IEEE P802.11  
Wireless LANs

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| PMK Caching for FILS | | | | |
| Date: 2014-01-10 | | | | |
| Author(s): | | | | |
| Name | Affiliation | Address | Phone | email |
| Kavitha Kamarthy | Aruba Networks | 1322 Crossman ave, Sunnyvale, CA | +1 408 227 4500 | kkamarthy at aruba networks dot com |
| Chirag Vaidya | Aruba Networks | 1322 Crossman ave, Sunnyvale, CA | +1 407 227 4500 | cvaidya at aruba networks dot com |
| Dan Harkins | Aruba Networks | 1322 Crossman ave, Sunnyvale, CA | +1 408 227 4500 | dharkina at aruba networks dot com |

Abstract

This document proposes a technique for fast re-authentication using FILS.

***Instruct editor to modify section 4.10.7 as indicated:***

**4.10.7 PMK Caching**

The Authenticator and Supplicant may cache PMKSAs, which include the IEEE Std 802.1X state. A PMKSA can be deleted from the cache for any reason and at any time.

When not performing FILS authentication, the STA may supply a list of PMK or PSK key identifiers in the (Re)Association Request frame. Each key identifier names a PMKSA; the PMKSA may contain a single PMK. The Authenticator specifies the selected PMK or PSK key identifier in Message 1 of the 4-Way Handshake. The selection of the key identifiers to be included within the (Re)Association Request frame and Message 1 of the 4-Way Handshake is out of the scope of this standard.

When performing FILS authentication, the STA may supply a list of PMK identifiers in the initial Authentication frame. Each PMK identifier names a PMKSA; the PMKSA contains a single PMK. If the AP has retained an identified PMKSA and wishes to facilliate a faster connection, it indicates use of a single identified PMKSA in its Authentication frame back to the STA. The STA and AP then use the PMK from the cached PMKSA in FILS handshaking to mutually authenticate. FILS authenticators that support PMK caching identify themselves to STAs using a cache identifier. A STA that has successfully established a PMKSA at an AP identifying a particular FILS authenticator can attempt to use PMK caching in a subsequent attempt at an AP that uses the same cache identifier.

***Intruct the editor to modify table 8-28 by inserting a new element at order 17 and incrementing the order of the existing 17 and all subsequent rows, and to modify table 8-29 as indicated:***

**8.3.3.11 Authentication frame format**

**Table 8-28—Authentication frame body**

|  |  |  |
| --- | --- | --- |
| Order | Information | Notes |
| 17 | PMKID List | The PMKID List is present in FILS Authenticaiton frames as defined in Table 8-29 (Presence of fields and elements in Authentication frames). |

**Table 8-29—Presence of fields and elements in Authentication frames**

|  |  |  |  |
| --- | --- | --- | --- |
| Authentication  Algorithm | Authentication  Transaction  Sequence no. | Status  code | Presence of fields 4-20 |
| FILS | 1 | Status | FILS Identity is present  FILS Authentication type is present.  FILS Nonce is present.  FILS Wrapped Data is present if FILS authentication uses a TTP.  Finite cyclic group is present if FILSAuthentication type field indicates PFS.  Element is present if FILSAuthentication type field  indicates PFS.  PMKID list is present if STA is asserting cached PMKs. |
| FILS | 2 | Status | FILS Identity is present if Status is zero.  FILS Authentication type is present if Status is zero.  FILS Nonce is present if Status is zero.  FILS Wrapped Data is present if Status is zero and a TTP is used.  Finite cyclic group is present if FILSAuthentication type field indicates PFS.  Element is present if FILSAuthentication type field indicates PFS.  PMKID list is present if AP agrees to perform PMK caching. |

***Instruct editor to modify section 8.4.2.185 as indicated:***

**8.4.2.185 FILS Indication element**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Element ID | Length | FILS  Information | Cache  Identifier  (conditional) | Domain information  (conditional) | Public Key  Information  (conditional) |

Octets 1 1 2 16 Variable Variable

**Figure 8-401cz—FILS Indication element**

The definitions of FILS Information field is as follows:

B0 B2 B3 B5 B6 B7 B8 B9 B10 B11 B12 B15

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| FILS Security  Type | Number of  Domains | IP Address  Assignment  Method | Subnet-ID  Token present | Public Cache Reserved  Key Supported  Info |

Bits 3 3 2 1 2 1 4

**Figure 8-401da—FILS Information field definition**

When PMK caching is supported the Authenticator of the AP chooses a random 16 octet number to identify itself to stations, inserts that number into the Cache Identifier field of the FILS Indication element, and sets the Cache Supported bit in the FILS Information field.

***Instruct the editor to create a new section 8.4.2.190:***

**8.4.2.190 PMKID list element**

The PMKID list contains a PMKID count followed by that number of PMKIDs. The size of each PMKID is 16 octets so the length of the PMKID list element is based on the number of PMKIDs included. The maximum number of PMKIDs in the list is 15 due to limitations on the size of an element (255 octets).

|  |  |  |  |
| --- | --- | --- | --- |
| Element ID | Length | PMKID count | Sequence of PMKIDs |

Octets: 1 1 1 Variable

**Figure 183dv—PMKID list IE**

***Instruct the editor to modify section 11.5.1.3.2 as indicated:***

**11.5.1.3.2 Security association in an ESS**

When FT is not enabled, a STA roaming within an ESS establishes a new PMKSA by one of three schemes:

* In the case of FILS authentication, the STA may optionally repeat the same actions as for initial contact and authentication. Alternately, the STA may attempt to use PMK caching and use an existing PMK from a previous FILS session to authenticate. Note that a STA can take advantage of the fact that it can initiate FILS authentication to multiple APs while maintaining a single association with one AP, and finalize the FILS authentication with one AP.

***Instruct editor to modify section 11.11.2.2.1 as indicated:***

**11.11.2.2.1 Key establishment with FILS shared key authentication**

STA may initiate FILS shared key authentication either with a FILS capable AP that is connected to a TTP authentication server that shares a valid key, called an rRK as defined in [IETF RFC 6696], with the STA; or, to a FILS capable AP with whom it shares a cached PMKSA. If neither of these cases apply,, a full EAP exchange may be performed via IEEE Std 802.1X authentication to establish rRK as defined in [IETF RFC 6696] or another form of FILS authentication may be used to establish a shared PMKSA.

If the STA chooses to initiate FILS shared key authentication the STA first chooses a random16 octet nonce. It then determines whether to attempt PMKSA caching. If so, it generates a list of one or more PMKSA identifiers, otherwise it constructs an EAP-Initiate/Re-auth packet as specified in [IETF RFC6696], with the following additional clarification:

* Regarding EAP-RP Flags
  + The 'B' flag shall be set to 0, indicating that this is not an EAP-RP bootstrap message.
  + The 'L' flag shall be set to 1, indicating that the TTP with whom the STA shares the rPK is to provide the lifetimes of rRK and rMSK in the EAP-Finish/Re-auth Packet.
* The "Cryptosuite" field shall not be set to 1.

If PFS is desired, the STA selects a finite cyclic group from the dot11RSNAConfigDLGGroupTable, generates an ephemeral secret private key, and performs the group's scalar-op (see 11.3.4.1) with its random ephemeral private key and the generator from the selected finite cyclic group to compute an ephemeral public key.

The STA then constructs an Authentication frame with the Authentication algorithm number set to <ANA-1> and the Authentication transaction sequence number set to one (1). The STA's FILS Identity shall be indicated using the FILS Identity element (see 8.4.2.179), the random nonce shall be encoded in the FILS nonce field (see 8.4.1.55), and the FILS authentication type shall be set to indicate the specific type of FILS authentication. If PMKSA caching is beint attempted, the PMKID list shall be constructed out of the list of PMKSA identifiers that are shared with the target AP, otherwise the EAP-Initiate/Re-auth packet shall be copied into the FILS authentication wrapped data field (see 8.4.2.188). If PFS is desired, the chosen finite cyclic group shall be encoded in the Finite Cyclic Group field (see 8.4.1.42) and the ephemeral public key shall be encoded in the Element field (see 8.4.1.40) according to the element to octet-string conversion in 11.3.7.2.4.

The STA shall transmit the Authentication frame to the AP.

If Authentication frame includes a Finite Cyclic Group field, then the AP shall first determine whether the indicated finite cyclic group in the received FILS authentication frame is supported. If not, it shall respond with an Authentication frame with the Authentication algorithm number set to <ANA-1> and the Status set to 77 (Authentication is rejected because the offered finite cyclic group is not supported) and shall terminate the exchange. If the group is supported or if PFS is not being used in this exchange, the AP shall check whether PMKSA caching is being attempted by the presence of the PMKID list element. If so, the AP checks whether any PMKSA identifier offered in the PMKID list matches an identifider for a cached PMKSA. If not, the AP shall respond with an Authentication frame with the Authentication algorithm number set to <ANA-1> and the Status set to 53 (invalid PMKID) and shall terminate the exchange. If PMKSA caching is not being attempted, the AP shall extract the EAP-Initiate/Re-auth data from the FILS authentication wrapped data field (see 8.4.2.188) and shall forward it to the TTP. When applicable, the AP communicates with the TTP using the same protocols with which it uses when authenticating with EAP. Suitable protocols include, but are not limited to, remote authentication dial-in user service (RADIUS) (IETF RFC 2863-2000) and Diameter (IETF RFC 3588-2003).

If PFS is being used, the AP shall also generate an ephemeral private key and perform the group's scalar-op (see 11.3.4.1) to produce its own ephemeral public key. The AP may delay the generation of its ephemeral public/private key pair until after receiving a response from the TTP, if applicable.

When PMKSA caching is not being used, the TTP processes the EAP-Initiate/Re-auth packet as specified in RFC6696 and returns an EAP-Finish/Reauth packet to the AP. In the case of successful authentication by the TTP, the TTP returns the associated EAP-RP rMSK with the EAP-Finish/Re-auth packet.

If the TTP responds with a failure indication, then the AP shall produce an Authentication frame with the Authentication algorithm number set to <ANA-1> and the Status set to 15 (Authentication rejected because of challenge failure). If the TTP responds with a success indication (including the associated EAP-RP rMSK), then the AP shall generate its own nonce and construct an Authentication frame for the STA. This frame shall contain the FILS wrapped data which encapsulates EAP-Finish/Re-auth packet received from the TTP.

When PMKSA caching is being used, the PMK from the cached PMKSA is used as AK for FILS key derivation. When PMKSA caching is not being used the rMSK is used as AK for FILS key derivation. (See 11.11.2.3).

In addition, if PFS is used, the Element field of the Authentication frame sent by the AP contains the AP's ephemeral public key. In this frame, the AP shall set the Authentication sequence number to (2).If PFS is being used for the exchange, then the AP shall perform the group's scalar-op (see 11.3.4.1) with the STA's ephemeral public key and its own ephemeral private key to produce an ephemeral Diffie-Hellman shared secret, ss.

Upon transmission of the FILS Authentication response, the AP shall perform key derivation per 11.11.2.3.

The STA processes the received Authentication frame.

1. If the received Authentication frame does not include the Authentication algorithm number set to <ANA-1>, or if PMKSA caching was attempted and the received Authentication frame does not include a PMKID list, or if PMKSA caching was not attempted and the received Authentication frame does not include an EAP-Finish/Re-auth packet, then the STA shall abandon the FILS authentication
2. If the received Authentication frame includes the Status set to 15 (Authentication rejected because of challenge failure) or 53 (invalid PMKID), then the STA shall abandon the FILS authentication
3. The STA ensures that the AP transmitted PFS parameters consistent with the desire of the STA (indicated by whether or not the STA transmitted an ephemeral public key).
4. If the STA transmitted an ephemeral public key, and the received Authentication frame does not include a well-encoded ephemeral public key, then the STA shall abandon the FILS authentication.
5. If the STA did not transmit an ephemeral public key desired PFS, and the received Authentication frame includes an ephemeral public key, then the STA shall abandon the FILS authentication.
6. If applicable, the STA processes the EAP-Finish/Re-auth packet as per RFC6696 –
7. If the 'R' flag = 0, indicating success, then the STA shall generate rMSK.
8. If the 'R' flag = 1, indicating failure, then the STA shall abandon the FILS authentication.
9. If PFS is being used for the exchange, then the STA shall perform the group's scalar-op (see 11.3.4.1) with the AP's ephemeral public key and its own ephemeral private key to produce an ephemeral Diffie-Hellman shared secret, ss.
10. The STA shall perform key derivation per 11.11.2.3.

If the STA doesn't get Authentication response, then the STA shall perform retransmission procedure as defined in IETF RFC 6696. If the retransmission procedure fails, then the STA shall abandon the FILS authentication and may perform full EAP authentication via IEEE 802.1X authentication.

***Instruct editor to modify section 11.11.2.3 as indicated:***

**11.11.2.3 Key derivation with FILS authentication**

Key derivation with FILS Authentication uses the KDF from 11.6.1.7.2 to produce six keys, two key encryption keys (KEK and KEK2), two confirmation keys (KCK and KCK2), a Pairwise Master Key (PMK), and a traffic key (TK). The inputs to the KDF are the two 16 octet nonces NSTA and NAP produced by the STA and AP, a constant label, the EAP-RP secret result if shared key authentication is being used, and, the Diffie-Hellman shared secret, ss, if PFS is being used or public key authentication is being used. The length of the KEK and KEK2 shall be 128 bits, and the length of the KCK, KCK2,and PMK shall be 256 bits, and therefore the output from the KDF shall be 1024+TK\_bits, where TK\_bits is determined from table 11-4.

KCK2 | KEK2 | KCK | KEK | PMK | TK = KDF-X(NSTA | NAP, “FILS KECK PTK Derivation”,

[AuthKey]|[ ss]))

Where X is 1024+TK\_bits from table 11-4.If shared key authentication was used AK is either the PMK from a cached PMKSA or is the rMSK, output by the EAP-RP exchange, if PMKSA caching was not used, and ss is the result of the Diffie-Hellman exchange if public key authentication was used or if PFS was used with shared key authentication.

When not using PMK caching, key derivation creates a new PMKSA. The PMK identifier for the new PMKSA shall be the SHA256 hash of the non-AP STA’s nonce and the AP’s nonce concatenated to 128 bits:

PMKID = L(SHA256(NSTA | NAP | STA-MAC | AP-BSSID), 0, 128)

When using PMK caching, a new PMKSA is not created. Instead, the PMKSA used in PMK caching remains and continues to be identified by the appropriate PMKID.

Upon completion of the key derivation computation, the shared secret ss and an rMSK, if applicable, shall be irretrievably destroyed.

The KCK2 shall only be used with key confirmation (see 11.11.2.4). The KEK2 shall only be used with the encrypt-and-authenticate (see 11.11.2.5) and decrypt-and-verify (see 11.11.2.6) functions. Both keys KCK2 and KEK2 are temporary and shall only be used during the FILS authentication protocol.

The keys KCK, KEK, and TK may be used after successful completion of the FILS authentication protocol and shall be used in exactly the same way as same-named keys of IEEE 802.11-2012 (but now derived as specified above).

**References:**