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 22/818r4.

In the proposal, a non-AP STA and AP generate the same non-AP STA Random MAC address or addresses (RMA(s)) to be used in the next association(s) through following the same procedure. The non-AP STA can use the RMA(s) in its next association(s) and will be identified by the AP. In single RMA generation case, the non-AP STA can use the generated RMA in each frame in its next association. In multiple RMA(s) case, 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.

R1: Modified some text.

R2: For key generation K1, PBKDF2 is replaced with KDF-Hash-256, and PTK is replaced with KDK.

 K1 is renamed as RMAK, K2 generation is discarded. For RMA(s) generation, AES-CTR is replaced with KDF-Hash-48. Tag usage is discarded. Relevant text is modified.

SP/Motion: Do you agree the proposed text in 11-22/888r2 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 Address (RRCM):** A privacy enhancement mechanism for non-AP STA and AP to generate one or more Random Mac Addresses (RMA) for use by non-AP STA to prevent non-AP STA from being tracked (by third parties) and still allow the non-AP STA to be identified by the AP in subsequent message exchanges. “Rule-based” implies that the non-AP STA and AP apply the same procedures for generating RMA or RMA(s) locally at their sides.

**RMAK (RMA Key):** RMAK is the key that is used to generate one or more Random Mac Addresses (RMA) for RRCM procedure.

***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 Address (RRCM)**

**12.2.11.1 General**

To improve its privacy, a non-AP STA may desire to use a random MAC address (RMA) while still being identifiable by the same AP in subsequent associations. Rule-based Random and Changing MAC address (RRCM) allows for identification despite randomly changed MAC address at later associations., When a non-AP STA associates to an AP with one MAC address, it can still be recognized by the AP and ESS after the non-AP STA changes its MAC address before reconnecting to the same AP and ESS.

Through RRCM, a non-AP STA and AP can generate the same ‘randomized’ MAC address or addresses to be used by the non-AP STA in the next association(s) based on a common procedure through a total of three parameters. Among these parameters, two of them (Seed, Counter) are exchanged between the non-AP STA and AP, and one of them (the key – RMAK) is generated locally at both sides.

A non-AP STA and AP may generate a single RMA, which the non-AP STA can use in all message exchanges, or multiple RMAs (RMA1, RMA2 etc.), which the non-AP STA can use in different message exchanges (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 Capabilities Element.

The relevant items (the generation of RMA(s) and RMAK) for RRCM are explained in 12.2.11.2. The identification procedure is explained in 12.2.11.3.

**12.2.11.2 RMA and Key Generation**

The procedures to generate the RMA(s) and key, RMAK, are as follows:

**RMAK** = KDF-Hash-256(KDK, "RMA Key", Min(ANonce, SNonce) || Max(ANonce, SNonce)

**RMAn** = KDF-Hash-48(RMAK, "Next RMAs", seed || n)

Where,

* KDF-Hash-256 will generate 256 bits key, RMAK. Hash is the Hash algorithm used in the AKM that the STA and AP agreed upon. KDK is derived from PTK for RRCM procedure. ANonce and SNonce are the generated values from 4-way Handshake. “RMA Key” is the string name for RMAK and is treated as an ASCII string.
* KDF-Hash-48 will generate 48-bit RMA. Seed is a 128-bit random bit string generated at non-AP STA. n is initialized with 1 and incremented by 1 until n is equal to Counter, which is the number of generated RMA(s). As an example, if three RMAs are generated, Counter=3 implies that n=1 is used to generate RMA1, n=2 is used to generate RMA2, n=3 is used to generate RMA3. The length of the counter is 16 bits, resulting in maximum 2^16 different RMA(s) generation in each association.

NOTE1-- In each association, the non-AP STA may decide to generate one or more RMA(s), where each parameter {RMAK, Seed} is re-generated and Counter is reset to one.

NOTE2-- I/G = 0 and U/L = 1 bits shall be replaced in each generated RMA, see subclause 12.2.10.

NOTE3--RMA(s) may be saved on non-AP STA and AP/ESS side until new RMA(s) are generated.
NOTE4 – When RRCM is negotiated, The PTK is partitioned into KCK, KEK, TK, and a KDK. KDK is used to derive RMAK.

**12.2.11.3 Identification Procedure**

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

Non-AP STA behaviour:

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

AP behaviour:

After receiving {Seed, Counter} from the non-AP STA in the the EAPOL-Key message 2/4 or Association Request frame in FILS authentication mode, the AP first checks the {Counter} value to determine the number of RMA(s) it needs to generate locally. The AP generates the same number of RMA(s) that non-AP STA generated (see RMA generation in subclause **12.2.11.2**).

After the non-AP STA have been disassociated, {RMAK, Seed} are deleted and {Counter} is reset to 1, while RMA(s) are stored at non-AP STA and at the (previously) associated AP or ESS.

The non-AP STA may use the generated RMAs for messaging, preparing, and establishing the next association. The AP or ESS can then identify the non-AP STA despite changing MAC addresses through comparison of the MAC addresses with its stored RMAs.

Note— The usage of which RMA(s) 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 | Counter |

 Octets 16 2

Figure 12-49—RRCM KDE format

Seed and Counter are values to generate one or more RMA(s) through 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, Counter} 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, Counter} 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 and Counter 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 | Counter |

Octets 1 1 1 16 2

Figure 9-xxx - RRCM element format

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

Seed and Counter are values to generate one or more RMA for RRCM procedure. For details, see subclause **12.2.11.**

***11) Add the following changes relevant to the use of KDK***

**a. (P342,line 1) 4.10.3.2 AKM operations with AS**

— If WUR frame protection is negotiated or RRCM genration is negotiated , derive a fresh WTK from the KDK

**b. (P3173,line30) under 12.6.1.1.6 PTKSA**

PTK(11ba), where the PTK includes the KDK when WUR frame protection is negotiated or RRCM is generated.

**c. (P3199,Line 64) under 12.7.1.1 General**

a) Pairwise key hierarchy, to protect individually addressed traffic(11ba), where the PTK includes a KDK if WUR frame protection is negotiated or RRCM genration is negotiated and excludes the KDK otherwise.

**d. (P3201, Line 50) under 12.7.1.3 Pairwise key hierarchy**

The PTK is partitioned into KCK, KEK, (11ba)a temporal key, and a KDK if WUR frame protection is negotiated or RRCM genration is negotiated ;otherwise the PTK is partitioned into KCK, KEK, and a temporal key. The temporal key is used by the MACto protect individually addressed communication between the Authenticator’s and Supplicant’s respectiveSTAs. If WUR frame protection is negotiated, the KDK is used to derive a WTK, which is used by the MACof the WUR AP to protect and by the MAC of the WUR non-AP STA to validate individually addressed WUR Wake-up frames. PTKs are used between a single Supplicant and a single Authenticator. If RRCM genration is negotiated,the KDK is used to derive a RRMK, which is used to generate a batch of RMAs that are carried by the non-AP STA and identified by the AP.

**e. (P3202, Line 59) under 12.7.1.3 Pairwise key hierarchy**

where (11ba)Length = KCK\_bits + KEK\_bits + TK\_bits + KDK\_bits, if WUR frame protection is being negotiated or RRCM genration is being negotiated ;

**f.(P3203, Line 4) under 12.7.1.3 Pairwise key hierarchy**

(11ba)If WUR frame protection is being negotiated or RRCM genration is being negotiated, the KDK shall be computed as the next

KDK\_bits bits of the PTK:

KDK = L(PTK, KCK\_bits+KEK\_bits+TK\_bits, KDK\_bits)

Otherwise, the KDK is not derived

**g. (P3203,Line 32) under 12.7.1.3 Pairwise key hierarchy**

11ba)If WUR frame protection is negotiated, the WTK shall be derived from the KDK using the KDF

defined in 12.7.1.6.2:

WTK = KDF-Hash-Length(KDK, “WUR Temporal Key”, Min(AA,SPA) || Max(AA,SPA) ||

Min(ANonce,SNonce) || Max(ANonce,SNonce)

where

— KDF-Hash-Length is the key derivation function as defined in 12.7.1.6.2 (Key derivation function

(KDF)) using the hash algorithm identified by the AKM suite selector (see Table 9-188 (AKM suite

selectors)).

— Length is the total number of bits to derive, i.e., number of bits of the WTK, and is equal to 128.

If RRCM is negotiated, the RRCMK shall be derived from the KDK using the KDF defined in 12.7.1.6.2. see subclause 12.2.11.2 **RMA and Key Generation**

**h. (P3211,Line 24) under 12.7.1.6.4 PMK-R1**

1ba)When WUR frame protection is negotiated or RRCM genration is negotiated, each PTK has six component keys, KCK, KEK, a

temporal key, KCK2, KEK2, and a KDK derived as follows:

(11ba)The KCK, KEK, temporal key, KCK2, and KEK2 shall be computed in the same way as when WUR frame protection is not negotiated.

(11ba)The KDK shall be computed as the next KDK\_bits bits of the PTK:

KDK = L(PTK, KCK\_bits+KEK\_bits+TK\_bits+KCK2\_bits+KEK2\_bits, KDK\_bits)

(11ba)The value of KDK\_bits is equal to the value of PMK\_bits (see 12.7.1.3 (Pairwise key hierarchy)).

**i. (insert the following change after the referenced baseline context in P3211,line 38) under 12.7.1.6.4 PMK-R1**

(11ba)If WUR frame protection is negotiated, the WTK shall be derived from the KDK using the KDF

defined in 12.7.1.6.2 (Key derivation function (KDF))):

WTK = KDF-Hash-Length(KDK, “WUR Temporal Key”, SNonce || ANonce || BSSID ||

STA-ADDR)

where

— KDF-Hash-Length is the key derivation function as defined in 12.7.1.6.2 (Key derivation function

(KDF)) using the hash algorithm identified by the AKM suite selector (see Table 9-188 (AKM suite

selectors)).

— Length is the total number of bits to derive, i.e., number of bits of the WTK, and is equal to 128.

(11ba)The WTK is used to protect individually addressed WUR Wake-up frames, as defined in 29.10 (WUR

frame protection).

If RRCM is negotiated, the RRCMK shall be derived from the KDK using the KDF defined in 12.7.1.6.2. see subclause 12.2.11.2 **RMA and Key Generation**

**j. (P3226, line 42) under 12.7.6.2 4-way handshake message 1**

b) Derives PTK(11ba), the derived PTK including the Key derivation key (KDK) if WUR frame protection is being negotiated or RRCM genration is being negotiated .

**k. (P3269, line 54) under 12.11.2.5.3 PTKSA Key derivation with FILS authentication**

When the negotiated AKM is 00-0F-AC:16,FILS-FT is 256 bits; when the negotiated AKM is 00-0F-AC:17, FILS-FT is 384 bits; otherwise, FILS-FT is

not derived(11ba); when WUR frame protection is negotiated or RRCM genration is negotiated, the length of KDK is equal to the value of PMK\_bits (see 12.7.1.3 (Pairwise key hierarchy)); otherwise, the KDK is not derived.

**m. (P3270,line7) under 12.11.2.5.3 PTKSA Key derivation with FILS authentication**

11ba)When WUR frame protection is negotiated or RRCM genration is negotiated while doing FT initial mobility domain association using

FILS authentication,

KDK = L(PTK(#1778), ICK\_bits + KEK\_bits + TK\_bits + FILS-FT\_bits, KDK\_bits)

(11ba)When WUR frame protection is negotiated while not doing FT initial mobility domain association

using FILS authentication,

KDK = L(PTK(#1778), ICK\_bits + KEK\_bits + TK\_bits, KDK\_bits)

**n. (insert the following change after the referenced baseline context P3270,line 46) under 12.11.2.5.3 PTKSA Key derivation with FILS authentication**

11ba)If WUR frame protection is negotiated, the WTK shall be derived from the KDK using the KDF

defined in 12.7.1.6.2 (Key derivation function (KDF)):

WTK = KDF-Hash-Length(KDK, “WUR Temporal Key”, SPA || AA || SNonce || ANonce [ ||DHss ])

If RRCM is negotiated, the RRCMK shall be derived from the KDK using the KDF defined in 12.7.1.6.2. see subclause 12.2.11.2 **RMA and Key Generation**