IEEE P802.11
Wireless LANs

|  |
| --- |
| Fixing PTK Derivation |
| Date: 2025-09-16 |
| Author(s): |
| Name | Affiliation | Address | Phone | email |
| Dan Harkins | HPE |  |  |  |
|  |  |  |  |  |

Abstract

This submission proposes resolution to CIDs 2095 and 2096.

|  |  |  |
| --- | --- | --- |
|  **CID** |  **Comment** | **Proposed Resolution** |
| 2095 | Need to create a transcript of the EAPOL exchange and use that in the KDF to generate the PTK later on. | Require STAs (incl APs) that use 802.1X authentication over authentication frames to generate a hashed transcript of the exchange. Each authentication frame in the sequence gets added to the running hash. The result of a successful EAPOL over Auth frames exchange will be a PMK, PMKID, and a transcript hash. Those two things can then be used later on to generate the PTK. There is no need for nonces and DH keys and the like. This will bind the entire exchange to the key it generated. |
| 2096 | This key confirmation and key derivation step should be a distinct authentication algorithm. Do not overload the EAPOL exchange with all this nonce and DH stuff. None of that is needed. | Require STAs (incl APs) that do EAPOL over Auth frames to immediately proceed to this exchange after successful completion of the EAPOL exchange. The result of the EAPOL exchange will be a PMK and a transcript hash. Those two things need to be bound with a domain separation tag in a KDF to derive the PTK. This is very simple. There are no DHs needed, no nonces. |

*Instruct editor to modify section 12.16.5 as indicated:*

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

If an AP sets the IEEE 802.1X Authentication Utilizing Authentication Frame Support field in the RSNXE

that it transmits to 1, then a non-AP STA (originator) with dot11EPPIEEE8021XAuthenticationUtilizingAuthenticationFrameActivated equal to true may signal its

Supplicant to authenticate with the AP (responder) using IEEE Std 802.1X-2020 utilizing Authentication frames.

If any AP affiliated with an AP MLD sets the IEEE 802.1X Authentication Utilizing Authentication Frame Support field in the RSNXE that it transmits to 1, then a non-AP MLD (originator) with dot11EPPIEEE8021XauthenticationUtilizingAuthenticationFrameActivated equal to true may signal its Supplicant to authenticate with the AP MLD (responder) using IEEE Std 802.1X-2020 utilizing Authentication frames by transmitting the Authentication frames to the AP through a non-AP STA affiliated with the non-AP MLD.

When the originator is a non-AP MLD and the responder is an AP MLD, the RA field of an Authentication frame in response to an Authentication frame from the peer shall be set to the TA field of the Authentication frame from the peer.

When performing IEEE 802.1X authentication utilizing Authentication frames, the non-AP MLD and the AP MLD shall compute a digest of the transcript of the messages comprising the IEEE 802.1X exchange. The transcript consists of the body of each Authentication in the order in which they were sent and received. The body is defined as the portion of the Authentication frame following the status code. The transcript of the exchange is hashed using the hash algorithm associated with the AKM from the AKM Suite Selector element.

If an originator chooses to initiate IEEE 802.1X authentication utilizing Authentication frames, it first selects an IEEE 802.1X AKM that is supported by the responder. The originator then shall construct the first Authentication frame of the exchange as follows:

* Authentication Algorithm Number field is set to 8 (IEEE 802.1X authentication).
* Authentication Transaction Sequence Number field is set to 1.
* The Encapsulation field carries an EAPOL PDU from the PAE.
* Include the AKM Suite Selector element indicating the selected IEEE 802.1X AKM.

The originator sends the first Authentication frame to the responder.

Upon receiving the first Authentication frame, the responder shall:

* Validate that the AKM indicated in AKM Suite Selector element is an IEEE 802.1X AKM. Otherwise, processing status is set to STATUS\_INVALID\_AKMP.
* Validate that the selected IEEE 802.1X AKM indicated in AKM Suite Selector element is supported. Otherwise processing status is set to STATUS\_INVALID\_AKMP.
* If the validation is successful, extract an EAPOL PDU from the Encapsulation field, and forward the EAPOL PDU to the PAE.

The responder then shall construct the second Authentication frame of the exchange as follows:

* Authentication Algorithm Number field is set to 8 (IEEE 802.1X authentication).
* Authentication Transaction Sequence Number field is set to 2.
* Status Code field indicates the processing status.
* The Encapsulation Length field indicates 0 if the status is set to STATUS\_INVALID\_AKMP.
* The Encapsulation field (if present) carries an EAPOL PDU from the PAE.
* Include the AKM Suite Selector element indicating the same IEEE 802.1X AKM indicated in the first Authentication frame.

Once the processing is complete, the responder sends the second Authentication frame to the originator. If

the processing status returned in the frame was not SUCCESS, the responder shall terminate the authentication.

*Instruct editor to modify section 12.16.8.2 as indicated:*

**12.16.8.2 IEEE 802.1X**

If an originator or a responder defined in 12.16.5 (IEEE 802.1X authentication utilizing Authentication frames) sets the (Re)Association Frame Encryption Support field in the RSNXE to 1, then the originator or the responder supports the additional rules defined in this subclause when performing IEEE 802.1X Authentication frame exchange.

An originator that sets the (Re)Association Frame Encryption Support field in the RSNXE to 1 receives an RSNXE from the responder with the (Re)Association Frame Encryption Support field set to 1, and intends to continue association after authentication shall do the following in the first Authentication frame:

* Include an RSNE to indicate the AKM and the pairwise cipher suite. The Version field shall be set to 1. The Pairwise Cipher Suite Count field shall be set to 1. The AKM Suite Count field shall be set to 1. The PMKID count field and the PMKID List field is set corresponding to PMKSA identifiers if exists. All other fields shall be as specified in 9.4.2.5 (TIM element) and 12.6.3 (RSNA policy selection in an infrastructure BSS).
* Include an RSNXE.

A responder that sets the (Re)Association Frame Encryption Support field in the RSNXE to 1 receives the first Authentication frame with an RSNE, and RSNXEshall:

* Verify that the AKM indicated in the RSNE is supported. Otherwise, the responder shall reject the first message with status code set to STATUS\_INVALID\_AKMP.
* Verify that the pairwise cipher indicated in the RSNE is supported. Otherwise, the responder shall reject the first message with status code set to STATUS\_INVALID\_PAIRWISE\_CIPHER.
* Verify that a PMKSA named via a PMKID in the RSNE exists for the specified AKM in the RSNE if one or more PMKIDs are included.

• If a PMKSA is identified, the responder shall use PMKSA caching, shall not process the EAPOL PDU in the first Authentication frame, and shall not include the EAPOL PDU in the second authentication frame.

• If no PMKSA is identified, continue the IEEE 802.1X authentication.

If the first Authentication frame is not rejected, the responder shall:

* Use PMKSA caching if a PMKSA is identified via a PMKID in the RSNE in the first Authentication frame and before sending the second Authentication frame:

• Derive PTK with the identified PMKSA as defined below.

The responder shall do the following in the second Authentication frame:

* Include an RSNE to indicate the AKM and pairwise cipher indicated in the first Authentication frame.

• If a PMKSA is identified via a PMKID in the RSNE in the first Authentication frame, the responder shall include the PMKID corresponding to the PMKSA in the RSNE.

• Otherwise, the responder shall not include any PMKID in the RSNE.

* Not include an AKM Suite Selector element.

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

* Validate that there is an RSNE included in the second Authentication frame and there is no AKM Suite Selector element in the second Authentication frame. If the validation fails, the originator shall discard the frame and terminate further protocol processing.
* Validate that the Encapsulation Length field is set to 0 and validate that the PMKID included in the second Authentication frame matches one of the PMKID(s) indicated in the first Authentication frame if the originator includes one or more PMKIDs in the first Authentication frame, and the second Authentication frame includes a PMKID. If verification succeeds, the originator shall use PMKSA caching with the PMKSA identified by the PMKID indicated in the second Authentication frame and shall not continue the IEEE 802.1X Authentication frame exchange. If verification fails, the originator shall discard the frame and terminate further protocol processing.
* Validate that there is no PMKID included in the second Authentication frame if the originator does not include any PMKID in the first Authentication frame. If verification fails, the originator shall discard the frame and terminate further protocol processing.
* Derive the PTK as defined belowif a PMKSA is identified.

If a PMKSA is not identified through PMKSA caching, before sending the Authentication frame carrying EAP Success, a responder shall:

* Derive the PTK as defined below.

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

* Derive the PTK as defined below.

The PTK shall be derived using the PMK, either cached or as received from the successful completion of EAP, and a hash of the transcript of the EAP exchange as:

 PTK = HKDF-expand(HKDF-extract(T, PMK), “IEEE 802.11 Auth PTK Derivation”, PTKLen)

Where:

 PMK is the shared secret created as a result of running the PQC algorithm

 T is a digest of the transcript from the authentication algorithm that generated the PMK

 PTKLen is the length of the PTK from 12.7.1.3

**References:**