### IEEE P802.11Wireless LANs

|  |
| --- |
| 11be D0.3 CR for 12.4 |
| Date: 2021-02-17 |
| Author(s): |
| Name | Affiliation | Address | Phone | email |
| Po-Kai Huang | Intel Corporation | 2200 Mission College Blvd, Santa Clara, CA 950542200  |  | po-kai.huang@intel.com |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

Abstract

This submission proposes resolutions for the following CIDs:

2864, 2284, 2285, 2286, 2487, 2576, Mark’s comments

Revisions:

* Rev 0: Initial version of the document.
* Rev 1: Editorial revision.
* Rev 2: Further revision based on the received feedback.
* Rev 3: Editorial revision based on comments received offline.
* Rev 4: Revision based on the discussion offline.
* Rev 5: Further editorial revision based on the discussion offline.

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 TGbe D0.3 Draft. This introduction is not part of the adopted material.

***Editing instructions formatted like this are intended to be copied into the TGbe D0.3 Draft (i.e. they are instructions to the 802.11 editor on how to merge the text with the baseline documents).***

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

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **CID** | **Commenter** | **P.L** | **Clause** | **Comment** | **Proposed Change** | **Resolution** |
| 2487 | Po-Kai Huang | 114.64 | 12.4.5.2 | The formula of val computation needs to be specified between two MLDs. Simply changing (STA-A-MAC, STA-B-MAC) to (MLD-A-MAC, MLD-B-MAC) for the formula between two MLDs | Changing (STA-A-MAC, STA-B-MAC) to (MLD-A-MAC, MLD-B-MAC) for the formula between two MLDs. | Revised – We revise the description of STA and MLD with general description “SAE entity” to simplify the description across the board. We then revise the paragraphs describing MAC addresses correspondingly.TGbe editor to make the changes shown in 11-21/0260r5 under all headings that include CID 2487 |
|  | Mark Rison | 114.63 | 12.4.5.2 | "between two STAs or MLD-A-MAC and MLD-B-MAC shall be used in the computation of val" is not clear | say MAC-A and MAC-B in the equation, and then in the “where” below describe how they map in the MLD and non-MLD cases | Revised – We revise the description of STA and MLD with general description “SAE entity” to simplify the description across the boardWe then revise the paragraphs describing MAC addresses correspondingly.TGbe editor to make the changes shown in 11-21/0260r5 under all headings that include CID 2487 |
| 2864 | Stephen McCann | 112.30 | 12.4.1 | An "AP STA" is an "AP". | Change all occurances of "AP STA" to "AP". | Revised – We revise the description of STA and MLD with general description “SAE entity” to simplify the description across the board. Note that authentication can only be done between two STAs or two MLDs based on 11.3.TGbe editor to make the changes shown in 11-21/0260r5 under all headings that include CID 2864 |
| 2284 | Michael Montemurro | 113.32 | 12.4.3 | This should just be MLD AP since that is the identity of the Authenticator. | Change "AP or APs affiliated with the AP MLD, respectively," to "AP MLD" | Revised – We create separate paragraph to clarify the following. For an AP MLD, the dot11RSNAConfigPasswordValueTable for all affiliated APs of the AP MLD shall be identitical to the dot11RSNAConfigPasswordValueTable for the AP MLD. Consequently, all affiliated APs of the AP MLD shall advertise the same values for the SAE Password Identifiers In Use and SAE Password Identifiers Used Exclusively subfields of the Extended Capabilities field of the Extended Capabilities element.TGbe editor to make the changes shown in 11-21/0260r5 under all headings that include CID 2284 |
| 2285 | Michael Montemurro | 113.32 | 12.4.3 | I hope for MLD operations that with MLDs this is still a BSS. | Delete "or an AP MLD" | Revised – We create separate paragraph to clarify the following. For an AP MLD, the dot11RSNAConfigPasswordValueTable for all affiliated APs of the AP MLD shall be identitical to the dot11RSNAConfigPasswordValueTable for the AP MLD. Consequently, all affiliated APs of the AP MLD shall advertise the same values for the SAE Password Identifiers In Use and SAE Password Identifiers Used Exclusively subfields of the Extended Capabilities field of the Extended Capabilities element.TGbe editor to make the changes shown in 11-21/0260r5 under all headings that include CID 2284 |
| 2286 | Michael Montemurro | 113.32 | 12.4.3 | This should just be MLD AP since that is the identity of the Authenticator. | Replace "or Aps affiliated with the AP MLD" with "or AP MLD" | Revised – We create separate paragraph to clarify the following. For an AP MLD, the dot11RSNAConfigPasswordValueTable for all affiliated APs of the AP MLD shall be identitical to the dot11RSNAConfigPasswordValueTable for the AP MLD. Consequently, all affiliated APs of the AP MLD shall advertise the same values for the SAE Password Identifiers In Use and SAE Password Identifiers Used Exclusively subfields of the Extended Capabilities field of the Extended Capabilities element.TGbe editor to make the changes shown in 11-21/0260r5 under all headings that include CID 2284 |
|  | Mark Rison | 113.32 | 12.4.3 | "32 In an infrastructure BSS or an AP MLD" -- A BSS and an MLD are not the same kind of thing. | delete and change “with the AP MLD, respectively,” to “with an AP MLD” | Revised – We create separate paragraph to clarify the following. For an AP MLD, the dot11RSNAConfigPasswordValueTable for all affiliated APs of the AP MLD shall be identitical to the dot11RSNAConfigPasswordValueTable for the AP MLD. Consequently, all affiliated APs of the AP MLD shall advertise the same values for the SAE Password Identifiers In Use and SAE Password Identifiers Used Exclusively subfields of the Extended Capabilities field of the Extended Capabilities element.TGbe editor to make the changes shown in 11-21/0260r5 under all headings that include CID 2284 |
| 2576 | Rojan Chitrakar | 114.63 | 12.4.5.2 | It is not apparent how the two MLDs find out each other's MLD MAC Addresses. Presumably, AP MLD would advertise its MLD MAC Address in Beacon/Probe Response frames and non-AP MLD would include its MLD MAC Address in the Authentication frame. However, can't find this information explicitely anywhere in D0.3. Clause 9 only mentions that ML element may be carried in Beacon/ Probe Response frames and Authentication frames but the content of ML element in these frames are not explained in clause 35. | Explain how the two MLDs would find out each other's MLD MAC Addresses for the PWE generation in SAE. | Rejected –  We clarify below that discovery of MLD address has been addressed in other places of the spec.An authentication frame sent by a STA affiliated with an MLD includes an ML element that indicates the MLD MAC address of that MLD (see 35.3.5.4 Usage and rules of Basic variant Multi-link element in the context of multi-linksetup). A Beacon frame sent by an AP affiliated with an AP MLD includes an ML element that indicates the MLD MAC address of that AP MLD, when that Beacon frame indicates SAE AKM (see 35.3.4.3 (Multi-link element usage rules in the context of discovery). A Probe Response frame sent by an AP affiliated with an AP MLD includes an ML element that indicates the MLD MAC address of the AP MLD in response to a MLD Probe Request frame (see 35.3.4 (Discovery of an AP MLD)).  |

**Discussion:** *None.*

**Propose:**

***TGax editor: Change 12.4 Authentication using a password as follows (track change on):***

* Authentication using a password
* SAE overview

***Change the now-shifted third paragraph and split it into two paragraphs as follows:***

Two SAE entities may authenticate each other by proving possession of a password. (#2864)

Authentication protocols that employ passwords need to be resistant to off-line dictionary attacks.

***Change the now-shifted fifth paragraph as follows:***

Simultaneous authentication of equals (SAE) is a variant of *Dragonfly*, a password-authenticated key exchange based on a zero-knowledge proof. SAE is used by SAE entities to authenticate with a password; it has the following security properties: (#2487)

* The successful termination of the protocol results in a PMK shared between the two SAE entities. (#2487)
* An attacker is unable to determine either the password or the resulting PMK by passively observing an exchange or by interposing itself into the exchange by faithfully relaying messages between the two SAE entities. (#2487)
* An attacker is unable to determine either the password or the resulting shared key by modifying, forging, or replaying frames to an honest, uncorrupted SAE entity. (#2487)
* An attacker is unable to make more than one guess at the password per attack. This implies that the attacker cannot make one attack and then go offline and make repeated guesses at the password until successful. In other words, SAE is resistant to dictionary attack.
* Compromise of a PMK from a previous run of the protocol does not provide any advantage to an adversary attempting to determine the password or the shared key from any other instance.
* Compromise of the password does not provide any advantage to an adversary in attempting to determine the PMK from the previous instance.

***Change the now-shifted seventh paragraph as follows:***

The parties involved are called *Entity-A* and *Entity-B*. They are identified by their MAC addresses, denoted A-MAC and B-MAC. Between two MLDs, the MAC addresses of Entity-A and Entity-B are their MLD MAC addresses. Between two STAs, the MAC addresses of Entity-A and Entity-B are their STA MAC addresses. . Two SAE entities begin the protocol when they discover a peer by receiving Beacon or Probe Response frame(s), or when they receive an Authentication frame indicating SAE authentication from a peer. (#2487)

* Representation of a password

***Change as follows:***

Passwords are used in SAE to deterministically compute a secret element in the negotiated group, called a password element. The input to this process needs to be in the form of a binary string. For the protocol to successfully terminate, it is necessary for each side to produce identical binary strings for a given password, even if that password is in character format. There is no canonical binary representation of a character and ambiguity exists when the password is a character string. To eliminate this ambiguity, an SAE entity shall represent a character-based password as a UTF-8 string that is processed according to the OpaqueString profile of IETF RFC 8265, the output of which is an octet string. The octet string representation of the password, after being processed, is stored in the dot11RSNAConfigPasswordValueTable. When a “password” is called for in the description of SAE that follows the credential from the dot11RSNAConfigPasswordValueTable is used. (#2487)

Similarly, to address ambiguity when identifying passwords, an SAE entity shall represent a password identifier as a UTF-8 string that is processed according to the UsernameCasePreserved profile of IETF RFC 8265, the output of which is an octet string that is stored in the dot11RSNAConfigPasswordValueTable. When a “password identifier” is called for in the description of SAE that follows, the identifier from the dot11RSNAConfigPasswordValueTable is used. (#2487)

In an infrastructure BSS for which an SAE AKM is indicated, the AP shall set the SAE Password Identifiers In Use subfield of the Extended Capabilities field of the Extended Capabilities element to 1 if any entry in the dot11RSNAConfigPasswordValueTable has a non-NULL dot11RSNAConfigPasswordIdentifier, and shall set it to 0 otherwise. Similarly, an AP shall set the SAE Password Identifiers Used Exclusively subfield of the Extended Capabilities field of the Extended Capabilities element to 1 if every entry in the dot11RSNAConfigPasswordValueTable has a non-NULL dot11RSNAConfigPasswordIdentifier and shall set it to 0 otherwise. (#2284)

For an AP MLD, the dot11RSNAConfigPasswordValueTable for all affiliated APs of the AP MLD shall be identitical to the dot11RSNAConfigPasswordValueTable for the AP MLD. Consequently, all affiliated APs of the AP MLD shall advertise the same values for the SAE Password Identifiers In Use and SAE Password Identifiers Used Exclusively subfields of the Extended Capabilities field of the Extended Capabilities element.(#2284)

* Finite cyclic groups
* General

***Change the first two paragraphs as follows:***

SAE uses discrete logarithm cryptography to achieve authentication and key agreement. Each party to the exchange derives ephemeral public and private keys with respect to a particular set of domain parameters that define a finite cyclic group. Groups may be based on either finite field cryptography (FFC) or on elliptic curve cryptography (ECC). Each component of a group is referred to as an *element*. Groups are negotiated using an identifying number from a repository maintained by IANA as “Group Description” attributes for IETF RFC 2409 (IKE) [B14][B28]. The repository maps an identifying number to a complete set of domain parameters for the particular group. Not all groups defined in this repository are suitable. Only FFC groups whose prime is at least 3072 bits and ECC groups defined over a prime field whose prime is at least 256 bits are suitable for use with SAE. ECC groups defined over a characteristic 2 finite field or ECC groups with a co-factor greater than 1 shall not be used with SAE (see NIST Special Publication 800-57). For the purpose of interoperability, an SAE entity shall implement support for group 19, an ECC group defined over a 256-bit prime order field. (#2487)

More than one group may be configured on an SAE entity for use with SAE by using the dot11RSNAConfigDLCGroupTable. Configured groups are prioritized in ascending order of preference. If only one group is configured, it is, by definition, the most preferred group. (#2487)

* Elliptic curve cryptography (ECC) groups
* Hash-to-curve generation of the password element with ECC groups

***Change the first paragraph as follows:***

An SAE entityindicates support for direct hashing to obtain an ECC password element by setting the SAE hash-to-element bit in the Extended RSN Capabilities field in all Beacon and Probe Response frames. An SAE entity that uses a password identifier shall use the hash-to-curve method. An SAE initiator that has identified a peer that supports this technique (through receipt of Beacon or Probe Response frames) shall derive a secret element, PT, according to the following technique and indicate this by setting the status code in the SAE Commit message to SAE\_HASH\_TO\_ELEMENT. An SAE initiator shall not indicate support for this form of element derivation unless its peer has already signalled support for this method. If an SAE Commit message is received with status code equal to SAE\_HASH\_TO\_ELEMENT the peer shall generate the PWE using the following technique and reply with its own SAE Commit message with status code set to SAE\_HASH\_TO\_ELEMENT. (#2487)

* Finite field cryptography (FFC) groups
* Direct Generation of the password element with FFC groups

***Change the first paragraph as follows:***

An SAE peer indicates support for direct hashing to obtain the FFC password element by setting the SAE hash-to-element bit in the Extended RSN Capabilities field in all Beacon and Probe Response frames. An SAE entity that uses a password identifier shall use the direct hashing technique. An SAE initiator that has identified a peer that supports the following technique (through receipt of Beacon or Probe Response frames) shall derive PT according to the following technique and indicate this by setting the status code in the SAE Commit message to SAE\_HASH\_TO\_ELEMENT. An SAE initiator shall not indicate support for this form of PWE derivation unless its peer has already signalled support. If an SAE Commit message is received with status code equal to SAE\_HASH\_TO\_ELEMENT the peer shall generate the PWE using the following technique and reply with its own SAE Commit message with status code set to SAE\_HASH\_TO\_ELEMENT. (#2487)

* SAE protocol
* PWE and secret generation

***Change the second paragraph as follows:***

When an SAE entity supports directly hashing to a group element (according to 12.4.4.2.3 (Hash-to-curve generation of the password element with ECC groups) or 12.4.4.3.3 (Direct Generation of the password element with FFC groups)) it computes a secret element, PT, offline at provisioning time for all groups it wishes to support with that password. Prior to initiating SAE to an SAE entity which also supports the direct form of hashing to a group element, or upon receipt of an SAE Commit message indicating it was generated using a direct form of hashing to a group element, it shall generate the PWE by hashing the two peer MAC addresses to produce a digest, reducing the digest modulo the order of the particular group, *r*, interpreting the reduced digest as an integer and using it with the secret element to generate the PWE: (#2487)

 *val = H(0n, MAX(A-MAC, B-MAC) || MIN(A-MAC, B-MAC))*

 *val = val* modulo *(r – 1) + 1*

 *PWE = scalar-op(val, PT)*

where 0n is a salt of all zeros whose length equals the length of the digest from the hash function used to instantiate H() (see Table 12-1 (Hash algorithm based on length of prime)). (#2487)

***Change the fourth paragraph as follows:***

After generation of the ***PWE***, each SAE entity shall generate a secret value, *rand*, and a temporary secret value, *mask*, each of which shall be chosen randomly such that 1 < *rand* < *r* and 1 < *mask* < *r* and (*rand + mask*)mod *r* is greater than 1, where *r* is the (prime) order of the group. If their sum modulo r is not greater than 1, they shall both be irretrievably deleted and new values shall be randomly generated. The values *rand* and *mask* shall be random numbers produced from a quality random number drawn from a uniform distribution generator. These values shall never be reused on distinct protocol runs. (#2487)

* Processing of a peer’s SAE Commit message

***Change the first two paragraphs as follows:***

If the peer’s SAE Commit message contains a password identifier, the value of that identifier shall be used in construction of the password element (PWE) for this exchange. If a password identifier is present in the peer’s SAE Commit message and there is no password with the given identifier an SAE entity shall fail authentication. (#2487)

If the peer’s SAE Commit message contains a Rejected Groups element, the list of rejected groups shall be checked to ensure that all of the groups in the list are groups that would be rejected. If any groups in the list would not be rejected then processing of the SAE Commit message terminates and the SAE entity shall reject the peer’s authentication. While the rejected groups are appended to the Rejected Groups element as they are rejected (see 12.4.7.4 (Encoding and decoding of SAE Commit messages)) there is no inherent order to the groups in the list. The order in which they are sent and received shall be retained when deriving keys. (#2487)

***Change the fifth and sixth paragraphs as follows:***

If either scalar validation or element validation fails, the SAE entity shall reject the peer’s authentication. If both the scalar and element from the peer’s SAE Commit message are successfully validated, a shared secret element, *K*, shall be derived using the scalar and element (*peer-commit-scalar* and ***PEER-COMMIT-ELEMENT***, respectively) from the peer’s SAE Commit message and the SAE entity’s secret value. (#2487)

***K***= scalar-op(*rand*, (elem-op(scalar-op(*peer-commit-scalar*, ***PWE***), ***PEER-COMMIT-ELEMENT***)))

If the shared secret element, ***K***, is the identity element for the negotiated group (the value one for an FFC group or the point-at-infinity for an ECC group) the SAE entity shall reject the peer’s authentication. Otherwise, a secret value, *k*, shall be computed as: (#2487)

*k* = F(***K***)

* Anti-clogging tokens

***Change the first paragraph as follows:***

An SAE entity is required to do a considerable amount of work upon receipt of an SAE Commit message. This opens up the possibility of a distributed denial-of-service attack by flooding an SAE entity with bogus SAE Commit messages from forged MAC addresses. To prevent this from happening, an SAE entity shall maintain an *Open* counter in its SAE state machine indicating the number of open and unfinished protocol instances (see 12.4.5.1 (Message exchanges)). When that counter hits or exceeds dot11RSNASAEAntiCloggingThreshold, the SAE entity shall respond to each SAE Commit message with a rejection that includes an Anti-Clogging Token field statelessly bound to the sender of the SAE Commit message. The sender of the SAE Commit message shall then include the Anti-Clogging Token field in a subsequent SAE Commit message. (#2487)

* SAE finite state machine
* Events and output
* Parent process events and output

***Change the fourth paragraph as follows:***

Receipt of frames containing SAE messages signals the following events to the SAE parent process:

* *Authentication frame with Transaction Sequence number 1*—This event indicates that an SAE Commit message has been received from a peer. (#2487)
* *Authentication frame with Transaction Sequence number 2*—This event indicates that an SAE Confirm message has been received from a peer. (#2487)