

Draft **ETSI EN 300 175-7** V2.8.5 (2021-12)



**Digital Enhanced Cordless Telecommunications (DECT)  
Common Interface (CI);  
Part 7: Security features**

---

**Reference**REN/DECT-00357

---

**Keywords**authentication, DECT, IMT-2000, mobility, radio,  
security, TDD, TDMA

---

**ETSI**650 Route des Lucioles  
F-06921 Sophia Antipolis Cedex - FRANCE

---

Tel.: +33 4 92 94 42 00 Fax: +33 4 93 65 47 16

Siret N° 348 623 562 00017 - APE 7112B  
Association à but non lucratif enregistrée à la  
Sous-Préfecture de Grasse (06) N° w061004871

---

**Important notice**

---

The present document can be downloaded from:

<http://www.etsi.org/standards-search>

The present document may be made available in electronic versions and/or in print. The content of any electronic and/or print versions of the present document shall not be modified without the prior written authorization of ETSI. In case of any existing or perceived difference in contents between such versions and/or in print, the prevailing version of an ETSI deliverable is the one made publicly available in PDF format at [www.etsi.org/deliver](http://www.etsi.org/deliver).

Users of the present document should be aware that the document may be subject to revision or change of status.

Information on the current status of this and other ETSI documents is available at

<https://portal.etsi.org/TB/ETSIDeliverableStatus.aspx>

If you find errors in the present document, please send your comment to one of the following services:

<https://portal.etsi.org/People/CommitteeSupportStaff.aspx>

---

**Notice of disclaimer & limitation of liability**

---

The information provided in the present deliverable is directed solely to professionals who have the appropriate degree of experience to understand and interpret its content in accordance with generally accepted engineering or other professional standard and applicable regulations.

No recommendation as to products and services or vendors is made or should be implied.

In no event shall ETSI be held liable for loss of profits or any other incidental or consequential damages.

Any software contained in this deliverable is provided "AS IS" with no warranties, express or implied, including but not limited to, the warranties of merchantability, fitness for a particular purpose and non-infringement of intellectual property rights and ETSI shall not be held liable in any event for any damages whatsoever (including, without limitation, damages for loss of profits, business interruption, loss of information, or any other pecuniary loss) arising out of or related to the use of or inability to use the software.

---

**Copyright Notification**

---

No part may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying and microfilm except as authorized by written permission of ETSI.

The content of the PDF version shall not be modified without the written authorization of ETSI.

The copyright and the foregoing restriction extend to reproduction in all media.

© ETSI 2021.  
All rights reserved.

# Contents

|  |    |
|--|----|
| Intellectual Property Rights .....                                   | 10 |
| Foreword.....  | 10 |
| Modal verbs terminology.....   | 11 |
| Introduction .....   | 11 |
| 1 Scope .....  | 15 |
| 2 References .....   | 16 |
| 2.1 Normative references .....                                       | 16 |
| 2.2 Informative references.....                                      | 17 |
| 3 Definition of terms, symbols and abbreviations.....                | 17 |
| 3.1 Terms.....   | 17 |
| 3.2 Symbols.....   | 18 |
| 3.3 Abbreviations .....  | 18 |
| 4 Security architecture.....   | 20 |
| 4.1 Background .....   | 20 |
| 4.2 Security services.....   | 20 |
| 4.2.1 Authentication of a PT .....                                   | 20 |
| 4.2.2 Authentication of an FT .....                                  | 20 |
| 4.2.3 Mutual authentication .....                                    | 21 |
| 4.2.4 Data confidentiality.....                                      | 21 |
| 4.2.5 User authentication .....                                      | 21 |
| 4.3 Security mechanisms .....  | 21 |
| 4.3.0 General.....   | 21 |
| 4.3.1 Authentication of a PT (type 1 procedure).....                 | 21 |
| 4.3.2 Authentication of an FT (type 1 procedure).....                | 23 |
| 4.3.3 Mutual authentication .....                                    | 24 |
| 4.3.4 Data confidentiality.....                                      | 24 |
| 4.3.4.0 General .....  | 24 |
| 4.3.4.1 Derived Cipher Key (DCK) .....                               | 25 |
| 4.3.4.2 Static Cipher Key (SCK).....                                 | 25 |
| 4.3.4.3 Default Cipher Key (DefCK) .....                             | 25 |
| 4.3.5 User authentication .....                                      | 25 |
| 4.3.6 Authentication of a PT (type 2 procedure).....                 | 26 |
| 4.3.7 Authentication of a FT (type 2 procedure).....                 | 28 |
| 4.4 Cryptographic parameters and keys .....                          | 30 |
| 4.4.1 Overview .....   | 30 |
| 4.4.2 Cryptographic parameters.....                                  | 31 |
| 4.4.2.0 Description of parameters .....                              | 31 |
| 4.4.2.1 Provisions related to the generation of random numbers ..... | 33 |
| 4.4.3 Cryptographic keys .....                                       | 33 |
| 4.4.3.0 General .....  | 33 |
| 4.4.3.1 Authentication key K .....                                   | 33 |
| 4.4.3.2 Authentication session keys KS and KS' .....                 | 34 |
| 4.4.3.3 Cipher key CK .....  | 35 |
| 4.5 Security processes .....   | 36 |
| 4.5.1 Overview .....   | 36 |
| 4.5.2 Derivation of authentication key, K.....                       | 36 |
| 4.5.2.0 General .....  | 36 |
| 4.5.2.1 K is derived from UAK.....                                   | 36 |
| 4.5.2.2 K is derived from AC.....                                    | 36 |
| 4.5.2.3 K is derived from UAK and UPI.....                           | 37 |
| 4.5.3 Authentication processes .....                                 | 37 |
| 4.5.3.0 General .....  | 37 |
| 4.5.3.1 Processes for the derivation of KS and KS'.....              | 37 |
| 4.5.3.2 Processes for the derivation of DCK, RES1 and RES2.....      | 37 |

|           |   |    |
|-----------|---|----|
| 4.5.4     | Key stream generation .....   | 38 |
| 4.5.5     | CCM Authenticated Encryption .....                                      | 39 |
| 4.6       | Combinations of security services.....                                  | 39 |
| 4.6.0     | Service combinations and related considerations .....                   | 39 |
| 4.6.1     | Combinations of security algorithms .....                               | 40 |
| 4.6.1.0   | General .....   | 40 |
| 4.6.1.1   | Limitations related to capering algorithms.....                         | 40 |
| 5         | Algorithms for security processes .....                                 | 40 |
| 5.1       | Background .....  | 40 |
| 5.1.0     | General.....  | 40 |
| 5.1.1     | A algorithm .....   | 41 |
| 5.1.1.0   | A algorithm, general.....   | 41 |
| 5.1.1.1   | A algorithm, DSAA based (A-DSAA).....                                   | 41 |
| 5.1.1.2   | A algorithm, DSAA2 based (A-DSAA2).....                                 | 41 |
| 5.1.1.3   | A algorithm, proprietary.....   | 42 |
| 5.2       | Derivation of session authentication key(s).....                        | 42 |
| 5.2.1     | A11 process .....   | 42 |
| 5.2.2     | A21 process .....   | 43 |
| 5.3       | Authentication and cipher key generation processes.....                 | 43 |
| 5.3.1     | A12 process .....   | 43 |
| 5.3.2     | A22 process .....   | 44 |
| 5.4       | CCM algorithm .....   | 45 |
| 6         | Integration of security .....   | 45 |
| 6.1       | Background .....  | 45 |
| 6.2       | Association of keys and identities .....                                | 45 |
| 6.2.1     | Authentication key.....   | 45 |
| 6.2.1.0   | General .....   | 45 |
| 6.2.1.1   | K is derived from UAK.....  | 45 |
| 6.2.1.2   | K derived from AC.....  | 46 |
| 6.2.1.3   | K derived from UAK and UPI .....  | 46 |
| 6.2.2     | Cipher keys .....   | 46 |
| 6.2.3     | Cipher keys for CCM.....  | 47 |
| 6.2.3.0   | General .....   | 47 |
| 6.2.3.1   | Single use of the keys for CCM .....                                    | 47 |
| 6.2.3.2   | Cipher keys for CCM encryption of C/L multicast channels .....          | 48 |
| 6.3       | NWK layer procedures .....  | 48 |
| 6.3.1     | Background.....   | 48 |
| 6.3.2     | Authentication exchanges .....  | 48 |
| 6.3.3     | Authentication procedures .....   | 50 |
| 6.3.3.1   | Authentication of a PT type 1 procedure.....                            | 50 |
| 6.3.3.2   | Authentication of an FT type 1 procedure.....                           | 50 |
| 6.3.3.3   | Authentication of a PT type 2 procedure.....                            | 51 |
| 6.3.3.4   | Authentication of an FT type 2 procedure.....                           | 51 |
| 6.3.4     | Transfer of Cipher Key, CK.....   | 52 |
| 6.3.5     | Re-Keying.....  | 52 |
| 6.3.6     | Encryption with Default Cipher Key .....                                | 52 |
| 6.3.7     | Transfer of Cipher Key CK for CCM .....                                 | 52 |
| 6.3.7.0   | General .....   | 52 |
| 6.3.7.1   | Transfer by Virtual Call setup CC procedure.....                        | 52 |
| 6.3.7.2   | Transfer using MM procedures for CCM re-keying and sequence reset.....  | 53 |
| 6.3.8     | Transfer of Cipher Keys for CCM encryption of multicast channels .....  | 53 |
| 6.3.8.1   | General .....   | 53 |
| 6.3.8.2   | Multicast encryption parameter assignment procedure, FT initiated ..... | 53 |
| 6.3.8.2.0 | General .....   | 53 |
| 6.3.8.2.1 | Transport of the security parameters .....                              | 54 |
| 6.3.8.2.2 | <<INFO TYPE>> coding .....  | 54 |
| 6.3.8.3   | Multicast encryption parameter retrieval procedure, PT initiated .....  | 54 |
| 6.3.8.3.0 | General .....   | 54 |
| 6.3.8.3.1 | Transport of the security parameters .....                              | 55 |
| 6.3.8.3.2 | <<INFO TYPE>> coding.....   | 55 |

|           |  |    |
|-----------|--|----|
| 6.3.8.4   | Error cases.....   | 55 |
| 6.3.8.4.1 | FT initiated parameter assignment procedure - PT reject.....                                 | 55 |
| 6.3.8.4.2 | PT initiated parameter retrieval procedure - FT reject.....                                  | 55 |
| 6.3.8.4.3 | Coding of the {MM-INFO-REJECT} in the error cases.....                                       | 56 |
| 6.3.9     | Transfer of Cipher Keys to Wireless Relay Stations (WRS).....                                | 56 |
| 6.3.9.1   | General.....   | 56 |
| 6.3.9.2   | Security considerations.....   | 56 |
| 6.3.9.3   | Indication of cipher key FT initiated procedure.....   | 56 |
| 6.3.9.4   | Cipher key retrieval procedure. PT initiated.....  | 57 |
| 6.3.9.5   | Error cases.....   | 59 |
| 6.3.9.5.1 | PT initiated cipher key retrieval procedure - FT reject.....                                 | 59 |
| 6.4       | MAC layer procedures.....  | 60 |
| 6.4.1     | Background.....  | 60 |
| 6.4.2     | MAC layer field structure.....   | 60 |
| 6.4.3     | Data to be encrypted.....  | 62 |
| 6.4.4     | Encryption process.....  | 62 |
| 6.4.5     | Initialization and synchronization of the encryption process.....                            | 65 |
| 6.4.5.0   | General.....   | 65 |
| 6.4.5.1   | Construction of CK.....  | 65 |
| 6.4.5.2   | The Initialization Vector (IV).....  | 65 |
| 6.4.5.3   | Generation of two Key Stream segments.....   | 65 |
| 6.4.6     | Encryption mode control.....   | 66 |
| 6.4.6.1   | Background.....  | 66 |
| 6.4.6.2   | MAC layer messages.....  | 66 |
| 6.4.6.3   | Procedures for switching to encrypt mode.....  | 66 |
| 6.4.6.3.1 | General.....   | 66 |
| 6.4.6.3.2 | PT procedure for switching from clear to encrypt mode with a DCK.....                        | 67 |
| 6.4.6.3.3 | FT procedure for switching from clear to encrypt mode with a DCK.....                        | 67 |
| 6.4.6.3.4 | PT procedure for switching from clear to encrypt mode with a Default Cipher Key (DefCK)..... | 68 |
| 6.4.6.3.5 | Error handling - poor link.....  | 70 |
| 6.4.6.4   | Procedures for switching to clear mode.....  | 72 |
| 6.4.6.5   | Procedures for re-keying.....  | 73 |
| 6.4.6.5.1 | Re-keying to a DCK.....  | 73 |
| 6.4.6.5.2 | Re-keying to a DefCK.....  | 74 |
| 6.4.6.5.3 | FT Indication of re-keying to a DefCK.....   | 75 |
| 6.4.6.6   | Insertion of WAIT.....   | 76 |
| 6.4.7     | Handover of the encryption process.....  | 77 |
| 6.4.7.0   | General.....   | 77 |
| 6.4.7.1   | Bearer handover, uninterrupted ciphering.....  | 77 |
| 6.4.7.2   | Connection handover, uninterrupted ciphering.....  | 77 |
| 6.4.7.3   | External handover - handover with ciphering.....   | 78 |
| 6.4.8     | Modifications for half and long slot specifications (2-level modulation).....                | 78 |
| 6.4.8.1   | Background.....  | 78 |
| 6.4.8.2   | MAC layer field structure.....   | 78 |
| 6.4.8.3   | Data to be encrypted.....  | 79 |
| 6.4.8.4   | Encryption process.....  | 79 |
| 6.4.8.5   | Initialization and synchronization of the encryption process.....                            | 80 |
| 6.4.8.6   | Encryption mode control.....   | 80 |
| 6.4.8.7   | Handover of the encryption process.....  | 80 |
| 6.4.9     | Modifications for double slot specifications (2-level modulation).....                       | 80 |
| 6.4.9.1   | Background.....  | 80 |
| 6.4.9.2   | MAC layer field structure.....   | 80 |
| 6.4.9.3   | Data to be encrypted.....  | 81 |
| 6.4.9.4   | Encryption process.....  | 81 |
| 6.4.9.5   | Initialization and synchronization of the encryption process.....                            | 82 |
| 6.4.9.6   | Encryption mode control.....   | 82 |
| 6.4.9.7   | Handover of the encryption process.....  | 82 |
| 6.4.10    | Modifications for multi-bearer specifications.....   | 83 |
| 6.4.11    | Modifications for 4-level, 8-level, 16-level and 64-level modulation formats.....            | 83 |
| 6.4.11.1  | Background.....  | 83 |
| 6.4.11.2  | MAC layer field structure.....   | 83 |
| 6.4.11.3  | Data to be encrypted.....  | 83 |

|            |   |     |
|------------|---|-----|
| 6.4.11.4   | Encryption process .....  | 84  |
| 6.4.11.4.0 | General .....   | 84  |
| 6.4.11.4.1 | Encryption process for the A-field and for the unprotected format .....           | 84  |
| 6.4.11.4.2 | Encryption process for the single subfield protected format .....                 | 85  |
| 6.4.11.4.3 | Encryption process for the multi-subfield protected format .....                  | 86  |
| 6.4.11.4.4 | Encryption process for the constant-size-subfield protected format .....          | 88  |
| 6.4.11.4.5 | Encryption process for the encoded protected format (MAC service $I_{PX}$ ) ..... | 88  |
| 6.4.11.5   | Initialization and synchronization of the encryption process .....                | 90  |
| 6.4.11.6   | Encryption mode control .....   | 90  |
| 6.4.11.7   | Handover of the encryption process .....  | 90  |
| 6.4.12     | Procedures for CCM re-keying and sequence reset .....                             | 90  |
| 6.5        | Security attributes .....   | 90  |
| 6.5.1      | Background .....  | 90  |
| 6.5.2      | Authentication protocols .....  | 91  |
| 6.5.2.0    | General .....   | 91  |
| 6.5.2.1    | Authentication of a PT type 1 procedure .....                                     | 91  |
| 6.5.2.2    | Authentication of an FT type 1 procedure .....                                    | 92  |
| 6.5.2.3    | Authentication of a PT type 2 procedure .....                                     | 93  |
| 6.5.2.4    | Authentication of an FT type 2 procedure .....                                    | 94  |
| 6.5.3      | Confidentiality protocols .....   | 95  |
| 6.5.4      | Access-rights protocols .....   | 97  |
| 6.5.5      | Key numbering and storage .....   | 98  |
| 6.5.5.0    | General .....   | 98  |
| 6.5.5.1    | Authentication keys .....   | 98  |
| 6.5.5.2    | Cipher keys .....   | 98  |
| 6.5.6      | Key allocation .....  | 99  |
| 6.5.6.1    | Introduction .....  | 99  |
| 6.5.6.2    | UAK allocation (DSAA algorithm) .....   | 100 |
| 6.5.6.3    | UAK allocation (DSAA2 algorithm) .....  | 101 |
| 6.6        | DLC layer procedures .....  | 101 |
| 6.6.1      | Background .....  | 101 |
| 6.6.2      | CCM Authenticated Encryption .....  | 102 |
| 6.6.2.0    | CCM overview .....  | 102 |
| 6.6.2.1    | CCM operation .....   | 102 |
| 6.6.2.2    | Key management .....  | 103 |
| 6.6.2.3    | CCM Initialization Vector .....   | 103 |
| 6.6.2.3.0  | CCM Initialization Vector: overview .....   | 103 |
| 6.6.2.3.1  | CCM Initialization Vector: first byte .....                                       | 103 |
| 6.6.2.3.2  | CCM Initialization Vector: bytes 8-11 .....                                       | 104 |
| 6.6.2.3.3  | CCM Initialization Vector: bytes 12 .....   | 104 |
| 6.6.2.4    | CCM Sequence Number .....   | 104 |
| 6.6.2.5    | CCM Start and Stop .....  | 105 |
| 6.6.2.6    | CCM Sequence resetting and re-keying .....  | 105 |
| 6.6.2.7    | CCM encryption for multicast channels .....                                       | 105 |
| 6.6.2.7.0  | General .....   | 105 |
| 6.6.2.7.1  | Applicable types of multicast channels and identifiers .....                      | 105 |
| 6.6.2.7.2  | Process for encryption of multicast channels .....                                | 105 |
| 6.6.2.7.3  | DLC service for encrypted multicast channels .....                                | 105 |
| 6.6.2.7.4  | Encryption key for multicast channels .....                                       | 105 |
| 6.6.2.7.5  | CCM and DLC sequence numbers .....  | 106 |
| 6.6.2.7.6  | Initialization Vector for multicast channels .....                                | 106 |
| 6.6.2.7.7  | Security provisions regarding the key .....                                       | 107 |
| 6.6.2.8    | CCM encryption for service channels .....   | 107 |
| 6.6.2.8.0  | General .....   | 107 |
| 6.6.2.8.1  | Initialization Vector for service channels .....                                  | 108 |
| 6.7        | Security meta-procedures .....  | 108 |
| 6.7.1      | General .....   | 108 |
| 6.7.2      | Re-keying .....   | 108 |
| 6.7.2.1    | Aim and strategy .....  | 108 |
| 6.7.2.2    | Re-keying procedure .....   | 108 |
| 6.7.2.3    | Re-keying procedure with Wireless Relay Stations (WRSs) .....                     | 109 |

|   |  |            |
|---|--|------------|
| 6.7.2.3.1   | General .....  | 109        |
| 6.7.2.3.2   | Key aging model.....   | 110        |
| 6.7.3   | Early encryption.....  | 110        |
| 6.7.3.1   | Aim and strategy .....   | 110        |
| 6.7.3.2   | The Default Cipher Keys (DefCK) .....  | 110        |
| 6.7.3.3   | The Default Cipher Key Index .....   | 111        |
| 6.7.3.4   | Generation and refresh strategy.....   | 111        |
| 6.7.3.5   | Running the procedure .....  | 111        |
| 6.7.3.6   | Security considerations .....  | 111        |
| 7   | Use of security features .....   | 112        |
| 7.1   | Background .....   | 112        |
| 7.2   | Key management options .....   | 112        |
| 7.2.1   | Overview of security parameters relevant for key management.....                         | 112        |
| 7.2.2   | Generation of authentication keys .....  | 113        |
| 7.2.3   | Initial distribution and installation of keys .....                                      | 114        |
| 7.2.4   | Use of keys within the fixed network .....   | 115        |
| 7.2.4.0   | Use of keys within the fixed network: general.....                                       | 115        |
| 7.2.4.1   | Use of keys within the fixed network: diagrams for authentication type 1 scenarios ..... | 117        |
| 7.2.4.2   | Use of keys within the fixed network: diagrams for authentication type 2 scenarios ..... | 120        |
| 7.3   | Confidentiality service with a Cordless Radio Fixed Part (CRFP).....                     | 122        |
| 7.3.1   | General.....   | 122        |
| 7.3.2   | CRFP initialization of PT cipher key.....  | 122        |
| <b>Annex A (informative): Security threats analysis.....</b>                      |  | <b>123</b> |
| A.1   | Introduction .....   | 123        |
| A.2   | Threat A - Impersonating a subscriber identity.....                                      | 124        |
| A.3   | Threat B - Illegal use of a handset (PP).....  | 124        |
| A.4   | Threat C - Illegal use of a base station (FP) .....                                      | 124        |
| A.5   | Threat D - Impersonation of a base station (FP) .....                                    | 125        |
| A.6   | Threat E - Illegally obtaining user data and user related signalling information .....   | 125        |
| A.7   | Conclusions and comments .....   | 126        |
| <b>Annex B (informative): Security features and operating environments .....</b>  |  | <b>128</b> |
| B.1   | Introduction .....   | 128        |
| B.2   | Definitions.....   | 128        |
| B.3   | Enrolment options .....  | 128        |
| <b>Annex C (informative): Reasons for not adopting public key techniques.....</b> |  | <b>130</b> |
| <b>Annex D (informative): Overview of security features .....</b>                 |  | <b>131</b> |
| D.1   | Introduction .....   | 131        |
| D.2   | Authentication of a PT .....   | 131        |
| D.3   | Authentication of an FT .....  | 132        |
| D.4   | Mutual authentication of a PT and an FT.....   | 132        |
| D.4.0   | General .....  | 132        |
| D.4.1   | Direct method.....   | 132        |
| D.4.2   | Indirect method 1.....   | 132        |
| D.4.3   | Indirect method 2.....   | 132        |
| D.5   | Data confidentiality .....   | 132        |
| D.5.0   | General .....  | 132        |
| D.5.1   | Cipher key derivation as part of authentication.....                                     | 133        |
| D.5.2   | Static cipher key .....  | 133        |

|  |   |            |
|--|---|------------|
| D.6  | User authentication.....  | 133        |
| D.7  | Key management in case of roaming .....                               | 133        |
| D.7.1  | Introduction .....  | 133        |
| D.7.2  | Use of actual authentication key K.....                               | 133        |
| D.7.3  | Use of session keys.....  | 134        |
| D.7.4  | Use of precalculated sets .....                                       | 134        |
| <b>Annex E (informative): Limitations of DECT security.....</b>                  |   | <b>135</b> |
| E.1  | Introduction .....  | 135        |
| E.2  | Protocol reflection attacks .....                                     | 135        |
| E.3  | Static cipher key and short Initial Vector (IV) .....                 | 135        |
| E.4  | General considerations regarding key management.....                  | 136        |
| E.5  | Use of a predictable challenge in FT authentication .....             | 136        |
| <b>Annex F (informative): Security features related to target networks .....</b> |   | <b>137</b> |
| F.1  | Introduction .....  | 137        |
| F.1.0  | General .....   | 137        |
| F.1.1  | Notation and DECT reference model .....                               | 137        |
| F.1.2  | Significance of security features and intended usage within DECT..... | 137        |
| F.1.3  | Mechanism/algorithm and process requirements .....                    | 138        |
| F.2  | PSTN reference configurations .....                                   | 139        |
| F.2.1  | Domestic telephone .....  | 139        |
| F.2.2  | PBX.....  | 140        |
| F.2.3  | Local loop.....   | 142        |
| F.3  | ISDN reference configurations.....                                    | 143        |
| F.3.1  | Terminal equipment .....  | 143        |
| F.3.2  | Network termination 2.....  | 144        |
| F.3.3  | Local loop.....   | 144        |
| F.4  | X.25 reference configuration.....                                     | 144        |
| F.4.1  | Data Terminal Equipment (DTE).....                                    | 144        |
| F.4.2  | PAD equipment .....   | 145        |
| F.5  | GSM reference configuration.....                                      | 145        |
| F.5.1  | Base station substation .....   | 145        |
| F.5.2  | Mobile station.....   | 145        |
| F.6  | IEEE 802 reference configuration.....                                 | 145        |
| F.6.1  | Bridge.....   | 145        |
| F.6.2  | Gateway.....  | 145        |
| F.7  | Public access service reference configurations .....                  | 146        |
| F.7.1  | Fixed public access service reference configuration .....             | 146        |
| <b>Annex G (informative): Compatibility of DECT and GSM authentication .....</b> |   | <b>147</b> |
| G.1  | Introduction .....  | 147        |
| G.2  | SIM and DAM functionality .....                                       | 147        |
| G.3  | Using an SIM for DECT authentication.....                             | 148        |
| G.4  | Using a DAM for GSM authentication .....                              | 148        |
| <b>Annex H (normative): DECT Standard Authentication Algorithm (DSAA).....</b>   |   | <b>149</b> |
| <b>Annex I (normative): Security system parameters .....</b>                     |   | <b>150</b> |
| I.1  | Security timers.....  | 150        |



|                               |   |            |
|-------------------------------|---|------------|
| <b>Annex J (normative):</b>   | <b>DECT Standard Cipher (DSC)</b> .....                                 | <b>152</b> |
| <b>Annex K (normative):</b>   | <b>Clarifications, bit mappings and examples for DSAA and DSC</b> ..... | <b>153</b> |
| K.1                           | Ambiguities concerning the DSAA.....                                    | 153        |
| K.2                           | Ambiguities concerning the DSC DECT-standard cipher.....                | 154        |
| <b>Annex L (normative):</b>   | <b>DECT Standard Authentication Algorithm #2 (DSAA2)</b> .....          | <b>156</b> |
| L.1                           | Introduction .....  | 156        |
| L.2                           | Operation of the Authentication Algorithm .....                         | 156        |
| L.2.1                         | DSAA2-1.....  | 156        |
| L.2.2                         | DSAA2-2.....  | 157        |
| L.3                           | Test Sets .....   | 158        |
| L.3.1                         | DSAA2-1.....  | 158        |
| L.3.2                         | DSAA2-2.....  | 161        |
| L.4                           | DSAA2 Examples .....  | 164        |
| L.4.0                         | General .....   | 164        |
| L.4.1                         | Subscription with Key Allocation .....                                  | 164        |
| L.4.1.0                       | Message sequence and coding .....                                       | 164        |
| L.4.1.1                       | PP AC Authentication.....   | 165        |
| L.4.1.2                       | FP AC Authentication.....   | 166        |
| L.4.2                         | DCK Allocation through PP UAK Authentication.....                       | 166        |
| L.4.2.0                       | Message sequence and coding .....                                       | 166        |
| L.4.2.1                       | PP UAK Authentication.....  | 167        |
| L.4.2.2                       | Derivation of 64 bit DCK for DSC .....                                  | 167        |
| L.5                           | DCK to CK mapping.....  | 167        |
| <b>Annex M (normative):</b>   | <b>DECT Standard Cipher #2 (DSC2)</b> .....                             | <b>169</b> |
| M.1                           | Introduction .....  | 169        |
| M.2                           | Operation of the Cipher.....  | 169        |
| M.3                           | Test Sets .....   | 170        |
| M.4                           | DSC2 Test Set .....   | 172        |
| M.5                           | Mapping of DECT values into AES-128 plaintext.....                      | 174        |
| <b>Annex N (normative):</b>   | <b>CCM Authenticated Encryption</b> .....                               | <b>175</b> |
| N.1                           | Introduction .....  | 175        |
| N.1.0                         | General .....   | 175        |
| N.1.1                         | Key management.....   | 175        |
| N.2                           | Operation of the CCM encryption algorithm .....                         | 175        |
| N.2.0                         | Description of the CCM algorithm: general .....                         | 175        |
| N.2.1                         | Description of the CCM algorithm: encryption.....                       | 176        |
| N.2.1.0                       | Overview .....  | 176        |
| N.2.1.1                       | Block ciphers .....   | 176        |
| N.2.1.2                       | Counter function (CTR).....   | 176        |
| N.2.1.3                       | AES block "B" and generation of the encryption stream.....              | 177        |
| N.2.1.4                       | AES block "A" and generation of the Message Integrity Code (MIC) .....  | 177        |
| N.2.1.5                       | "c" stream.....   | 177        |
| N.2.2                         | Description of the CCM algorithm: decoding .....                        | 177        |
| <b>Annex O (informative):</b> | <b>Change history</b> .....   | <b>178</b> |
| History                       | .....   | 179        |

# Intellectual Property Rights

## Essential patents

IPRs essential or potentially essential to normative deliverables may have been declared to ETSI. The declarations pertaining to these essential IPRs, if any, are publicly available for **ETSI members and non-members**, and can be found in ETSI SR 000 314: "*Intellectual Property Rights (IPRs); Essential, or potentially Essential, IPRs notified to ETSI in respect of ETSI standards*", which is available from the ETSI Secretariat. Latest updates are available on the ETSI Web server (<https://ipr.etsi.org/>).

Pursuant to the ETSI Directives including the ETSI IPR Policy, no investigation regarding the essentiality of IPRs, including IPR searches, has been carried out by ETSI. No guarantee can be given as to the existence of other IPRs not referenced in ETSI SR 000 314 (or the updates on the ETSI Web server) which are, or may be, or may become, essential to the present document.

## Trademarks

The present document may include trademarks and/or tradenames which are asserted and/or registered by their owners. ETSI claims no ownership of these except for any which are indicated as being the property of ETSI, and conveys no right to use or reproduce any trademark and/or tradename. Mention of those trademarks in the present document does not constitute an endorsement by ETSI of products, services or organizations associated with those trademarks.

**DECT™**, **PLUGTESTS™**, **UMTS™** and the ETSI logo are trademarks of ETSI registered for the benefit of its Members. **3GPP™** and **LTE™** are trademarks of ETSI registered for the benefit of its Members and of the 3GPP Organizational Partners. **oneM2M™** logo is a trademark of ETSI registered for the benefit of its Members and of the oneM2M Partners. **GSM®** and the GSM logo are trademarks registered and owned by the GSM Association.

# Foreword

This draft European Standard (EN) has been produced by ETSI Technical Committee Digital Enhanced Cordless Telecommunications (DECT), and is now submitted for the combined Public Enquiry and Vote phase of the ETSI standards EN Approval Procedure.

The present document is part 7 of a multi-part deliverable ([1] to [8]). Full details of the entire series can be found in part 1 [1].

The following cryptographic algorithms are subject to controlled distribution:

- a) DECT Standard Authentication Algorithm (DSAA);
- b) DECT Standard Cipher (DSC).

These algorithms are distributed on an individual basis. Further information and details of the current distribution procedures can be obtained from the ETSI Secretariat at the address on the first page of the present document.

Further details of the DECT system may be found in ETSI TR 101 178 [i.1] and ETSI ETR 043 [i.2].

| <b>Proposed national transposition dates</b>   |                                 |
|--|---------------------------------|
| Date of latest announcement of this EN (doa):  | 3 months after ETSI publication |
| Date of latest publication of new National Standard or endorsement of this EN (dop/e): | 6 months after doa              |
| Date of withdrawal of any conflicting National Standard (dow):                         | 18 months after doa             |

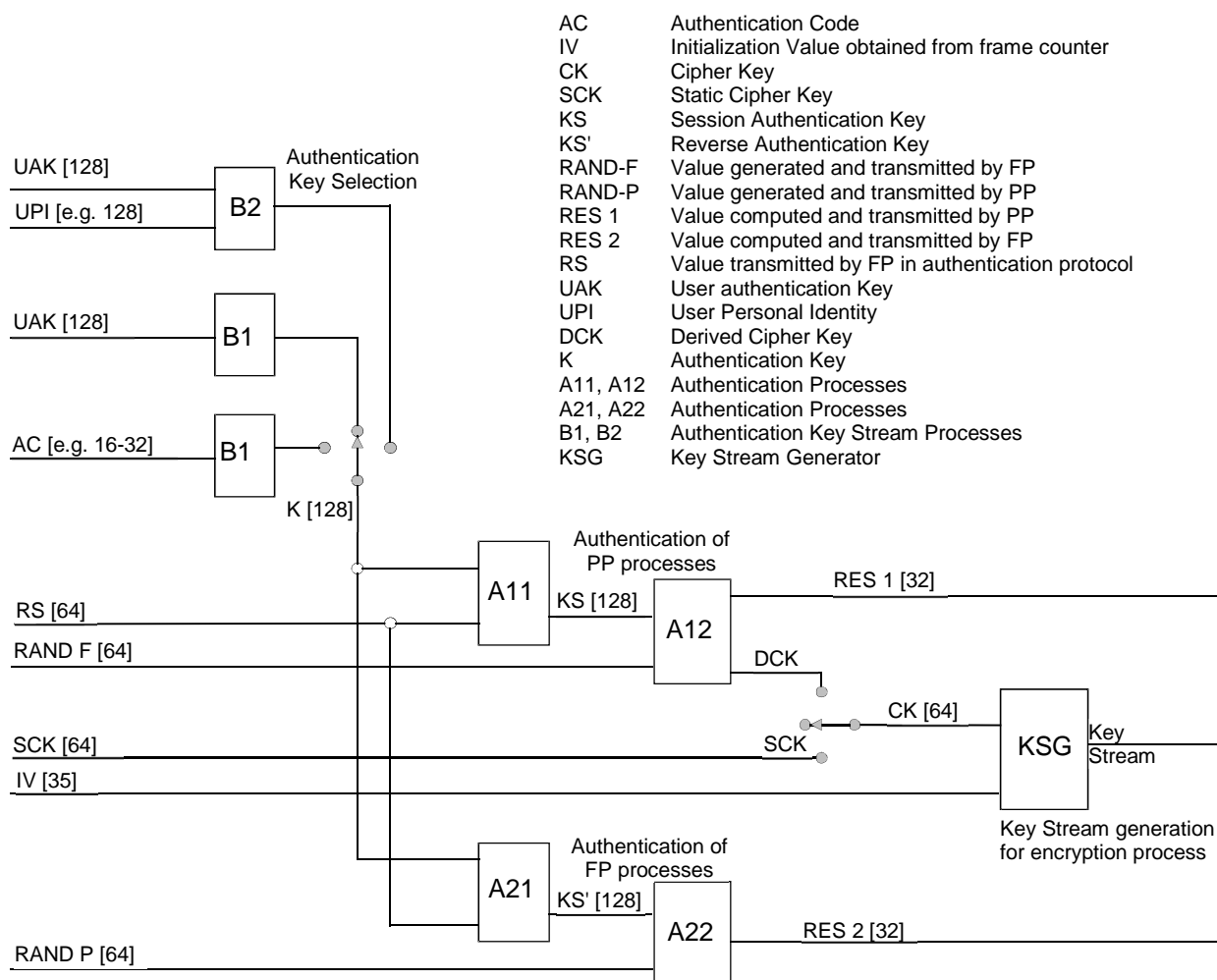
## Modal verbs terminology

In the present document "shall", "shall not", "should", "should not", "may", "need not", "will", "will not", "can" and "cannot" are to be interpreted as described in clause 3.2 of the [ETSI Drafting Rules](#) (Verbal forms for the expression of provisions).

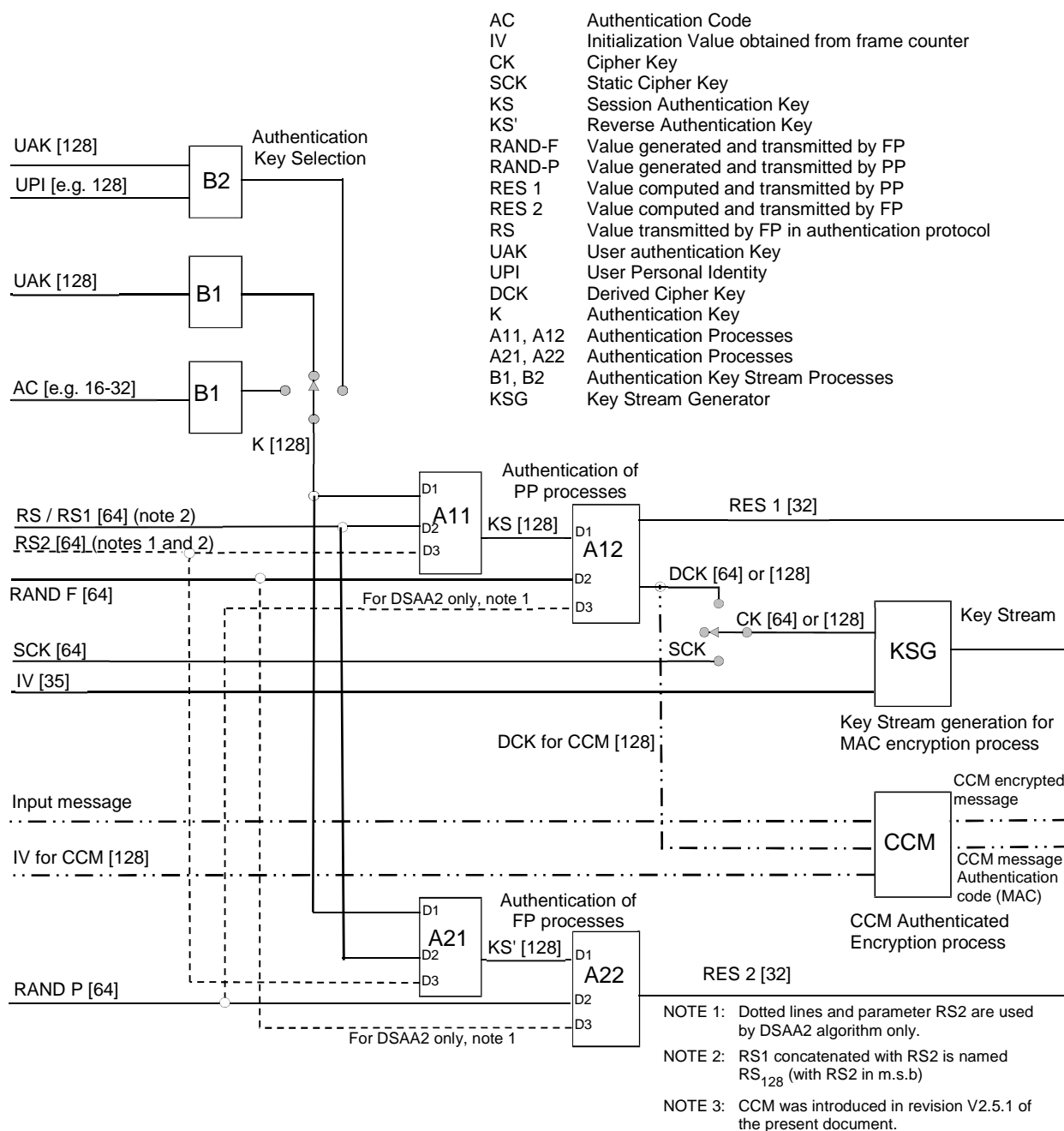
"must" and "must not" are **NOT** allowed in ETSI deliverables except when used in direct citation.

## Introduction

The present document contains a detailed specification of the security features which may be provided by DECT systems. An overview of the processes required to provide all the features detailed in the present document is presented in figures 0.1 and 0.2.



**Figure 0.1: Overview of DECT historic security processes  
(until revision V2.3.1 of the present document)**



**Figure 0.2: Overview of DECT current security processes  
(from revisions V2.4.1 and V2.5.1 of the present document)**

The present document consists of four main clauses (clauses 4 to 7), together with a number of informative/normative and important annexes (A to O). The purpose of this introduction is to briefly preview the contents of each of the main clauses and the supporting annexes.

Each of the main clauses starts with a description of its objectives and a summary of its contents. Clause 4 is concerned with defining a security architecture for DECT. This architecture is defined in terms of the security services which may be offered (see clause 4.2), the mechanisms which are used to provide these services (see clause 4.3), the security parameters and keys required by the mechanisms (challenges, keys, etc.), and which are passed across the air interface or held within DECT Portable Parts (PPs), Fixed Parts (FPs) or other network entities (for example management centres) (see clause 4.4), the processes which are required to provide the security mechanisms (see clause 4.5) and the recommended combinations of services (see clause 4.6).

Clause 5 is concerned with specifying how certain cryptographic algorithms are to be used for the security processes. Three algorithms are required:

- an authentication algorithm;
- a key stream generator for MAC layer encryption; and
- a key stream generator plus a Message Authentication Code generator for CCM authenticated encryption.

The key stream generator is only used for the MAC encryption process, and this process is specified in clause 4.5.4.

The key stream generator plus a Message Authentication Code generator for CCM encryption are used for the CCM authenticated encryption and this process is described in clauses 4.5.5 and 6.6.

For both encryption processes, the authentication algorithm may be used to derive authentication session keys and cipher keys, and is the basis of the authentication process itself. The way in which the authentication algorithm is to be used to derive authentication session keys is specified in clause 5.2. The way in which the algorithm is to be used to provide the authentication process and derive cipher keys is specified in clause 5.3.

Neither the key stream generator nor the authentication algorithm are specified in the clause 5 of the present document. Only their input and output parameters are defined. In principle, the security features may be provided by using appropriate proprietary algorithms. The use of proprietary algorithms may, however, limit roaming in the public access service environment, as well as the use of PPs in different environments.

For example, for performance reasons, the key stream generator for MAC layer encryption will need to be implemented in hardware in PPs and FPs. The use of proprietary generators will then limit the interoperability of systems provided by different manufacturers.

Five standard algorithms have been specified. These are the DECT Standard Authentication Algorithm (DSAA, see annex H), the DECT Standard Authentication Algorithm #2 (DSAA2, see annex L), the DECT Standard Cipher (DSC, see annex J), the DECT Standard Cipher #2 (DSC2, see annex M) and the CCM Authenticated Encryption Algorithm (see annex N).

The DECT Standard Authentication Algorithm #2 (DSAA2, see annex L) and the DECT Standard Cipher #2 (DSC2, see annex M) are based on AES [10] and were introduced with the revision V2.4.1 of the present document.

The CCM Authenticated Encryption Algorithm (CCM, see annex N) is also based on AES [10] and was introduced with the revision V2.5.1 of the present document.

The DECT Standard Authentication Algorithm (DSAA) and the DECT Standard Cipher (DSC) are confidential. Because of their confidential nature, these algorithms are not included in the present document. However, the algorithms will be made available to DECT equipment manufacturers. The DSAA may also need to be made available to public access service operators who, in turn, may need to make it available to manufacturers of authentication modules.

The DECT Standard Authentication Algorithm #2 (DSAA2), the DECT Standard Cipher #2 (DSC2) and the CCM Algorithm (CCM) are publicly available and they are defined in annex L (DSAA2), annex M (DSC2) and annex N (CCM) of the present document.

Clause 6 is concerned with integrating the security features into the DECT system. Four aspects of integration are considered. The first aspect is the association of user security parameters (in particular, authentication keys) with DECT identities. This is the subject of clause 6.2. The second aspect of integration is the definition of the NWK layer protocol elements and message types needed for the exchange of authentication parameters across the air interface. This is dealt with in clause 6.3. The MAC layer procedures for the encryption of data passed over the air interface are the subject of clause 6.4. Finally, clause 6.5 is concerned with security attributes which DECT systems may support, and the NWK layer messages needed to enable PPs and FPs to identify which security algorithms and keys will be used to provide the various security services.

Clause 7 is concerned with key management issues. Careful management of keys is fundamental to the effective operation of a security system, and clause 7.2 is intended to provide guidance on this subject. The clause includes an explanation of how the DECT security features may be supported by different key management options.

For example, schemes which allow authentication keys to be held in a central location within a public access service network are described, as are schemes which allow authentication keys to be derived locally in public access service base stations. The clause is very much less specific than the other clauses in the present document. This is because the key management issues discussed are not an integral part of the CI. In the end it is up to network operators and service providers to decide how they are going to manage their cryptographic keys. The present document can at best provide some suggestions and guidelines.

The main text is supplemented by a set of informative annexes. There are two types of annex. Those of the first type provide background information justifying the inclusion of a particular service, or the use of a particular type of mechanism in the security features. Those of the second type provide guidance on the use and management of certain of the security features. The content of each of the annexes is briefly reviewed below.

Annex A contains the results of a security threats analysis which was undertaken prior to designing the DECT security features.

Annex B is concerned with the impact of the security features on roaming, in particular with the concurrent use of a PP in public access service, wireless Private Branch eXchange (PBX) and residential environments.

Annex C is provided for background information. It contains a justification for some of the decisions taken by EG-1, for example, why symmetric rather than public key (asymmetric) cryptographic mechanisms were selected.

Annex D provides an overview of the DECT security features specified in the present document.

No security system is perfect, and annex E discusses the limitations of the DECT security features.

Annex F relates the security features specified in the present document to the DECT environments identified in ETSI TR 101 178 [i.1]. Each of the local networks identified in the reference model is considered in turn. For each of these networks a security profile is suggested. The networks considered are Public Switched Telephone Network (PSTN), Integrated Services Digital Network (ISDN), Recommendation ITU-T X.25 [i.3], Global System for Mobile communications (GSM), Local Area Networks (LANs) and public access service.

Annex G consists of a brief discussion of the compatibility of DECT and GSM authentication. In particular, the concept of a DECT Authentication Module (DAM) is considered and its functionality compared with the functionality of the GSM Subscriber Interface Module (SIM).

Annex H refers to the DECT Standard Authentication Algorithm.

Annex I refers to Security system parameters.

Annex J refers to the DECT Standard Cipher.

Annex K contains normative clarifications, bit mappings and examples for DSAA and DSC.

Annex L contains the definition of the DECT Standard Authentication Algorithm #2 (DSSA2).

Annex M contains the definition of the DECT Standard Cipher #2 (DSC2) algorithm.

Annex N contains the definition of the CCM Authenticated Encryption (CCM) algorithm.

---

# 1 Scope

The present document is one of the parts of the specification of the Digital Enhanced Cordless Telecommunications (DECT) Common Interface (CI).

The present document specifies the security architecture, the types of cryptographic algorithms required, the way in which they are to be used, and the requirements for integrating the security features provided by the architecture into the DECT CI. It also describes how the features can be managed and how they relate to certain DECT fixed systems and local network configurations.

The security architecture is defined in terms of the security services which are to be supported at the CI, the mechanisms which are to be used to provide the services, and the cryptographic parameters, keys and processes which are associated with these mechanisms.

The security processes specified in the present document are each based on one of three cryptographic algorithms:

- an authentication algorithm;
- a key stream generator for MAC layer encryption; and
- a key stream generator plus a Message Authentication Code generator for CCM authenticated encryption.

The architecture is, however, algorithm independent, and either the DECT standard algorithms, or appropriate proprietary algorithms, or indeed a combination of both can, in principle, be employed. The use of the employed algorithm is specified in the present document.

Integration of the security features is specified in terms of the protocol elements and processes required at the Network (NWK) and Medium Access Control (MAC) layers of the CI.

The relationship between the security features and various network elements is described in terms of where the security processes and management functions may be provided.

The present document does not address implementation issues. For instance, no attempt is made to specify whether the DSAA or DSAA2 should be implemented in the PP at manufacture, or whether the DSAA, DSAA2 or a proprietary authentication algorithm should be implemented in a detachable module. Similarly, the present document does not specify whether the DSC or DSC2 should be implemented in hardware in all PPs at manufacture, or whether special PPs should be manufactured with the DSC, DSC2 or proprietary ciphers built into them. The security architecture supports all these options, although the use of proprietary algorithms may limit roaming and the concurrent use of PPs in different environments.

Within the standard authentication algorithms, DSAA2, DSC2 and CCM are stronger than DSAA and DSC and provide superior protection. DSAA2 and DSC2 are based on AES [10] and were created in 2011. CCM is also based on AES [10] and was added to the standard in 2012.

The present document includes New Generation DECT, a further development of the DECT standard introducing wideband speech, improved data services, new slot types and other technical enhancements.

The present document also includes DECT Ultra Low Energy (ULE), a low rate data technology based on DECT intended for M2M applications with ultra low power consumption.

---

## 2 References

### 2.1 Normative references

References are either specific (identified by date of publication and/or edition number or version number) or non-specific. For specific references, only the cited version applies. For non-specific references, the latest version of the referenced document (including any amendments) applies.

Referenced documents which are not found to be publicly available in the expected location might be found at <https://docbox.etsi.org/Reference>.

NOTE: While any hyperlinks included in this clause were valid at the time of publication ETSI cannot guarantee their long term validity.

The following referenced documents are necessary for the application of the present document.

- [1] ETSI EN 300 175-1: "Digital Enhanced Cordless Telecommunications (DECT); Common Interface (CI); Part 1: Overview".
- [2] ETSI EN 300 175-2: "Digital Enhanced Cordless Telecommunications (DECT); Common Interface (CI); Part 2: Physical Layer (PHL)".
- [3] ETSI EN 300 175-3: "Digital Enhanced Cordless Telecommunications (DECT); Common Interface (CI); Part 3: Medium Access Control (MAC) layer".
- [4] ETSI EN 300 175-4: "Digital Enhanced Cordless Telecommunications (DECT); Common Interface (CI); Part 4: Data Link Control (DLC) layer".
- [5] ETSI EN 300 175-5: "Digital Enhanced Cordless Telecommunications (DECT); Common Interface (CI); Part 5: Network (NWK) layer".
- [6] ETSI EN 300 175-6: "Digital Enhanced Cordless Telecommunications (DECT); Common Interface (CI); Part 6: Identities and addressing".
- [7] Void.
- [8] ETSI EN 300 175-8: "Digital Enhanced Cordless Telecommunications (DECT); Common Interface (CI); Part 8: Speech and audio coding and transmission".
- [9] ETSI TS 100 977: "Digital cellular telecommunications system (Phase 2+); Specification of the Subscriber Identity Module - Mobile Equipment (SIM-ME) Interface (3GPP TS 11.11 Release 1999)".
- [10] FIPS Publication 197 (2001): "Advanced Encryption Standard (AES)", National Institute of Standards and Technology (NIST).
- [11] IETF RFC 3610: "Counter with CBC-MAC (CCM)".
- [12] ETSI EN 300 700: "Digital Enhanced Cordless Telecommunications (DECT); Wireless Relay Station (WRS)".



## 2.2 Informative references

References are either specific (identified by date of publication and/or edition number or version number) or non-specific. For specific references, only the cited version applies. For non-specific references, the latest version of the referenced document (including any amendments) applies.

NOTE: While any hyperlinks included in this clause were valid at the time of publication ETSI cannot guarantee their long term validity.

The following referenced documents are not necessary for the application of the present document but they assist the user with regard to a particular subject area.

- [i.1] ETSI TR 101 178: "Digital Enhanced Cordless Telecommunications (DECT); A High Level Guide to the DECT Standardization".
- [i.2] ETSI ETR 043: "Digital Enhanced Cordless Telecommunications (DECT); Common Interface (CI); Services and facilities requirements specification".
- [i.3] Recommendation ITU-T X.25: "Interface between Data Terminal Equipment (DTE) and Data Circuit-terminating Equipment (DCE) for terminals operating in the packet mode and connected to public data networks by dedicated circuit".
- [i.4] ETSI ETR 056: "Digital Enhanced Cordless Telecommunications (DECT); System description document".
- [i.5] IEEE 802™: "IEEE Standard for Local and Metropolitan Area Networks: Overview and Architecture".
- [i.6] ETSI TS 102 939-1: "Digital Enhanced Cordless Telecommunications (DECT); Ultra Low Energy (ULE); Machine to Machine Communications; Part 1: Home Automation Network (phase 1)".
- [i.7] ETSI TS 102 939-2: "Digital Enhanced Cordless Telecommunications (DECT); Ultra Low Energy (ULE); Machine to Machine Communications; Part 2: Home Automation Network (phase 2)".
- [i.8] ETSI EN 300 444: "Digital Enhanced Cordless Telecommunications (DECT); Generic Access Profile (GAP)".

---

## 3 Definition of terms, symbols and abbreviations

### 3.1 Terms

For the purposes of the present document, the terms given in ETSI EN 300 175-1 [1] and the following apply:

**Cipher Block Chaining Message Authentication Code (CBC-MAC):** cryptographic technique for constructing a message authentication code from a block cipher

**Counter with CBC-MAC (CCM):** authenticated encryption algorithm designed to provide both authentication and confidentiality

**IV:** Initialization Vector (used in ciphering processes)

**K:** authentication Key, or in ciphering contexts may refer to the Cipher Key

**KS':** FT authentication Session Key

**KS:** PT authentication Session Key

**Nonce:** part of the Initialization Vector used with CCM

**RAND\_F:** RANDom challenge issued by an FT

**RAND\_P:** RANDom challenge issued by a PT

**RES1:** RESponse calculated by a PT

**RES2:** RESponse calculated by an FT

**RS:** cryptographic parameter used in processes A11 and A21 for the calculation of authentication session keys

**RS<sub>128</sub>:** 128 bit variant of RS

**RS<sub>128</sub>':** the RS<sub>128</sub> used for the generation of KS'

**RS1/RS2:** components of RS<sub>128</sub> when RS<sub>128</sub> is assembled from two parts

**XRES1:** eXpected RESponse calculated by a FT

**XRES2:** eXpected RESponse calculated by a PT

## 3.2 Symbols

For the purposes of the present document, the following symbols apply:

|                    |  |
|--------------------|--|
| C <sub>F</sub>     | higher layer signalling Channel (fast)                                       |
| CI                 | Common Interface   |
| CK                 | Cipher Key   |
| CL <sub>S</sub>    | higher layer Connectionless channel (slow)                                   |
| C <sub>S</sub>     | higher layer signalling Channel (slow)                                       |
| C <sub>T</sub>     | one C <sub>S</sub> or CL <sub>S</sub> channel segment                        |
| G <sub>F</sub>     | higher layer information control channel                                     |
| I <sub>N</sub>     | higher layer Information channel (unprotected) in general                    |
| I <sub>P</sub>     | higher layer Information channel (protected) in general                      |
| I <sub>PQ</sub>    | higher layer Information channel (protected) with single subfield format     |
| I <sub>PX</sub>    | higher layer Information channel, encoded protected, minimum delay operation |
| Lc                 | a DLC layer C-plane protocol entity  |
| M <sub>T</sub>     | MAC control channel on A-tail field, or one message on such channel          |
| N <sub>T</sub>     | identities information channel or one message in such channel                |
| Q <sub>T</sub>     | system information and multiframe marker (MAC logical channel)               |
| S <sub>F</sub> (N) | First of the two KSSs produced by the KSG                                    |
| S <sub>P</sub> (N) | Second of the two KSSs produced by the KSG                                   |

## 3.3 Abbreviations

For the purposes of the present document, the following abbreviations apply:

|         |   |
|---------|---|
| A       | Algorithm   |
| AC      | Authentication Code                               |
| AE      | Authentication Elements                           |
| AES     | Advanced Encryption Standard                      |
| AM      | Authentication Messages                           |
| AUTH    | AUTHentication                                    |
| BCT     | Business Cordless Telephone                       |
| CBC-MAC | Cipher Block Chaining Message Authentication Code |
| CC      | Call Control                                      |
| CCM     | Counter with CBC-MAC                              |
| C-plane | Control plane                                     |
| CRC     | Cyclic Redundancy Check                           |
| CRFP    | Cordless Radio Fixed Part                         |
| CTR     | CounTeR function                                  |
| DAM     | DECT Authentication Module                        |
| DBPSK   | Differential Binary Phase Shift Keying            |

|           |  |
|-----------|--|
| DCK       | Derived Cipher Key   |
| DefCK     | Default Cipher Key   |
| DLC       | Data Link Control  |
| DQPSK     | Differential Quaternary Phase Shift Keying                         |
| DSAA      | DECT Standard Authentication Algorithm                             |
| DSAA2     | DECT Standard Authentication Algorithm #2                          |
| DSC       | DECT Standard Cipher (algorithm)                                   |
| DSC2      | DECT Standard Cipher (algorithm) #2                                |
| DTE       | Data Terminal Equipment  |
| ECN       | Exchanged Connection Number (MAC layer)                            |
| EG        | Experts Group  |
| EI        | Element Identifier   |
| FP        | DECT Fixed Part  |
| FT        | Fixed radio Termination  |
| GFSK      | Gaussian Frequency Shift Keying                                    |
| GSM       | Global System for Mobile communications                            |
| HDB       | Home Data Base   |
| ICC       | Integrated Chip Card   |
| IE        | Information Element  |
| IPUI      | Integrated Portable User Identity                                  |
| ISDN      | International Services Digital Network                             |
| IV        | Initial Vector   |
| IWU       | InterWorking Unit  |
| K         | authentication Key   |
| KDF       | Key Derivation Function  |
| KS'       | FT authentication Session Key                                      |
| KS        | PT authentication Session Key                                      |
| KSG       | Key Stream Generator   |
| KSS       | Key Stream Segment   |
| LAN       | Local Area Network   |
| LAPC      | DLC protocol entity  |
| LBN       | Logical Bearer Number  |
| LCN       | Logical Connection Number (DLC layer)                              |
| LEN       | LENgth   |
| LSB       | Least Significant Byte   |
| MAC (CCM) | Message Authentication Code (in CCM context)                       |
| MAC       | Medium Access Control layer  |
| ME        | Management Entity  |
| MIC       | Message Integrity Code   |
| MM        | Mobility Management  |
| MSB       | Most Significant Bit   |
| MSC       | Mobile Switching Centre  |
| NLMS      | NWK Layer Message Structure  |
| NWK       | NetWoRk  |
| PAD       | Packet Assembler/Disassembler                                      |
| PARI      | Primary Access Rights Identity                                     |
| PARK      | Portable Access Rights Key   |
| PAS       | Public Access Service  |
| PBX       | Private Branch eXchange  |
| PDU       | Protocol Data Unit   |
| PIN       | Personal Identity Number   |
| PMID      | Portable part MAC IDentity (MAC layer)                             |
| PP        | DECT Portable Part   |
| PSTN      | Public Switched Telephone Network                                  |
| PT        | Portable radio Termination   |
| PVC       | Permanent Virtual Circuit  |
| QAM       | Quadrature Amplitude Modulation                                    |
| RA        | Redundancy check of the A-field                                    |
| RFP       | Radio Fixed Part   |
| ROM       | Read Only Memory   |
| RSA       | (Rivest, Shamir and Adleman) algorithm for public-key cryptography |
| RU        | Residential Unit   |

|         |  |
|---------|--|
| SCK     | Static Cipher Key  |
| SDU     | Service Data Unit  |
| SIM     | Subscriber Interface Module  |
| SN      | Send sequence Number   |
| TPUI    | Temporary Portable User Identity   |
| UAK     | User Authentication Key  |
| ULE     | Ultra Low Energy   |
| UPI     | User Personal Identification   |
| U-plane | User plane   |
| VC      | Virtual Call   |
| VDB     | Visitors Data Base   |
| WRS     | Wireless Relay Station   |
| XOR     | eXclusive OR   |
| ZAP     | ability first to assign and then to re-program the account data held in the PP |

---

## 4 Security architecture

### 4.1 Background

Clause 4.2 contains a description of each of the security services provided in the DECT system. Five services are provided:

- authentication of a PT;
- authentication of a FT;
- mutual authentication;
- data confidentiality; and
- user authentication.

For a discussion of the way in which these security services may be applied in different DECT environments, the reader should refer to annex F.

A description of the mechanisms which are used to provide the security services is given in clause 4.3. Throughout clause 4.3 a number of security parameters and processes are referred to. A description of these parameters is given in clause 4.4, and the processes are defined in clause 4.5.

Clause 4.6 describes how the various security services may be combined.

### 4.2 Security services

#### 4.2.1 Authentication of a PT

This is an FT initiated service which enables an FT to authenticate a PT making or receiving a call through it.

The service is invoked at the beginning of a call. It may be re-invoked at any time during a call.

Authentication of a PT is a NWK layer service.

#### 4.2.2 Authentication of an FT

This is a PT initiated service which enables a PT to authenticate an FT through which it is making or receiving a call.

The service is invoked at the beginning of a call, and may be re-invoked at any time during a call.

Authentication of an FT is a NWK layer service.

### 4.2.3 Mutual authentication

This service enables a PT and an FT, through which a call is connected, to authenticate each other.

This service may be provided by combining a number of other security services as described in clause 4.6.

### 4.2.4 Data confidentiality

This service provides for the confidentiality of user data and certain control data transmitted between a PT and an FT.

Data confidentiality is requested at the NWK layer, although the service is provided at the MAC layer.

The service is provided only over the CI. It does not provide any cryptographic protection for data passed through the fixed or any other accessed networks.

### 4.2.5 User authentication

The user authentication service allows an FT to authenticate a user of a PT by checking a User Personal Identity (UPI) value associated with that user. This service is similar to on-line PIN verification provided by banking systems.

The user authentication service is initiated by the FT. It is invoked at the beginning of a call. It may be re-invoked at any time during a call.

## 4.3 Security mechanisms

### 4.3.0 General

The following security mechanisms are defined in the present document:

- Authentication of a PT (type 1 procedure)
- Authentication of a FT (type 1 procedure)
- Mutual authentication
- Derived Cipher Key (DCK)
- Static Cipher Key (SCK)
- Default Cipher Key (DefCK)
- User authentication
- Authentication of an PT (type 2 procedure)
- Authentication of an FT (type 2 procedure)

Authentication of PT and FT type 1 procedures shall be used when DECT Standard Authentication Algorithm (DSAA) or a proprietary algorithm is used. Authentication of PT and FT type 2 procedures shall be used when DECT Standard Authentication Algorithm #2 (DSAA2) is used. Type 2 procedures differ in the parameters exchanged over the air interface procedures.

All other procedures may be used with either, authentication type 1 or type 2 procedures. The specific provisions or differences depending on the authentication used, when needed, are noted in the procedure descriptions.

#### 4.3.1 Authentication of a PT (type 1 procedure)

This mechanism is used when the DECT Standard Authentication Algorithm (DSAA) or a proprietary algorithm is used. See clause 4.3.6 for the equivalent procedure to be used when the DECT Standard Authentication Algorithm #2 (DSAA2) is used.

The purpose of this clause is to define the mechanism which is used to provide the authentication of a PT service defined in clause 4.2.1.

The service is provided using a cryptographic challenge-response mechanism. The FT issues a challenge to the PT, which responds by returning the result of a computation performed using the challenge and an authentication key associated with the PT. The FT compares the response from the PT with the value it expects to receive, and deems the authentication to be successful if the two values agree. In this way the PT is authenticated by demonstrating knowledge of the authentication key associated with it.

The authentication exchange, which includes a key management feature, is illustrated in figure 4.1 and proceeds as follows:

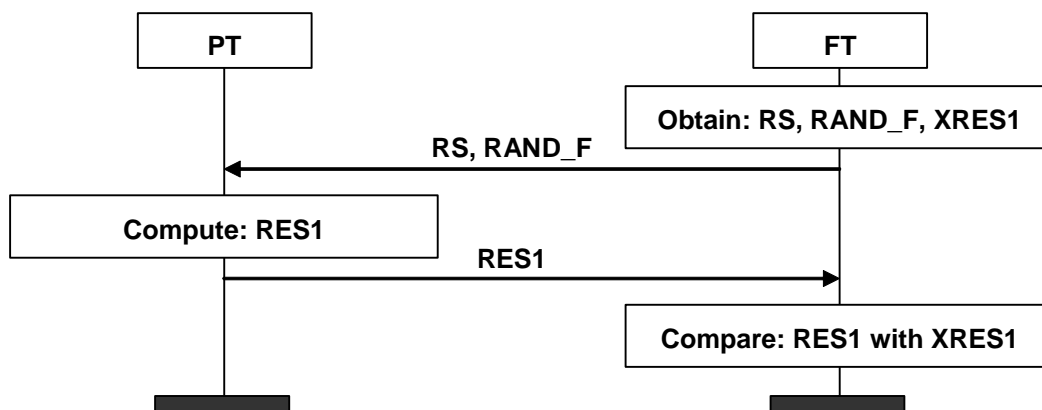
- 1) The FT obtains (and/or generates, and/or computes) a value RS, a value RAND\_F and a value XRES1.

The value XRES1 is the expected result of a computation applied to RS, RAND\_F and the authentication key K associated with the PT.

The computation is performed in two stages using the authentication processes A11 and A12 defined in clause 4.5.3. The first stage uses A11 to produce a value KS from RS and K. The second stage uses A12 to produce XRES1 from RAND\_F and KS. These two computations may be performed by different entities within the FP or the fixed network and, provided the value RS is not changed, the computation of KS need not be repeated for every instance of authentication. All values may be computed in advance of the instance of authentication.

The FT sends the values RS and RAND\_F to the PT.

- 2) On receipt of RS and RAND\_F, the PT uses the authentication process A11 to compute KS from RS and the authentication key K and then uses the authentication process A12 to compute RES1 from KS and RAND\_F. It then sends RES1 to the FT.
- 3) On receipt of RES1, the FT compares this value with XRES1. If the two values are identical, the FT accepts the authenticity of the PT.



**Figure 4.1: Authentication of PT (type 1 procedure)**

The value KS is called an authentication session key. It may be used in a roaming environment to provide a visited system with an authentication key for a particular visiting PT, without the PT's authentication key K being divulged to the visited system. The PT's home system, or a management centre associated with that system, selects an RS for the visited system and generates a KS from RS and the PT's authentication key K using the process A11. The visited system is provided with the pair (RS, KS) for the particular PT. Whenever the visited system authenticates the PT it sends the given value of RS (along with its challenge Rand\_F) to the PT so that the PT may compute KS.

In general, the way in which the FT obtains the values RS, RAND\_F and XRES1 depends upon the particular DECT application and the fixed network management. Thus a detailed specification of all options is outside the scope of the present document. Some options are however outlined in clause 7. Clause 7 also contains a more detailed description of the way in which the session authentication keys may be used in a roaming environment as outlined above.

A detailed description of the parameters RAND\_F and RS is given in clause 4.4.2. The description specifies requirements for the generation of these parameters.

The authentication key  $K$  is described in clause 4.4.3. The authentication process A11 used to compute  $KS$  from  $RS$  and  $K$  is described in clause 4.5.3.1, and the authentication process A12 used to compute  $XRES1$  (and  $RES1$ ) from  $RAND\_F$  and  $KS$  is described in clause 4.5.3.2.

### 4.3.2 Authentication of an FT (type 1 procedure)

This mechanism is used when the DECT Standard Authentication Algorithm (DSAA) or a proprietary algorithm is used. See clause 4.3.7 for the equivalent procedure to be used when the DECT Standard Authentication Algorithm #2 (DSAA2) is used.

The purpose of this clause is to define the mechanism which is used to provide the authentication of an FT service defined in clause 4.2.2.

The service is provided using a cryptographic challenge-response mechanism. The PT sends a challenge to the FT, which responds by returning the result of a computation performed using the challenge and an authentication key associated with the PT. The PT computes the expected value for the computation and deems the authentication to be successful if this value agrees with the one received from the FT. In this way, the FT is authenticated by demonstrating that it can provide the result of a computation that depends upon knowledge of an authentication key associated with the challenging PT.

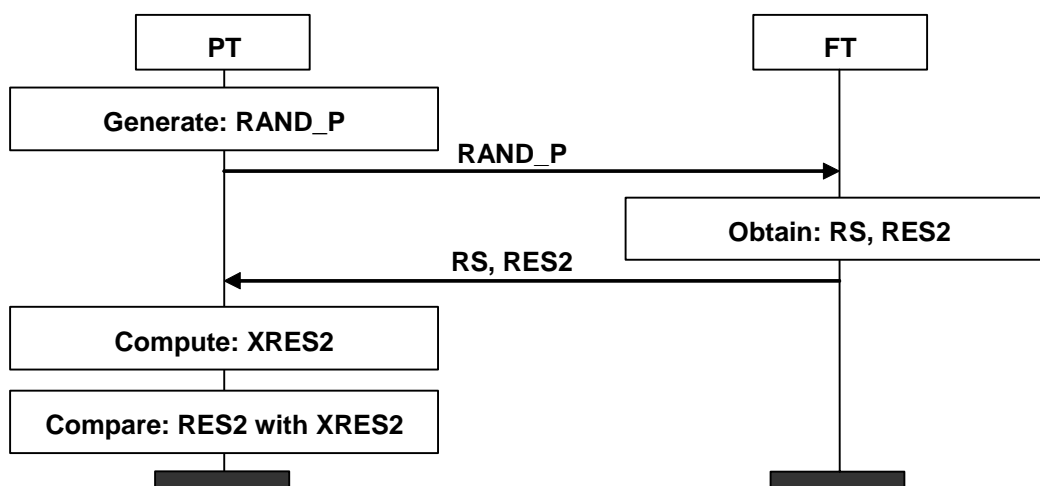
**NOTE:** Throughout the text the authentication key associated with the PT, and used to authenticate the FT, is denoted  $K$ . The reader should, however, note that this key may be different from that associated with the PT and used to authenticate the PT as described in clause 4.3.1. Similarly, the values  $RS$  used in the two mechanisms are not necessarily the same.

The authentication exchange, which includes a key management feature, is illustrated in figure 4.2 and proceeds as follows:

- 1) The PT generates a value  $RAND\_P$  and sends it to the FT.
- 2) The FT obtains (and/or computes) a value  $RS'$  and a value  $RES2$ , and sends them both to the PT. The  $RS$  used for this purpose is named  $RS'$ .

The value  $RES2$  is the result of a computation applied to  $RS'$ ,  $RAND\_P$  and the authentication key  $K$  associated with the PT. The computation is performed in two stages using the authentication processes A21 and A22 defined in clause 4.5.3. The first stage uses A21 to produce a value  $KS'$  from  $RS'$  and  $K$ . The second stage uses A22 to produce  $RES2$  from  $RAND\_P$  and  $KS'$ . These two computations may be performed by different entities within the fixed network. Moreover, the computation of  $KS'$  may be performed in advance of the instance of authentication and, provided the value of  $RS'$  is not changed, the computation does not need to be repeated for subsequent instances of authentication.

- 3) On receipt of  $RS'$  and  $RES2$ , the PT applies the authentication process A21 to compute  $KS'$  from  $RS'$  and the authentication key  $K$ , and then uses the authentication process A22 to compute an expected result  $XRES2$  from  $KS'$  and  $RAND\_P$ . If  $XRES2$  is equal to the value  $RES2$  received from the FT, the PT accepts the authenticity of the FT.



**Figure 4.2: Authentication of FT (type 1 procedure)**

The value  $KS'$  is called an authentication session key. It may be used in a roaming environment to provide a visited system with a key by which it may authenticate itself to a particular visiting PT, without the PT's authentication key  $K$  being divulged to the visited system. The PT's home system, or a management centre associated with that system, selects an  $RS'$  for the visited system and generates a  $KS'$  from  $RS'$  and the PT's authentication key  $K$ . The visited system is provided with the pair  $(RS', KS')$  for the particular PT. The visited FT, or a management centre associated with it, always uses the given value of  $KS'$  to compute  $RES2$  from  $RAND\_P$ , and always sends the given  $RS$  in the response to the PT's challenge.

In general, the precise way in which the FT obtains the values  $RS'$  and  $RES2$  will depend upon the particular DECT application and the fixed network management. Thus a detailed specification of all options is outside the scope of the present document. Some options are, however, outlined in clause 7. A detailed description of the parameters  $RAND\_P$  and  $RS'$  is given in clause 4.4.2. The description specifies requirements for the generation of these parameters.

The authentication key,  $K$ , is described in clause 4.4.3.

The authentication process A21, used to compute  $KS'$  from  $RS'$  and  $K$ , is described in clause 4.5.3.1, and the authentication process A22 used to compute  $XRES2$  (and  $RES2$ ) from  $RAND\_P$  and  $KS'$ , is described in clause 4.5.3.2.

### 4.3.3 Mutual authentication

Mutual authentication is achieved by combining certain of the other services as described in clause 4.6. No explicit mutual authentication mechanism is required.

### 4.3.4 Data confidentiality

#### 4.3.4.0 General

In order to provide the data confidentiality service defined in clause 4.2.4, both the PT and the FT shall share a cipher key  $CK$ . This key is then used in conjunction with a key stream generator to generate a key stream for encrypting data at the MAC layer. The key stream generation process is specified in clause 4.5.4. The detailed specification of how encryption is performed at the MAC layer is the subject of clause 6.4. The purpose of this clause is to specify how a  $CK$  may be established. Two ways are provided, thereby providing two types of key:

- a Derived Cipher Key (DCK); and
- a Static Cipher Key (SCK).



#### 4.3.4.1 Derived Cipher Key (DCK)

The DCK is established as part of the procedure used to authenticate the PT, either type 1 or type 2, specified in clauses 4.3.1 or 4.3.6. More precisely, the authentication process A12, when applied to KS and RAND\_F (and RAND\_P when authentication type 2 is used), has two output values:

- the value RES1 (or XRES1); and
- the value DCK.

The generated Derived Cipher Key (DCK) has 64 bits when type 1 authentication procedure (clause 4.3.1) is used and 128 bits when type 2 authentication procedure (clause 4.3.2) is used.

The PT computes DCK as part of the procedure described in 2) of clause 4.3.1 or 7) in clause 4.3.6. The FT obtains DCK as part of the procedure for obtaining the values RS, RAND\_F and XRES1 as described in 1) of clause 4.3.1 or 10) of clause 4.3.6. The DCK shall be made available to the appropriate radio fixed part of the FT.

This technique allows a new cipher key to be established for each call, and used for the duration of that call, provided the PT is authenticated at the beginning of the call.

Under certain circumstances (see clause 4.6), the DCK may be retained and used by the PT and FT for providing confidentiality for all calls until the next authentication of the PT takes place. At that point, it is replaced by the new DCK.

It should be noted that authentication of an FT cannot be used to establish a cipher key.

#### 4.3.4.2 Static Cipher Key (SCK)

With the cipher key derivation process defined in clause 4.3.4.1, the cipher key is obtained as part of the PT authentication process, with PT authentication being a necessary pre-requisite for its derivation. However, in certain applications, it may be desirable to be able to provide confidentiality without having to first apply authentication. In this case the parties may use a cipher key which has been established by other means. Such a key is called a Static Cipher Key (SCK).

NOTE: Although this mechanism allows for confidentiality of user and control data without the need for authentication, no service is included within the DECT security features to provide for management of SCKs.

#### 4.3.4.3 Default Cipher Key (DefCK)

With the cipher key derivation process defined in clause 4.3.4.1, it is also possible to generate a derived cipher key which can only be used as a default cipher key. This derived cipher key is stored in both FP and PP and later on used by MAC to immediately encrypt with connection establishment.

Such a cipher key has a system wide unique index which is referred to in the MAC procedure for activation of default encryption.

NOTE: Encryption with the default cipher key cannot be requested via the {CIPHER-REQUEST} message but only via the MAC procedure for activation of encryption with the default cipher key index (see clause 6.4.6.2).

### 4.3.5 User authentication

User authentication is achieved by applying the authentication of the PT service using an authentication key K which is derived from a UPI value.

Both PT authentication types, 1 or 2 may be used.

If authentication type 2 is used, an independent parameter  $RS_{128}$  shall be generated by the FT to be used in the process and shall not be reused in any other authentication process with the same or any other Key (K).

NOTE: In authentication type 1 it is advisable, but not mandated, to follow the same rule.

The UPI is entered manually into the PT by the user whenever the user authentication service is invoked. Within the PT the UPI is used to generate the authentication key K, possibly in combination with other key material already present in the PT. The key K is also generated by the system responsible for the user's subscription data (using the same key material) for use by the fixed system.

A mechanism is specified in clause 4.5.2 to combine a UPI with a User Authentication Key (UAK) to produce an authentication key K for the user authentication service. A user authentication key is part of a user's subscription data, established at the time of subscription registration, and is described in clause 4.4.3.

It should be stressed that user authentication requires a user to enter his UPI each time user authentication is required.

### 4.3.6 Authentication of a PT (type 2 procedure)

This mechanism is used when the DECT Standard Authentication Algorithm #2 (DSAA2) is used. See clause 4.3.1 for the equivalent procedure when the DECT Standard Authentication Algorithm (DSAA) or a proprietary algorithm is used.

NOTE 1: The possible use of the type 2 procedure in combination with proprietary algorithms is left for further study.

The purpose of this clause is to define the mechanism which is used to provide the authentication of a PT service defined in clause 4.2.1.

The service is provided using a cryptographic challenge-response mechanism. The FT issues a challenge to the PT, which responds by returning the result of a computation performed using the received challenge, a second challenge added by the PT and an authentication key associated with the PT. The FT compares the response from the PT with the value it expects to receive, and deems the authentication to be successful if the two values agree. In this way the PT is authenticated by demonstrating knowledge of the authentication key associated with it.

The differences in this mechanism compared to the DSAA variant (clause 4.3.1) are the following:

- The security processes architecture shown in figure 0.2 (introduction clause), including the parameter flows shown in dotted lines, applies.
- Two random numbers generated from both peers and exchanged over the air interface are used for the calculation of RES1 and XRES1.
- The size of the parameter RS is increased to 128 bits ( $RS_{128}$ ).  $RS_{128}$  is obtained by concatenating two 64 bits parameters, RS1 and RS2 (with RS2 in the most significant bits).
- It is explicitly stated that all random parameters, including  $RS_{128}$ , shall be used only once.

NOTE 2: The consideration of  $RS_{128}$  as composed of two 64 bits parameters, RS1 and RS2 is done to allow flexibility in the design of complex Fixed Part structures (scenarios with separation home/visited environment). For all air interface procedures both parameters operate always combined as a single  $RS_{128}$  of 128 bits.

The authentication exchange, which includes a key management feature, is illustrated in figure 4.3 and proceeds as follows:

- 1) The FT obtains (and/or generates, and/or computes) a random number  $RS_{128}$  not used before.

In complex Fixed parts,  $RS_{128}$  may be obtained by concatenation of two 64 bits random numbers named RS1 and RS2 (with RS2 in the most significant bits), potentially generated in two different places of the Fixed part.

NOTE 3: However this is irrelevant for the DECT air interface procedures (it may be an FP implementation option or in the scope of other standards defining Fixed Part architectures for specific applications).

- 2) With the "fresh" value of  $RS_{128}$  the FT generates KS by means of the authentication process A11.

It is not required to run process A11 generating a new KS for every PT authentication request. The FT shall internally store the last generated KS and the  $RS_{128}$  used for its generation.

$RS_{128}$  shall be used for only one execution of process A11.

NOTE 4: As  $RS_{128}$  is also used in process A21 (see clause 4.3.7), this means that a new  $RS_{128}$  should be generated after every execution of either process A11 or process A21.

NOTE 5: By the same reason, the FT should store internally the value of KS together with the  $RS_{128}$  used for its generation, that may be different from the last generated  $RS_{128}$  due to the potential execution of process A21.

- 3) The FT generates a random value,  $RAND\_F$ , used only once for this authentication process.
- 4) The FT sends to the PT an authentication request including the parameters  $RAND\_F$  and  $RS_{128}$ .
- 5) The PT generates a random value,  $RAND\_P$ , used only once for this authentication process.
- 6) The PT checks if it has internally stored a KS associated to the received value of  $RS_{128}$ . Otherwise, the PT executes authentication process A11 using K and the received  $RS_{128}$  as inputs.
- 7) The PT executes the authentication process A12 using as inputs KS,  $RAND\_F$  and  $RAND\_P$ . It obtains the parameter RES1 (32 bits) and a new Derived Ciphering Key of 128 bits (see clause 4.3.4.1).
- 8) The PT sends the computed RES1 together with the random value  $RAND\_P$  to the FT by means of an {AUTHENTICATION-REPLY} message.
- 9) On receipt of RES1 and  $RAND\_P$ , the FT uses the authentication process A12 to compute XRES1 (inputs: KS,  $RAND\_F$ ,  $RAND\_P$ ) and compares this value with the RES1 sent by the PT. If the two values are identical, the FT accepts the authenticity of the PT.
- 10) As result of the execution of process A12, the FT obtains also a new value of the DCK of 128 bits (see clause 4.3.4.1).

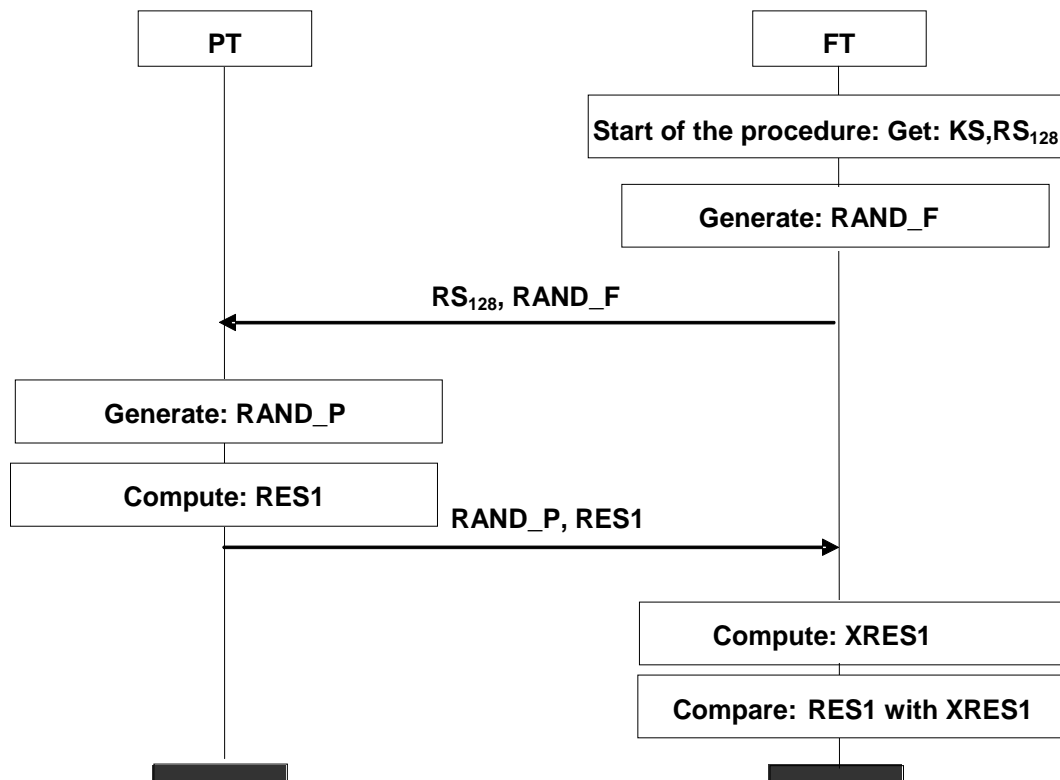


Figure 4.3: Authentication of PT (type 2 procedure)

The value KS is called an authentication session key. It may be used in a roaming environment to provide a visited system with an authentication key for a particular visiting PT, without the PT's authentication key K being divulged to the visited system. The PT's home system, or a management centre associated with that system, gets or generates a random value for  $RS_{128}$  and generates a KS from  $RS_{128}$  and the PT's authentication key K using the process A11. The visited system is provided with the pair ( $RS_{128}$ , KS) for the particular PT. Whenever the visited system authenticates the PT it sends the given value of  $RS_{128}$  (along with its challenge RAND\_F) to the PT so that the PT may compute KS.

For flexibility, the value of  $RS_{128}$  is considered as composed of two 64 bits random numbers RS1 and RS2 concatenated. This provision is introduced to allow the scenario when one of them (RS1) is generated at the home environment while the other (RS2) is generated at the visited domain and retrieved by the home environment in order to generate KS by means of process A11.

NOTE 6: See figure 7.5 in clause 7.2.4.2 for an example of a network where RS1/RS2 are generated in different places (RS1 is generated in the home domain and RS2 in the visited domain).

In general, the way in which the FT obtains the values  $RS_{128}$ , RAND\_F and XRES1 depends upon the particular DECT application and the fixed network management. Thus a detailed specification of all options is outside the scope of the present document. Some options are however outlined in clause 7. However, the following restriction applies when authentication type 2 is used:

- The scenario "Roaming using precalculated sets" described in figure 7.3 is not supported when authentication type 2 is used.

A detailed description of the parameters RAND\_F, RAND\_P and  $RS_{128}$  is given in clause 4.4.2. The description specifies requirements for the generation of these parameters.

The authentication key K is described in clause 4.4.3. The authentication process A11 used to compute KS from RS and K is described in clause 4.5.3.1, and the authentication process A12 used to compute XRES1 (and RES1) from RAND\_F and KS is described in clause 4.5.3.2.

### 4.3.7 Authentication of a FT (type 2 procedure)

This mechanism is used when the DECT Standard Authentication Algorithm #2 (DSAA2) is used. See clause 4.3.2 for the equivalent procedure when the DECT Standard Authentication Algorithm (DSAA) or a proprietary algorithm is used.

NOTE 1: The possible use of the type 2 procedure in combination with proprietary algorithms is left for further study.

The purpose of this clause is to define the mechanism which is used to provide the authentication of an FT service defined in clause 4.2.2.

The service is provided using a cryptographic challenge-response mechanism. The PT sends a challenge to the FT, which responds by returning the result of a computation performed using the challenge received from the PT, a second challenge added by the FT and an authentication key associated with the PT. The PT computes the expected value for the computation, and deems the authentication to be successful if this value agrees with the one received from the FT. In this way, the FT is authenticated by demonstrating that it can provide the result of a computation that depends upon knowledge of an authentication key associated with the challenging PT.

NOTE 2: Throughout the text the authentication key associated with the PT, and used to authenticate the FT, is denoted K. The reader should, however, note that this key may, in theory, be different from that K used to authenticate the PT as described in clause 4.3.6. However, this is not expected to be the current practice, and K would be, in general, unique for each PP and used for both, PT and FT authentication processes.

NOTE 3: On the other hand, the values of the session keys (KS) used for both PT and FT authentication will be always different due to the requirement of using  $RS_{128}$  only once, and therefore different for processes A11 and A21.

The differences in this mechanism compared to the DSAA variant (clause 4.3.2) are the following:

- The security processes architecture shown in figure 0.2 (introduction clause), including the parameter flows shown in dotted lines, applies.

- Two random numbers generated from both peers and exchanged over the air interface are used for the calculation of RES2 and XRES2.
- The size of the parameter RS is increased to 128 bits ( $RS_{128}$ ).  $RS_{128}$  is obtained by concatenating two 64 bits parameters, RS1 and RS2 (with RS2 in the most significant bits) (see note 2 of clause 4.3.6).
- It is explicitly stated that all random parameters, including  $RS_{128}$ , shall be used only once.

The authentication exchange, which includes a key management feature, is illustrated in figure 4.4 and proceeds as follows:

- 1) The PT generates a "fresh" value RAND\_P not used before.
- 2) The PT constructs an {AUTHENTICATION-REQUEST} message, including the parameter RAND\_P, and sends it to FT.
- 3a) The FT obtains (computes or retrieves) the last value of KS generated by the process A21 (that will be named KS') and the associated value of  $RS_{128}$  that was used for its generation (that will be named  $RS_{128}'$ ); OR
- 3b) The FT generates a new value of  $RS_{128}$  (that will be named  $RS_{128}'$ ), executes process A21 using  $RS_{128}'$  and K as inputs and generates KS (that will be named KS').

$RS_{128}'$  shall be used only once for each execution of process A21 and shall be different from any  $RS_{128}$  used in the execution of process A11.

NOTE 4: However,  $RS_{128}'$  may be stored until next execution of process A21 and reused by several executions of process A22.

$KS'/RS_{128}'$  are, by definition, the  $RS_{128}$  used and the KS generated by the last execution of process A21.

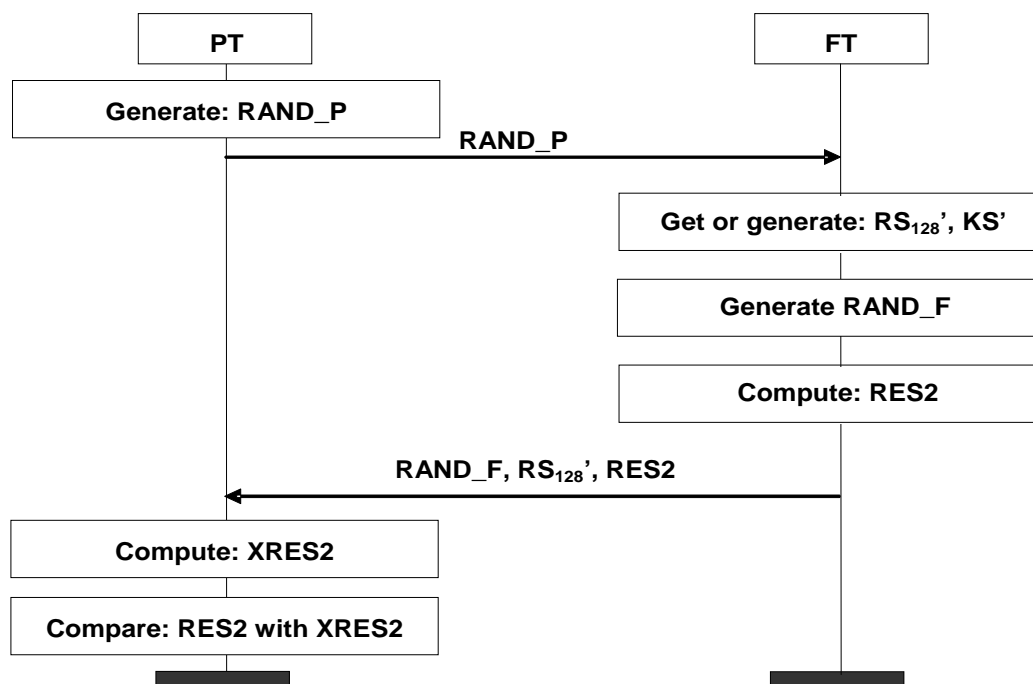
NOTE 5: The nomenclature  $KS'/RS_{128}'$  is used to differentiate them from a different pair  $KS/RS_{128}$  generated by process A11 related to the PT authentication.

- 4) The FT generates a "fresh" value of random parameter RAND\_F not used before.
- 5) The FT computes the value of RES2 running authentication process A22 and using as inputs: RAND\_F, RAND\_P and KS'.
- 6) The FT sends to the PT an {AUTHENTICATION-REPLY} message including the calculated value RES2 together with the parameters RAND\_F and  $RS_{128}'$  used in its generation.
- 7) The PT independently calculates the value of XRES2 running process A22 and using KS',  $RS_{128}'$ , RAND\_P and RAND\_F as inputs.

In order to get KS', the PT may check if the received  $RS_{128}'$  matches with the value of  $RS_{128}'$  it has stored from last execution of process A21. In such a case, the PT may use the last stored value of KS'. Otherwise, it shall run process A21 with the received  $RS_{128}'$  to produce a new KS'.

- 8) If XRES2 is equal to the value RES2 received from the FT, the PT accepts the authenticity of the FT.

NOTE 6: Depending on the application, and the level of authentication required, it may be possible to allow the storage of session keys KS' (the step 2a) or it may be necessary to mandate an execution of process A21 (the step 2b) for each instance of the FT authentication process. This provision, if needed, may be defined by an application profile.



**Figure 4.4: Authentication of FT**

The value  $KS'$  is called an authentication session key. It may be used in a roaming environment to provide a visited system with a key by which it may authenticate itself to a particular visiting PT, without the PT's authentication key  $K$  being divulged to the visited system. The PT's home system, or a management centre associated with that system, selects an  $RS_{128}'$  for the visited system and generates a  $KS'$  from  $RS_{128}'$  and the PT's authentication key  $K$  using process A21. The visited system is provided with the pair  $(RS_{128}', KS')$  for the particular PT. The visited FT, or a management centre associated with it, always uses the given value of  $KS'$  and  $RS_{128}'$  to compute  $RES2$  and always sends the given  $RS_{128}'$  in the response to the PT's challenge.

For flexibility, the value of  $RS_{128}$  (in this case named  $RS_{128}'$ ) is considered as composed of two 64 bits random numbers  $RS1$  and  $RS2$  concatenated. This provision is introduced to allow the scenario when one of them ( $RS1$ ) is generated at the home environment while the other ( $RS2$ ) is generated at the visited domain and retrieved by the home environment in order to generate  $KS$  by means of process A11.

In general, the way in which the FT obtains the values  $RS_{128}$ ,  $RAND_F$  and  $XRES1$  depends upon the particular DECT application and the fixed network management. Thus a detailed specification of all options is outside the scope of the present document. Some options are however outlined in clause 7. However, the following restriction applies:

- The scenario "Roaming using recalculated sets" described in figure 7.3 is not supported when authentication type 2 is used.

A detailed description of the parameters  $RAND_F$ ,  $RAND_P$  and  $RS_{128}$  is given in clause 4.4.2. The description specifies requirements for the generation of these parameters.

The authentication key  $K$  is described in clause 4.4.3. The authentication process A11 used to compute  $KS$  from  $RS$  and  $K$  is described in clause 4.5.3.1 and the authentication process A12 used to compute  $XRES1$  (and  $RES1$ ) from  $RAND_F$  and  $KS$  is described in clause 4.5.3.2.

## 4.4 Cryptographic parameters and keys

### 4.4.1 Overview

In this clause the cryptographic parameters and types of keys used in the security architecture are described.

A length in bits is given for each parameter or key. This is the length of the parameter or key for use with the DECT standard algorithms.

The expression "non-repeating" means that it should be highly unlikely that the value should repeat itself within the lifetime of the authentication key. A "non-repeating" value could be, for example, the output from a counter which is unlikely to repeat during the lifetime of the authentication key, or a date/time stamp.

The expression "randomly generated" means that it should not be possible to predict the value with a chance that is significantly larger than 0 (for example greater than  $(0,5)^{64}$  for the values defined in the present document).

## 4.4.2 Cryptographic parameters

### 4.4.2.0 Description of parameters

The following authentication parameters are used and (except XRES1 and XRES2) transmitted over the air interface.

#### **RAND\_F:**

This parameter is sent from the FT to the PT as part of the authentication of a type 1 PT exchange defined in clause 4.3.1 and as part of type 2 PT and FT exchanges defined in clauses 4.3.6 and 4.3.7.

It has a length of 64 bits.

It is generated by the FT or within a local network. In a roaming scenario, it may be generated within the subscriber's home network (see clause 7.2 for a discussion of the options for the generation of RAND\_F in a roaming environment). A new value of RAND\_F shall be randomly generated in all cases for each instance of authentication of a particular PT (execution of process A12), and also for each instance of FT authentication, when type 2 exchanges are used (execution of process A22).

#### **RS:**

This parameter is used to enable roaming between networks. It is an element of the mechanism which allows authentication session keys to be made available to visited networks (see also the discussion of KS and KS' in clause 4.4.3).

It is sent from the FT to the PT as part of the authentication of a PT or authentication of an FT exchange defined in clauses 4.3.1 and 4.3.2 respectively.

When used in authentication type 1 procedures, it has a length of 64 bits (see next entry for the 128 variant RS<sub>128</sub>, to be used in authentication type 2 procedures).

It is generated by the FP, within a local network or, in a roaming scenario, within the subscriber's home network.

Different values of RS may be used for authentication of the PT and authentication of the FT even though the two services may be invoked during a single call. It is, however, envisaged that, in this situation, the same value will be used.

The same value may be used a number of times and for a number of different subscribers. For example, in the roaming environment, a single constant value of RS may be selected by a home network for use by a particular visited network (see clause 7.2 for more details).

When used in authentication type 1 procedures, the present document advises but not requires RS to be generated as a random number. The following advisable practices are also recommended:

- Do not use a pure static value for RS. Update the RS value from time to time.
- Update the value of RS after any change in the K value.

#### **RS<sub>128</sub>:**

This is the variant of RS to be used when type 2 authentication exchanges are used (see clauses 4.3.6 and 4.3.7).

It has a length of 128 bits.

$RS_{128}$  shall be a random number used only once for each execution of either process A11 or process A21.

$RS_{128}$  parameter is used to enable roaming between networks. It is an element of the mechanism which allows authentication session keys to be made available to visited networks (see also the discussion of KS and KS' in clause 4.4.3).

It is sent from the FT to the PT as part of the authentication of a PT or authentication of an FT exchange defined in clauses 4.3.6 and 4.3.7 respectively.

It is generated by the FP, within a local network. In a roaming scenario, it may be generated within the subscriber's home network, or part of the parameter may be generated in the home network and part of it in the visited network.

In the last case, it shall be assumed that  $RS_{128}$  is composed of two Random numbers of 64 bits, RS1 and RS2 concatenated (RS2 in the most significant bits).

**NOTE:** The consideration of  $RS_{128}$  as composed of two 64 bits parameters, RS1 and RS2 is done to allow flexibility in the design of complex Fixed Part structures (scenarios with separation home/visited environment). For all air interface procedures both parameters operate always combined as a single  $RS_{128}$ . See figure 7.5 in clause 7.2.4.2 for an example of a network where RS1/RS2 are generated in different places (RS1 is generated in the home domain and RS2 in the visited domain).

$RS_{128}$  may be generated directly as a 128 bit random number. In systems where RS1 and RS2 are used,  $RS_{128}$  may be built by concatenating two 64 bit random numbers (RS1/RS2), or by concatenating an RS1 produced following the rules defined for RS with an RS2 that shall be a 64 bits random number used only once.

Different values of  $RS_{128}$  shall be used for authentication of the PT and authentication of the FT even though the two services may be invoked during a single call. In authentication type 2, it is not allowed that the same value of  $RS_{128}$  to be re-used for both PT and FT authentications.

The same value of  $RS_{128}$  may be used a number of times for several executions of the PT authentication procedure, running each of them the process A12.

Conversely, the same value (but different from the previous one) may be used a number of times for several executions of the FT authentication procedure, running each of them the process A22.

The terminology  $RS_{128}'$  refers to the  $RS_{128}$  used in the FT authentication procedures (input to process A21).

See clause 7.2 for more details about roaming scenarios.

#### **RES1:**

This value is sent from the PT to the FT, as part of the authentication of a PT exchange defined in clause 4.3.1 (type 1) or 4.3.6 (type 2), where it is compared with XRES1.

It has a length of 32 bits.

It is computed by the PT from RAND\_F and KS (and RAND\_P if authentication type 2 is used) using the authentication process A12.

#### **XRES1:**

This value is compared with RES1 at the FT as part of the authentication of a PT mechanism. Authentication succeeds if the two values are equal.

It has a length of 32 bits.

It is computed from RAND\_F and KS (and RAND\_P if authentication type 2 is used) using the authentication process A12. Computation of XRES1 may be performed by the FT or within a local network. In a roaming scenario, it shall be generally computed in the visited domain. However if authentication type 1 is used, it may be computed within the subscriber's home network (see clause 7.2 for more details concerning the possible locations for computing XRES1 in a roaming environment).



**RAND\_P:**

This parameter is sent from the PT to the FT as part of the type 1 authentication of an FT exchange defined in clause 4.3.2 and as part of type 2 PT and FT exchanges defined in clauses 4.3.6 and 4.3.7.

It has a length of 64 bits.

It is generated by the PT.

A new value of RAND\_P shall be randomly generated in all cases for each instance of FT authentication (execution of process A22), and also for each instance of PT authentication (execution of process A12), when type 2 exchanges are used. The generation mechanism shall produce random, non-repeating values.

RAND\_P is never reused for the execution of more than one authentication processes. The only exception to this rule is the Key Allocation procedure when a single RAND\_P parameter is reused by two authentication processes (PT and FT authentication). See clause 6.5.6.

**RES2:**

This value is sent from the FT to the PT, as part of the authentication of an FT exchange defined in clause 4.3.2, where it is compared with XRES2.

It has a length of 32 bits.

It is computed from RAND\_P and KS' (and RAND\_F if authentication type 2 is used) using the authentication process A22. Computation of RES2 may be performed by the FT or within a local network.

**XRES2:**

This value is compared with RES2 at the PT as part of the authentication of an FT mechanism. Authentication succeeds if the two values are equal.

It has a length of 32 bits.

It is computed by the PT from RAND\_P and KS' (and RAND\_F if authentication type 2 is used) using the authentication process A22.

#### 4.4.2.1 Provisions related to the generation of random numbers

For the purposes of the present document, provisions related to the generation of "non repeating", "not used before", "new" or "fresh" random number parameters, shall be understood as a requirement to produce the parameter running a properly designed random number generator, with the capability to produce a new, non predictable value. It is not required, and not recommended, the storage of the previously generated series of parameters in order to fulfil with absolute guarantee the "not repeating" rule.

NOTE: Giving the number of bits of all random parameters used by the present document, assuming a proper design of the random number generators, the probability of a potential repetition of values is negligible.

### 4.4.3 Cryptographic keys

#### 4.4.3.0 General

There are three types of cryptographic key used to provide the DECT security features: the authentication key K, the authentication session keys KS and KS', and the cipher key CK.

#### 4.4.3.1 Authentication key K

This is the key value used in the authentication mechanisms defined in clauses 4.3.1 and 4.3.2.

The authentication key, K, has a length of 128 bits.

The authentication key may be derived from other keys associated with the user. Such keys may be specific to a particular DECT application or a particular authentication service. Three such keys are identified below. The processes for deriving K from these keys are given in clause 4.5.2.

**User Authentication Key (UAK):**

This is secret authentication data contained within the subscriber's (user's) registration data. It is uniquely associated with the particular subscriber (user) and the subscription.

It may be used to derive K when authentication of the PT or authentication of an FT service is applied. It may be used in combination with the UPI to derive K when the user authentication service is applied.

The UAK is held in non-volatile memory within the PP (or within a detachable DECT Authentication Module (DAM)).

**User Personal Identity (UPI):**

This is usually a short value (for example 16 bits to 32 bits).

It is used in combination with the UAK to derive K when the user authentication service is applied.

A single UAK may have at most one UPI associated with it.

The UPI is not stored in the PT. It is entered manually into the PT by the user each time the user authentication service is required.

**Authentication Code (AC):**

This is usually a short value (for example 16 bits to 32 bits).

It may be used to derive K when authentication of the PT or authentication of an FT service is applied.

The AC may be held in non-volatile memory within the PP or it may be manually entered by the user when required for an authentication service. This depends upon the application.

NOTE: There is no conceptual difference between a UAK and an AC. It is intended that an AC should be used only for those applications which require a temporary or short-term key for PT or FT authentication. For all other applications a UAK is more appropriate.

For a discussion of the ways in which K (UAK, UPI and AC) may be distributed to the PT and managed within fixed networks the reader is referred to clause 7.2.

#### 4.4.3.2 Authentication session keys KS and KS'

**Authentication session key, KS:**

In a roaming environment it may be necessary or highly convenient for a PT's home network to provide visited networks with information sufficient for them to authenticate the PT, without necessarily having to divulge the authentication key K and without the need for a visited network to contact the home network at every instance of authentication. An authentication session key KS may be used for this purpose. The visited network is provided (by the home network) with a KS for use with a particular PT and may use it for an indefinite number of calls.

The key KS is computed from K and RS using the authentication process A11 as specified in clause 4.5.3.1.

Computation of KS within the PT occurs as part of the authentication of a PT exchange.

Computation of KS on the fixed side of the air interface may be performed within the home network, within a local network or at the RFP. The location depends upon the application and the key management procedures in force. Although computation of KS is part of the authentication of a PT mechanism, it does not have to be performed at an instance of authentication or repeated for each instance of authentication. It only needs to be repeated if the value of RS (or K) is changed.

KS has a length of 128 bits.

**Authentication session key, KS':**

KS' is defined as the session key (KS) generated by authentication process A21 and intended to be used in FT authentication procedures.

In a roaming environment it may be necessary for a PT's home network to provide visited networks with information sufficient for them to be authenticated by the PT, without necessarily having to divulge the authentication key K and without the need for a visited network to contact the home network at every instance of authentication. An authentication session key KS' may be used for this purpose. The visited network is provided (by the home network) with a KS' for use with a particular PT, and may use it for an indefinite number of calls.

The key KS' is computed from K and RS using the authentication process A21 as specified in clause 4.5.3.1. The RS used for this purpose is named RS' (or RS<sub>128</sub>' if authentication type 2 exchange is used).

Computation of KS' within the PT occurs as part of the authentication of the FT exchange.

Computation of KS' on the fixed side of the air interface may be performed within the home network, within a local network or at the FT. The location depends upon the application and the key management procedures in force. Although computation of KS' is part of the authentication of the FT mechanism, it does not have to be performed at an instance of authentication or repeated for each instance of authentication. It only needs to be repeated if the value of RS (or K) is changed.

KS' has a length of 128 bits.

NOTE 1: The session keys KS and KS' may be derived from different values of K although it is envisaged that the values will usually be the same. On the other hand, KS and KS' may be usually derived from different values of RS (mandatory if authentication type 2 is used, and advisable in any case). One scenario where the values for K may well be different is when both user authentication and authentication of the FT are separately applied. In this case, the value for K used to apply user authentication may be derived from a UAK and a UPI (see clause 4.4.3.1), whilst the value used for authentication of the FT may be derived simply from the UAK.

NOTE 2: If the same values for K are used to derive KS and KS', the values of KS and KS' will be different; when DSAA algorithm is used, authentication processes A11 and A21 produce different outputs for identical inputs (see clause 4.5.3.1). When DSAA2 is used, it is not allowed to reuse RS<sub>128</sub>, therefore the values of KS and KS' will be also different.

#### 4.4.3.3 Cipher key CK

This is a value which is used as the key for the encryption process. It may be a DCK. In this case it is computed using the authentication process A12 (as specified in clause 4.5.3.2) during authentication of the PT.

Alternatively it may be a SCK shared by the FT and the PT.

CK may have a length of 64 bits or 128 bits.

- DCK generated by DSAA2 algorithm has a length of 128 bits.
- DCK generated by DSAA or any other CK have a length of 64 bits.
- When DSC is used, a CK of 64 bits shall be used. In the event of a DCK generated by DSAA2, the 64 least significant bits shall be used.
- When DSC2 is used, a CK of 128 bits shall be used. In the event of a DCK generated by DSAA, a proprietary algorithm, or any other type of CK are used, it shall be extended to 128 bits by adding "0" in the 64 most significant bits.
- When CCM is used, a CK of 128 bits shall be used. In the event of a DCK generated by DSAA, a proprietary algorithm, or any other type of CK are used, it shall be extended to 128 bits by adding "0" in the 64 most significant bits.

## 4.5 Security processes

### 4.5.1 Overview

This clause is divided into three main parts. In the first part, clause 4.5.2, the processes which are supported for the derivation of the authentication key  $K$  are defined. The second part, clause 4.5.3, is concerned with the authentication processes. In the third part, clause 4.5.4, the basic encryption process is defined.

For an overview of the DECT security processes the reader is referred to figure 0.2. This illustrates the relationship between the processes defined in this clause and the parameters and keys defined in clause 4.4. The lengths of values are those for use with the DECT standard algorithms.

### 4.5.2 Derivation of authentication key, $K$

#### 4.5.2.0 General

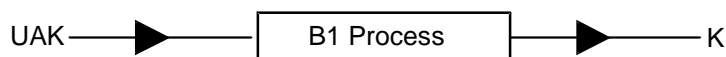
Three processes have been defined for the derivation of the authentication key  $K$ . These are specified in clauses 4.5.2.1, 4.5.2.2 and 4.5.2.3.

Throughout this clause the following notation is used:

- $LEN\_X$  denotes the bit-length of value  $X$ ;
- $X[i]$  denotes the  $i$ 'th bit of  $X$  for  $0 \leq i \leq LEN\_X - 1$ ;
- $X[0]$  denotes the least significant bit of  $X$ .

#### 4.5.2.1 $K$ is derived from UAK

In this case the authentication key is derived from UAK using a process B1 as illustrated in figure 4.5.



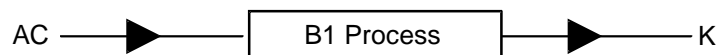
**Figure 4.5: Derivation of  $K$  from UAK**

The process B1 is defined as follows:

- $K[i] = UAK [i \text{ MODULO } LEN\_UAK], 0 \leq i \leq LEN\_K - 1$ .

#### 4.5.2.2 $K$ is derived from AC

In this case  $K$  is derived from AC using the process B1 as illustrated in figure 4.6.



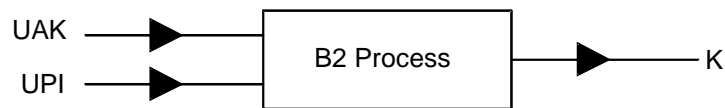
**Figure 4.6: Derivation of  $K$  from AC**

The process B1 is defined as follows:

- $K[i] = AC [i \text{ MODULO } LEN\_AC], 0 \leq i \leq LEN\_K - 1$ .

### 4.5.2.3 K is derived from UAK and UPI

In this case, K is derived from UAK and UPI using a process B2 as illustrated in figure 4.7.



**Figure 4.7: Derivation of K from UAK and UPI**

The process B2 is defined as follows:

- $K[i] = (UAK [i \text{ MODULO } LEN\_UAK] + UPI [i \text{ MODULO } LEN\_UPI]) \text{ MODULO } 2,$   
 $0 \leq i \leq LEN\_K - 1.$

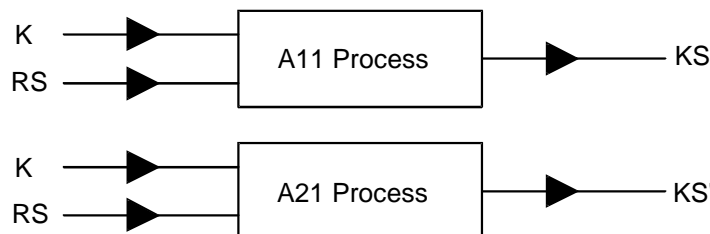
## 4.5.3 Authentication processes

### 4.5.3.0 General

Authentication of a PT and authentication of an FT each require the use of an authentication process. In each case the authentication process is subdivided into two processes. The first process is used for the derivation of the authentication session key KS or KS' respectively. The second process is used for the derivation of DCK and RES1 (XRES1) or of RES2 (XRES2) respectively.

### 4.5.3.1 Processes for the derivation of KS and KS'

KS is derived from K and RS using process A11. KS' is derived from K and RS using process A21. This is illustrated in figure 4.8.



NOTE 1: The value for K used as input to the A11 process may be different from the value input to the A21 process, however this will not be the usual practice.

NOTE 2: The value for RS used as input to the A11 process may be different from the value used as input to the A21 process. In the case of authentication type 2, this is mandatory. For authentication type 1 it is an optional practice.

NOTE 3: For processes based on DSAA2, RS has 128 bits ( $RS_{128}$ ).

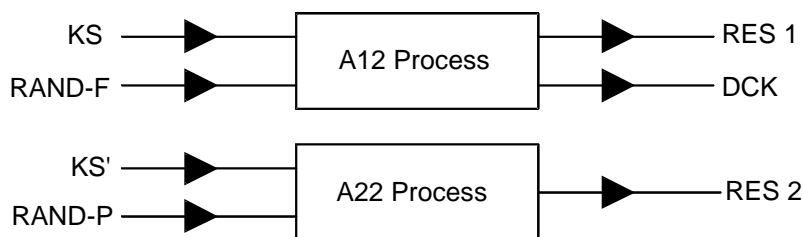
**Figure 4.8: Derivation of KS and KS'**

### 4.5.3.2 Processes for the derivation of DCK, RES1 and RES2

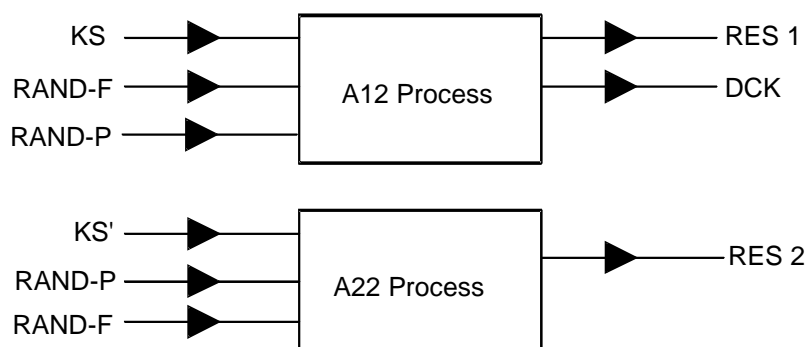
The processes A11 and A21 may be based on the use of an authentication algorithm as specified in clause 5.2. Either the DSAA, DSAA2 or an appropriate proprietary authentication algorithm, may be used. Requirements for the authentication algorithm needed for these processes are discussed in more detail in clause 5.2.

Alternatively, if the roaming features provided by the use of KS and KS' are not required, then a proprietary implementation may obtain KS and KS' directly from K (see clauses 5.2.1 and 5.2.2). In this case, the protocol element RS may be input directly to a proprietary A12 (or A22) process along with RAND\_F (respectively RAND\_P). For an example see clause G.3.

The values DCK and RES1 (XRES1) are derived from KS and RAND\_F (and RAND\_P if authentication type 2 is used) using a process A12. The value RES2 (XRES2) is derived from KS' and RAND\_P (and RAND\_F if authentication type 2 is used) using a process A22. This is illustrated in figures 4.9 and 4.10.



**Figure 4.9: Derivation of RES1, DCK and RES2 for type 1 authentication**



**Figure 4.10: Derivation of RES1, DCK and RES2 for type 2 authentication**

The processes A12 and A22 are based on the use of an authentication algorithm as specified in clause 5.3. Either the DSAA (annex H), the DSAA2 (annex L) or an appropriate proprietary authentication algorithm, may be used. Requirements for the authentication algorithm needed for these processes are discussed in more detail in clause 5.3.

When DSAA2 algorithm is used, then authentication type 2 (figure 4.10) shall be used.

As an alternative, if the roaming features provided by the use of session authentication keys is not required, and if a proprietary process is used to derive KS (and KS') directly from K, then the parameter RS may also be input directly to a proprietary A12 (respectively A22) process. For an example see clause G.3.

#### 4.5.4 Key stream generation

As part of the encryption process, a Key Stream Generator is used to generate a key stream from a CK (of type DCK or SCK) and an initialization value IV, as depicted in figure 4.11.



**Figure 4.11: Key stream generation**

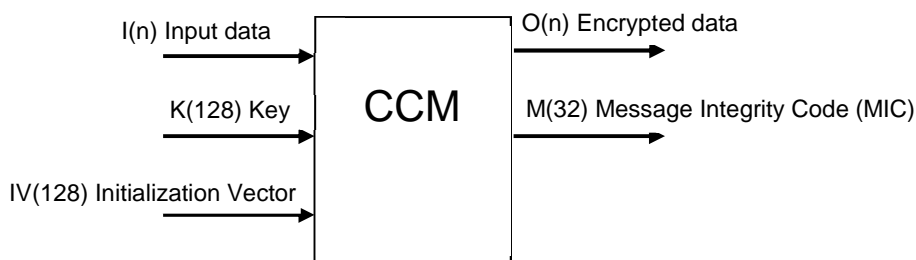
The way in which the key stream is used to encrypt data is described in clause 6.4.4. Either the DSC (annex J), the DSC2 (annex M) or an appropriate proprietary algorithm, may be used for the KSG.

CK should have a length of 128 bits if DSC2 algorithm is used and 64 bits in any other case.

See clause 4.4.3.3 to see how the CK is adapted (extended or truncated) when an authentication algorithm generating a CK of different size is used.

## 4.5.5 CCM Authenticated Encryption

DECT supports Authenticated Encryption based on CCM (Counter with CBC-MAC) (as defined by IETF RFC 3610 [11]), using 128 bit block ciphers based on AES [10]. CCM operates at DECT DLC level, and is used by DLC service LU14. The security process model of CCM Authenticated Encryption is shown in figure 4.12.



**Figure 4.12: CCM Authenticated Encryption**

The Key for CCM authenticated encryption may be generated, for instance, by DSAA2, which is also based on AES with block size 128 and, therefore, has compatible security strength.

The operation of the CCM authenticated encryption and the different parts and values of the Initialization Vector are described in clause 6.6.2.

By cryptographic reasons, the Keys used in CCM should be used only once for each value of the nonce (bytes 1-13 of the Initialization Vector). In practice, it means that a new key should be generated and used, each time the CCM sequence is reset (see clause 6.6.2.3).

In addition to that, the key used for CCM authenticated encryption, should not be reused by any other security process. This means that if both CCM and DSC or DSC2 MAC encryption are used, different keys should be used.

The generation and storage of Keys for CCM is described in clause 6.2.3 of the present document.

The activation of new keys for CCM is performed as described in clause 6.3.7.

## 4.6 Combinations of security services

### 4.6.0 Service combinations and related considerations

The purpose of this clause is to specify the envisaged combinations of security services. These combinations of services are as follows:

- (S1) authentication of a PT;
- (S2) authentication of a PT followed by authentication of an FT;
- (S3) authentication of a PT followed by data confidentiality using the DCK;
- (S4) authentication of a PT followed by authentication of an FT followed by data confidentiality using the DCK;
- (S5) data confidentiality using a SCK;
- (S6) data confidentiality using the DCK established at the last instance of S3 or S4.

The following points should be noted concerning these combinations of services:

- 1) whenever S2 or S4 are applied, it is envisaged that the same authentication key K will be used for authentication of the PT and authentication of the FT. This is however not necessary, and an exception may be where user authentication is applied;
- 2) S6 can only be used in those networks which have a management facility which can maintain the current DCK for a PT, and ensure that it is made available to an FT whenever needed;

- 3) the above list of combinations of services does not distinguish between user authentication and authentication of the PT. If the application requires user authentication, then this is achieved by applying S1, S2, S3 or S4 with the authentication key (for authentication of the PT) derived using the UPI as explained in clause 4.3.5;
- 4) combinations S2 and S4 provide direct mutual authentication. Indirect mutual authentication is provided by S3, S5 or S6. In the case of S3, the principle underlying authentication of the FT is that if the FT does not know the authentication key K then it is unable to derive the DCK, so is unable to encrypt or decrypt data sent to or received from the PT. In the case of S5 or S6, the principle underlying the mutual authentication is that both parties shall know the cipher key in order to successfully encrypt and decrypt data;
- 5) it is not foreseen that S4 will be used in practice as the confidentiality service using the DCK from PT authentication, is sufficient to achieve FT authentication;
- 6) although other combinations of services are possible, for some combinations, certain security weaknesses may result (see annex E);
- 7) data confidentiality using the DCK (S6) provides a continuous form of authentication because only that PT which knows the DCK can correctly encrypt and decrypt the call.

## 4.6.1 Combinations of security algorithms

### 4.6.1.0 General

The purpose of this clause is to specify restrictions to possible combinations of security algorithms.

#### 4.6.1.1 Limitations related to ciphering algorithms

When using the ciphering algorithm DSC2, the following provisions should be followed:

- 1) DSC2 should not be used in combination with DSAA.
- 2) The authentication algorithms to be used in combination with DSC2 should be able to provide DCK of 128 bits.
- 3) Use of static cipher keys combined with DSC2 should be avoided.

NOTE: It is envisioned that the normal case would be the use of DSC2 combined with DSAA2 authentication. Nevertheless, the use of DSC2 combined with non-DECT or proprietary authentication algorithms may be possible in some scenarios. In such a case, the proprietary authentication algorithm should be at least as strong as DSC2. Otherwise the DSC2 protection may be jeopardized.

# 5 Algorithms for security processes

## 5.1 Background

### 5.1.0 General

In this clause, the security processes defined in clause 4.5 are specified. Two sorts of processes will be described: the derivation of authentication session keys (see clause 5.2) and the authentication and derived cipher key generation processes (see clause 5.3).

These processes will be described in terms of a common algorithm A, the input and output parameters of which are specified in clause 5.1.1. Throughout this clause the following notation is used:

- $LEN\_X$  denotes the bit-length of value X;
- $X[i]$  denotes the i'th bit of X for  $0 \leq i \leq LEN\_X - 1$ ; where
- $X[0]$  denotes the least significant bit of X.



## 5.1.1 A algorithm

### 5.1.1.0 A algorithm, general

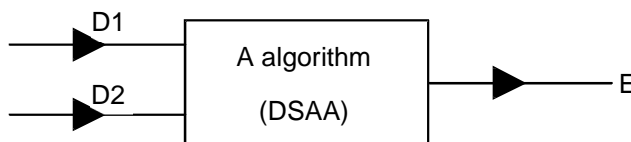
The term "A algorithm" is used to denote the algorithm used for implementing the authentication processes A11, A12, A21 and A22.

DECT supports the following specific algorithms for implementing the authentication processes:

- DECT Standard Authentication Algorithm (DSAA): this algorithm is only available on a restricted basis. See annex H.
- DECT Standard Authentication Algorithm #2 (DSAA2): this algorithm was introduced with revision V2.4.1 of the present document and is publicly available. See annex L.
- Proprietary algorithms.

### 5.1.1.1 A algorithm, DSAA based (A-DSAA)

The A Algorithm, DSAA version has two inputs D1 and D2 and one output E. A block diagram of the A-DSAA algorithm is given in figure 5.1.



**Figure 5.1: A algorithm, version DSAA (A-DSAA)**

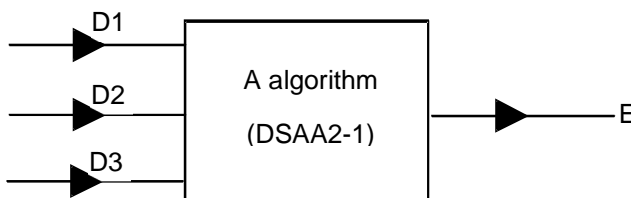
This algorithm is only available on a restricted basis (see annex H). Requirements for the DSAA algorithm are summarized in annex D. Annex G illustrates a proprietary implementation by describing how the GSM SIM may be used to provide authentication in DECT.

### 5.1.1.2 A algorithm, DSAA2 based (A-DSAA2)

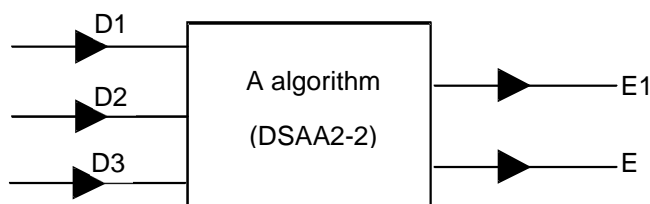
The A Algorithm, DSAA2 version (A-DSAA2) has three inputs D1, D2 and D3 and one or two outputs, named E and E1.

This algorithm is defined in annex L. The algorithm has two variants, DSAA2-1 and the DSAA2-2. The operation of DSAA2-1 is defined in clause L.2.1 and the operation of DSAA2-2 is defined in clause L.2.2.

A block diagram of the A algorithm, variants DSAA2-1 and DSAA2-2 algorithm is given in figures 5.2 and 5.3.



**Figure 5.2: A algorithm, version DSAA2, variant DSAA2-1**



**Figure 5.3: A algorithm, version DSAA2, variant DSAA2-2**

The following mappings shall be done:

- Authentication process A11 shall be implemented with algorithm DSAA2-1.
- Authentication process A12 shall be implemented with algorithm DSAA2-2.
- Authentication process A21 shall be implemented with algorithm DSAA2-1.
- Authentication process A22 shall be implemented with algorithm DSAA2-2.

### 5.1.1.3 A algorithm, proprietary

In the terminology used by the present document, a "proprietary algorithm" is defined as any other authentication algorithm different to the standard algorithms defined by the present document.

Therefore the "proprietary algorithm" may be either a real proprietary algorithm or an algorithm defined by non DECT specification.

The present document provides mechanisms for using such external or proprietary algorithms for implementation of the A processes. The terminology "proprietary algorithm" is used by the present document for referring to the external or proprietary algorithm in general. Nevertheless, in messages and information elements definitions, it is possible to allocate specific codes for well-known external authentication algorithms when needed.

## 5.2 Derivation of session authentication key(s)

### 5.2.1 A11 process

In the A11 process the session authentication key KS is derived from the authentication key K and the value RS.

For the derivation of KS the A algorithm is used, the input and output of which are specified in clause 5.1.1. The inputs of the A algorithm are set as:

If DSAA or a proprietary algorithm is used:

- $D1[i] = K[i \text{ MODULO } LEN\_K], 0 \leq i \leq LEN\_D1 - 1$ ; and
- $D2[i] = RS[i \text{ MODULO } LEN\_RS], 0 \leq i \leq LEN\_D2 - 1$ .

If DSAA2 is used:

- $D1[i] = K[i \text{ MODULO } LEN\_K], 0 \leq i \leq LEN\_D1 - 1$ ;
- $D2[i] = RS1[i \text{ MODULO } LEN\_RS1], 0 \leq i \leq LEN\_D2 - 1$ ; and
- $D3[i] = RS2[i \text{ MODULO } LEN\_RS2], 0 \leq i \leq LEN\_D3 - 1$ .

The output KS of the A11 process is now defined in terms of the output E of the A Algorithm by:

For DSAA and DSAA2:

- $KS[i] = E[i \text{ MODULO } LEN\_E], 0 \leq i \leq LEN\_KS - 1$ .

For proprietary algorithms:

The same equation may be used or, if authentication session keys are not required, then process A11 may optionally be replaced by:

- $KS [i] = K [i \text{ MODULO } LEN\_K], 0 \leq i \leq LEN\_KS - 1.$

NOTE: For a discussion of how roaming may be managed in this case see clause 7.2.

## 5.2.2 A21 process

In the A21 process the session authentication key  $KS'$  is derived from the authentication key  $K$  and the value  $RS$ .

For the derivation of  $KS'$  the A Algorithm is used, the input and output of which is specified in clause 5.1.1. The inputs of the A Algorithm are set as:

If DSAA or a proprietary algorithm is used:

- $D1 [i] = K [i \text{ MODULO } LEN\_K], 0 \leq i \leq LEN\_D1 - 1;$  and
- $D2 [i] = RS [i \text{ MODULO } LEN\_RS], 0 \leq i \leq LEN\_D2 - 1.$

If DSAA2 is used:

- $D1 [i] = K [i \text{ MODULO } LEN\_K], 0 \leq i \leq LEN\_D1 - 1;$
- $D2 [i] = RS1 [i \text{ MODULO } LEN\_RS1], 0 \leq i \leq LEN\_D2 - 1;$  and
- $D3 [i] = RS2 [i \text{ MODULO } LEN\_RS2], 0 \leq i \leq LEN\_D3 - 1.$

The output  $KS'$  of the A21 process is now defined in terms of the output  $E$  of the A algorithm by:

For DSAA only:

- $KS' [i] = (E [i \text{ MODULO } LEN\_E] + i) \text{ MODULO } 2, 0 \leq i \leq LEN\_KS' - 1.$

For DSAA2 only:

- $KS' [i] = E [i \text{ MODULO } LEN\_E], 0 \leq i \leq LEN\_KS' - 1.$

For proprietary algorithms:

The proprietary algorithm may decide to use the equation for DSAA, the equation for DSAA2 or, if authentication session keys are not required, then process A21 may optionally be replaced by:

- $KS' [i] = (K [i \text{ MODULO } LEN\_K] + i) \text{ MODULO } 2, 0 \leq i \leq LEN\_KS' - 1.$

NOTE 1: When DSAA algorithm is used, it is envisaged that for most instances of authentication of the PT and authentication of the FT the same  $K$  and  $RS$  values will be input to the A11 and A21 processes. In this case the outputs  $KS$  and  $KS'$  will be different. This is important since, as discussed in annex E, if  $KS$  and  $KS'$  are the same then it may be possible to launch certain reflection attacks. If different values for  $K$  or  $RS$  are used, then the system operator is advised to avoid pairs for which  $KS$  and  $KS'$  are the same.

NOTE 2: When DSAA2 algorithm is used, different values of  $RS_{128}$  are always used as inputs for A11 and A21 processes. Therefore,  $KS$  and  $KS'$  will be always different.

## 5.3 Authentication and cipher key generation processes

### 5.3.1 A12 process

In the A12 process the value  $RES1$  ( $XRES1$ ) and the derived cipher key  $DCK$  are derived from the session key  $KS$  and the value  $RAND\_F$ .

For the derivation of RES1 (XRES1) and DCK the A algorithm is used, the parameters of which are specified in clause 5.1.1.

The inputs of the A algorithm are set as:

If DSAA or a proprietary algorithm is used:

- $D1 [i] = KS [i \text{ MODULO } LEN\_KS], 0 \leq i \leq LEN\_D1 - 1$ ; and
- $D2 [i] = RAND\_F [i \text{ MODULO } LEN\_RAND\_F], 0 \leq i \leq LEN\_D2 - 1$ .

If DSAA2 is used:

- $D1 [i] = K [i \text{ MODULO } LEN\_K], 0 \leq i \leq LEN\_D1 - 1$ ;
- $D2 [i] = RAND\_F [i \text{ MODULO } LEN\_RAND\_F], 0 \leq i \leq LEN\_D2 - 1$ ; and
- $D3 [i] = RAND\_P [i \text{ MODULO } LEN\_RAND\_P], 0 \leq i \leq LEN\_D3 - 1$ .

The outputs RES1 and DCK of the A12 process are now defined in terms of the outputs E (and E1 if DSAA2) of the A algorithm by:

If DSAA or a proprietary algorithm is used:

- $RES1 [i] = E [i \text{ MODULO } LEN\_E], 0 \leq i \leq LEN\_RES1 - 1$ ; and
- $DCK [i] = E [(i + LEN\_RES1) \text{ MODULO } LEN\_E], 0 \leq i \leq LEN\_DCK - 1$ .

If DSAA2 is used:

- $RES1 [i] = E1 [i \text{ MODULO } LEN\_E1], 0 \leq i \leq LEN\_RES1 - 1$ ; and
- $DCK [i] = E [i \text{ MODULO } LEN\_E], 0 \leq i \leq LEN\_DCK - 1$ .

### 5.3.2 A22 process

In the A22 process the value RES2 (XRES2) is derived from the session key  $KS'$  and the value  $RAND\_P$ .

For the derivation of RES2 (XRES2) the A algorithm is used, the parameters of which are specified in clause 5.1.1. The inputs of the A algorithm are set as:

If DSAA or a proprietary algorithm is used:

- $D1 [i] = KS' [i \text{ MODULO } LEN\_KS'], 0 \leq i \leq LEN\_D1 - 1$ ; and
- $D2 [i] = RAND\_P [i \text{ MODULO } LEN\_RAND\_P], 0 \leq i \leq LEN\_D2 - 1$ .

If DSAA2 is used:

- $D1 [i] = KS' [i \text{ MODULO } LEN\_KS'], 0 \leq i \leq LEN\_D1 - 1$ ; and
- $D2 [i] = RAND\_P [i \text{ MODULO } LEN\_RAND\_P], 0 \leq i \leq LEN\_D2 - 1$ ;
- $D3 [i] = RAND\_F [i \text{ MODULO } LEN\_RAND\_F], 0 \leq i \leq LEN\_D3 - 1$ .

The output RES2 of the A22 process is now defined in terms of the output E (or E1 if DSAA2 is used) of the A algorithm by:

If DSAA or a proprietary algorithm is used:

- $RES2 [i] = E [i \text{ MODULO } LEN\_E], 0 \leq i \leq LEN\_RES2 - 1$ .

If DSAA2 is used:

- $RES2 [i] = E1 [i \text{ MODULO } LEN\_E1], 0 \leq i \leq LEN\_RES2 - 1$ .

## 5.4 CCM algorithm

DECT supports Authenticated Encryption based on CCM (Counter with CBC-MAC) (as defined by IETF RFC 3610 [11]). The algorithm for CCM, which uses 128 bit AES [10] block ciphers, is defined in annex N of the present document. See also clause 6.6 on DLC layer procedures.

The DCK destined to be used by CCM is generated by process A12 exactly in the same way as other DCK. In order to preserve the security strength of CCM, the DSAA2 algorithm or an external or proprietary algorithm of at least 128 bits and sufficient strength should be used.

The differentiation between Keys destined to CCM or to regular MAC encryption (DSC or DSC2) is done by means of the parameter <Cipher-Key-nr.>. This parameter is transmitted in the <<Auth-type>> IE (used in the {AUTHENTICATION-REQUEST} message) and also in the <<Cipher-Info>> IE.

---

# 6 Integration of security

## 6.1 Background

The purpose of this clause is to specify how the security features defined in previous clauses are to be integrated into the DECT air interface. Four aspects of integration are considered. Clause 6.2 is concerned with the first aspect: the way in which the cryptographic keys are associated with the DECT identities defined in ETSI EN 300 175-6 [6]. Clause 6.3 deals with the NWK layer messages and procedures needed for the authentication exchanges. Clause 6.4 is concerned with the MAC layer procedures and messages needed for encryption of data over the air interface. The contents of clause 6.3 and clause 6.4 are previewed in more detail in clause 6.3.1 and clause 6.4.1 respectively. Finally, clause 6.5 is concerned with the NWK layer messages needed to enable a PT and an FT to identify the security algorithms and keys which will be used. Clause 6.5 also contains a description of the NWK layer control of ciphering and a method for over-the-air allocation of a UAK from a manually distributed AC.

## 6.2 Association of keys and identities

### 6.2.1 Authentication key

#### 6.2.1.0 General

The purpose of this clause is to specify how an authentication key K is associated with the DECT identities defined in ETSI EN 300 175-6 [6]. The association is specified for each of the three ways of deriving K identified in clause 4.5.2.

**NOTE:** Throughout it is assumed that a PT has one or more IPUIs, each of which relates it to a subscription. Such an IPUI is the unique identity of the PT for within a particular system or network of FPs. This system may, for example, consist of a simple residential FP, the collection of all FPs belonging to a public access service system or the set of FPs for a single PBX system.

#### 6.2.1.1 K is derived from UAK

If K is derived from a UAK, then in the PT the UAK is associated with either an IPUI or an IPUI/PARK pair. All authentication information required by fixed system elements (for example FT, local network, management centre for subscriber's home network) is associated with either the IPUI or the IPUI/PARK pair.

Included in this authentication information is data which may be provided to an FT or a visited network in a roaming scenario. For example, the subscriber's home network may provide a visited network with pairs (RS,KS) and (RS,KS') for the particular PT, in which case these pairs would be associated with either the IPUI or the IPUI/PARK pair.

**NOTE 1:** For a discussion of the authentication data which may be transferred in a roaming environment, see clause 7.2.

There may be more than one UAK associated with a single IPUI or IPUI/PARK pair. In this case the different keys are numbered as described in clause 6.5.5.

NOTE 2: The <<AUTH-TYPE>> information element identifies if the key is related to the IPUI or IPUI/PARK pair, see also clause 6.5.5. For more information on the coding of the <<AUTH-TYPE>> see ETSI EN 300 175-5 [5].

### 6.2.1.2 K derived from AC

If K is derived from an AC, then in the PT the AC is associated with either an IPUI or an IPUI/PARK pair.

All authentication information required by fixed system elements (for example FT, local network) is associated with either the IPUI or the IPUI/PARK pair.

There may be more than one AC associated with a single IPUI or IPUI/PARK pair. In this case the different keys are numbered as described in clause 6.5.5.

NOTE: The <<AUTH-TYPE>> information element identifies if the key is related to the IPUI or IPUI/PARK pair, see also clause 6.5.5. For more information on the coding of the <<AUTH-TYPE>> see ETSI EN 300 175-5 [5].

### 6.2.1.3 K derived from UAK and UPI

If K is derived from an UAK and an UPI, then in a PT the UAK and the manually entered UPI are both associated with either an IPUI or an IPUI/PARK pair.

All authentication information required by fixed system elements (for example FT, local network, management centre for the subscriber's home network) is associated with either the IPUI or the IPUI/PARK pair.

A single UAK may have at most one UPI associated with it.

NOTE: The <<AUTH-TYPE>> information element identifies if the key is related to the IPUI or IPUI/PARK pair, see also clause 6.5.5. For more information on the coding of the <<AUTH-TYPE>> see ETSI EN 300 175-5 [5].

## 6.2.2 Cipher keys

If the cipher key DCK is derived as part of authentication of the PT as described in clause 4.3.4.1, then it is always associated with the same identity as the authentication key from which it is derived.

There may be more than one DCK associated with a single IPUI or IPUI/PARK pair. In this case the keys are numbered as described in clause 6.5.5 (see note 1).

If the DCK is used as a default cipher key, then the FP has to store it including its default cipher key index. It is valid during the validity of the registration. The PP has to store at least the last assigned DefCK and its default cipher key index during the validity of the registration. The default cipher key index numbering is described in clause 6.5.5.

If the cipher key SCK is a static key, as described in clause 4.3.4.2, then it is associated with either an IPUI or an IPUI/PARK pair.

There may be more than one SCK associated with a single IPUI/PARK pair. In this case the keys are numbered as described in clause 6.5.5.

NOTE 1: Allowing a number of different DCKs, which will be updated at different times, could result in management problems for the fixed network. In most applications it is envisaged that one DCK per IPUI is sufficient.

NOTE 2: The <<CIPHER-INFO>> information element identifies if the key is related to the IPUI or IPUI/PARK pair, see also clause 6.5.5. For more information on the coding of the <<CIPHER-INFO>> see ETSI EN 300 175-5 [5].

CK should have a length of 128 bits if DSC2 algorithm is used and 64 bits in any other case.

See clause 4.4.3.3 to see how the CK is adapted (extended or truncated) when an authentication algorithm generating a CK of different size is used.

## 6.2.3 Cipher keys for CCM

### 6.2.3.0 General

Keys used by CCM (Counter with CBC-MAC) authenticated encryption (see clauses 4.5.5 and 6.6) should be Derived Cipher Keys (DCK) of 128 bits and are generated by a PT authentication process, in the same way as DCKs intended for encryption at MAC level. In order to preserve the security strength of CCM, the DSAA2 algorithm or an external or proprietary algorithm of at least 128 bits and sufficient strength should be used.

The differentiation between Cipher Keys destined to CCM or to regular MAC encryption during the Authentication procedures is done by means of a different value of the <Cipher Key nr> parameter. This parameter is used in the IEs <<Auth-type>> and <<Cipher-Info>>. The value to be used of this parameter shall be defined by the application profile.

NOTE: In ETSI TS 102 939-1 [i.6] (DECT ULE; Part 1), the used value of <Cipher Key nr> for CCM keys is "9".

### 6.2.3.1 Single use of the keys for CCM

In order to preserve the high security strength of CCM, the Keys used in CCM should be used only once for each value of the nonce (bytes 1-13 of the initialization vector, see clause 6.6.2). In practice, it means that a new key should be generated and used, each time the CCM is started or its sequence is reset.

In addition to that, the key used for CCM authenticated encryption, should not be reused by any other security process. This means that if both CCM and DSC or DSC2 MAC encryption are used, different keys (with different <Cipher Key nr.> should be used.

The second limitation is enforced by the use of a separate and dedicated value of the <Cipher Key nr.> for CCM keys.

In order to enforce the first security requirement the following provisions shall apply:

- When a DCK for CCM is generated by an authentication procedure, it shall be stored labelled as "unused".
- The first time such key is "used" by the CCM encoder, it shall be marked as "used". Due to the different mechanisms for transmission of the Key to lower layers (to DLC layer in this case) and insertion in the cipher engine, it is defined that the Key is considered used if and only if any segment of a message encrypted with such key has been transmitted over the air i/f (see notes 1 and 2).
- Any execution of a procedure for transferring a generated DCK to the CCM cipher engine, should found the "key" labelled as "unused" at both peers. Otherwise the procedure shall fail. The proper error code should be provided in order to allow both entities to perform the proper action (see note 3).

NOTE 1: Since the procedure for starting the CCM or for transferring the Key may be complex and include several operations, this rule means that in case of impossibility to complete the procedure due to errors, the key should not marked as "used" if there has not been transmission of any CCM segment. This allows re-attempting the procedure with the same key. However as soon as a single segment has been transmitted (even in one direction only), a new key should be generated to re-try the procedure.

NOTE 2: In case of errors, it may happen that the key ends marked as "used" in one side and remains "unused" at the other. In such a case, a new key should be generated to re-try any activation procedure. If it is attempted with the same key, the side that has the key marked as "used" should be in charge of aborting the procedure.

NOTE 3: The expected proper action is that the FT side starts a PT authentication requesting storage of a new DCK under the <Cipher Key nr.> used by CCM.

The following mechanisms for activating a CCM key are defined or foreseen:

- 1) CC NWK procedure of Virtual Call setup invoking the use of DLC service LU14 (EFREL-CCM). The starting of such call implies starting a DLC link with CCM encryption and a new DCK key, previously generated by MM, is used.
- 2) CC Service Change procedure for "NWK resume" of a Virtual Call or Permanent a Virtual Circuit (PVC) that uses service LU14 (EFREL-CCM). The NWK resume procedure implies starting a new DLC link with CCM encryption and a new DCK key, previously generated by MM, is used.

- 3) Possible NWK layer/MAC layer procedures for starting and stopping CCM encryption (for further study).
- 4) Possible NWK layer/MAC layer procedures for CCM re-keying (for further study).
- 5) Possible NWK layer/MAC layer procedures for resetting CCM sequence and re-keying (for further study).

Only the two firsts procedures are defined in the current revision of the present document. All others are for further study.

### 6.2.3.2 Cipher keys for CCM encryption of C/L multicast channels

For CCM encryption of C/L multicast channels, separate keys, different and independent from the CCM keys used for the encryption of the C/O channels, shall be used. Such secondary keys shall be generated by the FP using a random number generator and shall be transferred to the relevant PT (to the PTs subscribed to such multicast channel) by means of the procedures described in the clause 6.3.8 of the present document.

In order to preserve the security of the process, the NWK layer procedures used to transfer of the keys (clause 6.3.8) shall be always transmitted over encrypted links.

The clause 6.6.2.7.7 "Security provisions regarding the key" defines additional provisions and recommendations regarding the generation and use of the Cipher keys for CCM encryption of C/L multicast channels. Such provisions shall apply.

## 6.3 NWK layer procedures

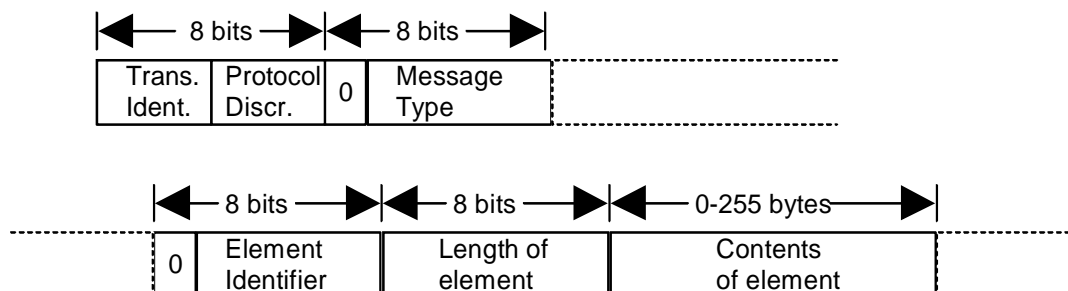
### 6.3.1 Background

The purpose of this clause is to specify the NWK layer procedures for the security mechanisms defined in clause 4.3.

The structure of the NWK layer messages is illustrated in figure 6.1, where the formats for the transfer of elements are depicted.

The NWK layer messages required for the authentication exchanges are specified in clause 6.3.2. Clause 6.3.3 contains a list of the sequences of messages which are sent for the specific authentication mechanisms defined in clauses 4.3.1, 4.3.2, 4.3.6 and 4.3.7 as well as a description of the processes carried out on each of the messages. Clause 6.3.4 identifies the need for primitives to establish the cipher key using the mechanisms defined in clause 4.3.4.

**NOTE:** This clause is concerned specifically with the NWK layer messages needed for the mechanisms defined in clause 4.3. Other NWK messages or information elements which are needed to identify the keys and algorithms to be used by these mechanisms are defined in clause 6.5.



**Figure 6.1: NWK Layer Message Structure (NLMS)**

For more information on NWK layer procedures, messages or information elements see ETSI EN 300 175-5 [5].

### 6.3.2 Authentication exchanges

The messages required for the authentication exchanges described in clauses 4.3.1, 4.3.2, 4.3.6 and 4.3.7 are defined in table 6.1. The sequence of messages for each of these exchanges is given in clause 6.3.3.



The contents of the elements in an AUTH\_MESSAGE depend upon whether the message is originated by a PT or an FT and the authentication type. The elements and their contents are specified for the originator of an AUTH\_MESSAGE in the column of table 6.1 headed AUTH\_ELEMENTS. Thus, for example, if {AUTHENTICATION-REQUEST} is originated by a PT, it contains the elements <<AUTH-TYPE>> and <<RAND>>, where the content of <<RAND>> is the parameter RAND\_P. If it is originated by an FT, then it contains <<AUTH-TYPE>>, <<RS>> and <<RAND>>, where the content of <<RAND>> is the parameter RAND\_F.

An eight bit code for the message type is given for each of the AUTH\_MESSAGES. The most significant bit is always 0. This is the bit depicted in figure 6.1. The coding given in figure 6.1 is taken from clause 7.4.5 of ETSI EN 300 175-5 [5]. In the event of any conflict between figure 6.1 and ETSI EN 300 175-5 [5], the coding in the latter shall be the prime source.

Only those messages which are needed for the authentication exchanges defined in clause 4.3 are listed. Moreover, the messages listed may contain additional elements which are not relevant for the security features specified in the present document. For full details of the messages and their contents the reader is referred to ETSI EN 300 175-5 [5].

**Table 6.1: Authentication Messages (AM)**

| AUTH_MESSAGE  | Message Type | AUTH_ELEMENTS                                      |  |
|---|--------------|--|--|
|   |              | PT   | FT   |
| {AUTHENTICATION-REQUEST}  | 01000000     | <<RAND>>(RAND_P)<br><<AUTH-TYPE>><br><<RES>>(RES1) | <<RS>> (RS or RS <sub>128</sub> )<br><<RAND>>(RAND_F)<br><<AUTH-TYPE>> |
| {AUTHENTICATION-REPLY}  | 01000001     | <<RES>>(RES1)<br><<RAND>>(RAND_P)                  | <<RS>> (RS or RS <sub>128</sub> )<br><<RES>>(RES2)<br><<RAND>>(RAND_F) |
| NOTE 1: The element <<RES>> (RES1) is only used in the PT originated {AUTHENTICATION-REQUEST} when this request is sent as part of the key allocation protocol defined in clause 6.5.6. It is therefore not relevant to the discussions in this clause. |              |  |  |
| NOTE 2: The element <<RAND>> (carrying either RAND_F or RAND_P) is only used in the {AUTHENTICATION-REPLY} when DSAA2 and authentication type 2 procedure are used (see clause 4.3).  |              |  |  |
| NOTE 3: <<RS>> carries RS <sub>128</sub> (128 bits) when DSAA2 and authentication type 2 procedures are used. Otherwise it carries RS (64 bits).  |              |  |  |

The AUTH\_ELEMENTS are listed in table 6.2. The elements <<RAND>>, <<RS>> and <<RES>> correspond to the parameters described in clause 4.4.2.

The element <<AUTH-TYPE>> is used to identify which keys and algorithms are to be used. This is described in clause 6.5.

The length of <<RAND>>, <<RS>> and <<RES>> may depend upon the A algorithm (see clause 5.1.1) which is used for the authentication processes. Table 6.2 specifies the exact length of the elements when the DSAA or DSAA2 are used.

When the DSAA is used, the following convention is adopted for the coding of the <<RAND>>, <<RS>> and <<RES>> information elements (the numbering of the octets in an information element and of the bits within an octet is as defined in ETSI EN 300 175-5 [5]):

- The most significant bit RAND[63] (respectively RES[31] and RS[63]) is represented by bit 8 of octet 3 (the first octet after the length indicator) of the information element.
- The least significant bit RAND[0] (respectively RES[0] and RS[0]) is represented by bit 1 of octet 10 (respectively octet 6 and octet 10) of the information element.

When the DSAA2 is used, the adopted convention is described as follows:

- The most significant bit RAND[63], RES[31] and RS[127] (carrying RS<sub>128</sub>) is represented by bit 8 of octet 3 (the first octet after the length indicator) of the information element.
- The least significant bit RAND[0] is represented by bit 1 of octet 10. The least significant bit RES[0] is represented by bit 1 of octet 6. The least significant bit RS[0] is represented by bit 1 of octet 18.

An eight bit code for the Element Identifier (EI) is given for each of the AUTH\_ELEMENTS. The coding given in table 6.2 is taken from clause 7.7.1 of ETSI EN 300 175-5 [5]. In the event of any conflict between table 6.2 and ETSI EN 300 175-5 [5], the coding in the latter shall be the prime source.

**Table 6.2: Authentication Elements (AE)**

| AUTH_ELEMENT  | Element Identifier | Length in bytes when DSAA is used | Length in bytes when DSAA2 is used |
|---------------|--------------------|-----------------------------------|------------------------------------|
| <<RAND>>      | 00001100           | 10                                | 10                                 |
| <<RES>>       | 00001101           | 6                                 | 6                                  |
| <<RS>>        | 00001110           | 10                                | 18                                 |
| <<AUTH-TYPE>> | 00001010           | 5                                 | 5                                  |

### 6.3.3 Authentication procedures

#### 6.3.3.1 Authentication of a PT type 1 procedure

This procedure shall be used when the DECT Standard Authentication Algorithm (DSAA) or a proprietary algorithm is used.

**Table 6.3: Authentication of a PT type 1 procedure**

| SEQUENCE | DESCRIPTION   |
|----------|---|
| 1        | FT obtains RS, RAND_F and XRES1                           |
| 2        | FT constructs {AUTHENTICATION-REQUEST} and sends it to PT |
| 3        | PT calculates RES1  |
| 4        | PT constructs {AUTHENTICATION-REPLY} and sends it to FT   |
| 5        | FT compares RES1 with XRES1                               |

If the values compared in sequence 5 are not equal, then the authentication procedure has failed.

#### 6.3.3.2 Authentication of an FT type 1 procedure

This procedure shall be used when the DECT Standard authentication Algorithm (DSAA) or a proprietary algorithm is used.

**Table 6.4: Authentication of an FT type 1 procedure**

| SEQUENCE | DESCRIPTION   |
|----------|---|
| 1        | PT generates RAND_P                                       |
| 2        | PT constructs {AUTHENTICATION-REQUEST} and sends it to FT |
| 3        | FT obtains RS, and value RES2                             |
| 4        | FT constructs {AUTHENTICATION-REPLY} and sends it to PT   |
| 5        | PT calculates XRES2 and compares it with RES2             |

If the values compared in sequence 5 are not equal, then the authentication procedure has failed.

### 6.3.3.3 Authentication of a PT type 2 procedure

This procedure shall be used when the DECT Standard Authentication Algorithm #2 (DSAA2) is used.

**Table 6.5: Authentication of a PT type 2 procedure**

| SEQUENCE | DESCRIPTION  |
|----------|--|
| 1        | FT obtains a random number $RS_{128}$ not used before  |
| 2        | FT generates KS by means of the authentication process A11   |
| 3        | FT generates a random value, RAND_F, used only once for this authentication process  |
| 4        | FT constructs {AUTHENTICATION-REQUEST} including the parameters RAND_F and $RS_{128}$ , and sends it to PT   |
| 5        | PT generates a random value, RAND_P, used only once for this authentication process  |
| 6        | PT checks if it has internally stored a KS associated to the received value of $RS_{128}$ . Otherwise, the PT executes authentication process A11 using K and the received $RS_{128}$ as inputs  |
| 7        | The PT executes the authentication process A12 using as inputs KS, RAND_F and RAND_P. It obtains the parameter RES1 (32 bits) and a new Derived Ciphering Key of 128 bits  |
| 8        | PT constructs {AUTHENTICATION-REPLY} including RAND_P and the computed RES1  |
| 9        | On receipt of RES1 and RAND_P, the FT uses the authentication process A12 to compute XRES1 (inputs: KS, RAND_F, RAND_P) and compares this value with the RES1 sent by the PT. If the two values are identical, the FT accepts the authenticity of the PT |
| 10       | As result of the execution of process A12, the FT obtains also a new value of the DCK of 128 bits  |

If the values compared in sequence 9 are not equal, then the authentication procedure has failed.

### 6.3.3.4 Authentication of an FT type 2 procedure

This procedure shall be used when the DECT Standard Authentication Algorithm #2 (DSAA2) is used.

**Table 6.6: Authentication of an FT type 2 procedure**

| SEQUENCE | DESCRIPTION  |
|----------|--|
| 1        | PT generates a "fresh" value RAND_P not used before  |
| 2        | PT constructs {AUTHENTICATION-REQUEST} including the parameter RAND_P and sends it to FT   |
| 3a<br>OR | FT obtains (computes or retrieves) the last value of KS generated by the process A21 (that will be named KS') and the associated value of $RS_{128}$ that was used for its generation (that will be named $RS_{128}'$ ) OR   |
| 3b       | FT generates a new value of $RS_{128}$ (that will be named $RS_{128}'$ ), executes process A21 using $RS_{128}'$ and K as inputs and generates KS (that will be named KS')   |
| 4        | FT generates a "fresh" value of random parameter RAND_F not used before  |
| 5        | FT computes the value of RES2 running authentication process A22 and using as inputs: RAND_F, RAND_P and KS'   |
| 6        | FT sends to the PT an {AUTHENTICATION-REPLY} message including the calculated value RES2 together with the parameters RAND_F and $RS_{128}'$ used in its generation  |
| 7        | PT independently calculates the value of XRES2 running process A22 and using KS', $RS_{128}'$ , RAND_P and RAND_F as inputs<br>In order to get KS', the PT may check if the received $RS_{128}'$ matches with the value of $RS_{128}'$ it has stored from last execution of process A21. In such a case, the PT may use the last stored value of KS'. Otherwise, it shall run process A21 with the received $RS_{128}'$ to produce a new KS' |
| 8        | If XRES2 is equal to the value RES2 received from the FT, the PT accepts the authenticity of the FT  |

If the values compared in sequence 8 are not equal, then the authentication procedure has failed.

### 6.3.4 Transfer of Cipher Key, CK

This clause refers to the transferring of the Cipher Key to be used by MAC encryption (cipher algorithms DSC or DSC2). For transferring a Derived Cipher Key (DCK) to be used by CCM, see clause 6.3.7.

NWK layer primitives required to establish the cipher key at the MAC layer are defined in clause 6.5.3.

If the size of the CK obtained from the authentication algorithm or derived using a B process is different to the size required by the KSG algorithm, the size adaptation (see clause 4.4.3.3) shall be done before transferring it to the MAC layer.

In systems where several algorithms are supported for the KSG, the primitive shall also carry identification of the KSG algorithm to be used.

### 6.3.5 Re-Keying

Re-keying is the change of the cipher key (derived cipher key or default cipher key) used for encryption during an ongoing call. For generating of a new derived cipher key, the Authentication of PP procedure is used. The actual change of the cipher key is performed on MAC layer and described in clause 6.4.

### 6.3.6 Encryption with Default Cipher Key

For the generation of a default cipher key, the Authentication of PP procedure is used. As soon as the default cipher key is generated between FP and PP, the MAC is able to establish encryption for the following connections directly after connection establishment without the necessity to exchange NWK messages prior to this. The used default cipher key is identified within the MAC procedures by use of the system wide unique default cipher key index as described in clause 6.4.

By means of the re-keying procedure, the used cipher key can be changed from default cipher key to another derived cipher key as soon as another derived cipher key is available.

### 6.3.7 Transfer of Cipher Key CK for CCM

#### 6.3.7.0 General

The CCM Derived Cipher Key, generated by a PT authentication process as described in clause 5.4, is transferred to the user entity (the CCM cipher engine, that operates at DLC layer, see clause 6.6) by any of the following mechanisms.

In any case, the provisions given in clause 6.2.3.1 to ensure the single use of the key shall apply for this transfer process.

#### 6.3.7.1 Transfer by Virtual Call setup CC procedure

When a Virtual call (or a circuit mode call), that uses DLC service LU14 is setup, then a DCK for CCM is automatically transferred between the NWK layer, MM entity to the CCM cipher engine that operates at DLC layer (U-plane).

NOTE: Invocation of LU14 may be done explicitly, at <<Call-Attributes>> IE, or via a Basic Service.

In order to identify the Key to be used, the IE <<Cipher-Info>> shall be added to the CC Call setup message.

The provisions given in clause 6.2.3.1 to ensure the single use of the key shall apply. It should be noted that a key is only considered "used" if the CC setup process has succeeded and at least one segment of CCM encrypted data has been transmitted by any of the peers. It means that in case of CC setup procedure failure the key is, in general, not considered "used" and can be reused in a re-attempt.

If the DCK for CCM is labelled "used" at any of the sides, the CC setup procedure shall fail. The error shall be reported using the code "No Cipher Key available" in the <<RELEASE-REASON>> IE.

### 6.3.7.2 Transfer using MM procedures for CCM re-keying and sequence reset

These procedures are for further study.

In absence of dedicated procedures, the resetting of CCM sequence and also the re-keying, may be performed by executing a complete Release (CC Release) of the call (or VC) that use CCM, and setting a new call. The CC setup procedure will automatically take from MM a DCK for CCM and will transfer it to the cipher engine as described in clause 6.3.7.1.

## 6.3.8 Transfer of Cipher Keys for CCM encryption of multicast channels

### 6.3.8.1 General

When CCM encryption is used for C/L downlink multicast channels (see clause 6.6.2.7), separate keys, different and independent from the CCM keys used for the encryption of the C/O channels, shall be used. Such secondary keys shall be generated by the FP using a random number generator and shall be transferred to the relevant PT (to the PTs subscribed to such multicast channel) by means of the procedures described in this present clause.

There are two normal case procedures:

- Multicast encryption parameter assignation procedure, FT initiated
- Multicast encryption parameter retrieval procedure, PT initiated

And the corresponding error case handling procedures are:

- FT initiated parameter assignation procedure - PT reject
- PT initiated parameter retrieval procedure - FT reject

In order to preserve the security of the process, the NWK layer procedures used to transfer of the keys (clause 6.3.8) shall be always transmitted over encrypted links.

### 6.3.8.2 Multicast encryption parameter assignation procedure, FT initiated

#### 6.3.8.2.0 General

This will be the normal assignation procedure when the FT subscribes a PT to a multicast channel.

This procedure will typically be executed immediately following the subscription of a PT to a multicast channel.

NOTE: In DECT ULE phase 2 (see ETSI TS 102 939-2 [i.7]) this is normally done during the initial Service Call, immediately after the <<ULE MAC CONFIGURATION>> allocation.

Typically, both the <<KEY>> and the CCM sequence number will be allocated. However they may be allocated in one or in two successive executions (it is a choice of the RFP).

Procedure is FT initiated.

The sequence shall be as follows:

| FT |  | PT |
|----|--|----|
|    |  |    |
|    | {MM-INFO-SUGGEST} info-type, instance#, key, sequence nr |    |
|    | ----->   |    |
|    |  |    |
|    | {MM-INFO-ACCEPT} info-type, instance#                    |    |
|    | <-----   |    |

Figure 6.1a: FT parameter assignation procedure

### 6.3.8.2.1 Transport of the security parameters

The encryption key (128 bits) shall be transported by the IE <<KEY>> in the {MM-INFO-SUGGEST} message. The CCM sequence number (48 bits) shall be transported by the IE <<RS>> in the same message. At least one of them (or both) should be transported by the message.

The security parameters shall not be inserted in the {MM-INFO-ACCEPT} message.

### 6.3.8.2.2 <<INFO TYPE>> coding

The IE <<INFO TYPE>> shall always include in both messages the code of the transferred parameter(s), which may be the key, the CCM sequence number or both.

The applicable parameters shall be:

|               |  |
|---------------|--|
| 0 1 1 0 1 0 1 | Cipher Key for CCM encryption of multicast channels          |
| 0 1 1 0 1 1 0 | CCM Sequence number for CCM encryption of multicast channels |

The instance indicator may also be coded inside the <<INFO TYPE>> IE. Absence of the instance indicator shall be interpreted as Instance #0. If used, it shall be inserted in both messages.

The applicable codes for the instance indicator are:

|               |                               |
|---------------|-------------------------------|
| 0 1 1 1 0 0 0 | Multicast channel instance #0 |
| 0 1 1 1 0 0 1 | Multicast channel instance #1 |
| 0 1 1 1 0 1 0 | Multicast channel instance #2 |
| 0 1 1 1 0 1 1 | Multicast channel instance #3 |
| 0 1 1 1 1 0 0 | Multicast channel instance #4 |
| 0 1 1 1 1 0 1 | Multicast channel instance #5 |
| 0 1 1 1 1 1 0 | Multicast channel instance #6 |
| 0 1 1 1 1 1 1 | Multicast channel instance #7 |

A maximum of one instance indicator shall be used.

Other parameters possible in the IE <<INFO TYPE>> are not needed and shall not be used.

## 6.3.8.3 Multicast encryption parameter retrieval procedure, PT initiated

### 6.3.8.3.0 General

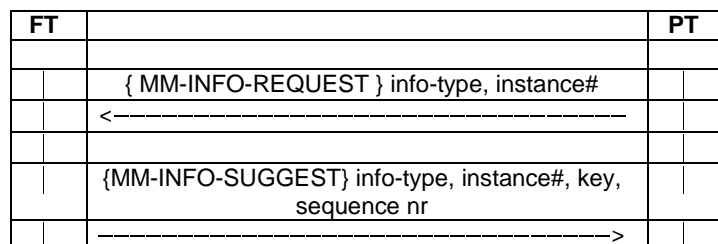
This is the exceptional procedure used by the PT when it "misses" either the Key or the CCM sequence.

It allows the PT to request either, the <<KEY>> or the CCM sequence number or both.

It is foreseen that the execution route may include an interleaved PT authentication procedure in order to check the legitimacy of the requesting PT. Messages for both MM processes are different and should not be any ambiguity.

Procedure is PT initiated.

The sequence shall be as follows:



**Figure 6.1b: PT parameter retrieval procedure**

The parameter retrieval procedure is supervised by the timer <MM\_info.1> in the PT. At the first expiry of the timer <MM\_info.1> the PT should retransmit the {MM-INFO-REQUEST} message. If the timer <MM\_info.1> expires a second time, the PT shall abort the procedure and release the transaction. Timer <MM\_info.1> may be restarted by the FT at any time by sending a <<TIMER-RESTART>> information element in a {MM-NOTIFY} message.

NOTE: The timer value for <MM\_info.1> is defined in the NWK standard (ETSI EN 300 175-5 [5], clause A.5).

### 6.3.8.3.1 Transport of the security parameters

The same provisions of clause 6.3.8.2.1 shall apply with the only difference that the {MM-INFO-SUGGEST} is now the closing message and that there is no {MM-INFO-ACCEPT} message.

### 6.3.8.3.2 <<INFO TYPE>> coding

The same provisions of clause 6.3.8.2.2 shall apply with the difference that there is no {MM-INFO-ACCEPT} message. The IE <<INFO TYPE>> in the {MM-INFO-REQUEST} shall be coded indicating the requested information identically to the {MM-INFO-SUGGEST}.

## 6.3.8.4 Error cases

### 6.3.8.4.1 FT initiated parameter assignment procedure - PT reject

This is the error case associated to the FT initiated procedure.

Procedure is FT initiated.

The sequence shall be as follows:

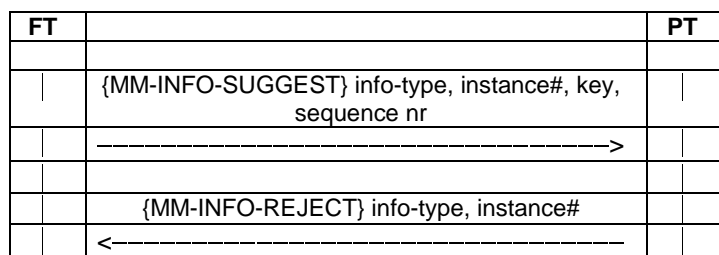


Figure 6.1c: FT parameter assignment procedure - reject

### 6.3.8.4.2 PT initiated parameter retrieval procedure - FT reject

This is the error case associated to the PT initiated procedure.

Procedure is PT initiated.

The sequence shall be as follows:

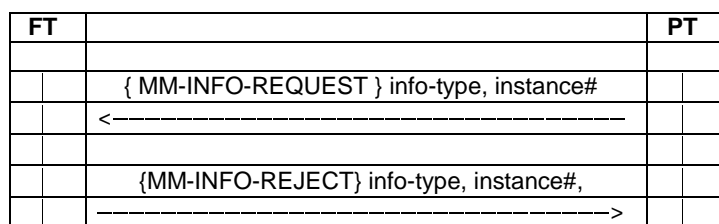


Figure 6.1d: PT parameter retrieval procedure - reject

#### 6.3.8.4.3 Coding of the {MM-INFO-REJECT} in the error cases

The IE <<INFO TYPE>> in the {MM-INFO-REJECT} shall be coded identically to the initiating message. No security parameters shall be carried in the {MM-INFO-REJECT}.

### 6.3.9 Transfer of Cipher Keys to Wireless Relay Stations (WRS)

#### 6.3.9.1 General

The encryption model used by CRFP type Wireless Relay Stations (WRS) (see ETSI EN 300 700 [12]) requires the transfer to the WRS of the encryption key used by the lower segment (see ETSI EN 300 700 [12], clause 4.4). Two procedures are defined to allow this transfer, FT initiated and WRS initiated procedures. FT initiated is assumed to be used in most cases. WRS initiated may be used in special cases when a PP (or another WRS) initiates a connection towards a WRS, and this last does not have a valid cipher key for the operation.

Both procedures may also be used to transfer Derived Cipher Keys (DCK) and Default Cipher Keys (DefCK).

Both procedures shall be initiated in local mode to the concerned WRS (with "local mode" meaning as defined in ETSI EN 300 700 [12], clause 7.4.17.1.2 and concerned WRS meaning the WRS whose key is to be transferred). Before initiating the procedure, the initiating node shall check that the link between the FP and the concerned WRS is in already in local mode, or otherwise, it shall initiate the transfer to local mode as defined in ETSI EN 300 700 [12], clause 7.4.17.2.

Only CRFP type WRSs are supported. Therefore, the terms "WRS" and "CRFP" are synonymous for the purposes of clause 6.3.9.

#### 6.3.9.2 Security considerations

In order not to undermine the security protection given by the present document the following provision applies:

The link where a cipher key is exchanged shall be encrypted. This encryption shall be done:

- If the transferred key is a DCK, by a DCK.
- If the transferred key is a DefCK, by either a DCK or by another DefCK.

If the security procedure "re-keying" (see clause 6.3.5) is used, then the provisions regarding the aging of the cipher keys described in clause 6.7.2.3.2 and the calculation of the initial age of the transferred key shall be observed.

#### 6.3.9.3 Indication of cipher key FT initiated procedure

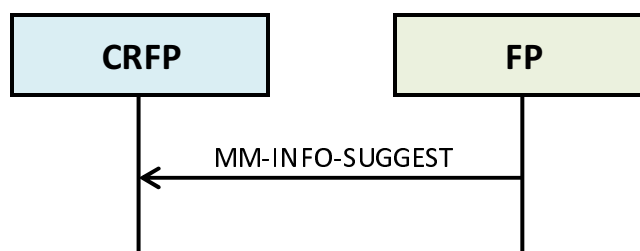
The FT initiated procedure is named "Indication of cipher key". This procedure may be initiated by the FP as soon as it knows that a WRS needs a cipher key (DCK or DefCK) for a given node.

NOTE 1: This will be the normal key transfer procedure and should be used when the FP updated a key (DCK or DefCK) into a PP (or in another WRS) by means of an authentication procedure and knows that such PP (or WRS) is connected to the FP by means of (an) intermediate WRS(s).

Procedure is FT initiated.



The sequence shall be as follows:



**Figure 6.1e: FT initiated Indication of WRS cipher key procedure**

NOTE 2: This procedure is a particular case of the "Parameter retrieval procedure" defined in ETSI EN 300 175-5 [5], clause 13.7. The provisions given in ETSI EN 300 175-5 [5] regarding primitives and timers can be observed as design guidelines.

The following parameters shall be used in the {MM-INFO-SUGGEST} message.

**Table 6.6a: Values used within the {MM-INFO-SUGGEST} message**

| Information element | Field within the information element | Standard values within the field/IE          | Normative action/comment  |
|---------------------|--------------------------------------|--|---|
| <<Info-type>>       |                                      |  |   |
|                     | <ext>                                | 0  |   |
|                     | <Parameter type>                     | 0100010'B                                    | CK transfer   |
| <<KEY>>             |                                      |  |   |
|                     | <Key type>                           | 10010000<br>10010001<br>10010100<br>10010101 | DCK for DSC<br>DCK for DSC2<br>Default Cipher Key for DSC<br>Default Cipher Key for DSC2  |
|                     | <Key>                                | Any  | In the case of Default Cipher Key (<Key type> 10010100'B or 10010101'B) the <Key> data field also includes the associated Default Cipher Key Index in 2 byte format. See ETSI EN 300 175-5 [5], clause 7.7.24 |

The <Key type> field indicates the ciphering algorithm to be used with the transferred key (either DSC or DSC2). This is important because the WRS does not know what ciphering algorithms are supported by the PT or indeed which ciphering algorithm the FP will select in its {CIPHER-REQUEST}. When using the key, the WRS shall use the specified ciphering algorithm (see also ETSI EN 300 700 [12], clause 7.7.6).

Multiple <<KEY>> Information Elements may be included in the {MM-INFO-SUGGEST} message by utilizing the repeat mechanism (see ETSI EN 300 175-5 [5], clause 7.5.6), i.e. by the inclusion of the <<REPEAT-INDICATOR>> specifying coding 1 "non-prioritized list" prior to the list of <<KEY>> Information Elements. This allows multiple keys to be transferred in the same message.

NOTE 3: This mechanism may be used for transferring to the same WRS a DCK plus a DefCK, or for transferring multiple DefCK.

When multiple <<KEY>> Information Elements are used, care should be taken to ensure that the maximum supported message length is not exceeded. If necessary, more cipher keys can be transferred by sending additional {MM-INFO-SUGGEST} messages.

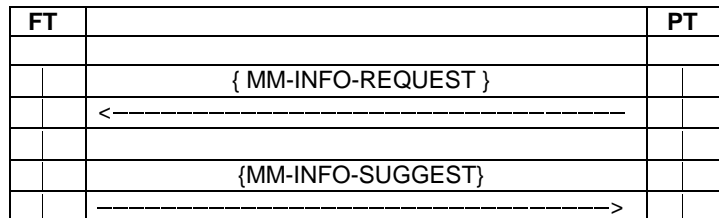
#### 6.3.9.4 Cipher key retrieval procedure. PT initiated

The PT initiated procedure may be used when a WRS has not received or "misses" a cipher key that it needs for a security procedure. This could happen, for instance, by failure of the previous FT initiated procedure (which is one-way and not acknowledged).

NOTE 1: It is foreseen that the execution route may include an interleaved PT authentication procedure in order to check the legitimacy of the requesting PT. Messages for both MM processes are different and should not be any ambiguity.

Procedure is PT initiated.

The sequence shall be as follows:



**Figure 6.1f: PT initiated cipher key retrieval procedure**

NOTE 2: This procedure is a particular case of the "Parameter retrieval procedure" defined in ETSI EN 300 175-5 [5], clause 13.7. The provisions given in ETSI EN 300 175-5 [5] regarding primitives and timers can be observed as design guidelines.

The following parameters shall be used in the {MM-INFO-REQUEST} message.

**Table 6.6b: Values used within the {MM-INFO-REQUEST} message**

| Information element | Field within the information element   | Standard values within the field/IE                  | Normative action/comment  |
|---------------------|--|--|---|
| <<Info-type>>       |  |  |   |
|                     | <ext>  | 0  |   |
|                     | <Parameter type>   | 0100010'B  | CK transfer   |
| <<KEY>>             |  |  |   |
|                     | <Key type>   | 10010000<br>10010001<br>10010100<br>10010101         | DCK for DSC<br>DCK for DSC2<br>Default Cipher Key for DSC (or unknown algorithm)<br>Default Cipher Key for DSC2 |
|                     | <Key>  | Carries only the Default Cipher Key index (2 octets) | See ETSI EN 300 175-5 [5], clause 7.7.24  |
| NOTE:               | To request a default cipher key, the requesting node should include the requested Default Cipher Key Index. The <Key type> shall be coded indicating the Ciphering algorithm, when known. However, this discrimination shall be ignored by the other peer that shall always code the algorithm type as the real algorithm associated to the given key. The recommended practice when the algorithm is unknown is coding the <Key type> in the request message with the DSC or unknown value ('10010100'B). |  |   |

Multiple <<KEY>> Information Elements may be included in the {MM-INFO-REQUESTTT} message by utilizing the repeat mechanism (see ETSI EN 300 175-5 [5], clause 7.5.6), i.e. by the inclusion of the <<REPEAT-INDICATOR>> specifying coding 1 "non-prioritized list" prior to the list of <<KEY>> Information Elements. This allows multiple keys to be requested in the same message.

NOTE 3: This mechanism may be used for requesting to the FP a DCK plus a DefCK, or for requesting multiple DefCK.

The following parameters shall be used in the {MM-INFO-SUGGEST} message.

Table 6.6c: Values used within the {MM-INFO-SUGGEST} message

| Information element | Field within the information element | Standard values within the field/IE          | Normative action/comment  |
|---------------------|--------------------------------------|--|---|
| <<Info-type>>       |                                      |  |   |
|                     | <ext>                                | 0  |   |
|                     | <Parameter type>                     | 0100010'B                                    | CK transfer   |
| <<KEY>>             |                                      |  |   |
|                     | <Key type>                           | 10010000<br>10010001<br>10010100<br>10010101 | DCK for DSC<br>DCK for DSC2<br>Default Cipher Key for DSC<br>Default Cipher Key for DSC2  |
|                     | <Key>                                | Any  | In the case of Default Cipher Key (<Key type> 10010100'B or 10010101'B) the <Key> data field also includes the associated Default Cipher Key Index in 2 byte format. See ETSI EN 300 175-5 [5], clause 7.7.24 |

The <Key type> field indicates the ciphering algorithm to be used with the transferred key (either DSC or DSC2). This is important because the WRS does not know what ciphering algorithms are supported by the PT or indeed which ciphering algorithm the FP will select in its {CIPHER-REQUEST}. When using the key, the WRS shall use the specified ciphering algorithm (see also ETSI EN 300 700 [12], clause 7.7.6).

Multiple <<KEY>> Information Elements may be included in the {MM-INFO-SUGGEST} message by utilizing the repeat mechanism (see ETSI EN 300 175-5 [5], clause 7.5.6), i.e. by the inclusion of the <<REPEAT-INDICATOR>> specifying coding 1 "non-prioritized list" prior to the list of <<KEY>> Information Elements. This allows multiple keys to be transferred in the same message.

NOTE 4: This mechanism may be used for transferring to the same WRS a DCK plus a DefCK, or for transferring multiple DefCK.

When multiple <<KEY>> Information Elements are used, care should be taken to ensure that the maximum supported message length is not exceeded. If necessary, more cipher keys can be transferred by sending additional {MM-INFO-SUGGEST} messages.

The parameter retrieval procedure is supervised by the timer <MM\_info.1> in the PT. At the first expiry of the timer <MM\_info.1> the PT should retransmit the {MM-INFO-REQUEST} message. If the timer <MM\_info.1> expires a second time, the PT shall abort the procedure and release the transaction. Timer <MM\_info.1> may be restarted by the FT at any time by sending a <<TIMER-RESTART>> information element in a {MM-NOTIFY} message.

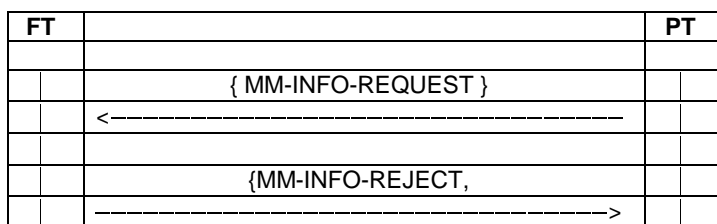
NOTE 5: The timer value for <MM\_info.1> is defined in the NWK standard (ETSI EN 300 175-5 [5], clause A.5).

### 6.3.9.5 Error cases

#### 6.3.9.5.1 PT initiated cipher key retrieval procedure - FT reject

In the PT initiated procedure, there is the option for the FT for rejecting the procedure. This may happen either due to error/inconsistency in the parameters supplied by the PP in the messages, or by error in a PP authentication that the FP may have executed interleaved with the procedure. The error is notified to the PT by sending a {MM-INFO-REJECT} as response.

The sequence shall be as follows:



**Figure 6.1g: PT initiated cipher key retrieval procedure - FT reject**

The following parameters shall be used in the {MM-INFO-REJECT} message.

**Table 6.6d: Values used within the {MM-INFO-REJECT} message**

| Information element | Field within the information element | Standard values within the field/IE | Normative action/comment |
|---------------------|--------------------------------------|-------------------------------------|--------------------------|
| <<Info-type>>       |                                      |                                     |                          |
|                     | <ext>                                | 0                                   |                          |
|                     | <Parameter type>                     | 0100010'B                           | CK transfer              |

## 6.4 MAC layer procedures

### 6.4.1 Background

The purpose of this clause is to specify all the MAC layer processes needed for the provision of data confidentiality over the air interface.

Clause 6.4.2 provides a brief overview of the MAC layer field structure to a level adequate for a specification of the encryption procedures. For more detail on the MAC layer, and for an explanation of terminology and notation, the reader is referred to ETSI EN 300 175-3 [3].

Clause 6.4.3 contains details of which data will be encrypted when operating in encryption mode.

Clause 6.4.4 details the encryption process itself and explains where this is placed within the MAC layer processes.

Clause 6.4.5 is concerned with initialization and synchronization of the encryption process, whilst clause 6.4.6 is concerned with the procedures for switching from clear to encrypted mode. Clause 6.4.7 contains details of how encryption is managed during handover.

Throughout clauses 6.4.2 to 6.4.7 the simple case of a full slot is assumed.

Clause 6.4.8 details the differences for a half and long slot specification. Clause 6.4.9 details the differences for a double slot specification. Clause 6.4.10 details the differences for multiple connections between a single FT and PT.

### 6.4.2 MAC layer field structure

The MAC layer field structure for a full slot physical channel using 2 level modulation, in as far as it is relevant for the encryption process, is illustrated in figure 6.2.



### 6.4.3 Data to be encrypted

When operating in encrypt mode, the following rules apply to the encryption of data in the various fields.

#### A-field:

- H field Never encrypted;
- T field Encrypted precisely when it is of type  $C_T$ ;
- RA field Never encrypted.

#### B-field:

- X-field Never encrypted.

#### Unprotected format:

All data bits encrypted (see figure 6.2).

#### Multisubfield protected format:

All data bits encrypted (see figure 6.3).

CRC bits (RB0...RBn) are not encrypted.

#### Singlesubfield protected format:

All data bits encrypted (see figure 6.4).

CRC bits (RB0) are not encrypted.

#### Constant- size subfield protected format:

All data bits encrypted.

CRC bits (RB0...RBn) are not encrypted.

#### E-type mux format (including E+U type):

It is handled as multisubfield protected format (see figure 6.3).

The payload in all subfields is encrypted irrespective of the carried logical channel.

CRC bits (RB0...RBn) are not encrypted.

#### $I_P$ encoded protected format ( $I_{PX}$ ):

The payload bits are encrypted before being processed by the coding engine. The two KSS are generated with the same length as for the unprotected format, but only the first bits of each segment are used. The number of bits that are used depends on the value of  $r$ .

NOTE: For any protected B-field format, the general rule is that the redundancy checks RB0, RB1, RB2 and RBn, as well as the redundancy added by the codec, are never encrypted.

### 6.4.4 Encryption process

For each pair of simplex bearers (duplex or double-simplex) transmitted in the same frame, the key stream generator, defined in clause 4.5.4, produces using CK and IV as input, a Key Stream of  $2m$  bits, where  $m$  is equal to:

- For full slot, 2-level modulation (D32 D-field structure),  $m = 360$ .
- In general,  $m = nt + nb$  with  $nt$  = number of bits of the Tail of the A-field (H and RA are not considered) and  $nb$  = total number of bits of the B-field (including CRCs).

The generated Key Stream is split on two Key Stream Segments (KSS) of size  $m$ . The first KSS produced by the generator is named  $S_F(N)$  and the second one is named  $S_P(N)$ .

The bits in each KSS are labelled  $KSS(0), \dots, KSS(m-1)$ , where  $KSS(0)$  is the first bit output from the generator and  $KSS(m-1)$  is the last.

For instance, for D-field structure D32, each segment consist of 360 consecutive bits ( $m = 360$ ). The bits of a KSS are labelled  $KSS(0), \dots, KSS(359)$ , where  $KSS(0)$  is the first bit output from the generator and  $KSS(359)$  is the last.

The bits in a KSS are used to encrypt the contents of the D-field.

The following description is given for D-field structure D32.

The encryption process is as follows:

- If T is of type  $C_T$ , the bits  $KSS(0), \dots, KSS(39)$  are XORed with the bits  $a_8, \dots, a_{47}$  from T, so that is  $KSS(0)$  is XORed with  $a_8$ , etc. The labelling of the bits in T is that specified in ETSI EN 300 175-3 [3].
- If T is not of type  $C_T$ , then bits  $KSS(0), \dots, KSS(39)$  are discarded.
- If the B-field has the unprotected format, the bits  $KSS(40), \dots, KSS(359)$  are XORed with the bits  $b_0, \dots, b_{319}$  from B, so that  $KSS(40)$  is XORed with  $b_0$ , etc.
- If the B-field has the singlesubfield protected format, the bits  $KSS(40), \dots, KSS(343)$  are XORed with the bits  $b_0, \dots, b_{303}$  from B0, so that  $KSS(40)$  is XORed with  $b_0$ , etc. Bits  $KSS(344), \dots, KSS(359)$  are not used. If the B-field has the multisubfield protected format, the bits  $KSS(40), \dots, KSS(103)$  are XORed with the bits  $b_0, \dots, b_{63}$  from B0, so that  $KSS(40)$  is XORed with  $b_0$ , etc.
- Bits  $KSS(104), \dots, KSS(119)$  are not used.
- Bits  $KSS(120), \dots, KSS(183)$  are XORed with the bits  $b_{80}, \dots, b_{143}$  from B1, so that  $KSS(120)$  is XORed with  $b_{80}$ , etc.
- Bits  $KSS(184), \dots, KSS(199)$  are not used.
- Bits  $KSS(200), \dots, KSS(263)$  are XORed with the bits  $b_{160}, \dots, b_{223}$  from B2, so that  $KSS(200)$  is XORed with  $b_{160}$ , etc.
- Bits  $KSS(264), \dots, KSS(279)$  are not used.
- Bits  $KSS(280), \dots, KSS(343)$  are XORed with the bits  $b_{240}, \dots, b_{303}$  from B3, so that  $KSS(280)$  is XORed with  $b_{240}$ , etc.
- Bits  $KSS(344), \dots, KSS(359)$  are not used.
- If the B-field has the encoded protected format (MAC service  $I_{PX}$ ) the KSS is used as follows:
  - bits  $KSS(40), \dots, KSS(m)$ , where  $m = 320 \times r + 39$ , with  $r$  the codec ratio, are XORed with the bits  $b_0, \dots, b_n$  of the B field payload, where  $n = 320 \times r - 1$ , before any stage of processing by the coding engine.
  - The remaining bits of the KSS (until  $KSS(359)$ ) are discarded.

The decryption operation is identical to the encryption operation. For this reason, the term encryption is used where decryption would be more precise. However, it will be apparent from the context whether the process is being used to encrypt clear data prior to transmission, or to recover clear data from received encrypted data.

The placement of the encryption process within the MAC layer is chosen so that when encryption is applied the redundancy checks RA, RB0, RB1, RB2 and RB3 are always computed on the encrypted data, as depicted in figure 6.5.

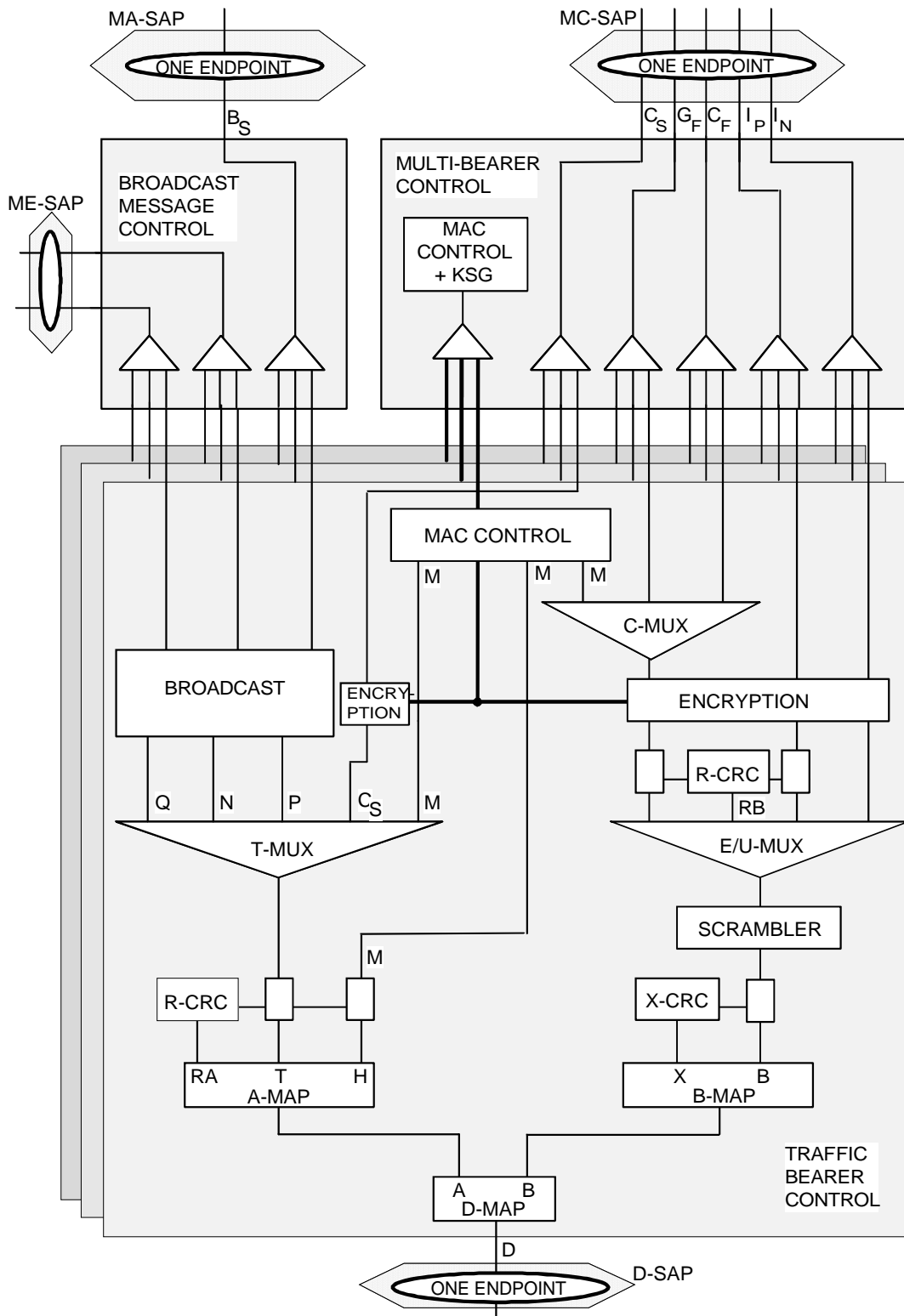


Figure 6.5: Placement of encryption



## 6.4.5 Initialization and synchronization of the encryption process

### 6.4.5.0 General

For a MAC connection, both the PT and the FT are initially in clear (non-encrypt) mode. Both switch to encrypt mode for a specific frame number and remain in this mode until switched back to clear mode for another specific frame number. The way in which the mode is switched is specified in clause 6.4.6.

The respective KSGs are initialized by the PT and the FT loading them with the same CK and initialization value IV.

#### 6.4.5.1 Construction of CK

The construction of CK is described in clause 4.3.4.

For KSG implementing DSC2 algorithm, the cipher key will always be represented by the binary vector CK[0], ..., CK[127], where CK[0] is the least significant bit.

For KSG implementing DSC or proprietary algorithms, the cipher key will always be represented by the binary vector CK[0], ..., CK[63], where CK[0] is the least significant bit.

#### 6.4.5.2 The Initialization Vector (IV)

The IV is a 35-bit value which is represented in the form IV[0], ..., IV[34] where IV[0] is the least significant bit. The IV value depends upon the bearer which is to be encrypted.

For basic connections it is defined as follows:

- Bits IV[0], ..., IV[3] correspond to the (interpolated) frame number, with IV[0] the least significant bit of the frame number; bits IV[4], ..., IV[27] correspond to the multiframe number, with IV[4] the least significant bit of the multiframe number; bits IV[28], ..., IV[34] are set to 0.

For advanced connections it is defined as follows:

- Bits IV[0], ..., IV[27] are defined as above.; bits IV[28], ..., IV[31] correspond to LBN\*, defined as (LBN\* = 15-LBN), with IV[28] the least significant bit of the LBN\*; bits IV[32], ..., IV[34] are set to 0.

The KSG is re-initialized for every frame which is to be encrypted. Thus, for a basic connection, if frames with numbers M, M + 1, ..., L are encrypted, then the cipher is initialized with M as IV for the first frame, M + 1 as IV for the second frame, ..., L as IV for the last frame in this sequence.

#### 6.4.5.3 Generation of two Key Stream segments

When a frame (with frame number N) is to be encrypted, a pair of consecutive key stream segments is generated. Thus if  $S_F(N)$  denotes the first KSS in the pair and  $S_P(N)$  denotes the second, then the concatenated sequence  $S_F(N), S_P(N)$  is 720 consecutive output bits of the KSG.

In duplex mode there are two D32-fields per frame, one of which is transmitted and one of which is received. When in encrypt mode, both D32-fields are encrypted. The order in which the key stream segments in a pair are used to encrypt and decrypt data in duplex mode is as follows:

- The segment  $S_F(N)$  is used by the FT to encrypt the D32-field transmitted to the PT in frame number N. The segment  $S_P(N)$  is used by the FT to decrypt the D32-field received from the PT in frame number N.
- The segment  $S_F(N)$  is used by the PT to decrypt the D32-field received from the FT in frame number N. The segment  $S_P(N)$  is used by the PT to encrypt the D32-field transmitted to the FT in frame number N.

The encryption process is specified in clause 6.4.4.

The PT and the FT rely upon synchronized frame numbering to maintain synchronization of their key stream segments.

For double simplex bearers the use of the two segments is as follows:

- The segment  $S_F(N)$  is used to encrypt and decrypt the first simplex bearer of the pair transmitted in frame number  $N$  and the segment  $S_P(N)$  is used to encrypt and decrypt the second simplex bearer of the pair.

## 6.4.6 Encryption mode control

### 6.4.6.1 Background

Switching a MAC connection from clear to encrypt mode and from encrypt to clear mode is achieved by an exchange of MAC layer T-field messages of type  $M_T$  (see clause 6.4.2). The relevant  $M_T$  messages shall be exchanged onto a duplex bearer of the connection.

The transmission of MAC layer T-field messages of type  $M_T$  is restricted. A PT may transmit such messages only in (the second half of) even numbered frames. An FT may transmit such messages only in (the first half of) odd numbered frames. Messages of type  $M_T$  are always unencrypted (see clause 6.4.3).

At the MAC layer switching from clear to encrypt mode and from encrypt to clear mode is always initiated by the PT.

The switching to the new mode (encrypted or clear) shall affect all the bearers of the connection. Additional bearers coming afterwards shall acquire the current mode at their establishment.

### 6.4.6.2 MAC layer messages

A MAC layer T-field message of type MT is defined to enable switching from clear to encrypt mode. The START.xxx message may be either with or without default cipher key index (see ETSI EN 300 175-3 [3], clause 7.2.5.7).

**START.REQ:** This is sent by the PT. It is transmitted in every even numbered frame until the receipt of a START.CONF or a timeout, or after STOP.CFM was sent by the PT during re-keying (see clause 6.4.6.5).

**START.CONF:** This is sent by the FT. It is transmitted in every odd numbered frame after receipt of a START.REQ and before receipt of a START.GRANT or a timeout.

**START.REJECT:** This is sent by the FT and is only allowed in case the START.REQ was sent with a default cipher key index. It is transmitted in every odd numbered frame after receipt of a START.REQ or a timeout (see figure 6.8).

**START.GRANT:** This is sent by the PT. It is transmitted in the even numbered frame immediately following the receipt of a START.CONF.

A MAC layer T-field message of type MT is defined to enable switching from encrypt to clear mode.

**STOP.REQ:** This is sent by the PT. It is transmitted in every even numbered frame until the receipt of a STOP.CONF or a timeout.

**STOP.CONF:** This is sent by the FT. It is transmitted in every odd numbered frame after receipt of a STOP.REQ and before receipt of a STOP.GRANT or a timeout.

**STOP.GRANT:** This is sent by the PT. It is transmitted in the even numbered frame immediately following the receipt of a STOP.CONF.

### 6.4.6.3 Procedures for switching to encrypt mode

#### 6.4.6.3.1 General

The procedures followed by a PT to switch from clear to encrypt mode are described in clauses 6.4.6.3.2 to 6.4.6.3.5 and illustrated in figures 6.6 to 6.9 (including some bad link cases).

#### 6.4.6.3.2 PT procedure for switching from clear to encrypt mode with a DCK

The PT receives a MAC\_ENC\_EKS-req primitive. This initiates a switching process.

The PT starts to transmit in the next even numbered frame (numbered  $2S$  in figure 6.9) an MT message with the start ciphering request command START.REQ. This is repeated in five successive even numbered frames or until the PT receives either a start ciphering confirm command START.CONF or a rejecting command START.REJECT (only allowed in case the START.REQ was sent with a default cipher key index). The START.CONF (or valid START.REJECT) event is reported with a MAC\_ENC\_EKS-cfm primitive. If the START.CONF (or valid START.REJECT) is not observed, the connection shall be released and the DLC layer is informed using the MAC\_DIS-ind primitive.

In the event that a valid START.REJECT was received after a START.REQ was sent with a default cipher key index, then the procedure ends and the connection will continue clear (see figure 6.8). However, higher-layer timers are still used to ensure the call is eventually encrypted properly (e.g. see ETSI EN 300 444 [i.8], clause 8.45.3).

The PT starts decryption of messages immediately the START.CONF message is received, i.e. the D32-field containing the START.CONF message is treated as being encrypted. All B-field and CT channel data received by the PT after sending the first request and before receiving the first START.CONF is ignored.

NOTE: The PT may begin decryption from frame  $2S + 1$  but all resultant output is ignored until the START.CONF is observed.

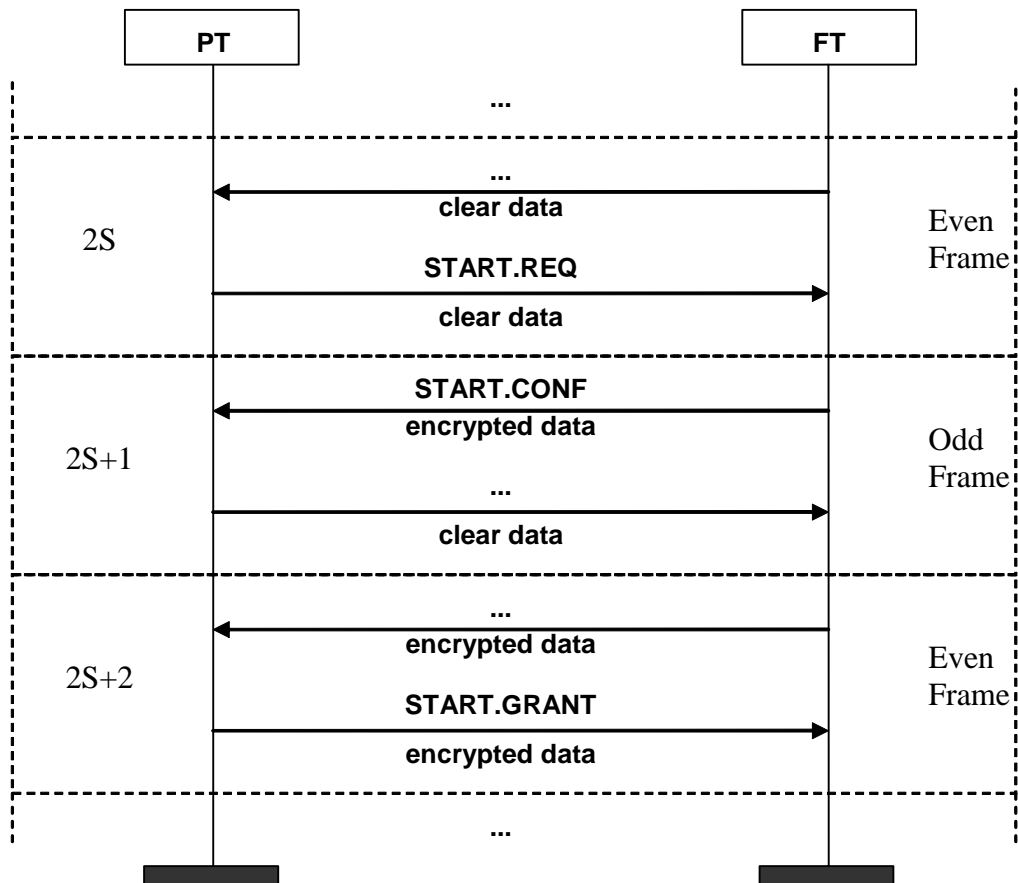
The PT acknowledges receipt of the START.CONF by the transmission of a START.GRANT in the next even numbered frame. Encryption of transmitted data begins with this START.GRANT frame and continues in all subsequent frames.

#### 6.4.6.3.3 FT procedure for switching from clear to encrypt mode with a DCK

The FT receives a START.REQ from the PT in frame  $2S$  for some  $S$ . The FT switches immediately into encrypt mode and starts transmission of START.CONF commands in frame  $2S + 1$ . This command is repeated in all successive odd numbered frames until  $2S + 11$  or until the FT receives a START.GRANT. If a START.GRANT is not received the connection is released and the DLC layer is informed using the MAC\_DIS-ind primitive.

All B-field and CT channel data received by the FT after (but not including) frame  $2S + 1$  is ignored until the START.GRANT is observed. Frame  $2S + 1$  is received in clear. Decryption of data begins immediately on receipt of the frame including the START.GRANT. The START.GRANT is reported with a MAC\_ENC\_EKS-ind primitive.

NOTE: Decryption may begin at the FT from frame  $2S + 2$  but the resultant output is ignored until the START.GRANT is observed.



**Figure 6.6: Encryption start - good link example**

#### 6.4.6.3.4 PT procedure for switching from clear to encrypt mode with a Default Cipher Key (DefCK)

The procedure is analogous to the "PT procedure for switching from clear to encrypt mode with a DCK" defined in clause 6.4.6.3.2 with the difference that the START Ciphering with Default Cipher Key messages (REQ, CONF and GRANT) are used. The cipher key index shall be included in all START messages.

The encryption algorithm to be used with the DefCK is the one predefined for the chosen DefCK (defined at time of generation).

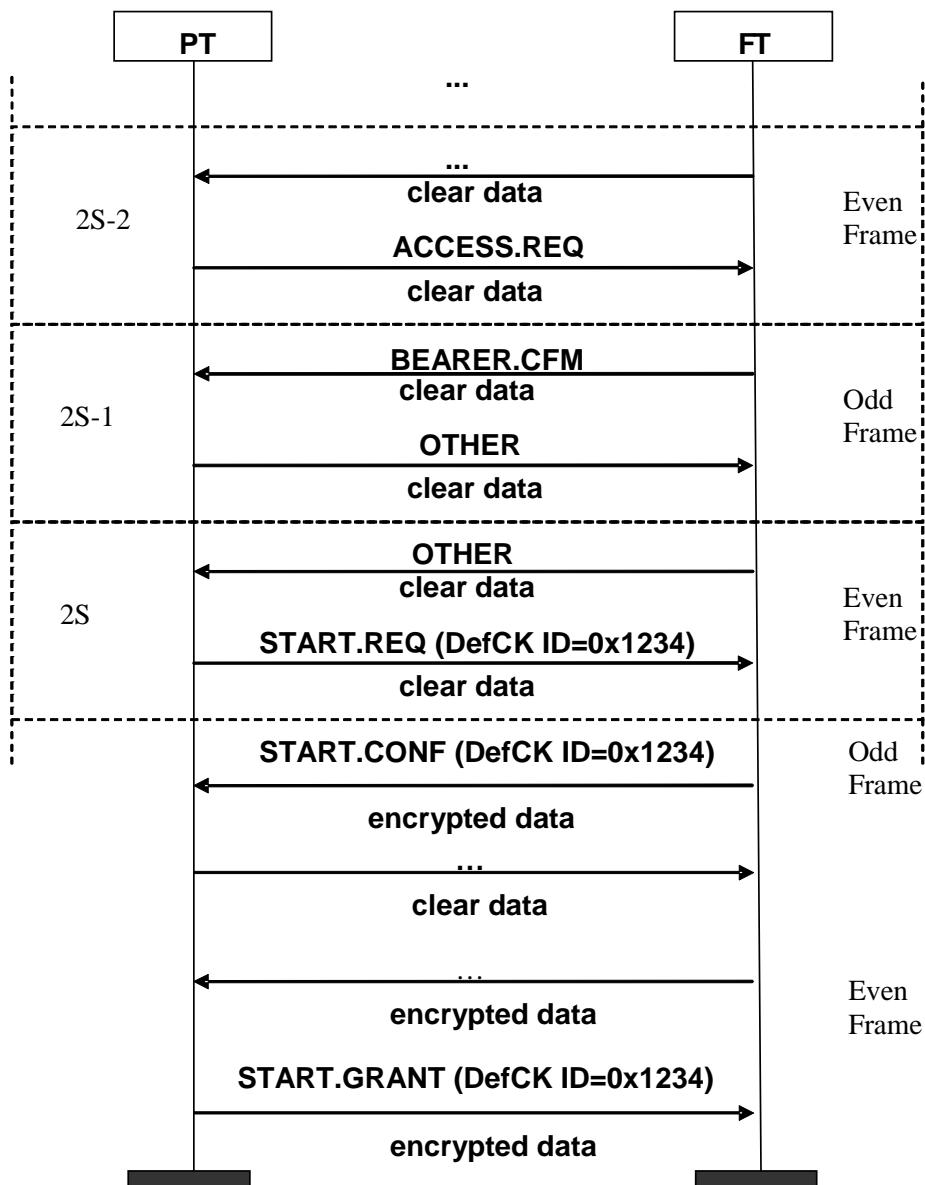
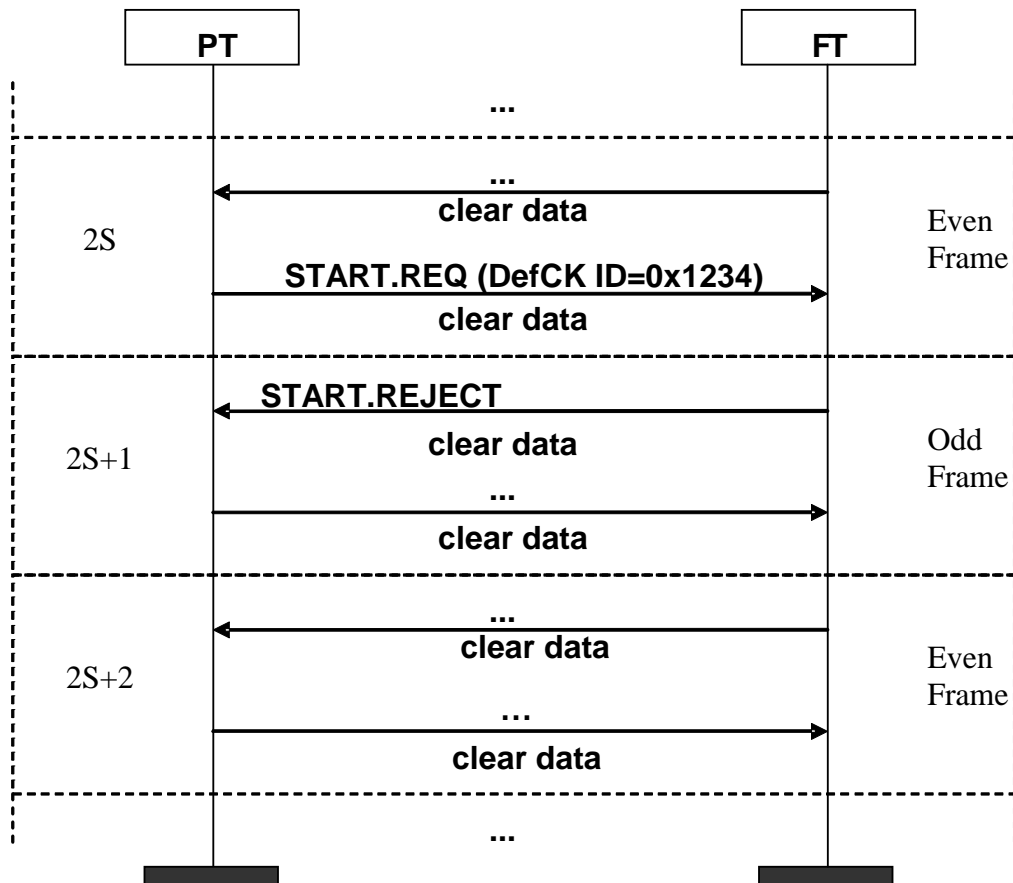


Figure 6.7: Encryption start immediately after bearer setup with default cipher key index - good link example



**Figure 6.8: Encryption start with default cipher key index and rejected by FT-good link example**

#### 6.4.6.3.5 Error handling - poor link

Figure 6.9 shows an example of example the PT procedure for switching from clear to encrypt mode with a DCK executed over a poor link with several error cases and their resolution.

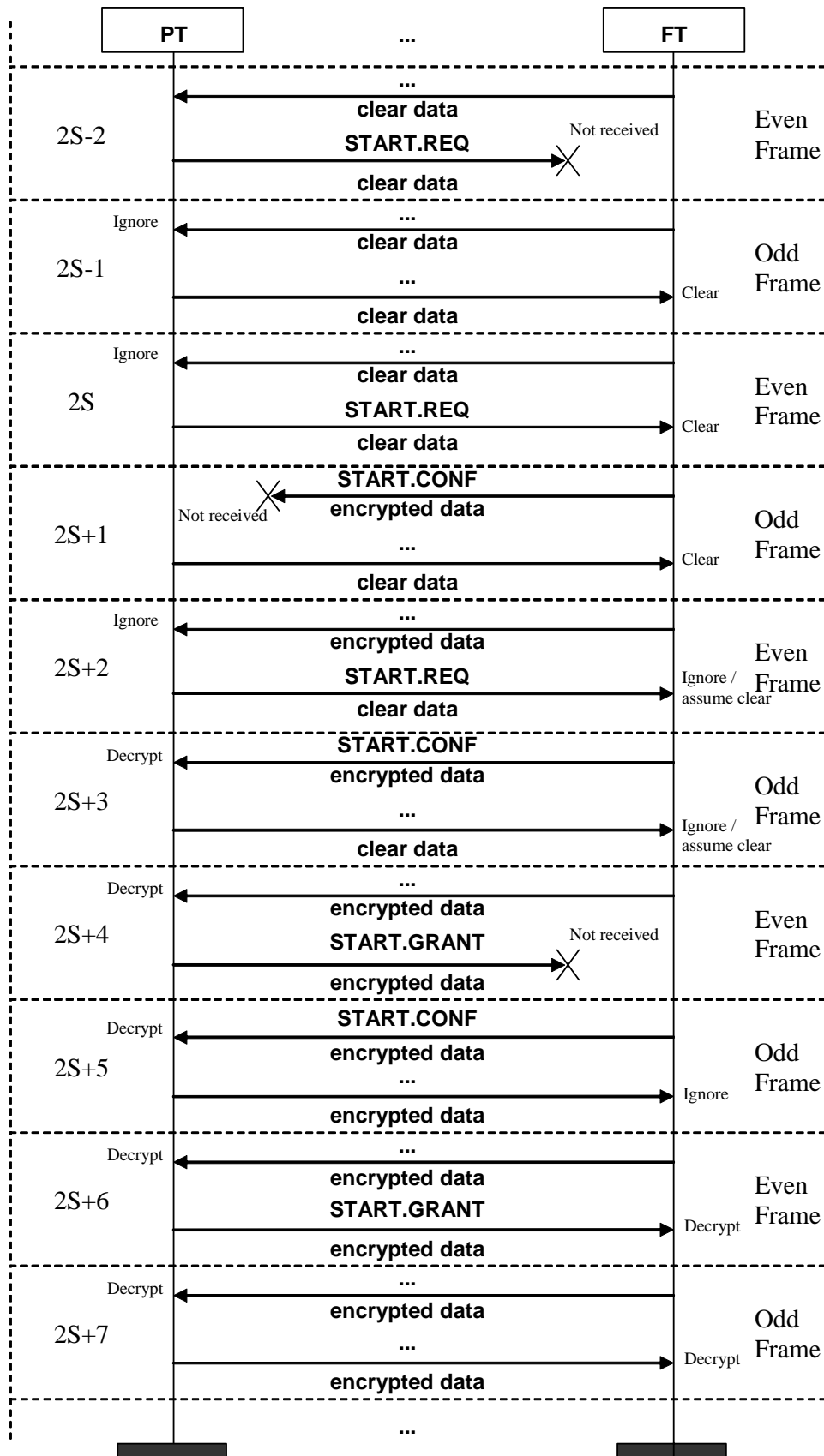


Figure 6.9: Encryption start - poor link example

#### 6.4.6.4 Procedures for switching to clear mode

This feature is provided to facilitate handover with interrupted ciphering (see clause 6.4.7). Encryption should be resumed once the handover is complete. The procedures for switching to clear mode are illustrated in figure 6.10.

##### **PT procedure for switching from encrypt to clear mode**

The PT receives a MAC\_ENC\_EKS-req primitive. This initiates a switching process.

The PT starts to transmit in the next even numbered frame an  $M_T$  message with the stop ciphering request command STOP.REQ. This is repeated in five successive even numbered frames or until the PT receives a stop ciphering confirm command STOP.CONF. The STOP.CONF event is reported with a MAC\_ENC\_EKS-cfm primitive. If the STOP.CONF is not observed, the connection is released and the DLC layer is informed using the MAC\_DIS-ind primitive.

The PT stops decryption of messages immediately the STOP.CONF message is received, i.e. the D-field containing the STOP.CONF message is treated as being clear. All B-field and  $C_T$  channel data received by the PT after sending the first request and before receiving the first STOP.CONF is ignored.

NOTE 1: Decryption may stop from frame  $2S + 1$  (in figure 6.12) but all resultant output is ignored until the STOP.CONF is observed.

The PT acknowledges receipt of the STOP.CONF by the transmission of a STOP.GRANT in the next even numbered frame. Transmission of clear data begins with this STOP.GRANT frame and continues in all subsequent frames.

##### **FT procedure for switching from encrypt to clear mode**

The FT receives a STOP.REQ from the PT in frame  $2S$  for some  $S$ . The FT switches immediately into clear mode and starts transmission of STOP.CONF commands in frame  $2S + 1$ . This command is repeated in all successive odd numbered frames until  $2S + 11$  or until the FT receives a STOP.GRANT. If a STOP.GRANT is not received the connection is released and the DLC layer is informed using the MAC\_DIS-ind primitive.

All B-field and  $C_T$  channel data received by the FT after (but not including) frame  $2S + 1$  is ignored until the STOP.GRANT is observed. Frame  $2S + 1$  is received encrypted. Decryption of data stops immediately on receipt of the frame including the STOP.GRANT. The STOP.GRANT is reported with a MAC\_ENC\_EKS-ind primitive.

NOTE 2: Decryption may be stopped at the FT from frame  $2S + 2$  but the resultant output is ignored until the STOP.GRANT is observed.



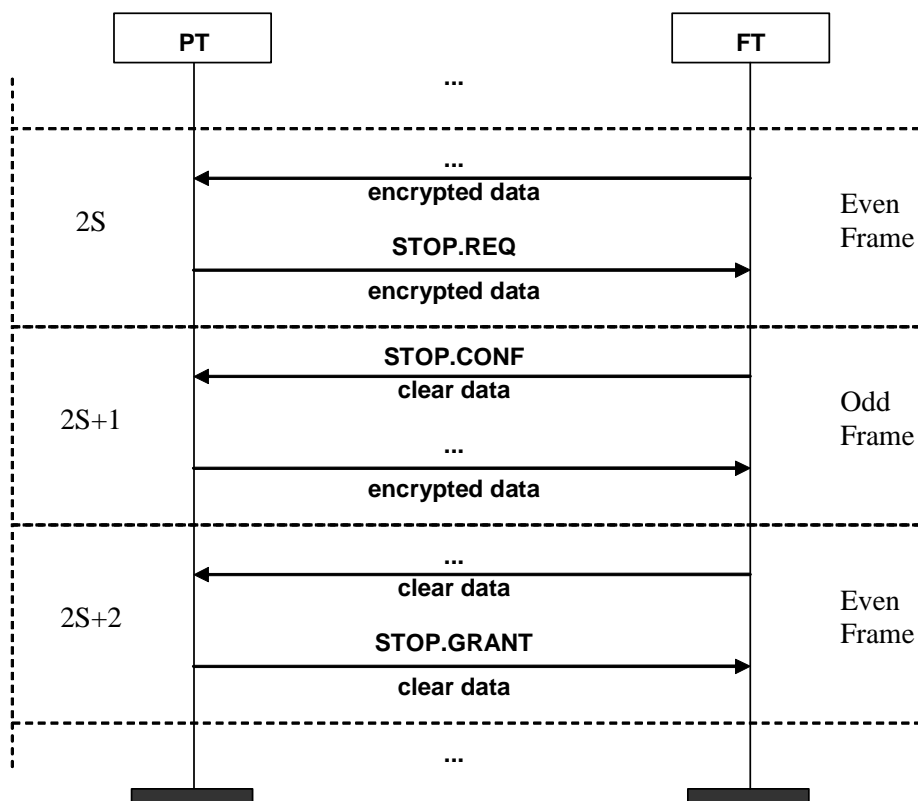


Figure 6.10: Encryption stop - good link example

## 6.4.6.5 Procedures for re-keying

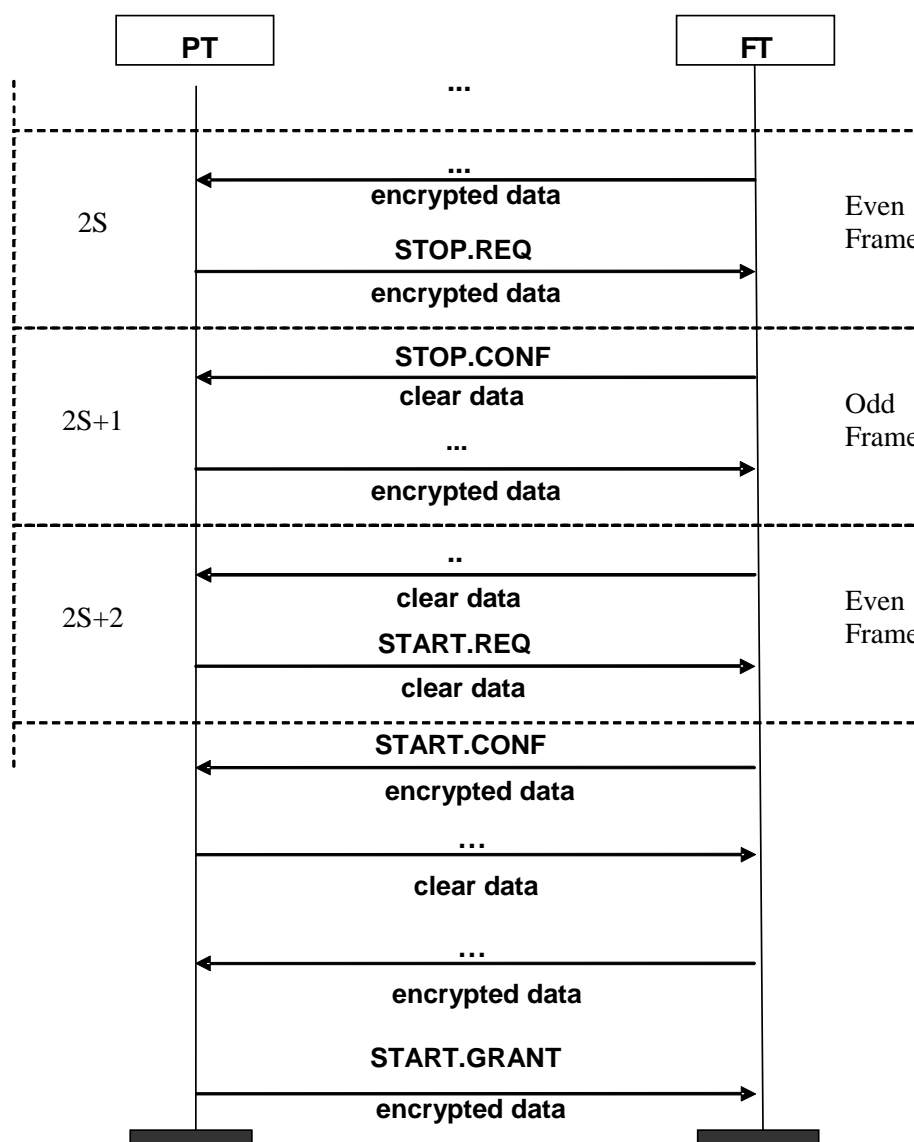
### 6.4.6.5.1 Re-keying to a DCK

The procedure is provided to change the DCK during an encrypted call.

The procedure may also be used for changing from a Default Cipher Key (DefCK) to a DCK.

This is the re-keying procedure used in normal operation.

The procedure is similar to the procedure of clause 6.4.6.3 with the difference that there is already an encrypted call established. In order to change the DCK, the PT sends a STOP.REQ (see clause 6.4.6.4). After receiving STOP.CFM, the PT sends immediately the START.REQ to restart encryption with the newly activated DCK.



**Figure 6.11: Encryption change - good link example**

#### 6.4.6.5.2 Re-keying to a DefCK

This procedure allows changing the key used in the encryption process from a DCK to a DefCK.

This is an exceptional procedure intended for some special operation cases where a fast change of cipher keys is required. It is provided to facilitate some handover cases.

NOTE 1: It should be assumed that in most cases the FT involved in this procedure will be a CRFP.

The procedure is analogous to the re-keying procedure described in clause 6.4.6.5.1 with the difference that the START messages (REQ, CFM and GRANT) will be replaced by the START with Default Cipher Key messages, including the desired cipher key index.

The Cipher Key index is generally chosen by the PT. See also the procedure described in clause 6.4.6.5.3.

The cipher algorithm to be used with the DefCK is the one predefined for the chosen DefCK (indicated by the cipher key index). Since each DefCK has a cipher algorithm predefined (defined at time of generation), this procedure can also change the encryption algorithm.

This procedure is only defined as PT initiated. A related procedure FT initiated (clause 6.4.6.5.3) may be used by the FT to trigger the PT procedure.

NOTE 2: Due to the security considerations this procedure is intended to be used only in special cases as part of specific procedures (such as handovers). Such cases should be specifically indicated in other clauses of the present document or in other DECT specifications.

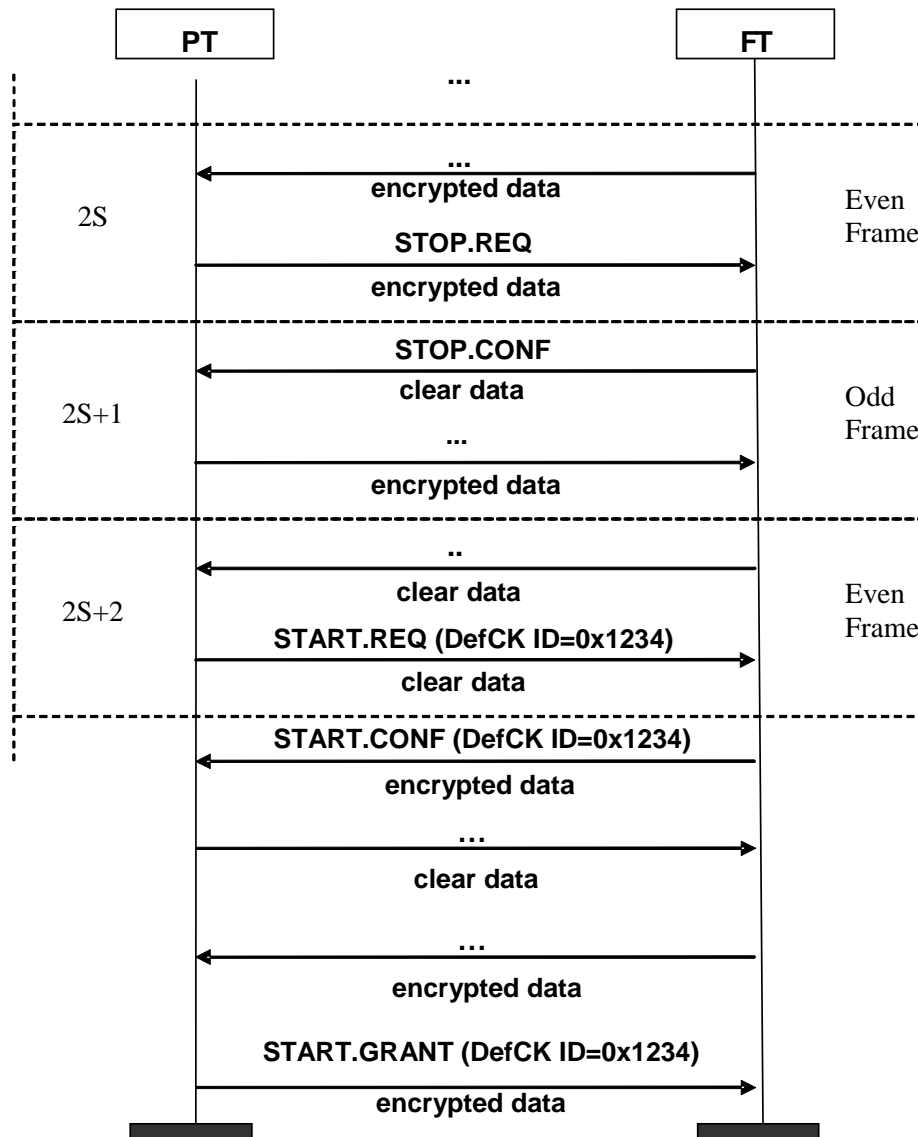


Figure 6.11a: Encryption change to a DefCK

#### 6.4.6.5.3 FT Indication of re-keying to a DefCK

The FT may send to the PT a request to initiate the procedure of re-keying to a DefCK. This may be of interest in certain special cases. For instance, during inter-cell bearer handover (including cases with WRS) where the key was not known, and a change of cipher algorithm was required.

NOTE: It should be assumed that in most cases the FT will be a WRS.

The procedure is implemented by sending FT to PT the message START.REQ with default cipher key.

The expected PT response is either the initiation of the procedure for re-keying to a DefCK (clause 6.4.6.5.2) in the next frame, or simply ignoring the procedure.

Due to the security considerations of the procedure, it is intended that PT will only obey the command in certain cases related to operations (such as handovers) where the procedure may have sense. Such cases should be specifically indicated in other clauses of the present document or in other DECT specifications. In any other case the PT shall ignore the command.

If the FT does not receive reply to the START.REQ message (STOP.REQ sent by the PT), it shall repeat the message in all successive odd numbered frames until  $2S + 11$  or until it receives the STOP.REQ from the PT.

The field cipher key index in the START message may be used to code the desired DefCK or may be coded with a conventional value in cases where it is intended that the PT choose the key. The conventional value may also be used for discriminating the intended cipher algorithm. The convention for such coding will be defined in other DECT specifications.

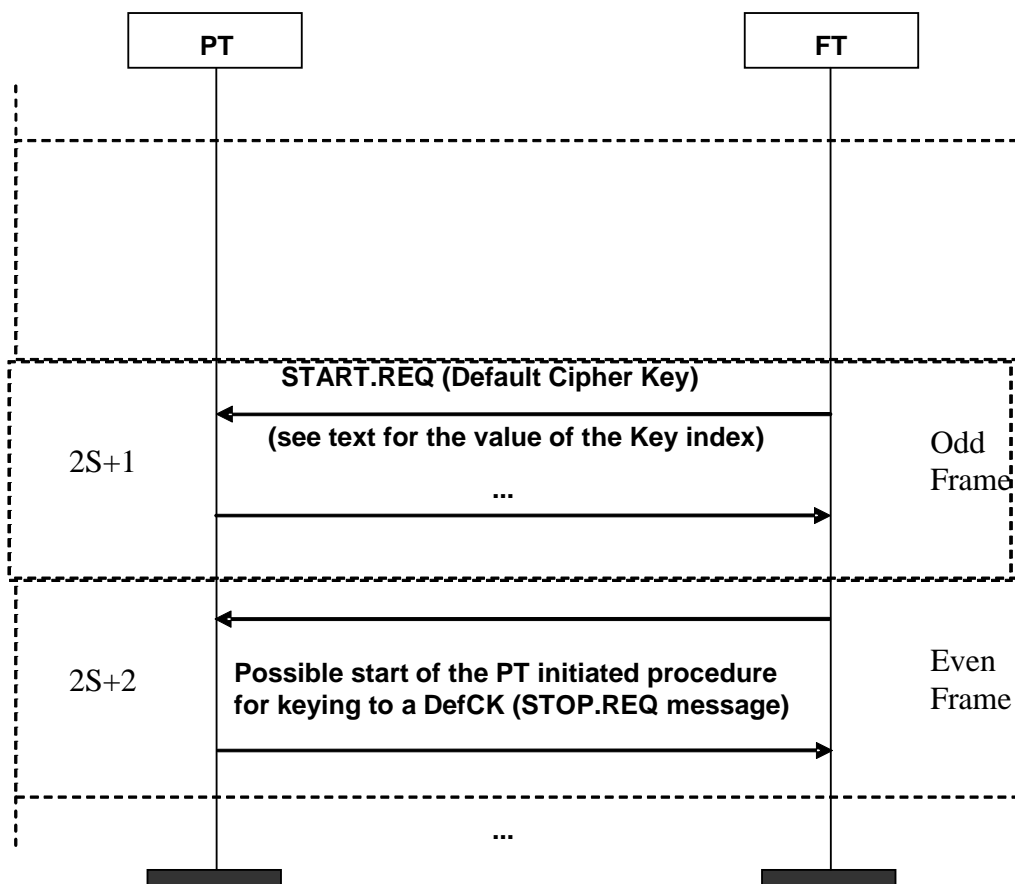


Figure 6.11b: FT indication of encryption change to a DefCK

#### 6.4.6.6 Insertion of WAIT

In the procedures described in clause 6.4.6.3 the FT may insert wait messages to allow for latency in key retrieval inside the FP. When receiving START.REQ with a default cipher key index the FT may reply with a WAIT message in next frame if not able to complete the procedure in time. The encoding of the WAIT message shall follow the definition in ETSI EN 300 175-3 [3], clause 7.2.5.2.3 for encryption switching of basic connections and clause 7.2.5.3.7 in ETSI EN 300 175-3 [3] for encryption switching of advanced connections.

When the PT receives WAIT in the frame following a transmission of START.REQ with a default cipher key index it may reset its retransmission counter for allowing more than the usual five retransmission of START.REQ.

If the encryption switching procedure has not been completed within one second, the PT shall release the connection and DLC layer is informed using the MAC\_DIS\_ind primitive.

This procedure extends the duration of the clear state. For security reasons, it is advisable not to send U-plane and C-plane ( $C_S$  or  $C_F$  channels) data until the procedure has been completed and the encryption has been activated.

## 6.4.7 Handover of the encryption process

### 6.4.7.0 General

Two types of handover are considered:

- 1) internal handover:
  - bearer handover;
  - connection handover;
- 2) external handover.

In principle, the techniques associated with connection handover can be used in place of bearer handover and the techniques associated with external handover can be used in place of connection handover or bearer handover.

External handover caters for the situation where the cipher key cannot be transferred between FTs in a secure manner. A method by which the cipher key could be securely transferred between FTs would allow this to be removed and the same procedure as used for connection handover to be applied.

### 6.4.7.1 Bearer handover, uninterrupted ciphering

Bearer handover is a MAC layer operation. It is transparent to the operation of the encryption processes.

The new bearer switches to the appropriate encryption mode during bearer establishment (see ETSI EN 300 175-3 [3]). If encryption is enabled the new bearer uses the same CK and IV as the original bearer.

### 6.4.7.2 Connection handover, uninterrupted ciphering

Connection handover is a DLC layer operation. During handover of a PT from one connection to another connection on the same FT, the PT will maintain both connections for a certain period. During this period the same sequence of key stream segments will be used for each connection.

NOTE: Connection 1 and connection 2 may belong to the same or different RFPs.

The notation  $S(N)$  is used to denote the pair of key stream segments ( $S_F(N)$ ,  $S_P(N)$ ) for frame number  $N$ .

During the first phase of handover, communication on connection 2 is in clear mode. Encryption is enabled on connection 2 by the PT using the procedures of clause 6.4.6.

For connection handover and using the default cipher key on connection 1, both procedures described in clause 6.4.6.3 (figures 6.6 and 6.7) are allowed to start switching to encrypt mode on connection 2 for the PT.

The FT shall support both procedures as described in clause 6.4.6.3 (figures 6.6 and 6.7) to start switching to encrypt mode on connection 2 in case default encryption is supported.

Once encryption has been established on connection 2, connection 1 may be released.

The handover is illustrated in figure 6.12.

| Frame number | Connection 1 | Connection 2           |
|--------------|--------------|------------------------|
| N            | S(N)         | clear                  |
| N + 1        | S(N + 1)     | clear/start set-up     |
| .            | .            | .                      |
| .            | .            | .                      |
| N + k - 1    | S(N + k - 1) | clear/start encryption |
| N + k        | S(N + k)     | S(N + k)               |
| N + k + 1    | S(N + k + 1) | S(N + k + 1)           |
| .            | .            | .                      |
| .            | .            | .                      |
| N + m - 1    | S(N + m - 1) | S(N + m - 1)           |
| N + m        | release      | S(N + m)               |
| N + m + 1    | -            | S(N + m + 1)           |

**Figure 6.12: Connection handover, uninterrupted ciphering**

### 6.4.7.3 External handover - handover with ciphering

The new connection is set up in clear mode. It is unnecessary to cease ciphering on the old connection prior to set-up of the new connection. It may not be possible to initiate ciphering on the new leg until the old connection is either released or switched to clear mode. Switching to encrypted mode is described in clause 6.4.6.3. Switching to clear mode is described in clause 6.4.6.4.

If the PP is to resume encryption using the same cipher key as it used with the original FP, then a copy of the cipher key shall be made available to the new FP during handover. The transfer of the cipher key shall be made entirely within the fixed network and shall not involve transmission over the air interface. If this is possible, consideration should be given to the use of a seamless handover as described in clause 6.4.7.2.

If such a transfer of key is not possible, then the data confidentiality service shall be re-invoked and a new cipher key established. To establish a new derived cipher key, re-authentication of the PP will be necessary.

## 6.4.8 Modifications for half and long slot specifications (2-level modulation)

### 6.4.8.1 Background

The purpose of this clause is to specify how the encryption processes given in the previous clauses are modified for the half slot and long slot formats.

### 6.4.8.2 MAC layer field structure

The MAC layer field structure for a half slot and long slot physical channels using 2-level modulation is depicted in figure 6.13.

|     |     |    |          |       |
|-----|-----|----|----------|-------|
| 8   | 40  | 16 | j        | 4     |
| H   | T   | RA | data     | X     |
| A   |     |    | B        | X     |
| D08 |     |    |          |       |
| a0  | a63 | b0 | b(j - 1) | x0 x3 |

**Figure 6.13: Half slot and long slot field structures**

As with the full slot structure, two types of B-field are defined: E-type and U-type. These carry the same types of channel data as in the full slot case.

In a B-field of U-type which is carrying  $I_N$  channel data all j data bits are channel data. This is the unprotected format for the B-field and corresponds to the U08a mapping in ETSI EN 300 175-3 [3].

The multisubfield protected format for the B-field is illustrated in figure 6.14. The multisubfield protected format is used with the B-field mappings U08b and E08, as defined in ETSI EN 300 175-3 [3]. The B0-field carries 64 bits of control or  $I_p$  channel data and a 16-bit redundancy check  $R_{B0}$  on this data.

|      |     |      |     |    |                   |   |
|------|-----|------|-----|----|-------------------|---|
| B0   |     | B1   | ... | BX | Padding<br>(note) | X |
| data | RB0 |      |     |    |                   |   |
| 64   | 16  | j-80 |     |    |                   | 4 |

NOTE: Padding only for long slot with  $j=672$  (see ETSI EN 300 175-3 [3], clause 6.2.1.3.3).

**Figure 6.14: Half slot and long slot B-field protected format**

### 6.4.8.3 Data to be encrypted

When operating in encrypt mode the same rules apply to the encryption of data as given in clause 6.4.3. The value of  $m$  is equal to  $j + 40$  bits.

In the A field, only the Tail (T) is encrypted. For the protected format B-field there are two portions excluded from encryption, namely the redundancy check  $R_{Bj}$  and the X-field. The corresponding bits in the KSS stream that match the position of the redundancy checks  $R_{Bj}$  are generated but not used. No KSS bits are generated for the A-field Heading (H) and RA and for the X field.

E-type mux format (including E+U type) is handled as multisubfield protected format.

In protected multisubfield format or in E-type mux mode, padding field bits when needed are not encrypted. However the corresponding bits in KSS are generated.

In  $I_{p\_}$  encoded protected format ( $I_{pX}$ ), the payload bits are encrypted before being processed by the coding engine. The two KSS are generated with the same length as in the unprotected format, but only the first bits of each segment are used. The number of bits that are used depends on the value of  $r$ .

### 6.4.8.4 Encryption process

For each pair of D08-fields which carries encrypted data the KSG outputs two key stream segments each of  $40 + j$  consecutive bits. The bits of a KSS are labelled as  $KSS(0) \dots, KSS(39 + j)$ .

The encryption process is as follows:

- If T is of type  $C_T$ , the bits  $KSS(0), \dots, KSS(39)$  are XORed with the bits  $a_8, \dots, a_{47}$  from T. Thus  $KSS(0)$  is XORed with  $a_8$ , etc. If T is not of type  $C_T$ , then bits  $KSS(0), \dots, KSS(39)$  are discarded.
- If the B-field has the unprotected format, the bits  $KSS(40), \dots, KSS(39 + j)$  are XORed with the bits  $b_0, \dots, b_{j-1}$  from B, so that  $KSS(40)$  is XORed with  $b_0$ , etc.
- If the B-field has the multisubfield protected format, the bits  $KSS(40), \dots, KSS(103)$  are XORed with the bits  $b_0, \dots, b_{63}$  from B0, so that  $KSS(40)$  is XORed with  $b_0$ , etc.
- Bits  $KSS(104), \dots, KSS(119)$  are not used. If  $j > 80$ , bits  $KSS(120), \dots, KSS(39 + j)$  are XORed with the bits  $b_{80}, \dots, b_{j-1}$ , so that  $KSS(120)$  is XORed with  $b_{80}$ , etc.
- E-type B-field multiplexer (including E+U type) is handled as the multisubfield protected format.
- In protected multisubfield format or in E-type mux mode, padding field bits when needed are not encrypted. However the corresponding bits in KSS are generated.
- If the B-field has the encoded protected format (MAC service  $I_{pX}$ ) the KSS is used as follows:
  - Bits  $KSS(40), \dots, KSS(m)$ , where  $m = j \times r + 39$ , with  $r$  the codec ratio, are XORed with the bits  $b_0, \dots, b_n$  of the B field payload, where  $n = j \times r - 1$ , before any stage of processing by the coding engine.

- The remaining bits of the KSS (until  $KSS(39 + j)$ ) are discarded.

#### 6.4.8.5 Initialization and synchronization of the encryption process

This is the same as described in clause 6.4.5, except that the concatenated key stream segments are  $80 + 2j$  bits long.

#### 6.4.8.6 Encryption mode control

This is exactly the same as described in clause 6.4.6.

#### 6.4.8.7 Handover of the encryption process

This is exactly the same as described in clause 6.4.7.

### 6.4.9 Modifications for double slot specifications (2-level modulation)

#### 6.4.9.1 Background

The purpose of this clause is to specify how the encryption processes given in the previous clauses are modified for the double slot formats.

#### 6.4.9.2 MAC layer field structure

The MAC layer field structure for a double slot physical channel using 2-level modulation is depicted in figure 6.15.

|     |     |    |      |       |
|-----|-----|----|------|-------|
|     |     |    |      |       |
|     |     |    |      |       |
|     |     |    |      |       |
|     |     |    |      |       |
|     |     |    |      |       |
|     |     |    |      |       |
| 8   | 40  | 16 | 800  | 4     |
| H   | T   | RA | data | X     |
| A   |     |    | B    | X     |
| D80 |     |    |      |       |
| a0  | a63 | b0 | b799 | x0 x3 |

**Figure 6.15: Double slot field structures**

As with the full slot structure, two types of B-field are defined: E-type and U-type. These carry the same types of channel data as in the full slot case.

In a B-field of U-type which is carrying  $I_N$  channel data all 800 data bits are channel data. This is the unprotected format for the B-field and corresponds to the U80a mapping in ETSI EN 300 175-3 [3].

The multisubfield protected format for the B-field is illustrated in figure 6.16. The multisubfield protected format is used with the B-field mappings U80b and E80, as defined in ETSI EN 300 175-3 [3]. The  $B_j$ -field carries 64 bits of control or  $I_p$  channel data and a 16-bit redundancy check  $R_{Bj}$  on this data.

|      |     |      |     |     |      |     |   |
|------|-----|------|-----|-----|------|-----|---|
| B0   |     | B1   |     | ... | B9   |     | X |
| data | RB0 | data | RB1 |     | data | RB9 |   |
| 64   | 16  | 64   | 16  |     | 64   | 16  | 4 |

**Figure 6.16: Double slot B-field multisubfield protected format**



The singlesubfield protected format for the B-field is illustrated in figure 6.17. The singlesubfield protected format is used with the B-field mappings U80c, as defined in ETSI EN 300 175-3 [3]. The B0-field carries 768 bits of control or  $I_p$  channel data and a 32-bit redundancy check  $R_{B0}$  on this data.

|      |          |   |
|------|----------|---|
| B0   |          | X |
| Data | $R_{B0}$ | X |
| 768  | 32       | 4 |

**Figure 6.17: Double slot B-field singlesubfield protected format**

### 6.4.9.3 Data to be encrypted

When operating in encrypt mode the same rules apply to the encryption of data as given in clause 6.4.3. The value of  $m$  is equal to 840 bits.

In the A field, only the Tail (T) is encrypted. For the protected format B-field there are two portions excluded from encryption, namely the redundancy check  $R_{Bj}$  and the X-field. The corresponding bits in the KSS stream that match the position of the redundancy checks  $R_{Bj}$  are generated but not used. No KSS bits are generated for the A-field Heading (H) and RA and for the X field.

E-type mux format (including E+U type) is handled as multisubfield protected format.

In  $I_p$  encoded protected format ( $I_{PX}$ ), the payload bits are encrypted before being processed by the coding engine. The two KSS are generated with the same length as in the unprotected format, but only the first bits of each segment are used. The number of bits that are used depends on the value of  $r$ .

### 6.4.9.4 Encryption process

For each pair of D80-fields which carries encrypted data the key stream generator outputs two key stream segments each of 840 consecutive bits. The bits of a KSS are labelled as KSS(0) ..., KSS(839).

The encryption process is as follows:

- If T is of type  $C_T$ , the bits KSS(0), ..., KSS(39) are XORed with the bits  $a_8, \dots, a_{47}$  from T. Thus KSS(0) is XORed with  $a_8$ , etc. If T is not of type  $C_T$ , then bits KSS(0), ..., KSS(39) are discarded.
- If the B-field has the unprotected format, the bits KSS(40), ..., KSS(839) are XORed with the bits  $b_0, \dots, b_{799}$  from B. That is KSS(40) is XORed with  $b_0$ , etc.
- If the B-field has the singlesubfield protected format, the bits KSS(40), ..., KSS(807) are XORed with the bits  $b_0, \dots, b_{767}$  from B. That is KSS(40) is XORed with  $b_0$ , etc. Bits KSS(808), ..., KSS(839) are not used.
- If the B-field has the multisubfield protected format, the bits KSS(40), ..., KSS(103) are XORed with the bits  $b_0, \dots, b_{63}$  from B0, so that KSS(40) is XORed with  $b_0$ , etc.
- Bits KSS(104), ..., KSS(119) are not used.
- Bits KSS(120), ..., KSS(183) are XORed with the bits  $b_{80}, \dots, b_{143}$  from B1, so that KSS(120) is XORed with  $b_{80}$ , etc.
- Bits KSS(184), ..., KSS(199) are not used.
- Bits KSS(200), ..., KSS(263) are XORed with the bits  $b_{160}, \dots, b_{223}$  from B2, so that KSS(200) is XORed with  $b_{160}$ , etc.
- Bits KSS(264), ..., KSS(279) are not used.
- Bits KSS(280), ..., KSS(343) are XORed with the bits  $b_{240}, \dots, b_{303}$  from B3, so that KSS(280) is XORed with  $b_{240}$ , etc.

- Bits KSS(344), ..., KSS(359) are not used.
- Bits KSS(360), ..., KSS(423) are XORed with the bits  $b_{320}, \dots, b_{383}$  from B4, so that KSS(360) is XORed with  $b_{320}$ , etc.
- Bits KSS(424), ..., KSS(439) are not used.
- Bits KSS(440), ..., KSS(503) are XORed with the bits  $b_{400}, \dots, b_{463}$  from B5, so that KSS(440) is XORed with  $b_{400}$ , etc.
- Bits KSS(504), ..., KSS(519) are not used.
- Bits KSS(520), ..., KSS(583) are XORed with the bits  $b_{480}, \dots, b_{543}$  from B6, so that KSS(520) is XORed with  $b_{480}$ , etc.
- Bits KSS(584), ..., KSS(599) are not used.
- Bits KSS(600), ..., KSS(663) are XORed with the bits  $b_{560}, \dots, b_{623}$  from B7, so that KSS(600) is XORed with  $b_{560}$ , etc.
- Bits KSS(664), ..., KSS(679) are not used.
- Bits KSS(680), ..., KSS(743) are XORed with the bits  $b_{640}, \dots, b_{703}$  from B8, so that KSS(680) is XORed with  $b_{640}$ , etc.
- Bits KSS(744), ..., KSS(759) are not used.
- Bits KSS(760), ..., KSS(823) are XORed with the bits  $b_{720}, \dots, b_{783}$  from B9, so that KSS(760) is XORed with  $b_{720}$ , etc.
- Bits KSS(824), ..., KSS(839) are not used.
- E-type B-field multiplexer (including E+U type) is handled as multisubfield protected format.
- If the B-field has the encoded protected format (MAC service  $I_{PX}$ ) the KSS is used as follows:
  - Bits KSS(40), ..., KSS( $m$ ), where  $m = 800 \times r + 39$ , with  $r$  the codec ratio, are XORed with the bits  $b_0, \dots, b_n$  of the B field payload, where  $n = 800 \times r - 1$ , before any stage of processing by the coding engine.
  - The remaining bits of the KSS (until KSS(839)) are discarded.

#### 6.4.9.5 Initialization and synchronization of the encryption process

This is the same as described in clause 6.4.5, except that the concatenated key stream segments are 1 680 bits long.

#### 6.4.9.6 Encryption mode control

This is exactly the same as described in clause 6.4.6.

#### 6.4.9.7 Handover of the encryption process

This is exactly the same as described in clause 6.4.7.

## 6.4.10 Modifications for multi-bearer specifications

Between one single FT and PT several connections may exist. Each connection may consist of a single or multiple bearers.

For connections with multiple bearers, encryption of each bearer is performed. The IV for each bearer is derived from the frame number, multiframe number as described in clause 6.4.5.

Double simplex bearers have two D-fields per frame, but unlike duplex bearers, both are transmitted by one side and received by the other side. The  $S_F(N)$  is used by the transmitting side to encrypt the first D-field transmitted in a frame and the  $S_P(N)$  to encrypt the second D-field transmitted. The receiving side consequently uses  $S_F(N)$  to decrypt the first D-field received in the frame and  $S_P(N)$  to decrypt the second D-field received.

Encryption may be enabled or disabled for each connection independently using the processes described in clause 6.4.6 applied to one duplex bearer of the connection. Once enabled/disabled, encryption for all bearers for that connection is enabled/disabled.

NOTE 1: The MAC\_ENC\_EKS-ind and the MAC\_ENC\_EKS-cfm primitives report the encryption mode for the whole connection.

All connections relating to the same U-plane call and/or the same broadband data link shall be switched to the same encryption mode.

NOTE 2: Unless LAPC data is routed to an Lc that controls an enciphered connection, C-plane data transferred from this LAPC is not encrypted even if it belongs to a call whose U-plane data is encrypted.

## 6.4.11 Modifications for 4-level, 8-level, 16-level and 64-level modulation formats

### 6.4.11.1 Background

The purpose of this clause is to specify how the encryption processes given in the previous clauses are modified for the 4-level, 8-level, 16-level and 64-level modulation formats.

Several modulation levels are defined for the S-field, A-field, B-field, X-field and Z-field in ETSI EN 300 175-2 [2]. The main combinations of modulation schemes are shown in table 6.7.

**Table 6.7: Main combinations of modulation schemes**

| Configuration | S-field        | A-field        | B + X + Z-field |
|---------------|----------------|----------------|-----------------|
| 1a            | GFSK           | GFSK           | GFSK            |
| 1b            | $\pi/2$ -DBPSK | $\pi/2$ -DBPSK | $\pi/2$ -DBPSK  |
| 2             | $\pi/2$ -DBPSK | $\pi/2$ -DBPSK | $\pi/4$ -DQPSK  |
| 3             | $\pi/2$ -DBPSK | $\pi/2$ -DBPSK | $\pi/8$ -D8PSK  |
| 4a            | $\pi/2$ -DBPSK | $\pi/4$ -DQPSK | $\pi/4$ -DQPSK  |
| 4b            | $\pi/2$ -DBPSK | $\pi/8$ -D8PSK | $\pi/8$ -D8PSK  |
| 5             | $\pi/2$ -DBPSK | $\pi/2$ -DBPSK | 16-QAM          |
| 6             | $\pi/2$ -DBPSK | $\pi/2$ -DBPSK | 64-QAM          |

### 6.4.11.2 MAC layer field structure

The MAC layer field structure for a double slot, long slot, full slot and half slot physical channel for 4-level, 8-level, 16-level and 64-level modulation is described in ETSI EN 300 175-3 [3].

### 6.4.11.3 Data to be encrypted

When operating in encrypt mode the same rules apply to the encryption of data as given in clause 6.4.3. However, for the protected format B-field there are two portions excluded from encryption, namely the redundancy check  $R_{Bj}$  and the X-field.

In  $I_P$  encoded protected format ( $I_{PX}$ ), the payload bits are encrypted before being processed by the coding engine. The two KSS are generated with the same length as in the unprotected format, but only the first bits of each segment are used. The number of bits that are used depends on the value of  $r$ .

#### 6.4.11.4 Encryption process

##### 6.4.11.4.0 General

For each pair of D-fields which carries encrypted data the key stream generator outputs two key stream segments each of KSSmax bits, where the value of KSSmax depends on the slot type and the level of modulation.

**Table 6.8: Length of key stream segments KSSmax**

| Configuratio<br>n | length of Key Stream Segment KSSmax (bits) |            |                     |                      |                      |
|-------------------|--|------------|---------------------|----------------------|----------------------|
|                   | double slot                                | full slot  | half slot<br>(j=80) | long slot<br>(j=640) | long slot<br>(j=672) |
| 1a                | 40 + 800                                   | 40 + 320   | 40 + 80             | 40 + 640             | 40 + 672             |
| 1b                | 40 + 800                                   | 40 + 320   | 40 + 80             | 40 + 640             | 40 + 672             |
| 2                 | 40 + 1 600                                 | 40 + 640   | 40 + 160            | 40 + 1 280           | 40 + 1 344           |
| 3                 | 40 + 2 400                                 | 40 + 960   | 40 + 240            | 40 + 1 920           | 40 + 2 016           |
| 4a                | 80 + 1 600                                 | 80 + 640   | 80 + 160            | 80 + 1 280           | 80 + 1 344           |
| 4b                | 120 + 2 400                                | 120 + 960  | 120 + 240           | 120 + 1 920          | 120 + 2 016          |
| 5                 | 40 + 3 200                                 | 40 + 1 280 | 40 + 320            | 40 + 2 560           | 40 + 2 688           |
| 6                 | 40 + 4 800                                 | 40 + 1 920 | 40 + 480            | 40 + 3 840           | 40 + 4 032           |

The bits of a KSS are labelled as KSS(0) ...KSS(KSSmax - 1).

##### 6.4.11.4.1 Encryption process for the A-field and for the unprotected format

The encryption process for the configurations 1a, 1b, 2, 3, 5 and 6 is as follows:

- If T is of type  $C_T$ , the bits KSS(0), ..., KSS(39) are XORed with the bits  $a_8, \dots, a_{47}$  from T. Thus KSS(0) is XORed with  $a_8$ , etc.
- If T is not of type  $C_T$ , then bits KSS(0), ...,KSS(39) are discarded.
- If the B-field has the unprotected format then bit  $b_i$  is XORed with bit KSS(i + 40). That is KSS(40) is XORed with  $b_0$ , etc.

The encryption process for the configuration 4a is as follows:

- If T is of type  $C_T$ , the bits KSS(0), ..., KSS(79) are XORed with the bits  $a_8, \dots, a_{87}$  from T. Thus KSS(0) is XORed with  $a_8$ , etc.
- If T is not of type  $C_T$ , then bits KSS(0), ...,KSS(79) are discarded.
- If the B-field has the unprotected format then bit  $b_i$  is XORed with bit KSS(i + 80). That is KSS(80) is XORed with  $b_0$ , etc.

The encryption process for the configuration 4b is as follows:

- If T is of type  $C_T$ , the bits KSS(0), ..., KSS(119) are XORed with the bits  $a_8, \dots, a_{127}$  from T. Thus KSS(0) is XORed with  $a_8$ , etc.
- If T is not of type  $C_T$ , then bits KSS(0), ...,KSS(119) are discarded.
- If the B-field has the unprotected format then bit  $b_i$  is XORed with bit KSS(i + 120). That is KSS(120) is XORed with  $b_0$ , etc.

For the different slot types and modulation schemes different numbers of bits in the B-field  $i_{\max}$  are available. Table 6.9 lists the different slot formats and modulation schemes and the resulting mapping of bits in the B-field to bits of the KSS to be XORed with:

**Table 6.9: Mapping of KSS bits to B-field bits for unprotected format and encoded protected format**

| Configuration | Double slot              |                        |              | Full slot                |                        |              | Half slot (j=80)         |                     |              |
|---------------|--------------------------|------------------------|--------------|--------------------------|------------------------|--------------|--------------------------|---------------------|--------------|
|               | Data bits in the B-field | $b_i$                  | XORed with   | Data bits in the B-field | $b_i$                  | XORed with   | Data bits in the B-field | $b_i$               | XORed with   |
| 1a            | 800                      | $b_0 \dots b_{799}$    | KSS(i + 40)  | 320                      | $b_0 \dots b_{319}$    | KSS(i + 40)  | 80                       | $b_0 \dots b_{79}$  | KSS(i + 40)  |
| 1b            | 800                      | $b_0 \dots b_{799}$    | KSS(i + 40)  | 320                      | $b_0 \dots b_{319}$    | KSS(i + 40)  | 80                       | $b_0 \dots b_{79}$  | KSS(i + 40)  |
| 2             | 1 600                    | $b_0 \dots b_{1\,599}$ | KSS(i + 40)  | 640                      | $b_0 \dots b_{639}$    | KSS(i + 40)  | 160                      | $b_0 \dots b_{159}$ | KSS(i + 40)  |
| 3             | 2 400                    | $b_0 \dots b_{2\,399}$ | KSS(i + 40)  | 960                      | $b_0 \dots b_{959}$    | KSS(i + 40)  | 240                      | $b_0 \dots b_{239}$ | KSS(i + 40)  |
| 4a            | 1 600                    | $b_0 \dots b_{1\,599}$ | KSS(i + 80)  | 640                      | $b_0 \dots b_{639}$    | KSS(i + 80)  | 160                      | $b_0 \dots b_{159}$ | KSS(i + 80)  |
| 4b            | 2 400                    | $b_0 \dots b_{2\,399}$ | KSS(i + 120) | 960                      | $b_0 \dots b_{959}$    | KSS(i + 120) | 240                      | $b_0 \dots b_{239}$ | KSS(i + 120) |
| 5             | 3 200                    | $b_0 \dots b_{3\,199}$ | KSS(i + 40)  | 1 280                    | $b_0 \dots b_{1\,279}$ | KSS(i + 40)  | 320                      | $b_0 \dots b_{319}$ | KSS(i + 40)  |
| 6             | 4 800                    | $b_0 \dots b_{4\,799}$ | KSS(i + 40)  | 1 920                    | $b_0 \dots b_{1\,919}$ | KSS(i + 40)  | 560                      | $b_0 \dots b_{559}$ | KSS(i + 40)  |

| Configuration | Long slot (j=640)        |                        |              | Long slot (j=672)        |                        |              |
|---------------|--------------------------|------------------------|--------------|--------------------------|------------------------|--------------|
|               | Data bits in the B-field | $b_i$                  | XORed with   | Data bits in the B-field | $b_i$                  | XORed with   |
| 1a            | 640                      | $b_0 \dots b_{639}$    | KSS(i + 40)  | 672                      | $b_0 \dots b_{671}$    | KSS(i + 40)  |
| 1b            | 640                      | $b_0 \dots b_{639}$    | KSS(i + 40)  | 672                      | $b_0 \dots b_{671}$    | KSS(i + 40)  |
| 2             | 1 280                    | $b_0 \dots b_{1\,279}$ | KSS(i + 40)  | 1 344                    | $b_0 \dots b_{1\,343}$ | KSS(i + 40)  |
| 3             | 1 920                    | $b_0 \dots b_{1\,919}$ | KSS(i + 40)  | 2 016                    | $b_0 \dots b_{2\,015}$ | KSS(i + 40)  |
| 4a            | 1 280                    | $b_0 \dots b_{1\,279}$ | KSS(i + 80)  | 1 344                    | $b_0 \dots b_{1\,343}$ | KSS(i + 80)  |
| 4b            | 1 920                    | $b_0 \dots b_{1\,919}$ | KSS(i + 120) | 2 016                    | $b_0 \dots b_{2\,015}$ | KSS(i + 120) |
| 5             | 2 560                    | $b_0 \dots b_{2\,559}$ | KSS(i + 40)  | 2 688                    | $b_0 \dots b_{2\,687}$ | KSS(i + 40)  |
| 6             | 3 840                    | $b_0 \dots b_{3\,839}$ | KSS(i + 40)  | 4 032                    | $b_0 \dots b_{4\,031}$ | KSS(i + 40)  |

#### 6.4.11.4.2 Encryption process for the single subfield protected format

The generation of KSS and the encryption of the A field is handled as described for the unprotected format.

If the B-field has the single subfield protected format, then for the configurations 1a, 1b, 2, 3, 5 and 6 bit  $b_i$  is XORed with bit KSS(i + 40). That is KSS(40) is XORed with  $b_0$ , etc.

For the different slot types and modulation schemes different numbers of bits in the B-field  $i_{\max}$  are available. Table 6.10 lists the different slot formats and modulation schemes and the resulting mapping of bits in the B-field to bits of the KSS to be XORed with:

**Table 6.10: Mapping of KSS bits to B-field bits for single subfield protected formats**

| Configuration | Double slot              |                        |                  | Full slot                |                        |                  |
|---------------|--------------------------|------------------------|------------------|--------------------------|------------------------|------------------|
|               | Data bits in the B-field | $b_i$                  | XORed with       | Data bits in the B-field | $b_i$                  | XORed with       |
| 1a            | 768                      | $b_0 \dots b_{767}$    | KSS( $i + 40$ )  | 304                      | $b_0 \dots b_{303}$    | KSS( $i + 40$ )  |
| 1b            | 768                      | $b_0 \dots b_{767}$    | KSS( $i + 40$ )  | 304                      | $b_0 \dots b_{303}$    | KSS( $i + 40$ )  |
| 2             | 1 568                    | $b_0 \dots b_{1\,567}$ | KSS( $i + 40$ )  | 608                      | $b_0 \dots b_{607}$    | KSS( $i + 40$ )  |
| 3             | 2 368                    | $b_0 \dots b_{2\,367}$ | KSS( $i + 40$ )  | 928                      | $b_0 \dots b_{927}$    | KSS( $i + 40$ )  |
| 4a            | 1 568                    | $b_0 \dots b_{1\,567}$ | KSS( $i + 80$ )  | 608                      | $b_0 \dots b_{607}$    | KSS( $i + 80$ )  |
| 4b            | 2 368                    | $b_0 \dots b_{2\,367}$ | KSS( $i + 120$ ) | 928                      | $b_0 \dots b_{927}$    | KSS( $i + 120$ ) |
| 5             | 3 168                    | $b_0 \dots b_{3\,167}$ | KSS( $i + 40$ )  | 1 248                    | $b_0 \dots b_{1\,247}$ | KSS( $i + 40$ )  |
| 6             | 4 768                    | $b_0 \dots b_{4\,767}$ | KSS( $i + 40$ )  | 1 888                    | $b_0 \dots b_{1\,887}$ | KSS( $i + 40$ )  |

| Configuration | Long slot ( $j=640$ )    |                        |                  | Long slot ( $j=672$ )    |                        |                  |
|---------------|--------------------------|------------------------|------------------|--------------------------|------------------------|------------------|
|               | Data bits in the B-field | $b_i$                  | XORed with       | Data bits in the B-field | $b_i$                  | XORed with       |
| 1a            | 608                      | $b_0 \dots b_{607}$    | KSS( $i + 40$ )  | 640                      | $b_0 \dots b_{639}$    | KSS( $i + 40$ )  |
| 1b            | 608                      | $b_0 \dots b_{607}$    | KSS( $i + 40$ )  | 640                      | $b_0 \dots b_{639}$    | KSS( $i + 40$ )  |
| 2             | 1 248                    | $b_0 \dots b_{1\,247}$ | KSS( $i + 40$ )  | 1 312                    | $b_0 \dots b_{1\,311}$ | KSS( $i + 40$ )  |
| 3             | 1 888                    | $b_0 \dots b_{1\,887}$ | KSS( $i + 40$ )  | 1 984                    | $b_0 \dots b_{1\,983}$ | KSS( $i + 40$ )  |
| 4a            | 1 248                    | $b_0 \dots b_{1\,247}$ | KSS( $i + 80$ )  | 1 312                    | $b_0 \dots b_{1\,311}$ | KSS( $i + 80$ )  |
| 4b            | 1 888                    | $b_0 \dots b_{1\,887}$ | KSS( $i + 120$ ) | 1 984                    | $b_0 \dots b_{1\,983}$ | KSS( $i + 120$ ) |
| 5             | 2 528                    | $b_0 \dots b_{2\,527}$ | KSS( $i + 40$ )  | 2 556                    | $b_0 \dots b_{2\,555}$ | KSS( $i + 40$ )  |
| 6             | 3 808                    | $b_0 \dots b_{3\,807}$ | KSS( $i + 40$ )  | 4 000                    | $b_0 \dots b_{3\,999}$ | KSS( $i + 40$ )  |

#### 6.4.11.4.3 Encryption process for the multi-subfield protected format

The generation of KSS and the encryption of the A field is handled as described for the unprotected format.

The following indicates how the B-subfields are encrypted for multisubfield protected B-field modes.

Table 6.11 should be read as follows:

- $b(x)_i$  indicates the  $i^{\text{th}}$  bit  $b$  of the  $x^{\text{th}}$  subfield ( $b_i$  of subfield  $x$ ). The indicated bit is XORed according to the corresponding formula.
- $KSS((80 \times x) + i + 40)$  indicates the KSS bit corresponding to the  $i^{\text{th}}$  bit  $b$  of the  $x^{\text{th}}$  subfield.

**Table 6.11: Encryption of B-subfields for multisubfield protected B-field modes**

| Configu-ration | Double slot              |  |                                | Full slot                |  |                                | Half slot (j=80)         |   |                                |
|----------------|--------------------------|--|--------------------------------|--------------------------|--|--------------------------------|--------------------------|---|--------------------------------|
|                | Data bits in the B-field | $b(x)_i$   | XORed with                     | Data bits in the B-field | $b_i$  | XORed with                     | Data bits in the B-field | $b_i$   | XORed with                     |
| 1a             | $10 \times 64$           | $(b(0)_0 \dots b(0)_{63})$<br>... $(b(9)_0 \dots b(9)_{63})$   | $KSS((80 \times x) + i + 40)$  | $4 \times 64$            | $(b(0)_0 \dots b(0)_{63}) \dots$<br>$(b(3)_0 \dots b(3)_{63})$   | $KSS((80 \times x) + i + 40)$  | $1 \times 64$            | $(b(0)_0 \dots b(0)_{63})$                                      | $KSS((80 \times x) + i + 40)$  |
| 1b             | $10 \times 64$           | $(b(0)_0 \dots b(0)_{63})$<br>... $(b(9)_0 \dots b(9)_{63})$   | $KSS((80 \times x) + i + 40)$  | $4 \times 64$            | $(b(0)_0 \dots b(0)_{63}) \dots$<br>$(b(3)_0 \dots b(3)_{63})$   | $KSS((80 \times x) + i + 40)$  | $1 \times 64$            | $(b(0)_0 \dots b(0)_{63})$                                      | $KSS((80 \times x) + i + 40)$  |
| 2              | $20 \times 64$           | $(b(0)_0 \dots b(0)_{63})$<br>... $(b(19)_0 \dots b(19)_{63})$ | $KSS((80 \times x) + i + 40)$  | $8 \times 64$            | $(b(0)_0 \dots b(0)_{63}) \dots$<br>$(b(7)_0 \dots b(7)_{63})$   | $KSS((80 \times x) + i + 40)$  | $2 \times 64$            | $(b(0)_0 \dots b(0)_{63}),$<br>$(b(1)_0 \dots b(1)_{63})$       | $KSS((80 \times x) + i + 40)$  |
| 3              | $30 \times 64$           | $(b(0)_0 \dots b(0)_{63})$<br>... $(b(29)_0 \dots b(29)_{63})$ | $KSS((80 \times x) + i + 40)$  | $12 \times 64$           | $(b(0)_0 \dots b(0)_{63}) \dots$<br>$(b(11)_0 \dots b(11)_{63})$ | $KSS((80 \times x) + i + 40)$  | $3 \times 64$            | $(b(0)_0 \dots b(0)_{63}) \dots$<br>$(b(2)_0 \dots b(2)_{63})$  | $KSS((80 \times x) + i + 40)$  |
| 4a             | $20 \times 64$           | $(b(0)_0 \dots b(0)_{63})$<br>... $(b(19)_0 \dots b(19)_{63})$ | $KSS((80 \times x) + i + 80)$  | $8 \times 64$            | $(b(0)_0 \dots b(0)_{63}) \dots$<br>$(b(7)_0 \dots b(7)_{63})$   | $KSS((80 \times x) + i + 80)$  | $2 \times 64$            | $(b(0)_0 \dots b(0)_{63}), \dots$<br>$(b(1)_0 \dots b(1)_{63})$ | $KSS((80 \times x) + i + 80)$  |
| 4b             | $30 \times 64$           | $(b(0)_0 \dots b(0)_{63})$<br>... $(b(29)_0 \dots b(29)_{63})$ | $KSS((80 \times x) + i + 120)$ | $12 \times 64$           | $(b(0)_0 \dots b(0)_{63}) \dots$<br>$(b(11)_0 \dots b(11)_{63})$ | $KSS((80 \times x) + i + 120)$ | $3 \times 64$            | $(b(0)_0 \dots b(0)_{63}) \dots$<br>$(b(2)_0 \dots b(2)_{63})$  | $KSS((80 \times x) + i + 120)$ |
| 5              | $40 \times 64$           | $(b(0)_0 \dots b(0)_{63})$<br>... $(b(39)_0 \dots b(39)_{63})$ | $KSS((80 \times x) + i + 40)$  | $16 \times 64$           | $(b(0)_0 \dots b(0)_{63}) \dots$<br>$(b(15)_0 \dots b(15)_{63})$ | $KSS((80 \times x) + i + 40)$  | $4 \times 64$            | $(b(0)_0 \dots b(0)_{63}) \dots$<br>$(b(3)_0 \dots b(3)_{63})$  | $KSS((80 \times x) + i + 40)$  |
| 6              | $60 \times 64$           | $(b(0)_0 \dots b(0)_{63})$<br>... $(b(59)_0 \dots b(59)_{63})$ | $KSS((80 \times x) + i + 40)$  | $24 \times 64$           | $(b(0)_0 \dots b(0)_{63}) \dots$<br>$(b(23)_0 \dots b(23)_{63})$ | $KSS((80 \times x) + i + 40)$  | $6 \times 64$            | $(b(0)_0 \dots b(0)_{63}) \dots$<br>$(b(5)_0 \dots b(5)_{63})$  | $KSS((80 \times x) + i + 40)$  |

| Configu-<br>ration | Long slot (j=640/672)          |  |                                |
|--------------------|--------------------------------|--|--------------------------------|
|                    | Data bits<br>in the<br>B-field | $b(x)_i$   | XORed with                     |
| 1a                 | $8 \times 64$                  | $(b(0)_0 \dots b(0)_{63}) \dots$<br>$(b(7)_0 \dots b(7)_{63})$   | $KSS((80 \times x) + i + 40)$  |
| 1b                 | $8 \times 64$                  | $(b(0)_0 \dots b(0)_{63}) \dots$<br>$(b(7)_0 \dots b(7)_{63})$   | $KSS((80 \times x) + i + 40)$  |
| 2                  | $16 \times 64$                 | $(b(0)_0 \dots b(0)_{63}) \dots$<br>$(b(15)_0 \dots b(15)_{63})$ | $KSS((80 \times x) + i + 40)$  |
| 3                  | $24 \times 64$                 | $(b(0)_0 \dots b(0)_{63}) \dots$<br>$(b(23)_0 \dots b(23)_{63})$ | $KSS((80 \times x) + i + 40)$  |
| 4a                 | $16 \times 64$                 | $(b(0)_0 \dots b(0)_{63}) \dots$<br>$(b(15)_0 \dots b(15)_{63})$ | $KSS((80 \times x) + i + 80)$  |
| 4b                 | $24 \times 64$                 | $(b(0)_0 \dots b(0)_{63}) \dots$<br>$(b(23)_0 \dots b(23)_{63})$ | $KSS((80 \times x) + i + 120)$ |
| 5                  | $32 \times 64$                 | $(b(0)_0 \dots b(0)_{63}) \dots$<br>$(b(31)_0 \dots b(31)_{63})$ | $KSS((80 \times x) + i + 40)$  |
| 6                  | $48 \times 64$                 | $(b(0)_0 \dots b(0)_{63}) \dots$<br>$(b(47)_0 \dots b(47)_{63})$ | $KSS((80 \times x) + i + 40)$  |

#### 6.4.11.4.4 Encryption process for the constant-size-subfield protected format

The generation of KSS and the encryption of the A field is handled as in the unprotected format.

The following indicates how the B-subfields are encrypted for constant-size-subfield protected B-field modes.

The constant-size-subfield protected B-field mode behaves as a multisubfield mode where the size of each subfield is equal to the size of the single B field for the same slot type in single-subfield format and 2-level modulation.

Only data bits are encrypted. CRC bits are not encrypted and the corresponding bits in KSS are discarded.

#### 6.4.11.4.5 Encryption process for the encoded protected format (MAC service $I_{PX}$ )

The generation of KSS and the encryption of the A field is handled as in the unprotected format. Each KSS is therefore a segment of  $KSS_{max}$  bits labelled  $KSS(0) \dots KSS(KSS_{max} - 1)$ , where  $KSS_{max}$  depends of the slot type and modulation scheme. The value of  $KSS_{max}$  for each combination is as given in table 6.8.

The number of B-field bits of each combination is defined as  $q$ , where  $q$  is the second summand given in table 6.8. For all modulation schemas except configurations 4a and 4b,  $q = KSS_{max} - 40$ . For configurations 4a and 4b refer to table 6.8.



The following indicates how the B field is encrypted for the encoded protected format (MAC service  $I_{pX}$ ):

- If the B-field has the encoded protected format (MAC service  $I_{pX}$ ) and modulation configuration is any schema other than 4a or 4b, the KSS is used as follows:
  - Bits KSS(40), ..., KSS( $m$ ), where  $m = q \times r + 39$ , with  $r$  the codec ratio, are XORed with the bits  $b_0, \dots, b_n$  of the B field payload, where  $n = q \times r - 1$ , before any stage of processing by the coding engine.
  - The remaining bits of the KSS (until KSS(KSSmax)) are discarded.
- If the B-field has the encoded protected format (MAC service  $I_{pX}$ ) and modulation configuration is schema 4a, the KSS is used as follows:
  - Bits KSS(80), ..., KSS( $m$ ), where  $m = q \times r + 79$ , with  $r$  the codec ratio, are XORed with the bits  $b_0, \dots, b_n$  of the B field payload, where  $n = q \times r - 1$ , before any stage of processing by the coding engine.
  - The remaining bits of the KSS (until KSS(KSSmax)) are discarded.
- If the B-field has the encoded protected format (MAC service  $I_{pX}$ ) and modulation configuration is schema 4b, the KSS is used as follows:
  - Bits KSS(120), ..., KSS( $m$ ), where  $m = q \times r + 119$ , with  $r$  the codec ratio, are XORed with the bits  $b_0, \dots, b_n$  of the B field payload, where  $n = q \times r - 1$ , before any stage of processing by the coding engine.
  - The remaining bits of the KSS (until KSS(KSSmax)) are discarded.

### 6.4.11.5 Initialization and synchronization of the encryption process

This is the same as described in clause 6.4.5, except that the concatenated key stream segments are 1 680 bits long.

### 6.4.11.6 Encryption mode control

This is exactly the same as described in clause 6.4.6.

### 6.4.11.7 Handover of the encryption process

This is exactly the same as described in clause 6.4.7.

## 6.4.12 Procedures for CCM re-keying and sequence reset

These procedures are in charge of changing the DCK used by the CCM encryption during an encrypted call and, if needed, to reset the DLC and CCM sequence numbers.

These procedures are for further study.

In absence of dedicated procedures, the resetting of CCM sequence and also the re-keying, may be performed by executing a complete Release (CC Release) of the virtual call and then setting a new call, or by executing a NWK layer suspend (CC Service Change) followed by a NWK layer resume. Both procedures create a new link with a reset sequence number and a new CCM key.

## 6.5 Security attributes

### 6.5.1 Background

Security attributes are used to determine which security algorithms and keys are used by a PT and an FT.

Clause 6.5.2 describes how an FT (respectively a PT) identifies the authentication algorithm and authentication key which it wishes to use for an instance of authentication of the PT (respectively authentication of the FT) by including an <<AUTH-TYPE>> information element in the {AUTHENTICATION-REQUEST} message. The structure of the <<AUTH-TYPE>> information element is defined in ETSI EN 300 175-5 [5]. The authentication algorithm identifier octet in <<AUTH-TYPE>> identifies the authentication algorithm (A algorithm in clause 5.1.1) and the Auth.Key type/Auth.key nr identifies the authentication key. An {AUTHENTICATION-REJECT} message (see table 6.12) is used to signal that an <<AUTH-TYPE>> is unacceptable to the recipient.

**Table 6.12: Authentication reject message**

| AUTH_MESSAGE            | Message Type | AUTH_ELEMENTS                        |
|-------------------------|--------------|--------------------------------------|
| {AUTHENTICATION-REJECT} | 01000011     | <<REJECT REASON>><br>(<<AUTH-TYPE>>) |

Clause 6.5.3 describes how an FT (respectively a PT) identifies the cipher algorithm which it wishes to use for an instance of data confidentiality by sending a {CIPHER-REQUEST} (respectively a {CIPHER-SUGGEST}) message which includes a <<CIPHER-INFO>> information element. The structure of the <<CIPHER-INFO>> information element is defined in ETSI EN 300 175-5 [5]. The cipher algorithm identifier octet in <<CIPHER-INFO>> identifies the cipher algorithm (KSG in clause 4.5.4) and the cipher key type/cipher key nr identifies the cipher key. A {CIPHER-REJECT} message (see table 6.13) is used to signal that the <<CIPHER-INFO>> is unacceptable.

**Table 6.13: Cipher messages**

| CIPHER_MESSAGE   | Message Type | CIPHER_ELEMENTS                        |
|------------------|--------------|--|
| {CIPHER-REQUEST} | 01001100     | <<CIPHER-INFO>>                        |
| {CIPHER-REJECT}  | 01001111     | <<REJECT REASON>><br>(<<CIPHER-INFO>>) |
| {CIPHER-SUGGEST} | 01001110     | <<CIPHER-INFO>>                        |

Clause 6.5.4 describes how a PT may send <<AUTH-TYPE>> and <<CIPHER-INFO>> elements in an {ACCESS-RIGHTS-REQUEST} message to indicate the authentication and cipher algorithms which it wishes to use in conjunction with the particular system (subscription). It is the intention that if these elements are accepted by the FT, then the identified algorithms will be associated with the system (subscription) and used whenever a call is made on that particular system (subscription). The FT signals acceptance of the elements by including them in an {ACCESS-RIGHTS-ACCEPT} message. It signals rejection of one or both of them by returning an {ACCESS-RIGHTS-REJECT} message. The access-rights messages are summarized in table 6.14.

**Table 6.14: Access-rights messages**

| ACCESS_MESSAGE          | Message Type | ELEMENTS   |
|-------------------------|--------------|--|
| {ACCESS-RIGHTS-REQUEST} | 01000100     | <<AUTH-TYPE>><br><<CIPHER-INFO>>                         |
| {ACCESS-RIGHTS-ACCEPT}  | 01000101     | <<AUTH-TYPE>><br><<CIPHER-INFO>>                         |
| {ACCESS-RIGHTS-REJECT}  | 01001011     | <<REJECT REASON>><br>(<<AUTH-TYPE>>,<br><<CIPHER-INFO>>) |

The key identifier octets in the <<AUTH-TYPE>> and <<CIPHER-INFO>> elements include key type and key number fields. The way in which these fields are used is described in clause 6.5.5.

In clause 6.5.6 an over-air key allocation protocol is defined. This protocol may be used to transform a manually distributed AC into a UAK. It is recommended that the procedure should only be used if a secure means to distribute the UAK cannot be implemented.

It should be noted that the NWK layer messages used in this clause may contain information elements other than those listed in tables 6.12, 6.13 and 6.14. For full details of the messages the reader is referred to ETSI EN 300 175-5 [5].

## 6.5.2 Authentication protocols

### 6.5.2.0 General

The protocols for authentication of a PT and authentication of an FT are described in clauses 6.5.2.1 and 6.5.2.2 respectively.

#### 6.5.2.1 Authentication of a PT type 1 procedure

This mechanism is used when the DECT Standard Authentication Algorithm (DSAA) or a proprietary algorithm is used. See clause 6.5.2.3 for the equivalent procedure to be used when the DECT Standard Authentication Algorithm #2 (DSAA2) is used.

The authentication protocol proceeds as follows:

- 1) the FT sends an {AUTHENTICATION-REQUEST} to the PT. The <<AUTH-TYPE>> element in this message identifies the authentication algorithm and authentication key which are requested (by the FT) to be used by the PT;
- 2) upon receipt of the {AUTHENTICATION-REQUEST}, the PT examines the <<AUTH-TYPE>> element to see if it is acceptable. This element is defined to be acceptable if the PT can implement the authentication algorithm and has the authentication key identified in the element (see note 2);
- 3) if the <<AUTH-TYPE>> element is acceptable, the PT returns an {AUTHENTICATION-REPLY} message with the <<RES>> element computed using the identified authentication algorithm and authentication key;
- 4) if the <<AUTH-TYPE>> element is unacceptable, the PT returns an {AUTHENTICATION-REJECT} message. The <<REJECT REASON>> element in the message is set to indicate the reason for the rejection. The possible rejection reasons are listed in table 6.15. The {AUTHENTICATION-REJECT} message may optionally include an <<AUTH-TYPE>> element constructed by the PT (see note 3). This element identifies an authentication algorithm and an authentication key which is acceptable to the PT;

- 5) upon receipt of an {AUTHENTICATION-REJECT}, the FT examines the <<REJECT REASON>> and then takes a prescribed action (see note 4). This will be based on the reason for the rejection and the reason for requesting authentication.

NOTE 1: See also clause 4.3.1 for description of the security process operations to be performed in order to generate and evaluate the parameters transported over the air i/f procedure.

NOTE 2: It is envisaged that in most applications the FT will know what authentication algorithm(s) and key(s) are supported by the PT from its knowledge of the IPUI, so that <<AUTH-TYPE>> should always be acceptable to the PT (see also clause 6.5.4).

NOTE 3: The {AUTHENTICATION-REJECT} message may optionally include a sequence of <<AUTH-TYPE>> elements, whereby the position of the element in the sequence indicates the order of preference.

NOTE 4: No attempt is made in the present document to specify the prescribed actions, only the options are presented. A prescribed action may depend upon the <<REJECT REASON>> and the circumstances under which authentication of the PT was requested.

A list of possible actions is given in table 6.15.

**Table 6.15: REJECT REASON, authentication of a PT**

| Code  | Reject reason (from PT)   | Actions open to FT |
|-------|---|--------------------|
| 11    | No authentication algorithm   | 1, 2               |
| 12    | Identified authentication algorithm not supported   | 1, 2, 3            |
| 13    | Identified authentication key not available   | 1, 2, 3            |
| 14    | UPI not entered (see note)  | 1, 2, 3            |
| NOTE: | The <<REJECT REASON>> "UPI not entered" is returned when user authentication is requested, the <<AUTH-TYPE>> element is acceptable to the PT but the user fails to enter his UPI. |                    |

#### Codes for Actions open to FT

- 1) Release call.
- 2) Proceed with call without authentication of the PT (see note 5).
- 3) Send new {AUTHENTICATION-REQUEST} with new <<AUTH-TYPE>> element. (This element may have been received in the {AUTHENTICATION-REJECT}.)

NOTE 5: The option to proceed without authentication is not recommended.

#### 6.5.2.2 Authentication of an FT type 1 procedure

This mechanism is used when the DECT Standard Authentication Algorithm (DSAA) or a proprietary algorithm is used. See clause 6.5.2.4 for the equivalent procedure to be used when the DECT Standard Authentication Algorithm #2 (DSAA2) is used.

The authentication protocol proceeds as follows:

- 1) the PT sends an {AUTHENTICATION-REQUEST} to the FT. The <<AUTH-TYPE>> element in the message identifies the authentication algorithm and authentication key which are requested (by the PT) to be used by the fixed network to derive the value RES in the {AUTHENTICATION-REPLY};
- 2) upon receipt of the {AUTHENTICATION-REQUEST}, the FT examines the <<AUTH-TYPE>> element to see if it is acceptable. The element is defined to be acceptable if the FT can return a <<RES>> generated using the identified authentication algorithm and authentication key (see note 2);
- 3) if the <<AUTH-TYPE>> element is acceptable, the FT returns an {AUTHENTICATION-REPLY} message with the <<RES>> element computed using the identified authentication algorithm and authentication key;

- 4) if the <<AUTH-TYPE>> element is unacceptable, the FT returns an {AUTHENTICATION-REJECT} message. The <<REJECT REASON>> element in the message is set to indicate the reason for the rejection. The possible rejection reasons are listed in table 6.16. The {AUTHENTICATION-REJECT} message may optionally include an <<AUTH-TYPE>> element (see note 3). This element identifies an authentication algorithm and an authentication key which is acceptable to the fixed network (see note 4);
- 5) upon receipt of an {AUTHENTICATION-REJECT}, the PT examines the <<REJECT REASON>> and then takes a prescribed action (see note 5). This will be based on the reason for the rejection and the reason for requesting authentication. A list of possible actions is given in table 6.16.

NOTE 1: See also clause 4.3.2 for description of the security process operations to be performed in order to generate and evaluate the parameters transported over the air i/f procedure.

NOTE 2: It is envisaged that in most applications the PT will know what authentication algorithm(s) and key(s) are acceptable to the FT from the IPUI with which it uses to identify itself and its knowledge of the subscription associated with the FT. Thus <<AUTH-TYPE>> should in most instances be acceptable to the FT (see also clause 6.5.4).

NOTE 3: The {AUTHENTICATION-REJECT} message may optionally include a sequence of <<AUTH-TYPE>> elements, whereby the position of the element in the sequence indicates the order of preference.

NOTE 4: It is envisaged that, in most applications, the FT will know what authentication algorithm(s) and key(s) are acceptable to the PT from the IPUI.

**Table 6.16: REJECT REASON, authentication of an FT**

| Code | Reject reason (from FT)                           | Actions open to PT |
|------|---|--------------------|
| 11   | No authentication algorithm                       | 1, 2               |
| 12   | Identified authentication algorithm not supported | 1, 2, 3            |
| 13   | Identified authentication key not available       | 1, 2, 3            |

#### Codes for Actions open to PT

- 1) Release call.
- 2) Proceed with call without authentication of the FT (see note 6).
- 3) Send new {AUTHENTICATION-REQUEST} with new <<AUTH-TYPE>> element. (This element may have been received in the {AUTHENTICATION-REJECT}.)

NOTE 5: No attempt is made in this clause to specify the prescribed actions, only the options are presented. A prescribed action may depend upon the <<REJECT REASON>> and the circumstances under which authentication of the FT was requested.

NOTE 6: The option to proceed without authentication is not recommended.

#### 6.5.2.3 Authentication of a PT type 2 procedure

This mechanism is used when the DECT Standard Authentication Algorithm #2 (DSAA2) is used. See clause 6.5.2.1 for the equivalent procedure to be used when the DECT Standard Authentication Algorithm (DSAA2) or a proprietary algorithm is used.

The authentication protocol proceeds as follows:

- 1) the FT sends an {AUTHENTICATION-REQUEST} to the PT. The <<AUTH-TYPE>> element in this message identifies the authentication algorithm and authentication key which are requested (by the FT) to be used by the PT;
- 2) upon receipt of the {AUTHENTICATION-REQUEST}, the PT examines the <<AUTH-TYPE>> element to see if it is acceptable. This element is defined to be acceptable if the PT can implement the authentication algorithm and has the authentication key identified in the element (see note 2);

- 3) if the <<AUTH-TYPE>> element is acceptable, the PT performs the security processes described in clause 4.3.6 and returns an {AUTHENTICATION-REPLY} message with the <<RES>> and <<RAND\_P>> elements computed using the identified authentication algorithm and authentication key;
- 4) if the <<AUTH-TYPE>> element is unacceptable, the PT returns an {AUTHENTICATION-REJECT} message. The <<REJECT REASON>> element in the message is set to indicate the reason for the rejection. The possible rejection reasons are listed in table 6.15. The {AUTHENTICATION-REJECT} message may optionally include an <<AUTH-TYPE>> element constructed by the PT (see note 3). This element identifies an authentication algorithm and an authentication key which is acceptable to the PT;
- 5) upon receipt of an {AUTHENTICATION-REJECT}, the FT examines the <<REJECT REASON>> and then takes a prescribed action (see note 4). This will be based on the reason for the rejection and the reason for requesting authentication.

NOTE 1: See also clause 4.3.6 for description of the security process operations to be performed in order to generate and evaluate the parameters transported over the air i/f procedure.

NOTE 2: It is envisaged that in most applications the FT will know what authentication algorithm(s) and key(s) are supported by the PT from its knowledge of the IPUI, so that <<AUTH-TYPE>> should always be acceptable to the PT (see also clause 6.5.4).

NOTE 3: The {AUTHENTICATION-REJECT} message may optionally include a sequence of <<AUTH-TYPE>> elements, whereby the position of the element in the sequence indicates the order of preference.

NOTE 4: No attempt is made in the present document to specify the prescribed actions, only the options are presented. A prescribed action may depend upon the <<REJECT REASON>> and the circumstances under which authentication of the PT was requested.

A list of possible actions is given in table 6.15.

#### Codes for Actions open to FT (table 6.15)

- 1) Release call.
- 2) Proceed with call without authentication of the PT (see note 5).
- 3) Send new {AUTHENTICATION-REQUEST} with new <<AUTH-TYPE>> element. (This element may have been received in the {AUTHENTICATION-REJECT}.)

NOTE 5: The option to proceed without authentication is not recommended.

#### 6.5.2.4 Authentication of an FT type 2 procedure

This procedure is used when the DECT Standard Authentication Algorithm #2 (DSAA2) is used. See clause 6.5.2.2 for the equivalent procedure to be used when the DECT Standard Authentication Algorithm (DSAA) or a proprietary algorithm is used.

The authentication protocol proceeds as follows:

- 1) the PT sends an {AUTHENTICATION-REQUEST} to the FT. The <<AUTH-TYPE>> element in the message identifies the authentication algorithm and authentication key which are requested (by the PT) to be used by the fixed network to derive the value RES in the {AUTHENTICATION-REPLY};
- 2) upon receipt of the {AUTHENTICATION-REQUEST}, the FT examines the <<AUTH-TYPE>> element to see if it is acceptable. The element is defined to be acceptable if the FT can return a <<RES>> and <<RAND\_F>> generated as described in clause 4.3.7 using the identified authentication algorithm and authentication key (see note 2);
- 3) if the <<AUTH-TYPE>> element is acceptable, the FT returns an {AUTHENTICATION-REPLY} message with the <<RES>> element computed using the identified authentication algorithm and authentication key;

- 4) if the <<AUTH-TYPE>> element is unacceptable, the FT returns an {AUTHENTICATION-REJECT} message. The <<REJECT REASON>> element in the message is set to indicate the reason for the rejection. The possible rejection reasons are listed in table 6.16. The {AUTHENTICATION-REJECT} message may optionally include an <<AUTH-TYPE>> element (see note 3). This element identifies an authentication algorithm and an authentication key which is acceptable to the fixed network (see note 4);
- 5) upon receipt of an {AUTHENTICATION-REJECT}, the PT examines the <<REJECT REASON>> and then takes a prescribed action (see note 5). This will be based on the reason for the rejection and the reason for requesting authentication. A list of possible actions is given in table 6.16.

NOTE 1: See also clause 4.3.7 for description of the security process operations to be performed in order to generate and evaluate the parameters transported over the air i/f procedure.

NOTE 2: It is envisaged that in most applications the PT will know what authentication algorithm(s) and key(s) are acceptable to the FT from the IPUI with which it uses to identify itself and its knowledge of the subscription associated with the FT. Thus <<AUTH-TYPE>> should in most instances be acceptable to the FT (see also clause 6.5.4).

NOTE 3: The {AUTHENTICATION-REJECT} message may optionally include a sequence of <<AUTH-TYPE>> elements, whereby the position of the element in the sequence indicates the order of preference.

NOTE 4: It is envisaged that, in most applications, the FT will know what authentication algorithm(s) and key(s) are acceptable to the PT from the IPUI.

#### Codes for Actions open to PT (table 6.16)

- 1) Release call.
- 2) Proceed with call without authentication of the FT (see note 6).
- 3) Send new {AUTHENTICATION-REQUEST} with new <<AUTH-TYPE>> element. (This element may have been received in the {AUTHENTICATION-REJECT}.)

NOTE 5: No attempt is made in this clause to specify the prescribed actions, only the options are presented. A prescribed action may depend upon the <<REJECT REASON>> and the circumstances under which authentication of the FT was requested.

NOTE 6: The option to proceed without authentication is not recommended.

### 6.5.3 Confidentiality protocols

The following protocol is used to start (or stop) the confidentiality service, identify the cipher algorithm (KSG) and cipher key used to provide the service, and to initiate the loading of the cipher key to the KSG and the exchange of MAC layer encryption mode control messages described in clause 6.4.6.

The protocol is as follows:

- 1) optionally the PT sends a {CIPHER-SUGGEST} message to the FT. The Y/N bit in the <<CIPHER-INFO>> element indicates whether the PT is suggesting that a ciphering session should be started (Y/N = 1), or whether it is suggesting that the ciphering session should be terminated (Y/N = 0). In the case of a suggestion to start ciphering, the <<CIPHER-INFO>> element also identifies the cipher algorithm (KSG) which is to be used (cipher algorithm identity) and the cipher key which is to be used to drive it (cipher key type and cipher key number). The FT may respond with a {CIPHER-REQUEST} (as described in 2)) or with a {CIPHER-REJECT} (see table 6.17);
- 2) the FT sends a {CIPHER-REQUEST} message to the PT (see note 1). The Y/N bit in the <<CIPHER-INFO>> element indicates whether the FT is requesting the start or termination of a ciphering session (see 1)). In the case of a request for starting ciphering, the <<CIPHER-INFO>> element in the message also identifies the cipher algorithm (KSG) which is to be used and the cipher key which is to be used to drive it. The FT establishes this cipher key for use in the selected KSG (see note 2);

- 3) upon receipt of the {CIPHER-REQUEST}, the PT examines the <<CIPHER-INFO>> element to see if it is acceptable. The <<CIPHER-INFO>> element is defined to be acceptable if the Y/N bit is consistent with the current cipher mode, and if the PT can implement the cipher algorithm and has the cipher key identified in the element;
- 4) if the <<CIPHER-INFO>> is acceptable (see note 3) and ciphering is requested (Y/N = 1), the PT establishes the cipher key for use in the KSG and initiates the transmission of the MAC layer START.REQ message defined in clause 6.4.6.2 (see note 4). If the call is already ciphered, the PT initiates the re-keying procedure (see clause 6.4.6.5) with transmission of the MAC layer STOP.REQ message defined in clause 6.4.6.2. If the <<CIPHER-INFO>> is acceptable and switching to clear mode is requested (Y/N = 0), the PT initiates the transmission of the MAC layer STOP.REQ message defined in clause 6.4.6.2);
- 5) if the <<CIPHER-INFO>> is unacceptable, the PT returns a {CIPHER-REJECT} message. The <<REJECT REASON>> element in the message is set to indicate the reason for the rejection. The possible reject reasons are listed in table 6.17. The {CIPHER-REJECT} message may optionally include an alternative <<CIPHER-INFO>> element constructed by the PT (see note 5). This element identifies a cipher algorithm and cipher key which can be used by the PT;
- 6) upon receipt of a {CIPHER-REJECT}, the FT examines the <<REJECT REASON>> element and then takes a prescribed action (see note 6) based on the reason for the rejection. A list of possible actions is given in table 6.17.

NOTE 1: It is important that the {CIPHER-REQUEST} is sent before the transfer of any C-plane data which is intended to be encrypted (for example dialled number).

NOTE 2: The FT sends DL\_ENC\_KEY-req primitive to the DLC layer. This primitive carries the selected cipher key and identifies the KSG which is to be used and the relevant MAC connection. Upon receipt of this primitive, the DLC stores the information (this is necessary for connection handover or for establishing additional connections for a broadband data link) and sends MAC\_ENC\_KEY-req primitive to the MAC layer. The MAC layer loads the received key for use with the identified KSG.

NOTE 3: It is envisaged that the FT will know the cipher algorithm and cipher key supported by the PT from the IPU (see also clause 6.5.4).

NOTE 4: To start/stop the ciphering, the PT sends DL\_ENC\_KEY-req and DL\_ENCRYPT-req primitives to the DLC layer. The first of these primitives carries the selected cipher key and the second identifies the KSG and MAC connection that are to be used. Upon receipt of these primitives, the DLC stores the information (necessary for connection handover or for establishing additional connections for a broadband data link) and sends MAC\_ENC\_KEY-req and MAC\_ENC\_EKS-req primitives to the MAC layer. Upon receipt of these the MAC layer loads the received key for use with the identified KSG and starts transmission of the START.REQ/STOP.REQ message defined in clause 6.4.6.2.

Once ciphering has been started/stopped at the PT, a MAC\_ENC\_EKS-cfm primitive is sent to the DLC which then sends a DL\_ENCRYPT-cfm to the NWK.

Once ciphering has been started/stopped at the FT, a MAC\_ENC\_EKS-ind is sent to the DLC which then sends a DL\_ENCRYPT-ind to the NWK.

NOTE 5: The {CIPHER-REJECT} message may optionally include a sequence of <<CIPHER-INFO>> elements, whereby the position of the element in the sequence indicates the order of preference.

NOTE 6: No attempt is made in the present document to specify the prescribed actions, only the options are presented.



**Table 6.17: REJECT REASON, request for ciphering**

| Code | Reject reason (from PT)                   | Actions open to FT |
|------|---|--------------------|
| 17   | No cipher algorithm                       | 1, 2               |
| 18   | Cipher algorithm not available            | 1, 2, 3            |
| 19   | Cipher key not available                  | 1, 2, 3, 4         |
| 20   | incompatible service                      | 1                  |
| Code | Reject reason (from FT)                   | Actions open to PT |
| 17   | No cipher algorithm                       | 1, 2               |
| 18   | Identified cipher algorithm not supported | 1, 2, 5            |
| 19   | Identified cipher key not available       | 1, 2, 5            |
| 20   | incompatible service                      | 1                  |

**Codes for Actions open to FT**

- 1) Release call.
- 2) Proceed with call without encryption.
- 3) Send new {CIPHER-REQUEST} with new <<CIPHER-INFO>> element.(This element may have been received in the {CIPHER-REJECT}).
- 4) Authenticate the PT (and thereby establish a new DCK) and then send new {CIPHER-REQUEST}.
- 5) Send new {CIPHER-SUGGEST} with a new <<CIPHER-INFO>> element. (This element may have been received in the {CIPHER-REJECT}).

**6.5.4 Access-rights protocols**

A PT may establish which authentication algorithm and cipher algorithm will be used for a particular system (subscription) as part of the access-rights exchange. The protocol is as follows:

- 1) the PT should include an <<AUTH-TYPE>> and a <<CIPHER-INFO>> in the {ACCESS-RIGHTS-REQUEST} message. The <<AUTH-TYPE>> element, if included, identifies the authentication algorithm which the PT wishes to use in conjunction with the particular system (subscription). The <<CIPHER-INFO>> element identifies the cipher algorithm which it wishes to use (see note 1);
- 2) upon receipt of an {ACCESS-RIGHTS-REQUEST} message, the FT (or a management element associated with the FT) examines the <<AUTH-TYPE>> and <<CIPHER-INFO>> elements to see if they are acceptable. An element of this type is defined to be acceptable if the FT supports use of the identified algorithm;
- 3) if the <<AUTH-TYPE>> and <<CIPHER-INFO>> elements are acceptable, the FT returns these elements in an {ACCESS-RIGHTS-ACCEPT} message (see note 2);
- 4) if the <<AUTH-TYPE>> or <<CIPHER-INFO>> element is not acceptable, the FT returns an {ACCESS-RIGHTS-REJECT}. The <<REJECT REASON>> element in the message indicates the reason(s) why the element(s) is (are) not acceptable. The possible reasons are given in table 6.18. The {ACCESS-RIGHTS-REJECT} may optionally include an <<AUTH-TYPE>> or a <<CIPHER-INFO>> element which identifies an algorithm acceptable to the FT (see note 3).

NOTE 1: If the PT does not support an authentication algorithm, respectively a cipher algorithm, then an <<AUTH-TYPE>> element, respectively a <<CIPHER-INFO>> element, is not sent.

NOTE 2: If an <<AUTH-TYPE>>, respectively a <<CIPHER-INFO>>, element is accepted, then the PT and the FT (or a management centre associated with the FT) assigns the identified algorithm to the subscription details.

NOTE 3: The FT may optionally include a preferential sequence of <<AUTH-TYPE>>, respectively <<CIPHER-INFO>>, elements.

**Table 6.18: REJECT REASON, access rights request**

| Code | Reject reason (from FT)                | Actions open to PT |
|------|--|--------------------|
| 11   | No authentication algorithm            | 1                  |
| 17   | No cipher algorithm                    | 1                  |
| 12   | Authentication algorithm not supported | 1, 2               |
| 18   | Cipher algorithm not supported         | 1, 3               |

**Codes for actions open to PT**

- 1) Abort access attempt.
- 2) Send new {ACCESS-RIGHTS-REQUEST} with new <<AUTH-TYPE>> element. (This element may have been received in the {ACCESS-RIGHTS-REJECT}).
- 3) Send new {ACCESS-RIGHTS-REQUEST} with new <<CIPHER-INFO>> element. (This element may have been received in the {ACCESS-RIGHTS-REJECT}).

**6.5.5 Key numbering and storage****6.5.5.0 General**

Authentication keys may be of three types, UAK, AC or UAK/UIP. Cipher keys may be of two types, DCK and SCK.

**6.5.5.1 Authentication keys**

Authentication keys are associated with a particular IPUI or IPUI/PARK pair. Separate sets of UAK and AC authentication keys are held for each IPUI or IPUI/PARK pair, and number ranges are associated with each key type. A UIP may be associated with each UAK.

Thus a particular IPUI or IPUI/PARK pair will have associated to it a range of UAKs (for example UAK1, ..., UAK4) and a range of ACs (for example AC1, AC2, AC3). Each range comprises a maximum number of 8 elements. With each UAK a UIP may be associated.

The key allocation procedure, described in clause 6.5.6, may be used to transform an initial AC into a UAK.

Within a PP UAKs and ACs shall be held in non-volatile memory. ACs are manually entered to non-volatile storage. With each UAK a UIP may be associated. Such a UIP is not stored within the PP but is entered every time it is required for the user authentication service.

Within an <<AUTH-TYPE>> element the 4 bit Auth.key type field identifies the authentication key type (currently, UAK, AC or UAK/UIP) and the 4 bit field Auth.key nr. identifies the number of the key of that particular type. The most significant bit of the Auth. key nr. identifies if the key is associated with the IPUI (MSB = 0) or the IPUI/PARK pair (MSB = 1). If the authentication key is of type UAK/UIP the Auth.key nr. refers to the UAK portion of the key.

Within an allocation-type element (see clause 6.5.6) the 4 bit AC number field identifies the AC which is to be used to generate a UAK. The most significant bit of the AC number identifies if the key is associated with the IPUI (MSB = 0) or the IPUI/PARK pair (MSB = 1). The newly generated UAK is to be assigned the number indicated in the 4 bit UAK number field.

**6.5.5.2 Cipher keys**

Cipher keys are associated with a particular IPUI or IPUI/PARK pair. Separate SCKs and DCKs are stored for each IPUI or IPUI/PARK pair. For each IPUI or IPUI/PARK pair at most 8 keys of each type may be stored (see note 4).

Thus a particular IPUI or IPUI/PARK pair will have associate to it a range of SCKs (for example SCK1, ..., SCK4) and a range of DCKs (for example DCK1). Each range comprises a maximum number of 8 elements (see note 1).

Within a PP SCKs and DCKs shall be held in non-volatile memory. SCKs are manually entered into this memory. DCKs may be automatically updated as part of the authentication of the PT procedure.

Within a <<CIPHER-INFO>> element the 4-bit cipher key type field identifies the type of a cipher key (currently, SCK or DCK) and the 4-bit cipher key nr. field identifies the number of the key of that particular type. The most significant bit of the cipher key number identifies if the key is associated with the IPUI (MSB = 0) or the IPUI/PARK pair (MSB = 1).

Within an <<AUTH-TYPE>> element the UPC bit indicates whether the DCK derived as part of the authentication of the PT should be stored or not. If UPC is set to 1 the DCK is stored under the number indicated in the 4 bit cipher key nr. field. If UPC is set to 0 the DCK is not stored (see notes 2, 3 and 4). Assignment of a DCK to a particular number causes the DCK currently held under that number to be overwritten.

Within an <<AUTH-TYPE>> element the DEF bit indicates whether the calculated key as part of the authentication of the PT is a default cipher key or not. If DEF is set to 1 the calculated key is stored under the DefCK-index indicated in the default cipher key index field. Assignment of a DefCK to a particular DefCK-index causes the DefCK currently held under that number to be overwritten.

NOTE 1: Static keys are not suitable for use on large systems where many PPs may have to access many FPs. They have been introduced for systems which may not support the authentication services. An example of such a system might be a residential system which supports confidentiality but not the authentication mechanism. In this case an SCK would be unique to a particular PT/FT pair forming a residential system.

NOTE 2: Where authentication of a PT is applied and followed by confidentiality, the normal mode of operation is to derive a new DCK and then use this key for the confidentiality service. Thus in the {AUTHENTICATION-REQUEST} the UPC bit in the <<AUTH-TYPE>> is set to 1 and the cipher key nr. field is set (for example) to 0001. In the {CIPHER-REQUEST} the cipher key type field in the <<CIPHER-INFO>> element is set to indicate DCK and the cipher key nr. field is set to 0001.

NOTE 3: When the system provides central control of keys, a DCK may be stored for use on a later call (avoiding the need to repeat the authentication of the PT procedure). However, if there is any doubt about the current value of a stored DCK, authentication of the PT is applied and a new DCK generated.

NOTE 4: For most applications it is envisaged that only one cipher key will be needed for each IPUI. For those systems which implement authentication of the PT this will be of type DCK, for those systems which do not it will be of type SCK.

## 6.5.6 Key allocation

### 6.5.6.1 Introduction

The purpose of this clause is to define a method for the initial allocation of a UAK using an over-the-air protocol. The UAK is to be associated with one IPUI or IPUI/PARK pair and the mechanism requires that an AC is already associated with that IPUI or IPUI/PARK pair.

The mechanism has been introduced for those systems where distribution of a 128 bit UAK to the user on paper or within a DAM is not considered to be feasible. However, the following points should be noted concerning the security of the protocol:

- 1) other methods for UAK distribution (for example in a DAM or on paper) are to be preferred (see the discussion in clause 7);
- 2) the AC from which the UAK is derived should be at least 32 bits long;
- 3) the way in which the AC is initially distributed does not form part of the present document (see note 1);
- 4) this mechanism is not to be used for roaming key allocation (see note 2).

NOTE 1: The AC is manually entered into the PP. Possible means of distribution include on paper or verbally (perhaps over a telephone line).

NOTE 2: In a roaming environment visitors are provided with the session key(s) KS (and KS') as described in clause 4.4.3.2.

### 6.5.6.2 UAK allocation (DSAA algorithm)

The protocol for over-the-air allocation of a UAK is defined below. The {KEY-ALLOCATE} message is summarized in table 6.19. All other messages used in the protocol are the authentication messages defined in clause 6.3.2 (see note 2).

- 1) the FT sends a {KEY-ALLOCATE} message to the PT. The Allocation-type information element in the message identifies the relevant authentication algorithm and the number of the AC which is to be used to derive the UAK, as well as the number which is to be assigned to the derived UAK. The {KEY-ALLOCATE} message includes the IE <<RS>> (carrying the RS parameter) and <<RAND>> (carrying RAND\_F);
- 2) upon receipt of the {KEY-ALLOCATE} message the PT examines the Allocation-type element to see if it is acceptable. This element is defined to be acceptable if the PT can implement the authentication algorithm and has the AC identified in the element;
- 3) if the Allocation-type element is unacceptable, the PT returns an {AUTHENTICATION-REJECT} message (see clause 6.5.2.1, item 4), and note that the possible reject reasons are those listed in table 6.16 with codes 11, 12 and 13);
- 4) if the allocation-type element is acceptable, the PT returns an {AUTHENTICATION-REQUEST} to the FT. The <<AUTH-TYPE>> element in this message identifies the authentication algorithm indicated in the accepted allocation-type, the Auth.key type is set to indicate AC and the Auth.key nr. is the number indicated in the accepted allocation-type. The value <<RES>> (RES1) in the message is calculated from the <<RAND>> (RAND\_F) and <<RS>> (RS) elements received in the {KEY-ALLOCATE} using the authentication algorithm and key identified in the accepted allocation-type;
- 5) upon receipt of the {AUTHENTICATION-REQUEST}, the FT checks <<RES>> (RES1), to complete the authentication of the PT. If the check is not successful the PT authentication has failed and the FT shall not continue with the procedure. If the check is successful the FT returns an {AUTHENTICATION-REPLY}. In this message, the element <<RS>> is the same as that in the original {KEY-ALLOCATE}. The element <<RES>> (RES2) is computed with this value of RS and the RAND (RAND\_P) received in the {AUTHENTICATION-REQUEST} using the authentication algorithm and key identified in the received <<AUTH-TYPE>>.

NOTE 1: It is allowed not to include the <<RS>> IE in the {AUTHENTICATION-REPLY} since RS should be the same transmitted in the {KEY-ALLOCATE} message.

The authentication session key value KS' computed during the process of computing <<RES>> (RES2) (see clause 5.2.2) is the derived UAK This is assigned to the UAK number identified in the original allocation-type.

Its status shall be marked as unconfirmed (see note 3);

- 6) upon receipt of the {AUTHENTICATION-REPLY}, the PT checks <<RES>> (RES2), in order to complete the authentication of the FT. If the check fails, the PT shall not continue with the procedure and shall not accept the key allocation. If the check succeeds the PT stores the authentication session key KS', computed during the process of computing <<RES>> (RES2) (see clause 5.2.2), under the UAK number identified in the original allocation-type. The PT erases the used AC.

With regard to clause 4.5.2, the UAK needs to be derived from the session key KS' in the following way:

$$\text{UAK}[i] = \text{KS}'[i \text{ MODULO } \text{LEN\_KS}'], 0 \leq i \leq \text{LEN\_UAK} - 1.$$

NOTE 2: It should be observed that the <<RES>> (RES1) element in the PT originated {AUTHENTICATION-REQUEST} is used only in this protocol. It is not required in this message for the authentication protocols defined in clause 6.5.2.

NOTE 3: The FT (or a key management facility associated to the FT) retains both the AC and the UAK (with unconfirmed status) until the PT has been successfully authenticated using the UAK. Once a successful authentication of the PT has been achieved using the UAK, the AC is erased and the unconfirmed status marking can be removed from the UAK.

**Table 6.19: Key allocation message**

| Key Allocation Message | Message Type | Elements   |
|------------------------|--------------|--|
| {KEY-ALLOCATE}         | 01000010     | <<Allocation-type>><br><<RAND>> (RAND_F)<br><<RS>> |

### 6.5.6.3 UAK allocation (DSAA2 algorithm)

The procedure is identical to clause 6.5.6.2 with the following differences:

- In step 1, the IE <<RS>> in {KEY-ALLOCATE} transports the parameter  $RS_{128}$  that has 128 bits. This  $RS_{128}$  is the one to be used in the PT authentication only (input to process A11).
- In step 4, the {AUTHENTICATION-REQUEST} message shall include the IE <<RAND>> carrying a RAND\_P parameter generated by the PT after the reception of the {KEY-ALLOCATE}. This RAND\_P will be used for both the PT and FT authentications.

NOTE 1: This is the only case when a RAND\_P parameter is reused for the execution of two processes (A12, A22) in DSAA2 based security.

- In step 5, the PT authentication checking performed by the FT shall be based on DSAA2 (using the RAND\_P parameter). The {AUTHENTICATION-REPLY} message shall compulsorily include the IE <<RS>> that shall carry a "fresh"  $RS_{128}$  parameter, independent to the one sent with the {KEY-ALLOCATE}. This  $RS_{128}$  shall be used for FT authentication and generation of the UAK.

The second  $RS_{128}$  sent with the {AUTHENTICATION-REPLY} shall be always freshly generated and the process A21 shall necessarily be executed at both sides. Process A22 shall also be necessarily executed at both sides as in any FT authentication.

NOTE 2: It is up to the application profile to mandate if the first  $RS_{128}$  parameter sent with the {KEY-ALLOCATE} should be "fresh" (forcing a new execution of process A11), or if it may be reused from previous PT authentications.

Authentication of PT shall be performed as described in clause 4.3.6 (type 2 procedure), however the message {KEY-ALLOCATE} carries the parameters and perform the role normally done by the {AUTHENTICATION-REQUEST} message in clause 4.3.6 description, and the message {AUTHENTICATION-REQUEST} carries the parameters and perform the role normally done by the {AUTHENTICATION-REPLY} in the description of clause 4.3.6.

Authentication of FT shall be performed exactly as described in clause 4.3.7 (type 2 procedure) with the following differences:

- The parameter RAND\_P is also re-used for the PT authentication.
- The  $RS_{128}$  sent with the {AUTHENTICATION-REPLY} shall be always freshly generated and process A21 shall necessarily be executed at both sides.

## 6.6 DLC layer procedures

### 6.6.1 Background

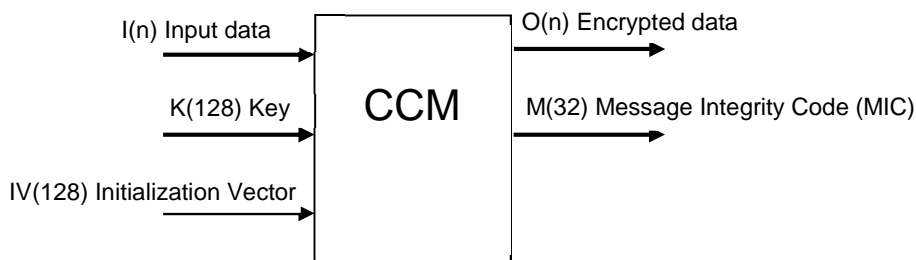
The purpose of this clause is to specify all the DLC layer processes needed for the provision of data confidentiality over the air interface.

## 6.6.2 CCM Authenticated Encryption

### 6.6.2.0 CCM overview

The DECT CCM Authenticated Encryption is intended to operate at DLC layer. CCM provides both packet authentication and encryption, and is based on IETF RFC 3610 [11]. The DECT implementation of CCM uses the Advanced Encryption Standard (AES) [10] as 128 bit block cipher (IETF RFC 3610 [11] may, in theory, use other ciphers). Apart from that and from the definition of the parameters of the Initialization Vector, it follows exactly the implementation proposed by the RFC.

CCM provides strong encryption of a DLC U-plane packet, and adds a Message Integrity Code (MIC) that is used to guarantee the authenticity of the packet. The CCM security model is shown in figure 6.18.



**Figure 6.18: CCM security process overview**

The internal structure and algorithms of CCM are defined in annex N of the present document.

CCM is used in DLC service LU14, where the authenticated and encrypted packet is the DLC SDU (see ETSI EN 300 175-4 [4], clause 11.16).

### 6.6.2.1 CCM operation

CCM encrypts a packet of arbitrary length  $n$  with the help of a Key (K) and an Initialization Vector (IV). It provides an output message (O ( $n$ )) of the same length, plus a Message Integrity Code (MIC). In the DECT implementation, the following lengths have been chosen:

- The Key has 128 bits.
- The Initialization Vector has 128 bits.
- The Message Integrity Code (MIC) has 32 bits.
- From algorithm point of view, the packet to be encrypted may have any size from zero to 65 535 octets.
- The zero length case means that it is possible to generate a MIC, or a sequence of MICs, without any message.

NOTE 1: The option of generating MICs without message may be used, for instance, in the design of authenticated stay-alive procedures.

NOTE 2: Due to the implications in traffic, the option of generating MICs without message should be used only when specifically indicated and according to the rules given in an application profile. Otherwise it should not be used.

- DECT implementation shall assume that the message length is multiple of 8 bits (or zero length).

Bytes 1 to 13 of the Initialization Vector are called the "nonce".

In DECT LU14 operation the Message Integrity Code (MIC) of 32 bits is concatenated at the end of the encrypted message. The resulting structure is transported using LU10 Enhanced Frame Relay Service.

### 6.6.2.2 Key management

By cryptographic reasons, the Keys used in CCM should be used only once for each value of the nonce (bytes 1-13 of the initialization vector). In practice, it means that a new key should be generated and used each time the CCM sequence is reset (see clause 6.6.2.4).

In addition to that, the key used for CCM authenticated encryption, should not be reused by any other security process. This means that if both CCM and DSC or DSC2 MAC encryption are used, different keys should be used.

The generation and storage of Keys for CCM is described in clause 6.2.3 of the present document.

The activation of new keys for CCM is performed as described in clause 6.3.7.

### 6.6.2.3 CCM Initialization Vector

#### 6.6.2.3.0 CCM Initialization Vector: overview

In the following clauses, octets and bits will be numbered starting with 0, as is the practice in IETF RFC 3610 [11].

The CCM Initialization Vector consists of 16 bytes (128 bits). Octets 1 to 13 are called the "nonce" (13 bytes).

| 0                    | 1                   | 2-7                        | 8-11                         | 12                  | 13                | 14-15                   |
|----------------------|---------------------|----------------------------|------------------------------|---------------------|-------------------|-------------------------|
| CCM Flag<br>(8 bits) | N_RESV.<br>(8 bits) | Packet number<br>(48 bits) | DECT identities<br>(32 bits) | dir/LCN<br>(8 bits) | Fixed<br>(8 bits) | SDU length<br>(16 bits) |

| Octet  | CCM IV fields     | Size (bytes) | Description   |
|--|-------------------|--------------|---|
| 0  | Flags             | 1            | CCM flag field (see clause 6.6.2.3.1). Set to "09"H   |
| 1  | N_RESERVED        | 1            | Reserved Nonce. It shall be set to "00"H  |
| 2-7  | CCM Packet number | 6            | CCM Packet number (big endian format)   |
| 8-11   | PARI/PMID         | 4            | DECT identities of the sending and receiving sides, coded as shown in clause 6.6.2.3.2  |
| 12   | dir/LCN           | 1            | Link Connection Number (LCN) in bits 0,1,2. Bits 3-6 shall be set to "0000"B. Bit 7 is set "0"B in direction from FT to PT and set to "1"B in direction from PT to FT. See clause 6.6.2.3.3 and notes 1 and 2 |
| 13   | Fixed             | 1            | Known (fixed) value. It shall be set to "00000000"B   |
| 14-15  | SDU length        | 2            | SDU size in octets (big endian format). Min value "1"H, max value "FFFF"H   |
| NOTE 1: The LCN is equal to the constant part of the ECN. See definition in ETSI EN 300 175-4 [4], clause 10.2.4.2.  |                   |              |   |
| NOTE 2: For very large systems with connection handover capabilities, or when by any other reason the LCN may change, the initial LCN used by the connection at link creation shall be used. |                   |              |   |

The coding of the Initialization Vector for C/L services (such as C/L downlink multicast) is defined in clause 6.6.2.7.6.

The coding of the Initialization Vector for Service Channels is defined in clause 6.6.2.8.1.

#### 6.6.2.3.1 CCM Initialization Vector: first byte

The byte 0 of the IV is built as follows:

| CCM IV byte 0 (flag) | Bits | Description                 | Value used in DECT |
|----------------------|------|-----------------------------|--------------------|
| Reserved             | 7    | Reserved, set to "0"        | 0                  |
| Adata                | 6    | Adata                       | 0                  |
| M'                   | 5-3  | M' field is set to (M-2)/2  | "001"B             |
| L'                   | 2-0  | L' = L-1 (set to L=2, L'=1) | "001"B             |

The used configuration in IETF RFC 3610 [11] and in DECT is A = "0", L = "2", M = "4", L' = "1", M' = "1".

### 6.6.2.3.2 CCM Initialization Vector: bytes 8-11

Bytes 8 to 11 shall carry a 32 bit stream containing the DECT identities of both peers involved in the link. It shall be coded with the Fixed Part Identity followed by the PMID.

The Fixed Part identity consists on the 12 least significant bits of the PARI.

The coding of such octets shall be as follows:

| CCM IV byte  | Bits | Description   |
|--|------|---|
| 8  | 0-7  | PARI, 8 most significant bits (of the 12 least significant bits of the PARI). |
| 9  | 4-7  | PARI, 4 least significant bits.   |
| 9  | 0-3  | PMID, 4 most significant bits.  |
| 10   | 0-7  | PMID, bits 5-12 (see note).   |
| 11   | 0-7  | PMID, 8 least significant bits.   |
| NOTE: Bit numbering for the PMID is based on standard DECT practice (bit 1 to bit 20). |      |   |

The coding of bytes 8-11 for C/L services (such as C/L downlink multicast) is defined in clause 6.6.2.7.6.2.

### 6.6.2.3.3 CCM Initialization Vector: bytes 12

For bidirectional links, each direction shall code its IV independently (therefore the IVs will be different).

The coding of such octets shall be as follows:

| CCM IV byte | Bits | Description  |
|-------------|------|--|
| 12          | 0-2  | LCN, Link Connection Number  |
| 12          | 3-6  | Reserved, encoded to "0000"B   |
| 12          | 7    | PP transmission direction. For C/O downlink links (FP to PP) shall be set to "0"B and for C/O uplink links (PP to FP) shall be set to "1"B |

The coding of byte 12 for C/L services (such as C/L downlink multicast) is defined in clause 6.6.2.7.6.3.

The coding of byte 12 of the Initialization Vector for Service channels is defined in clause 6.6.2.8.1.1.

### 6.6.2.4 CCM Sequence Number

The CCM packets are numbered with 6 bytes. (0...281474976710655). The numbering is coded as follows:

- CCM Packed number is defined as an extension to 48 bit of the DLC SN used to number the PDUs.
- The least significant bits of the CCM Packet numbers are equal to the DLC SN of the PDU that carries the first segment of the SDU.
- 8 or 9 bits (if FU10a) may be explicitly transported in the DLC SNs.
- The remaining bits 8-47 (or 9-47 if FU10a) are implicit, not transmitted, and are known by both peers. The implicit part of the counter is incremented every time the DLC SN overflows.

NOTE: Therefore, the Packet numbers used in the CCM initialization vectors are not, in general, consecutive.

At receiver side, the Packet number shall be rebuilt as follows:

- Lower 8 or 9 bits shall be taken from the DLC SN of the first PDU that carries the CCM SDU.
- Bits 8 (or 9) to 47 are not transmitted and should be built based on previous Packet number, incremented if needed when the lowest bits overflow.



### 6.6.2.5 CCM Start and Stop

For DLC services using CCM, the encryption mode starts from the beginning with the setting of the link. It is not allowed the transmission in clear. In order to do that, a Derived Key generated by the authentication process and suitable for CCM, should exist before the Virtual Call setup procedure. This Key will be the initial key of the CCM. If the Key does not exist, the CC setup process shall fail with the proper error code. Additional Security IEs may be added to CC setup messages for call (or VCs) invoking DLC services with CCM.

Therefore the NWK MM Procedures "Cipher Activation" and the MAC procedures "Encryption Control" are not necessary for setup of releasing a call (or VC) using CCM.

### 6.6.2.6 CCM Sequence resetting and re-keying

This feature is for further study.

### 6.6.2.7 CCM encryption for multicast channels

#### 6.6.2.7.0 General

This clause describes how to use CCM encryption to cipher the data transmitted over multicast channels.

#### 6.6.2.7.1 Applicable types of multicast channels and identifiers

A multicast channel is defined as a C/L downlink channel addressed to a group of PPs.

This feature has been created for the encryption of the multicast channels of DECT Ultra Low Energy (ULE multicast channels). However it may also be used with other types of multicast channels.

It shall be assumed that in a system there may exist  $x$  multicast channels and that a PT may be subscribed to a maximum of  $y$  of such channels. Be "X" the system level identifier of the multicast channel and "Y" a local (PT) identifier of the multicast channel instance. Number "X" is a system level identifier, allocated when the multicast channel is created and common to all PTs. The local number "Y" is allocated when a PT is subscribed to a multicast channel and has only local meaning for that PT-FT pair.

NOTE: See ETSI EN 300 175-3 [3], clause 9.1.4.7.3.

#### 6.6.2.7.2 Process for encryption of multicast channels

For the CCM encryption of multicast channels, the same general mechanism applicable to the encryption of C/O channels, and defined in this clause 6.6.2, shall apply, with specific differences regarding key, sequence numbers, and initialization vector as described in clauses 6.6.2.7.3 to 6.6.2.7.7.

All provisions of clauses 6.6.2.1 and 6.6.2.4 shall apply.

Regarding clause 6.6.2.5, the key for multicast should exist and should have been sent to at least one subscribed PPs before any transmission of U-plane data over the multicast channel.

#### 6.6.2.7.3 DLC service for encrypted multicast channels

DLC service LU14 shall be used with a separate instance for each system-level multicast channel (identified by the "X" number). The DLC LU14 service shall be in this case unidirectional.

Frame type FU10a shall be used.

Each LU14 multicast instance shall use independent sequence numbers, that shall also be independent of the sequence numbers used in the LU14 instances for C/O links.

#### 6.6.2.7.4 Encryption key for multicast channels

Each multicast instance shall use an independent encryption key. Such keys shall be auxiliary keys generated by the Fixed Part (see clause 6.2.3.2) and transmitted to the PP by means of the procedures for key allocation (FT initiated) and key retrieval (PT initiated) described in clause 6.3.8 of the present document.

### 6.6.2.7.5 CCM and DLC sequence numbers

When a multicast channel is created, the associated LU14 instance shall be opened by the FP with an associate CCM sequence number (6 bytes). The lowest 9 bits of such sequence number shall also be the DLC PDU sending sequence number.

The FP may start the CCM sequence at any arbitrary number.

When a PP is subscribed to a multicast channel, the FP shall transmit to the PP the current number of the CCM sequence by means of the key allocation (FT initiated) procedure (see clause 6.3.8).

If a PP misses the CCM sequence, it may request to the FT the current number by means of the key retrieval (PT initiated) procedure (see clause 6.3.8).

### 6.6.2.7.6 Initialization Vector for multicast channels

#### 6.6.2.7.6.0 CCM Initialization Vector: overview

The Initialization Vector for the multicast channel shall be coded as follows:

|                      |                     |                            |                              |                     |                   |                         |
|----------------------|---------------------|----------------------------|------------------------------|---------------------|-------------------|-------------------------|
| 0                    | 1                   | 2-7                        | 8-11                         | 12                  | 13                | 14-15                   |
| CCM Flag<br>(8 bits) | N_RESV.<br>(8 bits) | Packet number<br>(48 bits) | DECT identities<br>(32 bits) | dir/LCN<br>(8 bits) | Fixed<br>(8 bits) | SDU length<br>(16 bits) |

| Octet | CCM IV fields     | Size (bytes) | Description   |
|-------|-------------------|--------------|---|
| 0     | Flags             | 1            | CCM flag field (see clause 6.6.2.7.6.1). It shall be set to "09"H (this coding is identical to C/O links).              |
| 1     | N_RESERVED        | 1            | Reserved Nonce. It shall be set to "00"H (this coding is identical to C/O links).                                       |
| 2-7   | CCM Packet number | 6            | CCM Packet number (big endian format).  |
| 8-11  | PARI/multicast    | 4            | DECT identities of the sending FP (PARI) plus number of the multicast channel "X" coded as shown in clause 6.6.2.7.6.2. |
| 12    | dir/LCN           | 1            | These bits shall be coded to '00001000'B (value is different from C/O links. See clause 6.6.2.7.6.3).                   |
| 13    | Fixed             | 1            | Fixed value. It shall be set to "00000000"B (this coding is identical to C/O links).                                    |
| 14-15 | SDU length        | 2            | SDU size in octets (big endian format). Min value "1"H, max value "FFFF"H.  |

#### 6.6.2.7.6.1 CCM Initialization Vector: first byte

The byte 0 of the IV is built as follows:

| CCM IV byte 0 (flag) | Bits | Description                 | Value used in DECT |
|----------------------|------|-----------------------------|--------------------|
| Reserved             | 7    | Reserved, set to "0"        | 0                  |
| Adata                | 6    | Adata                       | 0                  |
| M'                   | 5-3  | M' field is set to (M-2)/2  | "001"B             |
| L'                   | 2-0  | L' = L-1 (set to L=2, L'=1) | "001"B             |

The used configuration in IETF RFC 3610 [11] and in DECT is A = "0", L = "2", M = "4", L' = "1", M' = "1".

#### 6.6.2.7.6.2 CCM Initialization Vector: bytes 8-11

Bytes 8 to 11 shall carry a 32 bit stream containing the DECT identities of the Fixed Part plus the system-wide identifier of the multicast channel "X".

The Fixed Part identity consists on the 12 least significant bits of the PARI.

The coding of such octets shall be as follows:

| CCM IV byte   | Bits | Description   |
|---|------|---|
| 8   | 0-7  | PARI, 8 most significant bits (of the 12 least significant bits of the PARI). |
| 9   | 4-7  | PARI, 4 least significant bits.   |
| 9   | 0-3  | Shall be coded to '0000'B.  |
| 10  | 0-7  | Multicast channel nr. "X", 8 most significant bits (see note).                |
| 11  | 0-7  | Multicast channel nr. "X", 8 least significant bits.                          |
| NOTE: If "X" has less than 16 bits, the unused most significant bits shall be coded to '0'. |      |   |

#### 6.6.2.7.6.3 CCM Initialization Vector: bytes 12

The coding of such octets shall be as follows:

| CCM IV byte | Bits | Description                                 |
|-------------|------|---|
| 12          | 0-2  | Shall be set to '0'                         |
| 12          | 3-6  | Reserved, encoded to "0001"B                |
| 12          | 7    | For C/L downlink links shall be set to "0"B |

#### 6.6.2.7.7 Security provisions regarding the key

By fundamental security considerations, the following requirement shall apply:

- 1) The FT should change the key at least one time for a complete round of the CCM sequence numbers.

NOTE 1: It should be noted that for many applications this timer may be irrelevant (it may take years).

The following additional provisions are recommended:

- 2) The key should not be reused for different multicast channels.
- 3) When the PT requests the KEY, by means of the PT initiated procedure, the FT should authenticate it -by means of an interleaved PT authentication procedure- before sending the {MM-INFO-SUGGEST} containing the key.

NOTE 2: The previous authentication may also be used -but not necessarily- for changing the primary key for CCM or for MAC encryption.

NOTE 3: An application profile may mandate these additional recommendations.

#### 6.6.2.8 CCM encryption for service channels

##### 6.6.2.8.0 General

From security point of view, a service channel is defined as any point to point channel carrying data requiring CCM encryption different from the regular U-plane transmission.

EXAMPLE: Channels exchanged over the C-plane using IEs such as <<IWU-to-IWU>> or <<IWU-packet>> as transporting containers.

When CCM encryption of service channels is required, they shall be encrypted with independent CCM contexts using the same key as regular C/O channels and a different initialization vector.

Each service channel is assumed to have its own CCM sequence. The sequence is reset when the key is changed, exactly as happens with the primary U-plane channel.

The present document provides Initialization Vectors for up to 8 instances of CCM encrypted ancillary channels. The application profile should indicate which instance has to be used for each service channel defined in the application.

The nomenclature AUX0 to AUX7 may be used in the application profile to indicate which IV has to be used.

### 6.6.2.8.1 Initialization Vector for service channels

#### 6.6.2.8.1.0 CCM Initialization Vector: general

The IV shall be as described in clause 6.6.2.3 except octet 12 that shall be coded as indicated in clause 6.6.2.8.1.1.

#### 6.6.2.8.1.1 CCM Initialization Vector: byte 12

The coding of such octets shall be as follows:

| CCM IV byte | Bits | Description   |                               |
|-------------|------|---|-------------------------------|
| 12          | 0-2  | Instance of the service channel.  | 0 to 7 instances AUX0 to AUX7 |
| 12          | 3-6  | Shall be encoded to "0010"B.  |                               |
| 12          | 7    | PP transmission direction. For C/O downlink links (FP to PP) shall be set to "0"B and for C/O uplink links (PP to FP) shall be set to "1"B. |                               |

EXAMPLE 1: Use of bits 0-2 showing AUX4 as the instance being coded

```
Bits  7 6 5 4 3 2 1 0
      - - - - - 1 0 0
```

EXAMPLE 2: Use of bits 3-6

```
Bits  7 6 5 4 3 2 1 0
      - 0 0 1 0 - - -
```

## 6.7 Security meta-procedures

### 6.7.1 General

By "security meta-procedures" it is meant procedures related to system security that are fundamentally controlled at Management Entity (ME) level. These meta-procedures make use of MAC and NWK procedures invoked in the proper way and sequence, and normally include timers and strategies to be performed by the ME.

### 6.7.2 Re-keying

#### 6.7.2.1 Aim and strategy

The Re-keying is a mechanism consisting on the periodic and regular change of the Cipher Key of an ongoing call, service call, or virtual connection in order to improve the security. The fundamental aim of the re-keying is to increase the computer resources needed for a brute-force attack to the cipher and/or the authentication algorithms. The re-keying strategy achieves its objectives if the time required by a potential hacker to break the algorithms with its available computer resources is significantly larger than the re-keying timer.

The re-keying is fundamentally intended to protect the relatively weak cipher algorithm DSC. The protection provided by the re-keying is not comparable to the protection provided by the use of stronger ciphers (such as DSC2), and this should be the primary route for security concerned applications. Nevertheless, it is believed that DSC combined with the re-keying strategy is effective against attacks attempting real-time phone tapping of DECT communications performed by regular hackers with their expected computer resources.

#### 6.7.2.2 Re-keying procedure

This re-keying ME procedure consists on the periodic modification of the cipher key used for encryption during an ongoing call and within a maximum time defined by a timer. The call may be a voice call, a service call, a virtual call or any other virtual connection.

To implement the re-keying procedure, the FP shall perform periodically authentication of PP procedures with generation and storage of a new DCK followed by Cipher switching procedures, in a way that between the generation and the last use of such DCK there is never a longer time than a given timer.

The timers <MM\_re-keying.1> and <MM\_re-keying.2> provided in clause I.1 are given as recommendations. The timer <MM\_re-keying.2> is only intended to be used only in combination with DSAA2 and DSC2 (in case of repeaters, only if all segments use these algorithms). With DSC or in any other case the timer <MM\_re-keying.1> is the recommended choice.

In absence of specific provisions on the matter in the applicable application profile, or if such profile does not exist, or only contains a reference to the present document, then it shall be understood that timers <MM\_re-keying.1> and <MM\_re-keying.2> shall be used.

For the purposes of <MM\_re-keying.1> and <MM\_re-keying.2>, the generation of the key is assumed to happen at the FT sending of the {AUTHENTICATION-REQUEST} message, and the last use of the key is assumed to happen at the FT sending of a {CIPHER-REQUEST} message that is confirmed by the reception of a MAC START.GRANT message.

The authentication procedure shall be executed using either DSAA (as clause 6.3.3.1, see also ETSI EN 300 444 [i.8], clause 8.27) or DSAA2 (as clause 6.3.3.3, see also ETSI EN 300 444 [i.8], clause 8.45) algorithms. DSAA2 procedure shall only be used if DSAA2 is supported by both peers.

The encryption algorithm may be either DSC (see annex J) or DSC2 (see annex M).

After receiving the {AUTHENTICATION-REPLY} message, the FP shall perform the Cipher switching initiated by FT. This procedure shall be performed as described in ETSI EN 300 175-5 [5], clause 13.8. See also clause 6.5.3 in the present document and ETSI EN 300 444 [i.8], clause 8.33.

The PP will respond with the initiation of the MAC procedure "re-keying to a DCK" as described in clause 6.4.6.5.1 of the present document. This procedure contains a final message "START.grant" that allows the FT to know the completion of the procedure in all cases.

NOTE: This is true even in cases when there are WRSs. See ETSI EN 300 700 [12], clause 7.7.

In case of no completion of the re-keying procedure before the expiration of the timer, the procedure has failed. The application profile may specify the action to be performed in this case. These actions may range from additional security measures to simply dropping the call.

### 6.7.2.3 Re-keying procedure with Wireless Relay Stations (WRSs)

#### 6.7.2.3.1 General

In cases of systems with repeaters (Wireless Relay Station, see ETSI EN 300 700 [12]), several additional rules shall be followed:

- Re-keying should be applied to all communication segments in the system (i.e. segments from FP to WRSs, from WRSs to PP and potentially from WRSs to other WRSs).
- Transfer of keys to WRSs shall be only done over encrypted "re-keyed" links (additional requirement to the general rule of "encrypted links") and the key for such links should be "fresh" according to the aging model.
- The key aging model described in clause 6.7.2.3.2 shall be used to control the age of the key and the evaluation of the re-keying timer.

The FP shall perform the re-keying operations for all nodes and segments, in the proper sequence and initiated early enough, to guarantee that no key in the connection path has an age higher than the re-keying timer.

NOTE 1: Not following these rules may result in a reduction of security (due to the exposure of a key over a vulnerable link).

NOTE 2: The re-keying procedure, even in systems with WRSs, is under control of the FP. Note that the FP has all elements to control the evolution of the procedure.

NOTE 3: The message "START.grant" of the re-keying MAC procedure (see clause 6.4.6.5.2 "Re-keying to a DefCK") is always relayed by the repeaters towards the FP (see ETSI EN 300 700 [12], clause 7.7.7). Therefore, the FP may be aware of the completion of any re-keying in any segment.

### 6.7.2.3.2 Key aging model

The underlying aim of re-keying is to minimize the exposure and use of a cipher key, in order to mitigate the threat of brute-force attacks for recovering the cipher key. In order to control the expiration of the re-keying timer in systems with repeaters, the following key aging model shall be used:

- 1) Any DCK in the system can be considered to have an "age" which starts aging as soon as the key is first exposed or used, and will continue to age thereafter (regardless of whether the key continues to be used or not).
- 2) The age of a DCK is considered to start at the moment of key generation (i.e. by means of the Authentication procedure). The exact starting point is taken to happen at the FT sending of the {AUTHENTICATION-REQUEST} message.
- 3) When any DCK is provisioned to an upper node of a segment (e.g. to a WRS) by means of the {MM-INFO-SUGGEST} message, then that DCK inherits the age value of the key used to protect that {MM-INFO-SUGGEST} message, if that value is older than the age as calculated from the key generation (see note). The key continues aging from this value.

NOTE: It should be assumed that the most usual case is that key inherits the value of the key used for transporting the {MM-INFO-SUGGEST} message.

## 6.7.3 Early encryption

### 6.7.3.1 Aim and strategy

The early encryption is a combined MAC layer/NWK layer mechanism intended to ensure the fast activation of encryption at the beginning of any call, including service calls and virtual calls. To achieve that, a special type of Cipher Key called Default Cipher Keys (DefCK) are generated and stored in advance of their intended use by means of a variation of the Authentication procedure. The encryption itself is designed to be activated using only MAC layer messages. This allows the quick enabling of the encryption at the beginning of a call, encrypting even the call CC setup messages that may contain the called party number.

### 6.7.3.2 The Default Cipher Keys (DefCK)

The Default Cipher Keys are a special type of encryption keys intended for use in the early encryption feature. They are characterized by the following:

- Cryptographically, they are identical to the DCKs and may have 64 bits or 128 bits depending on the cipher algorithm to be used.
- The cipher algorithm to be used is identical to the one used with the DCKs and may be DSC or DSC2.
- They are generated by the same authentication procedure used to generate DCK (but a different execution). A set of parameters in the authentication messages indicates that the key to be generated and stored will be a DefCK.
- Each DefCK has an associated system level parameter called "Default Cipher Key Index". This parameter is allocated by the FP at the time of generation (by the NWK layer) and will be used by the PP at the time of execution of the MAC procedure.

### 6.7.3.3 The Default Cipher Key Index

The Default Cipher Key Index is a 16 bit value with the structure shown in table 6.20.

**Table 6.20: Default Cipher key index**

| Key-Index |                | Meaning                  |
|-----------|----------------|--------------------------|
| m.s.b     | l.s.b          |                          |
| 0000      | 0000 0000 0000 | no cipher key index      |
| 0000      | 0000 0000 0001 | valid cipher key index   |
| 1111      | 1110 1111 1111 |                          |
| 1111      | 1111 0000 0000 | reserved                 |
| 1111      | 1111 1111 1110 |                          |
| 1111      | 1111 1111 1111 | invalid cipher key index |

The Default Cipher Key Index is included in the MM messages used for generating or transporting it and in the MAC messages used to activate the early encryption (see ETSI EN 300 175-3 [3], clause 7.2.5.7).

### 6.7.3.4 Generation and refresh strategy

The FP may allocate multiple DefCKs to a given PP by using different key index values. In addition to it, the FP may refresh or regenerate the value of a given DefCK, by running an authentication to generate a DefCK and using the same index value.

Due to the implementations constrains consequence of the need for storage of the keys in non-volatile memory, the present document does not impose any requirement about when the keys should be generated or refreshed and on how many keys a PP should have allocated. The application profiles may establish additional provisions about both matters.

### 6.7.3.5 Running the procedure

To perform the early encryption procedure, the FP performs some authentication of PP procedures with generation and storage of a DefCK.

The authentication procedure shall be executed using either DSAA (as clause 6.3.3.1, see also ETSI EN 300 444 [i.8], clauses 8.24 and 8.27) or DSAA2 (as clause 6.3.3.3, see also ETSI EN 300 444 [i.8], clauses 8.24 and 8.27) algorithms. DSAA2 procedure shall only be used if DSAA2 is supported by both peers.

The generation and storage of Default Cipher Key is indicated by the inclusion of the flag <DEF> in the IE <<AUTHENTICATION-TYPE>> as well as two additional octets < Default Cipher Key Index> at the end of this same Information element (see ETSI EN 300 175-5 [5]).

Once that a PP has been provisioned with one (or more) DefCKs, the PP will be ready to run the MAC procedure for encryption with Default Cipher Keys (early encryption) when required. This procedure will be performed as described in clause 6.4.6.3.4 (PT procedure for switching from clear to encrypt mode with a Default Cipher Key (DefCK)).

The encryption with Default Cipher Keys is seen as a temporary encryption mechanism. Once a call encrypted with DefCK has been setup, the FP should be in charge of performing a switch to a "regular" DCK as soon as possible. This may be done by running a further authentication with generation of a DCK followed by a Cipher switching procedure.

The present document does not impose any requirement on the timer for this transition that is left to the relevant application profiles.

NOTE: The reason of this recommendation is protecting the Default Cipher key by not exposing it unnecessarily. See clause 6.7.3.6 on security considerations.

### 6.7.3.6 Security considerations

Since a DefCK is generated by the same algorithm as a DCK and use potentially the same cipher algorithm, the primary security strength of a DefCK is similar to a DCK. However, there are significant differences due to the potentially longer time between generation and use. Note that this time may be of the order of days in a real system.

If a DefCK has been generated with the algorithm DSAA, and this authentication exchange has been observed by a hacker, it would be in theory possible performing a brute force attack over the DSAA, if enough time is available. This threat disappears if the stronger algorithm DSAA2 is used (brute force attack time in the range of several millions of years).

For a DefCK generated using DSAA2 and using DSC, the only practical strategy would be attacking the DSC algorithm. This would be facilitated by the potential long time of use of a given key. However, this would also be made difficult by the reduced exposure of the key (the connection is re-keyed from a DefCK to a fresh DCK in a very short time).

An optimal strategy when not using DSC2 ciphering, would be using DSAA2 refreshing the keys regularly. It is even possible having "fresh" keys (keys never used with the cipher) ready for use for each new call. However, this strategy may be excessive in practical terms and collides with strategies or reducing emissions and extending the battery duration of the handsets.

A DefCK generated with DSAA2 and using DSC2 is considered to be not vulnerable to brute force attacks and is therefore the recommended solution for security concerned applications (in systems with repeaters this DSAA2/DSC2 combination should be used consistently in all segments between FP to PP).

---

## 7 Use of security features

### 7.1 Background

In this clause the options for key management within the network are described.

In clause 7.2.1 an overview is given of the security parameters relevant for a discussion of key management. For more details the reader is referred to clause 4.4.3.

Clause 7.2.2 identifies different options for the generation of keys, and clause 7.2.3 contains a list of possibilities for the distribution of keys to a PT.

Finally, in clause 7.2.4, an overview is given of the ways in which the keys and security parameters may be managed within the fixed systems. Particular attention is paid to the options for managing authentication keys and authentication parameters in a roaming environment.

NOTE: The following assumptions are made throughout this clause:

- 1) the only authentication algorithm used is the DSAA and the only cipher (KSG) used is the DSC;
- 2) for any particular PT the same authentication key K is used for both authentication of the PT and authentication of the FT.

These assumptions are made in order to simplify the discussion. In practice, network management will have to explicitly include features for key and algorithm identification.

### 7.2 Key management options

#### 7.2.1 Overview of security parameters relevant for key management

At the operational level in the DECT security architecture two keys are used:

- an authentication key K; and
- a cipher key CK.

These two operational keys are derived from authentication and encryption parameters which are available in the PT and in the fixed systems.



To date three authentication parameters have been identified as possible parameters from which to derive the authentication key K (see clause 4.4.3):

- 1) the UAK. This is secret authentication data contained within the user (subscriber) registration data. The length of a UAK is unspecified. It is however expected to be substantially greater than the length of an AC (see note);
- 2) the AC. This is a short value (for example 16 bits to 32 bits). An AC may be held within the PT or it may be manually entered whenever it is required;
- 3) the UPI. This is a short value (for example 16 bits to 32 bits). It is entered manually in a PT by the user each time it is required. The UPI is required for user authentication and is always used in combination with a UAK.

NOTE: There is no conceptual difference between a UAK and an AC. It is intended that an AC should be used only for those applications which require a temporary or short term authentication key. An example of such an application might be short term registration of a PT with a hotel or restaurant.

Two possible ways to establish the cipher key CK have been identified:

- 1) the DCK. This cipher key is automatically generated as part of the authentication of a PT process. If indicated in the <<AUTH TYPE>> information element (DEF-bit), the derived cipher key is used as default cipher key;
- 2) the SCK. This is a cipher key which is entered in both a PT and a fixed system and which can be used for an indefinite period.

The static cipher key is provided for those applications where data confidentiality is required, but where the authentication of a PT mechanism is not implemented. An example of such an application might be a residential DECT system. In this application the SCK would be chosen by the user and manually entered into both fixed and portable parts of his system.

Table 7.1 indicates for which of the three different DECT environments, Public Access Service (PAS), Business Cordless Telecommunications (BCT) and Residential Use (RU), the parameters described above are thought to be applicable. It should be stressed that table 7.1 is only an indication and deviations are possible.

**Table 7.1: Application areas and keys**

|  | PAS               | BCT               | RU |
|--|-------------------|-------------------|----|
| UAK  | A                 | A                 | A  |
| AC   | A<br>(see note 3) | A                 | A  |
| UPI/UAK  | A<br>(see note 2) | A<br>(see note 2) | N  |
| SCK  | N                 | A                 | A  |
| NOTE 1: A = Applicable.<br>N = Not applicable. |                   |                   |    |
| NOTE 2: User authentication is required.       |                   |                   |    |
| NOTE 3: Only for initial subscription.         |                   |                   |    |

When discussing the management of operational keys, and the way in which they are used to provide the security services, it is not necessary to consider the parameters from which they are actually derived. Therefore, in the following clauses, only K and CK will be considered.

## 7.2.2 Generation of authentication keys

Two options are identified for the generation of authentication keys:

- generation independent of other information; and
  - generation related to other information.
- 1) **Generation independent of other information**

In this case, the keys are first generated independently of any other information related to the user and then distributed and stored. Typically the keys would be generated in a truly random or pseudo random way.

## 2) Generation related to other information

In this case, the keys are generated from other data associated with the user. Such keys can be either stored subsequent to generation, or generated in real time whenever they are required.

A typical application of the second method for generating K is DECT public access service. The authentication key is stored in the handset, but is directly calculated by the fixed system from the user subscription data when access is attempted. The key is generated from the user subscription data using a secret function. The appropriate subscription data shall be transmitted by the PT when access is attempted, and the FT shall have access to a system element which can generate the authentication key from this data.

The main advantage with this approach is that the network does not have to maintain a database of authentication keys. However, local derivation of keys from subscription data may pose a higher security risk than in the case where keys are generated centrally using random or pseudo-random generators.

Neither of the two methods for generating K is directly supported by an element in the security specification. Thus the present document contains neither a specification for a pseudo-random generator, nor a discussion of the requirements for a secret algorithm (or class of algorithms) which can be used to generate authentication keys from subscription data.

## 7.2.3 Initial distribution and installation of keys

The following options for the initial distribution of keys for installation in portable systems have been identified:

### 1) Over the air distribution and automatic installation

The keys can be distributed to the user equipment over the air and automatically installed into the equipment along with other subscription data. For example, in a public access service application, keys may be centrally generated by the public access service operator and transmitted over the air to the PT as part of subscription registration.

### 2) Remote distribution and manual installation

The keys can be distributed remotely, for example, on paper or by means of a telephone call, to the user who can enter them manually into the portable equipment.

### 3) Local distribution

The keys can be locally generated and distributed to the portable and fixed systems. For example, keys used in a residential application might be chosen by the user, manually installed in the PP via key pad entry and then also entered manually into the FP, or sent over the air to the FP for automatic installation into the fixed equipment.

### 4) Installation at manufacture

Keys might be installed in the equipment at manufacture.

### 5) Use of a DAM

The keys can be stored in a hardware DAM, which can be distributed and attached to the DECT equipment. For example, in a public access service or a business cordless telecommunications application, all authentication processes may be implemented in a DAM. The system operator would personalize the DAM by entering the authentication key, user identity, etc. and then send it to the user for attachment to his portable equipment.

No comments are made on the level of security provided by these various options since this will depend on operational circumstances. In the case of the first UAK over-the-air allocation it is clear no protection against detection is given unless the process identified in clause 6.5.6 is employed. This requires the pre-distribution of an AC via one of the options, for example items 2) to 5).

When selecting a method for distribution and installation of keys, the operator should bear in mind that the effectiveness of the security features defined in the present document depends upon the keys being kept secret (i.e. not divulged to unauthorized third parties). Thus the key should not be vulnerable to third party interception during distribution, and they should be held in portable and fixed equipment in such a way that they cannot readily be read out from the equipment by unauthorized individuals or processes. Stored key material should also be protected against unauthorized modification or deletion.

## 7.2.4 Use of keys within the fixed network

### 7.2.4.0 Use of keys within the fixed network: general

In order to execute the security processes, keys (for example K, KS, CK) or security parameters derived using these keys (for example RAND\_F, XRES1 pairs) have to be available to the FT. In the following clauses options for the provision of such material are outlined. Firstly the keys and parameters necessary for the authentication services are considered.

If the actual key K is available at the point in the FT where the authentication process is performed, there is only one straightforward option for the use of K.

#### a) Use of actual authentication key K

The actual authentication key K is directly used to perform the authentication process. The key management for roaming in this option is depicted in figure 7.1.

The actual authentication key K might not always be available at the point in the FT where the security functions are performed. This can, for example, be the case with a visiting PT or if a service operator chooses to hold this (sensitive) information centrally. In this case session authentication keys may be used.

#### b) Use of session keys

- The authentication is performed using a fixed RS-KS-KS' triple according to the security mechanisms described in clauses 4.3.1, 4.3.2 and 4.3.3. If K is the authentication key corresponding to the user (or user's equipment), KS and KS' is derived from K and RS using, respectively, the A11 and A21 processes as specified in clause 4.5.2.1.
- The RS-KS and RS'-KS' pairs are sent to the point in the FT where authentication is performed. It can be used for an indefinite period. The calculation of RES1, CK and RES2 is done in the normal way (see clause 4.5.2.2). If the encryption mechanism is used, the DCK is also calculated in the normal way. The key management for roaming in this option is depicted in figure 7.2.

NOTE: In DSAA based authentication, in some use cases (i.e. key allocation) KS and KS' are generated with the same RS. This never happens when DSAA2 and authentication type 2 are used, since different RSs (RS<sub>128</sub> and RS<sub>128'</sub>) are always used for the calculation of KS and KS'.

- Common A12 and A22 processes shall be available in the PT and at the point within the FT where the security functions are performed.
- This option is supported by the security specification to facilitate the authentication of visitors by a visited network (see clause 4.3.1). The home network of these visitors has to provide the visited network with information needed to authenticate a visitor, but it does not have to transfer the authentication key K. This option can also be used for authentication of an FT (see clause 4.3.2) and mutual authentication (see clause 4.3.3).
- The above option can be applied within a single network for its own subscribers.
- The security specification does not specify mechanisms for the distribution of RS-KS-KS' triples.

There are two other options for the use of the authentication parameters in the case where the actual authentication key is not available at the point in the FT where the security functions are performed.

#### c) Use of precalculated sets

- The FT can use precalculated sets of authentication information to perform authentication of a PT according to the security mechanism described in clause 4.3.1. For authentication of a PT (see clause 4.3.1) the set RS, RAND\_F, XRES1 and optionally DCK is transferred to the point in the FT where authentication is performed. The key management for roaming in this option is depicted in figure 7.3.
- This option is only secure if each set is used only once by the FT. Although useful primarily for the authentication of visitors by a visited network, the technique can also be applied within a single network for its own subscribers.

- A disadvantage compared with option (5b) is that the sets have to be updated regularly by the home network. An advantage is that the PT and the visited network do not need to have a common A12 process.
- The security specification does not specify mechanisms for the distribution of precalculated sets of authentication information.
- Use of precalculated keys is not possible when authentication type 2 (DSAA2) is used. The reason is the use of a RAND\_P parameter in process A12. RAND\_P is not available until the authentication procedure is performed.

**d) Check with home database**

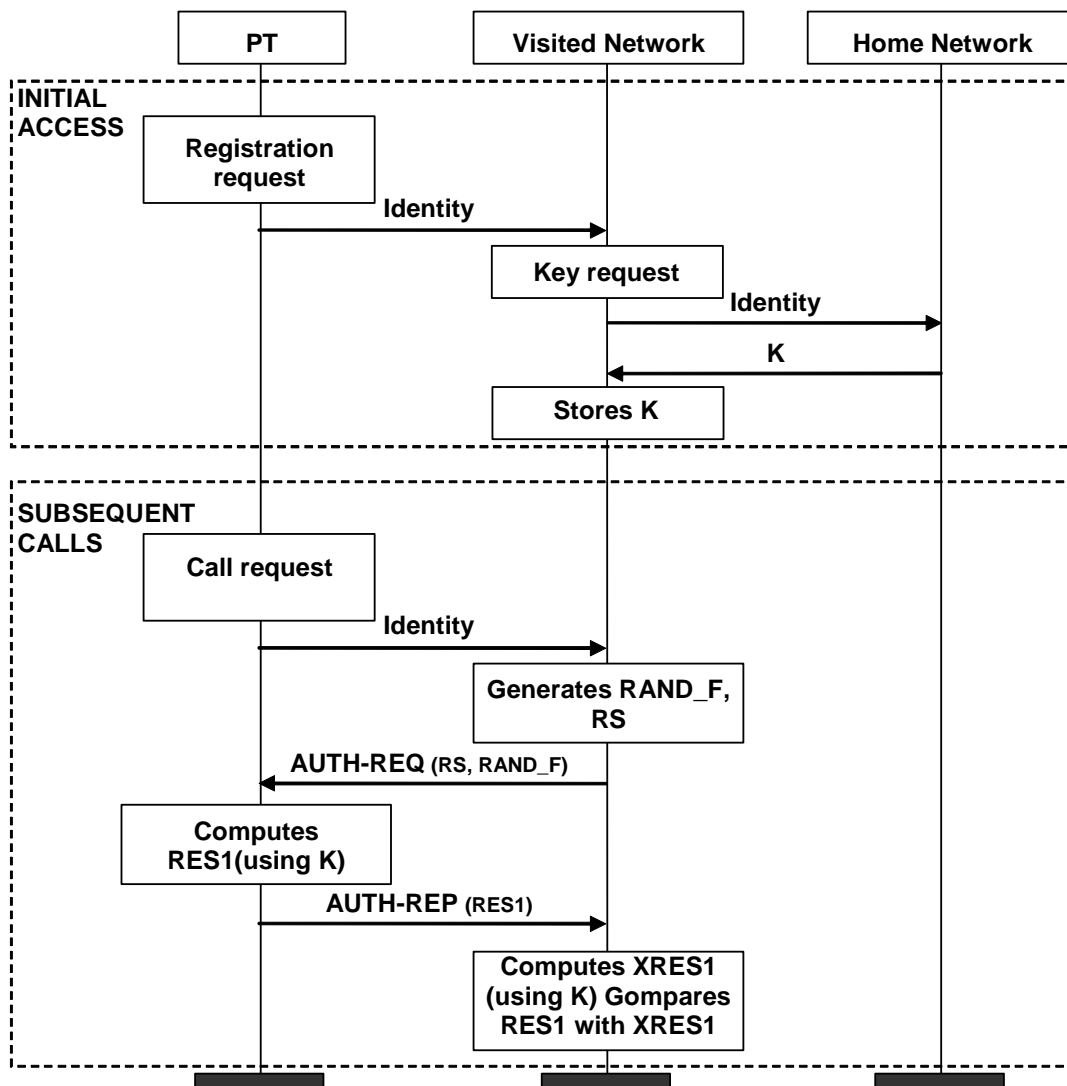
- The authentication of a PT (see clause 4.3.1) can also be realized by sending the RES1 obtained from the PT to a home database for checking.
- The security specification does not specify mechanisms for the transfer of RES1.
- In authentication type 2 (DSAA2), the RAND\_P parameter would have to be sent to the home database too.

There is only one practical option for the use of the cipher key CK.

**e) Use of the actual cipher key CK**

- The SCK or the DCK shall be available at the point in the FT where the encryption is performed.
- A DCK is automatically changed after each instance of authentication of the PT. An SCK is not changed automatically. If the SCK is used for applications needing a high level of security, then it should be changed regularly.
- The security specification does not contain features to support the management of SCKs.

### 7.2.4.1 Use of keys within the fixed network: diagrams for authentication type 1 scenarios

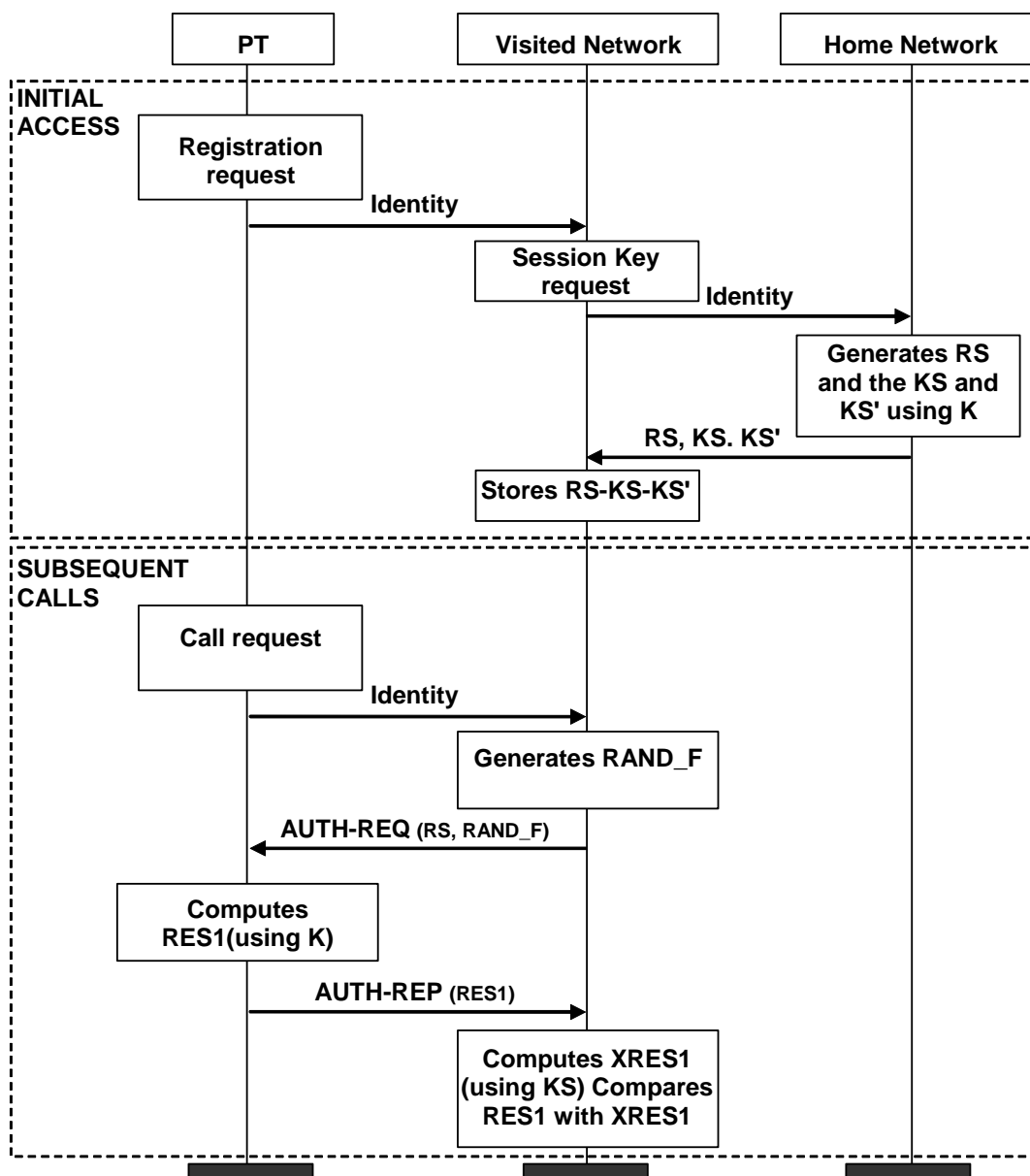


NOTE 1: Only authentication of the PT is depicted. The cases of authentication of the FT and mutual authentication are essentially the same. Generation of the DCK by the visited network is straightforward and is also not shown in the figure.

NOTE 2: AUTH-REQ is an abbreviation for {AUTHENTICATION-REQUEST} defined in clause 6.3.2. Similarly, AUTH-REP is an abbreviation for {AUTHENTICATION-REPLY}.

NOTE 3: Registration and Call procedures are shown only partially. Registration can be e.g. a DECT Location registration procedure.

**Figure 7.1: Roaming using authentication key K (authentication type 1)**

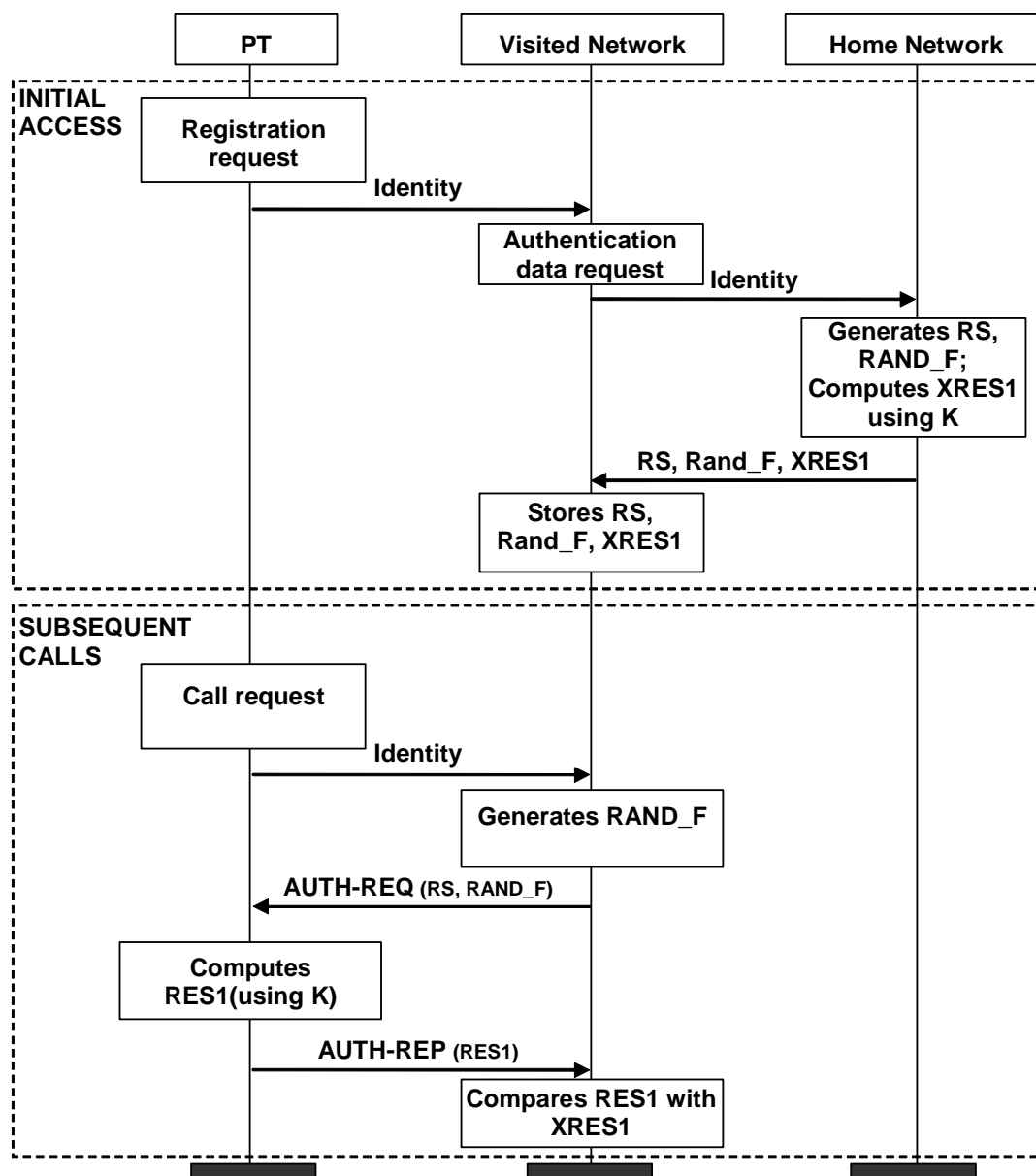


NOTE 1: Only authentication of the PT is depicted. The cases of authentication of the FT and mutual authentication are essentially the same. Generation of the DCK by the visited network is straightforward and is also not shown in the figure.

NOTE 2: AUTH-REQ is an abbreviation for {AUTHENTICATION-REQUEST} defined in clause 6.3.2. Similarly, AUTH-REP is an abbreviation for {AUTHENTICATION-REPLY}.

NOTE 3: Registration and Call procedures are shown only partially. Registration can be e.g. a DECT Location registration procedure.

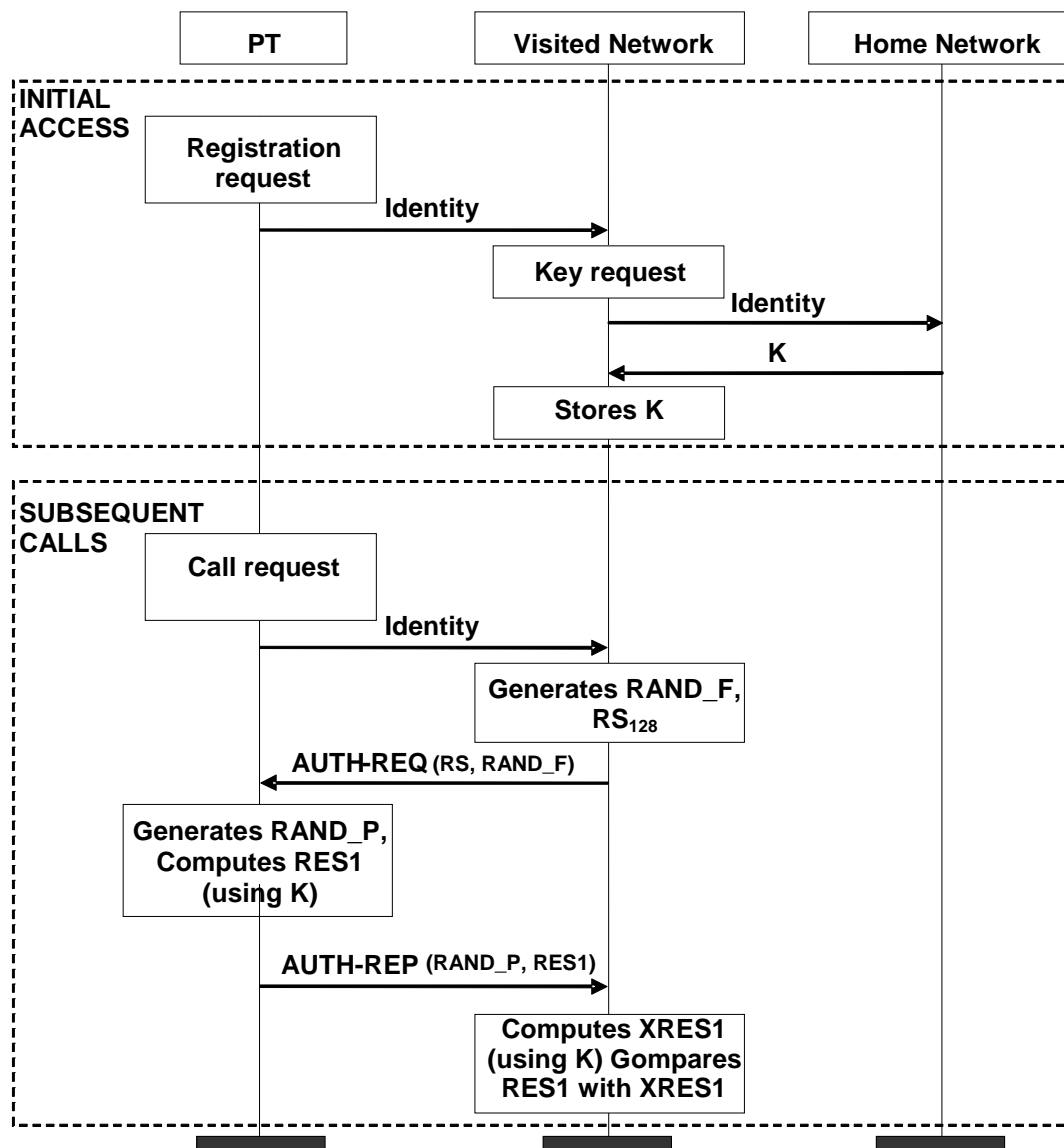
**Figure 7.2: Roaming using session keys (authentication type 1)**



- NOTE 1: Only authentication of the PT is depicted. The cases of authentication of the FT and mutual authentication are essentially the same.
- NOTE 2: If encryption is offered the home network also computes the DCK and transmits this to the visited network.
- NOTE 3: The home network can generate and transmit more than one set of authentication data at a time. In this case the visited network uses the sets one at a time, each set being used exactly once.
- NOTE 4: Registration and Call procedures are shown only partially. Registration can be e.g. a DECT Location registration procedure.
- NOTE 5: Use of precalculated keys is not possible when authentication type 2 (DSAA2) is used. The reason is the use of a RAND\_P parameter in process A12. RAND\_P is not available until the authentication procedure is performed.

**Figure 7.3: Roaming using precalculated sets (authentication type 1)**

### 7.2.4.2 Use of keys within the fixed network: diagrams for authentication type 2 scenarios



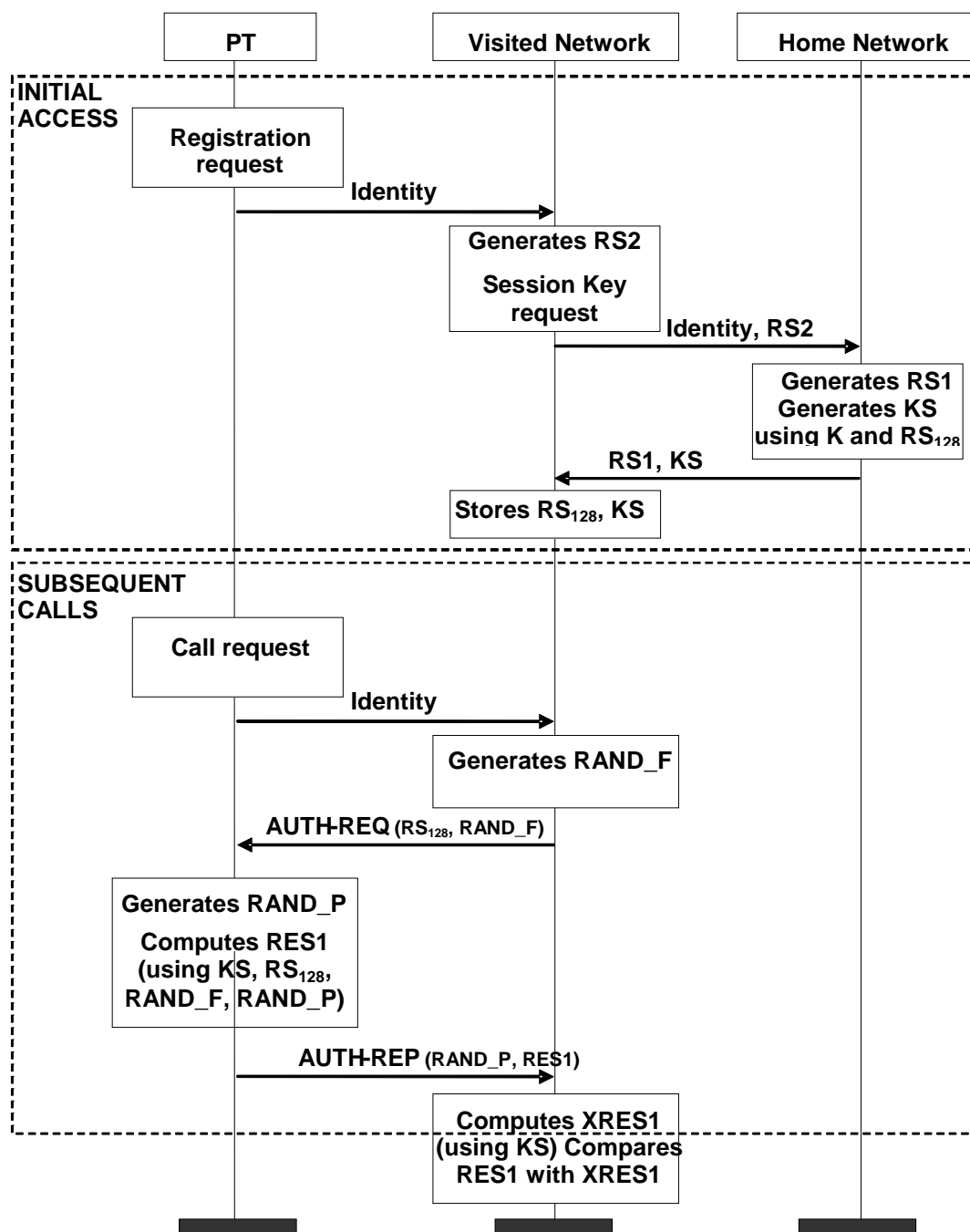
NOTE 1: Only authentication of the PT is depicted. The cases of authentication of the FT and mutual authentication are essentially the same. Generation of the DCK by the visited network is straightforward and is also not shown in the figure.

NOTE 2: AUTH-REQ is an abbreviation for {AUTHENTICATION-REQUEST} defined in clause 6.3.2. Similarly, AUTH-REP is an abbreviation for {AUTHENTICATION-REPLY}.

NOTE 3: Registration and Call procedures are shown only partially. Registration can be e.g. a DECT Location registration procedure.

**Figure 7.4: Roaming using authentication key K (authentication type 2)**





- NOTE 1: Only authentication of the PT is depicted. The cases of authentication of the FT and mutual authentication are essentially the same. However independent values of all parameters (including  $RS_{128}$  are used).
- NOTE 2: In type 2 procedure the value of the  $RS_{128}$  used for generation of KS' (for FT authentication) is always different and should be stored together with KS' (named  $RS_{128}'$ ).
- NOTE 3: This diagram shows the advanced case when  $RS_{128}$  is built from RS1/RS2, and these two parts are generated in different places (RS1 generated in the home network, RS2 generated in the visited network). Other simpler options are possible. See discussion on clause 4.4.2.
- NOTE 4: Generation of the DCK by the visited network is straightforward and is also not shown in the figure.
- NOTE 5: AUTH-REQ is an abbreviation for {AUTHENTICATION-REQUEST} defined in clause 6.3.2. Similarly, AUTH-REP is an abbreviation for {AUTHENTICATION-REPLY}.
- NOTE 6: Registration and Call procedures are shown only partially. Registration can be e.g. a DECT Location registration procedure.

**Figure 7.5: Roaming using session keys (authentication type 2)**

## 7.3 Confidentiality service with a Cordless Radio Fixed Part (CRFP)

### 7.3.1 General

The FT shall initiate the procedure for cipher key transfer of clause 7.3.2 when the CRFP requires a cipher key:

- when FT sends a {CIPHER-REQUEST} to a PT that is relayed via a CRFP;
- during bearer or connection handover when the FT receives a relayed handover request.

These procedures with the PT shall be temporarily stopped until the cipher key is transferred to the CRFP or until timeout of the connection.

### 7.3.2 CRFP initialization of PT cipher key

The FT shall have an active connection with the specific CRFP. If needed the FT shall suspend the connection with the PT and resume the connection with the CRFP as defined in clause 10.9.3 of ETSI EN 300 175-3 [3].

The FT shall initiate ciphering between the FT and CRFP that requires a PT CK using {CIPHER-REQUEST} message and the CRFP CK.

When the connection between the FT and the destination CRFP is completely ciphered, the FT shall send the PT CK to the CRFP using the {MM-INFO-SUGGEST} message.

---

# Annex A (informative): Security threats analysis

## A.1 Introduction

The purpose of the present annex is to provide an overview of possible security threats for the DECT system. For each of these threats it should be considered very carefully whether it is a realistic threat for the DECT system and if a protection mechanism against the threat has to be provided.

For this consideration, it is important to realize that there are two aspects to a threat:

- its likelihood of occurrence; and
- the possible impact on the system.

To provide a guideline, an average assessment has been produced, based on the individual scores of the threats, bearing in mind the likelihood and the security impact on the system. Marketing and commercial aspects (for example how should the security of DECT be compared to the security of GSM, or what do potential customers expect) were not considered.

To rate the threats, the following three levels were used:

- Level 1: weak or non-existing threat, not necessary to provide a countermeasure.
- Level 2: medium threat, for which it is desirable to provide a countermeasure.
- Level 3: strong threat, for which it is necessary to provide a countermeasure.

In the present document, the average score from the DECT security group is given. These scores have associated comments from the DECT security group, which are given in *italics* where appropriate.

The threats are divided into five categories:

- Threat A: impersonating a subscriber identity.
- Threat B: illegal use of a handset (PP).
- Threat C: illegal use of a base station (FP).
- Threat D: impersonation of a base station (FP).
- Threat E: illegally obtaining user and user related signalling information.

In general, these threats are considered under the assumption that no special security measures are taken. However, for threat E an attempt has been made to distinguish between the desired level of protection if a protection mechanism (encryption) is implemented.

DECT will be used in three different environments:

- Public Access Service (PAS).
- Business Cordless Telecommunication (BCT).
- Residential Use (RU).

In the present document a score is, in general, given for each of the threats in each separate environment, although in the end a suitable common denominator may have to be found.

## A.2 Threat A - Impersonating a subscriber identity

In this case, the identity of another (existing or non-existing) DECT subscriber is used to make a call. In most cases this is done to avoid call charges but it can also be done for reasons of anonymity and untraceability.

**Table A.1**

| Threat   | Environment |     |    |
|--|-------------|-----|----|
|  | PAS         | BCT | RU |
| Impersonate subscriber identity to avoid call charging (see note 1)  | 3           | 3   | 3  |
| Impersonate subscriber identity for anonymity or untraceability (see note 2)   | 2           | 1   | 1  |
| NOTE 1: If a successful attack is launched it will highly discredit the DECT system.   |             |     |    |
| NOTE 2: The likelihood of this threat seems low and the threat only exists in public access service, but the impact could be very high (criminals using the system). |             |     |    |

There is also a threat that incoming calls are, accidentally or by active misuse, routed to a wrong subscriber. The accidental case should be handled by the normal data communication protocol. The misuse is only a threat if the user data is sensitive and there probably is an automatic protection if encryption of user data is provided.

NOTE: Impersonation of subscriber identity only for anonymity is more severe than "for untraceability" and should be weighted as "2" in BCT and RU environments.

## A.3 Threat B - Illegal use of a handset (PP)

It might be possible to use PPs which are not allowed to be used in a DECT network, although the costs are billed in the right way. This could be the case with type approved PPs (for example stolen) or non-type approved PPs.

**Table A.2**

| Threat   | Environment |     |    |
|--|-------------|-----|----|
|  | PAS         | BCT | RU |
| Illicit use of type approved PP (see note 1)   | 2           | 2   | 2  |
| Use of non type approved PP (see note 2)   | 3           | 2   | 3  |
| NOTE 1: Considered to be a medium threat to the reputation of the system if there is no countermeasure, but a weak threat if simple countermeasures, like PIN on the PP or an electronic serial number together with a black list are implemented. |             |     |    |
| NOTE 2: Strong to medium threat, which goes down if a type approval procedure is applied to DECT.  |             |     |    |

NOTE: The use of non-type approved equipment is not considered to be a serious threat, and no marketing need for this protection is considered necessary. It may even be unmanageable. A type approval process will be applied to DECT equipment, and this is covered in the point (see note 2) in threat B.

## A.4 Threat C - Illegal use of a base station (FP)

In this case a call is made using a certain FP without authorization by the operator of the FP. This could, for example, be done to overload a FP (denial of service), to avoid call charges (if billing is to the FP) or just to avoid the costs for a FP.

No rating is given because the level of the threat C attack depends on how DECT is operated. If countermeasures against threat A are implemented, then automatic protection against the threat C attack is provided.

## A.5 Threat D - Impersonation of a base station (FP)

It might be possible to impersonate a FP or part of a FP functionality to attract calls, which are originally meant for another FP, in order to eavesdrop the user data or perhaps even to handle these calls ("steal phone customers").

A second threat is that a FP is impersonated in order to change data (ZAP code, subscription registration data) in PPs. Obviously this is only possible if the system allows FPs to make such changes.

**Table A.3**

| Threat  | Environment |     |    |
|---|-------------|-----|----|
|   | PAS         | BCT | RU |
| Impersonate a base station (FP) (see notes 1 and 2)   | 3           | 3   | 3  |
| NOTE 1: Considered to be a strong threat if a FP can change data in handsets.   |             |     |    |
| NOTE 2: Considered to be a strong threat in BCT if an impersonating FP could bypass an existing encryption mechanism without detection. |             |     |    |

Protection against this attack can be provided either directly, i.e. by a special authentication protocol, or indirectly by mandatory encryption of data which cannot be switched off by the FP.

NOTE: Unauthorized use of the ZAP function is of low priority. Impersonation of a FP to bypass message privacy, particularly in BCT, is considered a very serious threat.

## A.6 Threat E - Illegally obtaining user data and user related signalling information

In cordless systems there is always a threat to the privacy of the user information transmitted over the air interface. The nature of the threat depends on the protection afforded. As a guideline, five threats are considered. For the first threat it is assumed that no countermeasure is implemented.

- 1) Passive eavesdropping.

For the following threats it is assumed that a countermeasure (i.e. encryption) is implemented and that this countermeasure is attacked.

- 2) Active attack by someone having limited knowledge of the system (for example knowledge of signalling procedures but not of the encryption algorithm used).
- 3) Active attack with all knowledge but limited resources (for example PCs, limited time on mainframes).
- 4) Active attack with all knowledge and "unlimited" resources (for example mainframes, special hardware).
- 5) Academic attacks showing theoretical weaknesses, without being able to practically use them, but thereby discrediting the system.

**Table A.4: Threat: Obtaining user information**

| Level  | Environment |    |
|--|-------------|----|
|  | BCT         | RU |
| Passive eavesdropping (see note 1)   | 3           | 2  |
| Active attack, limited knowledge (see note 2)  | 1           | 1  |
| Active attack, all knowledge, limited resource   | 2           | 2  |
| Active attack, all knowledge, "unlimited resources"  | 2           | 2  |
| Academic attack (see note 3)   | 2           | 2  |
| NOTE 1: Considered to be a small threat in public access service because no privacy exists in this environment and, since the DECT cells are small, there are easier ways than monitoring the radio channel to eavesdrop a call. |             |    |
| NOTE 2: Threat is small if a reasonable encryption mechanism is implemented.   |             |    |
| NOTE 3: An academic attack will undermine confidence in the system.  |             |    |

NOTE: Privacy in residential applications is a desirable marketing option. There will be a market need for privacy in a public access service environment.

Apart from the calling-number and called-number, also other signalling elements (for example request for re-routing calls) might need protection.

**Table A.5: Threat: Obtaining user information**

| Level   | Environment |     |    |
|---|-------------|-----|----|
|   | PAS         | BCT | RU |
| Obtaining calling-number (see note)                                     | 1           | 1   | 1  |
| Obtaining called-number   | 1           | 2   | 1  |
| Obtaining other signalling information                                  | 1           | 2   | 1  |
| NOTE: Not considered to be a threat because of the small cells in DECT. |             |     |    |

## A.7 Conclusions and comments

The present annex gives an overview of possible security threats for the DECT system. It provides a rating which may be used as a guide to determine how realistic the threat actually is for the DECT system, bearing in mind the different environments in which DECT will be used.

The purpose of the annex is to establish a consensus view on the security threats for DECT and the countermeasures to be taken.

During the preparation of the EN the individual members of the DECT security group rated the threats. The average scoring, representing a consensus view is reflected in the present annex. The rating is based on the likelihood and the security impact of the threat. Marketing aspects were not taken into account.

Based on the threats assessment, the following conclusions were drawn:

- it is necessary that there is a means for the network operator to authenticate the subscriber identity;
- it is desirable that there is a mechanism to check that a PP is used in a way approved by the service operator, i.e. that a PP is not stolen and is type approved;
- it is necessary that a PP can authenticate a FP if the FP can be used to change information in the PP (for example ZAP code or other subscription registration information). However, a security mechanism for this authentication may be difficult to implement and manage, and it is advised not to allow a FP to make any changes in a PP;
- encryption of user data is essential in business applications and desirable for residential use. It is however not necessary in the public access service application. Also, in the business application, it is desirable to encrypt signalling elements.

Based on these conclusions, design was started of a security system for:

- the authentication of subscriber identities and if possible a method to extend this to the authentication of FPs;
- providing means for the privacy of user data in the business environment with, if feasible within the time scales, the possibility to extend this to residential use.

NOTE 1: The top priority for the work is the design of a security system for the authentication of subscriber identities.

NOTE 2: The provision of privacy for traffic (speech and data) in a business environment is considered essential.

NOTE 3: The provision of privacy in residential applications will be a desirable marketing option.

NOTE 4: While the assessment of the threat to privacy in a public access service environment is accepted, it is considered that there will be a market need for this feature. If the "cost" implications of its use are not too serious, application of privacy in a public access service environment can be at the operator's discretion. The priority is considered as low, but this application should not be ignored.

NOTE 5: The PT authentication of an FT to prevent unauthorized use of the ZAP function is considered to be of low priority.

NOTE 6: Threat D (impersonation of a FP) is considered as a very serious threat, particularly in a business environment. Impersonation of a FP would allow an attack for eavesdropping, because a pirate FP could bypass any message privacy.

---

## Annex B (informative): Security features and operating environments

### B.1 Introduction

DECT will be operational in three different environments: Public Access Service (PAS), Business Cordless Telecommunications (BCT) and Residential Use (RU). It should be possible to use the same handset in all three environments.

The purpose of the present annex is to investigate how a PP can roam from one environment to another or to another network in the same environment. This roaming only refers to enrolment (as defined in clause B.2), since in-call roaming between different environments and different networks will not be possible.

In the present document an overview is given of possible options for enrolment and their possible applications. It should be noted that certain schemes, such as unprotected over-air distribution of secret authentication information, are not considered in the present document.

Definitions for "subscription registration", "enrolment" and "roaming" will be given first. These will be used in the rest of the present document.

The definitions of subscription registration and roaming are taken from ETSI EN 300 175-1 [1].

---

### B.2 Definitions

**Subscription registration:** the infrequent process whereby a subscriber obtains access rights to one or more FPs.

NOTE 1: Subscription registration is usually required before a user can make or receive calls.

At subscription registration all the secret parameters required for the security services are made known to the PP and to the operator of a system.

**Enrolment (temporary registration):** a subset of registration, where a PP registers with an operator of a system (FP) (whether PAS, BCT or RU), and as a result of the enrolment both entities are made aware of some secret parameters for subsequent service provision. (The secret parameters in this case do not necessarily reveal the original secret parameters established as part of subscription registration.)

**Roaming:** the movement of a PP from one FP coverage area to another FP coverage area, where the capabilities of the FPs enable the PP to make or receive calls in both areas.

NOTE 2: Roaming requires the relevant FPs and PP to be interoperable.

A PP may roam between networks for which service provision has been established by an enrolment or a subscription registration.

---

### B.3 Enrolment options

There are 9 different cross application possibilities between environments and network, which are shown in the matrix below.

**Table B.1**

| to from | PAS     | BCT     | RU      |
|---------|---------|---------|---------|
| PAS     | A, B, C | D, E, F | D, E, F |
| BCT     | A       | D, E    | D, E    |
| RU      | A       | D, E    | D, E    |



In this matrix the possible options for cross environment enrolment are denoted by means of codes in the corresponding entries. The scenarios corresponding to the codes are described below:

- a) Normal subscription registration as a new subscriber for the PAS service. New key issued by visited PAS operator (on paper or in a detachable module). Insertion of this key in handset. This registration has no connection with registration in other networks or in other environments. If this kind of registration is done for several networks with a different (or possibly the same) key for each, it is multiple registration.
- b) On line or off line, temporary subscription registration as a new subscriber for the PAS service. New (local) key or authentication information issued by home PAS operator on line to visited PAS operator. No insertion of new keys in handset.
- c) Visited PAS operator confirms on line with the home operator that the PP is allowed to make a call. No authentication or authentication information to home PAS operator for checking (this is not possible if the use of authentication is mandatory for all FPs).
- d) Authentication key entered off line (manually or by insertion of a detachable module) into PP and, if necessary, into FP/PBX.
- e) Short authentication code entered off line, for temporary use (hotel, restaurant, friends at home). Costs charged to FP.
- f) Authentication information (for example local key) transferred only once by the PAS operator to a PBX or residential FP. This assumes an electronic communications contact between the different DECT environments.

---

## Annex C (informative): Reasons for not adopting public key techniques

The purpose of the present annex is to outline the reasons why EG-1 decided to use authentication procedures based on a symmetric algorithm, rather than on an asymmetric algorithm.

The expert group believe that asymmetric algorithms could have several advantages over symmetric algorithms within the DECT environment. However, they came to the conclusion that, for the reasons given below, the authentication procedure should be based on symmetric algorithms.

The two main constraints which were taken into account when considering the options were the following:

- the entire authentication process has to be completed within one second;
- no specific hardware, such as purpose-built encryption chips or digital signal processors, can be used for reasons of cost, size, power consumption and other more general ones.

The expert group also distinguished between public key (signature) and zero-knowledge schemes. The latter require, by definition, a separate mechanism for establishing a cipher key which is to be used in certain environments. For this reason, and the fact that most zero-knowledge schemes are covered by patents or patents pending, they do not seem to be suitable for the DECT application.

The three public key mechanisms considered were:

- elliptic curves;
- discrete logarithm; and
- RSA.

Elliptic curves have been around for a good hundred years. Their usefulness for cryptographic purposes was only established a few years ago. The expert group was of the opinion that this method has not yet been sufficiently scrutinized by the cryptographic community to recommend the use of elliptic curves for authentication purposes in DECT.

Mechanisms based on the use of discrete logarithms are considered to be secure as long as the size of the field is large enough. As this implies that even larger bit strings need to be handled than with RSA, the expert group ruled out their use.

Although the use of RSA was considered to be an attractive option, it was eventually ruled out on the basis of complexity and speed requirements. The main problem is that an algorithm is required which will execute in 100 ms to 200 ms and, when implemented in the PP, occupy only about 1 kbyte of ROM. In comparison, figures quoted for RSA implementations were of the order of 1 s to 2 s on an 80 286 processor with 20 kbytes to 30 kbytes of code. It should, however, be pointed out that smart cards incorporating special RSA logic have been announced for the coming year. These were considered to be an attractive option for the detachable module implementations (DAM implementations). Unfortunately their availability and performance are not yet known.

---

## Annex D (informative): Overview of security features

### D.1 Introduction

In the past it has been found necessary to secure mobile telecommunications. For DECT a security threat analysis has been made. The threats considered were:

- impersonation of a subscriber identity;
- illegal use of a handset (PP);
- illegal use of a base station (FP);
- impersonation of a base station (FP);
- illegal acquisition of user and user related signalling information.

It was decided that it was necessary to provide some countermeasures against these threats. Annex A contains the full results of this security threat analysis.

To provide an adequate level of protection against the security threats a number of security services have been specified within DECT. These are:

- authentication of a PT;
- authentication of an FT;
- mutual authentication of a PT and an FT;
- data confidentiality;
- user authentication.

The present annex contains a summary of these security services. Also it describes the key management scenarios in case of roaming.

---

### D.2 Authentication of a PT

Some threats, such as using the identity of another DECT subscriber to avoid call charges or for reasons of anonymity, and using stolen or non-type approved handsets, may be prevented by authentication of a PT.

Authentication of a PT enables an FT to authenticate a PT making or receiving a call through it. The service is initiated by the FT and is invoked at the beginning of a call (and may be re-invoked at any time during a call provided the call is not encrypted).

Authentication of a PT is realized by means of an authentication key which is known both to the PT and the FT. The FT and the PT compute a session authentication key from the authentication key. The FT sends a random value to the PT. Both parties encrypt this value with the session authentication key. The PT returns the encryption result to the FT, showing in this way its knowledge of the authentication key to the FT. The FT compares this value with its own encryption result. If the two values are identical the FT accepts the authenticity of the PT, if not the call is cleared.

The authentication key itself will never be sent over the air. One advantage of using a session authentication key is that the authentication key itself does not need to be revealed to visited networks.

---

## D.3 Authentication of an FT

Authentication of an FT is provided to prevent impersonation of a base station. This service is necessary if the base station can be used to change information in the handset (for example by ZAP code). Moreover, when a base station is impersonated, user data may more easily be revealed and calls may be handled in an irregular way.

Authentication of an FT is a PT initiated service which enables a PT to authenticate an FT through which it is making or receiving a call. The service is invoked at the beginning of a call (and may be re-invoked at any time during a call).

Again this service is realized using the authentication key known by the PT and the FT. The mechanism is the reverse of the authentication of a PT. The PT sends a random value to the FT and the FT returns the encrypted value, showing in this way its knowledge of the session authentication key to the PT. The PT accepts the authenticity of the FT if this value is identical to its own encryption result.

---

## D.4 Mutual authentication of a PT and an FT

### D.4.0 General

Mutual authentication enables a PT and an FT, through which a call is connected, to authenticate each other. It may be provided using one of the three mechanisms described in the following clauses.

#### D.4.1 Direct method

Direct mutual authentication is provided by combining the two mechanisms described in clauses D.2 and D.3. An FT authenticates a PT making or receiving a call through it, and afterwards the PT authenticates the FT.

#### D.4.2 Indirect method 1

Mutual authentication is also provided by combining authentication of a PT with data confidentiality. The PT is authenticated using the mechanism described in clause D.2, but the FT is not authenticated directly. From the session authentication key the PT and the FT compute a cipher key (as described in clause D.5.1). This key is used for data confidentiality. If the FT system does not know the authentication key, then it is unable to derive the cipher key, so it is unable to encrypt or decrypt data sent to and from the PT. The FT shows its authenticity by encrypting all its data sent to the PT. The mechanism requires the PT to enforce the data confidentiality service and drop any unencrypted call.

#### D.4.3 Indirect method 2

Mutual authentication is provided by using the data confidentiality service with the static cipher key mechanism described in clause D.5.2. In certain applications, a PT (or a set of PTs) and an FT may share such a fixed cipher key. This key is used for confidentiality of user and control data. The FT and the PT should both know the cipher key if the data is to be successfully encrypted and decrypted. The FT and the PT show their authenticity by encrypting all their data sent to each other. This mechanism requires both the PT and the FT to enforce data confidentiality, and for both parties to drop any unencrypted call.

---

## D.5 Data confidentiality

### D.5.0 General

Data confidentiality provides for the confidentiality of the user data and certain control data over the air interface between a PT and an FT.

It does not provide any cryptographic protection for data passed through the target network.

Both the PT and the FT establish a common cipher key. With this key, a key stream for encrypting data is generated. Two mechanisms for establishing the cipher key are provided.

## D.5.1 Cipher key derivation as part of authentication

If the PT is authenticated at the beginning of the call, using the mechanism defined in clause D.2, a new cipher key (DCK) will be established for each call, and used for the duration of that call, or until a new DCK is generated by means of another authentication of PP procedure (re-keying). In case a DefCK is generated, it will be valid until a new DefCK (with the same default cipher key index) is generated by means of another authentication of PP procedure. Both the PT and the FT compute the cipher key from the session authentication key.

## D.5.2 Static cipher key

In certain applications, a PT (or a set of PTs) and an FT may share a static (fixed) cipher key. This key is used for confidentiality of user and control data. Within the DECT security features no service is included to provide for management of static keys.

---

# D.6 User authentication

The user authentication service allows an FT to authenticate a user in a manner similar to the on-line PIN verification provided by banking systems. The service is initiated by the FT. It is invoked at the beginning of a call, and may be re-invoked at any time during a call provided the call is not encrypted.

This service requires a UPI value to be held by the user. This value is entered manually into the PT by the user. It is combined with the secret user authentication key (which is already in the PT), to determine the authentication key. This key is used in the authentication processes described in clauses D.2, D.3 and D.4.

---

# D.7 Key management in case of roaming

## D.7.1 Introduction

Management of keys is an essential aspect of the DECT security. For the effective operation of the security system, sound key management is essential.

In principle, key management is a network, user and operator topic and therefore cannot be standardized within the specification of security features for DECT. However, the specification includes elements which support the key management of the authentication key K in case of roaming.

For roaming it is not necessary that the actual authentication key K is stored at that point in the FT where the security functions are performed. The three options for when the FT has to use the authentication key K are described as follows:

- 1) use of actual authentication key K;
- 2) use of session keys;
- 3) use of pre-calculated sets.

## D.7.2 Use of actual authentication key K

The actual authentication key K is, in this case, directly used to perform the authentication process. It is sent by the home network to the visited network on request. This key, K, is used by the visited network in subsequent calls to authenticate the handset. The key management for roaming in this option is depicted in figure 7.1.

### D.7.3 Use of session keys

It might be desirable not to send the authentication key  $K$  to the visited network. The DECT security architecture specifically supports the use of a session key, instead of the key  $K$ , in this case. This session key can be used for an indefinite period by the visited network to authenticate the handset, without the authentication key  $K$  being revealed to the visited network. The key management for roaming is depicted in figure 7.2.

### D.7.4 Use of precalculated sets

A third option is that the home network calculates for each call a set of authentication data consisting of an authentication request (AUTH-REQ) and an authentication response (AUTH-RES). This set is transferred to the visited network on request, which can use it to authenticate the handset. The key management for roaming is depicted in figure 7.3.

---

## Annex E (informative): Limitations of DECT security

### E.1 Introduction

The proposed security features for DECT have been carefully selected to counter the threats identified for the DECT systems. To make cost effective solutions possible it is however accepted that the security profiles finally selected for DECT may contain only some of the security features described in the present document.

Furthermore, some security limitations were identified during the specification of the present security features. These were judged as having a negligible or acceptable impact on the system security. The judgement was therefore that it would not be appropriate or necessary to counter these threats, even if it was technically possible to do so. It has to be recognized that, for technical and economic reasons, security features do not give 100 % protection; perfect security is never possible.

The present annex aims at describing some of these remaining threats. This is done for completeness and, in some cases, for motivating and recommending suitable administrative procedures.

---

### E.2 Protocol reflection attacks

In this type of attack there is a malicious entity which can masquerade as an FT and as a PT. In the authentication schemes this entity will try to place itself between the genuine PT and FT, in order to attract the call and subsequently handle it in some way, advantageous for the malicious entity. The authentication procedure, be it PT authentication, FT authentication, or both combined, will take place exactly in the normal way. The only difference is that the protocol exchange is relayed via the false entity. After the authentication procedure is finished the false entity may take over the call completely, either as a false PT or as a false FT - the choice resides with the false entity.

This kind of attack is very hard to protect against, as the false entity need not be more intelligent than having to technically follow the protocols and quickly re-transmit the protocol elements. The attack is, in that respect, logically passive and not much more complicated than a successful monitoring of the radio link. To counter the take-over of calls in this way, authentication has to be repeated several times during the call. Even then the protection is limited; only continuous encryption gives total protection.

However, the negative consequences need not be too severe. If encryption is not used, the typical threat is that calls are re-directed to wrong destinations or are terminated at the false entity which may also act as the called party. If encryption is used there might be a threat if one party in the DECT connection could disable the encryption process or prevent it from starting. It is desirable that it is not possible for the encryption process to be terminated in such a way.

---

### E.3 Static cipher key and short Initial Vector (IV)

In the DECT encryption process the frame number is used as the IV for varying the encryption process. If the cipher key is derived from the authentication process, as described in the present document, this will also guarantee a change in the cipher process. If, however, the SCK option is used, the cipher key will be the same for many calls, and the needed variation is dependent on the IV alone. In this case the size of the IV may be important for the security of the cipher system, as there is a risk that calls being encrypted with the same IV (that is the same frame numbers are used) may be analysed to yield the two underlying plaintexts.

The analysis is not trivial, but may be considered possible if the redundancy in the plaintexts is essential and known to, or guessed by, the attacker. As more and more calls are made with the same cipher key, more and more of the frames are used and may be collected and used for subsequent analysis. This will be possible when the same frame numbers are used for the second time. Furthermore, as the frame numbers tick on in a predictable way, the attacker will know the important moments for monitoring a call. He may even initiate calls at these moments to the PT under attack in order to obtain the needed material.

Clearly the size of the IV is important for thwarting this attack. Currently the size is 28 bits, which gives a frame number repetition cycle of 31 days. Attacks will not be possible if the SCK is changed once a month. Even if the key is not changed that often, attacks may be judged to be improbable, or demanding too much, in the way of resources, to constitute a real threat. That decision will have to be made by the operator/user of the DECT systems.

---

## E.4 General considerations regarding key management

In all security systems involving secret cryptographic keys the overall security will stand or fall with the security of the keys. DECT security is thus limited by the security practices and standards used for generating, distributing and storing the secret keys.

Although not a formal part of the security features for the DECT air interface, some general observations on key management will be found in clause D.5.2. Over-the-air distribution of key material is one example of a procedure where the balance between risk of exposure and convenient management has to be carefully evaluated (and monitored for attacks).

The physical protection of the secret authentication key, and possibly the SCK, in the handset is another evident limit to system security. The read-out or otherwise unnoticed copying of the key should be considered to be a serious threat. This threat can be limited by a tamper-resistant physical design. Full tamper-proof designs are generally not possible for reasons of cost.

When Integrated Chip Cards (ICCs) are used for key storage (i.e. by using a DAM), a fairly high level of physical protection is offered to the keys. The use of DAMs also considerably simplifies key distribution whilst maintaining physical security.

---

## E.5 Use of a predictable challenge in FT authentication

The use of a non-repeating, but predictable, challenge RAND\_P in FT authentication (see clause 4.4.2) provides an opportunity for FT impersonation. The weakness can however be countered by ensuring that if RAND\_P generation is predictable, then FT authentication is always immediately preceded by PT authentication as proposed in clause 4.6.



---

## Annex F (informative): Security features related to target networks

### F.1 Introduction

#### F.1.0 General

The present annex is intended to indicate where the standard security services are implemented and how they interface with the DECT and attached networks. The attached target networks described in the present annex are those identified in ETSI ETR 056 [i.4].

It should be stressed that all of the models presented are purely functional. In many cases some of the functional building blocks will be physically co-located.

Each of the target networks in ETSI ETR 056 [i.4] is considered in turn. For each of these networks an explanation is given of the significance of each of the security services and a description is given of where the security mechanism may be implemented.

For each of the target environments example security profiles are given. These profiles encompass the services to be provided and the mechanisms and algorithms to be implemented (for example mutual authentication, DSAA, etc.).

#### F.1.1 Notation and DECT reference model

All of the functional models include a Home Data Base (HDB) and a Visitors Data Base (VDB). This clause gives a summary of the functionality of these data bases.

**Home Data Base (HDB):** the HDB manages home subscribers or PPs. The HDB stores all security parameters associated with the home subscriber and functionally forms a part of the local or global network.

**Visitors Data Base (VDB):** the VDB manages visiting subscribers or PPs. The VDB stores all security parameters associated with the visitor and functionally forms a part of the local or global network.

The security parameters held by the VDB and HDB will be described specifically in the applications below.

#### F.1.2 Significance of security features and intended usage within DECT

**Authentication of a PT:** the purpose of this service is twofold:

- 1) for public access systems it is to provide a cryptographically secure method of identifying subscriptions for billing purposes;
- 2) in private systems it is to prevent illicit access to private base stations in order to avoid charge, or for reasons of anonymity (see annex A "Security threats analysis").

Throughout the present annex no distinction is made between the ways which may be used to derive the authentication key (for example from an AC or a UAK), although in certain cases examples are provided.

**Data Confidentiality:** this is to provide privacy for user information (speech or data) and signalling information ( $C_T$  field) across the air interface.

The cipher key can be generated either as part of the authentication of the PT process or it may be a static cipher key.

**Authentication of an FT:** this service was introduced to counter the following threats (see annex A):

- 1) unauthorized loading of information (such as a Zap code) into a PP which may render the PP unusable;

- 2) FP impersonation in order to bypass privacy.

**Mutual authentication:** this is achieved by combining other services (see clause 4.6). The mechanisms required for implementation are those used for the component services.

**Direct method:** this is achieved by combining the FT and PT authentication services.

**Indirect method 1:** this is provided by combining data confidentiality using the DCK with authentication of the PT.

**Indirect method 2:** this is provided by applying the confidentiality mechanism using a SCK or a previous DCK.

**User authentication:** this is to provide authentication of the user as opposed to just authentication of user subscription data held in the portable equipment. It requires the user to enter a value (for example a UPI) whenever the service is invoked.

### F.1.3 Mechanism/algorithm and process requirements

This clause indicates where, and which, algorithms/processes need to be implemented in order to provide the security services. Table F.1 shows which processes/mechanisms are required within the PT. Table F.2 shows which processes/mechanisms are required within the FT.

**Table F.1: PT mechanisms**

| Security<br>Services/Algorithms Processes   | Authentication |    |      | Confidentiality |
|---|----------------|----|------|-----------------|
|   | PT             | FT | User |                 |
| Sequence/Random Number Generator  | N              | Y  | N    | N               |
| KSG   | N              | N  | N    | Y               |
| A1  | Y              | N  | Y    | Y*              |
| A2  | N              | Y  | N    | N               |
| Y: algorithm/process required;<br>Y*: algorithm/process required if authentication of the PT is used for the cipher key derivation;<br>N: not applicable;<br>A1: represents the combined A11 and A12 processes;<br>A2: represents the combined A21 and A22 processes.   |                |    |      |                 |
| NOTE 1: Processes A1 and A2 include the implementation of an authentication algorithm; this may be proprietary or the DSAA.   |                |    |      |                 |
| NOTE 2: If the authentication key is derived from a UAK or an AC (as described in clauses 4.5.2.1 and 4.5.2.2 respectively), then process B1 is required in the PT. Similarly if the authentication key is derived from a UAK and a UPI (as described in clause 4.5.2.3) then process B2 is required in the PT. |                |    |      |                 |

**Table F.2: FT mechanisms**

| Security<br>Services/Algorithms Processes   | Authentication |    |      | Confidentiality |
|---|----------------|----|------|-----------------|
|   | PT             | FT | User |                 |
| Random Number Generator   | Y              | N  | Y    | N               |
| KSG   | N              | N  | N    | Y               |
| A1  | Y              | N  | Y    | Y*              |
| A2  | N              | Y  | N    | N               |
| Y: algorithm/process required;<br>Y*: algorithm/process required if authentication of the PT is used for the cipherkey derivation;<br>N: not applicable;<br>A1: represents the combined A11 and A12 processes;<br>A2: represents the combined A21 and A22 processes.  |                |    |      |                 |
| NOTE 1: Processes A1 and A2 include the implementation of an authentication algorithm; this may be proprietary or the DSAA.   |                |    |      |                 |
| NOTE 2: If the authentication key is derived from a UAK or an AC (as described in clauses 4.5.2.1 and 4.5.2.2 respectively), then process B1 is required in the FT or in the fixed network element where the authentication key is derived. Similarly if the authentication key is derived from a UAK and a UPI (as described in clause 4.5.2.3) then process B2 is required in the FT or in the fixed network element where the authentication key is derived. |                |    |      |                 |

## F.2 PSTN reference configurations

### F.2.1 Domestic telephone

The domestic telephone reference configuration is shown in ETSI ETR 056 [i.4], figure 3.5. For this configuration two example security profiles P1 and P2 are suggested (see below) reflecting two distinct envisaged applications.

**P1 profile:** the P1 profile provides for the simplest case where visitors or multiple handsets are not catered for. The basic security requirements for this environment (as identified in annex A) are for confidentiality and mutual authentication.

The P1 profile provides the requirements for confidentiality using the DSC algorithm and mutual authentication via indirect method 2. Stored within local storage facilities within the fixed radio termination will be the IPUI, TPUI and the SCK. On receipt of the IPUI the SCK is loaded into the cipher.

By tables F.1 and F.2, the only algorithm/process which needs to be installed is the DSC. This will be installed within the FT and the PT.

**Table F.3: Profile P1**

| Domestic telephone example Profile P1 |                                       |
|---------------------------------------|---------------------------------------|
| DECT Security Service                 | Profile                               |
| Authentication of the PT              | N                                     |
| Authentication of the FT              | N                                     |
| User Authentication                   | N                                     |
| Mutual Authentication                 | Indirect method 2                     |
| Confidentiality                       | Using a static cipher key and the DSC |
| N:                                    | not applicable.                       |

**P2 profile:** this profile allows greater flexibility and caters for visitors and multiple handsets through the introduction of authentication of the PT service.

The P2 profile provides authentication of the PT using the DSAA, data confidentiality using the DSC and a DCK, and indirect mutual authentication.

By tables F.1 and F.2, the processes/algorithms required to be implemented are as follows: A1, DSC, random number generator. These processes/algorithms are implemented in the following areas:

**Algorithm/processes requiring implementation within the FT:**

- DSC;
- random number generator;
- process A1.

**Algorithm/processes requiring implementation within the PT:**

- DSC;
- process A1.

**Security parameters associated with the home subscriber which are stored within the local storage facilities and the PT:**

- IPUI/TPUI;
- authentication key.

**Security parameters associated with a visitor which are stored within the local storage facilities:**

- IPUI/TPUI;
- authentication key.

**Table F.4: Profile P2**

| Domestic telephone example Profile P2 |  |
|---------------------------------------|--|
| DECT Security Service                 | Profile  |
| Authentication of the PT              | Using the DSAA   |
| Authentication of the FT              | N  |
| User Authentication                   | N  |
| Mutual Authentication                 | Using indirect method 1  |
| Confidentiality                       | Cipher key derived using authentication of the PT. Encryption provided using the DSC |
| N:                                    | not applicable.  |

## F.2.2 PBX

The PBX reference configurations are depicted in figures 3.6 a) and b) of ETSI ETR 056 [i.4]. Two example profiles, P1 and P2, are given reflecting the requirements different companies may impose on the system.

NOTE 1: The profiles below are illustrative.

**P1 profile:** the P1 profile is to cater for the need for a temporary registration (for example in hotels and restaurants).

NOTE 2: A confidentiality service is not a requirement within restaurants; however, it does appear to be a requirement within hotels.

The profile in this case will be the same as the domestic telephone P2 case. However, the authentication process will take place within the local network as opposed to the FT. This results in the following implementation:

**Algorithm/processes requiring implementation within the local network:**

- random number generator;
- process A1.

**Algorithm/processes requiring implementation within the FT:**

- DSC.

**Algorithm/processes requiring implementation within the PT:**

- DSC;
- process A1.

The DCK is loaded into the DSC within the FT via the use of an IWU between the local network and the FT.

**Security parameters stored within the HDB and PT:**

The HDB and portable radio termination will contain:

- IPUI/TPUI;
- authentication key.

**Security parameters stored by the VDB:**

The VDB will store:

- IPUI/TPUI;
- authentication key.

NOTE 3: The HDB and VDB are a part of the local network.

**P2 profile:** the PBX P2 profile caters for the case when a private company wishes to use the system for confidentiality as well as to implement a proprietary authentication algorithm (or the DSAA) using, for example, a DAM.

The DAM will be inserted into the PT.

**Algorithm/processes requiring implementation within the local network:**

- random number generator;
- process A1.

**Algorithm/processes requiring implementation within the FT:**

- DSC.

The DCK is loaded into the DSC within the FT via the use of an IWU between the local network and the FT.

**Algorithm/processes requiring implementation within the DAM:**

- process A1.

**Algorithms/processes requiring implementation within the PT:**

- DSC.

**Security parameters stored within the HDB and DAM:**

- IPUI/TPUI;
- authentication key.

**Security parameters stored by the VDB:**

- IPUI/TPUI;
- authentication key.

NOTE 4: The HDB and VDB are a part of the local network.

NOTE 5: The P2 profile may also be adopted without the use of a DAM.

**Table F.5: Profile P2**

| PBX example Profile P2   |   |
|--------------------------|---|
| DECT Security Service    | Profile   |
| Authentication of the PT | Using a proprietary authentication algorithm or the DSAA.                         |
| Authentication of the FT | N   |
| User Authentication      | N   |
| Mutual Authentication    | Indirect method 1   |
| Confidentiality          | Cipher key derived using authentication of the PT. Encryption provided by the DSC |
| N:                       | not applicable.   |

## F.2.3 Local loop

The local loop reference configuration is depicted in figure 3.7 of ETSI ETR 056 [i.4]. For this configuration the example profile was driven by the following requirements:

- the need for authentication of the PT for billing purposes;
- the need for a confidentiality service to provide privacy over the air interface since within this target configuration the user would have the impression of being able to hold a private conversation.

In this target reference configuration the DECT network is integrated into the PSTN as a local loop replacement. An important point to note about this configuration is that authentication can take place either centrally by implementing the authentication of the PT processes within the global network, or locally within the FT. If central authentication is used an IWU with the FT is needed to load the DCK into the DSC, and all security parameters will be stored in the HDB and VDB within the global network. If local authentication is used, then the authentication of the PT processes will be implemented with the DSAA in the FT. For local authentication, local security parameter storage facilities will be required within the FT.

Central authentication will implement and store the following in the following locations:

### Algorithm/processes requiring implementation within the global network:

- random number generator;
- process A1.

### Algorithm/processes requiring implementation within the FT:

- DSC.

### Algorithm/processes requiring implementation within the PT:

- DSC;
- process A1.

The DCK is loaded into the DSC within the FT via the use of an IWU between the global network and the FT.

### Security parameters stored within the HDB and PT:

- IPUI/TPUI;
- authentication key.

### Security parameters stored within the VDB:

- IPUI/TPUI;
- KS, RS.

It should be noted in this configuration the handset is connected to a socket just like a normal phone, with the PT being located within the socket.

**Table F.6: Local Loop Profile**

| Local loop example profile |  |
|----------------------------|--|
| DECT Security Service      | Profile  |
| Authentication of the PT   | Suggested security profile   |
| Authentication of the FT   | Using the DSAA   |
| User Authentication        | N  |
| Mutual Authentication      | Using indirect method 1  |
| Confidentiality            | With the cipher key derived from authentication of the PT; the DSC being used for encryption |
| N:                         | not applicable.  |

## F.3 ISDN reference configurations

### F.3.1 Terminal equipment

This clause contains two example profiles P1 and P2 reflecting the two distinct configurations depicted in figure 3.8 and figure 3.8b of ETSI ETR 056 [i.4]. The profiles are different because of the way the PT is represented; either as part of the handset (profile P1, figure 3.8a), or fixed and hard-wired to the terminal (profile P2, figure 3.8b).

#### P1 profile

The P1 profile caters for the case where the local network offers the supplementary services of the ISDN network to the TE1 connection on a per handset basis. In this configuration the FT will be located on the customer premises, and the PT and end system are the handset. Since the handset is portable and can be assumed to be associated to a unique user, the services will be offered on a per handset basis as opposed to a per user basis. The configuration will be exactly the same as the domestic telephone P2 profile so as to cater for visitors and multiple handsets. However, it should be noted that the local network and FT are not physically co-located.

#### P2 profile

Since the PT and end system are fixed, the supplementary services of the ISDN network will be offered on a per user basis via the user authentication service. This means that many terminals may be attached to the same PT and still be billed on an individual basis.

The way this process works is as follows:

- the user wishing to access the ISDN network is first requested to enter his IPUI/TPUI and the personal portion of his authentication key (for example UPI);
- these parameters are then transferred to the PT where the authentication key is generated (for example from the UPI and a UAK) and the authentication process proceeds as normal.

The authentication will be implemented, and security parameters stored, within the local network and PT. The authentication process will be centrally operated. Such a local network will usually be privately operated with a local, or wide area extent, and will require an IWU with the FT to allow user selection of supplementary services and for the down loading of the DCK into the DSC. The base station in this model will consist of just the FT.

To speed up the authentication process the user authentication processes could be implemented within the FT with authentication taking place locally. However, in this case local storage facilities will have to be provided and it is suggested that the security parameters stored there should be the RS and KS (session authentication key).

#### Algorithm/processes requiring implementation within the local network:

- random number generator;
- process A1.

#### Algorithm/processes requiring implementation within the FT:

- DSC.

#### Algorithm/processes requiring implementation within the PT:

- DSC;
- process A1.

The DCK is loaded into the DSC within the FT via the use of an IWU between the local network and the FT.

#### Security parameters stored within the HDB:

- IPUI/TPUI;
- authentication key material (for example UAK and UPI).

**Security parameters stored within the PT:**

- Authentication key material (for example UAK).

The IPUI/TPUI will be entered by the user.

**Security parameters stored by the VDB:**

The VDB will store:

- IPUI/TPUI;
- authentication key.

**Table F.7: Profile P2**

| ISDN terminal equipment example Profile P2 |   |
|--|---|
| DECT Security Service                      | Profile   |
| Authentication of the PT                   | N   |
| Authentication of the FT                   | N   |
| User Authentication                        | Using (for example UPI)   |
| Mutual Authentication                      | Indirect method 1   |
| Confidentiality                            | Using the cipher key derived from user authentication. Encryption provided by the DSC |
| N:   | not applicable.   |

## F.3.2 Network termination 2

This configuration is depicted in figure 3.9 of ETSI ETR 056 [i.4]. Since the PT and end system are again fixed, the example profile will be the same as the ISDN terminal equipment P2.

## F.3.3 Local loop

The local loop configuration is shown in figure 3.10 of ETSI ETR 056 [i.4]. Since the PT and end system are again fixed, the example profile will be the same as the ISDN terminal equipment P2.

NOTE: In the configuration depicted in figure 3.10 of ETSI ETR 056 [i.4] the HDB and VDB will form part of the global network.

---

## F.4 X.25 reference configuration

### F.4.1 Data Terminal Equipment (DTE)

The configuration for this application is depicted in figure 3.11 of ETSI ETR 056 [i.4]. The example profile is to cater for the situation where the DTE has been supplied to a user with the aim of supplying all, or a fixed subset of, the supplementary services offered by the global network. In this case the supplementary services will be offered by the local network and the global network will bill the local network connection. Only one user (corporate company) is envisaged in this environment, the local network being under the control of the user, so that the profile provides authentication via the knowledge of a SCK. The profile will be the same as the domestic telephone P1 case.



## F.4.2 PAD equipment

This configuration is shown in figure 3.12 of ETSI ETR 056 [i.4]. The example profile caters for the situation where the local network offers the supplementary services of the global network by acting as a gateway to the X.25 network. The global network bills the local network connection, with the local network billing the end systems on a user basis. The billing on a per user basis is to allow multiple terminals to be attached to the PT (this also provides extra security by making the user enter a UPI before accessing the network). The PAD provider could be, and will be, in most cases the X.25 provider. The profile will be the same as the ISDN terminal equipment P2 case.

NOTE: Although the local loop and FT are functionally represented as the base station, they need not be physically co-located. This physical separation is important in terms of the storage of the authentication keys since the FT may be located on the customer's premises.

---

## F.5 GSM reference configuration

### F.5.1 Base station substation

This configuration is depicted in figure 3.13 of ETSI ETR 056 [i.4]. Here the DECT network acts as an interface to the MSC of the GSM network. The service will be provided by the local network provider who bills users on a handset basis, with the MSC billing the local network for the calls made. The requirements for this configuration will be for authentication of the handset for billing purposes and for the privacy of the air interface information. To cater for these requirements the suggested profile will be the PBX P1 profile.

### F.5.2 Mobile station

This configuration is depicted in figure 3.14 of ETSI ETR 056 [i.4]. The profile in this case will be the same as the base station substation as the requirements are the same.

---

## F.6 IEEE 802 reference configuration

### F.6.1 Bridge

Figure 3.15 of ETSI ETR 056 [i.4] depicts this configuration. Since the end system and portable radio termination will be fixed, to allow the local network to control user access when different users are sharing the same terminal, or multiple terminals are attached to the same portable radio termination, the user authentication service will be used. The profile will be the same as the terminal equipment P2 profile.

### F.6.2 Gateway

The gateway configuration is depicted in figure 3.16 of ETSI ETR 056 [i.4]. The profile will be the same as the terminal equipment P2 profile.

## F.7 Public access service reference configurations

### F.7.1 Fixed public access service reference configuration

This configuration is represented by figure 3.1 of ETSI ETR 056 [i.4]. Two profiles, P1 and P2, are described, one with a confidentiality service, the other without. It should be noted that the profiles are illustrative.

**P1 profile:** the primary requirement in this environment will be subscriber registration data authentication for billing purposes, hence the need for authentication of the PT. In a public access environment there is no requirement for confidentiality; however, there is a requirement for authentication of the FT. The P1 profile just caters for these needs.

**Table F.8: Profile P1**

| <b>Fixed public access service example Profile P1</b> |                           |
|---|---------------------------|
| <b>DECT Security Service</b>                          | <b>Profile</b>            |
| Authentication of the PT                              | Using the DSAA            |
| Authentication of the FT                              | Using the DSAA            |
| User Authentication                                   | N                         |
| Mutual Authentication                                 | Direct method (automatic) |
| Confidentiality                                       | N                         |
| N:  | not applicable.           |

**P2 profile:** example profile P2 is based on the use of the confidentiality service to provide the requirement for authentication of the FT. The profile will be the same as the PBX P1 profile.

## Annex G (informative): Compatibility of DECT and GSM authentication

### G.1 Introduction

DECT and GSM apply a similar but not identical cryptographic procedure for the authentication of a handset. Special modules (like smart cards) are used in GSM (where they are called SIMs) and may be used in DECT (where they are called DAMs) to store authentication information and perform the necessary calculations for authentication.

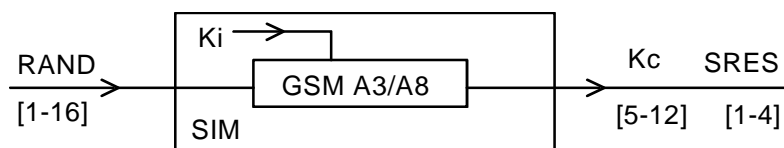
A desirable feature would be that a SIM could be used for authentication in DECT and a DAM could be used for authentication in GSM.

Clause G.1 explains how, and under which conditions, these features can be realized from a security point of view. Since GSM only offers mobile part authentication, only the corresponding PP authentication for DECT is considered.

A SIM contains all user related information and other system specific information and this is also true for a DAM. How this user and system specific information is transferred to and from the cards is not treated here. It should be clear, however, that the use of a DAM in GSM and a SIM in DECT will only be possible if they have the same functionality.

### G.2 SIM and DAM functionality

The security functionality of a SIM is depicted in figure G.1.



**Figure G.1: Security functionality of a SIM**

The SIM is activated by a specific command (RUN\_GSM\_ALGORITHM).

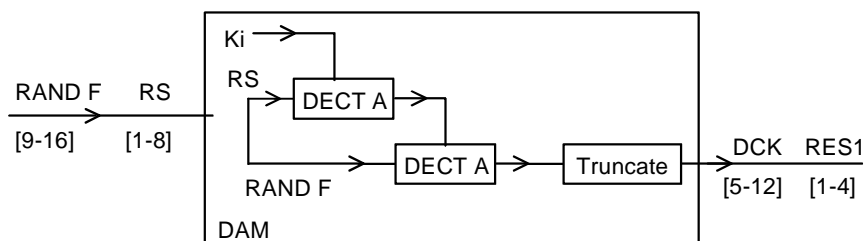
The single input of the SIM is a 16 byte value RAND which is loaded into the SIM by the handset microprocessor.

Using the GSM authentication key Ki and the GSM algorithm(s) A3/A8 two values are calculated:

- SRES (4 bytes); and
- Kc (8 bytes).

These values are serially transferred from the SIM to the handset microprocessor where the first four bytes (1 to 4) are the SRES and the following eight bytes (5 to 12) the Kc (cipher key). See ETSI TS 100 977 [9] for detailed information.

Though a DAM is in process of being specified, the needed functionality of a DAM with respect to a PP authentication is shown in figure G.2. It should be noted that a DAM could support more security functions than a SIM (for example authentication of a FP, mutual authentication). These new security functions should be activated by new commands.



**Figure G.2: Functionality of a DAM for PP authentication**

For authentication of the PT the DAM should be activated by a specific command which is identical to the corresponding SIM command (RUN\_GSM\_ALGORITHM).

Two inputs, RS and RAND\_F, are loaded to the DAM by the handset microprocessor. When the DSAA is used as algorithm A, both values are eight bytes long. In this case bytes [1 to 8] form RS and bytes [9 to 16] form RAND\_F.

Using the authentication key K and DECT algorithm A, two values, RES1 and DCK, are obtained via the intermediate value KS and a possible truncation.

These values are serially transferred from the DAM to the handset microprocessor. If the DECT standard algorithms are used, the RES1 is four bytes long and the DCK eight bytes. In this case the first four output bytes ([1 to 4]) form RES1 and the following eight bytes ([5 to 12]) DCK.

---

## G.3 Using an SIM for DECT authentication

An SIM can be used for DECT PP authentication provided that the following hold:

- 1) the FP knows that a GSM authentication (but DECT encryption and protocols) will be performed by the handset;
- 2) the "standard" lengths for the DECT parameters are used; i.e. RS, RAND\_F and DCK each have a length of eight bytes and RES1 has a length of four bytes;
- 3) SIM authentication and DAM PP authentication are activated by an identical command;
- 4) the protocol and interface between the DECT PP and the DAM are identical to the corresponding protocol and interface between the GSM mobile station and the SIM;
- 5) no session key mechanism for DECT is used.

The handset microprocessor using the command which is equivalent to (RUN\_GSM\_ALGORITHM) serially inputs RS and RAND\_F to the SIM. This serial input is interpreted by the SIM as the RAND. The SIM performs a GSM calculation and outputs the SRES and Kc which are interpreted by the handset as RES1 and DCK, respectively, and used as such in the DECT protocols.

---

## G.4 Using a DAM for GSM authentication

A DAM can be used for GSM authentication provided that the following hold:

- 1) the FP knows that a DECT authentication (but GSM encryption and protocols) will be performed by the mobile station and also knows the DECT authentication key K;
- 2) the "standard" lengths for the DECT parameters are used: i.e. RS, RAND\_F and DCK have a length of eight bytes and RES1 has a length of four bytes;
- 3) SIM authentication and DAM PP authentication are activated by an identical command;
- 4) the protocol and interface between the DECT PP and the DAM are identical to the corresponding protocol and interface between the GSM mobile station and the SIM.

The mobile station microprocessor using the command (RUN\_GSM\_ALGORITHM) serially inputs RAND (16 bytes) to the DAM. The first eight bytes of this value are interpreted by the DAM as the RS and the second eight bytes as RAND\_F. The DAM performs a DECT calculation and outputs RES1 and DCK, which are interpreted by the mobile station as SRES and Kc, respectively, and used as such in the GSM protocols.

---

## Annex H (normative): DECT Standard Authentication Algorithm (DSAA)

The DECT Standard Authentication Algorithm is only available on a restricted basis. For further information please contact ETSI.

# Annex I (normative): Security system parameters

## I.1 Security timers

### <MM\_re-keying.1>

|              |  |
|--------------|--|
| Description: | FT re-keying timer.  |
| FT value:    | 60 seconds.  |
| Start:       | A { AUTHENTICATION-REQUEST } message for re-keying is sent.                            |
| Stop:        | Next { AUTHENTICATION-REQUEST } for re-keying message is sent. The timer is restarted. |

### <MM\_re-keying.2>

|              |  |
|--------------|--|
| Description: | FT re-keying timer.  |
| FT value:    | 3 600 seconds.   |
| Start:       | A { AUTHENTICATION-REQUEST } message for re-keying is sent.                            |
| Stop:        | Next { AUTHENTICATION-REQUEST } for re-keying message is sent. The timer is restarted. |

NOTE: Timer <MM\_re-keying.2> is intended to be used only when cipher algorithm DSC2 is used in all segments between FT and PT.

### <MM\_early\_encryption.1>

|              |   |
|--------------|---|
| Description: | FT early encryption timer within which a default cipher key has to be generated.  |
| FT value:    | 30 seconds.   |
| Start:       | Encryption activation of the first call after end of obtaining access rights procedure, in case no default cipher key was generated until this point in time. |
| Stop:        | Default cipher key is generated (Authentication of PP with DEF-bit = 1).  |

### <MM\_encryption\_check.1>

|              |   |
|--------------|---|
| Description: | FT timer within which a call has to be encrypted. |
| FT value:    | 15 seconds.                                       |
| Start:       | Send/receive of { CC-SETUP }.                     |
| Stop:        | Encryption is activated.                          |

### <MM\_encryption\_check.2>

|              |  |
|--------------|--|
| Description: | PT timer within which a call has to be encrypted.          |
| PT value:    | 15 seconds.  |
| Start:       | Send/receive of { CC-SETUP }, respectively { CC-CONNECT }. |
| Stop:        | Encryption is activated.                                   |

**<MM\_registration.1>**

|              |   |
|--------------|---|
| Description: | FT timer within which the registration mode is active (Extended Fixed Part capability bit $a_{44} = 1$ ). |
| FT value:    | 120 seconds.  |
| PT value:    | Not used.   |
| Start:       | Subscription mode has been requested by the user: set $a_{44}$ "access rights supported" bit.             |
| Stop:        | As soon as on-air subscription procedure is successful, clear $a_{44}$ "access rights supported".         |

---

## Annex J (normative): DECT Standard Cipher (DSC)

The DECT Standard Cipher (DSC) is only available on a restricted basis. For further information please contact ETSI.



# Annex K (normative): Clarifications, bit mappings and examples for DSAA and DSC

## K.1 Ambiguities concerning the DSAA

The DSAA algorithm can be misinterpreted in terms of the way that the RES field should be extracted from the E output from the DSAA algorithm. This misinterpretation can cause fatal interoperability problems if two different implementations extract the RES field from opposite ends of the DSAA E output.

This problem is best resolved by means of an example.

The DSAA example consists of a key-allocation procedure conducted during the access rights procedure. The key allocation procedure uses all of the DSAA algorithm instantiations (i.e. A11, A12, A21 and A22) during the course of the transaction. This example, therefore, will ensure the highest level of assurance of DSAA correctness. In addition, the example is illustrated as a set of NWK layer messages.

In the example a DCK is also derived such that the example can also prevent any unnecessary ambiguities with the byte-ordering of the derived DCK.

### DSAA EXAMPLE:

Subscription procedure is started at PP. Authentication Code (AC) of "9124" is entered at PP keypad. After the subscription is accepted by the FP, the FP authenticates the PP to derive the DCK.

This results in the following message sequence between the PP and the FP. In all cases, the most significant byte of the NWK PDU is presented first, and the last byte of each message is the least significant byte.

First an Obtain access rights procedure is started which includes a Key allocation resulting into the derivation of a UAK, see table K.a.

**Table K.a: Subscription with Key allocation**

| Transmission direction | Message presentation  | Message coding  |
|------------------------|---|---|
| PP --->>> FP           | {ACCESS-RIGHTS-REQUEST}<br><<Portable-Identity>><br><<AUTH-TYPE>>   | 0x5 0x44<br>0x5 0x7 0x80 0xa8 0x0 0x4 0xff 0xff 0xff<br>0xa 0x3 0x1 0x48 0x0  |
| PP <<<--- FP           | {KEY-ALLOCATE}<br><<Allocation-type>><br><<RAND>><br><<RS>>   | 0x5 0x42<br>0xb 0x2 0x1 0x88<br>0xc 0x8 0xd0 0x1f 0xf9 0xe2 0x11 0x68 0x4 0x21<br>0xe 0x8 0x0 0x0 0x0 0x0 0x0 0x0 0x0 0x0     |
| PP --->>> FP           | {AUTH-REQUEST}<br><<AUTH-TYPE>><br><<RAND>><br><<RES>>  | 0x85 0x40<br>0xa 0x3 0x1 0x48 0x0<br>0xc 0x8 0xa0 0x3f 0xf2 0xc5 0x23 0xd0 0x8 0x42<br>0xd 0x4 0xf2 0x8c 0xf9 0x11            |
| PP <<<--- FP           | {AUTH-REPLY}<br><<RES>>   | 0x5 0x41<br>0xd 0x4 0xee 0xba 0x4d 0xb9   |
| PP <<<--- FP           | {ACCESS-RIGHTS-ACCEPT}<br><<Portable-Identity>><br><<Fixed-Identity>><br><<Location-area>><br><<AUTH-TYPE>> | 0x85 0x45<br>0x5 0x5 0x80 0x94 0x12 0x61 0x20<br>0x6 0x6 0xa0 0xa0 0x10 0x0 0x90 0x34<br>0x7 0x1 0x5f<br>0xa 0x3 0x1 0x18 0xf |

An authentication of the PP is now performed with the derived UAK to derive a DCK, see table K.b.

**Table K.b: DCK allocation through Authentication with UAK**

| Transmission direction | Message presentation                                  | Message coding  |
|------------------------|---|---|
| PP <<<--- FP           | {AUTH-REQUEST}<br><<AUTH-TYPE>><br><<RAND>><br><<RS>> | 0x5 0x40<br>0xa 0x3 0x1 0x18 0x18<br>0xc 0x8 0x9b 0x11 0x4f 0x25 0x54 0x51 0x88 0x59<br>0xe 0x8 0x0 0x0 0x0 0x0 0x0 0x0 0x0 0x0 |
| PP --->>> FP           | {AUTH-REPLY}<br><<RES>>                               | 0x85 0x41<br>0xd 0x4 0x96 0x1b 0x3a 0x9f  |

The following DCK value would be derived and stored by the PP:

|   |       |
|---|-------|
| MSB   | LSB   |
| 0x85 0x60 0xDC 0x32 0x4E 0xA4 0x9F 0x37                       |       |
| Bit 63  | Bit 0 |
| MSB = Most Significant Byte.<br>LSB = Least Significant Byte. |       |

---

## K.2 Ambiguities concerning the DSC DECT-standard cipher

This clause shows how to map the Derived Cipher Key (DCK) to the CK-vector of the encryption algorithm. Different implementation of the mapping leads into severe interoperability problems.

An example will use the DCK-result of clause K.1 and it is described how this is mapped to the CK of the encryption algorithm. In addition an exact description of the DCK-to-CK-mapping is given.

The example taken from clause K.1 leads into a CK which is shown in table K.1.

**Table K.1**

|     | MSB  |      |      |      |      |      |      | LSB  |
|-----|------|------|------|------|------|------|------|------|
| DCK | 0x85 | 0x60 | 0xDC | 0x32 | 0x4E | 0xA4 | 0x9F | 0x37 |
| CK  | 0x37 | 0x9F | 0xA4 | 0x4E | 0x32 | 0xDC | 0x60 | 0x85 |

This leads into the exact bit mapping given in table K.2.

Table K.2: DCK-to-CK mapping

| DCK-bit | CK-bit |
|---------|--------|
| 0       | 56     |
| 1       | 57     |
| 2       | 58     |
| 3       | 59     |
| 4       | 60     |
| 5       | 61     |
| 6       | 62     |
| 7       | 63     |
| 8       | 48     |
| 9       | 49     |
| 10      | 50     |
| 11      | 51     |
| 12      | 52     |
| 13      | 53     |
| 14      | 54     |
| 15      | 55     |
| 16      | 40     |
| 17      | 41     |
| 18      | 42     |
| 19      | 43     |
| 20      | 44     |
| 21      | 45     |
| 22      | 46     |
| 23      | 47     |

| DCK-bit | CK-bit |
|---------|--------|
| 24      | 32     |
| 25      | 33     |
| 26      | 34     |
| 27      | 35     |
| 28      | 36     |
| 29      | 37     |
| 30      | 38     |
| 31      | 39     |
| 32      | 24     |
| 33      | 25     |
| 34      | 26     |
| 35      | 27     |
| 36      | 28     |
| 37      | 29     |
| 38      | 30     |
| 39      | 31     |
| 40      | 16     |
| 41      | 17     |
| 42      | 18     |
| 43      | 19     |
| 44      | 20     |
| 45      | 21     |
| 46      | 22     |
| 47      | 23     |

| DCK-bit | CK-bit |
|---------|--------|
| 48      | 8      |
| 49      | 9      |
| 50      | 10     |
| 51      | 11     |
| 52      | 12     |
| 53      | 13     |
| 54      | 14     |
| 55      | 15     |
| 56      | 0      |
| 57      | 1      |
| 58      | 2      |
| 59      | 3      |
| 60      | 4      |
| 61      | 5      |
| 62      | 6      |
| 63      | 7      |

---

# Annex L (normative): DECT Standard Authentication Algorithm #2 (DSAA2)

## L.1 Introduction

The DECT Standard Authentication Algorithm is a message authentication code (MAC) and a key derivation function (KDF). The algorithm is loaded with key material and some non-secret input, and then used to generate either a secret session key, or a cipher key together with an authentication tag.

The DSAA2 algorithm has the following parameters:

- Inputs:
  - An  $m$ -bit secret input  $D1[0] .. D1[m-1]$ , where  $m \leq 128$
  - A first  $n$ -bit non-secret input  $D2[0] .. D2[n-1]$ , where  $n \leq 64$
  - A second  $r$ -bit non-secret input  $D3[0] .. D3[r-1]$ , where  $r \leq 64$
- Output:
  - A 128-bit output  $E$ ;
 where  $E$  can either be:
  - a 128-bit secret session key; or
  - a 32-bit tag  $E1$ , and a cipher key  $E2[0] .. E2[t-1]$ , where  $t \leq 128$ .

The secret session key derivation algorithm is named DSAA2-1 and the tag generation and cipher key derivation algorithm is named DSAA2-2.

---

## L.2 Operation of the Authentication Algorithm

### L.2.1 DSAA2-1

Let the 128-bit sequence  $K_0 .. K_{127}$  be defined as follows:

- $K_0 .. K_{m-1} = D1[0] .. D1[m-1]$
- If  $m < 128$ , then  $K_i = 0$  for  $m \leq i \leq 127$

Let the 128-bit sequence  $P_0 .. P_{127}$  be defined as follows:

- $P_0 .. P_{n-1} = D2[0] .. D2[n-1]$
- If  $n < 64$ , then  $P_i = 0$  for  $n \leq i \leq 63$
- $P_{64} .. P_{64+r-1} = D3[0] .. D3[r-1]$
- If  $r < 64$ , then  $P_{64+i} = 0$  for  $r \leq i \leq 63$

Perform an AES block cipher encryption, as specified in [10]. Use AES with the 128-bit key  $K_0 .. K_{127}$  and the 128-bit plaintext block  $P_0 .. P_{127}$ . Let the resulting ciphertext block be  $W_0 .. W_{127}$ .

Perform a second AES block cipher encryption. Use AES with the 128-bit key  $K_0 \dots K_{127}$  and the 128-bit plaintext block  $W_0 \dots W_{127}$ . Let the resulting ciphertext block be  $C1_0 \dots C1_{127}$ .

The 128 bits of this AES ciphertext block are used as secret session key E, namely  $E[0] \dots E[127] = C1_0 \dots C1_{127}$ .

## L.2.2 DSAA2-2

NOTE 1: The algorithm starts the same way as DSAA2-1.

Let the 128-bit sequence  $K_0 \dots K_{127}$  be defined as follows:

- $K_0 \dots K_{m-1} = D1[0] \dots D1[m-1]$
- If  $m < 128$ , then  $K_i = 0$  for  $m \leq i \leq 127$

Let the 128-bit sequence  $P_0 \dots P_{127}$  be defined as follows:

- $P_0 \dots P_{n-1} = D2[0] \dots D2[n-1]$
- If  $n < 64$ , then  $P_i = 0$  for  $n \leq i \leq 63$
- $P_{64} \dots P_{64+r-1} = D3[0] \dots D3[r-1]$
- If  $r < 64$ , then  $P_{64+i} = 0$  for  $r \leq i \leq 63$

Perform an AES block cipher encryption, as specified in [10]. Use AES with the 128-bit key  $K_0 \dots K_{127}$  and the 128-bit plaintext block  $P_0 \dots P_{127}$ . Let the resulting ciphertext block be  $W_0 \dots W_{127}$ .

NOTE 2: From this point on it differs from DSAA2-1.

The 32-bit tag E1 is extracted from W as follows:

- $E1[0] \dots E1[31] = W_0 \dots W_{31}$

Perform a second AES block cipher encryption. Use AES with the 128-bit key  $K_0 \dots K_{127}$  and the 128-bit plaintext block  $W_0 \dots W_{125} \ W_{126} \ (W_{127} \oplus 1)$ , i.e. just bit 127 is inverted. Let the resulting ciphertext block be  $C2_0 \dots C2_{127}$ .

The first t bits of this AES ciphertext block are used as the cipher key E2, i.e.  $E2[0] \dots E2[t-1] = C2_0 \dots C2_{t-1}$ .

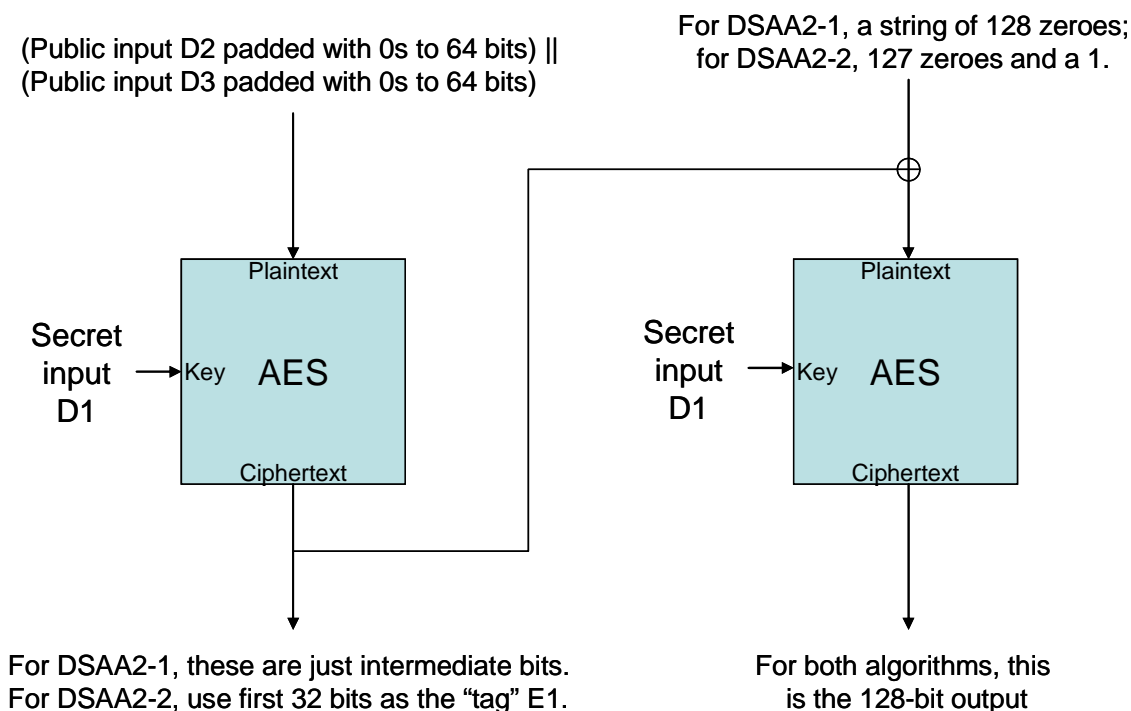


Figure L.1: The DSAA2-1 and DSAA2-2 algorithms

## L.3 Test Sets

### L.3.1 DSAA2-1

The first test set is shown twice, once in binary format, once in hexadecimal format. This is to explicitly show the relationship between the binary data and the hexadecimal representation. The remainder of the test sets are presented in hexadecimal format only. A 128-bit hexadecimal number will be broken up in four 32-bit words. For example, the number ACD4B08F62028DEB216108BA14558B55 will be written as:

ACD4B08F 62028DEB 216108BA 14558B55.

Where the length of a bit sequence is not a multiple of 4, the rightmost (least significant) bits of the final hex character are irrelevant, but are always set to zero in the test vectors. For instance, the 9-bit sequence 01011111 is written in hex as 5F8.

#### Test Set 1:

##### Binary format

$m = 128, n = 64, r = 64$

D1: 01000010 00000010 01011110 11100011 00111001 01110100 00111010 11110110  
01000111 10110101 01110111 10000000 00100101 11101001 10110110 01101101

D2: 10100000 11110011 01100010 01001110 10010100 10010110 01000000 10100000

D3: 00010111 11011010 01000111 01010001 11110101 10110010 10110001 10000000

AES Key K: 01000010 00000010 01011110 11100011 00111001 01110100 00111010 11110110  
01000111 10110101 01110111 10000000 00100101 11101001 10110110 01101101

AES Plaintext P: 10100000 11110011 01100010 01001110 10010100 10010110 01000000 10100000  
 00010111 11011010 01000111 01010001 11110101 10110010 10110001 10000000

AES Ciphertext W: 11111010 10101000 01100101 11100111 11010010 00010101 01100101 00101011  
 00000111 11000000 01000101 11001000 11101010 10001111 00110011 00100111

E: 00101001 01000000 00010110 11000000 00110100 11001101 00011010 01010101  
 01110110 00001000 00011000 00100110 00110110 01101010 01111110 01001010

**Hexadecimal format**

| Variable                                | Hexadecimal value                   |
|---|-------------------------------------|
| DSAA2-1 Secret Input D1[0]..D1[127]     | 42025EE3 39743AF6 47B57780 25E9B66D |
| DSAA2-1 Public Input D2[0]..D2[63]      | A0F3624E 949640A0                   |
| DSAA2-1 Public Input D3[0]..D3[63]      | 17DA4751 F5B2B180                   |
| AES Key $K_0 \dots K_{127}$             | 42025EE3 39743AF6 47B57780 25E9B66D |
| AES Plaintext $P_0 \dots P_{127}$       | A0F3624E 949640A0 17DA4751 F5B2B180 |
| AES Ciphertext $W_0 \dots W_{127}$      | FAA865E7 D215652B 07C045C8 EA8F3327 |
| DSAA2-1 Secret Session Key E[0]..E[127] | 294016C0 34CD1A55 76081826 366A7E4A |

**Test Set 2:**

| Variable                                | Hexadecimal value                   |
|---|-------------------------------------|
| DSAA2-1 Secret Input D1[0]..D1[127]     | 5DDB5285 D9072931 833B3556 7776898A |
| DSAA2-1 Public Input D2[0]..D2[63]      | 8493A329 2770FA99                   |
| DSAA2-1 Public Input D3[0]..D3[63]      | B950D6D2 E77E641D                   |
| AES Key $K_0 \dots K_{127}$             | 5DDB5285 D9072931 833B3556 7776898A |
| AES Plaintext $P_0 \dots P_{127}$       | 8493A329 2770FA99 B950D6D2 E77E641D |
| AES Ciphertext $W_0 \dots W_{127}$      | 2F2863B4 3F671819 7E292FCF 6BFAD22C |
| DSAA2-1 Secret Session Key E[0]..E[127] | 4C3FEA06 9EBF6CFE 419B0454 47FCFE34 |

**Test Set 3:**

| Variable                                | Hexadecimal value                   |
|---|-------------------------------------|
| DSAA2-1 Secret Input D1[0]..D1[127]     | 63340A1B D9BAF80A 25D20044 C354A708 |
| DSAA2-1 Public Input D2[0]..D2[63]      | 86167C7D 8D0EA8F4                   |
| DSAA2-1 Public Input D3[0]..D3[63]      | 74BBC76E C7C0F65E                   |
| AES Key $K_0 \dots K_{127}$             | 63340A1B D9BAF80A 25D20044 C354A708 |
| AES Plaintext $P_0 \dots P_{127}$       | 86167C7D 8D0EA8F4 74BBC76E C7C0F65E |
| AES Ciphertext $W_0 \dots W_{127}$      | 728A4748 EAB6F36F ED3B8F96 44C64623 |
| DSAA2-1 Secret Session Key E[0]..E[127] | 1461F1BF E0FAE42E 61F436A3 D32D5F4A |

**Test Set 4:**

| Variable                                | Hexadecimal value                   |
|---|-------------------------------------|
| DSAA2-1 Secret Input D1[0]..D1[119]     | BCD32DB0 21D5D009 93428051 F1CC37   |
| DSAA2-1 Public Input D2[0]..D2[59]      | 10439453 B0B8733                    |
| DSAA2-1 Public Input D3[0]..D3[55]      | AFE2C12D 7DC18F                     |
| AES Key $K_0 \dots K_{127}$             | BCD32DB0 21D5D009 93428051 F1CC3700 |
| AES Plaintext $P_0 \dots P_{127}$       | 10439453 B0B87330 AFE2C12D 7DC18F00 |
| AES Ciphertext $W_0 \dots W_{127}$      | D01DF4BD 4E2AC0E7 47E288E1 E9C805CD |
| DSAA2-1 Secret Session Key E[0]..E[127] | CE6AAEDB E96CDCE7 8650116C DD109180 |

## Test Set 5:

| Variable                                | Hexadecimal value                   |
|---|-------------------------------------|
| DSAA2-1 Secret Input D1[0]..D1[79]      | D0806449 99A0D8B7 3653              |
| DSAA2-1 Public Input D2[0]..D2[47]      | 89E394B2 77B6                       |
| DSAA2-1 Public Input D3[0]..D3[59]      | D48E6D18 F1C655E                    |
| AES Key $K_0 \dots K_{127}$             | D0806449 99A0D8B7 36530000 00000000 |
| AES Plaintext $P_0 \dots P_{127}$       | 89E394B2 77B60000 D48E6D18 F1C655E0 |
| AES Ciphertext $W_0 \dots W_{127}$      | 1A114590 82980B50 D212D04B 728B2D5E |
| DSAA2-1 Secret Session Key E[0]..E[127] | 9F05E22A DBCF67D8 29DB6B18 10B4BA50 |

## Test Set 6:

| Variable                                | Hexadecimal value                   |
|---|-------------------------------------|
| DSAA2-1 Secret Input D1[0]..D1[95]      | 070557F0 2963399B 76CD42E8          |
| DSAA2-1 Public Input D2[0]..D2[55]      | 59BE23A3 CA4FFF                     |
| DSAA2-1 Public Input D3[0]..D3[47]      | 4A877236 0C1A                       |
| AES Key $K_0 \dots K_{127}$             | 070557F0 2963399B 76CD42E8 00000000 |
| AES Plaintext $P_0 \dots P_{127}$       | 59BE23A3 CA4FFF00 4A877236 0C1A0000 |
| AES Ciphertext $W_0 \dots W_{127}$      | 02B9ED2C 77BCEC10 583FC575 5712A8BB |
| DSAA2-1 Secret Session Key E[0]..E[127] | 06B8A3FB 7C83F879 E30CDC92 023F78F0 |

## Test Set 7:

| Variable                                | Hexadecimal value                   |
|---|-------------------------------------|
| DSAA2-1 Secret Input D1[0]..D1[127]     | CA29ADAC B9651A3E BC77D383 D7A02763 |
| DSAA2-1 Public Input D2[0]..D2[63]      | E79CEA2D 91CD0FD9                   |
| DSAA2-1 Public Input D3[0]..D3[63]      | 79957A90 B4040F6E                   |
| AES Key $K_0 \dots K_{127}$             | CA29ADAC B9651A3E BC77D383 D7A02763 |
| AES Plaintext $P_0 \dots P_{127}$       | E79CEA2D 91CD0FD9 79957A90 B4040F6E |
| AES Ciphertext $W_0 \dots W_{127}$      | 843B2133 E4F5D4F2 34BC0DE8 C66249DD |
| DSAA2-1 Secret Session Key E[0]..E[127] | 95B73E80 B2263CEE A7943E32 7E8C8EF8 |

## Test Set 8:

| Variable                                | Hexadecimal value                   |
|---|-------------------------------------|
| DSAA2-1 Secret Input D1[0]..D1[110]     | 7FB30F86 31EFA426 6E1ABA5D 9E4E     |
| DSAA2-1 Public Input D2[0]..D2[58]      | 9D459058 B476DD0                    |
| DSAA2-1 Public Input D3[0]..D3[60]      | C9802A2C D2CB52D8                   |
| AES Key $K_0 \dots K_{127}$             | 7FB30F86 31EFA426 6E1ABA5D 9E4E0000 |
| AES Plaintext $P_0 \dots P_{127}$       | 9D459058 B476DD00 C9802A2C D2CB52D8 |
| AES Ciphertext $W_0 \dots W_{127}$      | FEB8FDAE F0346784 CE20BA38 E7ECD656 |
| DSAA2-1 Secret Session Key E[0]..E[127] | F2172D3A 00A04C51 11B1E506 556CBCF1 |

## Test Set 9:

| Variable                                | Hexadecimal value                    |
|---|--------------------------------------|
| DSAA2-1 Secret Input D1[0]..D1[124]     | 906C2484 7949FEDD F67E9F7E CE0036A0  |
| DSAA2-1 Public Input D2[0]..D2[60]      | E280BE2C 1C938EE8                    |
| DSAA2-1 Public Input D3[0]..D3[42]      | A20F2C14 4FA                         |
| AES Key $K_0 \dots K_{127}$             | 906C2484 7949FEDD F67E9F7E CE0036A0  |
| AES Plaintext $P_0 \dots P_{127}$       | E280BE2C 1C938EE8 A20F2C14 4FA00000  |
| AES Ciphertext $W_0 \dots W_{127}$      | AE015682 05900C3C C51B2F62 2AD88A86  |
| DSAA2-1 Secret Session Key E[0]..E[127] | DEAE8B48 65DF23EE 14B273E5 AAEEEE00F |



**Test Set 10:**

| Variable                                | Hexadecimal value                   |
|---|-------------------------------------|
| DSAA2-1 Secret Input D1[0]..D1[98]      | 631C95B1 79BB52EB BA6B28EF 4        |
| DSAA2-1 Public Input D2[0]..D2[42]      | 1E171CB2 AF6                        |
| DSAA2-1 Public Input D3[0]..D3[60]      | 6C0C2850 11136AB8                   |
| AES Key $K_0 \dots K_{127}$             | 631C95B1 79BB52EB BA6B28EF 40000000 |
| AES Plaintext $P_0 \dots P_{127}$       | 1E171CB2 AF600000 6C0C2850 11136AB8 |
| AES Ciphertext $W_0 \dots W_{127}$      | F6683A52 A8704F8A 3C8102C0 D92F56D7 |
| DSAA2-1 Secret Session Key E[0]..E[127] | 312AD124 B1F0541C 93D286FC D618451E |

**L.3.2 DSAA2-2**

The first test set is shown twice, once in binary format, once in hexadecimal format. This is to explicitly show the relationship between the binary data and the hexadecimal representation. The remainder of the test sets are presented in hexadecimal format only. A 128-bit hexadecimal number will be broken up in four 32-bit words. For example, the number ACD4B08F62028DEB216108BA14558B55 will be written as:

ACD4B08F 62028DEB 216108BA 14558B55.

Where the length of a bit sequence is not a multiple of 4, the rightmost (least significant) bits of the final hex character are irrelevant, but are always set to zero in the test vectors. For instance, the 9-bit sequence 01011111 is written in hex as 5F8.

**Test Set 1:****Binary format**

$m = 128, n = 64, r = 64, t = 128$

D1: 01000010 00000010 01011110 11100011 00111001 01110100 00111010 11110110  
01000111 10110101 01110111 10000000 00100101 11101001 10110110 01101101

D2: 10100000 11110011 01100010 01001110 10010100 10010110 01000000 10100000

D3: 00010111 11011010 01000111 01010001 11110101 10110010 10110001 10000000

AES Key K: 01000010 00000010 01011110 11100011 00111001 01110100 00111010 11110110  
01000111 10110101 01110111 10000000 00100101 11101001 10110110 01101101

AES Plaintext P: 10100000 11110011 01100010 01001110 10010100 10010110 01000000 10100000  
00010111 11011010 01000111 01010001 11110101 10110010 10110001 10000000

AES Ciphertext W: 11111010 10101000 01100101 11100111 11010010 00010101 01100101 00101011  
00000111 11000000 01000101 11001000 11101010 10001111 00110011 00100111

E1: 11111010 10101000 01100101 11100111

2<sup>nd</sup> AES Plaintext: 11111010 10101000 01100101 11100111 11010010 00010101 01100101 00101011  
00000111 11000000 01000101 11001000 11101010 10001111 00110011 00100110

C2: 01001101 01001000 01010011 00110011 00111100 01010110 01000001 10110111  
00100011 10100110 11101111 01000001 00010001 00101011 10000011 10111010

E2: 01001101 01001000 01010011 00110011 00111100 01010110 01000001 10110111  
00100011 10100110 11101111 01000001 00010001 00101011 10000011 10111010

## Hexadecimal format

| Variable                                    | Hexadecimal value                   |
|---|-------------------------------------|
| DSAA2-2 Secret Input D1[0]..D1[127]         | 42025EE3 39743AF6 47B57780 25E9B66D |
| DSAA2-2 Public Input D2[0]..D2[63]          | A0F3624E 949640A0                   |
| DSAA2-2 Public Input D3[0]..D3[63]          | 17DA4751 F5B2B180                   |
| AES Key $K_0 \dots K_{127}$                 | 42025EE3 39743AF6 47B57780 25E9B66D |
| AES Plaintext $P_0 \dots P_{127}$           | A0F3624E 949640A0 17DA4751 F5B2B180 |
| AES Ciphertext $W_0 \dots W_{127}$          | FAA865E7 D215652B 07C045C8 EA8F3327 |
| DSAA2-2 tag E[0]..E[31]                     | FAA865E7                            |
| Second AES Plaintext                        | FAA865E7 D215652B 07C045C8 EA8F3326 |
| Second AES Ciphertext $C2_0 \dots C2_{127}$ | 4D485333 3C5641B7 23A6EF41 112B83BA |
| DSAA2-2 Cipher Key E2[0]..E2[127]           | 4D485333 3C5641B7 23A6EF41 112B83BA |

## Test Set 2:

| Variable                                    | Hexadecimal value                   |
|---|-------------------------------------|
| DSAA2-2 Secret Input D1[0]..D1[127]         | 5DDB5285 D9072931 833B3556 7776898A |
| DSAA2-2 Public Input D2[0]..D2[63]          | 8493A329 2770FA99                   |
| DSAA2-2 Public Input D3[0]..D3[63]          | B950D6D2 E77E641D                   |
| AES Key $K_0 \dots K_{127}$                 | 5DDB5285 D9072931 833B3556 7776898A |
| AES Plaintext $P_0 \dots P_{127}$           | 8493A329 2770FA99 B950D6D2 E77E641D |
| AES Ciphertext $W_0 \dots W_{127}$          | 2F2863B4 3F671819 7E292FCF 6BFAD22C |
| DSAA2-2 tag E[0]..E[31]                     | 2F2863B4                            |
| Second AES Plaintext                        | 2F2863B4 3F671819 7E292FCF 6BFAD22D |
| Second AES Ciphertext $C2_0 \dots C2_{127}$ | CAEE9555 0F277C15 7E84AC35 B34A58CA |
| DSAA2-2 Cipher Key E[0]..E[79]              | CAEE9555 0F277C15 7E84              |

## Test Set 3:

| Variable                                    | Hexadecimal value                   |
|---|-------------------------------------|
| DSAA2-2 Secret Input D1[0]..D1[127]         | 63340A1B D9BAF80A 25D20044 C354A708 |
| DSAA2-2 Public Input D2[0]..D2[63]          | 86167C7D 8D0EA8F4                   |
| DSAA2-2 Public Input D3[0]..D3[63]          | 74BBC76E C7C0F65E                   |
| AES Key $K_0 \dots K_{127}$                 | 63340A1B D9BAF80A 25D20044 C354A708 |
| AES Plaintext $P_0 \dots P_{127}$           | 86167C7D 8D0EA8F4 74BBC76E C7C0F65E |
| AES Ciphertext $W_0 \dots W_{127}$          | 728A4748 EAB6F36F ED3B8F96 44C64623 |
| DSAA2-2 tag E[0]..E[31]                     | 728A4748                            |
| Second AES Plaintext                        | 728A4748 EAB6F36F ED3B8F96 44C64622 |
| Second AES Ciphertext $C2_0 \dots C2_{127}$ | 58F2181A 1A00A3F4 B1D2DE22 C9920B3D |
| DSAA2-2 Cipher Key E[0]..E[63]              | 58F2181A 1A00A3F4                   |

## Test Set 4:

| Variable                                    | Hexadecimal value                   |
|---|-------------------------------------|
| DSAA2-2 Secret Input D1[0]..D1[119]         | BCD32DB0 21D5D009 93428051 F1CC37   |
| DSAA2-2 Public Input D2[0]..D2[59]          | 10439453 B0B8733                    |
| DSAA2-2 Public Input D3[0]..D3[55]          | AFE2C12D 7DC18F                     |
| AES Key $K_0 \dots K_{127}$                 | BCD32DB0 21D5D009 93428051 F1CC3700 |
| AES Plaintext $P_0 \dots P_{127}$           | 10439453 B0B87330 AFE2C12D 7DC18F00 |
| AES Ciphertext $W_0 \dots W_{127}$          | D01DF4BD 4E2AC0E7 47E288E1 E9C805CD |
| DSAA2-2 tag E[0]..E[31]                     | D01DF4BD                            |
| Second AES Plaintext                        | D01DF4BD 4E2AC0E7 47E288E1 E9C805CC |
| Second AES Ciphertext $C2_0 \dots C2_{127}$ | A4DC5352 D1A5C424 5396F6C6 2CE53B22 |
| DSAA2-2 Cipher Key E[0]..E[119]             | A4DC5352 D1A5C424 5396F6C6 2CE53B   |

## Test Set 5:

| Variable                                    | Hexadecimal value                   |
|---|-------------------------------------|
| DSAA2-2 Secret Input D1[0]..D1[79]          | D0806449 99A0D8B7 3653              |
| DSAA2-2 Public Input D2[0]..D2[47]          | 89E394B2 77B6                       |
| DSAA2-2 Public Input D3[0]..D3[59]          | D48E6D18 F1C655E                    |
| AES Key $K_0 \dots K_{127}$                 | D0806449 99A0D8B7 36530000 00000000 |
| AES Plaintext $P_0 \dots P_{127}$           | 89E394B2 77B60000 D48E6D18 F1C655E0 |
| AES Ciphertext $W_0 \dots W_{127}$          | 1A114590 82980B50 D212D04B 728B2D5E |
| DSAA2-2 tag E[0]..E[31]                     | 1A114590                            |
| Second AES Plaintext                        | 1A114590 82980B50 D212D04B 728B2D5F |
| Second AES Ciphertext $C2_0 \dots C2_{127}$ | 98D88329 F6003CD7 C49027BA A49E29F3 |
| DSAA2-2 Cipher Key E[0]..E[79]              | 98D88329 F6003CD7 C490              |

## Test Set 6:

| Variable                                    | Hexadecimal value                   |
|---|-------------------------------------|
| DSAA2-2 Secret Input D1[0]..D1[95]          | 070557F0 2963399B 76CD42E8          |
| DSAA2-2 Public Input D2[0]..D2[55]          | 59BE23A3 CA4FFF                     |
| DSAA2-2 Public Input D3[0]..D3[47]          | 4A877236 0C1A                       |
| AES Key $K_0 \dots K_{127}$                 | 070557F0 2963399B 76CD42E8 00000000 |
| AES Plaintext $P_0 \dots P_{127}$           | 59BE23A3 CA4FFF00 4A877236 0C1A0000 |
| AES Ciphertext $W_0 \dots W_{127}$          | 02B9ED2C 77BCEC10 583FC575 5712A8BB |
| DSAA2-2 tag E[0]..E[31]                     | 02B9ED2C                            |
| Second AES Plaintext                        | 02B9ED2C 77BCEC10 583FC575 5712A8BA |
| Second AES Ciphertext $C2_0 \dots C2_{127}$ | 2ECD2991 DFD13C4C 8BA7BFD6 D8A7AAB4 |
| DSAA2-2 Cipher Key E[0]..E[95]              | 2ECD2991 DFD13C4C 8BA7BFD6          |

## Test Set 7:

| Variable                                    | Hexadecimal value                   |
|---|-------------------------------------|
| DSAA2-2 Secret Input D1[0]..D1[127]         | CA29ADAC B9651A3E BC77D383 D7A02763 |
| DSAA2-2 Public Input D2[0]..D2[63]          | E79CEA2D 91CD0FD9                   |
| DSAA2-2 Public Input D3[0]..D3[63]          | 79957A90 B4040F6E                   |
| AES Key $K_0 \dots K_{127}$                 | CA29ADAC B9651A3E BC77D383 D7A02763 |
| AES Plaintext $P_0 \dots P_{127}$           | E79CEA2D 91CD0FD9 79957A90 B4040F6E |
| AES Ciphertext $W_0 \dots W_{127}$          | 843B2133 E4F5D4F2 34BC0DE8 C66249DD |
| DSAA2-2 tag E[0]..E[31]                     | 843B2133                            |
| Second AES Plaintext                        | 843B2133 E4F5D4F2 34BC0DE8 C66249DC |
| Second AES Ciphertext $C2_0 \dots C2_{127}$ | CEFDD369 896F6A9E BA04FD14 D198052D |
| DSAA2-2 Cipher Key E[0]..E[127]             | CEFDD369 896F6A9E BA04FD14 D198052D |

## Test Set 8:

| Variable                                    | Hexadecimal value                   |
|---|-------------------------------------|
| DSAA2-2 Secret Input D1[0]..D1[110]         | 7FB30F86 31EFA426 6E1ABA5D 9E4E     |
| DSAA2-2 Public Input D2[0]..D2[58]          | 9D459058 B476DD0                    |
| DSAA2-2 Public Input D3[0]..D3[60]          | C9802A2C D2CB52D8                   |
| AES Key $K_0 \dots K_{127}$                 | 7FB30F86 31EFA426 6E1ABA5D 9E4E0000 |
| AES Plaintext $P_0 \dots P_{127}$           | 9D459058 B476DD00 C9802A2C D2CB52D8 |
| AES Ciphertext $W_0 \dots W_{127}$          | FEB8FDAE F0346784 CE20BA38 E7ECD656 |
| DSAA2-2 tag E[0]..E[31]                     | FEB8FDAE                            |
| Second AES Plaintext                        | FEB8FDAE F0346784 CE20BA38 E7ECD657 |
| Second AES Ciphertext $C2_0 \dots C2_{127}$ | 1713B004 3B2AA8B7 4722022F 85892BD9 |
| DSAA2-2 Cipher Key E[0]..E[108]             | 1713B004 3B2AA8B7 4722022F 8588     |

**Test Set 9:**

| Variable                                    | Hexadecimal value                   |
|---|-------------------------------------|
| DSAA2-2 Secret Input D1[0]..D1[124]         | 906C2484 7949FEDD F67E9F7E CE0036A0 |
| DSAA2-2 Public Input D2[0]..D2[60]          | E280BE2C 1C938EE8                   |
| DSAA2-2 Public Input D3[0]..D3[42]          | A20F2C14 4FA                        |
| AES Key $K_0 \dots K_{127}$                 | 906C2484 7949FEDD F67E9F7E CE0036A0 |
| AES Plaintext $P_0 \dots P_{127}$           | E280BE2C 1C938EE8 A20F2C14 4FA00000 |
| AES Ciphertext $W_0 \dots W_{127}$          | AE015682 05900C3C C51B2F62 2AD88A86 |
| DSAA2-2 tag E[0]..E[31]                     | AE015682                            |
| Second AES Plaintext                        | AE015682 05900C3C C51B2F62 2AD88A87 |
| Second AES Ciphertext $C2_0 \dots C2_{127}$ | CDC7608F 40F72236 37E34633 490AFF97 |
| DSAA2-2 Cipher Key E[0]..E[122]             | CDC7608F 40F72236 37E34633 490AFF8  |

**Test Set 10:**

| Variable                                    | Hexadecimal value                   |
|---|-------------------------------------|
| DSAA2-2 Secret Input D1[0]..D1[98]          | 631C95B1 79BB52EB BA6B28EF 4        |
| DSAA2-2 Public Input D2[0]..D2[42]          | 1E171CB2 AF6                        |
| DSAA2-2 Public Input D3[0]..D3[60]          | 6C0C2850 11136AB8                   |
| AES Key $K_0 \dots K_{127}$                 | 631C95B1 79BB52EB BA6B28EF 40000000 |
| AES Plaintext $P_0 \dots P_{127}$           | 1E171CB2 AF600000 6C0C2850 11136AB8 |
| AES Ciphertext $W_0 \dots W_{127}$          | F6683A52 A8704F8A 3C8102C0 D92F56D7 |
| DSAA2-2 tag E[0]..E[31]                     | F6683A52                            |
| Second AES Plaintext                        | F6683A52 A8704F8A 3C8102C0 D92F56D6 |
| Second AES Ciphertext $C2_0 \dots C2_{127}$ | 52B1A3EA 0949A87F BD91D56A 85E64B53 |
| DSAA2-2 Cipher Key E[0]..E[88]              | 52B1A3EA 0949A87F BD91D50           |

---

## L.4 DSAA2 Examples

### L.4.0 General

Subscription procedure is started at PP. Authentication Code (AC) of "9124" is entered at PP keypad. After the subscription is accepted by the FP, the FP authenticates the PP to derive the DCK.

This results in the following message sequence between the PP and the FP. In all cases, the least significant byte of the security related information elements is presented first, and the last byte is the most significant byte.

#### L.4.1 Subscription with Key Allocation

##### L.4.1.0 Message sequence and coding

First, an obtain access rights procedure is started which includes a Key allocation resulting into the derivation of a UAK, see table L.1.

Table L.1: NWK messages for subscription with key allocation

| Transmission direction | Message presentation  | Message coding   |
|------------------------|---|--|
| PP --->>> FP           | {ACCESS-RIGHTS-REQUEST}<br><<Portable-Identity>><br><<AUTH-TYPE>>     | 05 44<br>05 07 80 A8 00 04 FF FF FF<br>0A 03 02 48 00  |
| PP <<<---- FP          | {KEY-ALLOCATE}<br><<Allocation-type>><br><<RAND>><br><<RS>>           | 05 42<br>0B 02 02 88<br>0C 08 8A 9F DD 3C D9 2F 6D 1E<br>0E 10 0E E7 0C 67 12 60 74 BD E0 39 6C D0 40 65 58 90       |
| PP --->>> FP           | {AUTH-REQUEST}<br><<AUTH-TYPE>><br><<RAND>><br><<RES>>                | 85 40<br>0A 03 02 48 00<br>0C 08 09 6D 5F 46 CA 0C EF 9E<br>0D 04 BE DC 30 A6  |
| PP <<<---- FP          | {AUTH-REPLY}<br><<RAND>><br><<RES>><br><<RS>>                         | 05 41<br>0C 08 EB F0 D2 A4 27 FC F4 2F<br>0D 04 FC 5C 91 86<br>0E 10 DF 94 5F 3B 68 BB 92 B5 7C B8 F0 79 AA FC 5F 11 |
| PP <<<---- FP          | {ACCESS-RIGHTS-ACCEPT}<br><<Portable-Identity>><br><<Fixed-Identity>> | 85 45<br>05 07 80 A8 00 04 FF FF FF<br>06 07 A0 A5 01 11 11 11 10  |

Below is a summary of the inputs, intermediate results, and outputs of the A11, A12, A21, and A22 processes involved in procedure outlined in table L.1.

#### L.4.1.1 PP AC Authentication

Table L.2: Calculation for process A11 (DSAA2-1)

| Variable                                | Hexadecimal value                   |
|---|-------------------------------------|
| DSAA2-1 Secret Input D1[0]..D1[127]     | FFFF9124 FFFF9124 FFFF9124 FFFF9124 |
| DSAA2-1 Public Input D2[0]..D2[63]      | 0EE70C67 126074BD                   |
| DSAA2-1 Public Input D3[0]..D3[63]      | E0396CD0 40655890                   |
| AES Key $K_0 \dots K_{127}$             | FFFF9124 FFFF9124 FFFF9124 FFFF9124 |
| AES Plaintext $P_0 \dots P_{127}$       | 0EE70C67 126074BD E0396CD0 40655890 |
| AES Ciphertext $W_0 \dots W_{127}$      | 3B457FB9 969805D3 6CAB76AA 170DB3AD |
| DSAA2-1 Secret Session Key E[0]..E[127] | A7260187 22AA6A7D 3E79CDAA 1F613E26 |

Table L.3: Calculation for process A12 (DSAA2-2)

| Variable                                    | Hexadecimal value                   |
|---|-------------------------------------|
| DSAA2-2 Secret Input D1[0]..D1[127]         | A7260187 22AA6A7D 3E79CDAA 1F613E26 |
| DSAA2-2 Public Input D2[0]..D2[63]          | 8A9FDD3C D92F6D1E                   |
| DSAA2-2 Public Input D3[0]..D3[63]          | 096D5F46 CA0CEF9E                   |
| AES Key $K_0 \dots K_{127}$                 | A7260187 22AA6A7D 3E79CDAA 1F613E26 |
| AES Plaintext $P_0 \dots P_{127}$           | 8A9FDD3C D92F6D1E 096D5F46 CA0CEF9E |
| AES Ciphertext $W_0 \dots W_{127}$          | BEDC30A6 1EC2116F D07F3E92 B1ED4778 |
| DSAA2-2 tag E[0]..E[31]                     | BEDC30A6                            |
| Second AES Plaintext                        | BEDC30A6 1EC2116F D07F3E92 B1ED4778 |
| Second AES Ciphertext $C2_0 \dots C2_{127}$ | F8BFB805 18E9E5DA 380179D7 DB92AE28 |
| DSAA2-2 Cipher Key E2[0]..E2[127]           | F8BFB805 18E9E5DA 380179D7 DB92AE28 |

## L.4.1.2 FP AC Authentication

**Table L.4: Calculation for process A21 (DSAA2-1)**

| Variable                                | Hexadecimal value                   |
|---|-------------------------------------|
| DSAA2-1 Secret Input D1[0]..D1[127]     | FFFF9124 FFFF9124 FFFF9124 FFFF9124 |
| DSAA2-1 Public Input D2[0]..D2[63]      | DF945F3B 68BB92B5                   |
| DSAA2-1 Public Input D3[0]..D3[63]      | 7CB8F079 AAFC5F11                   |
| AES Key $K_0 \dots K_{127}$             | FFFF9124 FFFF9124 FFFF9124 FFFF9124 |
| AES Plaintext $P_0 \dots P_{127}$       | DF945F3B 68BB92B5 7CB8F079 AAFC5F11 |
| AES Ciphertext $W_0 \dots W_{127}$      | 851FF169 71303CEC 9CAB73D8 736677B9 |
| DSAA2-1 Secret Session Key E[0]..E[127] | CD257682 F4416053 7CD50DBF 1BDB145D |

**Table L.5: Calculation for process A22 (DSAA2-2)**

| Variable                                    | Hexadecimal value                   |
|---|-------------------------------------|
| DSAA2-2 Secret Input D1[0]..D1[127]         | CD257682 F4416053 7CD50DBF 1BDB145D |
| DSAA2-2 Public Input D2[0]..D2[63]          | 096D5F46 CA0CEF9E                   |
| DSAA2-2 Public Input D3[0]..D3[63]          | EBF0D2A4 27FCF42F                   |
| AES Key $K_0 \dots K_{127}$                 | CD257682 F4416053 7CD50DBF 1BDB145D |
| AES Plaintext $P_0 \dots P_{127}$           | 096D5F46 CA0CEF9E EBF0D2A4 27FCF42F |
| AES Ciphertext $W_0 \dots W_{127}$          | FC5C9186 D2147C06 0EBA5B66 7A88B826 |
| DSAA2-2 tag E[0]..E[31]                     | FC5C9186                            |
| Second AES Plaintext                        | FC5C9186 D2147C06 0EBA5B66 7A88B827 |
| Second AES Ciphertext $C2_0 \dots C2_{127}$ | CE5328AC 9D7FE109 B8FA9D2C B54D911E |
| DSAA2-2 Cipher Key E2[0]..E2[127]           | CE5328AC 9D7FE109 B8FA9D2C B54D911E |

## L.4.2 DCK Allocation through PP UAK Authentication

### L.4.2.0 Message sequence and coding

Secondly an authentication of the PP is performed with the derived UAK to derive a DCK, see table L.6.

**Table L.6: NWK messages for DCK allocation through PP UAK Authentication**

| Transmission direction | Message presentation                                  | Message coding  |
|------------------------|---|---|
| PP <<<<< FP            | {AUTH-REQUEST}<br><<AUTH-TYPE>><br><<RAND>><br><<RS>> | 05 40<br>0A 03 02 18 18<br>0C 08 6B A2 80 2B 91 0F F3 39<br>0E 10 3B 9F F6 1B DE 98 0B 45 EF A9 BF C5 D4 67 9C C9 |
| PP --->>> FP           | {AUTH-REPLY}<br><<RAND>><br><<RES>>                   | 85 41<br>0C 08 96 BD 28 62 C7 15 FC 88<br>0D 04 40 96 C0 AF   |

Below is a summary of the inputs, intermediate results, and outputs of the A11, and A12 processes involved in procedure outlined in table L.6.

### L.4.2.1 PP UAK Authentication

**Table L.7: Calculation for process A11 (DSAA2-1)**

| Variable                                | Hexadecimal value                   |
|---|-------------------------------------|
| DSAA2-1 Secret Input D1[0]..D1[127]     | CD257682 F4416053 7CD50DBF 1BDB145D |
| DSAA2-1 Public Input D2[0]..D2[63]      | 3B9FF61B DE980B45                   |
| DSAA2-1 Public Input D3[0]..D3[63]      | EFA9BFC5 D4679CC9                   |
| AES Key $K_0 \dots K_{127}$             | CD257682 F4416053 7CD50DBF 1BDB145D |
| AES Plaintext $P_0 \dots P_{127}$       | 3B9FF61B DE980B45 EFA9BFC5 D4679CC9 |
| AES Ciphertext $W_0 \dots W_{127}$      | 3B6E03AA C0ADA28A CE6A927A 564AA440 |
| DSAA2-1 Secret Session Key E[0]..E[127] | C3D25E38 117444ED 4761F67A DB94F80E |

**Table L.8: Calculation for process A12 (DSAA2-2)**

| Variable                                    | Hexadecimal value                   |
|---|-------------------------------------|
| DSAA2-2 Secret Input D1[0]..D1[127]         | C3D25E38 117444ED 4761F67A DB94F80E |
| DSAA2-2 Public Input D2[0]..D2[63]          | 6BA2802B 910FF339                   |
| DSAA2-2 Public Input D3[0]..D3[63]          | 96BD2862 C715FC88                   |
| AES Key $K_0 \dots K_{127}$                 | C3D25E38 117444ED 4761F67A DB94F80E |
| AES Plaintext $P_0 \dots P_{127}$           | 6BA2802B 910FF339 96BD2862 C715FC88 |
| AES Ciphertext $W_0 \dots W_{127}$          | 4096C0AF EEAA6826 91C381F8 62E26B1C |
| DSAA2-2 tag E[0]..E[31]                     | 4096C0AF                            |
| Second AES Plaintext                        | 4096C0AF EEAA6826 91C381F8 62E26B1D |
| Second AES Ciphertext $C2_0 \dots C2_{127}$ | 01A063BA 8E8A07CF 063C6E23 3405F14E |
| DSAA2-2 Cipher Key E2[0]..E2[127]           | 01A063BA 8E8A07CF 063C6E23 3405F14E |

### L.4.2.2 Derivation of 64 bit DCK for DSC

The following 64 bit DCK value would be derived for possible used by original DSC algorithm.

**Table L.9: Derivation of a 64 bit DCK for DSC**

|                               |        |
|-------------------------------|--------|
| LSB                           | MSB    |
| 01 A0 63 BA 8E 8A 07 CF       |        |
| Bit 0                         | Bit 63 |
| MSB = Most Significant Byte.  |        |
| LSB = Least Significant Byte. |        |

## L.5 DCK to CK mapping

This clause shows how to map the Derived Cipher Key (DCK) to the CK-vector of the encryption algorithm. Different implementation of the mapping leads into severe interoperability problems.

The examples will use the DCK-results of the previous clause and it is described how this is mapped to the CK of the encryption algorithm.

The following example shows how the 64 bit DCK from table L.9 is mapped to the CK used by the KSG of DSC (original).

**Table L.10: DCK to CK (for DSC) mapping**

|            |    |    |    |    |    |    |    |    |
|------------|----|----|----|----|----|----|----|----|
| <b>DCK</b> | 01 | A0 | 63 | BA | 8E | 8A | 07 | CF |
| <b>CK</b>  | CF | 07 | 8A | 8E | BA | 63 | A0 | 01 |

The next example concerns the CK for DSC2. The DCK for DSC2 is generated by process A12 (DSAA2-2), see table L.8, and is used directly. The mapping is as shown in next table L.11.

**Table L.11: DCK to CK (for DSC2) mapping**

|            |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
|------------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
| <b>DCK</b> | 01 | A0 | 63 | BA | 8E | 8A | 07 | CF | 06 | 3C | 6E | 23 | 34 | 05 | F1 | 4E |
| <b>CK</b>  | 01 | A0 | 63 | BA | 8E | 8A | 07 | CF | 06 | 3C | 6E | 23 | 34 | 05 | F1 | 4E |



---

## Annex M (normative): DECT Standard Cipher #2 (DSC2)

### M.1 Introduction

The DECT Standard Cipher is a stream cipher algorithm. A sequence generator is loaded with key material, and then used to generate a sequence of keystream bits. The keystream bits thus produced are bitwise XORed with a sequence of plaintext bits to yield a sequence of ciphertext bits.

The DSC2 algorithm has the following parameters:

- Inputs:
  - An  $m$ -bit cipher key  $CK[0] \dots CK[m-1]$ , where  $m \leq 128$
  - An  $n$ -bit initialization vector  $IV[0] \dots IV[n-1]$ , where  $n \leq 64$
- Output:
  - $N$  sequences of  $\lambda$  keystream bits  $KSS[0] \dots KSS[\lambda-1]$ , as required, where  $N = 2$  and  $\lambda \leq 4\ 840$

NOTE 1: The values of  $N = 2$  and  $\lambda \leq 4840$  come from the DECT system, not from the cipher (the cipher defined below could safely generate much longer keystream sequences if required).

NOTE 2: The DSC cipher (annex J) only accommodates a cipher key of length  $m \leq 64$ .

---

### M.2 Operation of the Cipher

Let the 128-bit sequence  $K_0 \dots K_{127}$  be defined as follows:

- $K_0 \dots K_{m-1} = CK[0] \dots CK[m-1]$
- If  $m < 128$ , then  $K_i = 0$  for  $m \leq i \leq 127$

Let  $L$  be the smallest integer such that  $128 \times L \geq \lambda$ . For  $0 \leq j \leq L-1$ , compute (up to) 128 bits of keystream as follows:

- Define a 128-bit sequence  $P_{j,0} \dots P_{j,127}$  by:
  - $P_{j,63} \dots P_{j,64-n} = IV[0] \dots IV[n-1]$
  - If  $n < 64$ , then  $P_{j,63-i} = 0$  for  $n \leq i \leq 63$
  - $P_{j,64} \dots P_{j,127}$  represent the 64-bit unsigned integer  $j$ , with  $P_{j,64}$  being the most significant bit and  $P_{j,127}$  the least significant bit. (For example, when  $j=3$ ,  $P_{j,64} \dots P_{j,125}$  are all zeroes and  $P_{j,126} = P_{j,127} = 1$ .)
- Perform an AES block cipher encryption, as specified in [10]. Use AES with the 128-bit key  $K_0 \dots K_{127}$  and the 128-bit plaintext block  $P_{j,0} \dots P_{j,127}$ . Let the resulting ciphertext block be  $C_{j,0} \dots C_{j,127}$ .
- The bits of this AES ciphertext block are used as keystream bits. If at least 128 new keystream bits are required - i.e. if  $\lambda \geq 128 \times (j+1)$  - then  $KSS[(128 \times j) + i] = C_{j,i}$  for  $0 \leq i \leq 127$ . If fewer than 128 new keystream bits are required - i.e. if  $\lambda < 128 \times (j+1)$  - then  $KSS[(128 \times j) + i] = C_{j,i}$  for  $0 \leq i \leq \lambda - (128 \times j) - 1$ .

## M.3 Test Sets

The test data sets presented below are for the cryptographic kernel function AES.

This first test set is shown twice, once in binary format, once in hexadecimal format. This is to show explicitly the relationship between the binary data and the hexadecimal representation. The remainder of the test sets are presented in hexadecimal format only. A 128-bit hexadecimal number will be broken up in four 32-bit words. For example, the number AE5069178ABA73C8CD5D0D6D45F37D0D will be written as:

AE506917 8ABA73C8 CD5D0D6D 45F37D0D.

### Binary format

AES Plaintext  $P_{3,0} \dots P_{3,127}$ : 00000000 00010001 00100010 00110011 01000100 01010101 01100110 01110111  
 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000011

AES Ciphertext  $C_{3,0} \dots C_{3,127}$ : 11001010 11110011 00100101 10110111 10010000 11011111 01111000 00001100  
 11100111 10011110 10011100 00000011 11100011 01010001 00110111 00110101

AES Key  $K_0 \dots K_{127}$ : 00000000 00000001 00000010 00000011 00000100 00000101 00000110 00000111  
 00001000 00001001 00001010 00001011 00001100 00001101 00001110 00001111

### Hexadecimal format

AES Plaintext  $P_{3,0} \dots P_{3,127}$ : 00112233 44556677 00000000 00000003

AES Ciphertext  $C_{3,0} \dots C_{3,127}$ : CAF325B7 90DF780C E79E9C03 E3513735

AES Key  $K_0 \dots K_{127}$ : 00010203 04050607 08090A0B 0C0D0E0F

| Test Set | AES Key $K_0 \dots K_{127}$            | Block j | AES Plaintext $P_{j,0} \dots P_{j,127}$ | AES Ciphertext $C_{j,0} \dots C_{j,127}$ |
|----------|--|---------|---|--|
| 1        | 00010203 04050607<br>08090A0B 0C0D0E0F | 3       | 00112233 44556677<br>00000000 00000003  | CAF325B7 90DF780C<br>E79E9C03 E3513735   |
| 2        | 18BE6784 4AE13D6C<br>20000000 00000000 | 5       | 6DF15AF1 41BB26E9<br>00000000 00000005  | 32A30C3E BE25BD4D<br>52F051D8 697F34B2   |
| 3        | 2EA612DB 153C6000<br>00000000 00000000 | 32      | 440D491C 4D064DB7<br>00000000 00000020  | 8E26C2B0 CE13908E<br>926D2121 63399B2D   |
| 4        | 39B32D12 074D4DC8<br>00000000 00000000 | 13      | 5D037A5A 767D4509<br>00000000 0000000D  | 19C2FB26 47032E3A<br>E8030B1B B34674D1   |
| 5        | 3B251E1F 6E5D1AD4<br>63CB6BFC 7F967FF5 | 11      | 323B2213 260D6B89<br>00000000 0000000B  | 8753B38E E1D28683<br>4612D684 B71A1120   |
| 6        | 301C0BDB 56AE0732<br>0120759A 235022EE | 36      | 58786B36 5CFD3E12<br>00000000 00000024  | 24E689DD C3950DBB<br>B496F109 B2E58EAB   |
| 7        | 3BF63A9E 797D5F49<br>0DDC4CAD 314F4000 | 0       | 49442E40 13661CD0<br>00000000 00000000  | 28B505F5 9336D2AB<br>F4E98E65 7B751544   |
| 8        | 42307EB7 60000000<br>00000000 00000000 | 15      | 409D12E1 798B121F<br>00000000 0000000F  | 8ECE05E6 4EC7F08A<br>BADFB269 50382207   |
| 9        | 58B026CA 36990902<br>7BB95772 139D7049 | 20      | 4A80187E 16C56899<br>00000000 00000014  | F939F162 D5CD9C9C<br>9512E4D4 0D57C218   |
| 10       | 40805DB2 33800000<br>00000000 00000000 | 32      | 0FBF2F14 6AD6047E<br>00000000 00000020  | 0FCFCF7F E4273435<br>5BA01401 E69FFAAE   |

The following test sets are shown in hexadecimal format only. They represent 128-bit AES plaintext and AES ciphertext blocks used to form the  $\lambda$  bit keystream. The first  $j$  blocks are used entirely, and the last block is truncated to its first  $\lambda - (128 \times j)$  bits. For example if  $\lambda = 720$  keystream bits are required, five blocks of 128 bits of output are generated in counter mode, and the output of the 6<sup>th</sup> block is truncated to its 80 leftmost bits.

Test Set 11:  $\lambda = 720$  bits

| Variable                                 | Hexadecimal value                   |
|--|-------------------------------------|
| DSC2 Cipher Key CK[0]..CK[127]           | 5E9D489C 19166172 6B7232E6 401D71F0 |
| DSC2 Initialization Vector IV[0]..IV[63] | 03847F4F 494A0677                   |
| AES Key $K_0 \dots K_{127}$              | 5E9D489C 19166172 6B7232E6 401D71F0 |
| AES Plaintext $P_{0,0} \dots P_{0,127}$  | 03847F4F 494A0677 00000000 00000000 |
| AES Ciphertext $C_{0,0} \dots C_{0,127}$ | B5D23E32 B712B63E 376360FB 522CB1F5 |
| DSC2 Keystream KSS[0]..KSS[127]          | B5D23E32 B712B63E 376360FB 522CB1F5 |
| AES Plaintext $P_{1,0} \dots P_{1,127}$  | 03847F4F 494A0677 00000000 00000001 |
| AES Ciphertext $C_{1,0} \dots C_{1,127}$ | 1EDD0ED9 8B555347 7227CA87 1B8E652A |
| DSC2 Keystream KSS[128]..KSS[255]        | 1EDD0ED9 8B555347 7227CA87 1B8E652A |
| AES Plaintext $P_{2,0} \dots P_{2,127}$  | 03847F4F 494A0677 00000000 00000002 |
| AES Ciphertext $C_{2,0} \dots C_{2,127}$ | DBB64A1C A85427BE AC153100 E95F5C0E |
| DSC2 Keystream KSS[256]..KSS[383]        | DBB64A1C A85427BE AC153100 E95F5C0E |
| AES Plaintext $P_{3,0} \dots P_{3,127}$  | 03847F4F 494A0677 00000000 00000003 |
| AES Ciphertext $C_{3,0} \dots C_{3,127}$ | F7904EAE 70CF0F20 949D1C47 8835EE69 |
| DSC2 Keystream KSS[384]..KSS[511]        | F7904EAE 70CF0F20 949D1C47 8835EE69 |
| AES Plaintext $P_{4,0} \dots P_{4,127}$  | 03847F4F 494A0677 00000000 00000004 |
| AES Ciphertext $C_{4,0} \dots C_{4,127}$ | D4CDE775 AECF7B7E 65C00034 3088C020 |
| DSC2 Keystream KSS[512]..KSS[639]        | D4CDE775 AECF7B7E 65C00034 3088C020 |
| AES Plaintext $P_{5,0} \dots P_{5,127}$  | 03847F4F 494A0677 00000000 00000005 |
| AES Ciphertext $C_{5,0} \dots C_{5,127}$ | D99C1529 E6D636F7 7E328DDB 0B3E21FD |
| DSC2 Keystream KSS[640]..KSS[719]        | D99C1529 E6D636F7 7E32              |

Test Set 12:  $\lambda = 720$  bits

| Variable                                 | Hexadecimal value                   |
|--|-------------------------------------|
| DSC2 Cipher Key CK[0]..CK[61]            | 285248DB 27251640                   |
| DSC2 Initialization Vector IV[0]..IV[63] | 13D329D8 0A2809CE                   |
| AES Key $K_0 \dots K_{127}$              | 285248DB 27251640 00000000 00000000 |
| AES Plaintext $P_{0,0} \dots P_{0,127}$  | 13D329D8 0A2809CE 00000000 00000000 |
| AES Ciphertext $C_{0,0} \dots C_{0,127}$ | 7E7246AF 0B138505 D1AFD2B7 8A8ADD1C |
| DSC2 Keystream KSS[0]..KSS[127]          | 7E7246AF 0B138505 D1AFD2B7 8A8ADD1C |
| AES Plaintext $P_{1,0} \dots P_{1,127}$  | 13D329D8 0A2809CE 00000000 00000001 |
| AES Ciphertext $C_{1,0} \dots C_{1,127}$ | D8D56D3C D8E8E137 2044DFFD 2DDF0257 |
| DSC2 Keystream KSS[128]..KSS[255]        | D8D56D3C D8E8E137 2044DFFD 2DDF0257 |
| AES Plaintext $P_{2,0} \dots P_{2,127}$  | 13D329D8 0A2809CE 00000000 00000002 |
| AES Ciphertext $C_{2,0} \dots C_{2,127}$ | BFB4BC89 7F20B8F4 B3BF0A77 CFFB95F7 |
| DSC2 Keystream KSS[256]..KSS[383]        | BFB4BC89 7F20B8F4 B3BF0A77 CFFB95F7 |
| AES Plaintext $P_{3,0} \dots P_{3,127}$  | 13D329D8 0A2809CE 00000000 00000003 |
| AES Ciphertext $C_{3,0} \dots C_{3,127}$ | 006D0A16 E03970F0 057CA562 4DCE6451 |
| DSC2 Keystream KSS[384]..KSS[511]        | 006D0A16 E03970F0 057CA562 4DCE6451 |
| AES Plaintext $P_{4,0} \dots P_{4,127}$  | 13D329D8 0A2809CE 00000000 00000004 |
| AES Ciphertext $C_{4,0} \dots C_{4,127}$ | 9AF557A7 2570FA1C AB64AACF 694EA025 |
| DSC2 Keystream KSS[512]..KSS[639]        | 9AF557A7 2570FA1C AB64AACF 694EA025 |
| AES Plaintext $P_{5,0} \dots P_{5,127}$  | 13D329D8 0A2809CE 00000000 00000005 |
| AES Ciphertext $C_{5,0} \dots C_{5,127}$ | 13BDBDE7 3CB9A76C E2329591 2ABDAE34 |
| DSC2 Keystream KSS[640]..KSS[719]        | 13BDBDE7 3CB9A76C E232              |

Test Set 13:  $\lambda = 720$  bits

| Variable                                 | Hexadecimal value                   |
|--|-------------------------------------|
| DSC2 Cipher Key CK[0]..CK[45]            | 06E30A6C 4328                       |
| DSC2 Initialization Vector IV[0]..IV[31] | 12C21003                            |
| AES Key $K_0 \dots K_{127}$              | 06E30A6C 43280000 00000000 00000000 |
| AES Plaintext $P_{0,0} \dots P_{0,127}$  | 12C21003 00000000 00000000 00000000 |
| AES Ciphertext $C_{0,0} \dots C_{0,127}$ | D3163CBE 4E3BD210 DC1900B5 1FA65322 |
| DSC2 Keystream KSS[0]..KSS[127]          | D3163CBE 4E3BD210 DC1900B5 1FA65322 |
| AES Plaintext $P_{1,0} \dots P_{1,127}$  | 12C21003 00000000 00000000 00000001 |
| AES Ciphertext $C_{1,0} \dots C_{1,127}$ | 76D1AEE3 36505B2B 1FFAB960 EE92EDC9 |
| DSC2 Keystream KSS[128]..KSS[255]        | 76D1AEE3 36505B2B 1FFAB960 EE92EDC9 |
| AES Plaintext $P_{2,0} \dots P_{2,127}$  | 12C21003 00000000 00000000 00000002 |
| AES Ciphertext $C_{2,0} \dots C_{2,127}$ | A58AAF8B F489D659 30210BA3 BB4706CD |
| DSC2 Keystream KSS[256]..KSS[383]        | A58AAF8B F489D659 30210BA3 BB4706CD |
| AES Plaintext $P_{3,0} \dots P_{3,127}$  | 12C21003 00000000 00000000 00000003 |
| AES Ciphertext $C_{3,0} \dots C_{3,127}$ | AF5CB0B6 058B7441 BA78C21D 71FCE174 |
| DSC2 Keystream KSS[384]..KSS[511]        | AF5CB0B6 058B7441 BA78C21D 71FCE174 |
| AES Plaintext $P_{4,0} \dots P_{4,127}$  | 12C21003 00000000 00000000 00000004 |
| AES Ciphertext $C_{4,0} \dots C_{4,127}$ | 0CBE6B10 93F9EEDC 1EA51372 57467E8E |
| DSC2 Keystream KSS[512]..KSS[639]        | 0CBE6B10 93F9EEDC 1EA51372 57467E8E |
| AES Plaintext $P_{5,0} \dots P_{5,127}$  | 12C21003 00000000 00000000 00000005 |
| AES Ciphertext $C_{5,0} \dots C_{5,127}$ | 8453DFBE 73D2465B 8EF4DFB7 57D3C4D1 |
| DSC2 Keystream KSS[640]..KSS[719]        | 8453DFBE 73D2465B 8EF4              |

## M.4 DSC2 Test Set

The test data sets presented below are for clarification of mapping between realistic DECT parameters and the cryptographic kernel function AES.

This test set data is shown in hexadecimal format, unless otherwise specified. A 128-bit hexadecimal number will be broken up in four 32-bit words.

### Double-Slot/Unprotected/Configuration 1a/1b, $\lambda = 1\ 360$ bits

| variable                                 | Value (Hexadecimal, except when otherwise noted) |
|--|--|
| Number of KSS bits                       | 1360 (decimal)                                   |
| DSC2 Cipher Key CK[0]..CK[127]           | 01A063BA 8E8A07CF 063C6E23 3405F14E              |
| Multi-frame number                       | 112233   |
| Frame number                             | 4  |
| LBN                                      | A  |
| LBN*                                     | 5  |
| AES Key $K_0 \dots K_{127}$              | 01A063BA 8E8A07CF 063C6E23 3405F14E              |
| AES Plain Text $P_{0,0} \dots P_{0,127}$ | 00000000 51122334 00000000 00000000              |
| AES Ciphertext $C_{0,0} \dots C_{0,127}$ | A5F3E18F 28BF5630 9FBB34B2 C0514789              |
| DSC2 Keystream KSS[0]..KSS[127]          | A5F3E18F 28BF5630 9FBB34B2 C0514789              |
| AES Plain Text $P_{1,0} \dots P_{1,127}$ | 00000000 51122334 00000000 00000001              |
| AES Ciphertext $C_{1,0} \dots C_{1,127}$ | D4C4E706 85E7B5F2 89E83A32 04C6F8E0              |
| DSC2 Keystream KSS[128]..KSS[255]        | D4C4E706 85E7B5F2 89E83A32 04C6F8E0              |
| AES Plain Text $P_{2,0} \dots P_{2,127}$ | 00000000 51122334 00000000 00000002              |
| AES Ciphertext $C_{2,0} \dots C_{2,127}$ | 83526749 43823181 C02E2B9D B79C192E              |
| DSC2 Keystream KSS[256]..KSS[383]        | 83526749 43823181 C02E2B9D B79C192E              |
| AES Plain Text $P_{3,0} \dots P_{3,127}$ | 00000000 51122334 00000000 00000003              |
| AES Ciphertext $C_{3,0} \dots C_{3,127}$ | 26EB9801 23B431D6 A9450A30 AC9C1E55              |
| DSC2 Keystream KSS[384]..KSS[511]        | 26EB9801 23B431D6 A9450A30 AC9C1E55              |

| variable   | Value (Hexadecimal, except when otherwise noted)  |
|--|---|
| AES Plain Text $P_{4,0}..P_{4,127}$  | 00000000 51122334 00000000 00000004   |
| AES Ciphertext $C_{4,0}..C_{4,127}$  | CFA42FBE 32518449 434F5D79 BC90799D   |
| DSC2 Keystream KSS[512]..KSS[639]  | CFA42FBE 32518449 434F5D79 BC90799D   |
| AES Plain Text $P_{5,0}..P_{5,127}$  | 00000000 51122334 00000000 00000005   |
| AES Ciphertext $C_{5,0}..C_{5,127}$  | 158D6BD0 4DBDAE69 06361ACB 03009C1D   |
| DSC2 Keystream KSS[640]..KSS[767]  | 158D6BD0 4DBDAE69 06361ACB 03009C1D   |
| AES Plain Text $P_{6,0}..P_{6,127}$  | 00000000 51122334 00000000 00000006   |
| AES Ciphertext $C_{6,0}..C_{6,127}$  | B20B01AB 2EEA602F 3689269D C4C23A28   |
| DSC2 Keystream KSS[768]..KSS[895]  | B20B01AB 2EEA602F 3689269D C4C23A28   |
| AES Plain Text $P_{7,0}..P_{7,127}$  | 00000000 51122334 00000000 00000007   |
| AES Ciphertext $C_{7,0}..C_{7,127}$  | A09785E8 5CD61D44 23A7BB99 6BD0D401   |
| DSC2 Keystream KSS[896]..KSS[1023]   | A09785E8 5CD61D44 23A7BB99 6BD0D401   |
| AES Plain Text $P_{8,0}..P_{8,127}$  | 00000000 51122334 00000000 00000008   |
| AES Ciphertext $C_{8,0}..C_{8,127}$  | F25CCE27 33843EEB B4999B01 858677DC   |
| DSC2 Keystream KSS[1024]..KSS[1151]  | F25CCE27 33843EEB B4999B01 858677DC   |
| AES Plain Text $P_{9,0}..P_{9,127}$  | 00000000 51122334 00000000 00000009   |
| AES Ciphertext $C_{9,0}..C_{9,127}$  | A86BBA05 3EE32E5B B22F674E 0C09A863   |
| DSC2 Keystream KSS[1152]..KSS[1279]  | A86BBA05 3EE32E5B B22F674E 0C09A863   |
| AES Plain Text $P_{10,0}..P_{10,127}$                                      | 00000000 51122334 00000000 0000000A   |
| AES Ciphertext $C_{10,0}..C_{10,127}$                                      | 287D60CC 937130E8 09DAA1CF 40EF772F   |
| DSC2 Keystream KSS[1280]..KSS[1359]  | 287D60CC 937130E8 09DA  |
| DSC2 Keystream KSS[0]..KSS[1359]   | A5 F3 E1 8F 28 BF 56 30 9F BB 34 B2 C0 51 47 89<br>D4 C4 E7 06 85 E7 B5 F2 89 E8 3A 32 04 C6 F8 E0<br>83 52 67 49 43 82 31 81 C0 2E 2B 9D B7 9C 19 2E<br>26 EB 98 01 23 B4 31 D6 A9 45 0A 30 AC 9C 1E 55<br>CF A4 2F BE 32 51 84 49 43 4F 5D 79 BC 90 79 9D<br>15 8D 6B D0 4D BD AE 69 06 36 1A CB 03 00 9C 1D<br>B2 0B 01 AB 2E EA 60 2F 36 89 26 9D C4 C2 3A 28<br>A0 97 85 E8 5C D6 1D 44 23 A7 BB 99 6B D0 D4 01<br>F2 5C CE 27 33 84 3E EB B4 99 9B 01 85 86 77 DC<br>A8 6B BA 05 3E E3 2E 5B B2 2F 67 4E 0C 09 A8 63<br>28 7D 60 CC 93 71 30 E8 09 DA |
| DSC2 Keystream KSS[0]..KSS[679]<br>(used in 1 <sup>st</sup> half frame)    | A5 F3 E1 8F 28 BF 56 30 9F BB 34 B2 C0 51 47 89<br>D4 C4 E7 06 85 E7 B5 F2 89 E8 3A 32 04 C6 F8 E0<br>83 52 67 49 43 82 31 81 C0 2E 2B 9D B7 9C 19 2E<br>26 EB 98 01 23 B4 31 D6 A9 45 0A 30 AC 9C 1E 55<br>CF A4 2F BE 32 51 84 49 43 4F 5D 79 BC 90 79 9D<br>15 8D 6B D0 4D   |
| DSC2 Keystream KSS[680]..KSS[1359]<br>(used in 2 <sup>nd</sup> half frame) | BD AE 69 06 36 1A CB 03 00 9C 1D B2 0B 01 AB 2E<br>EA 60 2F 36 89 26 9D C4 C2 3A 28 A0 97 85 E8 5C<br>D6 1D 44 23 A7 BB 99 6B D0 D4 01 F2 5C CE 27 33<br>84 3E EB B4 99 9B 01 85 86 77 DC A8 6B BA 05 3E<br>E3 2E 5B B2 2F 67 4E 0C 09 A8 63 28 7D 60 CC 93<br>71 30 E8 09 DA   |
| A-field data   | 01 2C 01 43 1D BD   |
| A-field encrypted (1 <sup>st</sup> half frame)                             | 01 89 F2 A2 92 95   |
| A-field encrypted (2 <sup>nd</sup> half frame)                             | 01 91 AF 2A 1B 8B   |

| variable                                       | Value (Hexadecimal, except when otherwise noted)  |
|--|---|
| B-field data                                   | BF 4E F6 BC FC 7D B5 AF F5 FF F8 F0 F0 73 DB 2C<br>6E DB F3 F4 B0 BC 6E BC F3 F7 FA 75 75 58 73 3B<br>37 5A 54 B5 F1 DD FB 70 DE 1E 7C 55 D6 B9 52 59<br>FF D7 F7 90 70 9B 55 1C D8 0F 53 D6 7A 7C FE BB<br>5B 33 1C AB D4 DF 3F 15 56 D8 FA 71 D8 D1 F1 F6 |
| B-field encrypted (1 <sup>st</sup> half frame) | 00 18 C6 23 47 49 07 6F A4 B8 71 24 34 94 DD A9<br>89 6E 01 7D 58 86 5C B8 35 0F 1A F6 27 3F 3A 78<br>B5 6B D5 75 DF F6 66 C7 42 07 52 73 3D 21 53 7A<br>4B E6 21 39 35 91 65 B0 44 11 06 19 DE 53 40 89<br>0A B7 55 E8 9B 82 46 A9 C6 A1 67 64 55 BA 21 BB |
| B-field encrypted (2 <sup>nd</sup> half frame) | A5 85 F5 BC 60 60 07 A4 F4 54 D6 1A 90 5C ED A5<br>48 46 37 36 8A 94 CE 2B 76 1F A6 A3 68 1C 50 9C<br>8C C3 3F 65 25 DC 09 2C 10 39 4F D1 E8 52 E6 C0<br>64 D6 72 16 07 47 FD 77 62 0A 6D 35 54 27 4C 94<br>3C 7D 10 A2 7C BC 17 68 36 14 69 00 E8 39 F8 2C |

## M.5 Mapping of DECT values into AES-128 plaintext

The DECT values frame number, multiframe number and LBN\* shall be mapped into the AES-128 plaintext vector as shown in figure M.1.

| AES-128 plaintext      |                  |                                |                          |                |
|------------------------|------------------|--------------------------------|--------------------------|----------------|
| $P_0$                  |                  |                                |                          | $P_{127}$      |
| 000...000<br>(32 bits) | LBN*<br>(4 bits) | Multiframe number<br>(24 bits) | Frame number<br>(4 bits) | j<br>(64 bits) |
| msb                    | lsb              | msb                            | lsb                      | msb            |
|                        |                  |                                |                          | lsb            |

Figure M.1: Mapping of DECT values into AES-128 plaintext

---

## Annex N (normative): CCM Authenticated Encryption

### N.1 Introduction

#### N.1.0 General

CCM (Counter with CBC-MAC) is an Authenticated Encryption algorithm designed to provide both packet authentication and encryption. It is defined by the IETF RFC 3610 [11]. The DECT implementation of CCM uses the Advanced Encryption Standard (AES) as 128 bit block cipher (IETF RFC 3610 [11] may, in theory, use other ciphers). Apart from that and from the definition of the parameters of the Initialization Vector, it follows exactly the implementation proposed by the RFC.

CCM is intended for use as cipher operating at DLC level, for instance in DLC service LU14. The AES block cipher is identical to the one used in DSAA2 authentication algorithm and DSC2 cipher. This allows compatible security strength between ciphering and key generation and easy implementation.

#### N.1.1 Key management

CCM uses 128 bit keys. In order to preserve the security strength, keys should be used only once for each value of the nonce (part of the initialization vector, see clause 6.6.2.3). In practice, it means that a new key should be generated and used, each time the CCM sequence is reset.

In addition to that, the key used for CCM authenticated encryption, should not be reused by any other security process. This means that if both CCM and DSC or DSC2 MAC encryption are used, different keys should be used.

The DSAA2 algorithm provides authentication and key generation mechanisms of compatible security level. DSAA is weaker than CCM. Therefore, the use of DSAA2 for authentication generation of the keys is recommended.

---

## N.2 Operation of the CCM encryption algorithm

### N.2.0 Description of the CCM algorithm: general

As defined in clause 4.5.5, CCM security process can be modelled as shown in figure N.1, where an input stream of arbitrary size  $I(n)$  is encrypted with a Key  $K(128)$  and Initialization Vector  $IV(128)$ , both of 128 bits. The security process generates an output stream  $O(n)$  of the same length plus a Message Integrity Code  $M(32)$  of 32 bits. The MIC is concatenated to the output stream, as shown, for instance, in DLC service LU14 (see clause 11.16 of ETSI EN 300 175-4 [4]).

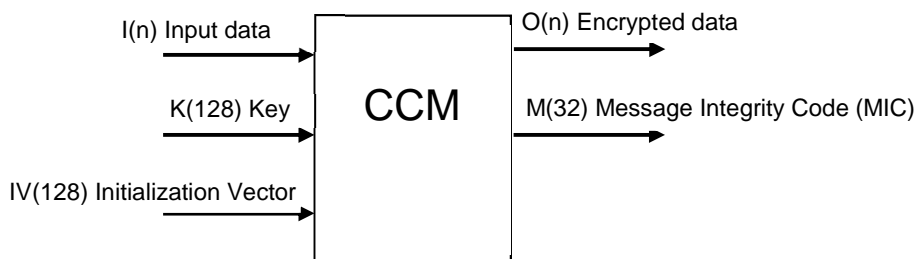


Figure N.1: CCM security process overview





### N.2.1.3 AES block "B" and generation of the encryption stream

The AES block "B" generates a sequence of 128 bits segments  $S_x$ , where  $x$  is the value of the counter.

$$S_x = \text{AES}(K, A_x)$$

The first segment,  $S_0$ , is reserved and will be used in the generation of the MIC. All other segments  $S_1, S_2 \dots S_n$  and concatenated in sequence and will form the encryption stream.

The Output stream  $O(n)$  is obtained by XORing the encryption stream by the message  $I(n)$ .

NOTE: IETF RFC 3610 [11] description uses the terminology  $B_1 \dots B_n$  for the 128 bit blocks of the message. This is shown in the diagram.

### N.2.1.4 AES block "A" and generation of the Message Integrity Code (MIC)

The AES block "A" is used to generate the Message Integrity Code (MIC) as function of the Key, the IV and the message itself as follows:

The outputs of the AES block "A" are named  $X_1 \dots X_n$ , being  $X_1$  the first generated segment.

$X_1$  is generated from the Key and the Initialization Vector (IV, named  $B_0$  in IETF RFC 3610 [11]) with the equation:

$$X_1 = \text{AES}(K, IV)$$

The following  $X$  segments ( $X_2 \dots X_n$ ) are generated with the equation:

$$X_{i+1} = \text{AES}(K, X_i \text{ XOR } B_i)$$

Where  $B_i$  ( $B_1 \dots B_n$ ) are 128 bit segments of the message  $I(n)$ . The last segment is padded with "0" to fill in the 128 bits.

The MAC value "T" is generated taken the first 32 bits from the last  $X$  segment,  $X_{n+1}$ .

The authentication value "U" is computed by XORing "T" with the first 32 bits of the segment  $S_0$  generated by AES block "B".

NOTE: It means that segment  $S_0$ , if generated by the first execution of "B", has to be stored. Another option is generating  $S_0$  after the other  $S_x$  segments, what is possible due to the nature of the CTR function.

### N.2.1.5 "c" stream

The MIC of 32 bits is added after the last bit of the encrypted message  $O(n)$ .

IETF RFC 3610 [11] uses the name "c" for this final stream encrypted message + MIC.

## N.2.2 Description of the CCM algorithm: decoding

Decoding of a CCM encoded stream is implemented by generating the ciphering stream from  $K$  and  $IV$  as described in clause N.2.1.3, and XORing the received message  $O(n)$  with this stream. The MIC can be computed from the decoded message  $I(n)$ , the Key and the IV as described in clause N.2.1.4. The message integrity is checked by comparing the computed MIC with the MIC transmitted at the end of the message. If the integrity is not correct, the decoder shall not reveal the message.

## Annex O (informative): Change history

The following table presents main changes from a published version to the next version (published or to be published).

| Subject/Comment   | Old   | New   |
|---|-------|-------|
| The enhancement of the DECT base standard to support higher data rates includes the 16 QAM/64 QAM modulation option and the Channel Coding based on the Turbo Code Principle.   | 1.6.1 | 1.7.1 |
| Some minor editorial improvements.  | 1.7.1 | 1.8.1 |
| Diagrams and tables redrawn to improve presentation and avoid misinterpretation, editorial errors and misleading statements repaired.   | 1.8.1 | 1.9.1 |
| New Generation DECT: A major revision of the DECT base standard introducing wideband speech, improved data services, new slot types and other technical enhancements.   | 1.9.1 | 2.1.1 |
| No changes.   | 2.1.1 | 2.2.1 |
| Backcompatible change in the initialization vector of the encryption process for advanced connections; Re-Keying and Default Cipher mechanism to allow immediate encryption.  | 2.2.1 | 2.3.1 |
| New authentication and ciphering algorithms DSAA2 and DSC2 based on AES. New authentication procedures. Encryption criteria for $I_{PX}$ encoded protected service. Editorial review.   | 2.3.1 | 2.4.1 |
| New CCM authenticated encryption algorithm based on AES and associated procedures (used by new DLC service LU14 and by DECT Ultra Low Energy); Implementation guidelines and CK mapping for DSAA2; Editorial review.  | 2.4.1 | 2.5.1 |
| Introduction of CCM encryption for multicast channels; CCM encryption of Service Channels (IWU-to-IWU transport); Multicast encryption parameter assignation and retrieval procedures using {MM-INFO-SUGGEST} transport.  | 2.5.1 | 2.6.1 |
| Review of security procedures; New description of early encryption: New description of re-keying; Procedure and model for re-keying in systems with WRSs; New Cipher Key retrieval procedure using {MM-INFO-SUGGEST}; Transport of Default Cipher Keys using {MM-INFO-SUGGEST}; Cipher switching procedure from DCK to Default Cipher Keys. | 2.6.1 | 2.7.1 |
| Correction of IV for ULE Multicast transmissions  | 2.7.1 | 2.8.1 |
| Editorial update  | 2.8.1 | 2.9.1 |

## History

| <b>Document history</b> |                |   |
|-------------------------|----------------|---|
| Edition 1               | October 1992   | Publication as ETSI ETS 300 175-7 (Historical)              |
| Edition 2               | September 1996 | Publication as ETSI ETS 300 175-7 (Historical)              |
| Edition 3               | September 1997 | Publication as ETSI ETS 300 175-7 (Historical)              |
| V1.4.2                  | June 1999      | Publication   |
| V1.5.1                  | February 2001  | Publication   |
| V1.6.1                  | February 2002  | Publication   |
| V1.7.1                  | July 2003      | Publication   |
| V1.8.1                  | November 2004  | Publication   |
| V1.9.1                  | September 2005 | Publication   |
| V2.1.1                  | August 2007    | Publication   |
| V2.2.1                  | November 2008  | Publication   |
| V2.3.1                  | June 2010      | Publication   |
| V2.4.1                  | April 2012     | Publication   |
| V2.5.1                  | August 2013    | Publication   |
| V2.6.1                  | July 2015      | Publication   |
| V2.7.1                  | November 2017  | Publication   |
| V2.8.1                  | December 2019  | Publication   |
| V2.8.5                  | December 2021  | EN Approval Procedure AP 20220323: 2021-12-23 to 2022-03-23 |