**3GPP TSG- Meeting #**

**, , -**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *CR-Form-v12.3* | | | | | | | | |
| **CHANGE REQUEST** | | | | | | | | |
|  | | | | | | | | |
|  |  | **CR** | **0282** | **rev** |  | **Current version:** |  |  |
|  | | | | | | | | |
| *For* [***HE******LP***](http://www.3gpp.org/3G_Specs/CRs.htm#_blank)*on using this form: comprehensive instructions can be found at* [*http://www.3gpp.org/Change-Requests*](http://www.3gpp.org/Change-Requests)*.* | | | | | | | | |
|  | | | | | | | | |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ***Proposed change affects:*** | UICC apps |  | ME | **X** | Radio Access Network |  | Core Network | **X** |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | | | | | | | | | | |
| ***Title:*** |  | | | | | | | | | |
|  |  | | | | | | | | | |
| ***Source to WG:*** | Deutsche Telekom AG, Huawei, HiSilicon | | | | | | | | | |
| ***Source to TSG:*** |  | | | | | | | | | |
|  |  | | | | | | | | | |
| ***Work item code:*** |  | | | | |  | ***Date:*** | | |  |
|  |  | | | |  | |  | | |  |
| ***Category:*** |  |  | | | | | ***Release:*** | | |  |
|  | *Use one of the following categories:* ***F*** *(correction)* ***A*** *(mirror corresponding to a change in an earlier release)* ***B*** *(addition of feature),* ***C*** *(functional modification of feature)* ***D*** *(editorial modification)*  Detailed explanations of the above categories can be found in 3GPP [TR 21.900](http://www.3gpp.org/ftp/Specs/html-info/21900.htm). | | | | | | | | *Use one of the following releases: Rel-8 (Release 8) Rel-9 (Release 9) Rel-10 (Release 10) Rel-11 (Release 11) … Rel-17 (Release 17) Rel-18 (Release 18) Rel-19 (Release 19)  Rel-20 (Release 20)* | |
|  |  | | | | | | | | | |
| ***Reason for change:*** | | TS 33.203 describes that "Hmac-sha-1-96" and "aes-cbc" are not recommended. But SHA-1 is no longer secure. Computing collisions is affordable, even for academic adversaries. It should never be used for cryptographic hashing.  To increase security this CR proposes to add the HMAC-SHA2-256 according to RFC 6234. This algorithm is state of the art and recommended by national security agencies like NIST or BSI. There exist already end devices which support SHA2. | | | | | | | | |
|  | |  | | | | | | | | |
| ***Summary of change:*** | | Add HMAC-SHA2-256 algorithm according to RFC 6234 | | | | | | | | |
|  | |  | | | | | | | | |
| ***Consequences if not approved:*** | | Security requirements according to national security agencies canot be aplied. | | | | | | | | |
|  | |  | | | | | | | | |
| ***Clauses affected:*** | |  | | | | | | | | |
|  | |  | | | | | | | | |
|  | | **Y** | **N** |  | | | |  | | |
| ***Other specs*** | |  |  | Other core specifications | | | | TS/TR ... CR ... | | |
| ***affected:*** | |  |  | Test specifications | | | | TS/TR ... CR ... | | |
| ***(show related CRs)*** | |  |  | O&M Specifications | | | | TS/TR ... CR ... | | |
|  | |  | | | | | | | | |
| ***Other comments:*** | |  | | | | | | | | |
|  | |  | | | | | | | | |
| ***This CR's revision history:*** | |  | | | | | | | | |

\*\*\* BEGIN OF CHANGE 1 \*\*\*

Annex H (normative):  
The use of "Security Mechanism Agreement for SIP Sessions" [21] for security mode set-up

The BNF syntax of RFC 3329 [21] is defined for negotiating security associations for semi-manually keyed IPsec or TLS in the following way:

security-client = "Security-Client" HCOLON sec-mechanism \*(COMMA sec-mechanism)

security-server = "Security-Server" HCOLON sec-mechanism \*(COMMA sec-mechanism)

security-verify = "Security-Verify" HCOLON sec-mechanism \*(COMMA sec-mechanism)

sec-mechanism = mechanism-name \*(SEMI mech-parameters)

mechanism-name = "ipsec-3gpp" / "tls"

mech-parameters = ( preference / algorithm / protocol / mode / encrypt-algorithm / spi‑c / spi‑s / port‑c / port‑s )

preference = "q" EQUAL qvalue

qvalue = ( "0" [ "." 0\*3DIGIT ] ) / ( "1" [ "." 0\*3("0") ] )

algorithm = "alg" EQUAL ("hmac-sha-1-96" / “hmac-sha2-256” / "aes-gmac" / "aes-gmac-us " / "null" )

protocol = "prot" EQUAL ( "ah" / "esp" )

mode = "mod" EQUAL ( "trans" / "tun" / "UDP-enc-tun" )

encrypt-algorithm = "ealg" EQUAL ("aes-cbc" / "aes-gcm" / "aes-gcm-us" / "null" )

spi‑c = "spi‑c" EQUAL spivalue

spi‑s = "spi‑s" EQUAL spivalue

spivalue = 10DIGIT; 0 to 4294967295

port‑c = "port‑c" EQUAL port

port‑s = "port‑s" EQUAL port

port = 1\*DIGIT

The changes compared to RFC 3329 [21] are:

"alg" parameter: Addition of “hmac-sha2-256”,"aes-gmac","aes-gmac-us" and "null". Removal of "hmac-md5-96"

"ealg" parameter: Addition of "aes-cbc, "aes-gcm-us", and "aes-gcm".   
 Removal of "des-ede3-cbc"

"mod" parameter: Addition of "UDP-enc-tun"

"Hmac-sha-1-96" and "aes-cbc" are not recommended.

The use of security association parameters is specified in clauses 7.1, 7.2, M.7.1 and M.7.2 of the present document. The parameters described by the BNF above have the following semantics:

Mechanism-name: For manually keyed IPsec, this field includes the value "ipsec-3gpp". "ipsec‑3gpp" mechanism extends the general negotiation procedure of RFC 3329 [21] in the following way:

1 The server shall store the Security-Client header received in the request before sending the response with the Security-Server header.

2 The client shall include the Security-Client header in the first protected request. In other words, the first protected request shall include both Security-Verify and Security-Client header fields.

3 The server shall check that the content of Security-Client headers received in previous steps (1 and 2) are the same.

Mech-parameters: Of the mech-parameters, only preference is relevant when the mechanism-name has the value "tls".

Preference: As defined in RFC 3329 [21].

Algorithm: Defines the authentication algorithm. The algorithm parameter is mandatory. The value "aes-gmac" refers to the authentication algorithm ENCR\_NULL\_AUTH\_AES\_GMAC defined in IETF RFC 4543 [74]. The value "aes-gmac-us" refers to the same algorithm with "aes-gmac" but only a different salt value generation method — "us" standing for unique salt. The value "null" shall only be used with encryption algorithm "aes-gcm".

Protocol: Defines the IPsec protocol. May have a value "ah" or "esp". If no Protocol parameter is present, the value will be "esp".

NOTE 1: According to clause 6 only "esp" (RFC 4303 [54]) is allowed for use in IMS.

Mode: Defines the mode in which the IPsec protocol is used. May have a value "trans" for transport mode, and value "tun" for tunneling mode. If no Mode parameter is present, the value will be "trans".

NOTE 2: Void.

Encrypt-algorithm: If present, defines the encryption algorithm. The value "aes-cbc" refers to the algorithm defined in IETF RFC 3602 [22]. The value "aes-gcm" refers to the encryption algorithm AES-GCM with a 16 octet ICV defined in IETF RFC 4106 [73]. The value "aes-gcm-us" refers to the same algorithm with "aes-gcm" but only a different salt value generation method — "us" standing for unique salt. If no Encrypt-algorithm parameter is present, the algorithm will be "null". The value "aes-gcm" shall only be used with authentication algorithm equal to "null".

Spi‑c: Defines the SPI number of the inbound SA at the protected client port.

Spi‑s: Defines the SPI number of the inbound SA at the protected server port.

Port‑c: Defines the protected client port.

Port‑s: Defines the protected server port.

It is assumed that the underlying IPsec implementation supports selectors that allow all transport protocols supported by SIP to be protected with a single SA.

\*\*\* END OF CHANGE 1 \*\*\*

\*\*\* BEGIN OF CHANGE 2 \*\*\*

Annex I (normative):  
Key expansion functions for IPsec ESP

**Integrity Keys:**

If the selected authentication algorithm is HMAC-SHA-1-96 then IKESP is obtained from IKIM by appending 32 zero bits to the end of IKIM to create a 160‑bit string.

If the selected authentication algorithm is HMAC-SHA2-256 then IKESP shall be derived using the key derivation function KDF defined in Annex B of TS 33.220 [66]. The input Key to the KDF function shall be equal to the concatenation of CKIM and IKIM: CKIM || IKIM.

If selected authentication algorithm is AES-GMAC as specified in RFC 4543 [74] with 128 bit key then IKESP = IKIM.

The salt value specified in Section 3.2 of RFC 4543 [74] shall be derived using the key derivation function KDF defined in Annex B of TS 33.220 [66]. The input Key to the KDF function shall be equal to the concatenation of CKIM and IKIM: CKIM || IKIM.

If the " algorithm " value is set to "aes-gmac" when negotiating the SA using RFC 3329[21] as shown in Annex H, the input S to the KDF function shall be formed from the following parameters:

- FC = 0x58.

- P0 = "AES\_GMAC\_SALT" .

- L0 = length of the string “AES\_GMAC\_SALT” (i.e. 0x00 0x0D).

The salt value shall consist of the 32 least significant bits of the 256 bits of the KDF output. This salt value derivation method is not recommended.

If the "algorithm" value is set to "aes-gmac-us" when negotiating the SA [21] as shown in Annex H, salt value for each IPsec SA shall consist of the 32 least significant bits of the 256 bits of the KDF output XOR’d with the 2 bits — one bit representing for the direction of the SA ("0" for UE to P-CSCF, "1" for P-CSCF to UE) and one bit representing for the role of the source (UE or P-CSCF) of the SA ("0" for client, "1" for server). The direction bit will be XOR’d with the LSB of the 32-bit string, which is extracted from the 256-bit output of the KDF. The role bit will be XOR’d with the second LSB of the 32-bit string, which is extracted from the 256-bit output of the KDF.

"Hmac-sha-1-96" and "aes-gmac" are not recommended.

**Encryption Keys:**

If selected encryption algorithm is AES‑CBC as specified in RFC 3602 [22] with 128 bit key then CKESP = CKIM .

If selected encryption algorithm is AES‑GCM as specified in RFC 4106 [73] with 128 bit key then CKESP = CKIM. The salt value specified in Section 4 of RFC 4106 [73] shall be derived using the key derivation function KDF defined in Annex B of TS 33.220 [66]. The input Key to the KDF function shall be equal to the concatenation of CKIM and IKIM: CKIM || IKIM.

When the " algorithm " value is "aes-gcm" when negotiating the SA[21] as shown in Annex H, the input S to the KDF function shall be formed from the following parameters:

- FC = 0x59

- P0 = “AES\_GCM\_SALT”

- L0 = length of the string “AES\_GCM\_SALT” (i.e. 0x00 0x0C)

The salt value shall consist of the 32 least significant bits of the 256 bits of the KDF output. This salt value derivation method is not recommended.

When the "algorithm" value is "aes-gcm-us" when negotiating the SA [21] as shown in Annex H, the salt value for each IPsec SA shall consist of the 32 least significant bits of the 256 bits of the KDF output XOR’d with the 2 bits — one bit representing for the direction of the SA ("0" for UE to P-CSCF, "1" for P-CSCF to UE) and one bit representing for the role of the source (UE or P-CSCF) of the SA ("0" for client, "1" for server). The direction bit will be XOR’d with the LSB of the 32-bit string, which is extracted from the 256-bit output of the KDF. The role bit will be XOR’d with the second LSB of the 32-bit string, which is extracted from the 256-bit output of the KDF.

"aes-cbc" and "aes-gcm" are not recommended.

\*\*\* END OF CHANGE 2 \*\*\*