TR45.0.A

Interface Specification for Common Cryptographic Algorithms, Revision B

August 6, 1996

NOTICE

TIA Engineering Standards and Publications are designed to serve the public interest through eliminating misunderstandings between manufacturers and purchasers, facilitating inter-changeability and improvement of products, and assisting the purchaser in selecting and obtaining with minimum delay the proper product for their particular need. Existence of such Standards and Publications shall not in any respect preclude any member or non-member of TIA from manufacturing or selling products not conforming to such Standards and Publications, nor shall the existence of such Standards and Publications preclude their voluntary use by those other than TIA members, whether the standard is to be used either domestically or internationally.

Standards and Publications are adopted by TIA without regard to whether or not their adoption may involve patents or articles, materials, or processes. By such action, TIA does not assume any liability to any patent owner, nor does it assume any obligation whatever to parties adopting the Recommended Standard or Publication.

Standards and Publications are adopted by EIA/TIA without regard to whether or not their adoption may involve patents or articles, materials, or processes. By such action, EIA/TIA does not assume any liability to any patent owner, nor does it assume any obligation whatever to parties adopting the Recommended Standard or Publication.

TIA TR45 Ad Hoc Authentication Group Documents

TIA TR45 Ad Hoc Authentication Group Documents contain information deemed to be of technical value to the industry, and are published at the request of the TR45 Ad Hoc Authentication Group without necessarily following the rigorous public review and resolution of comments which is a procedural part of the development of a TIA Standard.

Contact

TELECOMMUNICATIONS INDUSTRY ASSOCIATION
Engineering Department
2500 Wilson Boulevard, Suite 300
Arlington, Virginia 22201

Document History

Revision	Date	Remarks
0	02-05-93	Frozen for PN-3118 Ballot
0.1	04-21-93	Adopted by TR45 AHAG
1.00	10-20-94	Draft including data encryption and A-key checksum calculation
A	12-14-94	Major revision, incorporating ORYX data encryption algorithms and ANSI C algorithm descriptions
B	08-06-96	Added wireless residential extension authentication

Table of Contents

1. Introduction

This document describes the interfaces to cryptographic procedures for cellular system applications. These procedures are used to perform the security services of mobile station authentication, subscriber message encryption, and encryption key and subscriber voice privacy key generation within cellular equipment. The procedures are described in detail in "Common Cryptographic Algorithms."

The purpose of this specification is to describe the cryptographic functions without revealing the technical details that are subject to the export jurisdiction of the US Department of State as specified in International Traffic in Arms Regulations (ITAR), Title 22 CFR parts 120 through 130 inclusive. It is intended that developers of EIA/TIA standards for systems using these cryptographic functions use the information in this document in standards that are not subject to ITAR restrictions.

The procedures are described in the document as follows:

§2.1 describes the procedure to verify the manual entry of the subscriber authentication key (A-key).

§2.2 describes the generation of intermediate subscriber cryptovariables, Shared Secret Data (SSD), from the unique and private subscriber A-key.

§2.3 describes the procedure to calculate an authentication signature used by cellular base station equipment for verifying the authenticity of a mobile station.

§2.4 describes the procedure used for generating cryptographic keys.

§2.5 describes the procedure used for enciphering and deciphering subscriber data exchanged between the mobile station and the base station.

§2.6 describes the procedures for wireless residential extension authentication.

§2.7 describes the procedures for key and mask generation for encryption and decryption in cellular data services.

Manufacturers are cautioned that no mechanisms should be provided for the display at the mobile station (or any other equipment that may be interfaced with it) of valid A-key, SSD_A, SSD_B, or other cryptovariables associated with the cryptographic functions described in this document. The invocation of test mode in the mobile station must not alter the operational values of A-key, SSD_A, SSD_B or other cryptovariables.

1.1 Definitions

ACRE	Authorization and Call Routing Equipment. A network device which authorizes the Personal Base and provides automatic call routing.
ACRE_PHONE_NUMBER	A 24-bit pattern comprised of the last 6 digits of the ACRE's directory number.
A-key	A 64-bit cryptographic key variable stored in the semi-permanent memory of the mobile station and also known to the Authentication Center (AC or HLR/AC) of the cellular system. It is entered once from the keypad of the mobile station when the mobile station is first put into service with a particular subscriber, and usually will remain unchanged unless the operator determines that its value has been compromised. The A-key is used in the SSD generation procedure.
AND	Bitwise logical AND function.
Boolean	Describes a quantity whose value is either TRUE or FALSE.
CMEA	Cellular Message Encryption Algorithm.
DataKey	A 32-bit cryptographic key used for generation of masks for encryption and decryption in cellular data services.
Directory Number	The telephone network address.
ESN	The 32-bit electronic serial number of the mobile station.
Internal Stored Data	Stored data that is defined locally within the cryptographic procedures and is not accessible for examination or use outside those procedures.
LSB	Least Significant Bit.
MSB	Most Significant Bit.
PB	Personal Base. A fixed device which provides cordless telephone like service to a mobile station.
PBID	Personal Base Identification Code.
SSD	SSD is an abbreviation for Shared Secret Data. It consists of two quantities, SSD_A and SSD_B.
SSD_A	A 64-bit binary quantity in the semi-permanent memory of the mobile station and also known to the authenticating entity (normally the HLR or VLR). It is used in the computation of the authentication response.
SSD_A_NEW	The revised 64-bit quantity held separately from SSD_A, generated as a result of the SSD generation process.
SSD_B	A 64-bit binary quantity in the semi-permanent memory of the mobile station and also known to the authenticating entity (normally the HLR or VLR). It is used in the computation of the CMEA and VPM.
SSD_B_NEW	The revised 64-bit quantity held separately from SSD_B, generated as a result of the SSD generation process.
VPM	Voice Privacy Mask. This name describes a 520-bit entity that may be used for voice privacy functions as specified in cellular system standards.
WIKEY	Wireline Interface key. A 64-bit pattern stored in the PB and the ACRE in semi-permanent memory.

WIKEY_NEW	A 64-bit pattern stored in the PB and the ACRE. It contains the value of an updated WIKEY.
WRE_KEY	Wireless Residential Extension key. A 64-bit pattern stored in the PB and the mobile station in semi-permanent memory.

2. Procedures

2.1 Authentication Key (A-Key) Procedures

2.1.1 A-Key Checksum Calculation

Procedure name:	
A_Key_Checksum	
Inputs from calling process:	
A_KEY_DIGITS	20 decimal digits
ESN	32 bits
Inputs from internal stored data:	
	(internal definition only)
Outputs to calling process:	
A_KEY_CHECKSUM	6 decimal digits
Outputs to internal stored data:	
	None.

This procedure computes the checksum for an A-key to be entered into a mobile station. In a case where the number of digits to be entered is less than 20, the leading most significant digits will be set equal to zero.

The generation of the A-key is the responsibility of the service provider. A-keys should be chosen and managed using procedures that minimize the likelihood of compromise.

The checksum provides a check for the accuracy of the A-Key when entered into a mobile station. The checksum is calculated for the 20 A-Key digits input to the algorithm. The checksum is returned as 6 decimal digits for entry into the mobile station.

The first decimal digit of the A-Key to be entered is considered to be the most significant of the 20 decimal digits, followed in succession by the other nineteen. For example, the 20 digits

1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0

have a hexadecimal equivalent of

A B 5 4 A 9 8 C E B 1 F 0 A D 2.

2.1.2 A-Key Verification

Procedure name:	
A_Key_Verify	
Inputs from calling process:	
A_KEY_DIGITS	from 6 to 26 decimal digits
ESN	32 bits
Inputs from internal stored data:	
	(internal definition only)
Outputs to calling process:	
A_KEY_VERIFIED	Boolean
Outputs to internal stored data:	
A-key	64 bits
SSD_A	64 bits (set to zero)
SSD_B	64 bits (set to zero)

This procedure verifies the A-key entered into a mobile station from a keypad.

The default value of the A-key when the mobile station is shipped from the factory will be all binary zeros. The value of the A-key is specified by the operator and is to be communicated to the subscriber according to the methods specified by each operator. A multiple NAM mobile station will require multiple A-keys, as well as multiple sets of the corresponding cryptovariables per A-key. See "User Interface for Authentication Key Entry," TSB50, for details of A-key entry into the mobile station.

While A-key digits are being entered from a keypad, the mobile station transmitter shall be disabled.

When the A-key digits are entered from a keypad, the number of digits entered is to be at least 6, and may be any number of digits up to and including 26 digits. In a case where the number of digits entered is less than 26, the leading most significant digits will be set equal to zero, in order to produce a 26-digit quantity called the "entry value".

The verification procedure checks the accuracy of the 26 decimal digit entry value. If the verification is successful, the 64-bit pattern determined by the first 20 digits of the entry value will be written to the subscriber's semi-permanent memory as the A-key. Furthermore, the SSD_A and the SSD_B will be set to zero. The return value A_KEY_VERIFIED is set to TRUE. In the case of a mismatch, A_KEY_VERIFIED is set to FALSE, and no internal data is updated.

The first decimal digit of the "entry value" is considered to be the most significant of the 20 decimal digits, followed in succession by the other nineteen. The twenty-first digit is the most significant of the check digits, followed in succession by the remaining five. For example, the 26 digits

1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 3 1 1 3 6

includes the 20-digit representation of the A-key (the same as the example in 2.1.1) and the checksum 131136 (20040 hexadecimal).

2.2 SSD Generation and Update

2.2.1 SSD Generation Procedure

Procedure name:	
SSD_Generation	
Inputs from calling process:	
RANDSSD	56 bits
ESN	32 bits
Inputs from internal stored data:	
AAV	8 bits
A-key	64 bits
Outputs to calling process:	
None.	
Outputs to internal stored data:	
SSD_A_NEW	64 bits
SSD_B_NEW	64 bits

This procedure performs the calculation of Shared Secret Data. The result is held in memory as SSD_A_NEW and SSD_B_NEW until the SSD_Update procedure (§2.2.2) is invoked.

2.2.2 SSD Update Procedure

Procedure name:

SSD_Update

Inputs from calling process:

None.

Inputs from internal stored data:

SSD_A_NEW	64 bits
SSD_B_NEW	64 bits

Outputs to calling process:

None.

Outputs to internal stored data:

SSD_A	64 bits
SSD_B	64 bits

This procedure copies the values SSD_A_NEW and SSD_B_NEW into the stored SSD_A and SSD_B.

The values SSD_A_NEW and SSD_B_NEW calculated by the SSD_Generation procedure (§2.2.1) should be validated prior to storing them permanently as SSD_A and SSD_B. The base station and the mobile station should exchange validation data sufficient to determine that the values of the Shared Secret Data are the same in both locations. When validation is completed successfully, the SSD_Update procedure is invoked, setting SSD_A to SSD_A_NEW and setting SSD_B to SSD_B_NEW.

2.3 Authentication Signature Calculation Procedure

Procedure name:	
Auth_Signature	
Inputs from calling process:	
RAND_CHALLENGE	32 bits
ESN	32 bits
AUTH_DATA	24 bits
SSD_AUTH	64 bits
SAVE_REGISTERS	Boolean
Inputs from internal stored data:	
	(internal definition only)
Outputs to calling process:	
AUTH_SIGNATURE	18 bits
Outputs to internal stored data:	
Saved register data	(internal definition only)

This procedure is used to calculate 18-bit signatures used for verifying the authenticity of messages used to request cellular system services, and for verifying Shared Secret Data.

For authentication of mobile station messages and for base station challenges of a mobile station, RAND_CHALLENGE should be selected by the authenticating entity (normally the HLR or VLR). RAND_CHALLENGE must be received by the mobile station executing this procedure. Results returned by the mobile station should include check data that can be used to verify that the RAND_CHALLENGE value used by the mobile station matches that used by the authenticating entity.

For mobile station challenges of a base station, as performed during the verification of Shared Secret Data, the mobile station should select RAND_CHALLENGE. The selected value of RAND_CHALLENGE must be received by the base station executing this procedure.

When this procedure is used to generate an authentication signature for a message, AUTH_DATA should include a part of the message to be authenticated. The contents should be chosen to minimize the possibility that other messages would produce the same authentication signature.

SSD_AUTH should be either SSD_A or SSD_A_NEW computed by the SSD_Generation procedure, or SSD_A as obtained from the HLR/AC.

If the calling process sets SAVE_REGISTERS to TRUE, the internal register data used in the authentication signature calculation are stored for use in computing the encryption key and voice privacy mask (see 2.4). If the calling process sets SAVE_REGISTERS to FALSE, the contents of internal storage are not changed.

2.4 Encryption Key and VPM Generation Procedure

Procedure name:

Key_VPM_Generation

Inputs from calling process:

None.

Inputs from internal stored data:

SSD_B	64 bits
saved register data (see §2.3)	(internal definition only)

Outputs to calling process:

None.

Outputs to internal stored data:

CMEAKEY	(internal definition only)
VPM	520 bits

This procedure computes the key for message encryption and the voice privacy mask. Prior to invoking this procedure, the authentication signature calculation procedure (§2.3) must have been invoked with SAVE_REGISTERS set to TRUE. This procedure must be invoked prior to execution of the encryption procedure (§2.5).

For this procedure, the saved internal variables to be used are those from the last authentication signature calculation for which the calling process set SAVE_REGISTERS to true. This should generally be the authentication calculation for the message that establishes the call for which encryption and/or voice privacy is to be invoked.

The VPM is not to be changed during a call.

2.5 CMEA Encryption/Decryption Procedure

Procedure name:	
Encrypt	
Inputs from calling process:	
msg_buf[n]	n*8 bits, n > 1
Inputs from internal stored data:	
CMEAKEY[0-7]	64 bits
Outputs to calling process:	
msg_buf[n]	n*8 bits
Outputs to internal stored data:	
None.	

This algorithm encrypts and decrypts messages that are of length n*8 bits, where n is the number of message bytes, n > 1. Decryption is performed in the same manner as encryption.

The message is first stored in an n-byte buffer called `msg_buf[]`, such that each byte is assigned to one "`msg_buf[]`" value. `msg_buf[]` will be encrypted and the encrypted values returned in the same storage buffer.

This process uses the CMEA key to produce enciphered messages via a unique CMEA algorithm. The CMEA key generation procedure is described in §2.4.

2.6 Wireless Residential Extension Procedures

This section describes detailed cryptographic procedures for cellular mobile telecommunications systems offering auxiliary services. These procedures are used to perform the security services of Authorization and Call Routing Equipment (ACRE), Personal Base (PB) and Mobile Station (MS) authentication.

2.6.1 WIKEY Generation

Procedure name:

WIKEY_Generation

Inputs from calling process:

MANUFACT_KEY	122 bits
PBID	30 bits

Inputs from internal stored data:

AAV	8 bits

Outputs to calling process:

None.

Outputs to internal stored data:

WIKEY	64 bits

This procedure is used to calculate the WIKEY value generated during the manufacturing process. This WIKEY value is stored in semi-permanent memory of the PB.

2.6.2 WIKEY Update Procedure

Procedure name:

WIKEY_Update

Inputs from calling process:

RAND_WIKEY	56 bits
PBID	30 bits

Inputs from internal stored data:

WIKEY	64 bits
AAV	8 bits

Outputs to calling process:

None.

Outputs to internal stored data:

WIKEY_NEW	64 bits

This procedure is used to calculate a new WIKEY value.

2.6.3 Wireline Interface Authentication Signature Calculation Procedure

Procedure name:

WI_Auth_Signature

Inputs from calling process:

RAND_CHALLENGE	32 bits
PBID	30 bits
ACRE_PHONE_NUMBER	24 bits

Inputs from internal stored data:

WIKEY	64 bits
AAV	8 bits

Outputs to calling process:

AUTH_SIGNATURE	18 bits

Outputs to internal stored data:

None.

This procedure is used to calculate 18-bit signatures used for verifying WIKEY values.

For authentication of an ACRE, RAND_CHALLENGE is received from the PB as RAND_ACRE.

For authentication of a PB, RAND_CHALLENGE is received from the ACRE as RAND_PB.

The ACRE_PHONE_NUMBER is 24 bits comprised of the least significant 24 bits of the ACRE's directory number (4 bits per digit). The digits 1 through 9 are represented by their 4-bit binary values (0001 - 1001). The digit 0 is represented by the binary value 1010. In a case where the number of ACRE directory number digits is less than six, the leading most significant bits of the ACRE_PHONE_NUMBER will be set equal to binary zero. For example, the ACRE directory number

(987) 654-3210

has a binary ACRE_PHONE_NUMBER

0101 0100 0011 0010 0001 1010.

The ACRE directory number

8695

has a binary ACRE_PHONE_NUMBER of

0000 0000 1000 0110 1001 0101.

2.6.4 Wireless Residential Extension Authentication Signature Calculation Procedure

Procedure name:

WRE_Auth_Signature

Inputs from calling process:

RAND_WRE	19 bits
ESN	32 bits
PBID	30 bits

Inputs from internal stored data:

WRE_KEY	64 bits
AAV	8 bits

Outputs to calling process:

AUTH_SIGNATURE	18 bits

Outputs to internal stored data:

None.

This procedure is used to calculate 18-bit signatures used for verifying a mobile station.

2.7 Cellular Data Encryption

2.7.1 Data Encryption Key Generation Procedure

Procedure name:	
DataKey_Generation	
Inputs from calling process:	
RAND	32 bits
Inputs from internal stored data:	
SSD_B	64 bits
Outputs to calling process:	
None.	
Outputs to internal stored data:	
DataKey	32 bits
L	256*8 bits

This algorithm generates DataKey, a period key used for generation of encryption masks for cellular data and Group 3 fax, and L, a table used in mask generation.

The DataKey_Generation procedure is executed at the beginning of each call, using the values of SSD_B and RAND in effect at the start of the call. The values of DataKey and L shall not change during a call.

The calculation of DataKey depends only on SSD_B, and may be computed and saved when SSD is updated. The calculation of L depends on RAND, and shall be performed at the beginning of each call.

2.7.2 Data Encryption Mask Generation Procedure

Procedure name:	
Data_Mask	
Inputs from calling process:	
HOOK	32 bits
mask	array pointer
len	integer
Inputs from internal stored data:	
SSD_B	64 bits
Outputs to calling process:	
mask	len*8 bits
Outputs to internal stored data:	
None.	

This algorithm generates an encryption mask of length len*8 bits, where n is the number of mask bytes.

Implementations using data encryption must comply with the following requirements. These requirements apply to all data encrypted during a call.

- The bits of the HOOK variable that change most frequently shall be placed in the least significant octet.
- No value of HOOK shall be used more than once during a call.
- Mask bytes produced using a value of HOOK shall be used to encrypt only one set of data bytes.
- Mask bytes generated using a value of HOOK shall not be used for encryption of data sent in different directions of transmission nor for data sent on different logical channels in any direction of transmission.