|  |  |
| --- | --- |
| Project | **IEEE 802.21a**  **<https://mentor.ieee.org/802.21>** |
| Title | **Option III Draft updates** |
| DCN | 21-10-0209-00-0sec |
| Date Submitted |  |
| Source(s) | Fernando Bernal-Hidalgo (University of Murcia), Rafael Marin-Lopez (University of Murcia) |
| Re: |  |
| Abstract | This document elaborates changes in the current draft |
| Purpose | Proposes changes in the current draft |
| Notice | This document has been prepared to assist the IEEE 802.21 Working Group. It is offered as a basis for discussion and is not binding on the contributing individual(s) or organization(s). The material in this document is subject to change in form and content after further study. The contributor(s) reserve(s) the right to add, amend or withdraw material contained herein. |
| Release | The contributor grants a free, irrevocable license to the IEEE to incorporate material contained in this contribution, and any modifications thereof, in the creation of an IEEE Standards publication; to copyright in the IEEE’s name any IEEE Standards publication even though it may include portions of this contribution; and at the IEEE’s sole discretion to permit others to reproduce in whole or in part the resulting IEEE Standards publication. The contributor also acknowledges and accepts that IEEE 802.21 may make this contribution public. |
| Patent Policy | The contributor is familiar with IEEE patent policy, as stated in [Section 6 of the IEEE-SA Standards Board bylaws](http://standards.ieee.org/guides/opman/sect6.html" \l "_blank) <[http://standards.ieee.org/guides/bylaws/sect6-7.html#6](http://127.0.0.1:4664/cache?event_id=757737&schema_id=1&s=5X0vID10lu_E6yrIkWkNd4Wz2H8&q=hancock" \l "_blank)> and in *Understanding Patent Issues During IEEE Standards Development* [http://standards.ieee.org/board/pat/faq.pdf](http://standards.ieee.org/board/pat/faq.pdf" \l "_blank) |

|  |  |
| --- | --- |
| DRAFT SECTION | 3. DEFINITION |
| Modification type | ADD |
| Text | **PoS:** acts as authenticator for the service authentication. Moreover, it is the entity that interacts with PoAs to facilitate key distribution services.  **MIH Service AS:** It is a backend authentication server for the MIH service authentication.  **Candidate PoS:** A PoS that is a potential target to the MN’s movement  **Target PoS:** A PoS that is the PoS selected for the MN’s movement  **Serving PoS:** A PoS that is the current PoS which provides access to the supported network services.  **Candidate PoA:** A PoA that is a potential target to the MN’s network attachment.  **Target PoA:** A PoA that is the PoA selected to perform a key distribution  **Serving PoA:** A PoA that is the current PoA where the MN is attached. Moreover, it provides network access to the MN. |

|  |  |
| --- | --- |
| DRAFT SECTION | 4. DEFINITION |
| Modification type | ADD |
| Text | **PoS:** Point of Service  **PoA:** Point of Attachment  **MN:** Mobile Node  **MIH PDU:** MIH Packet Data Unit |

|  |  |
| --- | --- |
| DRAFT SECTION | 9.2.2.1. MIH Service Access Authentication |
| Modification type | ADD |
| Line | 26 |
| Text | Before providing any service to the MN, a media independent authentication must be performed in order to authenticate and authorize the credentials provided by the MN. Therefore, in order to achieve this objective, MN credentials and authentication servers (AS) are needed in the architecture. To clarify, Figure 2 depicts a general message flow (MN initiated) and Figure 3 depicts the same message flow but based on network initiated procedure, needed to achieve a Media Independent Authentication based on EAP.  Figure General message flow MN initiated  Figure General message flow network initiated |
| NOTE | **References in provided text may be need to be updated in draft.** |

|  |  |
| --- | --- |
| DRAFT SECTION | 9.2.2.1. MIH Service Access Authentication |
| Modification type | UPDATE |
| Lines | From 30 to 56 |
| Text | ***Capability Discovery Phase***. In this phase, both the MN and the PoS exchange unprotected MIH messages in order to obtain the services which can be selected by the MN. PoS provide a list of the available services and the MN is aware about the different capabilities provided by a certain PoS.    ***Media Independent Service Authentication phase***. Before to start the MN authentication, between the MN and the PoS (MN initiated or Network initiated) a negotiation is performed in order to agree the ciphersuite to be used and others useful parameters*.* Once this negotiation is completed*,* theMN (acting as EAP peer) authenticates against a PoS (which is acting as EAP authenticator) using a MIH Service AS (acting as EAP server). To achieve this, EAP is transported by MIH protocol between the MN and the PoS which manages the MN authentication. In order to carry out the authentication, the PoS may need a backed authentication server (i.e. MIH Service AS such as an AAA server) to verify the MN’s credentials. It is assumed in this document, that the EAP method performed between the EAP peer and the EAP Server is able to export key material. Thus, after performing the authentication, key material (i.e. MSK) will be shared between the MN and the PoS. So, this key material is exported to both MN’s and PoS’ MIH lower-layer and it is available to protect the rest of the MIH signalling exchanged between the PoS and the MN (how to MIH is protected is explained in section **2.5**). In order to protect the MIH signaling, the exported MSK is used as root key to derive new session keys which are use to protect the MIH packets. How this key hierarchy is derived is described in section **2.5**. Note that, the authentication procedure could be based in ERP in order to perform a fast re-authentication procedure; in that case, an rMSK is used as root key to derive the key hierarchy and protect the MIH signaling.  ***Service Access phase***. At this point, the MN is authenticated and authorized to use the MIH services, agreed and provided by the PoS. The Media Independent Service Authentication phase is already completed and the MIH protocol is protected by using the key material obtained in the Media Independent Service Authentication Phase. This phase is related with section **2.4** for key derivation and section **2.5** for protecting MIH protocol.  ***Termination phase***. When the MN and the PoS desire to release resources and the MN’ state related with the provided services. |
| NOTE | **References in provided text may be need to be updated in draft.** |

|  |  |
| --- | --- |
| DRAFT SECTION | 9.2.2.1. MIH Service Access Authentication |
| Modification type | UPDATE REFERENCE  This is the figure that is being referenced. |
| Line | 25 |
| Text |  |

|  |  |
| --- | --- |
| DRAFT SECTION | New Section (CIPHERSUITE) |
| Modification type | NEW |
| Text | |  |  | | --- | --- | | **Confidentiality algorithms** | **Reference** | | ENCR\_AES\_CBC | NIST 800-38A | | ENCR\_NULL |  | | **Integrity algorithms** | **Reference** | | INTR\_HMAC\_SHA\_96 | FIPS 180-1 | | INTR\_CMAC\_AES | NIST 800-38B | | **Confidentiality and Integrity algorithms** | **Reference** | | AUTH\_ENC\_AES\_CCM | NIST 800-38C | | **KDF algorithms** | **Reference** | | PRF\_CMAC\_AES | NIST 800-108 | | PRF\_HMAC\_SHA1 | NIST 800-108 |   The ciphersuites are coded as follows:   |  |  |  |  | | --- | --- | --- | --- | | **Code** | **Integrity Algorithm** | **Encryption Algorithm** | **Length (bits)** | | 00000000 | NULL | NULL | 0 | | 00000001 | AES\_CBC | NULL | 128 | | 00000010 | AES\_CBC | HMAC-SHA1-96 | 256 | | 00000011 | AES\_CBC | CMAC-AES | 256 | | 00000100 | NULL | HMAC\_SHA1-96 | 128 | | 00000101 | NULL | CMAC\_AES | 128 | | 00000110 | AES\_CCM | | 128 |   A default cipher suite is needed in order to avoid mismatch between MN and PoS in terms of cipher suite.  AES\_CCM needs counter generation function and a formatting function. Moreover, a NONCE generation is needed.These functions are defined in Appendix A of [NIST-SP800-38C] with the following parameters:   * The nonce length n is 12 * The tag length t is 16 * The value of q is 3 |
| NOTE | **This section will require some updates** |

|  |  |
| --- | --- |
| DRAFT SECTION | 6.5.4 Information elements |
| Modification type | ADD |
| Text | |  |  |  | | --- | --- | --- | | **Name of information element** | **Description** | **Data type** | | IE\_POS\_KEY\_DIST\_INFO | Information about the key distribution mechanisms supported by the PoS | KEY\_DIST\_LIST | | IE\_POS\_INTR\_ALG\_INFO | Information about the integrity algorithms supported by the PoS. | INT\_ALG\_LIST | | IE\_POS\_ENCR\_ALG\_INFO | Information about the encryption algorithms supported by the PoS. | CIPH\_ALG\_LIST | | IE\_POS\_KDF\_INFO | Information about the key derivation functions supported by the PoS. | KDF\_LIST | |
| Note | **The following information elements should be specified within the *IE\_CONTAINER\_NETWORK* within a new subcategory named *PoS-specific information elements*.** |

|  |  |
| --- | --- |
| DRAFT SECTION | 9.2.2.2 Key derivation and key hierarchy |
| Modification type | REPLACE  This section should be replaced with the text provided |
| Text | To protect the MIH messages during Service Access Phase, a key hierarchy needs to be derived from the key material (i.e. MSK) exported during EAP/ERP authentication carried out during the Media Independent Service Authentication phase. The proposed hierarchy is showed in the Figure 4.  Figure Key hierarchy proposed  Using the MSK or rMSK provided by the EAP authentication, as a root key, two keys are derived: MIIK (Media Independent Integrity Key), this key is used to provide integrity protection to the MIH protocol and MIEK (Media Independent Encryption Key), used to provide confidentiality to the MIH protocol. This key hierarchy has been based on the key hierarchy described in [ProAuthMihSec].  Note: it is assumed that the EAP method performed is available to export key material (MSK or rMSK).  How to generate the key hierarchy is described as follows:  Parameters:   * MSK or rMSK – master session key established through an EAP or EAP re-authentication. When the MSK or rMSK is 128 bits long, the PRF used for key derivation can be HMAC-SHA1, HMAC-SHA-256, or AES-CMAC. But if MSK is longer than 128 bits, then HMAC-SHA-1 or HMAC-SHA-256 shall be used as a PRF in key derivation. * L – the length of keying material. L = |MIIK| + |MIEK|, that is, the sum of the binary length of MIIK and MIEK. * h – The output length of the PRF used in key derivation.   + For HMAC-SHA-1, h = 160   + For HMAC-SHA-256, h = 256   + For CMAC-AES, h = 128 * n = the number of iterations of PRF in order to generate L bit keying material     The key derivation for MIH session keys can be described by the following procedure:  **Fixed values**:   * + 1. *h* - The length of the output of the PRF in bits, and     2. *t* - The length of the binary representation of the counter *I* and L. A default value of *t* is 32.   **Input**: *K* = *MSK or rMSK*, *Nonce-P*, *Nonce-S*, and *L*.  **Process:**   1. *n*:= ⎡*L*/*h*⎤. 2. If *n >* 2*t*-1*,* then indicate an error and stop. 3. *result(0)*:= *∅* 4. For *i* = 1 to *n*, do    1. *K*(*i*):= PRF (*K, “MISK”* || *[i]2* || *Nonce-P* || *Nonce-S* || *ciphersuite code* || *[L]2*)    2. *result (i) := result(i-1) || K(i)*   7. Return: *MISK*, i.e., the leftmost *L* bits of *result(n)*.  **Output**: *MISK*.  The MISK is parsed in such a way that  MISK = **MIIK** || **MIEK**. |
| Note | **This section will require updates.**  **References in provided text may be need to be updated in draft.** |

|  |  |
| --- | --- |
| DRAFT SECTION | New Section under section 9.2.2 called MIH Packet protection, OR  9.2.3 MIH Security Asssociation |
| Modification type | NEW or ADD (section 9.2.3 MIH Security Association) |
| Text | In this section we define how the MIH protocol messages are protected by using MIEK and MIIK. This is carried out in the Authenticated/Authorized Service Access Phase. Once Media Independent Service Authentication phase finishes with success, MIH lower-layer can access to key material, which is used to protect MIH messages. Specifically, using the MSK or rMSK as root key and using the key hierarchy defined in the previous section, the MIH protocol is protected by encrypting the MIH message using the MIEK and providing integrity protection uses MIIK. Whereas, encryption is optional, integrity protection is mandatory when MISK is available. Moreover, there are some algorithms which provide confidentiality and integrity (i.e. AUTH\_ENC\_AES\_CCM); therefore, MIEK will be used as the key for performing these algorithms.  Four situations related with the protection of the MIH messages are possible in this document: Confidentiality and Integrity protection by using an encryption and an integrity algorithm, respectively; Confidentiality and Integrity protection by using a single algorithm which provides confidentiality and integrity; only integrity protection and no protection. Figure 5 summarizes these cases.  Figure MIH packet protection procedure  MIH protocol is protected by means of the MIHF layer. A new MIHS (MIH Security) Header is needed to indicate that the payload after MIHS is a whole protected MIH message. This new header contains a new bit S used to indicate if the message is confidentiality protected or not, definition of this header is provided in [OPT2]. To merge both our proposal and [OPT2] we propose to define a *Security TLV* to transport the protected MIH PDU instead of the current defined TLS TLV. |
| Note | **References in provided text may be need to be updated in draft.** |

|  |  |
| --- | --- |
| DRAFT SECTION | NEW section (State termination) |
| Modification type | NEW |
| Text | A termination phase has been defined to provide a mechanism to allow the MN to release the resources kept by MN and PoS (such as keys, authorization lifetime, etc...) associated to the authentication.  We provide two alternatives to proceed: define a termination message or/and session lifetime. In the following we explain the meaning of each alternative.   * *Termination message*: Allows the MN to finalize, explicitly, the current state. This could be useful when the MN changes to another PoS. Allows release resources from the serving PoS which the MN leaves it. This option is currently supported by MIH\_Termination\_Auth primitive and MIH\_Termination\_Auth MIH protocol message. * *State timeout*: Provides a mechanism to finalize the established state after a certain period of time. This state timeout must be equal or lower than the EAP authentication timeout. Using this option it is guarantee that the state is always closed. This option is currently supported by the lifetime tlv included in MIH\_AUTH request and MIH\_AUTH response messages which will define in section 2.7.3.5. |
| Note | **References in provided text may be need to be updated in draft.** |

|  |  |
| --- | --- |
| DRAFT SECTION | 7.4.1.1.2 and 7.4.1.2.2 and 7.4.1.3.2 and 7.4.1.4.2 |
| Modification type | UPDATE |
| Text | CipherRequiredFlag and IndetityOpt must me removed. |

|  |  |
| --- | --- |
| DRAFT SECTION | 7.4.29.1.2 |
| Modification type | UPDATE |
| Text | **Nonce** must be removed.  **AUTH** must be removed |

|  |  |
| --- | --- |
| DRAFT SECTION | 7.4.29.1.2 |
| Modification type | ADD |
| Text | |  |  |  | | --- | --- | --- | | Name | Data type | Description | | Status | STATUS | Status of the authentication |   Add this TLV. |

|  |  |
| --- | --- |
| DRAFT SECTION | 7.4.29.1.3 |
| Modification type | UPDATE |
| Text | This primitive is generated after receiving an MIH\_Start\_Auth indication or an MIH\_Auth.confirm. Also in network initiated authentication this primitive could be generated by a trigger (e.g. radio link variations, resource management reasons, etc). |

|  |  |
| --- | --- |
| DRAFT SECTION | 7.4.29.2.2 |
| Modification type | UPDATE  Modify AuthenticationInformation description |
| Line | 58 |
| Text | This transport the EAP or ERP packet. |

|  |  |
| --- | --- |
| DRAFT SECTION | 7.4.29.3.2 |
| Modification type | UPDATE |
| Text | **Nonce** must be removed.  **AUTH** must be removed |

|  |  |
| --- | --- |
| DRAFT SECTION | 7.4.29.3.2 |
| Modification type | UPDATE  Modify AuthenticationInformation description |
| Line | 38 |
| Text | This transport the EAP or ERP packet. |

|  |  |
| --- | --- |
| DRAFT SECTION | 7.4.29.3.3 |
| Modification type | REMOVE |
| Line | From 51 to 52 |
| Text | *“This primitive provides the required information to authenticate an MN”* |

|  |  |
| --- | --- |
| DRAFT SECTION | NEW section (MIH\_Auth.confirm) |
| Modification type | ADD |
| Line | 59 after section 7.4.29.3.4 |
| Text | Function This primitive is used by an MIHF to indicate to an MIH User that a MIH\_Auth response was received. Semantics of service primitive MIH\_Auth.confirm (  SourceIdentifier,  AuthenticationInformation  )   |  |  |  | | --- | --- | --- | | Name | Data type | Description | | SourceIdentifier | MIHF\_ID | This identifies the invoker of this primitive. | | AuthenticationInformation | AUTH\_INFO\_VALUE | This transport the EAP or ERP packet. |  When generated This primitive is generated after receiving a MIH\_Auth response message. Effect on receipt A MIH User receiving this primitive must generate a MIH\_Auth.request primitive. |

|  |  |
| --- | --- |
| DRAFT SECTION | 7.4.30 and subsections |
| Modification type | UPDATE  In all subsections MIH\_Finish\_Auth |
| Text | MIH\_Termination\_Auth |

|  |  |
| --- | --- |
| DRAFT SECTION | 7.4.30.1.2 and 7.4.30.2.2 |
| Modification type | REMOVE |
| Text | IntegrityAuth |

|  |  |
| --- | --- |
| DRAFT SECTION | NEW section (MIH\_Termination\_Auth.confirm) |
| Modification type | ADD |
| Line | 34 after section 7.4.30.3.4 |
| Text | Function This primitive is used to notify to the corresponding MIH User about the termination of the established state. Semantics of service primitive MIH\_Termination\_Auth.confirm (  SourceIdentifier  )   |  |  |  | | --- | --- | --- | | Name | Data type | Description | | SourceIdentifier | MIHF\_ID | This identifies the invoker, which is a remote MIHF. |  When generated This primitive is generated by a MIHF to terminate the current state after receiving a MIH\_Termination\_Auth response message. Effect on receipt The established state terminates. |

|  |  |
| --- | --- |
| DRAFT SECTION | 8.6.1.1 and 8.6.1.2 |
| Modification type | REMOVE |
| Text | CipherRequiredFlag and IdentityOpt |

|  |  |
| --- | --- |
| DRAFT SECTION | 8.6.1.12 |
| Modification type | REPLACE  This section should be replaced with the text provided |
| Text | |  | | --- | | **MIH Header Fields (SID = 1, Opcode = 1, AID =7)** | | Destination Identifier = receiving MIHF ID  (Destination MIHF ID TLV) | | Source Identifier = sending MIHF ID  (Source MIHF ID TLV) | | Nonce (optional)  (Nonce TLV) | | AuthenticationInformation (optional)  (Authentication TLV) | | Lifetime (optional)  (Lifetime TLV) | | Status (optional)  (STATUS TLV) | | KeyDistributionMechanismList (optional)  (Key Distribution Mechanism List TLV) | | IntegrityAlgorithmList (optional)  (Integrity Protection List TLV) | | CipherAlgorithmList (optional)  (Encryption Algorithm List TLV) | | KDF-List (optional)  (KDF List TLV) | | IdentityOpt (optional)  (Identity TLV) | | AUTH (optional)  (AUTH TLV) | |

|  |  |
| --- | --- |
| DRAFT SECTION | 8.6.1.13 |
| Modification type | REPLACE  This section should be replaced with the text provided |
| Text | |  | | --- | | **MIH Header Fields (SID = 1, Opcode =2, AID =7)** | | Destination Identifier = receiving MIHF ID  (Destination MIHF ID TLV) | | Source Identifier = sending MIHF ID  (Source MIHF ID TLV) | | Nonce (optional)  (Nonce TLV) | | AuthenticationInformation (optional)  (Authentication TLV) | | Lifetime (optional)  (Lifetime TLV) | | Status (optional)  (STATUS TLV) | | KeyDistributionMechanismList (optional)  (Key Distribution Mechanism List TLV) | | IntegrityAlgorithmList (optional)  (Integrity Protection List TLV) | | CipherAlgorithmList (optional)  (Encryption Algorithm List TLV) | | KDF-List (optional)  (KDF List TLV) | | IdentityOpt (optional)  (Identity TLV) | | AUTH (optional)  (AUTH TLV) | |

|  |  |
| --- | --- |
| DRAFT SECTION | 8.6.1.14 and 8.6.1.15 |
| Modification type | UPDATE |
| Text | MIH\_Termination\_Auth |

|  |  |
| --- | --- |
| DRAFT SECTION | 8.6.1.14 and 8.6.1.15 |
| Modification type | UPDATE  IntegrityAuth row |
| Text | AUTH (optional)  (AUTH TLV) |

|  |  |
| --- | --- |
| DRAFT SECTION | NEW section (TLVs included by MIHF in MIH Messages) |
| Modification type | NEW |
| Text | |  |  |  | | --- | --- | --- | | Name | Data type | Description | | Lifetime | LIFETIME | This TLV represent the period of time that a key is valid for being used. | | AUTH | AUTH\_VALUE | Integrity data to verify that a message has not been modified. | |

|  |  |
| --- | --- |
| DRAFT SECTION | 10.2 Bundle media access authentication to MIH service access authentication |
| Modification type | Replace |
| Lines | From 41 to 51 |
| Text | Regarding the key hierarchy proposed for Media Independent Service Authentication phase in section 2.4, for the bundle option with WI#1 option B we propose to extend the existing hierarchy to derive different Media Specific Pairwise Master Keys (MS-PMKs). Using the MSK/rMSK exported from the EAP/ERP authentication, a new key called MS-ROOT is derived. A set of MS-PMKs can be derived from the MS-ROOT key, acting as root key. An MS-PMK can be used as shared secret between the PoS and the MN to carry out a media-specific EAP authentication in the reactive pull key distribution mechanisms or optimized proactive pull key distribution. Alternatively, they can be used directly by the target PoA and the MN to perform a security association protocol which allows to protect the data link.  Figure 14 Key hierarchy proposed  Note: If a Key Distribution Mechanism is not selected, MS-ROOT and MS-PMK do not need to be derived.  The MS-ROOT key can be derived by the following procedure.  **Input**: *K* (= *MSK or rMSK*), *Nonce-P*, *Nonce-S*, and *L*.  **Process:**   1. *MS-RK* := PRF (*K, “MSROOTKEY”* || *Nonce-P* || *Nonce-S* ) 2. *Return: MS-RK.*   **Output**: *MS-RK*.  MS-RK can be 128 bits, 160 bits or 256 bits depending on the PRF used.  Multiple MS-PMK can be derived from MS-ROOT key (MS-RK) for different PoAs. Each PoA is defined by its link-layer address.  It is assumed that the same PRF is used for both MIH session keys and for MS root key. When the MSK or rMSK is 128 bits long, the PRF used for key derivation can be HMAC-SHA1, HMAC-SHA-256, or AES-CMAC. But if MSK is longer than 128 bits, then HMAC-SHA-1 or HMAC-SHA-256 shall be used as a PRF in key derivation.  The MS-PMK can be derived by the following procedure.  **Input**: *MS-RK*, *MN\_LINK\_ID*, *POA\_LINK\_ID*.  **Process:**   1. *MS-PMK* := PRF (*MS-RK, “MS-PMK”* || *MN\_LINK\_ID* || *POS\_LINK\_ID* ) 2. *Return: MS-PMK.*   **Output**: *MS-PMK*.  MS-PMK can be 128 bits, 160 bits or 256 bits depending on the PRF used in the above. The MS-PMK will be distributed to the PoA identified by *POA\_LINK\_ID*. It will be used by the PoA to derive media specific session keys. The key derivation for media specific session keys is out of the scope of this standard. |
| Note | **This section will require updates.**  **References in provided text may be need to be updated in draft.**  **May it is better top ut this text under a new subsection under section 10.2** |

|  |  |
| --- | --- |
| DRAFT SECTION | 10.2.2 Reactive pull key distribution |
| Modification type | ADD  If you consider that it is needed |
| Text | It is performed after the MN moves to the target PoA. It assumes that both the MN and the PoS shares a symmetric key. In particular a specific derived MS-PMK for this purpose. The reactive key distribution implies the execution of a media-specific EAP/ERP authentication between the MN and the target PoA. In this media-specific EAP/ERP authentication, the EAP/ERP peer is the MN, the EAP/ERP authenticator is the target PoA and the PoS acts as AAA/EAP server. The EAP authentication is performed by using the MS-PMK shared between the MN and the PoS. Since an EAP method based on symmetric keys or ERP must be used for this reactive media-specific EAP authentication, which does not involve any MIH signaling. In order to optimize the mechanism a new identity, provided during the negotiation carried out in the media independent service authentication phase (see section 2.1), is used to contact with the PoS (acting as AAA/EAP). |

|  |  |
| --- | --- |
| DRAFT SECTION | 10.2.3 Proactive pull key distribution |
| Modification type | Replace |
| Text | This mechanism allows the MN to perform a proactively media-specific authentication with the target PoA without being directly connected to the wireless link of the target PoA by means sending link-layer frames through the PoS to the target PoA. |

|  |  |
| --- | --- |
| DRAFT SECTION | NEW section (Optimized proactive pull key distribution |
| Modification type | NEW after 10.2.3 |
| Text | This mechanism function is the same as Proactive pull key distribution, but it provides an optimization. In this optimization, the key hierarchy shared between the MN and the PoS could be used in order to derive a shared key to be used in the key distribution process, where the PoS will be acting as a local AAA and using the identity provided during the Media Independent Service Authentication phase the PoS could be contacted to perform the key distribution mechanism. |

|  |  |
| --- | --- |
| DRAFT SECTION | 7.4.31.1.2 and 7.4.31.3.2 and 7.4.32.1.2 and 7.4.32.3.2 |
| Modification type | REMOVE |
| Text | IntegrityAuth |

|  |  |
| --- | --- |
| DRAFT SECTION | 7.4.31.1.2 and 7.4.31.3.2 and 7.4.32.1.2 and 7.4.32.3.2 |
| Modification type | UPDATE  PoAIndentifier description |
| Text | This identifies the link address of the PoA. |

|  |  |
| --- | --- |
| DRAFT SECTION | 7.4.31.2.2 |
| Modification type | REPLACE |
| Text | MIH\_Push\_Key.indication (  MSPMK,  PoAIdentifier,  LinkId  )   |  |  |  | | --- | --- | --- | | Name | Data type | Description | | MSPMK | KEY | It is the MS-PMK to be pushed. | | PoAIdentifier | PoA\_ID | This identifies the link address of the PoA. | | LinkId | Link\_Addr | This contains the MAC address of the MN’s active interface. | | lifetime | LIFETIME | Indicates the period of time that a key tansported in MSPMK is valid for being used. | |

|  |  |
| --- | --- |
| DRAFT SECTION | 7.4.31.3.2 |
| Modification type | REPLACE |
| Text | MIH\_Push\_Key.response (  DestinationIdentifier,  PoAIndentifier,  Status  )   |  |  |  | | --- | --- | --- | | Name | Data type | Description | | DestinationIdentifier | MIHF\_ID | This identifies a remote MIHF that will be the destination of this request. | | PoAIdentifier | PoA\_ID | This identifies a PoA. | | Status | STATUS | Represent the operation result. | |

|  |  |
| --- | --- |
| DRAFT SECTION | NEW section (MIH\_Push\_Key.confirm) |
| Modification type | NEW  After section 7.4.31.3.4 |
| Text | Function This primitive is used to notify the MIH user (in MN side) about the status of the requested operation. Semantics of service primitive MIH\_Push\_Key.confirm (  PoAIdentifier,  Status  )   |  |  |  |  | | --- | --- | --- | --- | | Name | | Data type | Description | | PoAIdentifier | PoA\_ID | | This identifies a PoA. | | Status | STATUS | | Represent the operation state. |  When generated This primitive is generated after receiving a MIH\_Push\_Key message. Effect on receipt A media specific key must be installed in the MAC layer. |

|  |  |
| --- | --- |
| DRAFT SECTION | 7.4.32.1.1 |
| Modification type | REPLACE |
| Text | This primitive is used by the MN to request a key installation in a target PoA using link-layer frames to be authenticated. The authentication is performed proactively. |

|  |  |
| --- | --- |
| DRAFT SECTION | 7.4.32.1.2 and 7.4.32.2.2 and 7.4.32.3.2 |
| Modification type | REPLACE  ProactivePullInformation |
| Text | **Column DATA TYPE**: PROACT\_PULL\_LL\_FRAMES  **Column DESCRIPTION**: This contains link-layer frames both the MN and the PoA in other to perform a proactive authentication. |

|  |  |
| --- | --- |
| DRAFT SECTION | 7.4.32.3.2 |
| Modification type | REPLACE |
| Text | MIH\_Proact\_Pull\_key.response (  DestinationIdentifier,  PoAIndentifier,  ProactivePullInformation,  Status  )   |  |  |  | | --- | --- | --- | | Name | Data type | Description | | DestinationIdentifier | MIHF\_ID | This identifies a remote MIHF that will be the destination of this request. | | PoAIdentifier | PoA\_ID | This identifies a PoA. | | ProactivePullInformation | PROACT\_PULL\_LL\_FRAMES | This contains link-layer frames. | | Status | STATUS | Status of the authentication | |

|  |  |
| --- | --- |
| DRAFT SECTION | NEW section (MIH\_Proact\_Pull\_key.confirm) |
| Modification type | NEW  After section 7.4.32.3.4 |
| Text | Function This primitive is used to notify the corresponding MIH user about the reception of a MIH\_Proact\_Pull response. Semantics of service primitive MIH\_Proact\_Pull\_key.confirm (  ProactivePullInformation,  Status  )   |  |  |  | | --- | --- | --- | | Name | Data type | Description | | ProactivePullInformation | PROACT\_PULL\_LL\_FRAMES | This contains link-layer frames both the MN and the PoA in other to perform a proactive authentication. | | Status | STATUS | Status of the authentication |    When generated This primitive is generated by the MIHF in PoS after receiving a MIH\_Proact\_Pull\_key response message. Effect on receipt The MIH user must generate a MIH\_Proact\_Pull\_key.request primitive until the key distribution is completed. |

|  |  |
| --- | --- |
| DRAFT SECTION | 8.6.1.16 |
| Modification type | REPLACE |
| Text | |  | | --- | | **MIH Header Fields (SID = 1, Opcode = 1, AID = 9)** | | Destination Identifier = receiving MIHF ID  (Destination MIHF ID TLV) | | Source Identifier = sending MIHF ID  (Source MIHF ID TLV) | | PoAIdentifier  (PoA Identifier TLV) | | LinkId  (Link ID TLV) | | AUTH (optional)  (AUTH TLV) | |

|  |  |
| --- | --- |
| DRAFT SECTION | 8.6.1.17 |
| Modification type | REPLACE |
| Text | |  | | --- | | **MIH Header Fields (SID = 1, Opcode = 2, AID = 9)** | | Destination Identifier = receiving MIHF ID  (Destination MIHF ID TLV) | | Source Identifier = sending MIHF ID  (Source MIHF ID TLV) | | PoAIdentifier  (PoA Identifier TLV) | | Status  (Status TLV) | | AUTH (optional)  (AUTH TLV) | |

|  |  |
| --- | --- |
| DRAFT SECTION | 8.6.1.18 |
| Modification type | REPLACE |
| Text | |  | | --- | | **MIH Header Fields (SID = 1, Opcode = 1, AID = 10)** | | Destination Identifier = receiving MIHF ID  (Destination MIHF ID TLV) | | Source Identifier = sending MIHF ID  (Source MIHF ID TLV) | | PoAIdentifier  (PoA Identifier TLV) | | ProactivePullInformation  (Pull Information TLV) | | AUTH (optional)  (AUTH TLV) | |

|  |  |
| --- | --- |
| DRAFT SECTION | 8.6.1.19 |
| Modification type | REPLACE |
| Text | |  | | --- | | **MIH Header Fields (SID = 1, Opcode = 2, AID = 10)** | | Destination Identifier = receiving MIHF ID  (Destination MIHF ID TLV) | | Source Identifier = sending MIHF ID  (Source MIHF ID TLV) | | PoAIdentifier  (PoA Identifier TLV) | | ProactivePullInformation  (Pull Information TLV) | | Status (optional)  (Status TLV) | | AUTH (optional)  (AUTH TLV) | |

|  |  |
| --- | --- |
| DRAFT SECTION | ANNEX N |
| Modification type | ADD  If it is required or may be it could be in order section |
| Text | Capability Discovery phase messages (PoS initiated) |

|  |  |
| --- | --- |
| DRAFT SECTION | ANNEX N |
| Modification type | ADD  If it is required or may be it could be in order section |
| Text | Capability Discovery phase messages (MN initiated) |

|  |  |
| --- | --- |
| DRAFT SECTION | ANNEX N |
| Modification type | ADD  If it is required or may be it could be in order section |
| Text | Media Independent Service Authentication MN initiated messages  Some aspects which need further consideration:  Negotiation phase to negotiate the cipher suite to be used is performed in the firsts two MIH\_AUTH messages (request/response).  The authentication process is based in EAP.  KeyDistMechList identifies the Key Distribution Mechanism List TLV. This is used to select a key distribution mechanism (Push, Reactive Pull, Proactive Pull, or Optimized Proactive Pull). If this TLV is present, it is signaling that the bundle option is active and the key distribution mechanism selected by this TLV is going to be used to perform the bundle.  Flag P is used to indicate that the MN is trying to authenticate with a PoS in a different domain. This flag is in MIH Header on bit 0 in RESERVED2 field.  Status is used to indicate the state of the authentication, if it has been successful or not.  The last two messages have an AUTH TLV and they include the ciphersuite and the other related parameters negotiated before, in order to securely confirm that these parameters (cipher suite, key distribution mechanism, etc) have been those exchanged during the first MIH\_Auth exchange. This AUTH TLV is checked both the MN and the PoS. |

|  |  |
| --- | --- |
| DRAFT SECTION | ANNEX N |
| Modification type | ADD  If it is required or may be it could be in order section |
| Text | Media Independent Service Authentication Network initiated messages |

|  |  |
| --- | --- |
| DRAFT SECTION | ANNEX N |
| Modification type | ADD  If it is required or may be it could be in order section |
| Text | Media Independent Service Authentication MN initiated based on ERP  Some aspects to be considered:  The authentication process is based on ERP.  In the MN initiated scenario, the negotiation phase performed in the firsts two messages is in a different way. In this case, the MN proposes its supported parameters (cipher suite, key distribution mechanism, etc) and the PoS chooses which ones are going to be used, unlike the EAP authentication case where the PoS proposes and the MN chooses.  To check the negotiated parameters, in the last message the PoS sends the proposed parameters by the MN and its own selection. This message is integrity protected. At the MN side, the parameters can be checked due to the MN has its proposal and the selection made by the PoS is integrity protected. |

|  |  |
| --- | --- |
| DRAFT SECTION | ANNEX N |
| Modification type | ADD  If it is required or may be it could be in order section |
| Text | Media Independent Service Authentication Network initiated based on ERP  Some aspects to consider:  The authentication process is based on ERP.  Negotiation phase to negotiate the cipher suite to be used is performed in the firsts two MIH\_AUTH messages (request/response).  KeyDistMechList identifies the Key Distribution Mechanism List TLV. This is used to select a key distribution mechanism (Push, Reactive Pull, Proactive Pull, or Optimized Proactive Pull). If this TLV is present, it is signaling that the bundle option is active and the key distribution mechanism selected by this TLV is going to be used to perform the bundle.  Flag P is used to indicate that the MN is trying to authenticate with a PoS in a different domain. This flag is in MIH Header on bit 0 in RESERVED2 field.  Status is used to indicate the state of the authentication, if it has been successful or not.  The last two messages have an AUTH TLV and they include the ciphersuite and the other related parameters negotiated before, in order to securely confirm that these parameters (cipher suite, key distribution mechanism, etc) have been those exchanged during the first MIH\_Auth exchange. This AUTH TLV is checked both the MN and the PoS.  To initiate the Media independent Authentication by the network a trigger is needed. |

|  |  |
| --- | --- |
| DRAFT SECTION | ANNEX N |
| Modification type | ADD  If it is required or may be it could be in order section |
| Text | Finalization phase messages |

|  |  |
| --- | --- |
| DRAFT SECTION | ANNEX N |
| Modification type | ADD  If it is required or may be it could be in order section |
| Text | Push key distribution messages |

|  |  |
| --- | --- |
| DRAFT SECTION | ANNEX N |
| Modification type | ADD  If it is required or may be it could be in order section |
| Text | Proactive pull key distribution messages |

|  |  |
| --- | --- |
| DRAFT SECTION | ANNEX N |
| Modification type | ADD  If it is required or may be it could be in order section |
| Text | Optimized proactive pull key distribution messages |

|  |  |
| --- | --- |
| DRAFT SECTION | ANNEX N |
| Modification type | ADD  If it is required or may be it could be in order section |
| Text | Reactive pull key distribution messages |
| Note | To perform this key distribution mechanism there is no need of use MIH protocol. It is a media-specific authentication which could be optimized by acting the PoS like a AAA server. |

|  |  |
| --- | --- |
| DRAFT SECTION | NEW section (Data types definition) |
| Modification type | NEW |
| Text | |  |  |  | | --- | --- | --- | | **Data Type Name** | **Derived from** | **Definition** | | KEY\_DIST\_LIST | BITMAP(8) | A list of key distribution methods available.  Bitmap values:  Bit 0: Push Key distribution.  Bit 1: Proactive Pull Key Distribution.  Bit 2: Optimized Proactive Pull Key Distribution.  Bit 3: Reactive Pull Key Distribution.  Bit 4-7: (Reserved) | | INT\_ALG\_LIST | BITMAP(8) | A list of integrity algorithms available.  Bitmap values:  Bit 0: INTR\_HMAC\_SHA\_96  Bit 1: INTR\_CMAC\_AES  Bit 2: INTR\_NULL  Bit 3-7: (Reserved) | | CIPH\_ALG\_LIST | BITMAP(8) | A list of encryption algorithms available.  Bitmap values:  Bit 0: ENCR\_AES\_CBC  Bit 1: AUTH\_ENC\_AES\_CCM  Bit 2 : ENCR\_NULL  Bit 3-7: (Reserved) | | KDF-List | BITMAP(8) | A list of Key Derivation Functions available.  Bitmap values:  Bit 0: PRF\_CMAC\_AES  Bit 1: PRF\_HMAC\_SHA1  Bit 2-7: (Reserved) | | ID\_OPT | OCTET\_STRING | Represents a new identity provided for optimization purposes. | | NONCE\_VALUE | UNSIGNED\_INT(2) | Represents a random value. | | AUTH\_INFO\_VALUE | OCTET\_STRING | Represents the authentication information used to authenticate. | | AUTH\_VALUE | OCTET\_STRING | Represent an integrity string to verify data integrity. | | INTEGRITY\_DATA | OCTET\_STRING | Represents the integrity data of a message. | | KEY | OCTET\_STRING | Represents a key | | PROACT\_PULL\_LL\_FRAMES | OCTET\_STRING | Represents the information needed to carry out a key installation. | | STATUS | ENUMERATED | This is an extension of STATUS TLV defined in IEEE 802.21, it must be updated to support:  5: Authentication Failure | | LIFETIME | UNSIGNED\_INT(2) | Represents the period of time that a key is valid for being used. | | POA\_ID | LINK\_ADDR | A data type to represent an address of any link layer | |