|  |
| --- |
| CR for CID208 |
| Date: 2024-1-14 |
| Author(s): |
| Name | Affiliation | Address | Phone | email |
| Jay Yang | ZTE Corporation |  |  | Yang.zhijie@zte.com.cn |
| Yan Li | ZTE Corporation |  |  |  |
| Yurong qian | ZTE Corporation |  |  |  |

Abstract

This submission proposes resolutions for the following CIDs:

R0: initial the draft

Interpretation of a Motion to Adopt

A motion to approve this submission means that the editing instructions and any changed or added material are actioned in the TGbh D1.0 Draft. This introduction is not part of the adopted material.

***Editing instructions formatted like this are intended to be copied into the TGbh D1.0 Draft. (i.e. they are instructions to the 802.11 editor on how to merge the text with the baseline documents).***

***TGbh Editor: Editing instructions preceded by “TGbh Editor” are instructions to the TGbh editor to modify existing material in the TGbh draft. As a result of adopting the changes, the TGbh editor will execute the instructions rather than copy them to the TGbh Draft.***

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **CID** | **Commenter** | **Clause** | **P.L** | **Comment** | **Proposed Change** | **Resolution** |
| 208 | Po-Kai Huang | 12.13.7 | 39.1 | This condition of not deriving KEK is not correct. When the peer does not support both device ID or IRM, then KEK is also not needed, otherwise, the key location of TK for both sides will be wrong. A better solution seems to just introduce an additional capability bit say "KEK in PASN" then KEK is dervied if both sides set this bit to 1 and otherwise do not derive KEK. This will likley simply the design and prevent complicated condition check to avoid interop issue. | Commenter will submit a contributon for the proposal since description of the change for this in a comment resolution box is difficult. The general direction is to introduce new capability bit in RSNXE called KEK in PASN and derive KEK based on the capability bit set to 1 on both sides rather than feature support, which is likely to have interop issues since the draft also gets it wrong. Then we can have condition like if support IRM or device ID, then set this capability bit to 1. | Revised – Agree in principle with the commenter. TGbh editor to make the changes shown in 11-24/118r0under all headings that include CID 208 |
| 209 | Po-Kai Huang | 12.13.7 | 39.30 | This condition of not deriving KEK is not correct. When the peer does not support both device ID or IRM, then KEK is also not needed, otherwise, the key location of TK for both sides will be wrong. A better solution seems to just introduce an additional capability bit say "KEK in PASN" then KEK is dervied if both sides set this bit to 1 and otherwise do not derive KEK. This will likley simply the design and prevent complicated condition check to avoid interop issue. | Commenter will submit a contributon for the proposal since description of the change for this in a comment resolution box is difficult. The general direction is to introduce new capability bit in RSNXE called KEK in PASN and derive KEK based on the capability bit set to 1 on both sides rather than feature support, which is likely to have interop issues since the draft also gets it wrong. Then we can have condition like if support IRM or device ID, then set this capability bit to 1. | Revised – Agree in principle with the commenter. TGbh editor to make the changes shown in 11-24/118r0 under all headings that include CID 208 |

**12.13.7 PTKSA derivation with PASN authentication**

….(existing texts)….

(CID208)-If none of the following condition is met:

1. Dot11IRMActivated is true and IRM Active field in the Extended RSN Capabilities field from the peer is 1.
2. Dot11DeviceIDactivated is true and Device ID Active field in the Extended RSN Capabilities field from the peer is 1.

, PTK is composed of the Key Confirmation Key (KCK), Temporal Key (TK) and the Key Derivation Key (KDK) which are derived as follows:

KCK = L(PTK, 0, 256)

KCK is the first 256 bits of the PTK.

TK = L(PTK, 256, TK\_Length\_Bits)

TK is the transient key whose length is the same as a key for the pairwise cipher in RSNE provided by the AP in the second PASN frame This length is 16 octets for all ciphers, except for the ciphers 00-0F-AC:9 and 00-0F-AC:10 for which it is 32 octets.

KDK = L(PTK, 256 + TK\_Length\_Bits, KDK\_bits)

...(existing texts).....

***Insert the following text as shown.***

(CID208)-If any of the following condition is met:

1. -When dot1IRMActivated is true and IRM Active field in the Extended RSN Capabilities field from the peer is 1.
2. When dot11DeviceIDActivated is true and Device ID Active field in the Extended RSN Capabilities field from the peer is 1 ,

 PTK is composed of the Key Confirmation Key (KCK), Key Encryption Key (KEK), Temporal Key (TK) and the Key Derivation Key (KDK) which are derived as follows (see Table 12-11 (Integrity and key wrap algorithms)):

KCK = L(PTK, 0, 256)

KCK is the first 256 bits of the PTK.

KEK = (PTK, 256, KEK\_bits)

...(existing texts).....