Draft ETSI EN 300 175-7 V2.5.0 (2013-04)

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

Reference
REN/DECT-000268-7

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 - NAF 742 C
Association à but non lucratif enregistrée à la
Sous-Préfecture de Grasse (06) N° 7803/88

Contents

Intellectual Property Rights

IPRs essential or potentially essential to the present document may have been declared to ETSI. The information pertaining to these essential IPRs, if any, is 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 (http://ipr.etsi.org).

Pursuant to the ETSI IPR Policy, no investigation, 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.

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 TR 101 178 [i.1] and ETR 043 [i.2].

Proposed national transposition dates	
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):	6 months after doa

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.

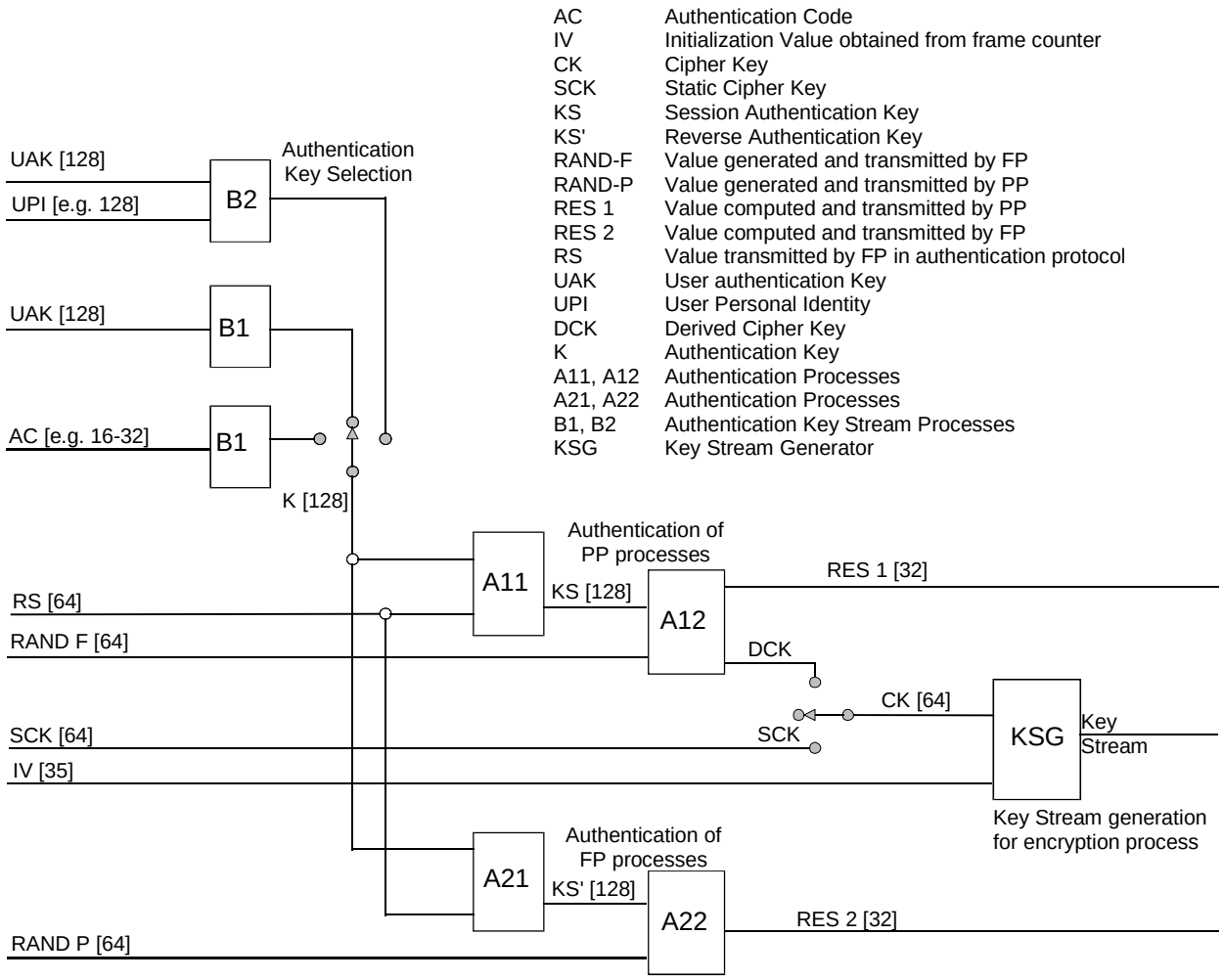


Figure 0.1: Overview of DECT historic security processes (until revision V2.3.1 of the present document)

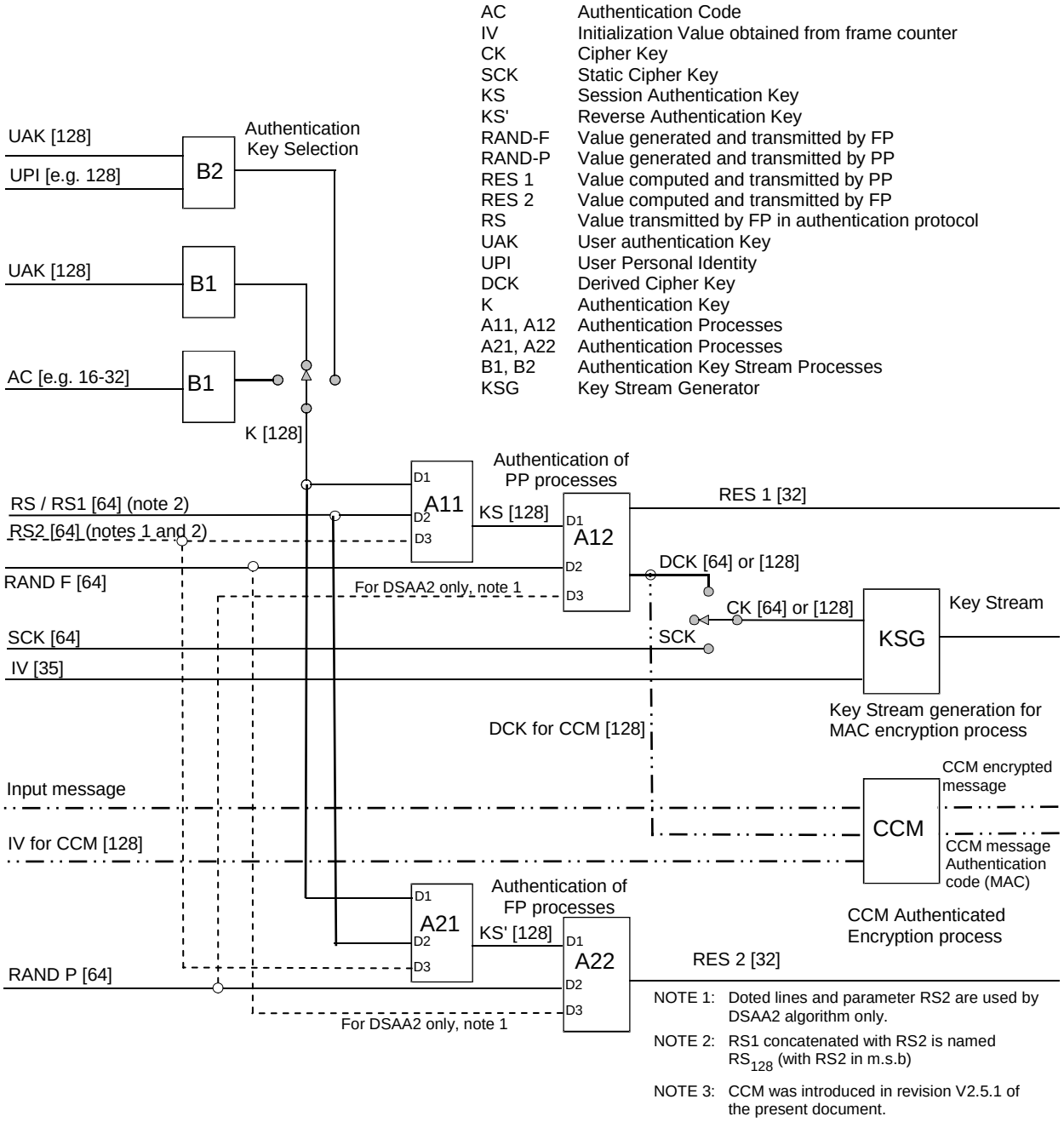


Figure 0.2: Overview of DECT current security processes (from revisions V2.4.1 and V2.5.1 of the present document)

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 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 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

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 reference document (including any amendments) applies.

Referenced documents which are not found to be publicly available in the expected location might be found at http://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.

2.1 Normative references

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)".

2.2 Informative references

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)".

3 Definitions and abbreviations

3.1 Definitions

For the purposes of the present document, the terms and definitions given in 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: a cryptographic parameter used in processes A11 and A21 for the calculation of authentication session keys

RS_{128}: 128 bit variant of RS

RS_{128}': the RS_{128} used for the generation of KS'

RS1 / RS2: components of RS_{128} when RS_{128} is assembled from two parts

XRES1: an eXpected RESponse calculated by a FT

XRES2: an eXpected RESponse calculated by a PT

3.2 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_F	higher layer signalling Channel (fast)
CI	Common Interface
CK	Cipher Key
CL_S	higher layer Connectionless channel (slow)

C-plane	Control plane
CRC	Cyclic Redundancy Check
CRFP	Cordless Radio Fixed Part
C_S	higher layer signalling Channel (slow)
C_T	one C_S or CL_S channel segment
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
G_F	higher layer information control channel
GFSK	Gaussian Frequency Shift Keying
GSM	Global System for Mobile communications
HDB	Home Data Base
ICC	Integrated Chip Card
IE	Information Element
I_N	higher layer Information channel (unprotected) in general
I_P	higher layer Information channel (protected) in general
I_{PQ}	higher layer Information channel (protected) with single subfield format
IPUI	Integrated Portable User Identity
I_{PX}	higher layer Information channel, encoded protected, minimum delay operation
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
Lc	a DLC layer C-plane protocol entity
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
MIC	Message Integrity Code
MM	Mobility Management
MSB	Most Significant Bit
MSC	Mobile Switching Centre
M_T	MAC control channel on A-tail field, or one message on such channel
NLMS	NWK Layer Message Structure
N_T	identities information channel or one message in such channel
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
Q_T	system information and multiframe marker (MAC logical channel)
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
$S_F(N)$	First of the two KSSs produced by the KSG
SIM	Subscriber Interface Module
SN	Send sequence Number
$S_P(N)$	Second of the two KSSs produced by the KSG
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
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

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.

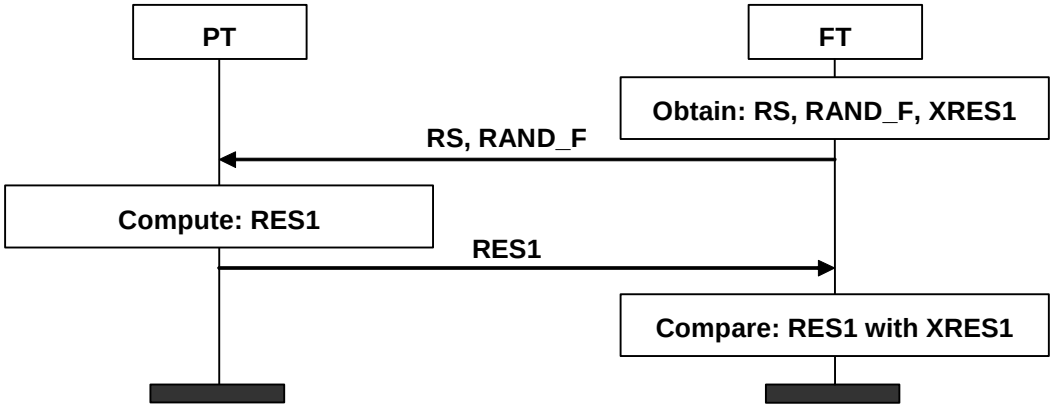


Figure 4.1: Authentication of PT (type 1 procedure)

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 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.

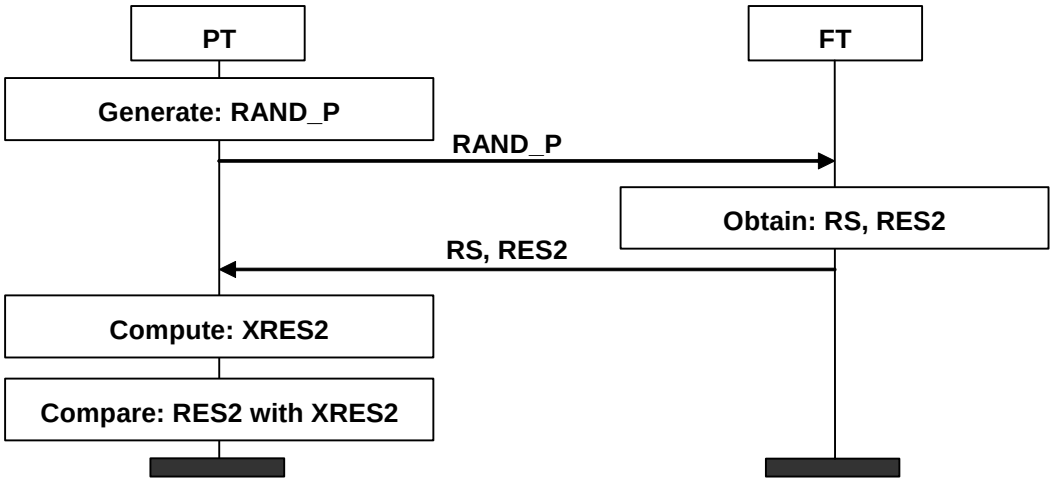


Figure 4.2: Authentication of FT (type 1 procedure)

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

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 type1 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 doted 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} the 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 RS128 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 PP 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).

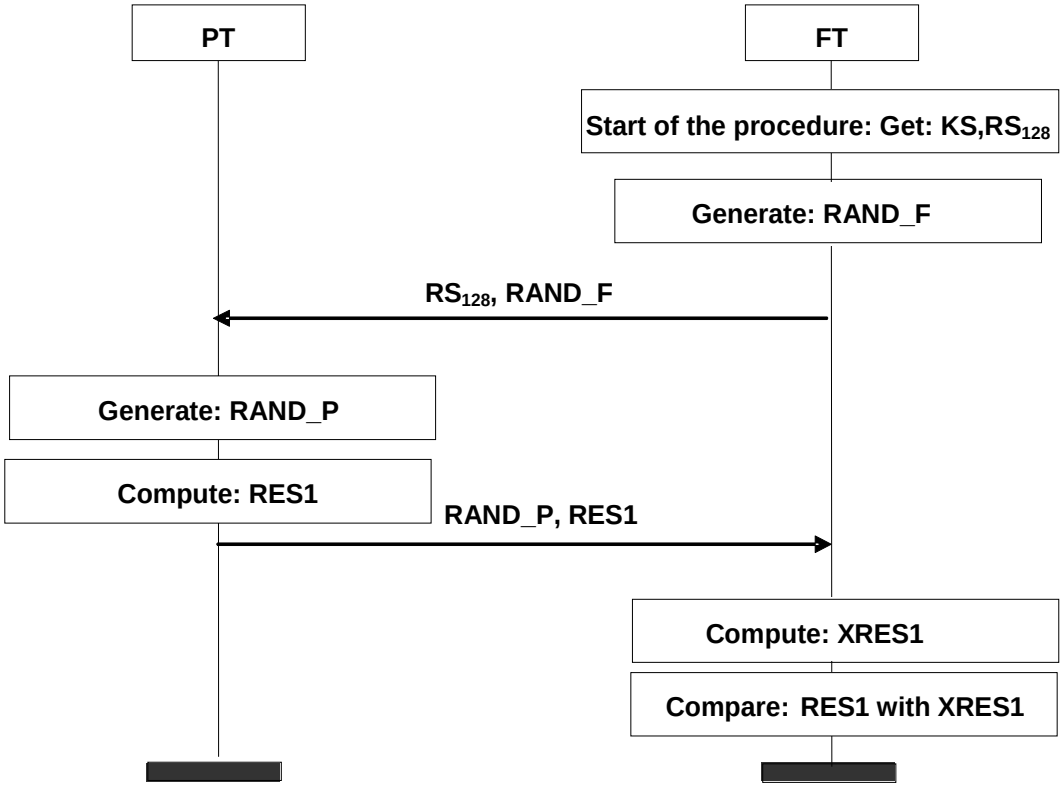


Figure 4.3: Authentication of PT (type 2 procedure)

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 doted 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.

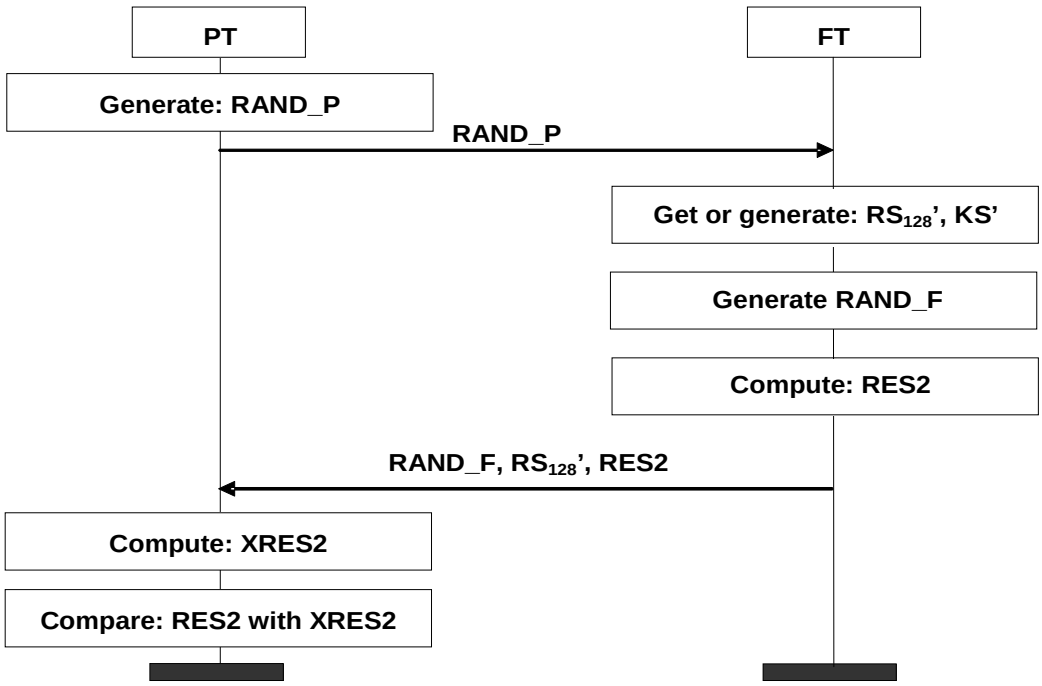


Figure 4.4: Authentication of FT

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} 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

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_{128}, 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_{128}:

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 build 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

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_{128}' 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_{128}, 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

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 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 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.

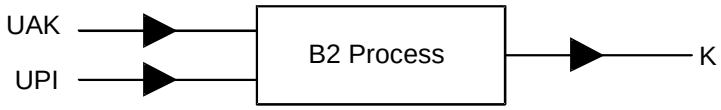


Figure 4.7: Derivation of K from UAK and UPI

The process B2 is defined as follows:

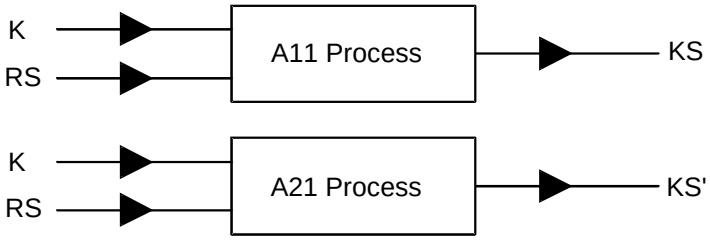
- K[i] = (UAK [i MODULO LEN_UAK] + UPI [i MODULO LEN_UPI]) MODULO 2,
  $0 \leq i \leq$ LEN_K - 1.

4.5.3 Authentication processes

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.

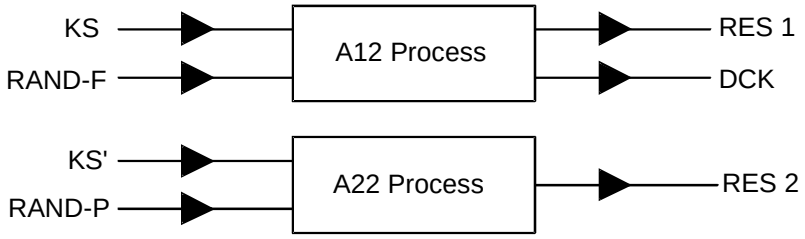


Figure 4.9: Derivation of RES1, DCK and RES2 for type 1 authentication

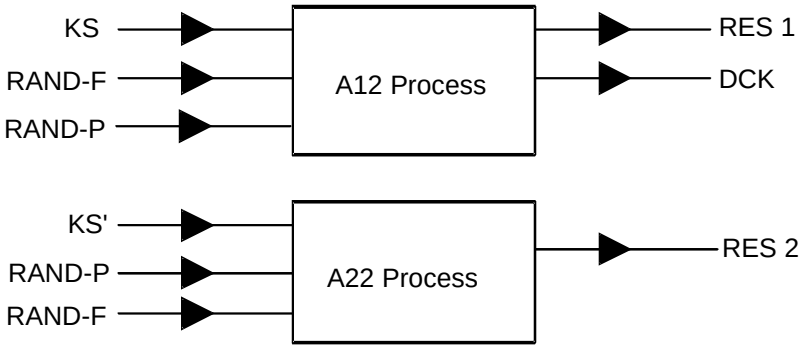


Figure 4.10: Derivation of RES1, DCK and RES2 for type 2 authentication

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 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.

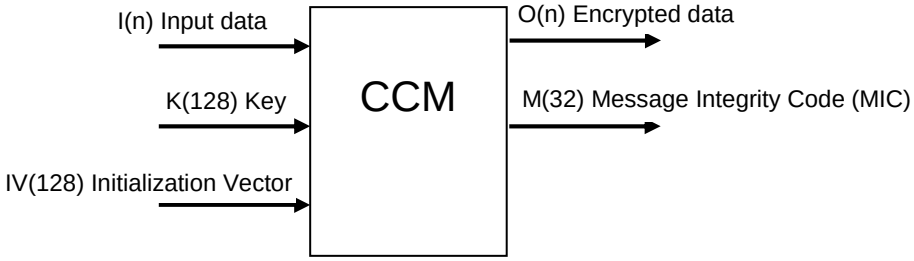


Figure 4.12: CCM Authenticated Encryption

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

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

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

4.6.1.1 Limitations related to capering 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

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

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.

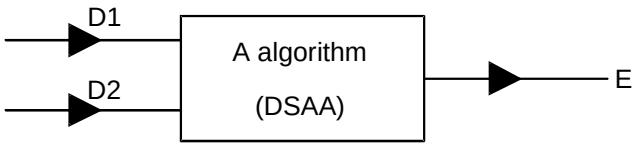


Figure 5.1: A algorithm, version DSAA (A-DSAA)

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, DSSA2-1 and the DSAA2-2. The operation of DSAA2-1 is defined in L.2.1 and the operation of DSAA2-2 is defined in 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.

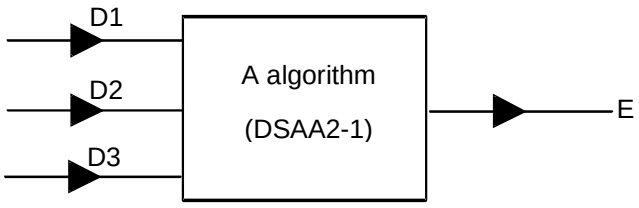


Figure 5.2: A algorithm, version DSAA2, variant DSAA2-1

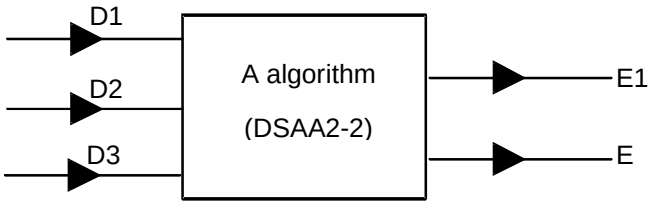


Figure 5.3: A algorithm, version DSAA2, variant DSAA2-2

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 MODULO LEN_K], $0 \leq i \leq$ LEN_D1 - 1; and
- D2 [i] = RS [i MODULO LEN_RS], $0 \leq i \leq$ LEN_D2 - 1.

If DSAA2 is used:

- D1 [i] = K [i MODULO LEN_K], $0 \leq i \leq$ LEN_D1 - 1;

- D2 [i] = RS1 [i MODULO LEN_RS1], $0 \leq i \leq LEN\_D2 - 1$; and
- D3 [i] = RS2 [i 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 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 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 MODULO LEN_K], $0 \leq i \leq LEN\_D1 - 1$; and
- D2 [i] = RS [i MODULO LEN_RS], $0 \leq i \leq LEN\_D2 - 1$.

If DSAA2 is used:

- D1 [i] = K [i MODULO LEN_K], $0 \leq i \leq LEN\_D1 - 1$;
- D2 [i] = RS1 [i MODULO LEN_RS1], $0 \leq i \leq LEN\_D2 - 1$; and
- D3 [i] = RS2 [i 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 MODULO LEN_E] + i) MODULO 2, $0 \leq i \leq LEN\_KS' - 1$.

For DSAA2 only:

- KS' [i] = E [i 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 MODULO LEN_K] + i) 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 MODULO LEN_KS], $0 \leq i \leq$ LEN_D1 - 1; and
- D2 [i] = RAND_F [i MODULO LEN_RAND_F], $0 \leq i \leq$ LEN_D2 - 1.

If DSAA2 is used:

- D1 [i] = K [i MODULO LEN_K], $0 \leq i \leq$ LEN_D1 - 1;
- D2 [i] = RAND_F [i MODULO LEN_RAND_F], $0 \leq i \leq$ LEN_D2 – 1; and
- D3 [i] = RAND_P [i 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 MODULO LEN_E], $0 \leq i \leq$ LEN_RES1 - 1; and
- DCK [i] = E [(i + LEN_RES1) MODULO LEN_E], $0 \leq i \leq$ LEN_DCK - 1.

If DSAA2 is used:

- RES1 [i] = E1 [i MODULO LEN_E1], $0 \leq i \leq$ LEN_RES1 - 1; and
- DCK [i] = E [i 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 MODULO LEN_KS'], $0 \leq i \leq$ LEN_D1 - 1; and
- D2 [i] = RAND_P [i MODULO LEN_RAND_P], $0 \leq i \leq$ LEN_D2 - 1.

If DSAA2 is used:

- D1 [i] = KS' [i MODULO LEN_KS'], $0 \leq i \leq$ LEN_D1 - 1; and
- D2 [i] = RAND_P [i MODULO LEN_RAND_P], $0 \leq i \leq$ LEN_D2 - 1;
- D3 [i] = RAND_F [i 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 MODULO LEN_E], $0 \leq i \leq$ LEN_RES2 - 1.

If DSAA2 is used:

- RES2 [i] = E1 [i 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 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 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

The purpose of this clause is to specify how an authentication key K is associated with the DECT identities defined in 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 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 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 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 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

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 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.

6.3 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 4.3.2 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.

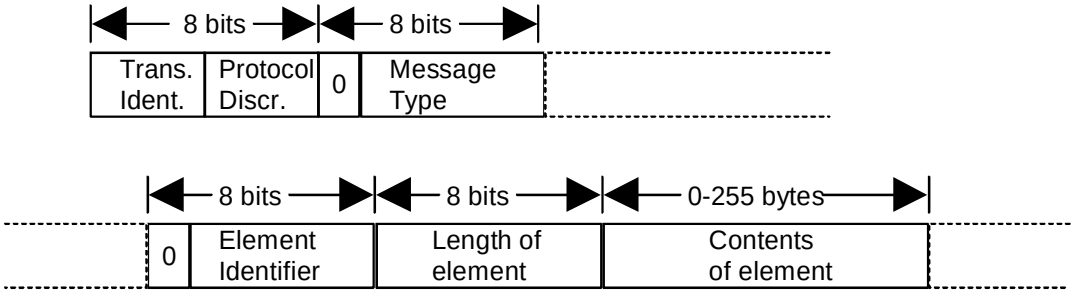


Figure 6.1: NWK Layer Message Structure (NLMS)

For more information on NWK layer procedures, messages or information elements see 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 EN 300 175-5 [5]. In the event of any conflict between figure 6.1 and 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 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) <<AUTH-TYPE>> <<RES>>(RES1)	<<RS>> (RS or RS_{128}) <<RAND>>(RAND_F) <<AUTH-TYPE>>
{AUTHENTICATION-REPLY}	01000001	<<RES>>(RES1) <<RAND>>(RAND_P)	<<RS>> (RS or RS_{128}) <<RES>>(RES2) <<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_{128} (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 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_{128}) 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 EN 300 175-5 [5]. In the event of any conflict between table 6.2 and 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 RS128 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 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 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 adaption (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

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.

6.4 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 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.

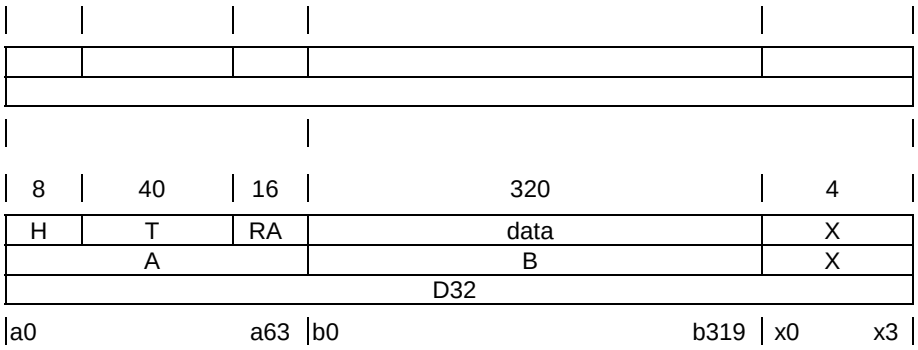


Figure 6.2: Field structures

At the lowest level, the D32-field is the interface to the physical channel. It contains a total of 388 bits, and consists of a 64-bit A-field and a 324-bit B-field.

The A-field consists of an 8-bit header H, a 40-bit tail T and a 16-bit redundancy field RA. The content of RA is a cyclic redundancy check computed on the contents of H and T.

The tail T can be one of five types: P_T, N_T, Q_T, M_T or C_T. The type of T depends on the information it carries, and is indicated by a code in H.

The B-field consists of a 320-bit data field and a 4-bit redundancy field X. The content of X is a cyclic redundancy check computed on the contents of the data field.

Two types of B-field are defined: E-type and U-type. A B-field of E-type is used to carry error protected MAC layer control data, C_F or G_F channel data. A B-field of U-type is used to carry I_N or I_P channel data.

In a B-field of U-type which is carrying I_N channel data all 320 data bits in the B-field are channel data. This is the unprotected format for the B-field. The B-field mapping is the U32a mapping described in EN 300 175-3 [3].

In a B-field of E-type, and in a B-field of U-type which is carrying I_P channel data, 64 bits of the 320 data bits are a redundancy code added at the MAC layer. This is the multisubfield protected format for the B-field. It is illustrated in figure 6.3. The 16-bit field RBj is a cyclic redundancy check computed on the data content of Bj. The multisubfield protected format is used with the B-field mappings U32b and E32 defined in EN 300 175-3 [3].

The E+U type mux B-field format is considered as E-type for all MAC encryption purposes.

I_{PQ} services use the U32c B-field mapping. This is the singlesubfield protected format consisting of a single data subfield of 304 bits protected by a single CRC field of 16 bit. It is illustrated in figure 6.4.

The E/U-Mux content of the B-field is identified by a code in the A-field header H. The protected format used for the I_P channel data (singlesubfield or multisubfield) is defined by the MAC service type of the connection (I_P or I_{PQ}) and negotiated during connection establishment or a later connection modification.

B0		B1		B2		B3		X
data	RB0	data	RB1	data	RB2	data	RB3	
64	16	64	16	64	16	64	16	4

Figure 6.3: Multisubfield protected B-field format

B0		X
data	RB0	
304	16	4

Figure 6.4: Singlesubfield protected B-field format

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 in 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 2*m* 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), ..., 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), ..., 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), ..., KSS(39) are XORed with the bits a_8, ..., 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 EN 300 175-3 [3].
- 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(359) are XORed with the bits b_0, ..., 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), ..., KSS(343) are XORed with the bits b_0, ..., b_{303} from B0, so that KSS(40) is XORed with b_0, etc. Bits KSS(344), ..., KSS(359) 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, ..., 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}, ..., 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}, ..., 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}, ..., b_{303} from B3, so that KSS(280) is XORed with b_{240}, etc.
- Bits KSS(344), ..., 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), ..., KSS(m), where $m = 320 \times r + 39$, with r the codec ratio, are XORed with the bits b_0, ..., 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.

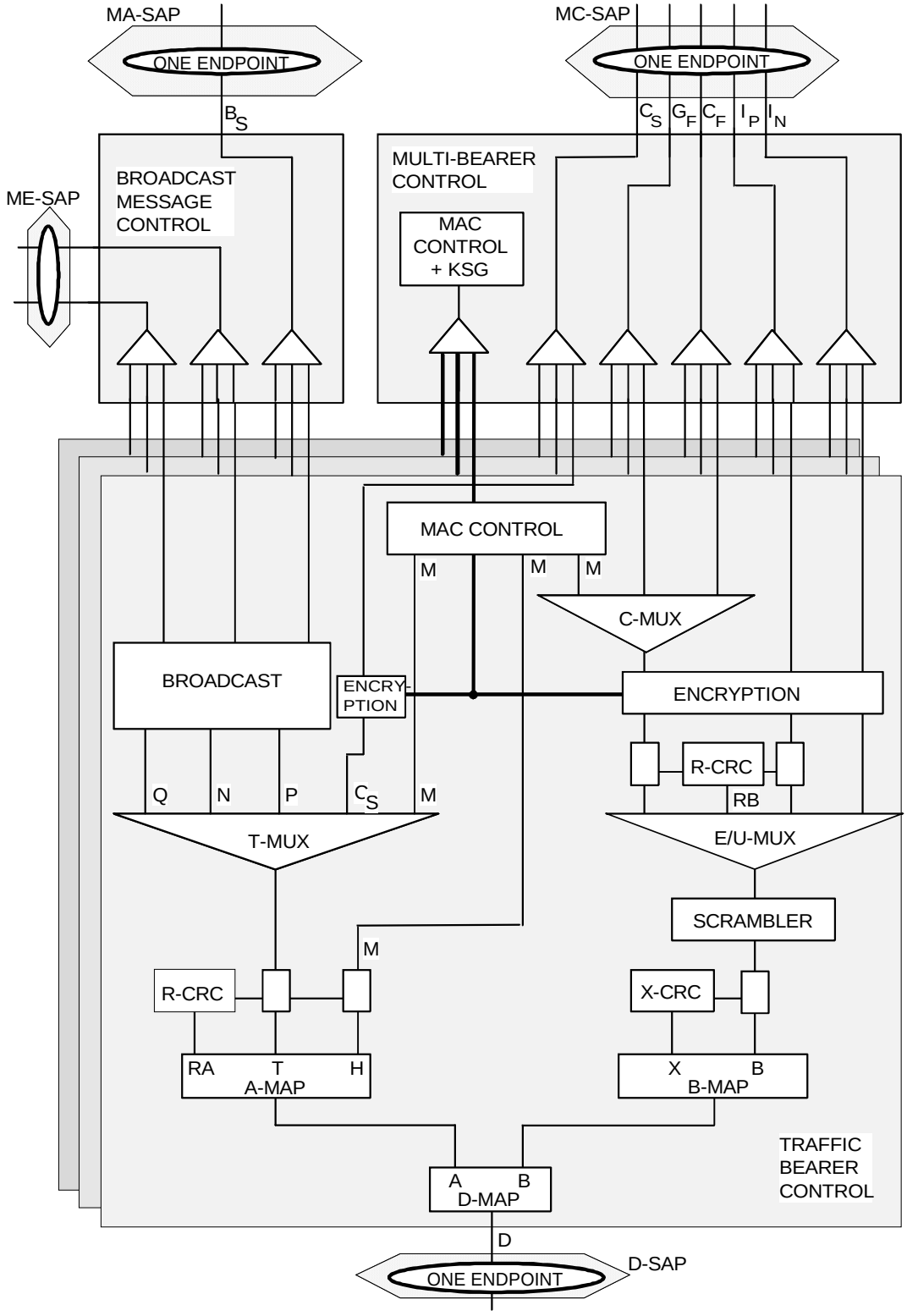


Figure 6.5: Placement of encryption

6.4.5 Initialization and synchronization of the encryption process

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 O.

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 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

The procedure followed by a PT to switch from clear to encrypt mode is described below, it is illustrated in figure 6.6 for a good link, and in figure 6.9 for a bad link.

PT procedure for switching from clear to encrypt 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 (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 event is reported with a MAC_ENC_EKS-cfm primitive. If the START.CONF is not observed, the connection is released and the DLC layer is informed using the MAC_DIS-ind primitive.

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 1: 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.

FT procedure for switching from clear to encrypt mode

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

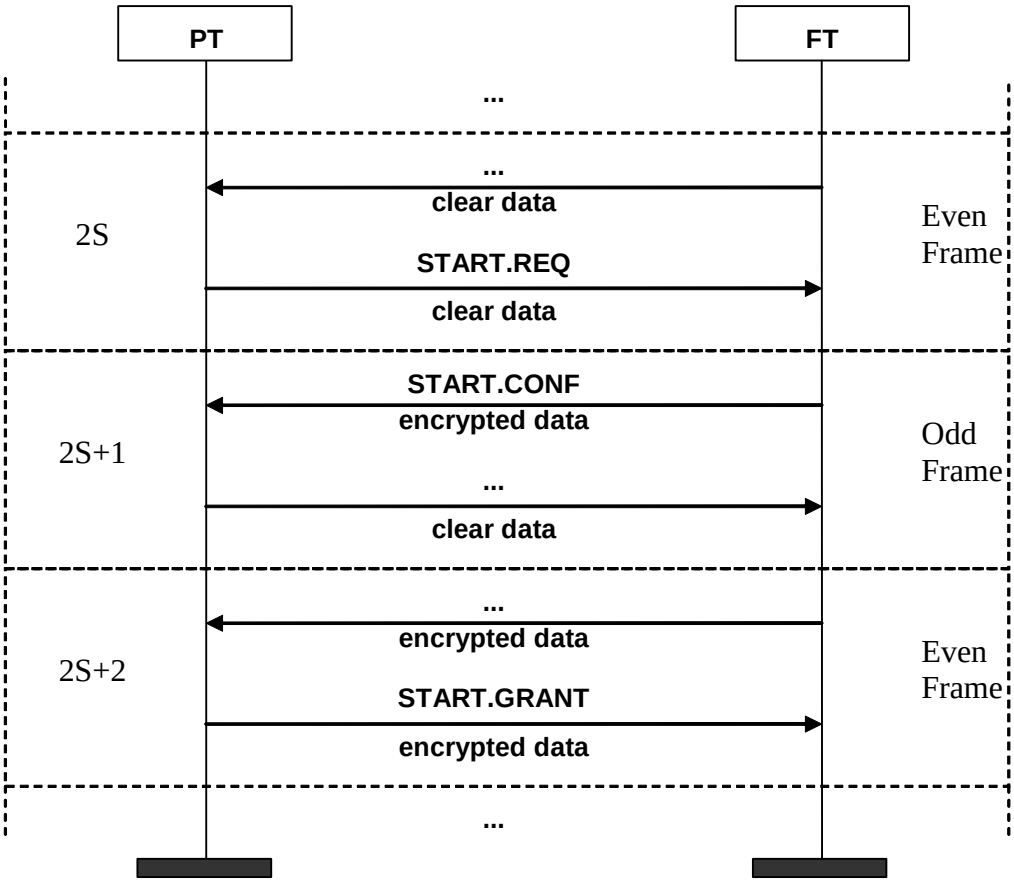


Figure 6.6: Encryption start - good link example

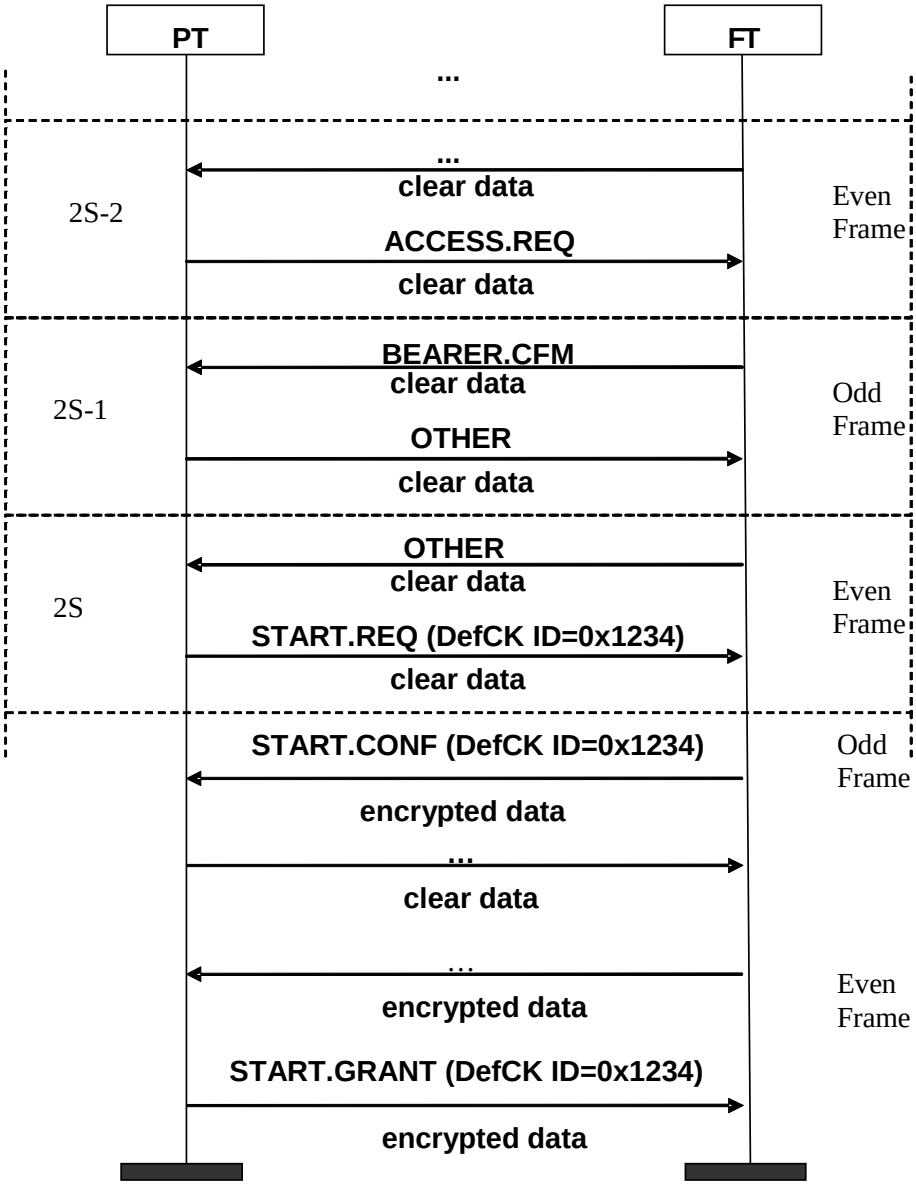


Figure 6.7: Encryption start immediately after bearer setup with default cipher key index - good link example

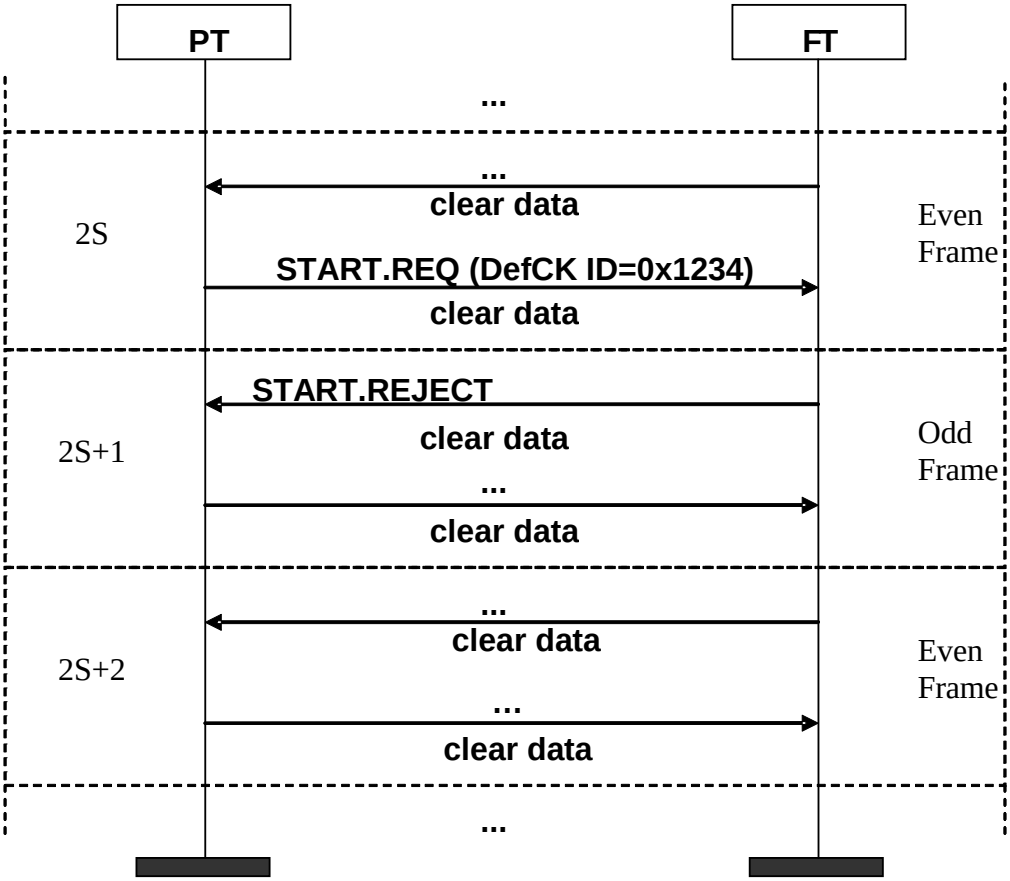


Figure 6.8: Encryption start with default cipher key index and rejected by FT-good link example

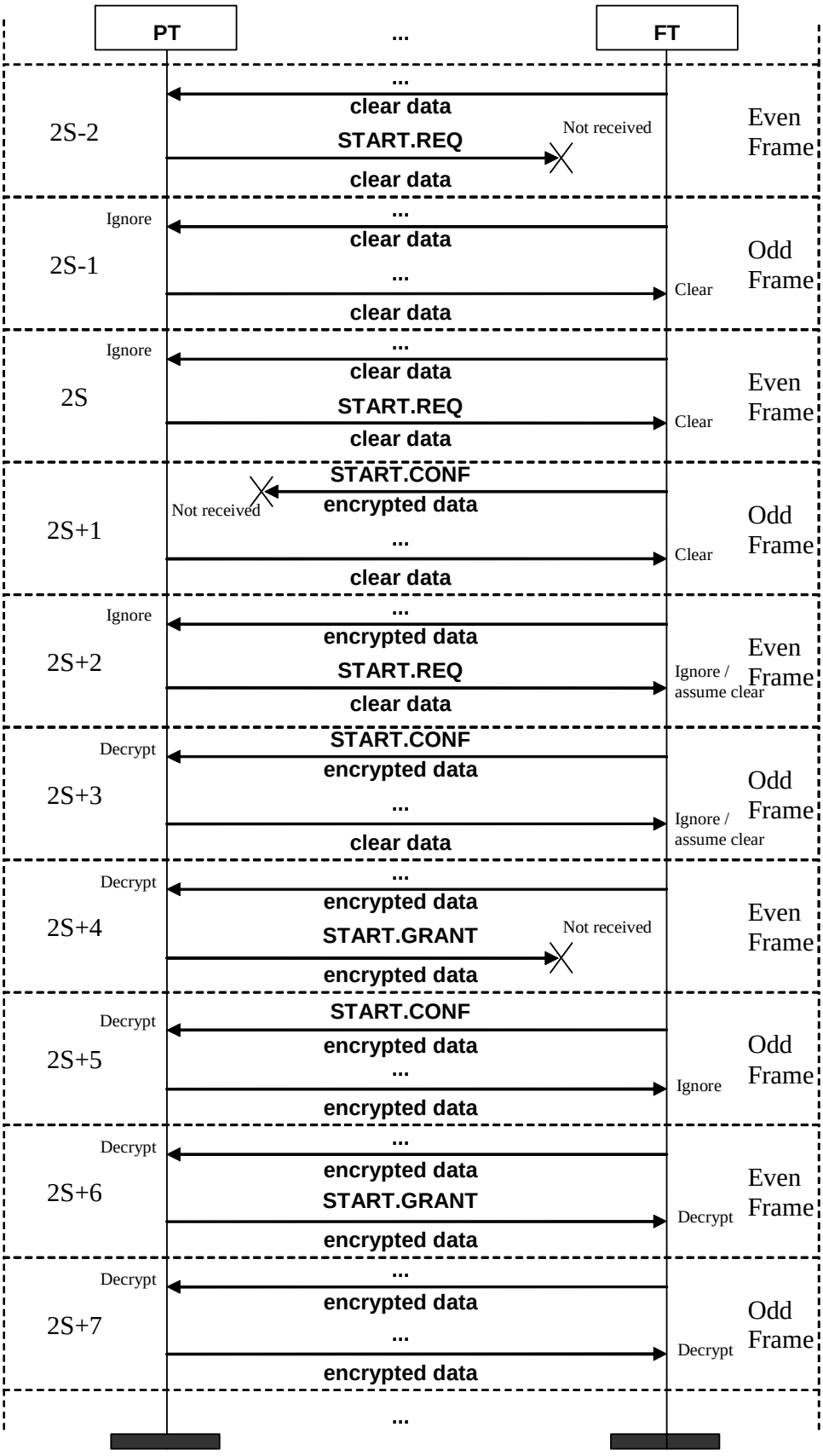


Figure 6.9: Encryption start - poor link example

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.

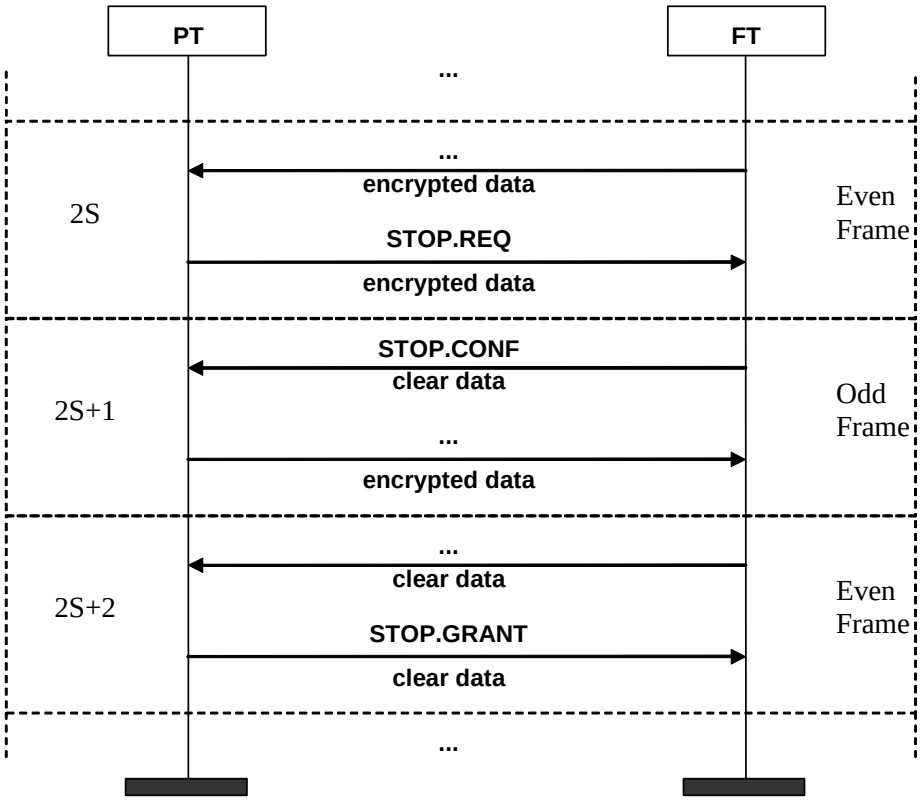


Figure 6.10: Encryption stop - good link example

6.4.6.5 Procedures for re-keying

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

It is started analogously to clause 6.4.6.3 with the difference that here 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.

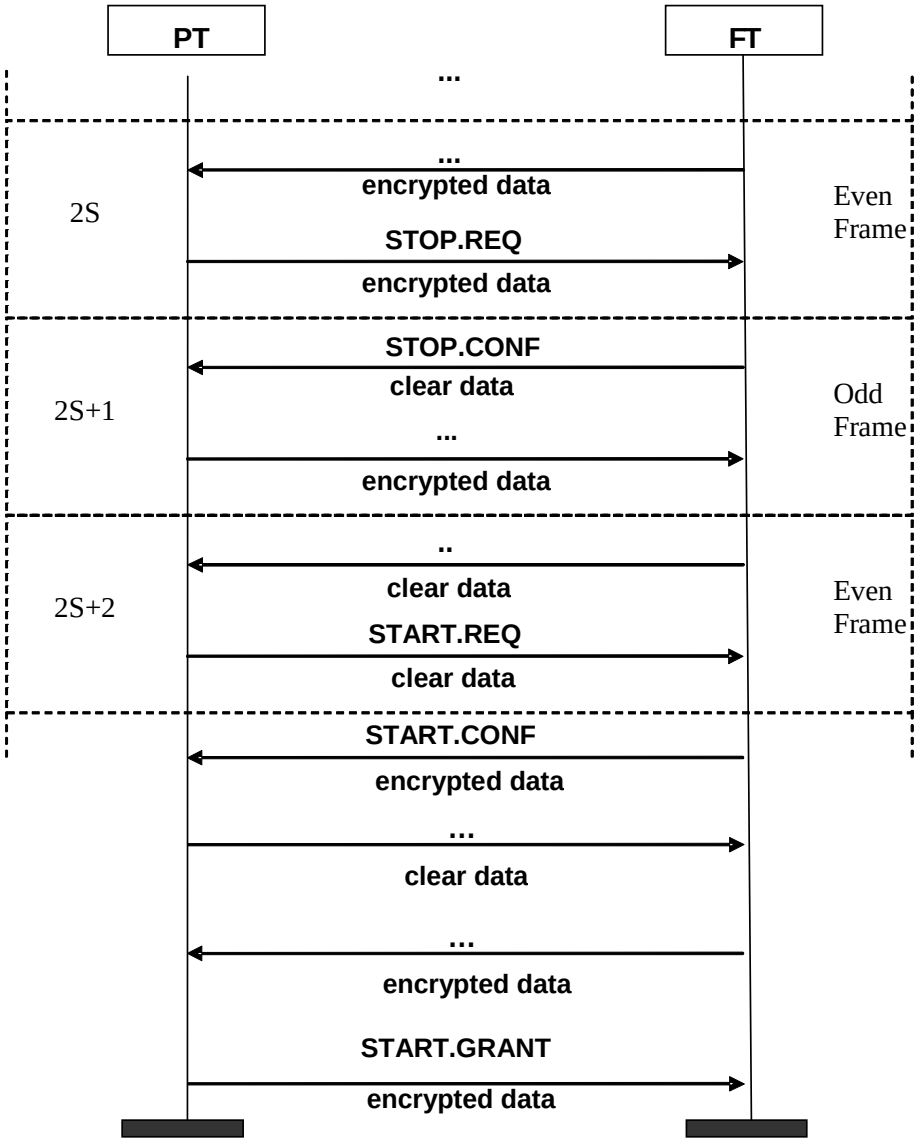


Figure 6.11: Encryption change - good link example

6.4.7 Handover of the encryption process

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 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.

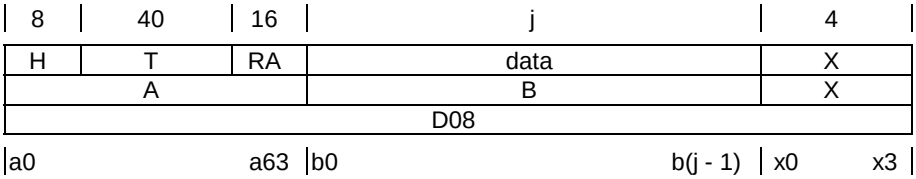


Figure 6.13: Half slot and long slot field structures

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 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 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 (note)	X
data	RB0					
64	16	j-80				4

NOTE: Padding only for long slot with j=672 (see 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) ..., KSS(39 + j).

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, ..., 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(39 + j) are XORed with the bits b_0, ..., 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), ..., KSS(103) are XORed with the bits b_0, ..., b_{63} from B0, so that KSS(40) is XORed with b_0, etc.
- Bits KSS(104), ..., KSS(119) are not used. If j > 80, bits KSS(120), ..., KSS(39 + j) are XORed with the bits b_{80}, ..., 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), ..., KSS(m), where $m = j \times r + 39$, with *r* the codec ratio, are XORed with the bits b_0, ..., 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.

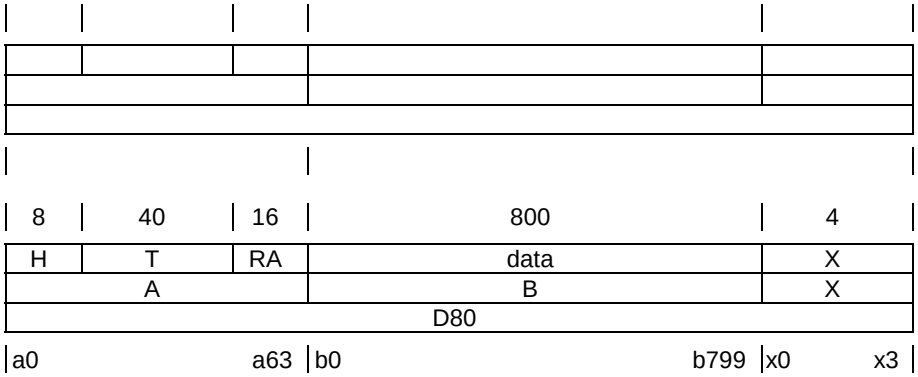


Figure 6.15: Double slot field structures

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 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 EN 300 175-3 [3]. The Bj-field carries 64 bits of control or I_p channel data and a 16-bit redundancy check R_{Bj} on this data.

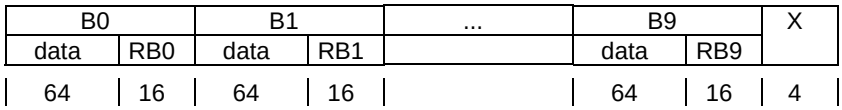


Figure 6.16: Double slot B-field multisubfield protected format

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 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.

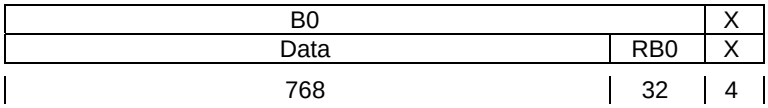


Figure 6.17: Double slot B-field singlesubfield protected format

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, ..., 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, ..., 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, ..., 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, ..., 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}, ..., 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}, ..., 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}, ..., 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}, ..., 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}, ..., 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}, ..., 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}, ..., 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}, ..., 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}, ..., 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, ..., 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 SF(N) is used by the transmitting side to encrypt the first D-field transmitted in a frame and the SP(N) to encrypt the second D-field transmitted. The receiving side consequently uses SF(N) to decrypt the first D-field received in the frame and SP(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 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	π/2-DBPSK	π/2-DBPSK	π/2-DBPSK
2	π/2-DBPSK	π/2-DBPSK	π/4-DQPSK
3	π/2-DBPSK	π/2-DBPSK	π/8-D8PSK
4a	π/2-DBPSK	π/4-DQPSK	π/4-DQPSK
4b	π/2-DBPSK	π/8-D8PSK	π/8-D8PSK
5	π/2-DBPSK	π/2-DBPSK	16-QAM
6	π/2-DBPSK	π/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 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

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

Configuration	length of Key Stream Segment KSSmax (bits)				
	double slot	full slot	half slot (j=80)	long slot (j=640)	long slot (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, ..., 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, ..., 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, ..., 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

	Double slot			Full slot			Half slot (j=80)		
Configuration	**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	$b_0 ... b_{799}$	KSS(i + 40)	320	$b_0 ... b_{319}$	KSS(i + 40)	80	$b_0 ... b_{79}$	KSS(i + 40)
1b	800	$b_0 ... b_{799}$	KSS(i + 40)	320	$b_0 ... b_{319}$	KSS(i + 40)	80	$b_0 ... b_{79}$	KSS(i + 40)
2	1 600	$b_0 ... b_{1\,599}$	KSS(i + 40)	640	$b_0 ... b_{639}$	KSS(i + 40)	160	$b_0 ... b_{159}$	KSS(i + 40)
3	2 400	$b_0 ... b_{2\,399}$	KSS(i + 40)	960	$b_0 ... b_{959}$	KSS(i + 40)	240	$b_0 ... b_{239}$	KSS(i + 40)
4a	1 600	$b_0 ... b_{1\,599}$	KSS(i + 80)	640	$b_0 ... b_{639}$	KSS(i + 80)	160	$b_0 ... b_{159}$	KSS(i + 80)
4b	2 400	$b_0 ... b_{2\,399}$	KSS(i + 120)	960	$b_0 ... b_{959}$	KSS(i + 120)	240	$b_0 ... b_{239}$	KSS(i + 120)
5	3 200	$b_0 ... b_{3\,199}$	KSS(i + 40)	1 280	$b_0 ... b_{1\,279}$	KSS(i + 40)	320	$b_0 ... b_{319}$	KSS(i + 40)
6	4 800	$b_0 ... b_{4\,799}$	KSS(i + 40)	1 920	$b_0 ... b_{1\,919}$	KSS(i + 40)	560	$b_0 ... b_{559}$	KSS(i + 40)

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

	Double slot			Full slot		
Configuration	**Data bits in the B-field**	**b_i**	**XORed with**	**Data bits in the B-field**	**b_i**	**XORed with**
1a	768	$b_0 ... b_{767}$	KSS(i + 40)	304	$b_0 ... b_{303}$	KSS(i + 40)
1b	768	$b_0 ... b_{767}$	KSS(i + 40)	304	$b_0 ... b_{303}$	KSS(i + 40)
2	1 568	$b_0 ... b_{1\,567}$	KSS(i + 40)	608	$b_0 ... b_{607}$	KSS(i + 40)
3	2 368	$b_0 ... b_{2\,367}$	KSS(i + 40)	928	$b_0 ... b_{927}$	KSS(i + 40)
4a	1 568	$b_0 ... b_{1\,567}$	KSS(i + 80)	608	$b_0 ... b_{607}$	KSS(i + 80)
4b	2 368	$b_0 ... b_{2\,367}$	KSS(i + 120)	928	$b_0 ... b_{927}$	KSS(i + 120)
5	3 168	$b_0 ... b_{3\,167}$	KSS(i + 40)	1 248	$b_0 ... b_{1\,247}$	KSS(i + 40)
6	4 768	$b_0 ... b_{4\,767}$	KSS(i + 40)	1 888	$b_0 ... b_{1\,887}$	KSS(i + 40)

	Long slot (j=640)			Long slot (j=672)		
Configuration	**Data bits in the B-field**	**b_i**	**XORed with**	**Data bits in the B-field**	**b_i**	**XORed with**
1a	608	$b_0 ... b_{607}$	KSS(i + 40)	640	$b_0 ... b_{639}$	KSS(i + 40)
1b	608	$b_0 ... b_{607}$	KSS(i + 40)	640	$b_0 ... b_{639}$	KSS(i + 40)
2	1 248	$b_0 ... b_{1\,247}$	KSS(i + 40)	1 312	$b_0 ... b_{1\,311}$	KSS(i + 40)
3	1 888	$b_0 ... b_{1\,887}$	KSS(i + 40)	1 984	$b_0 ... b_{1\,983}$	KSS(i + 40)
4a	1 248	$b_0 ... b_{1\,247}$	KSS(i + 80)	1 312	$b_0 ... b_{1\,311}$	KSS(i + 80)
4b	1 888	$b_0 ... b_{1\,887}$	KSS(i + 120)	1 984	$b_0 ... b_{1\,983}$	KSS(i + 120)
5	2 528	$b_0 ... b_{2\,527}$	KSS(i + 40)	2 556	$b_0 ... b_{2\,555}$	KSS(i + 40)
6	3 808	$b_0 ... b_{3\,807}$	KSS(i + 40)	4 000	$b_0 ... b_{3\,999}$	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.

The table should be read as follows:

- $b(x)_i$ indicates the i^{th} bit b of the x^{th} subfield (b_i of subfield x). The indicated bit is XORed according to the corresponding formula.

- KSS((80 × x) + i + 40) indicates the KSS bit corresponding to the i^{th} bit b of the x^{th} subfield.

Table 6.11: Encryption of B-subfields for multisubfield protected B-field modes

	Double slot			Full slot			Half slot (j=80)		
Configu-ration	**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 × 64	$(b(0)_0 ... b(0)_{63}) ... (b(9)_0 ... b(9)_{63})$	KSS((80 × x) + i + 40)	4 × 64	$(b(0)_0 ... b(0)_{63}) ... (b(3)_0 ... b(3)_{63})$	KSS((80 × x) + i + 40)	1 × 64	$(b(0)_0 ... b(0)_{63})$	KSS((80 × x) + i + 40)
1b	10 × 64	$(b(0)_0 ... b(0)_{63}) ... (b(9)_0 ... b(9)_{63})$	KSS((80 × x) + i + 40)	4 × 64	$(b(0)_0 ... b(0)_{63}) .. (b(3)_0 ... b(3)_{63})$	KSS((80 × x) + i + 40)	1 × 64	$(b(0)_0 ... b(0)_{63})$	KSS((80 × x) + i + 40)
2	20 × 64	$(b(0)_0 ... b(0)_{63}) ... (b(19)_0 ... b(19)_{63})$	KSS((80 × x) + i + 40)	8 × 64	$(b(0)_0 ... b(0)_{63}) ... (b(7)_0 ... b(7)_{63})$	KSS((80 × x) + i + 40)	2 × 64	$(b(0)_0 ... b(0)_{63}), (b(1)_0 ... b(1)_{63})$	KSS((80 × x) + i + 40)
3	30 × 64	$(b(0)_0 ... b(0)_{63}) ... (b(29)_0 ... b(29)_{63})$	KSS((80 × x) + i + 40)	12 × 64	$(b(0)_0 ... b(0)_{63}) ... (b(11)_0 ... b(11)_{63})$	KSS((80 × x) + i + 40)	3 × 64	$(b(0)_0 ... b(0)_{63}) ... (b(2)_0 ... b(2)_{63})$	KSS((80 × x) + i + 40)
4a	20 × 64	$(b(0)_0 ... b(0)_{63}) ... (b(19)_0 ... b(19)_{63})$	KSS((80 × x) + i + 80)	8 × 64	$(b(0)_0 ... b(0)_{63}) ... (b(7)_0 ... b(7)_{63})$	KSS((80 × x) + i + 80)	2 × 64	$(b(0)_0 ... b(0)_{63}), ... (b(1)_0 ... b(1)_{63})$	KSS((80 × x) + i + 80)
4b	30 × 64	$(b(0)_0 ... b(0)_{63}) ... (b(29)_0 ... b(29)_{63})$	KSS((80 × x) + i + 120)	12 × 64	$(b(0)_0 ... b(0)_{63}) ... (b(11)_0 ... b(11)_{63})$	KSS((80 × x) + i + 120)	3 × 64	$(b(0)_0 ... b(0)_{63}) ... (b(2)_0 ... b(2)_{63})$	KSS((80 × x) + i + 120)
5	40 × 64	$(b(0)_0 ... b(0)_{63}) ... (b(39)_0 ... b(39)_{63})$	KSS((80 × x) + i + 40)	16 × 64	$(b(0)_0 ... b(0)_{63}) ... (b(15)_0 ... b(15)_{63})$	KSS((80 × x) + i + 40)	4 × 64	$(b(0)_0 ... b(0)_{63}) ... (b(3)_0 ... b(3)_{63})$	KSS((80 × x) + i + 40)
6	60 × 64	$(b(0)_0 ... b(0)_{63}) ... (b(59)_0 ... b(59)_{63})$	KSS((80 × x) + i + 40)	24 × 64	$(b(0)_0 ... b(0)_{63}) ... (b(23)_0 ... b(23)_{63})$	KSS((80 × x) + i + 40)	6 × 64	$(b(0)_0 ... b(0)_{63}) ... (b(5)_0 ... b(5)_{63})$	KSS((80 × x) + i + 40)

	Long slot (j=640/672)		
Configu-ration	**Data bits in the B-field**	**$b(x)_i$**	**XORed with**
1a	8×64	$(b(0)_0 \ldots b(0)_{63}) \ldots (b(7)_0 \ldots b(7)_{63})$	KSS((80 × x) + i + 40)
1b	8×64	$(b(0)_0 \ldots b(0)_{63}) \ldots (b(7)_0 \ldots b(7)_{63})$	KSS((80 × x) + i + 40)
2	16×64	$(b(0)_0 \ldots b(0)_{63}) \ldots (b(15)_0 \ldots b(15)_{63})$	KSS((80 × x) + i + 40)
3	24×64	$(b(0)_0 \ldots b(0)_{63}) \ldots (b(23)_0 \ldots b(23)_{63})$	KSS((80 × x) + i + 40)
4a	16×64	$(b(0)_0 \ldots b(0)_{63}) \ldots (b(15)_0 \ldots b(15)_{63})$	KSS((80 × x) + i + 80)
4b	24×64	$(b(0)_0 \ldots b(0)_{63}) \ldots (b(23)_0 \ldots b(23)_{63})$	KSS((80 × x) + i + 120)
5	32×64	$(b(0)_0 \ldots b(0)_{63}) \ldots (b(31)_0 \ldots b(31)_{63})$	KSS((80 × x) + i + 40)
6	48×64	$(b(0)_0 \ldots b(0)_{63}) \ldots (b(47)_0 \ldots b(47)_{63})$	KSS((80 × 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 KSSmax bits labelled KSS(0)...KSS(KSSmax - 1), where KSSmax depends of the slot type and modulation scheme. The value of KSSmax 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 = KSSmax – 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, ..., 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, ..., 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, ..., 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 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>> (<<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 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>> (<<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>> <<CIPHER-INFO>>
{ACCESS-RIGHTS-ACCEPT}	01000101	<<AUTH-TYPE>> <<CIPHER-INFO>>
{ACCESS-RIGHTS-REJECT}	01001011	<<REJECT REASON>> (<<AUTH-TYPE>>, <<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 EN 300 175-5 [5].

6.5.2 Authentication protocols

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 4returns 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 IPUI (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

Authentication keys may be of three types, UAK, AC or UAK/UPI. 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 UPI 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 UPI 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 UPI may be associated. Such a UPI 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/UPI) 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/UPI 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.

It is 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:

UAK[i] = KS'[i MODULO LEN_KS'], $0 \leq i \leq$ 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>> <<RAND>> (RAND_F) <<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 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

The DECT CCM Authenticated Encryption is intended to operate at DLC layer. CCM provides both packet authentication and encryption, and is based on RFC 3610 [11]. The DECT implementation of CCM uses the Advanced Encryption Standard (AES) [10] as 128 bit block cipher (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.

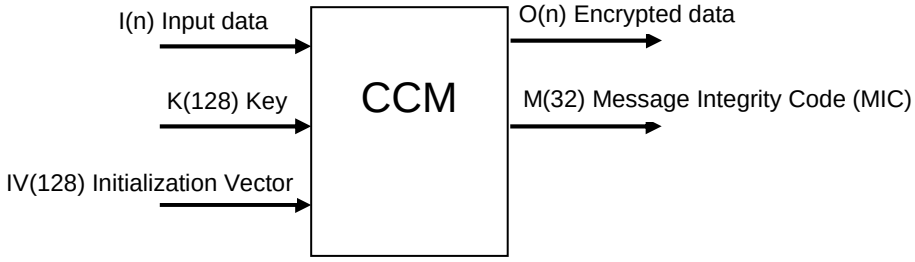


Figure 6.18: CCM security process overview

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 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 6.4.2.3).

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.

6.6.2.3 CCM Initialization Vector

In the following clauses, octets and bits will be numbered starting with 0, as is the practice in 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 (8 bits)	N_RESV. (8 bits)	Packet number (48bits)	DECT identities (32 bits)	dir/LCN (8 bits)	Fixed (8 bits)	SDU length (16 bits)

Octet	CCM IV fields	Size (bytes)	Description
0	Flags	1	CCM flag field (see clause 6.4.3.2.1). 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 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.			

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 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) is for further study.

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) is for further study.

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.

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 (see note 3)	A	A
UPI/UAK	A (see note 2)	A (see note 2)	N
SCK	N	A	A
NOTE 1: A = Applicable. 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

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_{128} and RS_{128}') 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

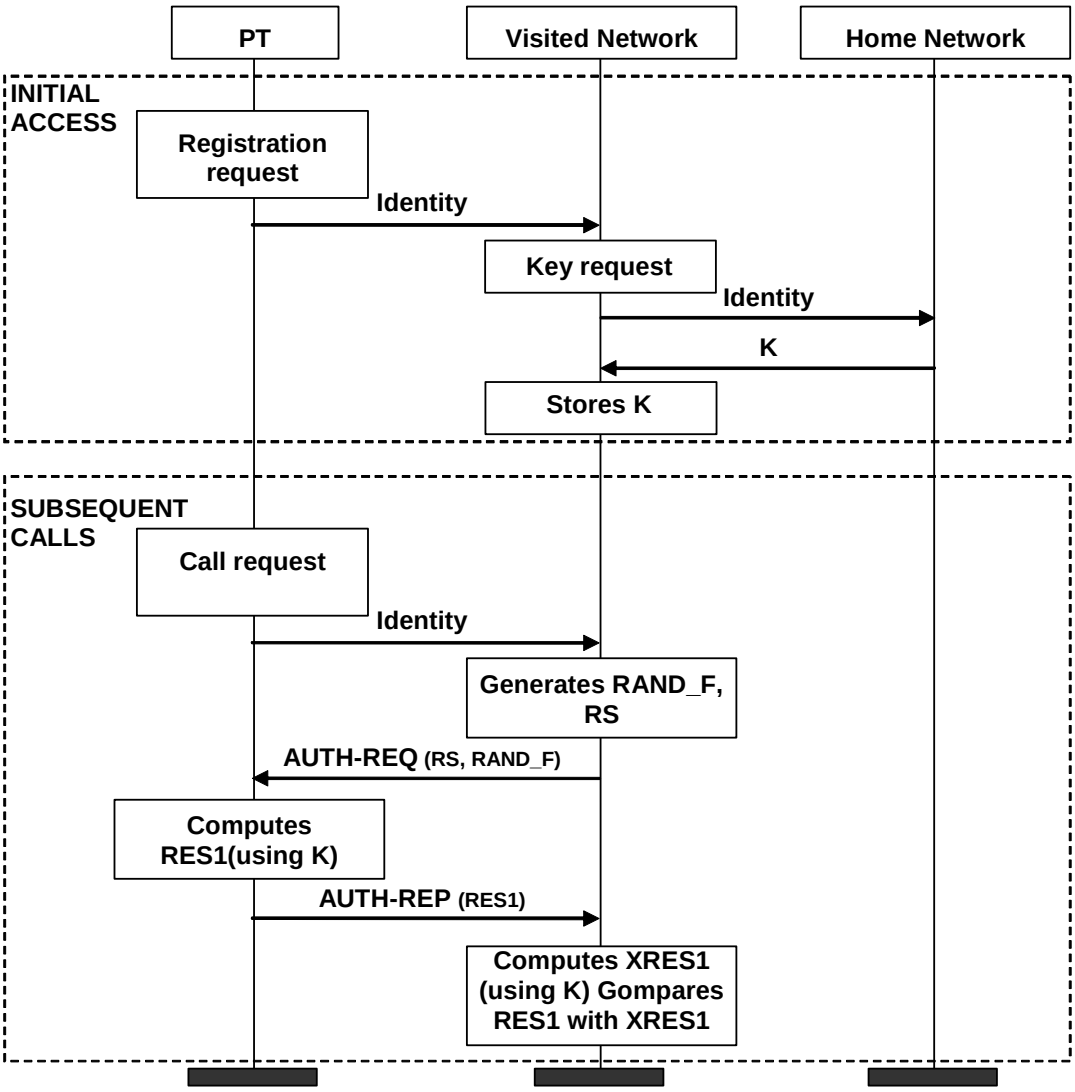
- The authentication of a PT (see clause 4.3.1) can also be realized by sending the RES1 obtained from the PT to a home database for checking.
- The security specification does not specify mechanisms for the transfer of RES1.
- In authentication type 2 (DSAA2), the RAND_P parameter would have to be sent 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

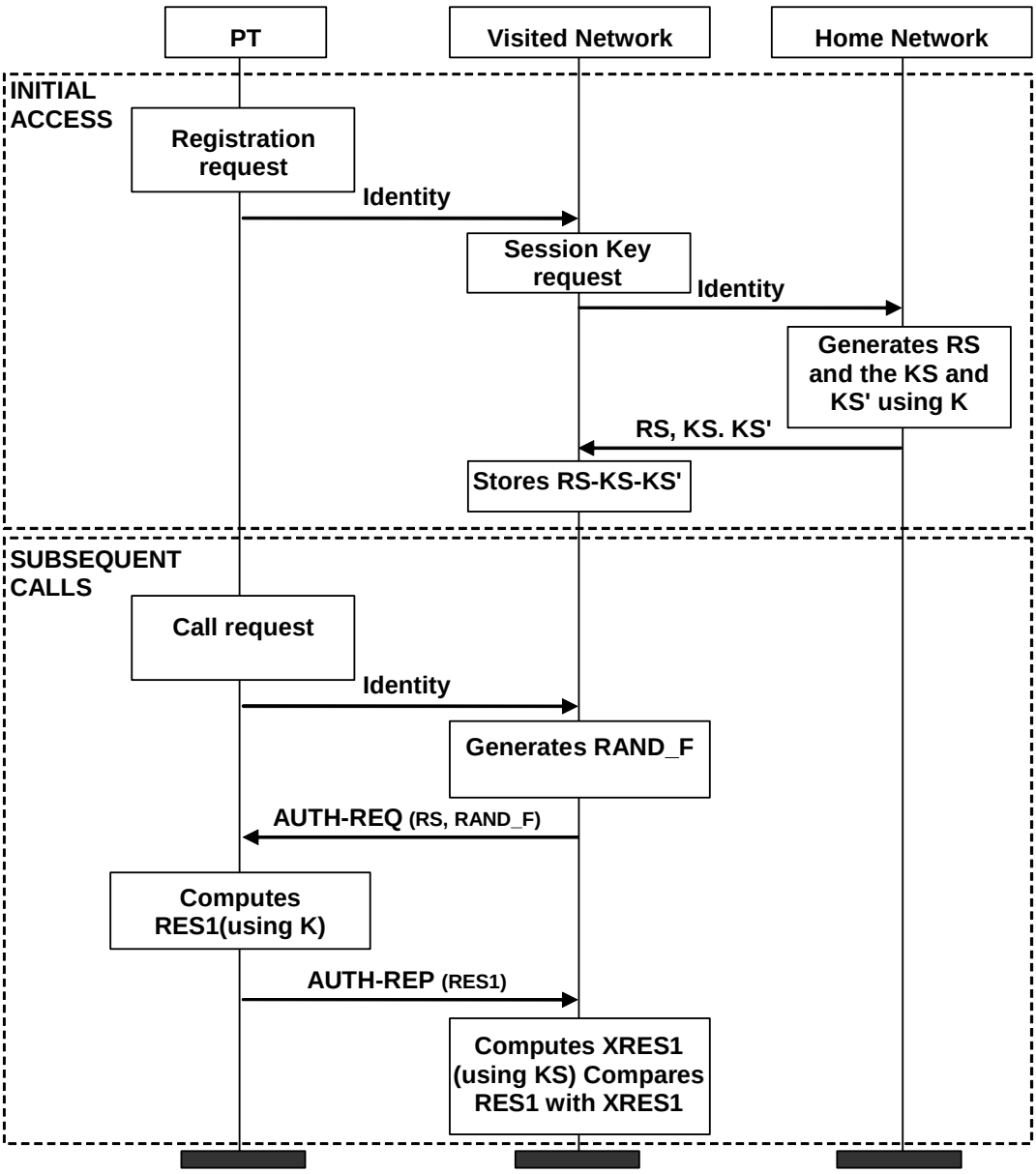
- The SCK or the DCK shall be available at the point in the FT where the encryption is performed.
- A DCK is automatically changed after each instance of authentication of the PT. An SCK is not changed automatically. If the SCK is used for applications needing a high level of security, then it should 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)

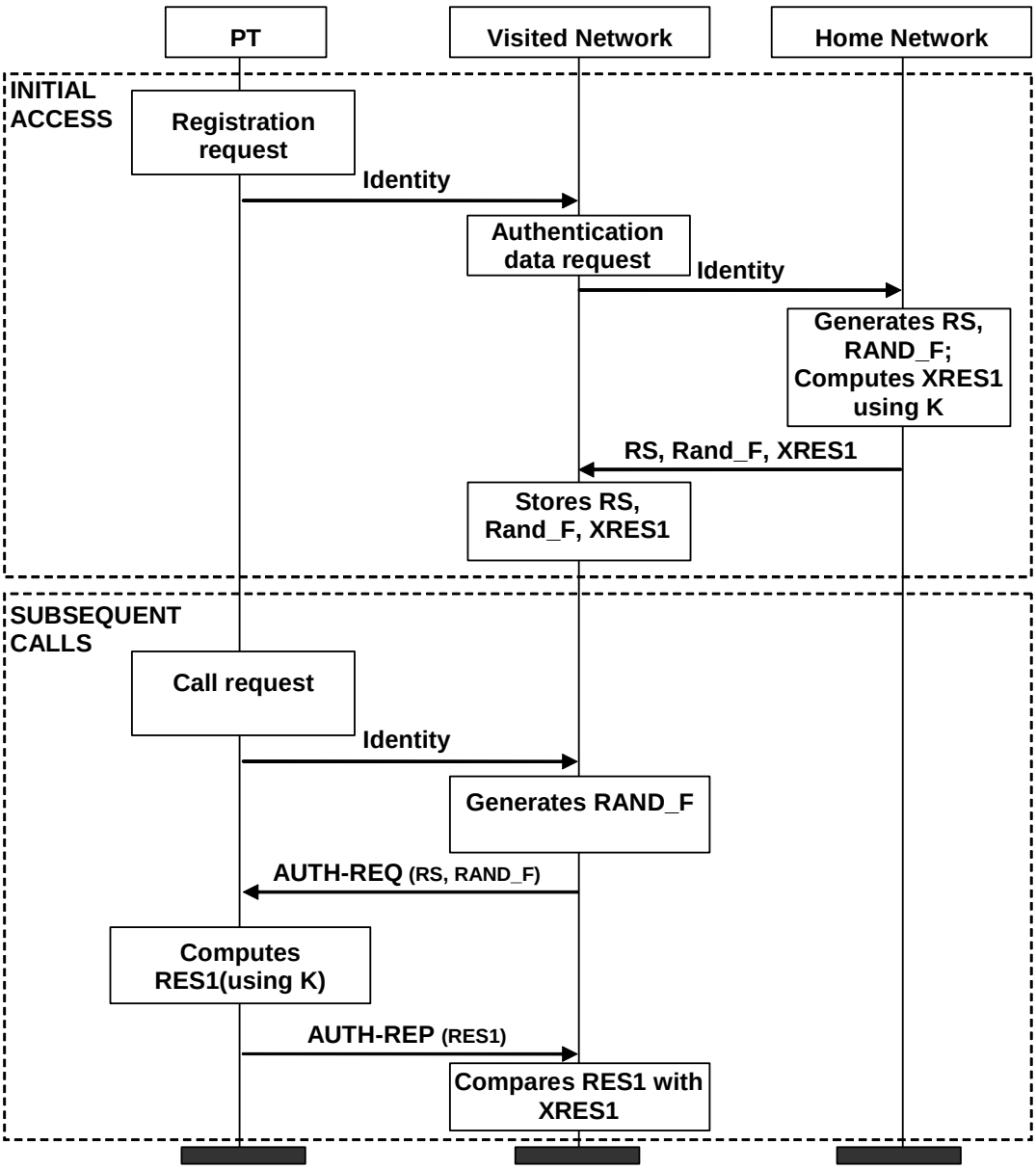


NOTE 1: Only authentication of the PT is depicted. The cases of authentication of the FT and mutual authentication are essentially the same. Generation of the DCK by the visited network is straightforward and is also not shown in the figure.

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.

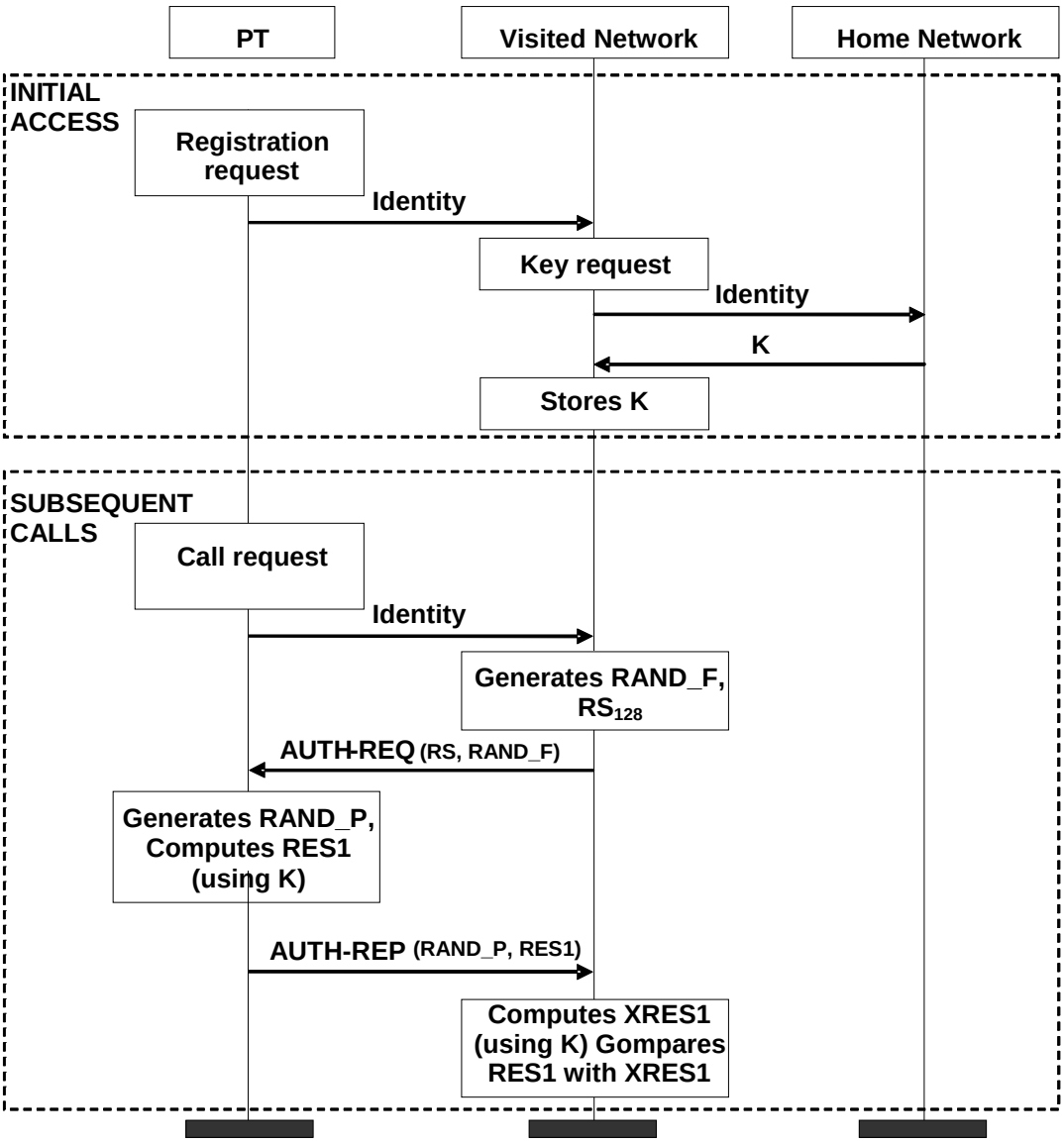
Figure 7.2: Roaming using session keys (authentication type 1)



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.

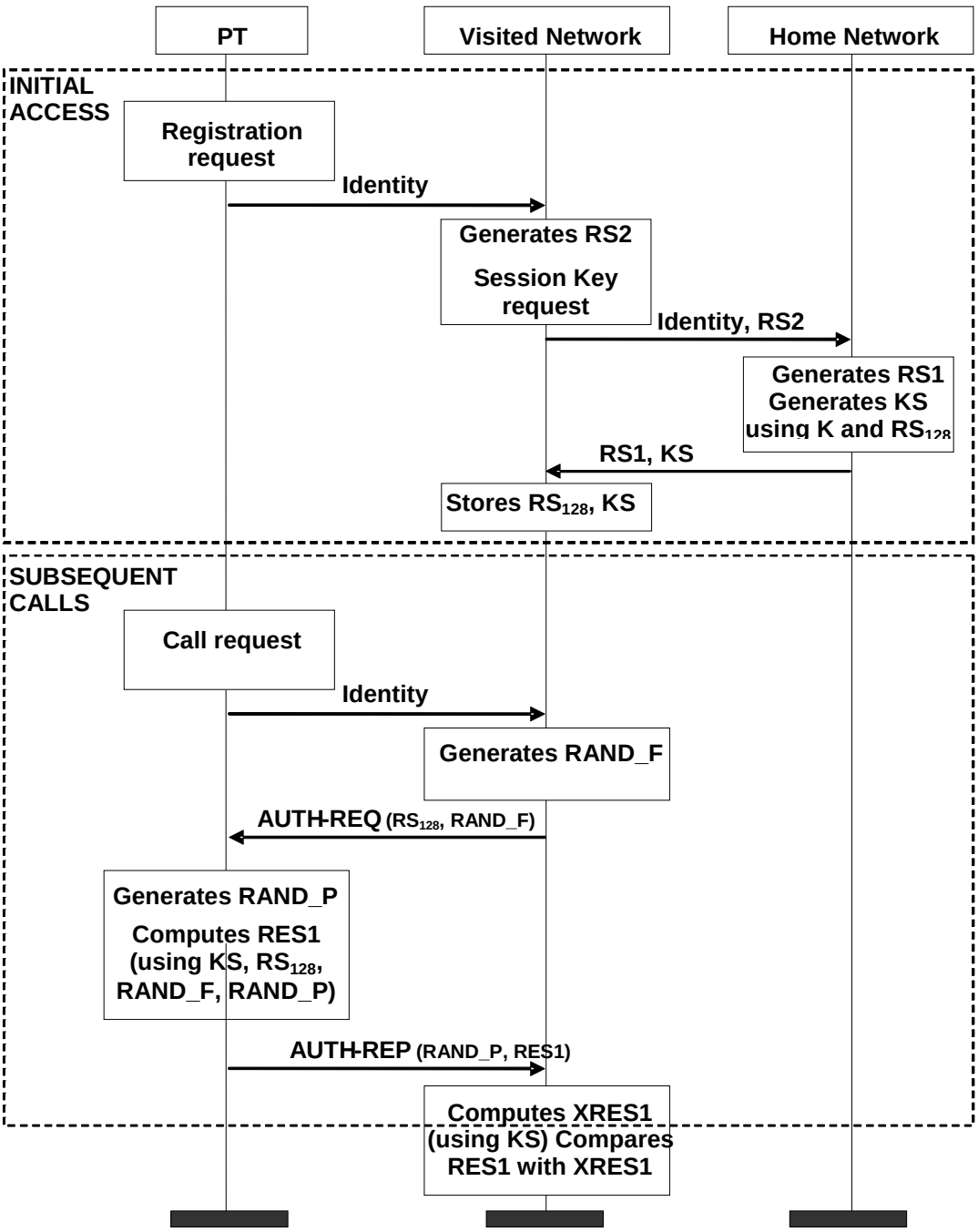
Figure 7.3: Roaming using precalculated sets (authentication type 1)

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 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 *italic*s 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

	Environment		
Threat	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

	Environment	
Level	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-routeing calls) might need protection.

Table A.5: Threat: Obtaining user information

	Environment		
Level	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 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 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

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

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

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 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 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 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;
Y*: algorithm/process required if authentication of the PT is used for the cipher key derivation;
N: not applicable;
A1: represents the combined A11 and A12 processes;
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 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;
Y*: algorithm/process required if authentication of the PT is used for the cipherkey derivation;
N: not applicable;
A1: represents the combined A11 and A12 processes;
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 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 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 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 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 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 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 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 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 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 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 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 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 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 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

Fixed public access service example Profile P1	
DECT Security Service	**Profile**
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.

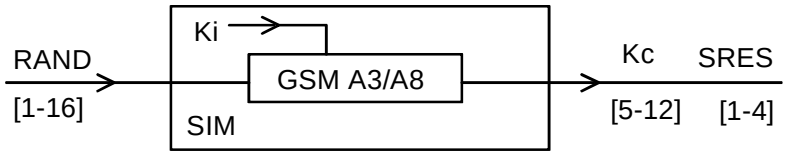


Figure G.1: Security functionality of a SIM

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 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.

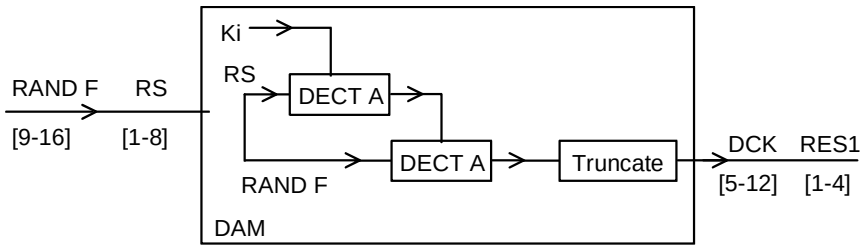


Figure G.2: Functionality of a DAM for PP authentication

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 (informative): Void

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} <<Portable-Identity>> <<AUTH-TYPE>>	0x5 0x44 0x5 0x7 0x80 0xa8 0x0 0x4 0xff 0xff 0xff 0xa 0x3 0x1 0x48 0x0
PP <<<--- FP	{KEY-ALLOCATE} <<Allocation-type>> <<RAND>> <<RS>>	0x5 0x42 0xb 0x2 0x1 0x88 0xc 0x8 0xd0 0x1f 0xf9 0xe2 0x11 0x68 0x4 0x21 0xe 0x8 0x0 0x0 0x0 0x0 0x0 0x0 0x0 0x0
PP --->>> FP	{AUTH-REQUEST} <<AUTH-TYPE>> <<RAND>> <<RES>>	0x85 0x40 0xa 0x3 0x1 0x48 0x0 0xc 0x8 0xa0 0x3f 0xf2 0xc5 0x23 0xd0 0x8 0x42 0xd 0x4 0xf2 0x8c 0xf9 0x11
PP <<<--- FP	{AUTH-REPLY} <<RES>>	0x5 0x41 0xd 0x4 0xee 0xba 0x4d 0xb9
PP <<<--- FP	{ACCESS-RIGHTS-ACCEPT} <<Portable-Identity>> <<Fixed-Identity>> <<Location-area>> <<AUTH-TYPE>>	0x85 0x45 0x5 0x5 0x80 0x94 0x12 0x61 0x20 0x6 0x6 0xa0 0xa0 0x10 0x0 0x90 0x34 0x7 0x1 0x5f 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} <<AUTH-TYPE>> <<RAND>> <<RS>>	0x5 0x40 0xa 0x3 0x1 0x18 0x18 0xc 0x8 0x9b 0x11 0x4f 0x25 0x54 0x51 0x88 0x59 0xe 0x8 0x0 0x0 0x0 0x0 0x0 0x0 0x0 0x0
PP --->>> FP	{AUTH-REPLY} <<RES>>	0x85 0x41 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. 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 $t \leq 128$.

We call DSAA2-1 the secret session key derivation algorithm and DSAA2-2 the tag generation and cipher key derivation algorithm.

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 .. K_{127} and the 128-bit plaintext block W_0 .. W_{127}. Let the resulting ciphertext block be $C1_0$ .. $C1_{127}$.

The 128 bits of this AES ciphertext block are used as secret session key E, namely E[0] .. E[127] = $C1_0$ .. $C1_{127}$.

L.2.2 DSAA2-2

NOTE 1: The algorithm starts the same way as DSAA2-1.

Let the 128-bit sequence K_0 .. 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}.

NOTE 2: From this point on it differs from DSAA2-1.

The 32-bit tag E1 is extracted from W as follows:

- E1[0] .. E1[31] = W_0 .. W_{31}

Perform a second AES block cipher encryption. Use AES with the 128-bit key K_0 .. K_{127} and the 128-bit plaintext block W_0 W_1.. W_{125} W_{126} ($W_{127} \oplus 1$), i.e. just bit 127 is inverted. Let the resulting ciphertext block be $C2_0$ .. $C2_{127}$.

The first t bits of this AES ciphertext block are used as the cipher key E2, i.e. E2[0] .. E2[t-1] = $C2_0$ .. $C2_{t-1}$.

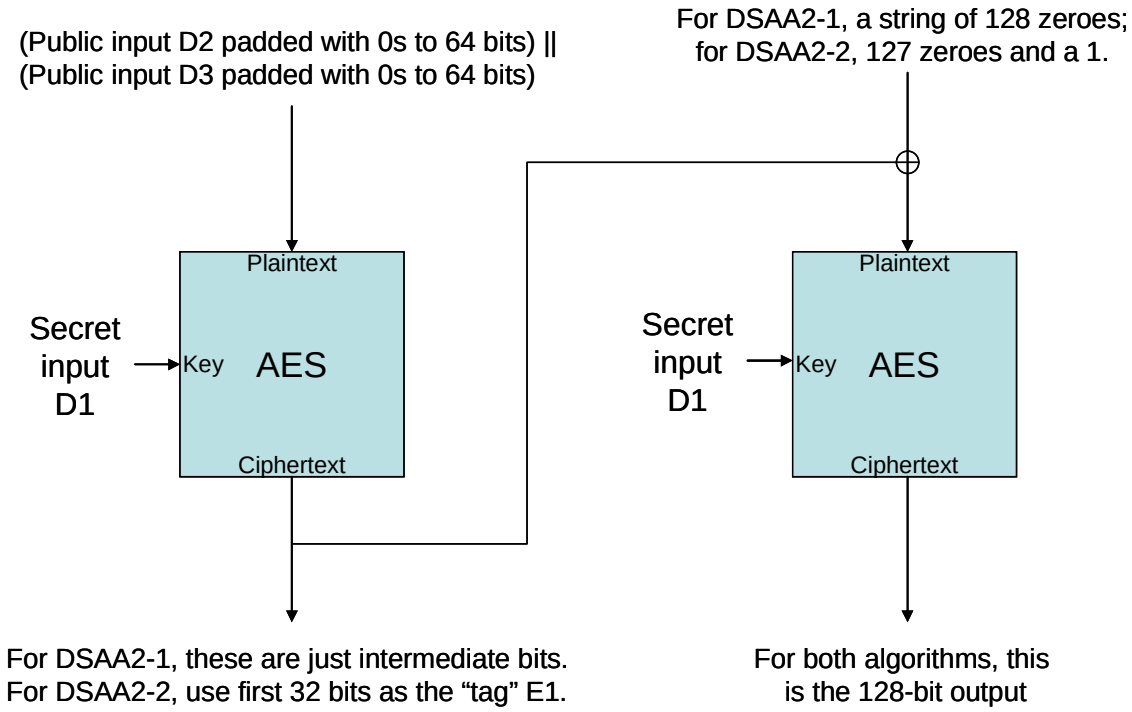


Figure L.1: The DSAA2-1 and DSAA2-2 algorithms

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 010111111 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 .. K_{127}	42025EE3 39743AF6 47B57780 25E9B66D
AES Plaintext P_0 .. P_{127}	A0F3624E 949640A0 17DA4751 F5B2B180
AES Ciphertext W_0 .. 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 .. K_{127}	5DDB5285 D9072931 833B3556 7776898A
AES Plaintext P_0 .. P_{127}	8493A329 2770FA99 B950D6D2 E77E641D
AES Ciphertext W_0 .. 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 .. K_{127}	63340A1B D9BAF80A 25D20044 C354A708
AES Plaintext P_0 .. P_{127}	86167C7D 8D0EA8F4 74BBC76E C7C0F65E
AES Ciphertext W_0 .. 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 .. K_{127}	BCD32DB0 21D5D009 93428051 F1CC3700
AES Plaintext P_0 .. P_{127}	10439453 B0B87330 AFE2C12D 7DC18F00
AES Ciphertext W_0 .. 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 .. K_{127}	D0806449 99A0D8B7 36530000 00000000
AES Plaintext P_0 .. P_{127}	89E394B2 77B60000 D48E6D18 F1C655E0
AES Ciphertext W_0 .. 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 .. K_{127}	070557F0 2963399B 76CD42E8 00000000
AES Plaintext P_0 .. P_{127}	59BE23A3 CA4FFF00 4A877236 0C1A0000
AES Ciphertext W_0 .. 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 .. K_{127}	CA29ADAC B9651A3E BC77D383 D7A02763
AES Plaintext P_0 .. P_{127}	E79CEA2D 91CD0FD9 79957A90 B4040F6E
AES Ciphertext W_0 .. 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 .. K_{127}	7FB30F86 31EFA426 6E1ABA5D 9E4E0000
AES Plaintext P_0 .. P_{127}	9D459058 B476DD00 C9802A2C D2CB52D8
AES Ciphertext W_0 .. 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 .. K_{127}	906C2484 7949FEDD F67E9F7E CE0036A0
AES Plaintext P_0 .. P_{127}	E280BE2C 1C938EE8 A20F2C14 4FA00000
AES Ciphertext W_0 .. W_{127}	AE015682 05900C3C C51B2F62 2AD88A86
DSAA2-1 Secret Session Key E[0]..E[127]	DEAE8B48 65DF23EE 14B273E5 AAEEE00F

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 .. K_{127}	631C95B1 79BB52EB BA6B28EF 40000000
AES Plaintext P_0 .. P_{127}	1E171CB2 AF600000 6C0C2850 11136AB8
AES Ciphertext W_0 .. 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 010111111 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

2nd 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 .. K_{127}	42025EE3 39743AF6 47B57780 25E9B66D
AES Plaintext P_0 .. P_{127}	A0F3624E 949640A0 17DA4751 F5B2B180
AES Ciphertext W_0 .. W_{127}	FAA865E7 D215652B 07C045C8 EA8F3327
DSAA2-2 tag E[0]..E[31]	FAA865E7
Second AES Plaintext	FAA865E7 D215652B 07C045C8 EA8F3326
Second AES Ciphertext $C2_0$ .. $C2_{127}$	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 .. K_{127}	5DDB5285 D9072931 833B3556 7776898A
AES Plaintext P_0 .. P_{127}	8493A329 2770FA99 B950D6D2 E77E641D
AES Ciphertext W_0 .. W_{127}	2F2863B4 3F671819 7E292FCF 6BFAD22C
DSAA2-2 tag E[0]..E[31]	2F2863B4
Second AES Plaintext	2F2863B4 3F671819 7E292FCF 6BFAD22D
Second AES Ciphertext $C2_0$ .. $C2_{127}$	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 .. K_{127}	63340A1B D9BAF80A 25D20044 C354A708
AES Plaintext P_0 .. P_{127}	86167C7D 8D0EA8F4 74BBC76E C7C0F65E
AES Ciphertext W_0 .. W_{127}	728A4748 EAB6F36F ED3B8F96 44C64623
DSAA2-2 tag E[0]..E[31]	728A4748
Second AES Plaintext	728A4748 EAB6F36F ED3B8F96 44C64622
Second AES Ciphertext $C2_0$ .. $C2_{127}$	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 .. K_{127}	BCD32DB0 21D5D009 93428051 F1CC3700
AES Plaintext P_0 .. P_{127}	10439453 B0B87330 AFE2C12D 7DC18F00
AES Ciphertext W_0 .. W_{127}	D01DF4BD 4E2AC0E7 47E288E1 E9C805CD
DSAA2-2 tag E[0]..E[31]	D01DF4BD
Second AES Plaintext	D01DF4BD 4E2AC0E7 47E288E1 E9C805CC
Second AES Ciphertext $C2_0$ .. $C2_{127}$	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 .. K_{127}	D0806449 99A0D8B7 36530000 00000000
AES Plaintext P_0 .. P_{127}	89E394B2 77B60000 D48E6D18 F1C655E0
AES Ciphertext W_0 .. W_{127}	1A114590 82980B50 D212D04B 728B2D5E
DSAA2-2 tag E[0]..E[31]	1A114590
Second AES Plaintext	1A114590 82980B50 D212D04B 728B2D5F
Second AES Ciphertext $C2_0$ .. $C2_{127}$	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 .. K_{127}	070557F0 2963399B 76CD42E8 00000000
AES Plaintext P_0 .. P_{127}	59BE23A3 CA4FFF00 4A877236 0C1A0000
AES Ciphertext W_0 .. W_{127}	02B9ED2C 77BCEC10 583FC575 5712A8BB
DSAA2-2 tag E[0]..E[31]	02B9ED2C
Second AES Plaintext	02B9ED2C 77BCEC10 583FC575 5712A8BA
Second AES Ciphertext $C2_0$ .. $C2_{127}$	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 .. K_{127}	CA29ADAC B9651A3E BC77D383 D7A02763
AES Plaintext P_0 .. P_{127}	E79CEA2D 91CD0FD9 79957A90 B4040F6E
AES Ciphertext W_0 .. W_{127}	843B2133 E4F5D4F2 34BC0DE8 C66249DD
DSAA2-2 tag E[0]..E[31]	843B2133
Second AES Plaintext	843B2133 E4F5D4F2 34BC0DE8 C66249DC
Second AES Ciphertext $C2_0$ .. $C2_{127}$	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 .. K_{127}	7FB30F86 31EFA426 6E1ABA5D 9E4E0000
AES Plaintext P_0 .. P_{127}	9D459058 B476DD00 C9802A2C D2CB52D8
AES Ciphertext W_0 .. W_{127}	FEB8FDAE F0346784 CE20BA38 E7ECD656
DSAA2-2 tag E[0]..E[31]	FEB8FDAE
Second AES Plaintext	FEB8FDAE F0346784 CE20BA38 E7ECD657
Second AES Ciphertext $C2_0$ .. $C2_{127}$	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 .. K_{127}	906C2484 7949FEDD F67E9F7E CE0036A0
AES Plaintext P_0 .. P_{127}	E280BE2C 1C938EE8 A20F2C14 4FA00000
AES Ciphertext W_0 .. W_{127}	AE015682 05900C3C C51B2F62 2AD88A86
DSAA2-2 tag E[0]..E[31]	AE015682
Second AES Plaintext	AE015682 05900C3C C51B2F62 2AD88A87
Second AES Ciphertext $C2_0$ .. $C2_{127}$	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 .. K_{127}	631C95B1 79BB52EB BA6B28EF 40000000
AES Plaintext P_0 .. P_{127}	1E171CB2 AF600000 6C0C2850 11136AB8
AES Ciphertext W_0 .. W_{127}	F6683A52 A8704F8A 3C8102C0 D92F56D7
DSAA2-2 tag E[0]..E[31]	F6683A52
Second AES Plaintext	F6683A52 A8704F8A 3C8102C0 D92F56D6
Second AES Ciphertext $C2_0$ .. $C2_{127}$	52B1A3EA 0949A87F BD91D56A 85E64B53
DSAA2-2 Cipher Key E[0]..E[88]	52B1A3EA 0949A87F BD91D50

L.4 DSAA2 Examples

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

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} <<Portable-Identity>> <<AUTH-TYPE>>	05 44 05 07 80 A8 00 04 FF FF FF 0A 03 02 48 00
PP <<<--- FP	{KEY-ALLOCATE} <<Allocation-type>> <<RAND>> <<RS>>	05 42 0B 02 02 88 0C 08 8A 9F DD 3C D9 2F 6D 1E 0E 10 0E E7 0C 67 12 60 74 BD E0 39 6C D0 40 65 58 90
PP --->>> FP	{AUTH-REQUEST} <<AUTH-TYPE>> <<RAND>> <<RES>>	85 40 0A 03 02 48 00 0C 08 09 6D 5F 46 CA 0C EF 9E 0D 04 BE DC 30 A6
PP <<<--- FP	{AUTH-REPLY} <<RAND>> <<RES>> <<RS>>	05 41 0C 08 EB F0 D2 A4 27 FC F4 2F 0D 04 FC 5C 91 86 0E 10 DF 94 5F 3B 68 BB 92 B5 7C B8 F0 79 AA FC 5F 11
PP <<<--- FP	{ACCESS-RIGHTS-ACCEPT} <<Portable-Identity>> <<Fixed-Identity>>	85 45 05 07 80 A8 00 04 FF FF FF 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 .. K_{127}	FFFF9124 FFFF9124 FFFF9124 FFFF9124
AES Plaintext P_0 .. P_{127}	0EE70C67 126074BD E0396CD0 40655890
AES Ciphertext W_0 .. 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 .. K_{127}	A7260187 22AA6A7D 3E79CDAA 1F613E26
AES Plaintext P_0 .. P_{127}	8A9FDD3C D92F6D1E 096D5F46 CA0CEF9E
AES Ciphertext W_0 .. W_{127}	BEDC30A6 1EC2116F D07F3E92 B1ED4778
DSAA2-2 tag E[0]..E[31]	BEDC30A6
Second AES Plaintext	BEDC30A6 1EC2116F D07F3E92 B1ED4779
Second AES Ciphertext $C2_0$ .. $C2_{127}$	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 .. K_{127}	FFFF9124 FFFF9124 FFFF9124 FFFF9124
AES Plaintext P_0 .. P_{127}	DF945F3B 68BB92B5 7CB8F079 AAFC5F11
AES Ciphertext W_0 .. 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 .. K_{127}	CD257682 F4416053 7CD50DBF 1BDB145D
AES Plaintext P_0 .. P_{127}	096D5F46 CA0CEF9E EBF0D2A4 27FCF42F
AES Ciphertext W_0 .. W_{127}	FC5C9186 D2147C06 0EBA5B66 7A88B826
DSAA2-2 tag E[0]..E[31]	FC5C9186
Second AES Plaintext	FC5C9186 D2147C06 0EBA5B66 7A88B827
Second AES Ciphertext $C2_0$ .. $C2_{127}$	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

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} <<AUTH-TYPE>> <<RAND>> <<RS>>	05 40 0A 03 02 18 18 0C 08 6B A2 80 2B 91 0F F3 39 0E 10 3B 9F F6 1B DE 98 0B 45 EF A9 BF C5 D4 67 9C C9
PP --->>> FP	{AUTH-REPLY} <<RAND>> <<RES>>	85 41 0C 08 96 BD 28 62 C7 15 FC 88 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 .. K_{127}	CD257682 F4416053 7CD50DBF 1BDB145D
AES Plaintext P_0 .. P_{127}	3B9FF61B DE980B45 EFA9BFC5 D4679CC9
AES Ciphertext W_0 .. 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 .. K_{127}	C3D25E38 117444ED 4761F67A DB94F80E
AES Plaintext P_0 .. P_{127}	6BA2802B 910FF339 96BD2862 C715FC88
AES Ciphertext W_0 .. W_{127}	4096C0AF EEAA6826 91C381F8 62E26B1C
DSAA2-2 tag E[0]..E[31]	4096C0AF
Second AES Plaintext	4096C0AF EEAA6826 91C381F8 62E26B1D
Second AES Ciphertext $C2_0$ .. $C2_{127}$	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 section 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

DCK	01	A0	63	BA	8E	8A	07	CF
CK	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

DCK	01	A0	63	BA	8E	8A	07	CF	06	3C	6E	23	34	05	F1	4E
CK	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] .. CK[m-1], where $m \leq 128$
  - An n-bit initialization vector IV[0] .. IV[n-1], where $n \leq 64$
- Output:
  - N sequences of λ keystream bits KSS[0] .. KSS[λ-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 .. K_{127} be defined as follows:

- K_0 .. K_{m-1} = CK[0] .. CK[m-1]
- If $m < 128$, then $K_i = 0$ for $m \leq i \leq 128$

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 $P_{j,0}$ .. $P_{j,127}$ by:
  - $P_{j,63}$ .. $P_{j,64-n}$ = IV[0] .. IV[n-1]
  - If $n < 64$, then $P_{j,63-i} = 0$ for $n \leq i \leq 63$
  - $P_{j,64}$ .. $P_{j,127}$ 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, $P_{j,64}$ .. $P_{j,125}$ 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 .. K_{127} and the 128-bit plaintext block $P_{j,0}$ .. $P_{j,127}$. Let the resulting ciphertext block be $C_{j,0}$ .. $C_{j,127}$.
- 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 $P_{3,0}$ .. $P_{3,127}$: 00000000 00010001 00100010 00110011 01000100 01010101 01100110 01110111

00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000011

AES Ciphertext $C_{3,0}$ .. $C_{3,127}$: 11001010 11110011 00100101 10110111 10010000 11011111 01111000 00001100

11100111 10011110 10011100 00000011 11100011 01010001 00110111 00110101

AES Key K_0 .. K_{127} : 00000000 00000001 00000010 00000011 00000100 00000101 00000110 00000111

00001000 00001001 00001010 00001011 00001100 00001101 00001110 00001111

Hexadecimal format

AES Plaintext $P_{3,0}$ .. $P_{3,127}$: 00112233 44556677 00000000 00000003

AES Ciphertext $C_{3,0}$ .. $C_{3,127}$: CAF325B7 90DF780C E79E9C03 E3513735

AES Key K_0 .. K_{127} : 00010203 04050607 08090A0B 0C0D0E0F

Test Set	AES Key K_0 .. K_{127}	Block j	AES Plaintext $P_{j,0}$ .. $P_{j,127}$	AES Ciphertext $C_{j,0}$ .. $C_{j,127}$
1	00010203 04050607 08090A0B 0C0D0E0F	3	00112233 44556677 00000000 00000003	CAF325B7 90DF780C E79E9C03 E3513735
2	18BE6784 4AE13D6C 20000000 00000000	5	6DF15AF1 41BB26E9 00000000 00000005	32A30C3E BE25BD4D 52F051D8 697F34B2
3	2EA612DB 153C6000 00000000 00000000	32	440D491C 4D064DB7 00000000 00000020	8E26C2B0 CE13908E 926D2121 63399B2D
4	39B32D12 074D4DC8 00000000 00000000	13	5D037A5A 767D4509 00000000 0000000D	19C2FB26 47032E3A E8030B1B B34674D1
5	3B251E1F 6E5D1AD4 63CB6BFC 7F967FF5	11	323B2213 260D6B89 00000000 0000000B	8753B38E E1D28683 4612D684 B71A1120
6	301C0BDB 56AE0732 0120759A 235022EE	36	58786B36 5CFD3E12 00000000 00000024	24E689DD C3950DBB B496F109 B2E58EAB
7	3BF63A9E 797D5F49 0DDC4CAD 314F4000	0	49442E40 13661CD0 00000000 00000000	28B505F5 9336D2AB F4E98E65 7B751544
8	42307EB7 60000000 00000000 00000000	15	409D12E1 798B121F 00000000 0000000F	8ECE05E6 4EC7F08A BADFB269 50382207
9	58B026CA 36990902 7BB95772 139D7049	20	4A80187E 16C56899 00000000 00000014	F939F162 D5CD9C9C 9512E4D4 0D57C218
10	40805DB2 33800000 00000000 00000000	32	0FBF2F14 6AD6047E 00000000 00000020	0FCFCF7F E4273435 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 λ 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^{th} block is truncated to its 80 leftmost bits.

Test Set 11: λ = 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 .. K_{127}	5E9D489C 19166172 6B7232E6 401D71F0
AES Plaintext $P_{0,0}$ .. $P_{0,127}$	03847F4F 494A0677 00000000 00000000
AES Ciphertext $C_{0,0}$ .. $C_{0,127}$	B5D23E32 B712B63E 376360FB 522CB1F5
DSC2 Keystream KSS[0]..KSS[127]	B5D23E32 B712B63E 376360FB 522CB1F5
AES Plaintext $P_{1,0}$ .. $P_{1,127}$	03847F4F 494A0677 00000000 00000001
AES Ciphertext $C_{1,0}$ .. $C_{1,127}$	1EDD0ED9 8B555347 7227CA87 1B8E652A
DSC2 Keystream KSS[128]..KSS[255]	1EDD0ED9 8B555347 7227CA87 1B8E652A
AES Plaintext $P_{2,0}$ .. $P_{2,127}$	03847F4F 494A0677 00000000 00000002
AES Ciphertext $C_{2,0}$ .. $C_{2,127}$	DBB64A1C A85427BE AC153100 E95F5C0E
DSC2 Keystream KSS[256]..KSS[383]	DBB64A1C A85427BE AC153100 E95F5C0E
AES Plaintext $P_{3,0}$ .. $P_{3,127}$	03847F4F 494A0677 00000000 00000003
AES Ciphertext $C_{3,0}$ .. $C_{3,127}$	F7904EAE 70CF0F20 949D1C47 8835EE69
DSC2 Keystream KSS[384]..KSS[511]	F7904EAE 70CF0F20 949D1C47 8835EE69
AES Plaintext $P_{4,0}$ .. $P_{4,127}$	03847F4F 494A0677 00000000 00000004
AES Ciphertext $C_{4,0}$ .. $C_{4,127}$	D4CDE775 AECF7B7E 65C00034 3088C020
DSC2 Keystream KSS[512]..KSS[639]	D4CDE775 AECF7B7E 65C00034 3088C020
AES Plaintext $P_{5,0}$ .. $P_{5,127}$	03847F4F 494A0677 00000000 00000005
AES Ciphertext $C_{5,0}$ .. $C_{5,127}$	D99C1529 E6D636F7 7E328DDB 0B3E21FD
DSC2 Keystream KSS[640]..KSS[719]	D99C1529 E6D636F7 7E32

Test Set 12: λ = 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 .. K_{127}	285248DB 27251640 00000000 00000000
AES Plaintext $P_{0,0}$ .. $P_{0,127}$	13D329D8 0A2809CE 00000000 00000000
AES Ciphertext $C_{0,0}$ .. $C_{0,127}$	7E7246AF 0B138505 D1AFD2B7 8A8ADD1C
DSC2 Keystream KSS[0]..KSS[127]	7E7246AF 0B138505 D1AFD2B7 8A8ADD1C
AES Plaintext $P_{1,0}$ .. $P_{1,127}$	13D329D8 0A2809CE 00000000 00000001
AES Ciphertext $C_{1,0}$ .. $C_{1,127}$	D8D56D3C D8E8E137 2044DFFD 2DDF0257
DSC2 Keystream KSS[128]..KSS[255]	D8D56D3C D8E8E137 2044DFFD 2DDF0257
AES Plaintext $P_{2,0}$ .. $P_{2,127}$	13D329D8 0A2809CE 00000000 00000002
AES Ciphertext $C_{2,0}$ .. $C_{2,127}$	BFB4BC89 7F20B8F4 B3BF0A77 CFFB95F7
DSC2 Keystream KSS[256]..KSS[383]	BFB4BC89 7F20B8F4 B3BF0A77 CFFB95F7
AES Plaintext $P_{3,0}$ .. $P_{3,127}$	13D329D8 0A2809CE 00000000 00000003
AES Ciphertext $C_{3,0}$ .. $C_{3,127}$	006D0A16 E03970F0 057CA562 4DCE6451
DSC2 Keystream KSS[384]..KSS[511]	006D0A16 E03970F0 057CA562 4DCE6451
AES Plaintext $P_{4,0}$ .. $P_{4,127}$	13D329D8 0A2809CE 00000000 00000004
AES Ciphertext $C_{4,0}$ .. $C_{4,127}$	9AF557A7 2570FA1C AB64AACF 694EA025
DSC2 Keystream KSS[512]..KSS[639]	9AF557A7 2570FA1C AB64AACF 694EA025
AES Plaintext $P_{5,0}$ .. $P_{5,127}$	13D329D8 0A2809CE 00000000 00000005
AES Ciphertext $C_{5,0}$ .. $C_{5,127}$	13BDBDE7 3CB9A76C E2329591 2ABDAE34
DSC2 Keystream KSS[640]..KSS[719]	13BDBDE7 3CB9A76C E232

Test Set 13: λ = 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 .. K_{127}	06E30A6C 43280000 00000000 00000000
AES Plaintext $P_{0,0}$ .. $P_{0,127}$	12C21003 00000000 00000000 00000000
AES Ciphertext $C_{0,0}$ .. $C_{0,127}$	D3163CBE 4E3BD210 DC1900B5 1FA65322
DSC2 Keystream KSS[0]..KSS[127]	D3163CBE 4E3BD210 DC1900B5 1FA65322
AES Plaintext $P_{1,0}$ .. $P_{1,127}$	12C21003 00000000 00000000 00000001
AES Ciphertext $C_{1,0}$ .. $C_{1,127}$	76D1AEE3 36505B2B 1FFAB960 EE92EDC9
DSC2 Keystream KSS[128]..KSS[255]	76D1AEE3 36505B2B 1FFAB960 EE92EDC9
AES Plaintext $P_{2,0}$ .. $P_{2,127}$	12C21003 00000000 00000000 00000002
AES Ciphertext $C_{2,0}$ .. $C_{2,127}$	A58AAF8B F489D659 30210BA3 BB4706CD
DSC2 Keystream KSS[256]..KSS[383]	A58AAF8B F489D659 30210BA3 BB4706CD
AES Plaintext $P_{3,0}$ .. $P_{3,127}$	12C21003 00000000 00000000 00000003
AES Ciphertext $C_{3,0}$ .. $C_{3,127}$	AF5CB0B6 058B7441 BA78C21D 71FCE174
DSC2 Keystream KSS[384]..KSS[511]	AF5CB0B6 058B7441 BA78C21D 71FCE174
AES Plaintext $P_{4,0}$ .. $P_{4,127}$	12C21003 00000000 00000000 00000004
AES Ciphertext $C_{4,0}$ .. $C_{4,127}$	0CBE6B10 93F9EEDC 1EA51372 57467E8E
DSC2 Keystream KSS[512]..KSS[639]	0CBE6B10 93F9EEDC 1EA51372 57467E8E
AES Plaintext $P_{5,0}$ .. $P_{5,127}$	12C21003 00000000 00000000 00000005
AES Ciphertext $C_{5,0}$ .. $C_{5,127}$	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, λ = 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..K_{127}	01A063BA 8E8A07CF 063C6E23 3405F14E
AES Plain Text P0,0..P0,127	00000000 51122334 00000000 00000000
AES Ciphertext C0,0..C0,127	A5F3E18F 28BF5630 9FBB34B2 C0514789
DSC2 Keystream KSS[0]..KSS[127]	A5F3E18F 28BF5630 9FBB34B2 C0514789
AES Plain Text $P_{1,0}$..$P_{1,127}$	00000000 51122334 00000000 00000001
AES Ciphertext $C_{1,0}$..$C_{1,127}$	D4C4E706 85E7B5F2 89E83A32 04C6F8E0
DSC2 Keystream KSS[128]..KSS[255]	D4C4E706 85E7B5F2 89E83A32 04C6F8E0
AES Plain Text $P_{2,0}$..$P_{2,127}$	00000000 51122334 00000000 00000002
AES Ciphertext $C_{2,0}$..$C_{2,127}$	83526749 43823181 C02E2B9D B79C192E
DSC2 Keystream KSS[256]..KSS[383]	83526749 43823181 C02E2B9D B79C192E
AES Plain Text $P_{3,0}$..$P_{3,127}$	00000000 51122334 00000000 00000003
AES Ciphertext $C_{3,0}$..$C_{3,127}$	26EB9801 23B431D6 A9450A30 AC9C1E55
DSC2 Keystream KSS[384]..KSS[511]	26EB9801 23B431D6 A9450A30 AC9C1E55
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 D4 C4 E7 06 85 E7 B5 F2 89 E8 3A 32 04 C6 F8 E0 83 52 67 49 43 82 31 81 C0 2E 2B 9D B7 9C 19 2E 26 EB 98 01 23 B4 31 D6 A9 45 0A 30 AC 9C 1E 55 CF A4 2F BE 32 51 84 49 43 4F 5D 79 BC 90 79 9D 15 8D 6B D0 4D BD AE 69 06 36 1A CB 03 00 9C 1D B2 0B 01 AB 2E EA 60 2F 36 89 26 9D C4 C2 3A 28 A0 97 85 E8 5C D6 1D 44 23 A7 BB 99 6B D0 D4 01 F2 5C CE 27 33 84 3E EB B4 99 9B 01 85 86 77 DC A8 6B BA 05 3E E3 2E 5B B2 2F 67 4E 0C 09 A8 63 28 7D 60 CC 93 71 30 E8 09 DA

variable	Value (Hexadecimal, except when otherwise noted)
DSC2 Keystream KSS[0]..KSS[679] (used in 1st half frame)	A5 F3 E1 8F 28 BF 56 30 9F BB 34 B2 C0 51 47 89 D4 C4 E7 06 85 E7 B5 F2 89 E8 3A 32 04 C6 F8 E0 83 52 67 49 43 82 31 81 C0 2E 2B 9D B7 9C 19 2E 26 EB 98 01 23 B4 31 D6 A9 45 0A 30 AC 9C 1E 55 CF A4 2F BE 32 51 84 49 43 4F 5D 79 BC 90 79 9D 15 8D 6B D0 4D
DSC2 Keystream KSS[680]..KSS[1359] (used in 2nd half frame)	BD AE 69 06 36 1A CB 03 00 9C 1D B2 0B 01 AB 2E EA 60 2F 36 89 26 9D C4 C2 3A 28 A0 97 85 E8 5C D6 1D 44 23 A7 BB 99 6B D0 D4 01 F2 5C CE 27 33 84 3E EB B4 99 9B 01 85 86 77 DC A8 6B BA 05 3E E3 2E 5B B2 2F 67 4E 0C 09 A8 63 28 7D 60 CC 93 71 30 E8 09 DA
A-field data	01 2C 01 43 1D BD
A-field encrypted (1st half frame)	01 89 F2 A2 92 95
A-field encrypted (2nd half frame)	01 91 AF 2A 1B 8B
B-field data	BF 4E F6 BC FC 7D B5 AF F5 FF F8 F0 F0 73 DB 2C 6E DB F3 F4 B0 BC 6E BC F3 F7 FA 75 75 58 73 3B 37 5A 54 B5 F1 DD FB 70 DE 1E 7C 55 D6 B9 52 59 FF D7 F7 90 70 9B 55 1C D8 0F 53 D6 7A 7C FE BB 5B 33 1C AB D4 DF 3F 15 56 D8 FA 71 D8 D1 F1 F6
B-field encrypted (1st half frame)	00 18 C6 23 47 49 07 6F A4 B8 71 24 34 94 DD A9 89 6E 01 7D 58 86 5C B8 35 0F 1A F6 27 3F 3A 78 B5 6B D5 75 DF F6 66 C7 42 07 52 73 3D 21 53 7A 4B E6 21 39 35 91 65 B0 44 11 06 19 DE 53 40 89 0A B7 55 E8 9B 82 46 A9 C6 A1 67 64 55 BA 21 BB
B-field encrypted (2nd half frame)	A5 85 F5 BC 60 60 07 A4 F4 54 D6 1A 90 5C ED A5 48 46 37 36 8A 94 CE 2B 76 1F A6 A3 68 1C 50 9C 8C C3 3F 65 25 DC 09 2C 10 39 4F D1 E8 52 E6 C0 64 D6 72 16 07 47 FD 77 62 0A 6D 35 54 27 4C 94 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 below:

AES-128 plaintext				
P_0				P_{127}
000...000 (32 bits)	LBN* (4 bits)	Multiframe number (24 bits)	Frame number (4 bits)	j (64 bits)
	msb lsb	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

CCM (Counter with CBC-MAC) is an Authenticated Encryption algorithm designed to provide both packet authentication and encryption. It is defined by the RFC 3610 [11]. The DECT implementation of CCM uses the Advanced Encryption Standard (AES) as 128 bit block cipher (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

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 EN 300 175-4 [4]).

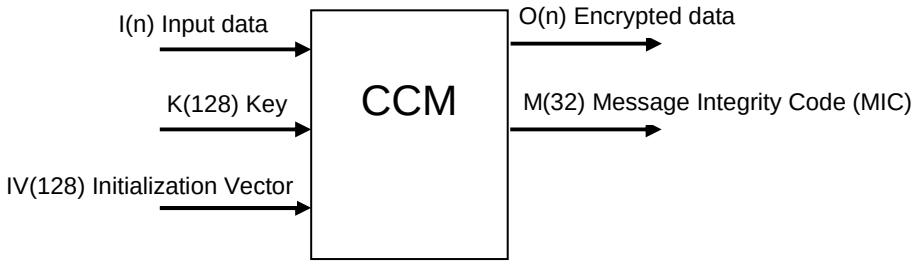
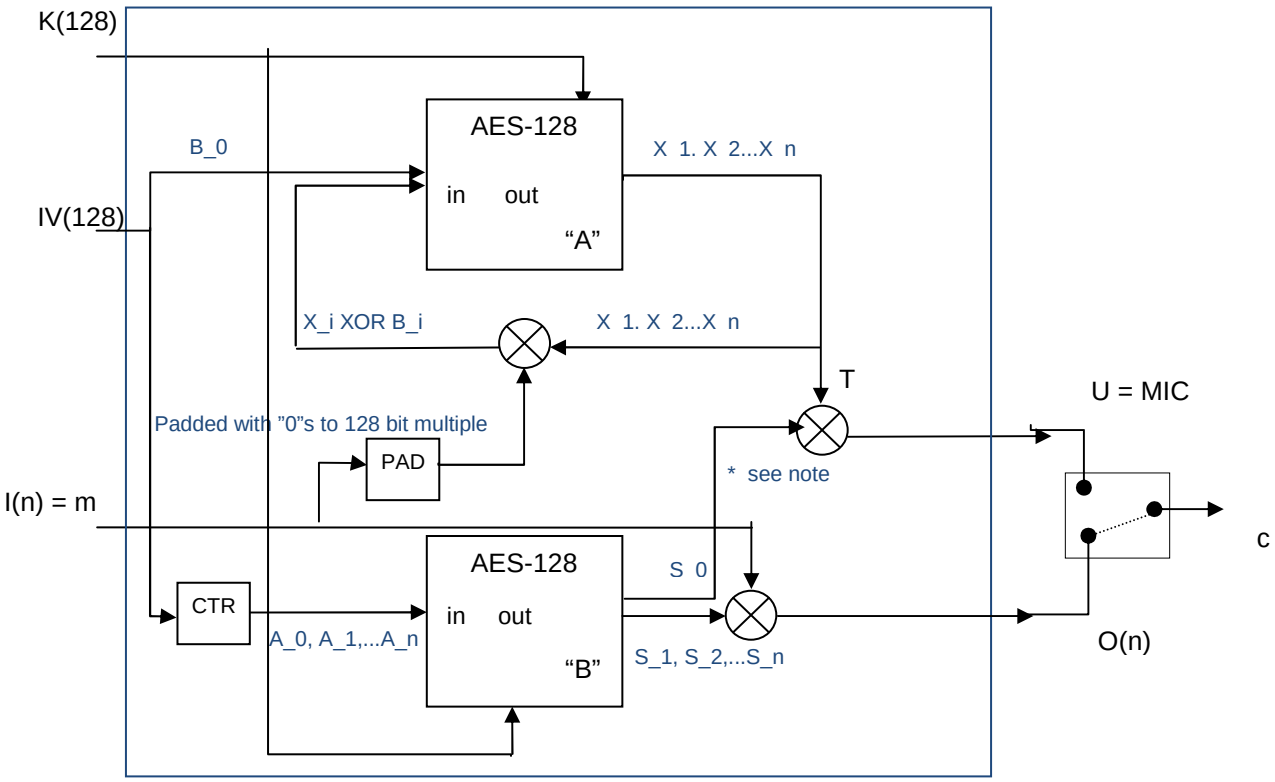


Figure N.1: CCM security process overview

N.2.1 Description of the CCM algorithm: encryption



NOTE: The segment S_0, if generated by the first execution of "B", has to be stored in order to be multiplied by "T" that is generated later. Another option is generating S_0 after the other S_x segments, what is possible due to the nature of the CTR function.

Figure N.2: CCM security processes

The internal structure of the CCM security process (encryption side) is shown in figure N.2.

N.2.1.1 Block ciphers

The CCM process contains two block ciphers named "A" and "B". Each one of these blocks execute the Advanced Encryption Standard Algorithm AES as defined by [10], with block size 128 bits. The block "A" is in charge of generating the Message Integrity Code (MIC). The block "B" is in charge of generating the encryption stream.

One of the two inputs of the AES ciphers is the Key (K(128)). The other input is generated from the IV and the message itself as described.

N.2.1.2 Counter function (CTR)

The input of cipher block "B" is generated from the Initialization Vector (IV) using the Counter function (CTR) which performs the following modifications:

- Byte 0 of the IV is changed to "01"H, fixed value.
- Bytes 1 to 13 (the nonce) are unchanged.
- Bytes 14 and 15 (the message length) are changed by a 16 bit counter (least significant octet in byte 15) starting with "0" and incremented for each execution of the AES cipher block "B".

The outputs of the CTR function are named A_0, A_1...A_n, where n is the value of the counter.

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.

$$S\_x = AES (K, A\_x)$$

The first segment, S_0, is reserved and will be used in the generation of the MIC. All other segments S_1, S_2...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: RFC 3610 [11] description uses the terminology B_1...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...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 RFC 3610 [11]) with the equation:

$$X\_1 = AES (K, IV)$$

The following X segments (X_2..X_n) are generated with the equation:

$$X\_i+1 = AES(K, X\_i \text{ XOR } B\_i)$$

Where B_i (B_1...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)).

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

History

Document history		
Edition 1	October 1992	Publication as ETS 300 175-7 (Historical)
Edition 2	September 1996	Publication as ETS 300 175-7 (Historical)
Edition 3	September 1997	Publication as 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
V2.5.0	April 2013	EN Approval Procedure AP 20130820: 2013-04-22 to 2013-08-20