ETSI TS 104 053-1 V1.2.1 (2025-02)

TECHNICAL SPECIFICATION

TETRA Air Interface Security, Algorithms Specifications;
Part 1: TETRA Encryption Algorithms, TEA Set A

Reference
RTS/TCCE-06224

Keywords
air interface, algorithm, DMO, security, TETRA, V+D

ETSI

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

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

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

Contents

Intellectual Property Rights

Essential patents

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

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

Trademarks

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

DECT™, **PLUGTESTS™**, **UMTS™** and the ETSI logo are trademarks of ETSI registered for the benefit of its Members. **3GPP™**, **LTE™** and **5G™** logo are trademarks of ETSI registered for the benefit of its Members and of the 3GPP Organizational Partners. **oneM2M™** logo is a trademark of ETSI registered for the benefit of its Members and of the oneM2M Partners. **GSM**® and the GSM logo are trademarks registered and owned by the GSM Association.

Foreword

This Technical Specification (TS) has been produced by ETSI Technical Committee TETRA and Critical Communications Evolution (TCCE).

The present document is part 1 of a multi-part deliverable covering the specifications of the TETRA standard encryption, authentication and key management algorithms, as identified below:

Part 1: "TETRA Encryption Algorithms, TEA Set A";

Part 2: "TETRA Encryption Algorithms, TEA Set B";

Part 3: "TETRA Authentication and Key Management Algorithms TAA1";

Part 4: "TETRA Authentication and Key Management Algorithms TAA2".

Modal verbs terminology

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

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

1 Scope

The present document specifies the Terrestrial Trunked Radio system (TETRA) set A encryption algorithms TEA1, TEA2, TEA3 and TEA4.

The TETRA Air interface security function provides mechanisms for confidentiality of control signalling and user speech and data at the air interface, authentication and key management mechanisms for the air interface and for the Inter-System Interface (ISI). TETRA Air Interface security mechanisms are described in the TETRA V+D security specification [1] and the TETRA Direct Mode security specification [2].

2 References

2.1 Normative references

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

Referenced documents which are not found to be publicly available in the expected location might be found at ETSI docbox.

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

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

[1] ETSI TS 100 392-7: "Terrestrial Trunked Radio (TETRA); Voice plus Data (V+D); Part 7: Security".

[2] ETSI TS 100 396-6: "Terrestrial Trunked Radio (TETRA); Direct Mode Operation (DMO); Part 6: Security".

2.2 Informative references

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

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

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

Not applicable.

3 Definition of terms, symbols and abbreviations

3.1 Terms

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

Cipher Key (CK): value that is used to determine the transformation of plain text to cipher text in a cryptographic algorithm

cipher text: data produced through the use of encipherment

decipherment: reversal of a corresponding reversible encipherment

encipherment: cryptographic transformation of data to produce cipher text

Initialization Vector (IV): sequence of symbols that randomize the KSG inside the encryption unit

key stream: pseudo random stream of symbols that is generated by a KSG for encipherment and decipherment

LENGTH: required length of the key stream in bits

plain text: un-encrypted source data

TEA set A: set of air interface encryption algorithms comprising TEA1, TEA2, TEA3 and TEA4

TEA set B: set of air interface encryption algorithms comprising TEA5, TEA6 and TEA7

TETRA algorithm: mathematical description of a cryptographic process used for either of the security processes authentication or encryption

3.2 Symbols

Void.

3.3 Abbreviations

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

CK Cipher key
ISI Inter Systems Interface
IV Initialization Vector
KSG Key Stream Generator
TC Technical Committee
TCCE TETRA and Critical Communications Evolution
TEA1 TETRA Encryption Algorithm No. 1
TEA2 TETRA Encryption Algorithm No. 2
TEA3 TETRA Encryption Algorithm No. 3
TEA4 TETRA Encryption Algorithm No. 4
TETRA TErrestrial Trunked Radio

4 TEA SET A Specifications

The algorithms TEA1, 2, 3 and 4 specified in the present document generate a sequence of key bytes from a Cipher Key CK and an Initialization Vector IV.

The key bytes are used to encrypt or to decrypt information transmitted via the TETRA system.

The Cipher Key has a length of 80 bits; the Initialization Vector is 29 bits.

The present document is organized as follows: clause 5 describes TEA1, clause 6 TEA2, clause 7 TEA3 and clause 8 TEA4.

Clauses 5.1, 6.1, 7.1 and 8.1 provide a functional specification of the functional components of the algorithms, clauses 5.2, 6.2, 7.2 and 8.2 specify how each of these components are used to generate the key bytes and clauses 5.3, 6.3, 7.3 and 8.3 contain diagrams and tables for the respective algorithms.

5 TEA1 ALGORITHM DESCRIPTION

5.1 TEA1 Functional Components

5.1.1 Summary of Components

The cryptographic algorithm TEA1 consists of the following functional components as depicted in Figure 1:

- A set of eight 8-bit wide shift registers, called Output Register (clause 5.1.3).
- A set of four 8-bit wide shift registers, called Key Register (clause 5.1.4).
- A byte permutation function *P* (byte substitution) (clause 5.1.5).
- Two functions *E* to expand the 16-bit output of two 8-bit registers to 32 bits word (clause 5.1.6).
- Two nonlinear functions f_1 and f_2 to compute a byte from the expanded 32-bit word (clause 5.1.7).
- An 8-bit permutation function *BP* (wire crossing) (clause 5.1.8).
- Five 8-bit wide XOR functions to combine Section outputs (bytes). And
- Functions in the feedback of the key register and the output register.

5.1.2 Notation

The symbol $\oplus$ denotes the bytewise addition modulo two (exclusive or); i.e. msb of byte x is added to msb of byte y , ..., lsb of byte x to lsb of byte y.

x (i)	*y(i)*	*x(i)* $\oplus$ *y(i)*
0	0	0
0	1	1
1	0	1
1	1	0
x(i) is bit *i* of byte *x* *y(i)* is bit *i* of byte *y* i = *0,...,7*		

5.1.3 Output Register

The output register is denoted by R_0, R_1..., R_6, and R_7, and functions as an 8-bit wide shift register. The output bytes of sections R_1, R_2 and R_4 up to R_6 are also used as input for the other functional components of the TEA1. The output of R_7 is added to the outputs of the other functional components and returned to the first section R_0 as depicted in Figure 1.

Each 19 steps the byte output of R_7 is used as key byte for encryption or decryption.

Before the process as described above can take place the output register is loaded with an Initialization Vector (see clause 5.2.3) and initialized (see clause 5.2.4).

5.1.4 Key Register

The key register, denoted by K_0, K_1, K_2 and K_3, is a four section 8-bit wide shift register. The bytes of a Cipher Key (CK) are loaded into the key register as shown in Figure 2, and before the Initialization Vector is loaded into the Output Register. During loading the CK bytes are added modulo two to the result of the modulo two addition of the output of sections K_0 and K_3. This result is substituted by the permutation function P and passed to section K_0. The loading process starts from the reset state 0000 of the K_i register.

During initialization and the generation of key bytes the same feedback process, however without the CK_i input (= 0), is continued as follows:

with K'_i denoting the next byte value (i.e. after one step) of the register section K'_i.

$K'_0 = P (K_3 \oplus K_0)$

$K'_1 = K_0$

$K'_2 = K_1$

$K'_3 = K_2$

The series of mixed and permutated bytes is also offered to the feedback circuit of the output register.

5.1.5 Byte Permutation Function

The byte permutation function P is a randomly chosen permutation in the set of all 256 possible bytes. The look up table for this permutation is given in Figure 3. The higher nibble of the output is the left one of the output value, e.g. P (27) = 6A.

5.1.6 Expander E

The expander function converts a two-byte input to a 32-bit word, i.e. eight 4-bit nibbles for the nonlinear functions f_1 and f_2. The structure of expander E and functions f_1 and f_2 is shown in Figure 4.

In this figure, bits 1-8 are the bits of R_2, respectively R_6.

Bits 9-16 are the bits of R_1, respectively R_5. Bits 1 and 9 are the most significant bits. The 4-bit nibble inputs to the S_i boxes are the result of reading R_2 R_1 (for the input of f_1), respectively R_6 R_5 (for the input of f_2). The bit left is the msb of each 4-bit nibble input.

5.1.7 Nonlinear Function f_1

The nonlinear functions f_1 and f_2 have the structure shown in Figure 4. Each of the boxes S_1 to S_8 receives the nibble input concerned from the expander E and computes a bit for the output byte. S_1 is the most significant bit in the output byte, and S_8 is the least significant one.

The functions f_1 and f_2 are given in the truth tables, Figure 5 and Figure 6, respectively. The hexadecimal value of the input nibble for each S_i is the one given in the top row of these figures. The computed output bit for the corresponding S_i; can be read in the column below the input nibble row.

5.1.8 8-bit Permutation BP

The permutation BP is a so-called wire-crossing with a fixed pattern. If the eight bits of R_4 are numbered 12345678 the order of the bits after BP becomes 58417326. The left bit in both bytes is the most significant; the right bit is the least significant.

5.1.9 Feedback of output register

The results of the functional components BP, f_1, f_2 and CK byte permutation P are added in the feedback path of the output register and affect. the operation (i.e. operation after loading the CK and Initialization vector) as follows:

with R'_i denoting the next byte value (i.e. after one step) of the output register Section R_i:

$R'_o = R_7 \oplus f_2 (R_6, R_5) \oplus BP (R_4) \oplus P_{out}$

$R'_1 = R_o$

$R'_2 = R_1$

$R'_3 = R_2$

$R'_4 = R_3 \oplus f_1 (R_2, R_1)$

$R'_5 = R_4$

$R'_6 = R_5$

$R'_7 = R_6$

5.2 Key Stream Generation

5.2.1 Summary

The algorithm consists of four main phases: the CK loading, the IV loading, the run-up and the key byte generation proper.

During the CK loading the initial state of the key register is determined. Next, the initial state of the output register is determined by loading of the Initialization Vector as described in clause 5.2.3. Thereafter, a run-up is carried out during which the initial contents of the Output Register (converted and adapted IV) and the K_i Register are changed by the effect of the *BP*, *E*, f_1, f_2 and *P* functions (see clause 5.2.4).

After the run-up is completed, each output cycle produces a key byte from the key byte stream as specified in clause 5.2.5. At any point during the run-up and the generation of the key byte stream, the state of the algorithm is determined by the states of the output and the K_i registers.

5.2.2 CK loading

The CK loading depends on the 80-bit Cipher Key, the feedback method and the permutation function *P*. The 10 CK bytes are loaded as depicted in clause 5.3, Figure 2. The reduced CK, i.e. the 32 bits available in the K_i register after the loading process, will affect the further initialization of the algorithm and the generation of key bytes.

5.2.3 IV loading

The output register is first loaded with a.29-bit Initialization Vector. This IV is converted to a 32-bit word by taking the three most significant bits as zeroes, and the remaining 29 bit as the given IV. For instance, if the running counter is the binary value:

11010   00011010 11100010 00000110

the 32-bit word becomes:

00011010   00011010 11100010 00000110.

The least significant byte of that 32-bit word is loaded into R_3, the next byte into R_4, the next one into R_5, and, finally, the most significant (the padded one) becomes the R_6 initial value. The cells R_0, R_1, R_2 and R_7 are loaded in the same order with the 32-bit word XORed with the constant mask 96724FA1. Then, we have the following initialization assignments, with the running counter as a four-byte number $F_1F_2F_3F_4$:

$R_7 = F_1 \oplus 96$

$R_6 = F_1$

$R_5 = F_2$

$R_4 = F_3$

$R_3 = F_4$

$R_2 = F_2 \oplus 72$

$R_1 = F_3 \oplus 4F$

$R_0 = F_4 \oplus Al$

Using the IV, mentioned in the example above, the result is:

1 8

10001 100 00011010 00011010

00000110 01101000 10100111

for the R_7, ..., R_4 and R_3, ..., R_0 bytes.

The eight bits of each R; numbered from 1 to 8 are loaded into the register locations:

R_i (7), R_i (6), ..., R_i (0) respectively.

5.2.4 Run-up

After the IV load into the output register all functional components of the TEA1 become operational and 53 initializing steps are performed to bring the algorithm in the CK and IV dependent starting point from which the generation of key bytes can start.

For run-up and key byte generation a step is defined as applying the formulae in clauses 5.1.4 and 5.1.9 once.

5.2.5 Key byte generation

After the run-up cycle one step is made to produce the first key byte. Successive key bytes are generated each 19 steps.