IEEE P802.11
Wireless LANs

|  |
| --- |
| Rule-based random MAC STA identification (RRCM) |
| Date: 2022-06-12 |
| Author(s): |
| Name | Affiliation | Address | Phone | email |
| Orhan Okan Mutgan | Nokia |  |  | okan.mutgan@nokia-sbell.com |
| Jay Yang | Nokia |  |  | yang.zhijie@nokia-sbell.com |

Abstract

This document provides a text for rule-based random MAC STA identification as presented in 22/818r1.

According to the requirements in 802.11bh, the proposal (22/818r1) introduces four different solution versions: v1, v2, v3, v4. This document focuses on v2 and v3.

In v2, a non-AP STA and AP generate the same Random MAC address (RMA) to be used in the next association(s) with the help of the same formula available to both sides. To verify that a non-AP STA and AP generate the same MAC address, the non-AP STA sends a tag to the AP, and AP verifies the tag locally. Upon verification, the non-AP STA can use the RMA in its next association(s) and will be identified by the AP.

v3 is an enchanment of v2, where a non-AP STA and AP generate multiple RMAs (RMA1, RMA2 etc.) during association procedure. In this scenario, the non-AP STA can use generated RMAs in different frames (e.g RMA1 in probe request frame, RMA2 in other frames) and still be identified by AP.

For simplicity, the name of the proposal is presented as Rule-based Random and Changing MAC (RRCM) in this document.

SP/Motion: Do you agree the proposed text in 11-22/888r0 should be incroprated into TGbh Amendment?

# Text modifications (Proposed text modifications are based on Draft 802.11REVme\_D1.1)

***1) Add following definition to 3.2***

**Rule-based Random and Changing MAC (RRCM):** A privacy enhanchment mechanism for non-AP STA and AP to generate Random Mac Address – RMA (used by non-AP STA) in order to prevent non-AP STA from being tracked (from third parties) and still allow the non-AP STA to be identified by the AP. “Rule-based” implies that the non-AP STA and AP generate the same RMA or RMAs locally at their sides.

***2) Add a new capability information to Table 9-190 Extended Capabilities field***

|  |  |  |
| --- | --- | --- |
| **Bit** | **Information** | **Notes** |
| <ANA> | RRCM Capability | The STA sets RRCM Capability subfield to 1 to indicate support for RRCM Capability and sets to 0 if not supported. |

***3) Add a new subclause in 12.2 Framework:***

**12.2.11 Rule-based Random and Changing MAC (RRCM)**

**12.2.11.1 General**

To improve privacy, a non-AP STA may use MAC randomization (STAs using random MAC address) while still being identified by the AP. Rule-based Random and Changing MAC (RRCM) identification mitiages this issue; namely, when a non-AP STA associates to an AP with its MAC address, the non-AP STA may change its MAC address (RMA = Random Mac Address) when it returns to the same AP and ESS, and still can be recognized by the AP and ESS.

in RRCM, A non-AP STA and AP generate the same MAC address or addresses (along with the Tag) to be used (by the non-AP STA) in the next association(s) with the help of the same formula available to both sides, using the same parameters. Among these parameters, several of them (such as Seed, Nonce, Counter, Tag) are exchanged between the non-AP STA and AP, and several of them (such as keys – K1 and K2) are generated locally at both sides. To verify that a non-AP STA and AP generate the same MAC address, the non-AP STA generates a tag (a hash value based on RMAs and the key) and sends the tag to the AP, and AP then verifies the tag. Upon matching of these two tags, the non-AP STA can use the generated RMAs in its next association(s) and be identified by the AP.

A non-AP STA and AP may generate single RMA, which the non-AP STA can use in all frames, or multiple RMAs (RMA1, RMA2 etc.), which the non-AP STA can use in different frames (e.g. RMA1 in probe request frame, RMA2 in other frames).

The STA advertises the support for RRCM by setting the RRCM Capability subfield to 1 in the Extended Capabilites Element.

The relevant items (the generation of RMAs, keys, and Tag) for RRCM are explained in 12.2.11.2. The identification procedure is explained in 12.2.11.3.

**12.2.11.2 RMA, Keys, and Tag Generation**

The equations to generate the RMA, keys (K1 and K2), and Tag are as folows:

**K1[128:]** = PBKDF2(PTK, ssid, 4096, 256)

**K2[:128]** = PBKDF2 (PTK, ssid, 4096, 256)

**RMA[:48]** = AES-CTR(K1, Nonce||counter, Seed)

**Tag** = HMAC-SHA-256(K2, RMA(s))

Where,

* PBKDF2 (defined in IETF RFC 2898) is used to generate K1 and K2, where passPhrase = PTK (see suclause J.4 Suggested pass-phrase-to-PSK mapping). The output of this function is 256 bits.
* K1 is the first 128 bits derived from PTK by the PBKDF2 formula
* K2 is the last 128 bits derived from PTK by the PBKDF2 formula
* AES-CTR is defined in IETF RFC 3686 to derive 48-bit RMA: The output value of each block in AES-CTR will be 128 bits (128 bits block size), in which the first 48 bits are used for RMA.
* Counter is 32 bits unsigned integer value.

The counter value is the number of generated RMA:

If only one RMA is generated, counter value is equal to zero, whereas in multiple RMA generation case, counter value is equal to zero initially and incremented until the number of generated RMA. As an example,

If single RMA is generated, counter=0,

If three RMA are generated, counter=0 generates first RMA, counter=1 generates second RMA, counter=2 generates third RMA.

* Seed is a 128-bit random bit string.

In each plaintext block in AES-CTR, the same 128-bit seed is used.

* Nonce is 128-bit random bit string.
* HMAC-SHA-256 is defined in IETF RFC 4868 to generate 256-bit Tag. The Tag is used to verify that non-AP STA and STA generate the same RMA.
* RMA(s) in the HMAC-SHA-256 formula is the resulting ciphertext that contains generated RMA string.

As en example, ciphertext will be 48 bits string for single RMA, and 48\*3=144 bits for three RMA.

The block diagram of RMA and Tag generation is as follows:



Figure xxx – An Example of Generation of 3 RMA and the Tag

NOTE1-- In each association, each parameter {K1, K2, Seed, Nonce} is re-generated and counter is reset to zero, resulting in new generated RMA(s) and Tag in each association.

NOTE2-- {K1, K2, Seed, Nonce} are deleted on STA side after disassociation.

NOTE3-- I/G = 0 and U/L = 1 bits shall be replaced in each generated RMA, see subclause 12.2.10.

NOTE4-- RMA and Tag shall be saved on non-AP STA and AP/ESS side until new RMA and Tag are generated.

NOTE5-- Note that in single or multiple RMA generation, only one Tag is sufficient.

**12.2.11.3 Identification Procedure**

During the association procedure, the non-AP STA and AP derive K1 and K2 from PTK (see K1 and K2 generation in subclause **12.2.11.2**).

Non-AP STA behaviour:

The non-AP STA then initializes {Seed, Nonce, Counter} values to locally generate one or more RMAs and one {Tag} (see RMA and Tag generation in subclause **12.2.11.2**). When using FILS authentication, the non-AP STA sends the {Seed, Nonce, Counter, Tag} in IE in the Association Request frame. When using FT, the non-AP STA sends the {Seed, Nonce, Counter, Tag} during the initial mobility domain association in enctypted Key Data field (RRCM KDE) in the EAPOL-Key message 2/4. {Seed, Nonce, Counter, Tag} is not exchanged during the FT protocol reassociations within the same ESS. For other cases, the non-AP STA sends the {Seed, Nonce, Counter, Tag} in enctypted Key Data field (RRCM KDE) in the EAPOL-Key message 2/4.

AP behaviour:

After receiving {Seed, Nonce, Counter, Tag} from the the EAPOL-Key message 2/4 or Association Request frame in FILS authentication mode sent by the non-AP STA, the AP first checks the {Counter} value to determine the number of RMAs it needs to generate locally. Upon determining, AP generates the same number of RMAs (that non-AP STA generated) and one {Tag} (see RMA and Tag generation in subclause **12.2.11.2**). AP then verifies its generated {Tag} with the {Tag} sent from non-AP STA, i.e. if AP generated {Tag} is equal to the non-AP STA generated {Tag}, AP must have generated the same RMAs.

After the non-AP STA is disassociated, {K1, K2, Seed, Nonce} are deleted, while {RMAs, Tag, Counter} are stored at non-AP STA and the (previously) associated AP or ESS.

The non-AP STA may use the generated RMAs in its frames for the next association, and the AP or ESS can then identify the non-AP STA with its RMAs.

Note— The usage of which RMA for which frame is based on implementation.

***4) Add a new KDE to Table 12-10 KDE selectors:***

|  |
| --- |
| * KDE selectors
 |
| OUI | Data type | Meaning |
| 00-0F-AC | 15 | WIGTK KDE |
| 00-0F-AC | 16 | RRCM KDE |
| 00-0F-AC | 17–255 | Reserved |
| Other OUI or CID | Any | Vendor specific |

***5) Add the new KDE (RRCM KDE) to 12.7.2 EAPOL-Key frames:***

The format of the RRCM KDE is shown in Figure 12-49 (RRCM KDE format).

|  |  |  |  |
| --- | --- | --- | --- |
| Seed | Nonce | Counter | Tag |

 Octets 16 16 4 32

Figure 12-49—RRCM KDE format

Seed, Nonce, and Counter are values to generate one or more RMA for RRCM procedure. The Tag corresponds to the calculated HMAC value for RRCM procedure. For details, see subclause **12.2.11.**

***5) Add “RRCM KDE” to 12.7.4 EAPOL-Key frame notation:***

 OCI KDE is a KDE containing operating channel information

 RRCM KDE is a KDE containing {Seed, Nonce, Counter, Tag} to be used for RRCM procedure

 RSNXE is described in 9.4.2.241 (RSN Extension element (RSNXE))

 PMKID identifies the PMKSA selected by the Authenticator

 “{a} or {b}” means that exactly one of either {a} or {b} is present as the {Key Data}

***6) Modify 12.7.6.1 General (under 12.7.6 4-way handshake):***

Message 1: Authenticator  Supplicant: EAPOL-Key(0,0,1,0,P,0,0,ANonce,0,{} or {PMKID})

Message 2: Supplicant  Authenticator: EAPOL-Key(0,1,0,0,P,0,0,SNonce,MIC,{RSNE} or {RSNE, OCI KDE} or {RSNE, RSNXE} or {RSNE, OCI KDE, RSNXE} or {RSNE, RRCM KDE} or {RSNE, OCI KDE, RRCM KDE} or {RSNE, RSNXE, RRCM KDE} or {RSNE, OCI KDE, RSNXE, RRCM KDE})

Message 3: AuthenticatorSupplicant:
EAPOL-Key(1,1,1,1,P,0,KeyRSC,ANonce,MIC,{RSNE,GTK[N]} or
{RSNE, GTK[N], OCI KDE} or {RSNE, GTK[N], RSNXE} or
{RSNE, GTK[N], OCI KDE, RSNXE})

Message 4: Supplicant  Authenticator: EAPOL-Key(1,1,0,0,P,0,0,0,MIC,{}).

***7) Modify 12.7.6.3 4-way handshake message 2:***

Key Information:

Key Descriptor Version = 1 (ARC4 encryption with HMAC-MD5) or 2 (NIST AES key wrap with HMAC-SHA-1-128) or 3 (NIST AES key wrap with AES-128-CMAC), in all other cases 0 – same as message 1

Key Type = 1 (Pairwise) – same as message 1

Reserved = 0

Install = 0

Key Ack = 0

Key MIC = 0 when using an AEAD cipher or 1 otherwise

Secure = 0 – same as message 1

Error = 0 – same as message 1

Request = 0 – same as message 1

Encrypted Key Data = 1 when using an AEAD cipher or when RRCM KDE is included, or 0 otherwise

Reserved = 0 – unused by this protocol version

* Key Data =
	+ - * Additionally, contains RRCM KDE to carry the {Seed, Nonce, Counter, Tag} for RRCM KDE procedure

***8) Add new row in Table 9-62 – Association Request frame body***

|  |  |  |
| --- | --- | --- |
| **Order** | **Information** | **Notes** |
| <ANA> | RRCM | The RRCM element is present when using FILS authentication; otherwise, it is not present. |

***9) Add a new row in Table 9-128 – Element IDs in 9.4.2.1 General (under 9.4.2 Elements)***

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Element** | **Element ID** | **Element ID Extension** | **Extensible** | **Fragmentable** |
| RRCM (see 9.4.2.296 RRCM element) | 255 | <ANA> | No | No |

***10) Add a new subclause 9.4.2.296 (under 9.4.2 Elements)***

9.4.2.296 RRCM element

The RRCM element contains Seed, Nonce, Counter, Tag fields that are used in RRCM procedure. The format of the RRCM element is shown in Figure 9-xxx (RRCM element format).

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Element ID | Length | Element ID Extension | Seed | Nonce | Counter | Tag |

Octets 1 1 1 16 16 4 32

Figure 9-xxx - RRCM element format

The Element ID, Length, and Element ID Extension fields are defined in 9.4.2.1 (General).

Seed, Nonce, and Counter are values to generate one or more RMA for RRCM procedure. The Tag corresponds to the calculated HMAC value for RRCM procedure. For details, see subclause **12.2.11.**