

16

17

18

19

20

21

22

## 23 11.6.2 EAPOL-Key frames

24

### 25 *Change as follows:*

26

27 b) **Key Information.** This field is 2 octets and specifies characteristics of the key. See Figure 11-33

28

29 (Key Information bit layout).

30

31 6) Key MIC (bit 8). When AKM negotiated is not 00-0F-AC:14, 00-0F-AC:15, 00-0F-AC:16, or

32

1. 00-0F-AC:17, this bit is set to 1 if a MIC is in this EAPOL-Key frame and is set to 0 if this
2. message contains no MIC. When using an AEAD cipher this bit is set to 0.
3. f) **EAPOL-Key IV**. This field is 16 octets. It contains the IV used with the KEK. It shall contain 0
4. when an IV is not required. It should be initialized by taking the current value of the global key

37

1. counter (see 11.6.11 (RSNA Authenticator key management state machine)) and then incrementing
2. the counter. Note that only the lower 16 octets of the counter value are used.

40 h) **Key MIC.** When AKM negotiated is not 00-0F-AC:14, 00-0F-AC:15, 00-0F-AC:16, or 00-0F-

41

1. AC:17, ~~T~~the EAPOL Key MIC is a MIC of the EAPOL-Key frames, from and including the EAPOL
2. protocol version field to and including the Key Data field, calculated with the Key MIC field set to
3. 0. If the Encrypted Key Data subfield (of the Key Information field) is 1, the Key Data field is
4. encrypted prior to computing the MIC. When using an AEAD cipher, the EAPOL Key MIC is not
5. used. The length of this field depends on the negotiated AKM as defined in 11.6.3 (EAPOL-Key

47

48 frame construction and processing).

49

50 j) **Key Data**.

51

52

1. If the Encrypted Key Data subfield (of the Key Information field) is 1, the entire Key Data field shall
2. be encrypted. If the Key Data field uses the NIST AES key wrap, then the Key Data field shall be
3. padded before encrypting if the key data length is less than 16 octets or if it is not a multiple of 8.
4. The padding consists of appending a single octet 0xdd followed by zero or more 0x00 octets. When

57

1. processing a received EAPOL-Key frame, the receiver shall ignore this trailing padding. If the Key
2. Data field uses an AEAD cipher, then the Key Data field shall not be padded and the AAD for the
3. encipherment operation shall be the data of the EAPOL-Key frame from the EAPOL protocol ver-
4. sion field (inclusive) to the Key Data field (exclusive). Key Data fields that are encrypted, but do
5. not contain the GroupKey or SMK KDE, shall be accepted.

63

64

65

1

2

## 3 11.6.3 EAPOL-Key frame construction and processing

4

5

### 6 *Insert new rows (ignoring the header row) in the table as follows:*

7

8

## 9 Table 11-8— Integrity and Key Wrap Algorithms

10

11

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **AKM** | **Integrity Algorithm** | **KCK bits** | **Size of MIC** | **Key-wrap algorithm** | **KEK bits** |
| 00-0F-AC:14 | AES--SIV-256 | 256 | 0 | AES-SIV-256 | 256 |
| 00-0F-AC:15 | AES-SIV-512 | 384 | 0 | AES-SIV-512 | 512 |
| 00-0F-AC:16 | AES-SIV-256 | 256 | 0 | AES-SIV-256 | 256 |
| 00-0F-AC:17 | AES-SIV-512 | 384 | 0 | AES-SIV-512 | 512 |

12

13

14

15

16

17

18

19

20

21

22

23

24

25

26

27

28

29

## 30 11.6.11.3 Authenticator state machine variables

31

32

33 *Change as follows:*

34

1. — *MICVerified* – This variable is set to true if the MIC on the received EAPOL-Key frame is verified
2. and is correct or if AEAD cipher is used and AEAD decryption steps succeed. Any EAPOL-Key

37

38 frames with an invalid MIC are dropped and ignored.

39

40

41

42

43

44

### 45 *Insert new subclauses as follows:*

46

47

## 48 11.6.12 Authenticated Public Key Exchange

49

## 50 11.6.12.1 General

51

52

1. Some authentication protocols for 802.11, like FILS and 802.1X/EAP, use certified public keys to validate
2. signatures (created using the private analog to the certified public key) in order to authenticate the peer.
3. Obtaining a certified public key typically entails gaining trust in a Certification Authority (CA), known as a
4. trusted 3rd party, demonstrating an identity, providing a public key, and proving possession of the private

57

1. analog to a public key. Only then does the CA construct, sign, and return a certificate. Provided that two par-
2. ties go through the same steps to a common CA, or to CAs which trust a common hierarchical issuer, the two
3. parties can use their certified public keys to authenticate. In some cases, this step and the Public Key Infra-
4. structure (PKI) it implies, can be prohibitive.

62

63

1. NOTE—A party typically enrolls in a CA by constructing a PKCS#10 request and receiving, from the CA, a certificate
2. in the form of a PKCS#7 response.
3. An alternative is for two parties to exchange their uncertified public keys in a manner that proves possession
4. of the private analog to the public key as well as provides a level of authentication to the exchange so each
5. party has reason to trust the other party's key. Once trust is gained in the public key and the peer proves pos-

4

1. session of the private analog to that key, it can be used to verify signatures generated by the peer in service
2. of an authentication protocol.

7

8 The Public Key Exchange (PKEX) is a protocol to provide that service. It is a simple exchange consisting of

9

1. two request-response messages, four messages in total. PKEX uses a shared key/code/word/phrase and pub-
2. lic key cryptography in order to achieve the following goals:

12

13

1. — The protocol will result in the exchange of trusted public keys or it will fail;
2. — A passive adversary is unable to subvert the exchange, insert any different public keys, learn the pub-

16

1. lic keys, or learn the key/code/word/phrase shared by the two peers;
2. — An active adversary that does not know the shared key/code/word/phrase cannot successfully com-

19

1. plete the exchange; and,
2. — An attacker is not able to perform an off-line dictionary attack against PKEX in order to determine

22

23 either public key or to determine the shared key/code/word/phrase.

24

1. Due to the nature of the exchange, only public keys suitable for DSA (specified in FIPS 186-4) or ECDSA
2. (specified in ISO/IEC 14888-3) can be exchanged using PKEX, and a non-AP STA cannot engage in multi-

27

28 ple, simultaneous PEX exchanges with more than one peer.

29

## 30 11.6.12.2 PKEX overview

31

32

1. PKEX is a variant of the encrypted key exchange (EKE). For a public key, P, and an encrypted public key,
2. C, encryption with key k, Ek(), and decryption with key k, Dk(), are defined as:

35

36

37 C = Ek(P) = elem-op(P, f(k))

38

39

40 P = Dk(C) = elem-op(C, -f(k))

41

42 where elem-op() is defined in 11.3.4 (Finite cyclic groups) and f() is a function that creates a secret element

43

44 in the same group as P from a secret k.

45

46 The PKEX protocol uses a cryptographic hash function with the KDF from 11.6.1.7.2 (Key derivation func-

47

1. tion) as well as to distill entropy from the shared key/code/word/phrase. The particular hash function to use
2. depends on the size of the prime, p, that defines the finite field in which the STA's public key is defined.

50

51 SHA-256: len(p) 256

52

53

54 SHA-384: 256 < len(p) 384

55

56

57 SHA-512: 384 < len(p)

58

59

60

61

62 For purposes of extensibility, PKEX is described as a true peer-to-peer protocol. This allows it to be used

63

1. between a STA and AP in a role-based exchange as well as between two STAs directly without any sort of
2. client/server roles.

## 1 11.6.12.3 PKEX messages

2

3 PKEX messages are self-protected action frames sent from one STA to another. Their format is described in

4

1. 8.6.16.7 (Public Key Exchange Key Commit format) and 8.6.16.8 (Public Key Exchange Key Confirmation
2. format).

7

## 8 11.6.12.4 PKEX Protocol

9

10

## 11 11.6.12.4.1 Initial provisioning for PKEX

12

13 If a STA does not have a public key to exchange, it shall generate one in a chosen finite cyclic group from

14

1. the dot11RSNAConfigDLCGroup table. PKEX uses the same IANA registry to identify a group's domain
2. parameter set as SAE. For interoperability purposes, a conformant STA shall support group nineteen (19),
3. an ECC group defined over a 256-bit prime order field.

18

19

1. Prior to sending a PKEX message, both STAs shall be provisioned with a shared key/code/word/phrase,
2. hereinafter a credential. It shall be interpreted as a UTF-8 string with no NULL termination. The credential
3. shall be used to generate a password element, PWE per 11.3.4.2.2 (Generation of the password element with
4. ECC groups) (for ECC groups) or 11.3.4.3.2(Generation of the password element with FFC groups) (for
5. FFC groups), in the same group as the public key with the one minor change: the MAC addresses are

25

1. removed from the pwd-seed value calculation in 11.3.4.2.2 (Generation of the password element with ECC
2. groups) (for ECC groups) and 11.3.4.3.2 (Generation of the password element with FFC groups) (for FFC
3. groups) and the equation becomes:

29

30 pwd-seed = H(base || counter)

31

32

1. The password element, PWE, generated using the shared credential shall be used as function f() from
2. [11.6.12.2 (PKEX overview)](#_bookmark4) thus the encryption and decryption operations become:

35

36

37 Encrypt: C = elem-op(P, PWE)

38

39 Decrypt: P = elem-op(C, -PWE)

40

41

## 42 11.6.12.4.2 Exchange of PKEX Key Commit messages

43

1. A STA begins the PKEX protocol at anytime after generation of PWE. An AP STA shall not initiate PKEX
2. but shall wait until a non-AP STA has initiated to it.

46

47

1. To begin the PKEX protocol a STA shall first generate a random nonce whose length is equal to the size of
2. the digest of the hash algorithm used by PKEX, as defined in [11.6.12.3 (PKEX messages).](#_bookmark5) It shall then
3. encrypt its public key, P, using PWE in the technique defined in [11.6.12.1 (General)](#_bookmark3) to produce an encrypted

51

52 public key, C: C=elem-op(P, PWE).

53

1. Next, the PKEX Key Commit Message shall be generated in the format of Table 8-354a (PKEX Key Com-
2. mit frame Action field format) in 8.6.16.7.2 (Public Key Exchange Key Commit frame details):

56

57

1. 1) The STA's random nonce shall be copied into the Challenge Text field of the Challenge Text ele-
2. ment, with the length being filled in appropriately;

60

1. 2) The number from the IANA registry for the group in which the public key was created shall be cop-
2. ied into the Finite Cyclic Group field;

63

1. 3) The encrypted public key shall be converted into (an) octet string(s) according to 11.3.7.2.4 (Ele-
2. ment to octet string conversion).
	1. If the STA knows the MAC address of the peer the PKEX Key Commit message shall be sent to that desti-
	2. nation MAC address, otherwise it shall be sent to the group address. A STA that has initiated PEXK shall
	3. wait for receipt of a PKEX Key Commit from the peer. The STA may choose to retransmit the PKEX Key

4

1. Commit message after a suitable waiting period of its own choosing and may choose to retransmit a limited
2. number of times, of its own choosing, before abandoning PKEX. The waiting period and retransmit limit are
3. not defined here because they have no effect on interoperability.

8

9

1. A STA that receives a PKEX Key Commit message that has not been provisioned with a credential shall
2. silently drop the message.

12

13

1. Upon receipt of a PKEX Key Commit message the STA will check whether the finite cyclic group is accept-
2. able. If not, the STA shall silently discard the message. If the group is acceptable the STA checks whether it
3. has a public key in that group to exchange. If it does not, and does not wish to create such a public key it
4. shall silently discard the message.

18

19

20 Next, the STA determines whether it has sent a PKEX Key Commit message to the STA that transmitted the

21

1. received message (the peer STA) or to the group address. If not, for example if the recipient is an AP STA,
2. the STA shall generate PWE, if necessary, as defined above, generate a PKEX Key Commit message, and
3. transmit it to the peer STA. Otherwise, and in any case, the PKEX Key Commit message is processed:

25

26

1. 1) The peer's nonce is retrieved from the Challenge Text field of the Challenge Text element in the
2. received frame;

29

1. 2) The encrypted public key is obtained by converting the octet string(s) to an element according to
2. 11.3.7.2.5 (Octet string to element conversion). If conversion fails, the PKEX Commit message is
3. silently discarded;

33

34 3) The encrypted public key, C, is decrypted using PWE to produce the STA’s public key, P, according

35

36 to the decryption function definition in [11.6.12.1 (General):](#_bookmark3) P = elem-op(C, -PWE);

37

1. 4) A shared element, S, is generated using scalar-op() from 11.3.4 (Finite Cyclic groups) with the pri-
2. vate analog to the STAs public key, priv, and the peer STA's decrypted public key, P, and a secret
3. value, s, is derived from S using function F() from 11.3.4 (Finite cyclic groups):

41

42 S = scalar-op(priv, P)

43

44 s = F(S)

45

1. 5) A key confirmation key, k, whose length, i, is the length of the digest produced by the hash function,
2. is derived by first reducing the two nonces with the hash function used with PKEX and then using
3. the result as a key with the KDF from 11.6.1.7.2 (Key derivation function) with s and the label
4. “PKEX Key Confirmation” as data:

50

51

52 x = Hash(min(STA-nonce, peer-nonce) || max(STA-nonce, peer-nonce))

53

54

55 k = KDF-i(x, s || "PKEX Key Confirmation")

56

57

58 where min() and max() operations for nonces are encoded as specified in 8.2.2 (Conventions).

59

60

1. When processing of the PKEX Key Commit message finishes, a STA transitions into the Exchange of
2. PKEX Key Confirmation messages.

63

64

65

## 1 11.6.12.4.3 Exchange of PKEX Key Confirm messages

2

3 As soon as PKEY Key Commit message processing completes, a PKEX Key Confirm message is generated

4

1. in the format of table Table 8-354b (PKEX Key Confirmation frame Action field format) in 8.6.16.8.2 (Pub-
2. lic Key Exchange Key Confirmation details).

7

8 First, a key confirmation and integrity check is calculated by passing the key, k, and data consisting of a con-

9

1. catenation of the two unencrypted public keys and the STA's MAC address to the HMAC version of the
2. hash function used by PKEX:

12

13 check = HMAC-Hash(k, STA-pubkey || peer-pubkey || STA-MAC)

14

15

1. where the public keys are converted into an octet string per 11.3.7.2.4 (Element to octet string conversion)
2. prior to concatenation and passing to the HMAC. The value of check shall be copied into the MIC field of
3. the PKEX Key Confirm message and the message transmitted to the peer whose MAC address is the trans-
4. mitter of the received PKEY Key Commit message. The PKEX Key Confirm message shall not be a group

20

1. addressed frame. The STA may choose to retransmit the PKEX Key Confirm message after a suitable wait-
2. ing period of its own choosing and may choose to retransmit a limited number of times, of its own choosing,
3. before abandoning PKEX. The waiting period and retransmit limit are not defined here because they have no
4. effect on interoperability.

25

26

1. Upon receipt of a PKEX Key Confirm message from the peer, a verifier shall be generated based on the
2. expected value of the MIC field of the received PKEX Key Confirm message:

29

30 verifier = HMAC-Hash(k, peer-pubkey || STA-pubkey || peer-MAC)

31

32

1. The verifier shall then be compared to the value in the MIC field of the received PKEX Key Confirm mes-
2. sage. If they differ, the PKEX shall be silently aborted and all state associated with this exchange shall be
3. irretrievably deleted. Otherwise, PKEX shall be deemed to have completed successfully and the peer's pub-

36

1. lic key can be trusted to be used in a subsequent authentication protocol. All states other than the peer's
2. MAC address and now-trusted public key shall be irretrievably deleted.

39

40

41

42

43

44

45

46

47

### 48 *Insert new subclauses as follows:*

49

50

# 51 11.11 Authentication for FILS

52

53

## 54 11.11.1 General

55

1. FILS authentication is an RSNA authentication protocol. The FILS authentication protocol authenticates
2. STAs to each other, using either a shared key or a public key. When FILS shared key authentication is used,

58

1. the authentication exchange can optionally be performed with PFS. When FILS public key authentication is
2. used, PFS is always used. When the FILS authentication protocol is performed with PFS, the STA and AP
3. derive ephemeral public and private keys with respect to a particular set of domain parameters that define a
4. finite cyclic group and then exchange public keys. The result of the FILS authentication includes a PTKSA.

63

64

65

1 The security of FILS authentication depends on the following assumptions:

2

3 — When FILS shared key authentication is used, each STA shares either a valid rRK as defined in

4

1. IETF RFC 6696 with a TTP that is capable of being used with EAP-RP, or a PMK cached from a
2. previous authenticated connection.
3. — When FILS public key authentication is used, each STA has a means to trust the public key of the

8

9 other STA.

1. — When PFS is used, a finite cyclic group is negotiated where solving the discrete logarithm problem is
2. computationally infeasible.

12

1. — When PFS is used, both the STA and AP have in common at least one finite cyclic group from the
2. dot11RSNAConfigDLCGroupTable.

15

16

17

18

## 19 11.11.2 FILS authentication protocol

20

21

## 22 11.11.2.1 General

23

1. The STA and AP perform key establishment using Authentication frames and perform key confirmation
2. using (Re)Association Request and (Re)Association Response frames.

26

27

1. After exchanging Authentication frames, the STA and AP derive a shared and secret key which will be used
2. to derive a set of secret keys (as defined in [11.11.2.5.2 (PMKSA key derivation with FILS authentication))](#_bookmark11)
3. that are authenticated after exchanging (Re)Association Request and (Re)Association Response frames.

31

32

1. When a shared key is used for FILS authentication, and if the STA shares a valid rRK with the TTP, then
2. EAP-RP as defined in IETF RFC 5295 and IETF RFC 6696 shall be used.

35

36

37

38

## 39 11.11.2.2 Discovery of a FILS capable AP

40

41

1. An AP indicates that it is capable of performing FILS authentication by indicating support for a FILS AKM
2. in an RSN element and including it, and the FILS Indication element, in Beacon and Probe Response
3. frames.

45

46

1. An AP may indicate that it is capable of performing FILS authentication by indicating support for a FILS
2. AKM in the FD RSN subfield in a FILS Discovery frame.

49

1. An AP indicates support for FILS shared key authentication by advertising between zero and seven realms
2. using a Domain Information subfield of the FILS Indication element that is part of Beacon, Probe Response,

52

1. and FILS Discovery frames. If the STA discovers a FILS-capable AP that advertises a hashed domain name
2. that matches the hashed value of the realm of the third party Authentication Server, with which the STA
3. shares a valid rRK as defined in IETF RFC 6696, the STA may begin the FILS authentication protocol with
4. the AP using EAP-RP. Domain name hashing is specified in 10.47.4 (FILS authentication and higher layer
5. setup capability indications). If a STA discovers a FILS-capable AP that does not advertise any realms, or

58

1. that advertises realms unknown to the STA, and the STA believes it shares a PMKSA with the AP, it may
2. begin the FILS authentication protocol with the AP using PMKSA caching.

61

62 An AP indicates support for FILS public key authentication by advertising up to seven public key indicators

63

1. in the FILS Indication element that is part of Beacon, Probe Response, and FILS Discovery frames. If the
2. STA discovers that it trusts the issuer of an AP’s X.509v3 certificate, or that it trusts its uncertified public
3. key identified by matching its hash, the STA may begin the FILS authentication protocol to the AP and per-
4. form mutual authentication using trusted public keys.

3

4

5

6

## 7 11.11.2.3 Key establishment with FILS shared key authentication

8

9

10

11

## 12 11.11.2.3.1 Overview

13

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

15

1. using FILS shared key authentication that uses shared symmetric keys between the STA and the authentica-
2. tion server.

18

1. A STA may initiate FILS shared key authentication either with a FILS capable AP that is connected to a
2. TTP Authentication Server that shares a valid key, called an rRK, as defined in IETF RFC 6696 with the

21

1. STA, or with a FILS capable AP with whom it shares a cached PMKSA. If neither of these cases applies, a
2. full EAP exchange may be performed via IEEE Std 802.1X authentication to establish rRK as defined in
3. IETF RFC 6696 or another form of FILS authentication may be used to establish a shared PMKSA.

25

26 EAP-RP signaling as defined in IETF RFC 5295 and IETF RFC 6696 is used to validate the mutual posses-

27

1. sion of rRK between the STA and the Authentication Server. EAP-RP signaling is encapsulated using a
2. FILS Wrapped Data element in the Authentication frame. The AP unwraps the encapsulated EAP-RP packet
3. received from the STA in the FILS Wrapped Data element and forwards the EAP-RP packet to the Authen-
4. tication Server using a transport that is out of scope of this specification. When the AP receives an EAP-RP
5. packet from the Authentication Server, the AP forwards the packet to the STA by encapsulating the EAP-RP

33

34 packet in the FILS Wrapped Data element of the Authentication frame.

35

36 The message sequence is depicted in [Figure 11.54 (FILS shared key authentication).](#_bookmark8)

37

38

39

40

41

42

43

44

45

46

47

48

49

50

51

52

53

54

55

56

57

58

59

60

61

## 62 Figure 11.54—FILS shared key authentication

63

64

65 The following subclauses are organized per each step as shown in [11.54 (FILS shared key authentication)](#_bookmark8).

1

2

## 3 11.11.2.3.2 Non-AP STA construction of Authentication frame

4

5

1. If the STA chooses to initiate FILS shared key authentication, it shall first choose a random 16-octet nonce
2. and then determine whether to attempt PMKSA caching. If PMKSA caching is attempted, it shall generate a
3. list of PMKSA identifiers. Otherwise, it shall construct an EAP-Initiate/Re-auth packet per IETF RFC 6696,

9

10 with the following clarifications:

11

12 — Regarding EAP-RP Flags:

13

1. —The B flag shall be set to 0, indicating that this is not an EAP-RP bootstrap message.
2. —The L flag shall be set to 1, indicating that the TTP with whom the STA shares the rRK is to pro-

16

1. vide the lifetimes of rRK and rMSK in the EAP-Finish/Re-auth Packet.
2. — The Cryptosuite field shall not be set to 1.

19

20

1. If PFS is desired, the STA selects a finite cyclic group from the dot11RSNAConfigDLGGroupTable, gener-
2. ates an ephemeral private key, and performs the group’s scalar-op (see 11.3.4.1 (General)) with its random
3. ephemeral private key and the generator from the selected finite cyclic group to compute an ephemeral pub-
4. lic key.

25

26

1. The STA then constructs an Authentication frame with the Authentication algorithm number set to 4 (FILS
2. authentication) (see 8.4.1.1 (Authentication Algorithm Number field)) and the Authentication transaction
3. sequence number set to 1. The random nonce shall be encoded in the FILS Nonce field (see 8.4.1.58 (FILS
4. Nonce field)), and the FILS Authentication Type field shall be set to one of the FILS shared key authentica-

31

1. tion as defined in Table 8-73a (Values of FILS Authentication Type field) depending on whether PFS is
2. used. If a list of PMKSA identifiers was generated, it shall be used to construct the PMKID List elements.
3. The EAP-Initiate/Re-auth packet, if generated, shall be copied into the FILS Wrapped Data field (see
4. 8.4.2.183 (FILS Wrapped Data element)). If PFS is desired, the chosen finite cyclic group shall be encoded
5. in the Finite Cyclic Group field (see 8.4.1.42 (Finite Cyclic Group field)) and the ephemeral public key shall

37

1. be encoded in the Element field (see 8.4.1.40 (Element field)) according to the element to octet-string con-
2. version in 11.3.7.2.4 (Element to octet string conversion).

40

41 The STA transmits the Authentication frame to the AP.

42

43

44

45

## 46 11.11.2.3.3 AP processing of Authentication frame

47

48

49 Upon reception of the Authentication frame, the AP shall do the following procedure:

50

51 a) If Authentication frame includes a Finite Cyclic Group field, then the AP shall first determine

52

1. whether the indicated finite cyclic group in the received FILS Authentication frame is supported.
2. b) If the indicated finite cyclic group in the received FILS Authentication frame is not supported, the
3. AP shall respond with an Authentication frame with the Authentication algorithm number set to 4

56

1. (FILS authentication) (see 8.4.1.1 (Authentication Algorithm Number field)) and the Status Code
2. field set to 77 (Authentication is rejected because the offered finite cyclic group is not supported)
3. and shall terminate the exchange.

60

1. c) The AP shall check whether PMKSA caching is being attempted by the presence of the PMKID List
2. element.

63

1. 1) If the PMKID List element is present, the AP checks whether any PMKSA identifier offered in
2. the PMKID List matches an identifier for a cached PMKSA. If so, the AP selects a PMKID that
3. matches and continues the FILS shared key authentication protocol using the PMK from the
4. identified PMKSA.

3

1. 2) If a PMKID List element is not present or if no PMKSA identifier offered in the PMKID list
2. matches any identifier for a cached PMKSA, the AP checks whether an EAP-Initiate/Re-auth
3. packet was included. If not, the AP shall respond with an Authentication frame with the

7

1. Authentication algorithm number set to 14 and the Status Code field set to 53 (invalid PMKID)
2. and shall terminate the exchange.

10

1. 3) If an EAP-Initiate/Re-auth packet is included, the AP shall extract the EAP-Initiate/Re-auth
2. data from the FILS Wrapped Data field (see 8.4.2.183 (FILS Wrapped Data element)) and shall
3. forward it to the Authentication Server. When applicable, the AP communicates with the
4. Authentication Server using the same protocols it uses when authenticating with EAP. Suitable

15

1. protocols include, but are not limited to, remote authentication dial-in user service RADIUS (as
2. specified in IETF RFC 2865) and Diameter (as specified in IETF RFC 6942).

18

19

1. If PFS is being used, the AP shall also generate an ephemeral private key and perform the group’s scalar-op
2. (see 11.3.4.1 (General)) to produce its own ephemeral public key. The AP may delay the generation of its
3. ephemeral public/private key pair until after receiving a response from the Authentication Server, if applica-
4. ble. The Authentication Server processes the EAP-Initiate/Re-auth packet as specified in IETF RFC 6696
5. and returns an EAP-Finish/Re-auth packet to the AP. In the case of successful authentication by the Authen-

25

1. tication Server, the Authentication Server returns the associated EAP-RP rMSK with the EAP-Finish/Re-
2. auth packet. If the Authentication Server responds with a failure indication, then the AP shall produce an
3. Authentication frame with the Authentication Algorithm Number field set to “Fast Initial Link Setup authen-
4. tication” 14 (see 8.4.1.1 (Authentication Algorithm Number field)), and the Status Code field set to 15
5. (Authentication rejected because of challenge failure). In the case of successful authentication by the

31

1. Authentication Server, the Authentication Server returns the associated EAP-RP rMSK with the EAP-Fin-
2. ish/Re-auth packet and processing terminates.

34

35

36 The AP proceeds by constructing an Authentication frame.

37

38

39

40

41

42 **11.11.2.3.4 AP construction of Authentication frame**

43

44

1. If the AP is not connected to, or does not recognize the Authentication Server identified by the STA using
2. the realm in the key Name-NAI field of the EAP-Initiate/Re-auth packet, then the AP shall send Authentica-
3. tion frame with Status Code field set to 113, “Authentication rejected due to unknown Authentication
4. Server” to the non-AP STA.

49

50

1. Otherwise, the AP shall generate its own nonce and construct an Authentication frame for the STA. This
2. frame shall contain the FILS wrapped data that encapsulates EAP-Finish/Re-auth packet received from the
3. Authentication Server. In addition, if PFS is used, the Element field of the Authentication frame sent by the

54

1. AP contains the AP’s ephemeral public key. In this frame, the AP shall set the Authentication sequence
2. number to 2.

57

58

1. If PFS is being used for the exchange, STA’s public key shall be converted from an octet string to an ele-
2. ment according to the conversion in 11.3.7.2.5 (Octet string to element conversion). Then the AP shall ver-
3. ify the STA's public key in a group-specific fashion as described in 5.6.2.3 of NIST SP 800-56A R2. If
4. verification fails, the AP shall terminate the FILS authentication protocol. Otherwise, the AP shall perform

63

1. the group's scalar-op (see 11.3.4.1 (General)) with the STA's ephemeral public key and its own ephemeral
2. private key to produce an ephemeral Diffie-Hellman shared secret, DHss.
3. The AP transmits the Authentication frame to the STA. Upon transmission of the FILS Authentication
4. frame, the AP shall proceed to key establishment per [11.11.2.5 (Key establishment with FILS authentica-](#_bookmark10)
5. [tion).](#_bookmark10)

4

5

## 6 11.11.2.3.5 Non-AP STA processing of Authentication frame

7

8 The STA processes the received Authentication frame as follows:

9

10

1. a) If the received Authentication frame does not include the Authentication Algorithm Number
2. equal to 4 (FILS authentication) (see 8.4.1.1 (Authentication Algorithm Number field)), or if
3. PMKSA caching was attempted and the received Authentication frame includes a PMKID that

14

1. does not match a PMKID in the Authentication frame sent by the STA; or if the received
2. Authentication frame doesn’t include either a PMKID or an EAP-Finish/Re-auth packet, the
3. STA shall abandon FILS authentication.

18

1. b) If the received Authentication frame includes the Status Code field equal to 15 (Authentication
2. rejected because of challenge failure) or 53 (invalid PMKID), then the STA shall abandon the
3. FILS authentication.

22

1. c) The STA verifies that the AP transmitted PFS parameters are consistent with the desire of the
2. STA (indicated by whether or not the STA transmitted an ephemeral public key):
3. 1 If the STA transmitted an ephemeral public key, and the received Authentication frame

26

1. does not include a well-encoded ephemeral public key, then the STA shall abandon the
2. FILS authentication.

29

1. 2 If the STA did not transmit an ephemeral public key desired PFS, and the received
2. Authentication frame includes an ephemeral public key, then the STA shall abandon the
3. FILS authentication.

33

1. d) If applicable, the STA processes the EAP-Finish/Re-auth packet as per IETF RFC 6696:
2. 1 If the ‘R’ flag = 0, indicating success, then the STA shall derive rMSK.

36

37 2 If the ‘R’ flag = 1, indicating failure, then the STA shall abandon the FILS authentication.

38

1. e) If PFS is being used for the exchange, the AP’s public key shall be converted from an octet
2. string to an element according to the conversion in 11.3.7.2.5 (Octet string to element conver-
3. sion). Then the STA shall verify the AP’s public key in a group-specific fashion as described in
4. 5.6.2.3 of NIST SP 800-56A R2. If verification fails, the STA shall terminate the FILS authen-
5. tication protocol. Otherwise, the STA shall perform the group's scalar-op (see 11.3.4.1 (Gen-

44

1. eral)) with the AP’s ephemeral public key and its own ephemeral private key to produce an
2. ephemeral Diffie-Hellman shared secret, DHss.
3. f) The STA shall perform key derivation per [11.11.2.5 (Key establishment with FILS authentica-](#_bookmark10)

48

49 [tion)](#_bookmark10) and key confirmation per [11.11.2.6 (Key confirmation with FILS authentication)](#_bookmark13).

50

1. If the STA was attempting EAP-RP Authentication and didn’t successfully receive an Authentication frame
2. within the time of dot11AuthenticationResponseTimeout, then the STA should perform retransmission pro-
3. cedure as defined in IETF RFC 6696. If the retransmission procedure fails, then the STA shall abandon the

54

55 FILS authentication and should perform full EAP authentication via IEEE 802.1X authentication.

56

1. If the STA was attempting PMKSA caching and did not receive an Authentication frame from the AP, the
2. STA shall attempt to use an alternate authentication method.

59

60

1. Upon successful processing of the Authentication frame, the STA shall proceed with key establishment per
2. [11.11.2.4 (Key establishment with FILS public key authentication).](#_bookmark9)

63

64

65

## 1 11.11.2.4 Key establishment with FILS public key authentication

2

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

4

5 using FILS public key authentication.

6

## 7 11.11.2.4.1 Prior to exchange [CID 10049]

8

9

1. FILS public key authentication performs key establishment with a Diffie-Hellman exchange. Prior to begin-
2. ning the exchange, the non-AP STA;

12

1. 1) Selects a finite cyclic group from the dot11RSNConfigDLCGroup table to perform the Diffie-Hell-
2. man exchange.

15

1. 2) Generates a random nonce, generates an ephemeral private key, and uses the selected group’s scalar-
2. op (see 11.3.4.1 (General)) with its private key to generate its ephemeral public key.

18

1. 3) Constructs an Authentication frame (see 8.3.3.11 (Authentication frame format)) as follows:
2. a) The Authentication algorithm number is set to 4 and the Authentication transaction sequence
3. number is set to 1.

22

23 b) The random nonce is encoded in the FILS Nonce field (see 8.4.1.58 (FILS Nonce field)).

24

1. c) FILS Authentication Type field indicates FILS public key authentication (2).
2. d) The chosen finite cyclic group is encoded in the Finite Cyclic Group field (see 8.4.1.42 (Finite
3. Cyclic Group field)).

28

1. e) The STA’s public key is encoded into the Element field (see 8.4.1.40 (Element field)) accord-
2. ing to the element to octet-string conversion in 11.3.7.2.4 (Element to octet string conversion).

31

32

33 The STA then transmits the Authentication frame to the AP.

34

## 35 11.11.2.4.2 Processing after receipt [CID 10049]

36

37 Upon receipt, the AP processes the STA’s Authentication frame as follows:

38

39

1. 1) If the finite cyclic group indicated by the Finite Cyclic Group field is not acceptable, the AP shall
2. respond with an Authentication frame with the status code of 77 (“Authentication is rejected
3. because the offered finite cyclic group is not supported”) and terminate the FILS authentication pro-

43

1. tocol.
2. 2) If the finite cyclic group is acceptable, the AP verifies the validity of the STA’s public key:

46

1. a) The public key is converted from an octet string to an element according to the conversion in
2. 11.3.7.2.5 (Octet string to element conversion).
3. b) The public key, as a group element, is verified in a group-specific fashion as described in

50

1. 5.6.2.3 of NIST SP 800-56A R23. If verification fails, the AP shall terminate the FILS authen-
2. tication protocol.
3. 3) The STA’s nonce and validated public key are extracted from the Authentication frame.

54

55

## 56 11.11.2.4.3 Post processing [CID 10049]

57

58 Next, the AP shall:

59

60

1. 1) Generates a random nonce, generates a random ephemeral private key, and then uses the agreed-
2. upon group’s scalar-op (see 11.3.4.1 (General)) with its private key to generate its ephemeral public
3. key.

64

65 2) Constructs an Authentication frame (see 8.3.3.11 (Authentication frame format)) as follows:

1. a) The Authentication algorithm number is set to 4, and the Authentication transaction sequence
2. number is set to 2.

3

4 b) The FILS Authentication Type field is set to 2, indicating FILS public key authentication.

5

1. c) The random nonce is encoded in the FILS Nonce field (see 8.4.1.58 (FILS Nonce field)).
2. d) The finite cyclic group is encoded in the Finite Cyclic Group field (see 8.4.1.42 (Finite Cyclic
3. Group field)).

9

1. e) The AP’s public key is encoded in the Element field (see 8.4.1.40 (Element field)) according to
2. the element to octet-string conversion in 11.3.7.2.4 (Element to octet string conversion).

12

13 3) Transmits the Authentication frame to the STA.

14

1. 4) Computes the Diffie-Hellman shared secret, DHss, based on the STA’s ephemeral public key and its
2. own private key with the chosen group’s scalar-op.

17

18 5) Perform key derivation (see [11.11.2.5 (Key establishment with FILS authentication)).](#_bookmark10)

19

## 20 11.11.2.4.4 Upon receipt [CID 10049]

21

22

23 Upon receipt, the STA:

24

25

1. 1) Verifies that the finite cyclic group in the AP’s response is equal to the group selected by the STA. If
2. these differ, the STA shall terminate the authentication exchange.
3. 2) Verifies the validity of the AP’s public key:

29

1. a) The public key is converted from an octet string to an element according to the conversion in
2. 11.3.7.2.5 (Octet string to element conversion).

32

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

36

37 3) Extracts the AP’s nonce and verified public key from the Authentication frame.

38

1. 4) Compute the Diffie-Hellman shared secret, DHss, based on the AP’s ephemeral public key and its
2. own private key with the chosen group’s scalar-op to derive DHss.

41

1. 5) Performs key derivation (see [11.11.2.5 (Key establishment with FILS authentication))](#_bookmark10) and begins
2. key confirmation (see [11.11.2.6 (Key confirmation with FILS authenticat](#_bookmark13)ion)).

44

45

46

47

## 48 11.11.2.5 Key establishment with FILS authentication

49

50

## 51 11.11.2.5.1 General

52

53 When not using PMKSA caching, a PMK is created using the Extract function of IETF RFC 5869. When

54

1. using PMKSA caching, a new PMKSA is not created. Instead, the PMKSA used for PMKSA caching
2. remains and continues to be identified by the appropriate PMKID. Regardless of whether PMKSA caching
3. is used or not, a PTKSA shall be generated with each FILS authentication exchange.

58

59

1. PTKSA creation uses the KDF from [11.6.1.7.2 (Key derivation function (KDF))](#_bookmark2) to derive the following
2. keys from the PMK: a key confirmation key (KCK); a key encryption key (KEK); and a temporal key (TK).

62

63

1. PTKSA key establishment shall immediately be followed by key confirmation per 11.11.2.5 (Key confirma-
2. tion with FILS authentication).

## 1 11.11.2.5.2 PMKSA key derivation with FILS authentication

2

3 The Extract function used to derive the PMK takes the two nonces as salt and the secret(s) from FILS Key

4

1. establishment as input keying material. A PMKID used to identify the PMKSA is generated using the hash
2. algorithm from the negotiated AKM on input data specific to the FILS Key Establishment step. The length
3. of the PMK shall be either 256 bits or 384 bits depending on the negotiated AKM, and the length of the
4. PMKID shall be 128 bits. If FILS shared key authentication was used to generate input keying material, the
5. PMK and PMKID are derived as:

10

11

12 PMK = HMAC-Hash(SNonce || ANonce, rMSK [ || DHss ])

13

14

15 PMKID = Truncate-128(Hash(EAP-Initiate/Reauth))

16

17

18

19

1. When FILS public key authentication is used to generate input keying material, the PMK and PMKID are
2. derived as:

22

23

24 PMK = HMAC-Hash(SNonce || ANonce, DHss)

25

26 PMKID = Truncate-128(Hash(gSTA || gAP))

27

28

29 where:

30

31 — SNonce is the STA nonce and ANonce is the AP nonce.

32

33 — rMSK is the shared secret from the EAP-RP exchange.

34

1. — DHss is the shared secret derived from the Diffie-Hellman exchange, when performed.
2. — The brackets indicate the inclusion of the shared secret when doing a Diffie-Hellman exchange; there
3. is no shared secret to include otherwise.

38

1. — EAP-Initiate/Reauth is the EAP-RP packet sent by the STA during key establishment with FILS
2. shared key authentication.

41

42 — gSTA is the STA's Diffie-Hellman value and gAP is the AP's Diffie-Hellman value.

43

44 — Hash is the AKM-specific hash function.

45

1. Upon completion of PMK and PMKID generation the shared secret, DHss, and rMSK, if applicable, shall be
2. irretrievably deleted.

48

49

50

51

52

## 53 11.11.2.5.3 PTKSA key derivation with FILS authentication

54

1. For PTKSA key generation, the inputs to the KDF are the PMK of the PMKSA, a constant label, and a con-
2. catenation of the STA’s MAC address, the AP’s BSSID, the STA’s nonce, and the AP’s nonce. When the

57

1. AKM negotiated is 00-0F-AC:14 or 00-0F-AC:16, the length of KEK shall be 256 bits, and the length of the
2. KCK 256 bits. When the AKM negotiated is 00-0F-AC:15 or 00-0F-AC:17, the length of the KEK shall be
3. 512 bits, and the length of KCK shall be 384 bits. When the AKM negotiated is 00-0F-AC:16, FILS-FT is
4. 256 bits; when AKM negotiated if 00-0F-AC:17, FILS-FT is 384 bits; otherwise, FILS-FT is not derived.
5. The total amount of bits extracted from the KDF shall therefore be 512+TK bits, 896+TK bits, or 1280+TK

63

1. bits depending on the AKM negotiated, where TK\_bits are determined from Table 11-4 (Cipher suite key
2. lengths):

1

2

3

4

5 where:

6

KCK || KEK || TK [ || FILS-FT ] = KDF-X(PMK, “FILS PTK Derivation”, SPA || AA || SNonce || ANonce)

1. — X is 512+TK\_bits, 768+TK bits, 896+TK bits, or 1280+TK bits from Table 11-4 (Cipher suite key
2. lengths) depending on the AKM negotiated.

9

1. — PMK is the PMK from the PMKSA, either created from an initial FILS connection or from a cached
2. PMKSA, when PMKSA caching is used.

12

1. — SPA is the STA’s MAC address and the AA is the AP’s BSSID.
2. — SNonce is the STA’s nonce and ANonce is the AP’s nonce.

15

1. — The brackets indicate the generation of FILS-FT when doing FT initial mobility domain association
2. using FILS authentication; FILS-FT is not generated otherwise.

18

19

20

21

## 22 11.11.2.6 Key confirmation with FILS authentication

23

## 24 11.11.2.6.1 General

25

26

1. Key confirmation for FILS authentication is a (Re)Association Request frame followed by a (Re)Associa-
2. tion Response frame. Components of the (Re)Association Request and (Re)Association Response frames
3. shall be protected using KEK.

30

31

32

33

## 34 11.11.2.6.2 (Re)Association Request for FILS key confirmation

35

36

1. The STA constructs a (Re)Association Request frame for FILS authentication per 8.3.3.5 (Association
2. Request frame format) and 8.3.3.7 (Reassociation Request frame format). Hash functions are used to gener-
3. ate the FILS Key Confirmation element and the specific hash function depends on the AKM negotiated

40 (8.4.2.24.3 (AKM suites)).

41

42

1. For FILS shared key authentication, the KeyAuth field of the FILS Key Confirmation element is constructed
2. by using the HMAC mode of the negotiated hash function with a key of KCK on a concatenation of the
3. STA’s nonce, the AP’s nonce, the STA’s MAC address, the AP’s BSSID, and conditionally the STA’s pub-
4. lic Diffie-Hellman value and the AP’s public Diffie-Hellman value, in that order:

47

48

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

50

51 where:

52

53

1. — Hash is the hash function specific to the negotiated AKM.
2. — SNonce is the STA's nonce, ANonce is the AP’s nonce.

56

1. — STA-MAC is the MAC address of the STA and AP-BSSID is the BSSID of the AP.
2. — gSTA is the STA’s Diffie-Hellman public value and gAP is the AP’s Diffie-Hellman public value.

59

1. — The brackets indicate the inclusion of the Diffie-Hellman public values when doing PFS with FILS
2. shared key authentication; there are no Diffie-Hellman public values to include otherwise.

62

63

1. For FILS public key authentication, the KeyAuth field of the FILS Key Confirmation element is a digital
2. signature using the STA’s private key, of the negotiated hash function on a concatenation of the STA’s pub-
3. lic Diffie-Hellman value, the AP’s public Diffie-Hellman value, the STA’s nonce, the AP’s nonce, the
4. STA’s MAC address, and the AP’s BSSID, in that order:

3

4

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

6

1. where Sig-STA( ) indicates a digital signature using the STA’s private key, analog to the STA’s trusted pub-
2. lic key. The form of signature depends on the type of public key used by the STA (IETF RFC 3447 for RSA;
3. FIPS 184-4 for DSA; and ISO/IEC 14888-3 for ECDSA). The data to be signed is first hashed and the hash

10

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

12

1. The (Re)Association Request frame shall be encrypted using the AEAD algorithm as defined in [11.11.2.7](#_bookmark14)
2. [(AEAD cipher mode for FILS)](#_bookmark14) with the KEK as the key. The AAD used with the AEAD algorithm for the

15

16 Association Request frame is constructed by concatenating the following data together in order:

17

18 — The STA’s MAC address,

19

1. — The AP’s BSSID,
2. — The STA’s nonce,

22

1. — The AP’s nonce,
2. — The contents of the (Re)Association Request frame from the Capability Information field (inclusive)
3. to the FILS Session element (inclusive).

26

27

1. The plaintext passed to the AEAD algorithm is the data that would follow the FILS Session element in an
2. unencrypted frame. The output of the AEAD algorithm becomes the data that follows the FILS Session ele-
3. ment in the encrypted and authenticated (Re)Association Request frame. The output of the algorithm is as

31

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

33

1. The AP decrypts and verifies the received (Re)Association Request frame with the AEAD algorithm as
2. defined in [11.11.2.7 (AEAD cipher mode for FILS)](#_bookmark14) with the KEK as the key. The AAD is reconstructed as
3. defined above and is passed, along with the ciphertext of the received frame, to the AEAD decryption oper-

37

38 ation.

39

1. If the output from the AEAD decryption operation returns a failure, the authentication exchange shall be
2. deemed a failure. If the output does not return failure, the output plaintext replaces the ciphertext as portion
3. of the frame that follows the FILS Session element and processing of the received frame continues by check-

43

44 ing the value of the FILS Key Confirmation element.

45

1. The AP verifies that the RSNE received in the (Re)Association Request frame has identical AKM suite and
2. cipher suites and RSN capabilities as were included in the RSNE in the Authentication frame from the STA.

48

49 If these fields differ, authentication shall be deemed a failure.

50

1. For FILS shared key authentication, the AP constructs a verifier, Key-Auth', in an identical manner as the
2. STA constructed its Key-Auth above.

53

54

1. The AP compares Key-Auth' with the KeyAuth field in the FILS Key Confirmation element of the received
2. frame. If they differ, authentication shall be deemed a failure.

57

1. For FILS public key authentication, the AP uses the STA’s (certified) public key from the FILS Public Key
2. element to verify that the signature contained in the KeyAuth field corresponds to the purported signature by

60

61 the STA over the concatenation of the following:

62

63 — the STA’s public Diffie-Hellman value gSTA,

64

65 — the AP’s public Diffie-Hellman value gAP,

1 — the STA’s nonce SNonce, the AP’s nonce ANonce,

2

1. — the STA’s MAC address STA-MAC,
2. — the AP’s BSSID AP-BSSID.

5

6

1. in that order, according to the signature scheme used. Furthermore, the AP checks all certificates in the cer-
2. tificate chain, both cryptographically and from a security policy perspective, according to the procedures for
3. checking certificates and certificate chains in IETF RFC 5280. If any of these verifications fail, authentica-
4. tion shall be deemed a failure.

11

12

1. If authentication is deemed a failure, KCK, KEK, TK and the PTKSA shall be irretrievably deleted and the
2. AP shall return an Authentication frame with a status code set to 112 (Authentication rejected due to FILS
3. authentication failure). If PMKSA caching was not being employed for this failed authentication attempt,
4. the PMKSA shall also be deleted. If PMKSA caching was being used, the cached PMKSA may not be

17

18 deleted.

19

20

21

22

## 23 11.11.2.6.3 (Re)Association Response for FILS key confirmation

24

1. The AP constructs a (Re)Association Response frame for FILS authentication per 8.3.3.6 (Association
2. Response frame format) and 8.3.3.8 (Reassociation Response frame format). As with the (Re)Association

27

1. Request frame, hash functions are used to generate the FILS Key Confirmation element and the specific
2. hash function depends on the AKM negotiated (see 8.4.2.24.3 (AKM suites)).

30

1. The AP constructs a Key Delivery element indicating the current GTK and Key RSC, the current IGTK and
2. IPN if management frame protection is enabled. The GTK is carried in a GTK KDE with Tx bit equal to 0. The

33

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

35

36 .

37

38

1. For FILS shared key authentication, the KeyAuth field of the FILS Key Confirmation element is constructed
2. by using the HMAC mode of the negotiated hash function with a key of KCK on a concatenation of the AP’s
3. nonce, the STA’s nonce, the AP’s BSSID, the STA’s MAC address, and conditionally the AP’s public Dif-
4. fie-Hellman value and the STA’s public Diffie-Hellman value, in that order:

43

44

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

46

47 where:

48

49

1. — Hash is the hash function specific to the negotiated AKM.
2. — ANonce is the AP’s nonce and SNonce is the STA’s nonce.

52

53 — AP-BSSID is the BSSID of the AP and STA-MAC is the MAC address of the STA.

54

1. — gAP is the AP’s Diffie-Hellman public value and gSTA is the STA’s Diffie-Hellman public value.
2. — The brackets indicate the inclusion of the Diffie-Hellman public values when doing PFS with FILS
3. shared key authentication; there are no Diffie-Hellman public values to include otherwise.

58

59

1. For FILS public key authentication, the KeyAuth field of the FILS Key Confirmation element is a digital
2. signature using the AP’s private key of the output from the negotiated hash function on a concatenation of
3. the AP’s public Diffie-Hellman value, the STA’s public Diffie-Hellman value, the AP’s nonce, the STA’s

63

1. nonce, AP’s BSSID, and the STA’s MAC address, in that order. The specific construction of the digital sig-
2. nature depends on the crypto-system of the public/private keypair:

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

2

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

4

1. The form of signature depends on the type of public key used by the AP (IETF RFC 3447 for RSA; FIPS
2. 184-4 for DSA; and ISO/IEC 14888-3 for ECDSA). The data to be signed is first hashed and the hash algo-
3. rithm used with the appropriate digital signature algorithm shall be specific to the negotiated AKM.

8

9 The (Re)Association Response frame shall be encrypted using the AEAD algorithm as defined in [11.11.2.7](#_bookmark14)

10

1. [(AEAD cipher mode for FILS)](#_bookmark14) with the KEK as the key. The AAD used with the AEAD algorithm for the
2. (Re)Association Response frame is constructed by concatenating the following data together in order:

13

14 — The AP’s BSSID,

15

1. — The STA’s MAC address,
2. — The AP’s nonce,

18

1. — The STA’s nonce,
2. — The contents of the (Re)Association Response frame from the Capability Information field (inclu-

21

22 sive) to the FILS Session element (inclusive).

23

1. The plaintext passed to the AEAD algorithm is the data that would follow the FILS Session element in an
2. unencrypted frame. The output of the AEAD algorithm becomes the data that follows the FILS Session ele-
3. ment in the encrypted and authenticated (Re)Association Request frame. The output of the algorithm is as

27

28 specified in IETF RFC 5116. The resulting (Re)Association Response frame shall be transmitted to the STA.

29

1. The STA decrypts and verifies the received (Re)Association Response frame with the AEAD algorithm as
2. defined in [11.11.2.5 (Key establishment with FILS authentication)](#_bookmark10) with the KEK as the key. The AAD is

32

1. reconstructed as defined in this subclause above and is passed with the ciphertext of the received frame to
2. the AEAD decryption operation.

35

1. The STA compares FILS Session of the received frame with the FILS Session it selected to identify the
2. FILS session. If they differ, authentication shall be deemed a failure.

38

39

1. If the output from the AEAD decryption operation returns failure, the authentication exchange shall be
2. deemed a failure. If the output does not return failure, the output plaintext replaces the ciphertext as portion
3. of the frame that follows the FILS Session element and processing of the received frame continues by check-
4. ing the value of the FILS Key Confirmation element.

44

45

1. For FILS shared key authentication, the STA constructs a verifier, Key-Auth', in an identical manner as the
2. AP constructed its Key-Auth above.The STA verifies that the RSNE received in the (Re)Association
3. Response frame has identical AKM suites and cipher suites and RSN capabilities as were included in the
4. RSNE in the Beacon, Probe Response, and Authentication frames from the AP. If these fields differ, authen-

50

51 tication shall be deemed a failure.

52

1. The STA compares Key-Auth' with the KeyAuth field in the FILS Key Confirmation element of the
2. received frame. If they differ, authentication shall be deemed a failure.

55

56

1. For FILS public key authentication, the STA uses the AP’s (certified) public key from the FILS Public Key
2. element to verify that the signature contained in the KeyAuth field corresponds to the purported signature by
3. the AP over the concatenation of the following:

60

61

1. — AP’s public Diffie-Hellman value gAP,
2. — the STA’s public Diffie-Hellman value gSTA,

64

65 — the AP’s nonce ANonce,

1. — the STA’s nonce SNonce,
2. — the AP’s BSSID AP-BSSID,

3

4 — the STA’s MAC address STA-MAC,

5

6 in that order, according to the signature scheme used. Furthermore, the AP checks all certificates in the cer-

7

1. tificate chain, both cryptographically and from a security policy perspective, according to the procedures for
2. checking certificates and certificate chains in IETF RFC 5280. If any of these verifications fail, authentica-
3. tion shall be deemed a failure.

11

12 If authentication is deemed a failure, the KCK, KEK, PMK, and TK shall be irretrievably deleted and the

13

1. STA shall abandon the exchange. Otherwise authentication succeeds and the STA and AP shall irretrievably
2. delete the nonpersistent secret keying material that is created by executing the key establishment with FILS
3. shared key authentication scheme [(11.11.2.3 (Key establishment with FILS shared key authentication))](#_bookmark7) or
4. the key establishment with FILS public key authentication scheme [(11.11.2.4 (Key establishment with FILS](#_bookmark9)
5. [public key authenticati](#_bookmark9)on)). The KCK, KEK, and PMK shall be used for subsequent key management as

19

1. specified in [11.5 (RSNA security association management](#_bookmark0)). If the lifetime of the rMSK is known, the STA
2. and AP shall set the lifetime of the PMKSA to the lifetime of the rMSK. Otherwise, the STA and AP shall
3. set the lifetime of the PMKSA to the value dot11RSNAConfigPMKLifetime.

23

24 Upon successful completion of the FILS authentication procedure, the STA shall process the Key Delivery

25

1. element in the (Re)Association Response frame. The STA installs the GTK and key RSC, and IGTK and
2. IPN if management frame protection is enabled.

28

29

30

31

## 32 11.11.2.7 AEAD cipher mode for FILS

33

1. FILS authentication uses an AEAD cipher mode to protect (Re)Association Request/Response and EAPOL-
2. Key frames. The AEAD cipher mode is determined by the specific FILS AKM negotiated.

36

37

1. AES-SIV-256 is used when the AKM negotiated is 00-0F-AC:14 or 00-0F-AC:16 and AES-SIV-512 is used
2. when the AKM negotiated is 00-0F-AC:15 or 00-0F-AC:17.

40

41

42

43

44

45

46

47

48

49

50

51

52

53

54

55

56

57

58

59

60

61

62

63

64

65