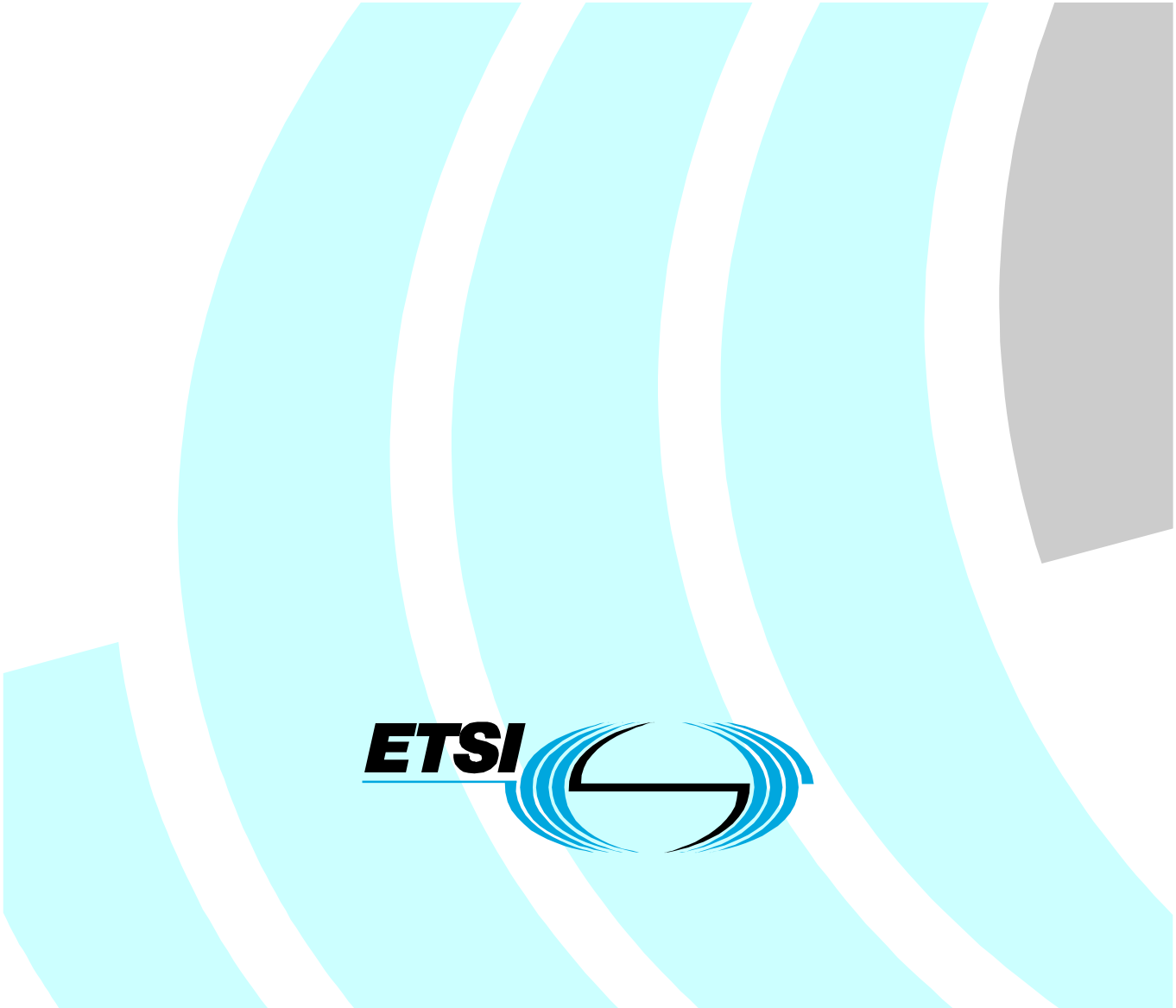
ETSI TS 102 165-2 V4.1.1 (2003-02)

Technical Specification

Telecommunications and Internet Protocol Harmonization Over Networks (TIPHON) Release 4; Protocol Framework Definition; Methods and Protocols for Security; Part 2: Counter Measures



Reference
DTS/TIPHON-08005-2R4

Keywords
IP, protocol, security, VoIP

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://webapp.etsi.org/IPR/home.asp).

All published ETSI deliverables shall include information which directs the reader to the above source of information.

Foreword

This Technical Specification (TS) has been produced by ETSI Project Telecommunications and Internet Protocol Harmonization Over Networks (TIPHON).

The present document is part 2 of a multi-part deliverable covering Methods and Protocols for security in TIPHON Release 4, as identified below:

Part 1: "Threat Analysis";

Part 2: "Counter Measures".

1 Scope

The present document defines by means of an information model, a functional entity behavioural model, and by validated SDL a model of the abstract behaviour of each service and service capability identified as being essential in TIPHON R4.

The present document defines by means of meta-protocol, algorithm boundary conditions, and guidance text the security countermeasures identified in TS 102 165-1 [1].

2 References

The following documents contain provisions which, through reference in this text, constitute provisions of the present document.

- References are either specific (identified by date of publication and/or edition number or version number) or non-specific.
- For a specific reference, subsequent revisions do not apply.
- For a non-specific reference, the latest version applies.

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

[1] ETSI TS 102 165-1: "Telecommunications and Internet Protocol Harmonization over Networks (TIPHON) Release 4; Protocol Framework Definition; Methods and Protocols for Security; Part 1: Threat Analysis".

[2] ETSI TR 101 877: "Telecommunications and Internet Protocol Harmonization Over Networks (TIPHON); Requirements Definition Study; Scope and Requirements for a Simple call".

[3] ETSI TS 101 878: "Telecommunications and Internet Protocol Harmonization Over Networks (TIPHON) Release 3; Service Capability Definition; Service Capabilities for a simple call".

[4] ITU-T Recommendation I.130: "Method for the characterization of telecommunication services supported by an ISDN and network capabilities of an ISDN".

[5] ISO/IEC 9798-2: "Information technology - Security techniques - Entity authentication - Part 2: Mechanisms using symmetric encipherment algorithms".

[6] ISO/IEC 9798-3: "Information technology - Security techniques - Entity authentication - Part 3: Entity authentication using a public key algorithm".

[7] ITU-T Recommendation Z.100: "Specification and description language (SDL)".

3 Definitions and abbreviations

3.1 Definitions

For the purposes of the present document, the terms and definitions given in TR 101 877 [2] and TS 101 878 [3] apply.

3.2 Abbreviations

For the purposes of the present document, the abbreviations given in TR 101 877 [2], TS 101 878 [3] and the following apply:

CK	Cipher Key
FE	Functional Entities
FFS	For Further Study
KSG	Key Stream Generator
KSS	Key Stream Segment
MSC	Message Sequence Chart
PDU	Protocol Data Unit
RSO	Random Seed
SDU	Service Data Unit
SpoA	Service point of Attachement
TpoA	Transport point of Attachement
TVP	Time Variant Parameter

4 Provision of counter-measures in TIPHON

4.1 Required security services

In TS 102 165-1 [1] the threats to a TIPHON system have been analysed and a set of recommended countermeasures identified that when implemented will reduce the overall risk to users of TIPHON systems. A subset of these countermeasures has been identified as essential to counter most threats.

The following security services are identified in [1] as required to minimize the risk to TIPHON to an acceptable level and are required to be standardized.

- A1 = Authentication of the terminal by the registrar (home of the user profile);
- A2 = Authentication of the registrar by the terminal;
- A3 = Authentication of the terminal by the Service point of Attachment (SpoA);
- A4 = Authentication of the SpoA by the terminal;
- A5 = Authentication of the SpoA by the registrar;
- A6 = Authentication of the registrar by the SpoA;
- C1 - C5 = Access control to services, to service data in databases, to data in terminals, and to the service provider's software and hardware, respectively;
- E1 = Confidentiality of user communication on the access interface;
- E2 = Confidentiality of signalling on the access interface;
- E3 = Confidentiality of signalling between SpoA entities;
- E6 = Confidentiality of TIPHON-id on signalling interfaces;
- E7 = Confidentiality of signalling between SpoA and Registrar;
- P1 = Bill limitations;
- P2 = Secure billing administration;
- P3 = Subscriber and terminal management;
- P9 = Security related reports to the service providers; and
- P10 = Secure subscription process.

The implementation method of these countermeasures can reduce the risk. However each countermeasure introduces new threats to the system by adding complexity, and different implementations of the same conceptual countermeasure may introduce different levels of risk. In some instances a specific countermeasure cannot be applied to a particular technology.

4.2 Location of standardization of security services

Not all security services are standardized in the present document. Table 1 identifies the location of standardized countermeasures.

Table 1: Where counter-measures are provided in TIPHON specifications

Security service	Form of standardization	Location
A1	Meta-protocol and algorithm specification	The present document, clause 6
A2	Meta-protocol and algorithm specification	The present document, clause 7
A3	Meta-protocol and algorithm specification	The present document, clause 8
A4	Meta-protocol and algorithm specification	The present document, clause 9
A5	Meta-protocol and algorithm specification	The present document, clause 10
A6	Meta-protocol and algorithm specification	The present document, clause 11
C1	Meta-protocol specification	TS 101 882-2
C2-C4	FFS	TBD
E1	Algorithm specification and application guideline	The present document, clause 12
E2	Algorithm specification and application guideline	The present document, clause 12
E3	Algorithm specification and application guideline	The present document, clause 12
E6	Application guideline	The present document, clause 12
E7	Algorithm specification and application guideline	The present document, clause 12
P1, P2, P3, P9, P10	Management framework requirements	TS 101 303

Services A1 and A5 are functionally identical and are described only once. In both cases the entity being authenticated has direct access to the shared secret, whereas the entity performing the authentication (the registrar) maintains the key to identity relationship through a third party (the authentication centre). The specific algorithms invoked in A1 and A5 may be different.

Services A2 and A6 are functionally identical and are described only once. The specific algorithms invoked in A2 and A6 may be different.

Services A3 and A4 are described in the present document as a single mutual authentication service A34.

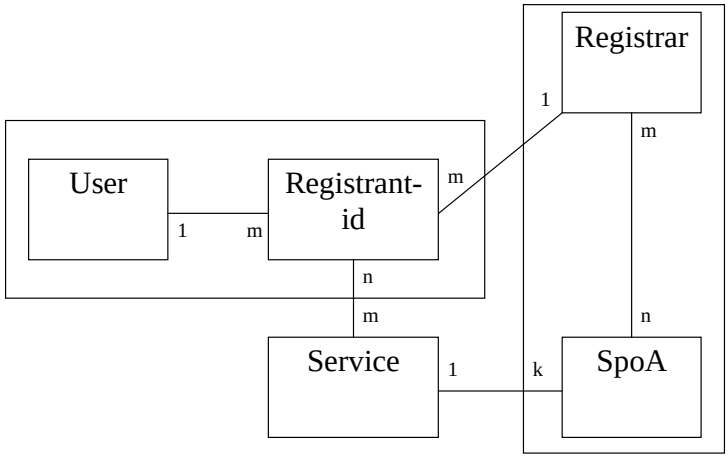
5 Authentication counter-measures

5.1 Introduction

5.1.1 Description

The primary purpose of the authentication service is to counter masquerade attacks. This may prevent attack on the system by determining that the user is legitimate, and may prevent an attack on the user by determining that the system is legitimate. The authentication services when successfully performed provide the first step in provision of access control to services (C1).

The authentication services are specified with respect to the entities and identities shown in figure 1.



NOTE 1: A single user may be associated with many registrant-ids
NOTE 2: A registrant-id shall be associated with only one user
NOTE 3: A registrant-id shall be associated with only one registrar
NOTE 4: A registrar may be associated with many registrant-ids
NOTE 5: A service may be associated with many SpoAs
NOTE 6: In any registration instance a service shall be associated with only one SpoA
NOTE 7: An SpoA shall be associated with only one Service
NOTE 8: A registrant-id may be associated with many Services

Figure 1: Ordinal relations in TIPHON

5.2 Keying policy in TIPHON

5.2.1 Release 4

In TIPHON Release 4 the policy towards keying will assume only symmetric keying methods. This method requires pre-arrangement between the authenticating entities but ensures that the authentication framework is able to provide a mapping to existing terminals which employ strong authentication (e.g. GSM, 3GPP-UMTS, DECT, TETRA).

In order to consider the authentication of the principal participants of TIPHON the following constraints on keying arise from the relations shown in figure 1.

- Registrant to Registrar shall use symmetric key authentication methods.
- Registrar to SpoA shall employ symmetric key authentication methods.
- Registrant to SpoA shall employ a symmetric session key to achieve authentication.

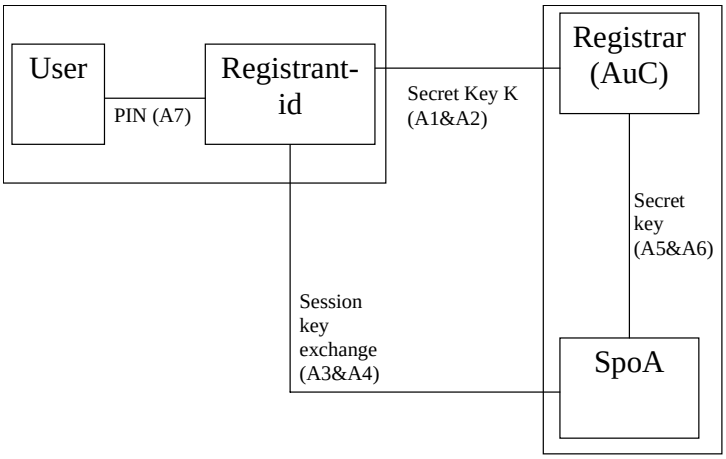


Figure 2: Key relationships in TIPHON

As a user may have many registrant-ids the user should be able to identify which to use and access to the registrant-id should be authenticated using a Personal Identification Number (PIN) or similar. The authentication service (A7) that provides for this protection of this identity is not described in this edition of the present document.

The symmetric keying authentication methods will be based upon the provisions described in ISO/IEC 9798-2 [5].

5.2.1.1 Review of service attributes

The authentication protocol should have the following properties:

- Bi-directional challenge-response type;
- Able to be initiated either explicitly or as part of the registration procedure;
- Able to be initiated by the terminal or the network;
- The recipient of the first authentication demand may instigate mutual authentication by use of a mutual authentication indicator, and by sending its challenge together with the response to the first challenge; and
- Where authentication is initiated as part of the registration the authentication timer TA shall always be less than or equal in value to any registration timer.

5.2.2 Release 5 and future

In TIPHON Release 5 and onwards the policy towards keying will encompass both symmetric and asymmetric keying methods.

The introduction of asymmetric methods may better counter the threats imposed by soft terminals and replace methods currently available that are based upon weak authentication methods (e.g. user-name and password). The public keying authentication methods will be based upon the provisions described in ISO/IEC 9798-3 [6].

6 A1 = Authentication of the terminal

6.1 Purpose

Service A1 offers strong authentication of a terminal to minimize the risk of masquerade of a terminal to the network.

6.2 Definition

The terminal shall contain a unique identity known to the registrar and authentication shall confirm this identity through proof of knowledge of a secret shared by the registrar and the terminal. This countermeasure is the corollary of A2.

6.3 Description

NOTE: The mechanism here is similar to the three pass authentication method defined in ISO/IEC 9798-2 [5].

The authentication method described is a symmetric secret key type using a challenge-response protocol. In this method one secret, the authentication key (K-A1), is shared by each of the authenticating parties, and there should be strictly two parties with knowledge of the secret. Authentication is achieved by the parties proving to each other knowledge of the shared secret. The authenticating parties are the authentication centre attached to the registrar and the terminal representing the user.

The following sequence of events illustrates the requirement:

1) The authentication centre shall generate a random *challenge* and send it to the terminal.

2) Both parties shall generate the intermediate result *Intermediate-Result* from algorithm A1-1 using the random *challenge* and K-A1 as inputs.

3) The terminal shall generate a random *seed* for the second stage of the authentication exchange.

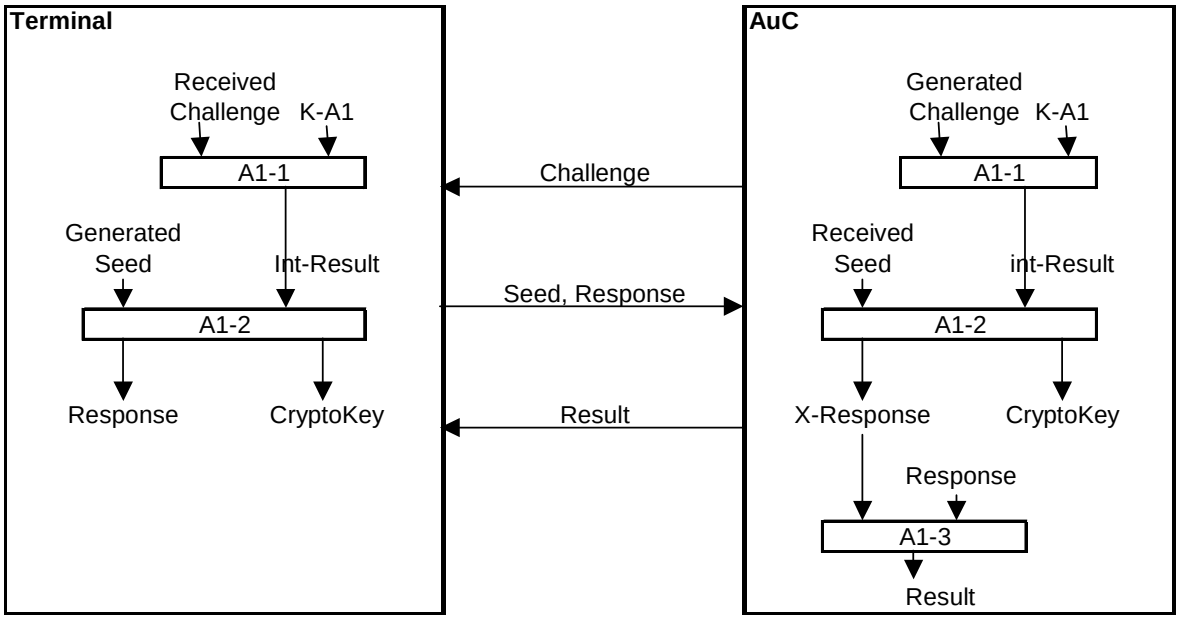
4) The terminal shall generate the cryptographic response *response* using algorithm A1-2 with inputs *seed* and *Intermediate-Result*.

5) The terminal shall send the random *seed* and the *response* to the authentication centre.

6) Upon receipt the authentication centre shall generate the expected cryptographic response using the received random *seed* and the pre-determined *Intermediate-Result* with algorithm A1-2. In addition A1-2 should generate an encryption key derived in this exchange for use in later confidentiality services.

7) The authentication centre shall compare the expected cryptographic response and the received *response*, if they are the same the terminal has proven knowledge of K-A1.

In addition to the above event sequence the protocol should be able to confirm randomness of the *challenge* and of the second stage *seed*.



NOTE: The split of the authentication algorithms allows K-A1 and algorithm A1-1 to be stored and maintained separately (remotely) from the location of A1-2.

Figure 3: Authentication service A1, algorithm invocations

6.4 Procedures

6.4.1 Provision/withdrawal

Authentication shall always be available.

6.4.2 Normal procedures

Authentication shall always be activated.

6.4.2.1 Invocation and operation

Authentication may be invoked on one or more of the following events:

- on registration to TIPHON;
- on change of physical point of attachment;
- on change of logical point of attachment;
- on demand by the user through some terminal function; and
- on demand by the authentication centre.

The registration and service attachment service defined in TS 101 882-2may be used as the master service for invocation of A1.

6.4.3 Exceptional procedures

6.4.3.1 Activation/deactivation/registration/interrogation

Not applicable.

6.4.3.2 Invocation and operation

If the expected response is not equal to the received response authentication is not proven and services A3, A4, A5 and A6 shall not be invoked. A network may reattempt service A1, however repeated failure may be considered as an attack on the algorithms and should be deterred by denying access to the initiator of the authentication.

6.5 Interactions with other TIPHON services

The authentication services are linked to the registration service and shall be operated in parallel to them.

The authentication services shall provide keying material for the confidentiality services E1, E2 and E7.

The authentication services shall provide the basis for the access control service C1.

6.6 Interworking considerations

The authentication algorithms used by each of the participant entities have to be matched.

6.7 Functional entity model

6.7.1 Description of model

Figure 4 shows the FEs and the relationships between them.

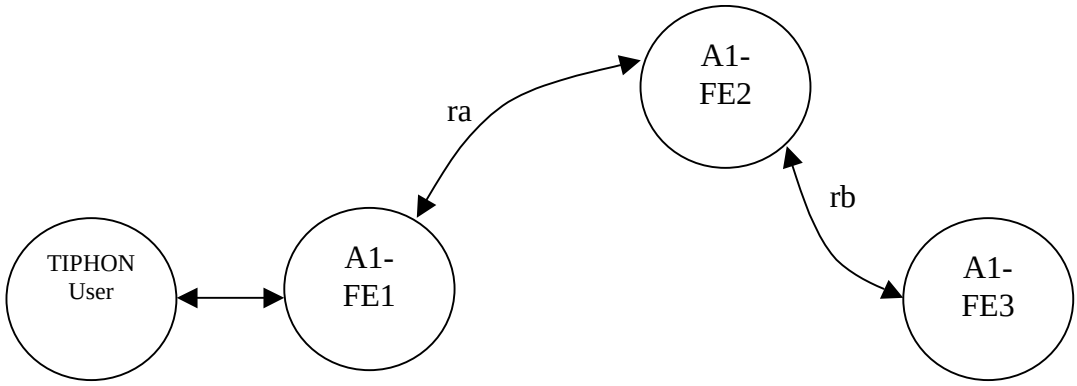


Figure 4: Relationships between functional entities for authentication service A1

Where:

TIPHON user	The entity representing the user who may be informed of the progress of an authentication exchange.
A1-FE1	The agent of the user being authenticated, contains algorithms A1-1 and A1-2.
A1-FE2	The local authorizing agent, contains algorithm A1-2 and A1-3.
A1-FE3	Holder of K, the shared secret upon which authentication is based, contains algorithm A1-1.

The information flows belonging to service A1 are:

- A1Auth (ra)
- A1AuthResult (ra)
- A1ChallengeRequest (rb)

6.8 Information flows

6.8.1 Definition of information flows

6.8.1.1 Relationship ra

6.8.1.1.1 A1Auth (req/ind/resp/conf)

A1Auth is a confirmed information flow that shall be sent across relation ra from FE2 to FE1 to indicate a request for authentication.

Information element	Value	Request	Confirmation
UserId		M	-
Challenge		M	-
Result	Success Key not recognized	-	M
Response		-	O (see note)
Seed		-	O (see note)
NOTE: Provided if result is success			

```
A1Auth-ri ::= SEQUENCE
{
    userId      AuthenticIdType,
    challenge   ChallengeType
}

A1Auth-rc ::= SEQUENCE
{
    result      AuthResultType,
    response    ResponseType OPTIONAL,
    seed        SeedType OPTIONAL
}
```

6.8.1.1.2 A1AuthResult

A1AuthResult is an unconfirmed information flow that shall be sent across relation ra from FE2 to FE1 to indicate the result of an authentication.

Information element	Value	Request	Confirmation
UserId		M	-
Result	Success Fail	M	-

```
A1AuthResult-ri ::= SEQUENCE
{
    userId      AuthenticIdType,
    result      BOOLEAN -- TRUE equals success, FALSE equals fail
}
```

6.8.1.2 Relationship rb

6.8.1.2.1 A1ChallengeRequest

A1ChallengeRequest is a confirmed information flow that shall be sent across relation rd from FE2 to FE4 to request the random generated challenge and intermediate result.

Information element	Value	Request	Confirmation
UserId		M	-
Result	Success UserId not known Key not available	-	M
GeneratedChallenge		-	O (see note)
Intermediate Result		-	O (see note)
NOTE: Provided if result is success.			

```
A1ChallengeRequest-ri ::= SEQUENCE
{
    userId      AuthenticIdType
}

A1ChallengeRequest-rc ::= SEQUENCE
{
    result              AuthResultType,
    generatedChallenge  ChallengeType OPTIONAL,
    intermediateResult  IntResType OPTIONAL
}
```

6.9 Information flow sequences

This clause specifies the information flow sequences for the authentication service.

NOTE 1: The information flow sequences are produced with a MSC editor; however, the scenarios are not MSCs but information flow sequences as defined in ITU-T Recommendation I.130 [4].

NOTE 2: In accordance with the ITU-T Recommendation I.130 [4] the invoking side is placed as the leftmost entity in the information flow sequences.

The step D for authentication shall provide signalling procedures in support of the information flow sequences specified in this clause. In addition, signalling procedures should be provided to cover other sequences arising from error situations, interactions with simple call, interactions with registration, different topologies, etc.

In the information flow sequences, authentication information flows are represented by arrows.

The following information flow sequences are intended as guidance in further development.

6.9.1 Information flows in A1

6.9.1.1 Normal behaviour

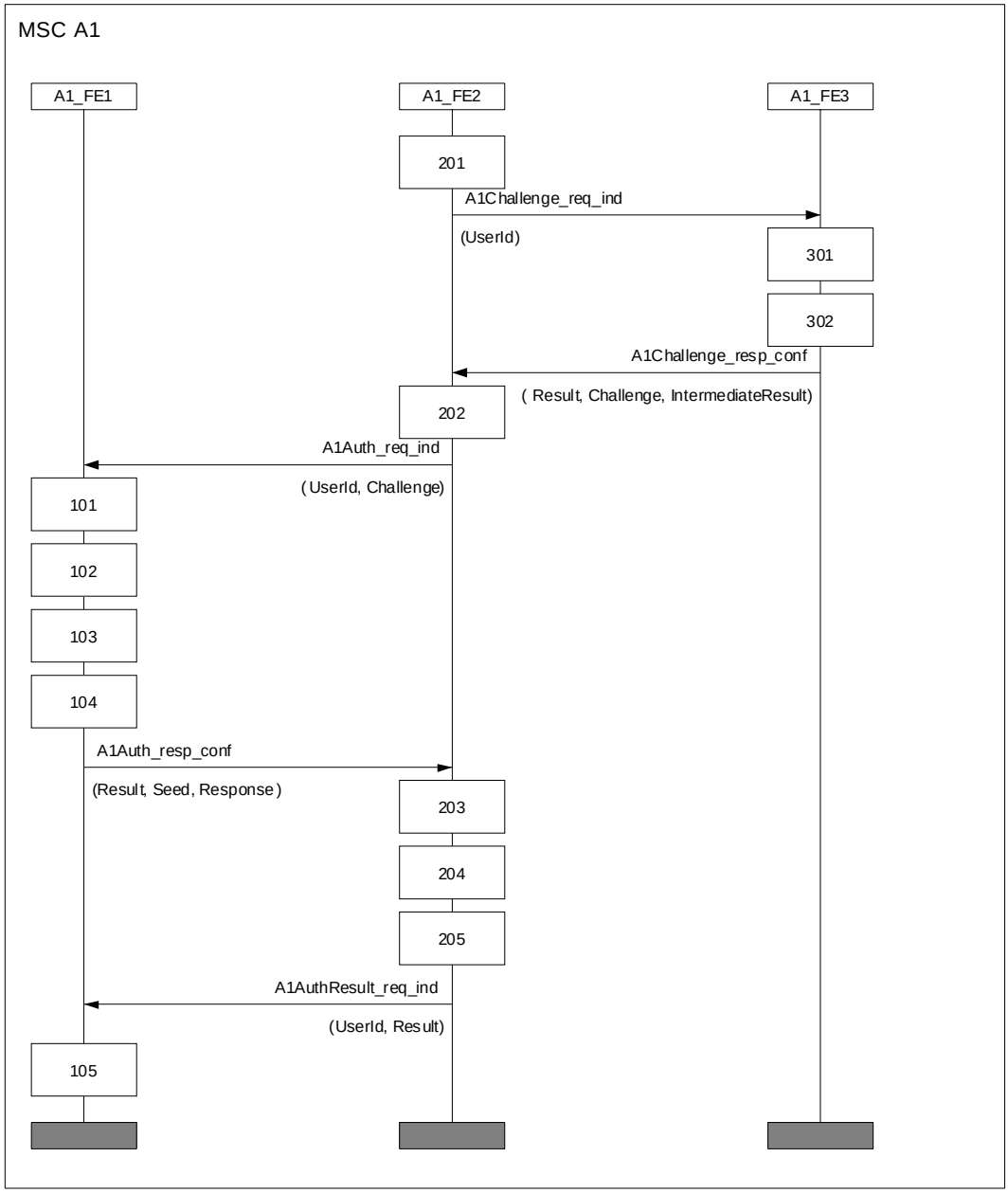


Figure 5: Authentication service A1 information flows

6.9.1.2 Exceptional behaviour

6.9.1.2.1 UserId not recognized by A1-FE3

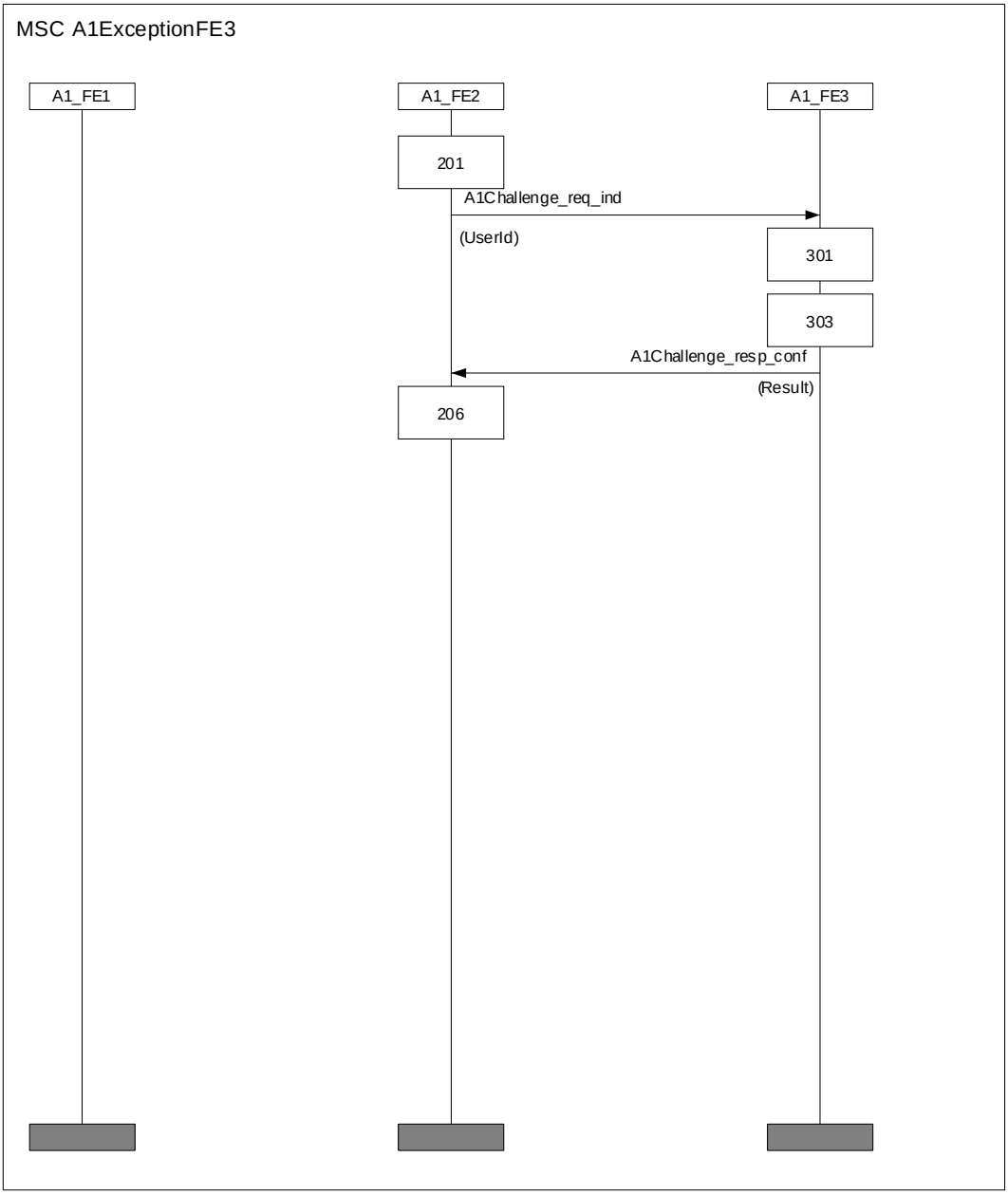


Figure 6: Information flow for exception where UserId not recognized by A1-FE3

6.9.1.2.2 Key is not available at A1-FE1

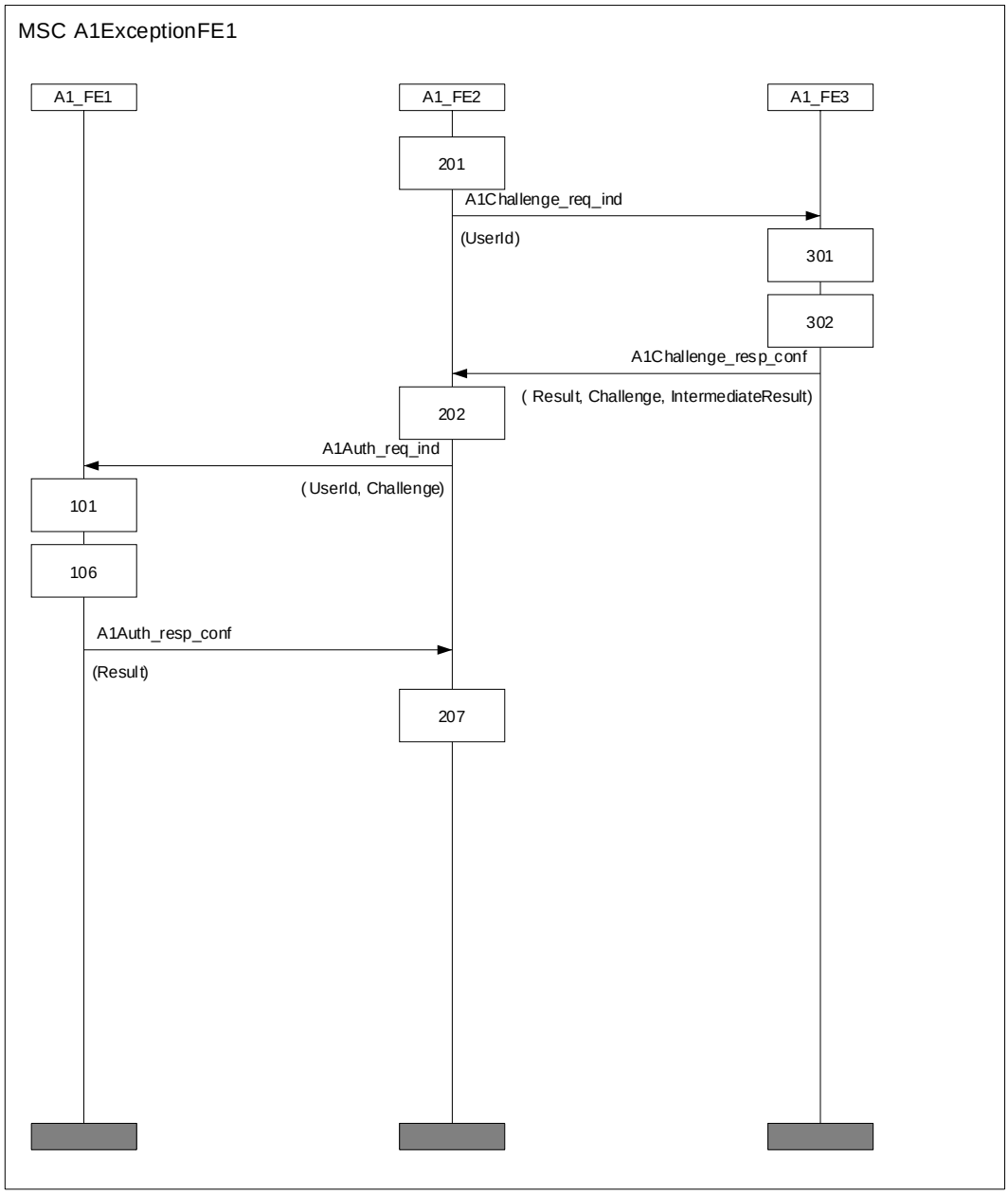


Figure 7: Information flow for exception where key is not available at A1-FE1

6.9.2 Functional entity actions

Throughout the descriptions of FE actions, the following conventions are used to identify information flows:

- An information flow is referred to as a "request" at the FE that sends it and as an "indication" at the FE that receives it.
- The corresponding confirmation is referred to as a "response" at the FE that sends it and as a "confirmation" at the FE that receives it.

The following FE actions shall occur at the points indicated in the information flow sequences of clause 6.9.1.

6.9.2.1 Actions of A1-FE1

101 A1-FE1 shall extract *challenge* from the received information flow and shall generate the intermediate result *Intermediate-Result* from algorithm A1-1 using the random *challenge* and K-A1 as inputs.

102 A1-FE1 shall generate a random *seed* for the second stage of the authentication exchange.

103 A1-FE1 shall generate the cryptographic response *response* using algorithm A1-2 with inputs *seed* and *Intermediate-Result*.

104 A1-FE1 shall send the random *seed* and the *response* to A1-FE2

105 A1-FE1 shall extract the result from the received information flow and if TRUE shall be authenticated, if FALSE shall not be authenticated and inhibit any further processing.

106 If K-A1 is not available *result* shall be set to "Key not found" and sent to A1-FE2.

6.9.2.2 Actions of A1-FE2

201 A1-FE2 shall request A1-FE3 to start the authentication process by sending the identity of the entity to be authenticated in an *A1Challenge* information flow.

202 On receipt of the response to action 201 A1-FE2 shall forward the received *Challenge* to A1-FE1 using the *A1Auth* information flow.

203 Upon receipt of the response to A1Auth A1-FE2 shall generate the expected cryptographic response (*X-Response*) using the received random *seed* and the previously received *Intermediate-Result* with algorithm A1-2.

204 A1-FE2 shall compare *X-Response* and the received *Response* using algorithm A1-3 generating *Result*. If *Result* is TRUE then A1-FE1 has been authenticated.

205 A1-FE2 shall send *Result* to A1-FE1 using information flow *A1AuthResult*.

206 If the result from A1-FE3 is not success further processing shall stop.

207 If the result from A1-FE1 is not success further processing shall stop.

6.9.2.3 Actions of A1-FE3

301 A1-FE3 shall identify K-A1 based on the identity received from A1-FE2 and shall generate a random *challenge* using it as input to algorithm A1-1 with K-A1 to generate *Intermediate-Result*.

302 A1-FE3 shall send *Intermediate-Result* and *challenge* to A1-FE2 using information flow *A1ChallengeRequest_resp_conf*.

303 If action 301 cannot identify K-A1 from the identity received FE3 shall set result to "Key not available", else is the userId received is not known FE3 shall set result to "UserID not known" and shall send *Result* to A1-FE2 using information flow *A1ChallengeRequest_resp_conf*.

6.9.3 Functional entity behaviour

The behaviour specified in this clause is intended to illustrate typical FE behaviour in terms of information flows sent and received.

The behaviour of each FE is shown using the Specification and Description Language (SDL) defined in ITU-T Recommendation Z.100 [7].

6.9.3.1 Behaviour of A1-FE1

The behaviour of A1-FE1 is shown in the SDL process diagram in figure 8.

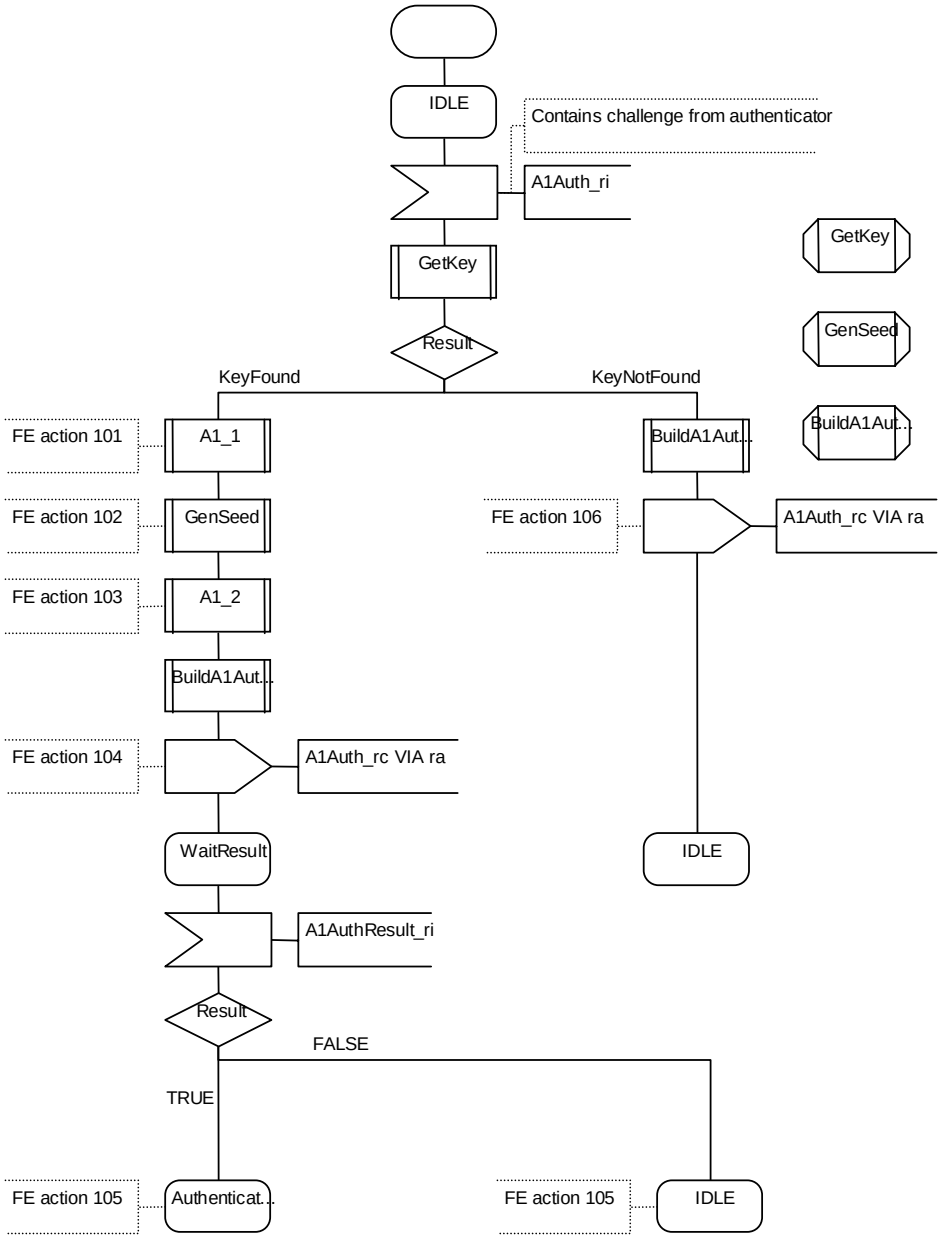


Figure 8: A1-FE1 process diagram

6.9.3.2 Behaviour of A1-FE2

The behaviour of A1-FE2 is shown in the SDL process diagram in figure 9.

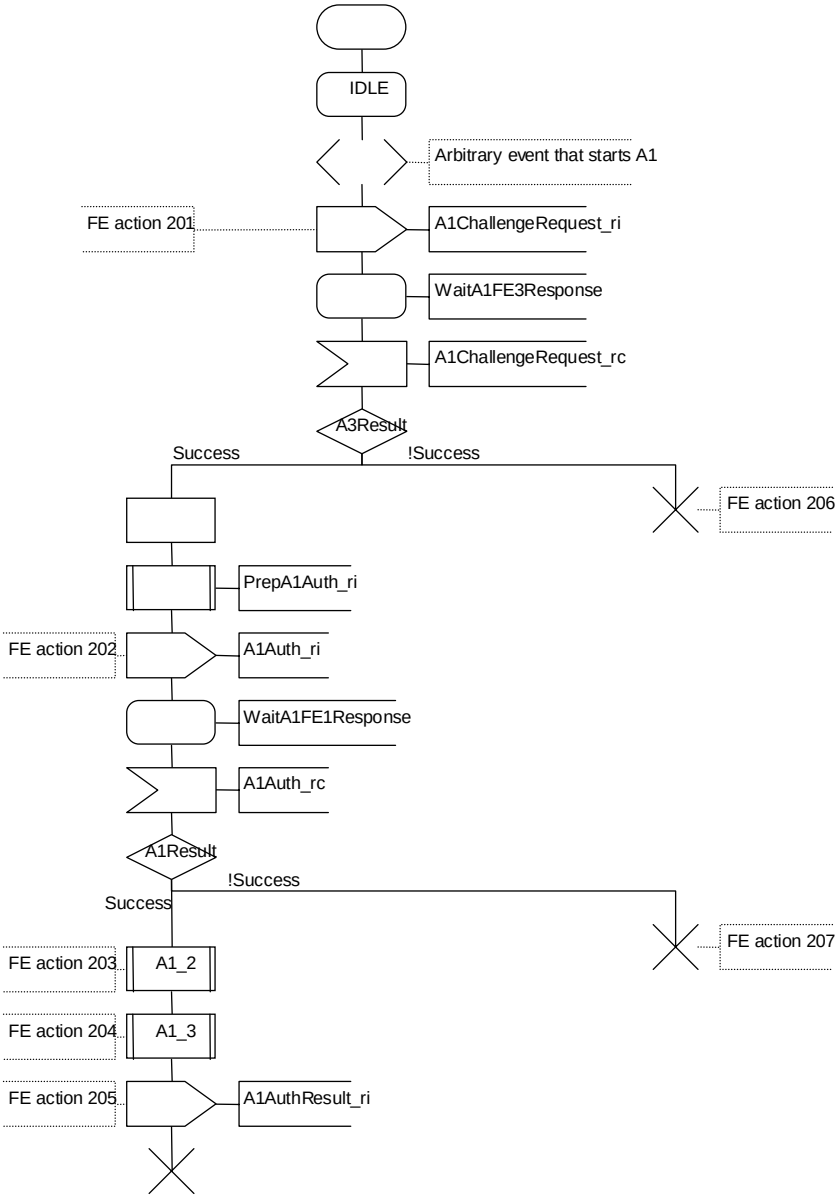


Figure 9: A1-FE2 process diagram

6.9.3.3 Behaviour of A1-FE3

The behaviour of A1-FE3 is shown in the SDL process diagram in figure 10.

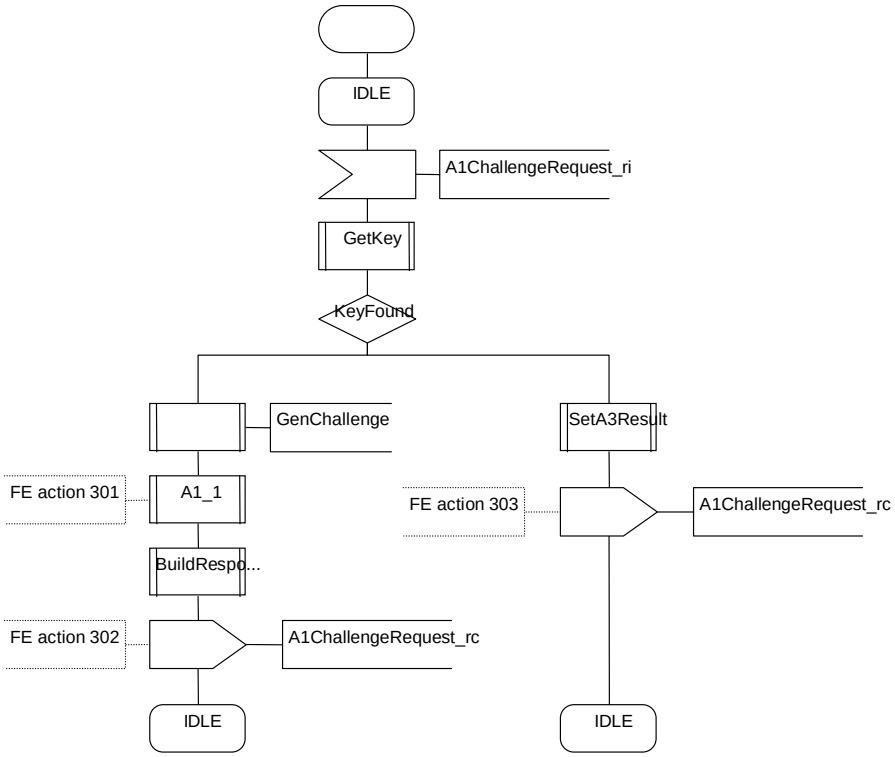


Figure 10: A1-FE1 process diagram

6.9.4 Allocation of functional entities to domains

The possible allocation of FEs to TIPHON is shown in table 2.

Table 2: Allocation of FEs to TIPHON domains

Scenario	A1-FE1	A1-FE2	A1-FE3
1	Terminal	Home Service provider	Home Service provider
2	Terminal	Visited Service provider	Home Service provider
3	Access network (terminal proxy)	Home Service provider	Home Service provider
4	Access network (terminal proxy)	Visited Service provider	Home Service provider

7 A2 = Authentication of the registrar

7.1 Purpose

Service A2 offers strong authentication of the network (registrar) to the terminal to minimize the risk of masquerade of a network to the terminal.

7.2 Definition

The terminal shall contain a unique identity that identifies the registrar and authentication shall confirm this identity through proof of knowledge of a secret shared by the registrar and the terminal. This countermeasure is the corollary of A1.

7.3 Description

NOTE: The mechanism here is similar to the three pass authentication method defined in ISO/IEC 9798-2 [5].

The authentication method described is a symmetric secret key type using a challenge-response protocol. In this method one secret, the authentication key (K-A2), is shared by each of the authenticating parties, and there should be strictly two parties with knowledge of the secret. Authentication is achieved by the parties proving to each other knowledge of the shared secret. The authenticating parties are the authentication centre attached to the registrar and the terminal representing the user.

The following sequence of events illustrates the protocol requirement:

1) The terminal shall generate a random *challenge* and send it to the authentication centre.

2) Both parties shall generate the intermediate result *Intermediate-Result* from algorithm A2-1 using the random *challenge* and K-A2 as inputs.

3) The authentication centre shall generate a random *seed* for the second stage of the authentication exchange.

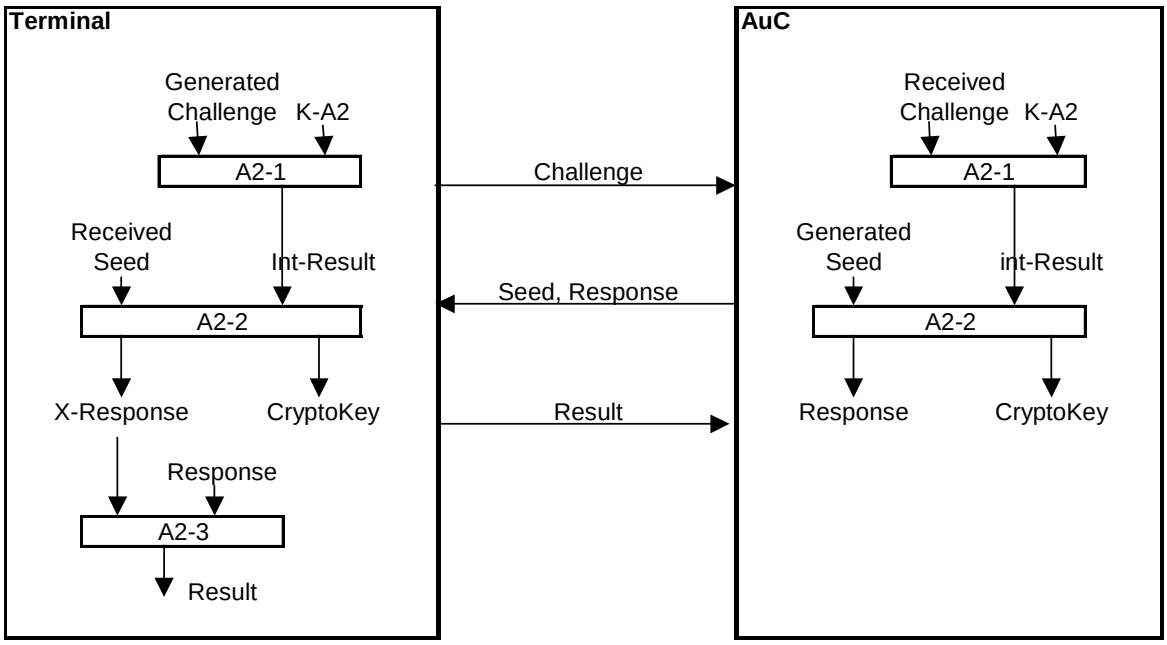
4) The authentication centre shall generate the cryptographic response *response* using algorithm A2-2 with inputs *seed* and *Intermediate-Result*.

5) The authentication centre shall send the random *seed* and the *response* to the terminal.

6) Upon receipt the terminal shall generate the expected cryptographic response using the received random *seed* and the pre-determined *Intermediate-Result* with algorithm A2-2. In addition A2-2 should generate an encryption key derived in this exchange for use in later confidentiality services.

7) The terminal shall compare the expected cryptographic response and the received *response*, if they are the same the authentication centre has proven knowledge of K-A2.

In addition to the above event sequence the protocol should be able to confirm randomness of the *challenge* and of the second stage *seed*.



NOTE: The split of the authentication algorithms allows K-A2 and algorithm A2-1 to be stored and maintained separately (remotely) from the location of A2-2.

Figure 11: Authentication service A2, algorithm invocations

7.4 Procedures

7.4.1 Provision/withdrawal

Authentication shall always be available.

7.4.2 Normal procedures

Authentication shall always be activated.

7.4.2.1 Invocation and operation

Authentication may be invoked on one or more of the following events:

- on registration to TIPHON™;
- on change of physical point of attachment;
- on change of logical point of attachment;
- on demand by the user through some terminal function; and
- on demand by the authentication centre.

7.4.3 Exceptional procedures

7.4.3.1 Activation/deactivation/registration/interrogation

Not applicable.

7.4.3.2 Invocation and operation

If the expected response is not equal to the received response authentication is not proven and services A3, A4, A5 and A6 shall not be invoked. A terminal may reattempt service A2, however repeated failure may be considered as an attack on the algorithms and should be deterred by denying access to the initiator of the authentication.

7.5 Interactions with other TIPHON services

The authentication services are linked to the registration service and shall be operated in parallel to them.

The authentication services shall provide keying material for the confidentiality services E1, E2 and E7.

The authentication services shall provide the basis for the access control service C1.

7.6 Interworking considerations

The authentication algorithms used by each of the participant entities have to be matched.

7.7 Functional entity model

7.7.1 Description of model

Figure 12 shows the FEs and the relationships between them.

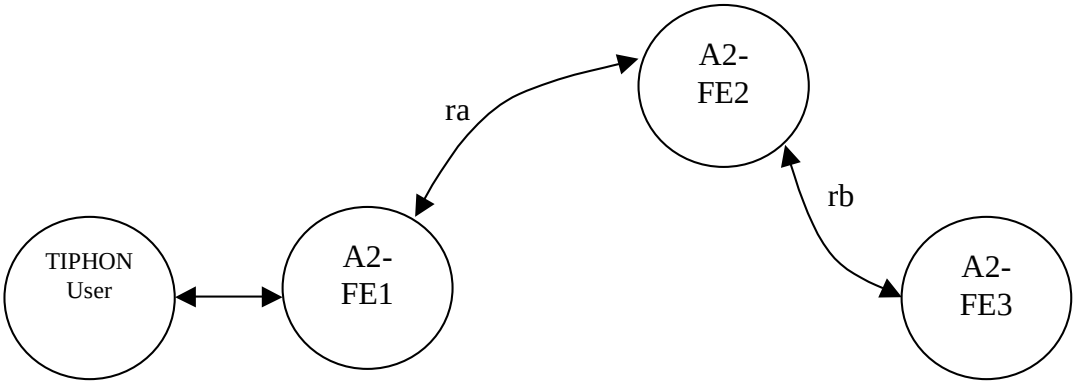


Figure 12: Relationships between functional entities for authentication service A2

Where:

TIPHON user	The entity representing the user who may be informed of the progress of an authentication exchange.
A2-FE1	The agent of the user being authenticated, contains algorithms A2-1, A2-2 and A2-3.
A2-FE2	The local authorizing agent, contains algorithms A2-2 and A2-3.
A2-FE3	Holder of K, the shared secret upon which authentication is based, contains algorithm A2-1.

The information flows belonging to service A2 are:

- A2Auth (ra)
- A2AuthResult (ra)
- A2IntermediateResultRequest (rb)

7.8 Information flows

7.8.1 Definition of information flows

7.8.1.1 Relationship ra

7.8.1.1.1 A2Auth

A2Auth is a confirmed information flow that shall be sent across relation ra from FE1 to FE2 to indicate a request for authentication.

Information element	Value	Request	Confirmation
UserId		M	-
Challenge		M	-
Response		-	M
Seed		-	M

```
A2Auth_req_ind ::= SEQUENCE
{
    userId      AuthenticIdType,
    challenge   ChallengeType
}

A2Auth_resp_conf ::= SEQUENCE
{
    response    ResponseType,
    seed        SeedType
}
```

7.8.1.1.2 A2AuthResult

A2AuthResult is an unconfirmed information flow that shall be sent across relation ra from FE1 to FE2 to indicate the result of an authentication.

Information element	Value	Request	Confirmation
UserId		M	-
Result	Success Fail	M	-

```
A2AuthResult_req_ind ::= SEQUENCE
{
    userId      AuthenticIdType,
    result      BOOLEAN -- TRUE equals success, FALSE equals fail
}
```

7.9 Information flow sequences

This clause specifies the information flow sequences for the authentication service.

NOTE 1: The information flow sequences are produced with a MSC editor; however, the scenarios are not MSCs but information flow sequences as defined in ITU-T Recommendation I.130 [4].

NOTE 2: In accordance with the ITU-T Recommendation I.130 [4] the invoking side is placed as the leftmost entity in the information flow sequences.

The step D for authentication shall provide signalling procedures in support of the information flow sequences specified in this clause. In addition, signalling procedures should be provided to cover other sequences arising from error situations, interactions with simple call, interactions with registration, different topologies, etc.

In the information flow sequences, authentication information flows are represented by arrows.

The following information flow sequences are intended as guidance in further development.

7.9.1 Information flow in A2, normal behaviour

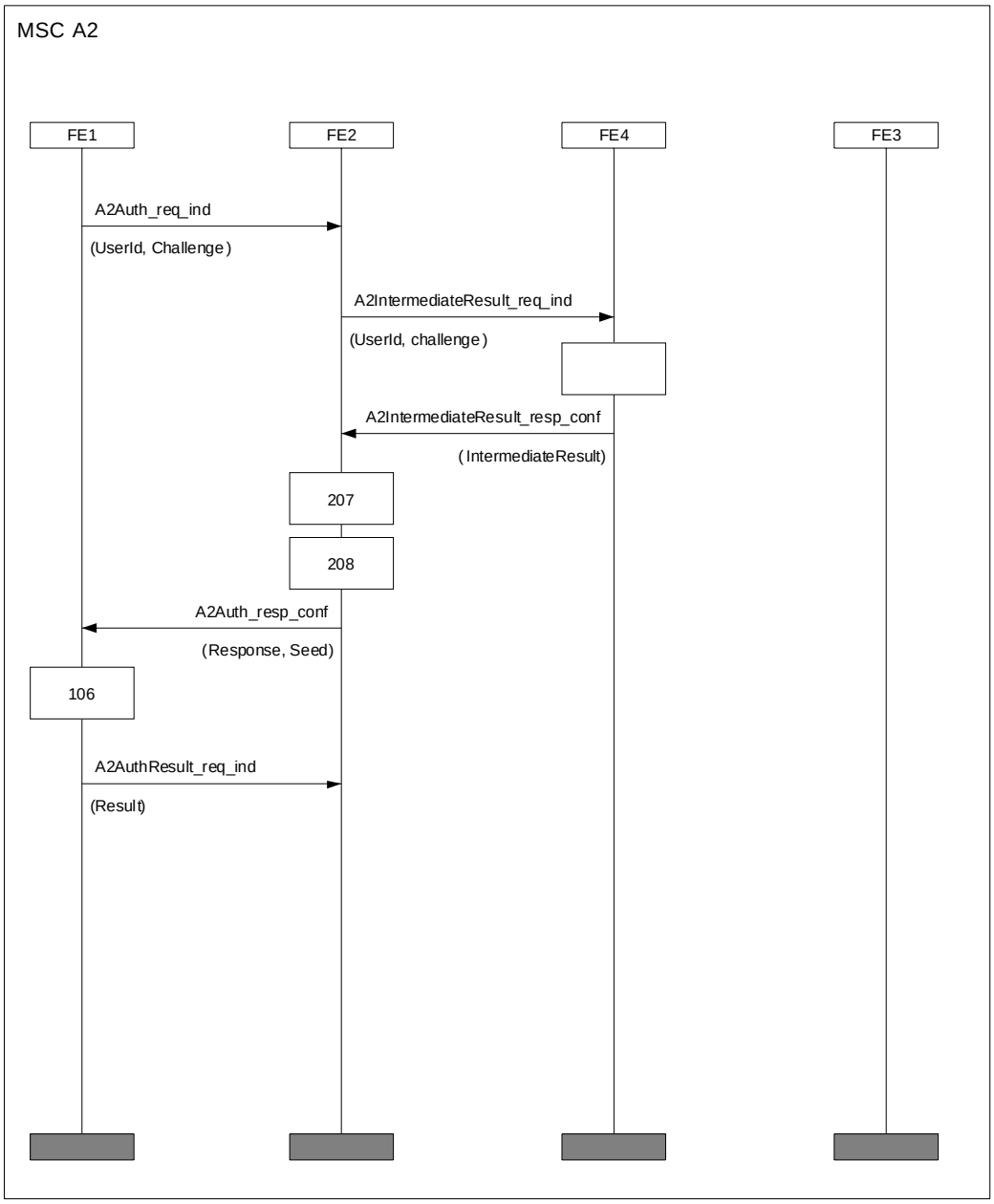


Figure 13: Authentication service A2 information flows

7.9.2 Functional entity actions

Throughout the descriptions of FE actions, the following conventions are used to identify information flows:

- An information flow is referred to as a "request" at the FE that sends it and as an "indication" at the FE that receives it.
- The corresponding confirmation is referred to as a "response" at the FE that sends it and as a "confirmation" at the FE that receives it.

The following FE actions shall occur at the points indicated in the information flow sequences of clause 7.9.1.

7.9.2.1 Actions of A2-FE1

101 A2-FE1 shall generate a random *challenge* and send it to A2-FE2.

102 A2-FE1 generate the intermediate result *Intermediate-Result* from algorithm A2-1 using the random *challenge* and K-A2 as inputs.

103 Upon receipt the terminal shall generate the expected cryptographic response using the received random *seed* and the pre-determined *Intermediate-Result* with algorithm A2-2. In addition A2-2 should generate an encryption key derived in this exchange for use in later confidentiality services.

104 The terminal shall compare the expected cryptographic response and the received *response*, if they are the same the authentication centre has proven knowledge of K-A2.

7.9.2.2 Actions of A2-FE2

201 A2-FE2 shall generate the intermediate result *Intermediate-Result* from algorithm A2-1 using the random *challenge* and K-A2 as inputs.

202 A2-FE2 shall generate a random *seed* for the second stage of the authentication exchange.

203 A2-FE2 shall generate the cryptographic response *response* using algorithm A2-2 with inputs *seed* and *Intermediate-Result*.

204 A2-FE2 shall send the random *seed* and the *response* to A2-FE1.

7.9.2.3 Actions of A2-FE3

301 A2-FE3 shall identify K-A1 based on the identity received from A2-FE2 and using the *challenge* received from A2-FE2 as input to algorithm A1-1 with K-A1 shall generate *Intermediate-Result*.

302 A2-FE3 shall send *Intermediate-Result* to A2-FE1 using information flow *A2ChallengeRequest_resp_conf*.

7.9.3 Allocation of functional entities to domains

The possible allocation of FEs to TIPHON is shown in table 3.

Table 3: Allocation of FEs to TIPHON domains

Scenario	A2-FE1	A2-FE2	A2-FE3
1	Terminal	Home Service provider	Home Service provider
2	Terminal	Visited Service provider	Home Service provider
3	Access network (terminal proxy)	Home Service provider	Home Service provider
4	Access network (terminal proxy)	Visited Service provider	Home Service provider

8 A3 and A4, A34 = Mutual authentication terminal and SpoA

8.1 Purpose

To prevent masquerade of a terminal to a SpoA by providing authentication through a mutual trusted third party (the registrar).

NOTE: The mechanism defined in this clause is optimized for symmetric keying methods but may employ asymmetric keying methods with no change in the affect of the countermeasure.

8.2 Definition

In order to offer service to the terminal the terminal shall be authenticated by the SpoA. This authentication shall be based upon information given by the registrar as a result of services A1, A2, A5 and A6 and from the access control service C1 described in TS 101 882-2.

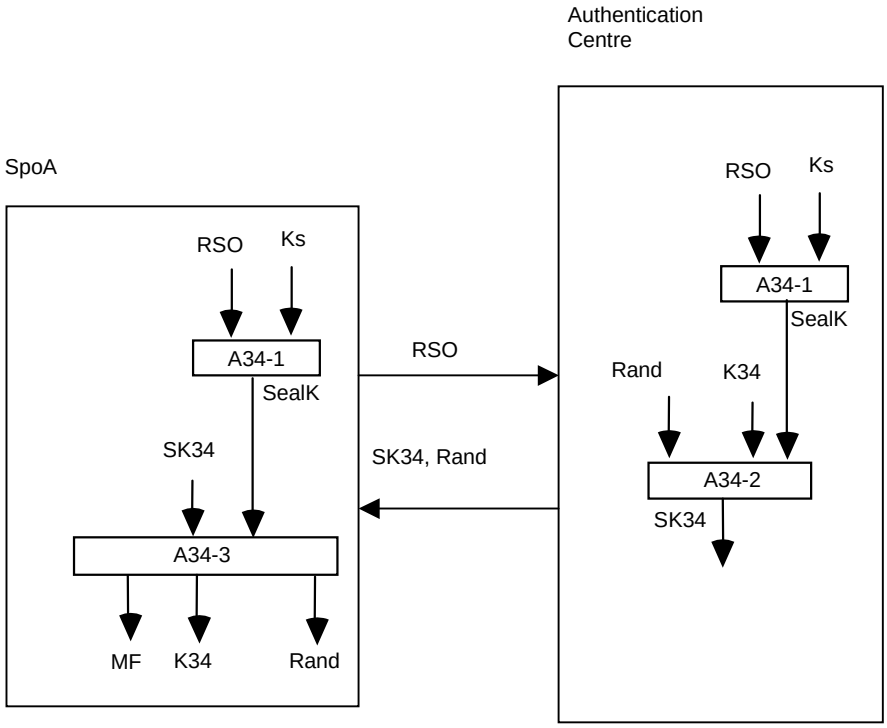
8.3 Description

NOTE: The mechanism here is similar to the 5 pass mutual authentication method described in clause 6.2 of ISO/IEC 9798-2 [5].

There can be no explicit challenge response protocol between the terminal and the SpoA as there is no shared secret knowledge.

The registrar acts as a trusted party to both the SpoA and to the terminal. The registrar shall generate a random session key K34 and distribute this to the SpoA and terminal in a secure manner.

The registrar shall prepare two packages containing K34 sealed by respectively the secret key K associated with the terminal, and the secret key Ks associated with the SpoA. On receipt of the appropriate package the SpoA and the Terminal shall be able to recover the session key K34. Authentication of the SpoA and the terminal is provided by using the received session key (K34) to encrypt random data generated by each party.

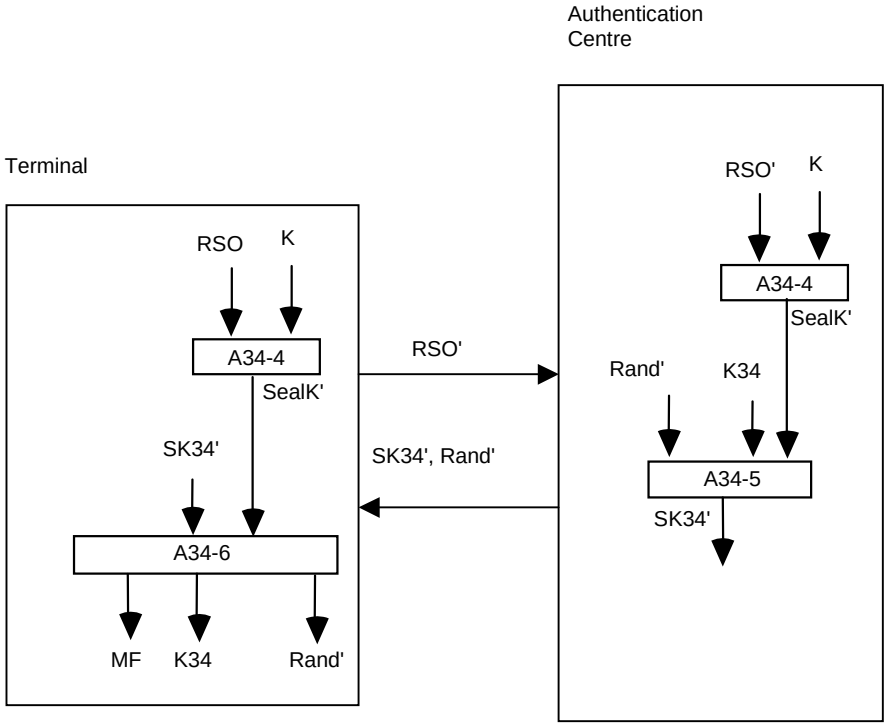


NOTE: Ks, as input to A34-1, should be equal to K-A2.

Figure 14: Distribution of a session key to SpoA

The first pass shown here uses algorithm A34-1 to generate a sealing key (SealK) from Ks and a Random Seed (RSO). The session key, K34, is sealed using algorithm A34-2 with inputs K34, SealK and a random number (Rand). The sealed key and the random number are sent to the SpoA which generates SealK using A34-1 and then unseals K34 using algorithm A34-3.

The first pass will conventionally be assigned to the authentication centre holding Ks, with the session key being generated locally at the registrar. This mitigates against an attack on the authentication centre being able to determine the session key in use, and also mitigates against an attack on the registrar being able to determine the shared secrets of either Terminal or SpoA.



NOTE: K, as input to A34-4, should be equal to K-A1.

Figure 15: Distribution of a session key to terminal

The first pass shown here uses algorithm A34-4 to generate a sealing key (SealK') from K and a Random Seed (RSO'). The session key, K34, is sealed using algorithm A34-5 with inputs K34, SealK' and a random number (Rand'). The sealed key and the random number are sent to the terminal which generates SealK' using A34-4 and then unseals K34 using algorithm A34-6.

8.3.1 Overall authentication exchange

The authentication exchange is summarized here:

1) The terminal creates a random number RSO' and sends it to the SpoA as a challenge.

2) The SpoA creates its own random number RSO and sends it to the registrar along with RSO' and the identity of the terminal.

3) The registrar requests the sealing keys from the authentication centre by sending RSO, RSO', and the identities of the SpoA and Terminal to the authentication centre.

4) The authentication centre runs algorithms A34-1 and A34-4 to generate SealK and SealK' respectively which are returned to the registrar.

5) The registrar creates the cryptographic token T34-A and sends it to the SpoA.

6) The SpoA recovers the session key K_{34} from token T34-A and creates token T34-B which it sends to the terminal.

7) The terminal recovers session key K_{34} from token T34-B and creates token T34-C which it sends to the SpoA.

8.3.1.1 Token definitions

The following cryptographic tokens are defined:

$T34\text{-}A = eK_S(RSO||K_{34}||Terminal\text{-}id) \,||\, eK_T(RSO'||K_{34}||SpoA\text{-}id)$

$T34\text{-}B = eK_T(RSO'||K_{34}||SpoA\text{-}id) \,||\, eK_{34}(RSO||RSO')$

$T34\text{-}C = eK_{34}(RSO||RSO'||ClientAuthorizationToken)$

NOTE: In each case the cryptographic tokens can be extended using arbitrary text fields but for clarity this is not shown.

Algorithm A34-4 in conjunction with A34-5, and algorithm A34-1 in conjunction with A34-2, provide the two halves of T34-A which allows the token definitions to be rewritten as:

$T34\text{-}A = Output1(A34\text{-}2) \,||\, Output1(A34\text{-}5)$

$T34\text{-}B = Output1(A34\text{-}5) \,||\, eK_{34}(RSO||RSO')$

In addition an algorithm, A34-7, is defined to provide CryptoElement:

$CryptoElement = eK34(RSO||RSO')$

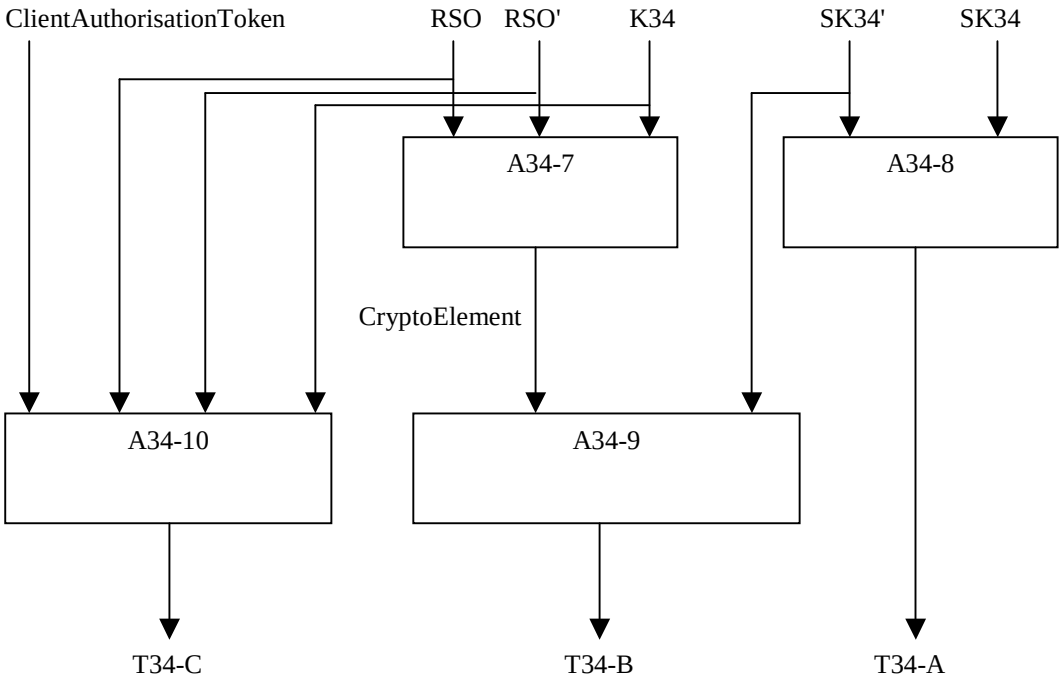


Figure 16: Algorithms to derive tokens T34-A, T34-B and T34-C

Allowing T34-B to be rewritten as:

$T34\text{-}B = Output1(A34\text{-}5) \,||\, Output1(A34\text{-}7)$

The CryptoElement is created by the SpoA and contains data from the terminal and random data from the SpoA. These are encrypted with the shared key K34. On receipt of CryptoElement at the terminal the two data elements are recovered. If the random data from the terminal matches that already given then the terminal is assured that the SpoA has the data and has K34 so the terminal has been able to authenticate that the SpoA is known to the trusted party.

8.4 Procedures

8.4.1 Provision/withdrawal

Authentication shall always be available.

8.4.2 Normal procedures

Authentication shall always be activated.

8.4.2.1 Invocation and operation

Authentication may be invoked on one or more of the following events:

- on registration to TIPHON™;
- on change of physical point of attachment;
- on change of logical point of attachment;
- on demand by the user through some terminal function; and
- on demand by the authentication centre.

8.4.3 Exceptional procedures

8.4.3.1 Activation/deactivation/registration/interrogation

Not applicable.

8.4.3.2 Invocation and operation

See clause 8.3.

8.5 Interactions with other TIPHON services

The authentication services are linked to the registration service and shall be operated in parallel to them.

The authentication services shall provide keying material for the confidentiality services E1, E2 and E7.

The authentication services shall provide the basis for the access control service C1.

8.6 Interworking considerations

The authentication algorithms used by each of the participant entities have to be matched.

8.7 Functional entity model

8.7.1 Description of model

Figure 16 shows the FEs and the relationships between them.

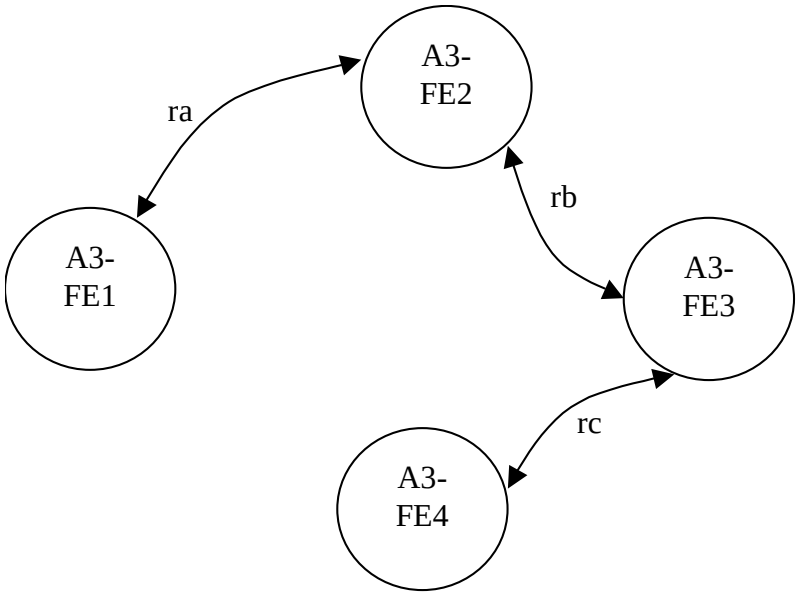


Figure 17: Relationships between functional entities for authentication service A34

Where:

A34-FE1	Agent at terminal that initiates authentication, contains algorithms A34-4 and A34-6.
A34-FE2	Agent representing the SpoA holds algorithms A34-1 and A34-3.
A34-FE3	Agent representing the Registrar (Trusted Third Party), contains algorithms A34-2 and A34-5.
A34-FE4	Agent representing the Authentication Centre, contains algorithms A34-1 and A34-4.

The information flows between A34-FE2, A34-FE3 and A34-FE4 are required to supply A34-FE2 with the session key to be shared with A34-FE1.

The information flows belonging to service A34 are:

- A34UserToSpoAAuth (ra)
- A34UserToSpoAAuthorizedAttach (ra)
- A34SpoAWithUserAuth (rb)
- A34SealingKeyRequest (rc)

8.8 Information flows

8.8.1 Definition of information flows

8.8.1.1 Relationship ra

8.8.1.1.1 A34UserToSpoAAuth

A34UserToSpoAAuth is a confirmed information flow sent across relation ra from FE1 to FE2 to initiate the exchange of session key K34.

Information element	Value	Request	Confirmation
UserId		M	-
RSO' (AccessChallenge)		M	-
T34-B		-	M

8.8.1.1.2 A34UserToSpoAAuthorizedAttach

A34UserToSpoAAuthorizedAttach is an unconfirmed information flow sent across relation ra from FE1 to FE2 to finalize the SpoA/Terminal authentication exchange.

Information element	Value	Request	Confirmation
T34-C		M	-

8.8.1.2 Relationship rb

8.8.1.2.1 A34SpoAWithUserAuth

A34SpoAWithUserAuth is a confirmed information flow sent across relation rb from FE2 to FE3 to initialize the SpoA/Terminal authentication exchange.

Information element	Value	Request	Confirmation
UserId		M	-
SpoA-id		M	
RSO' (AccessChallenge)		M	-
RSO (SpoA Challenge)		M	-
T34-A			M

8.8.1.3 Relationship rc

8.8.1.3.1 A34SealingKeyRequest

A34SealingKeyRequest is a confirmed information flow sent across relation rc from FE3 to FE4 to retrieve the keys used to seal the session key prior the session key being sent to each of the SpoA and Terminal.

Information element	Value	Request	Confirmation
UserId		M	M
SpoA-id		M	M
RSO		M	-
RSO'		M	-
SealK		-	M
SealK'		-	M

8.9 Information flow sequences

This clause specifies the information flow sequences for the authentication service.

NOTE 1: The information flow sequences are produced with a MSC editor; however, the scenarios are not MSCs but information flow sequences as defined in ITU-T Recommendation I.130 [4].

NOTE 2: In accordance with the ITU-T Recommendation I.130 [4] the invoking side is placed as the leftmost entity in the information flow sequences.

The step D for authentication shall provide signalling procedures in support of the information flow sequences specified in this clause. In addition, signalling procedures should be provided to cover other sequences arising from error situations, interactions with simple call, interactions with registration, different topologies, etc.

In the information flow sequences, authentication information flows are represented by arrows.

The following information flow sequences are intended as guidance in further development.

8.9.1 Information flow in A3, normal behaviour

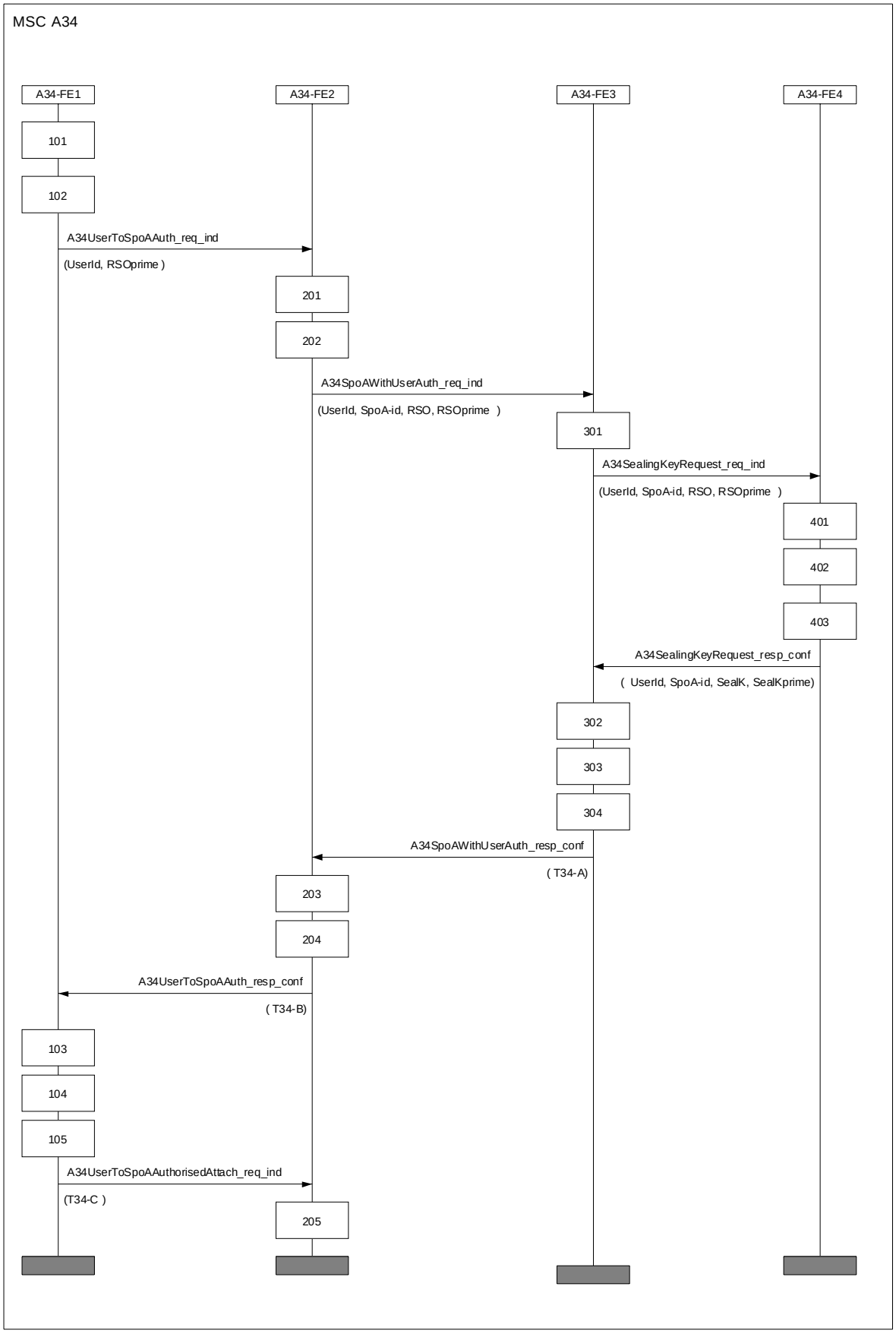


Figure 18: Authentication service A34 information flows

8.9.2 Functional entity actions

Throughout the descriptions of FE actions, the following conventions are used to identify information flows:

- An information flow is referred to as a "request" at the FE that sends it and as an "indication" at the FE that receives it.
- The corresponding confirmation is referred to as a "response" at the FE that sends it and as a "confirmation" at the FE that receives it.

The following FE actions shall occur at the points indicated in the information flow sequences of clause 8.9.1.

8.9.2.1 Actions of A34-FE1

101 A34-FE1 shall generate a random seed, RSO'.

102 Using A34UserToSpoAAuth-request A34-FE1 shall send RSO' as a challenge to A34-FE2 to initiate the authentication.

103 Using RSO' and K as input to A34-4 A34-FE1 shall generate SealK'.

104 Using SealK' from action 103 and SK34' (recovered from T34-B) from A34UserToSpoAAuth_resp_conf as inputs to A34-6 A34-FE1 shall recover K34 and Rand'. The Rand' recovered with A34-6 and Rand' received in A34UserToSpoAAuth_resp_conf shall be compared and if no errors exist they should be found equal.

105 Generates token T34-C using algorithm A34-9 and sends it to A34-FE2 using the A34UserToSpoAAuthorizedAttach-req-ind information flow.

8.9.2.2 Actions of A34-FE2

201 On receipt of A34UserToSpoAAuth-req-ind A34-FE2 shall extend the authentication by generating its own challenge RSO.

202 Using A34SpoAWithUserAuth-req-ind A34-FE2 shall send the challenges to A34-FE3.

203 On receipt of A34SpoAWithUserAuth-resp-conf A34-FE2 extracts A34 from T34-A using algorithm A34-3.

204 Using algorithm A34-8 and the FE1 part of T34-A A34-FE2 constructs token T34-B and sends it to A34-FE1 using the A34UserToSpoAAuth-resp-conf information flow.

205 On receipt of T34-C from the A34UserToSpoAAuthorizedAttach-req-ind information flow A34-FE2 shall confirm that RSO has the correct value and ensure that the ClientAuthorizationToken correctly identifies the client and service.

8.9.2.3 Actions of A34-FE3

301 On receipt of the challenges A34-FE3 shall forward them to the A34-FE4 using the A34SealingKeyRequest-req-ind information flow.

302 On receipt of A34SealingKeyRequest-resp-conf A34-FE3 shall extract the sealing keys SealK and SealK'.

303 A34-FE3 shall generate counter challenges Rand and Rand', using them with SealK and SealK' to seal K34 using algorithms A34-2 and A34-4 respectively.

304 A34-FE3 shall construct token A34-A from the outputs of A34-2 and A34-4 and send the token to A34-FE3 using the A34SpoAWithUserAuth-resp-conf information flow.

8.9.2.4 Actions of A34-FE4

401 From User-id identifies K, and generates SealK' using K and RSO' as inputs to algorithm A34-4.

402 From SpoA-id identifies Ks, and generates SealK using Ks and RSO as inputs to algorithm A34-1.

403 Sends SealK and SealK' to A34-FE3 using information flow A34SealingKeyRequest-resp-conf.

9 A5 = Authentication of the SpoA by the registrar

See description of A1.

10 A6 = Authentication of the registrar by the SpoA

See description of A2.

11 Confidentiality service

The confidentiality services in TIPHON are optional services, but where provided shall be provided as described in the present document.

11.1 Provided services

11.1.1 E1 = Confidentiality of user communication on the access interface

Encryption on the access interface for outgoing and incoming calls, and for subscription registration procedure can provide confidentiality to the user communication and to TIPHON signalling.

This service lies between the user terminal and the SpoA/TpoA in the transport domain.

11.1.2 E2 = Confidentiality of signalling on the access interface

The signalling can be protected on the access interface. Possible solutions include the use of encryption.

This service lies between the user terminal and the SpoA/TpoA in the service domain.

11.1.3 E3 = Confidentiality of signalling between SpoA entities

Security and other sensitive data such as session keys, call-forwarding number, and personal data can be protected by a number of mechanisms. Encryption is one such mechanism.

This service lies between SpoAs in the service domain.

11.1.4 E6 = Confidentiality of TIPHON-id on signalling interfaces

The TIPHON-id can be protected on signalling interfaces. This may be protected by use of an alias-id or by encryption of signalling.

11.1.5 E7 = Confidentiality of signalling between SpoA and Registrar

NOTE: E7 is considered as a special case of E3.

The link between SpoA and Registrar identifies the user of the service and may identify the key to be used in service E2.

This service lies between SpoA and registrar (special case of SpoA) in the service domain.

11.2 Confidentiality services E1 and E2 step B specification

11.2.1 Description

The confidentiality services E1 and E2 are provisioned by use of an encryption service that lies between the service encoding unit and the transport service. This means that any transport service will be able to carry the encrypted Service Data Unit.

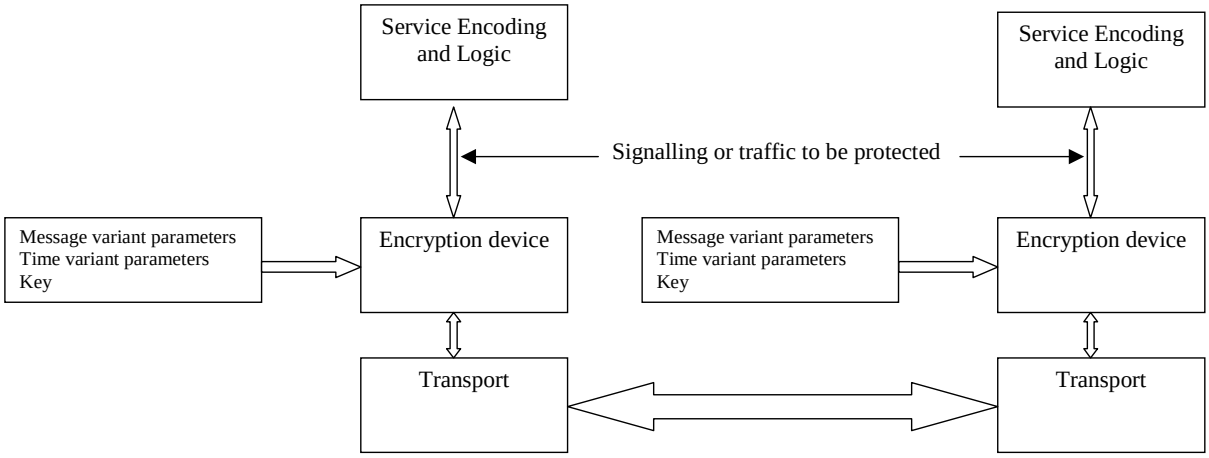
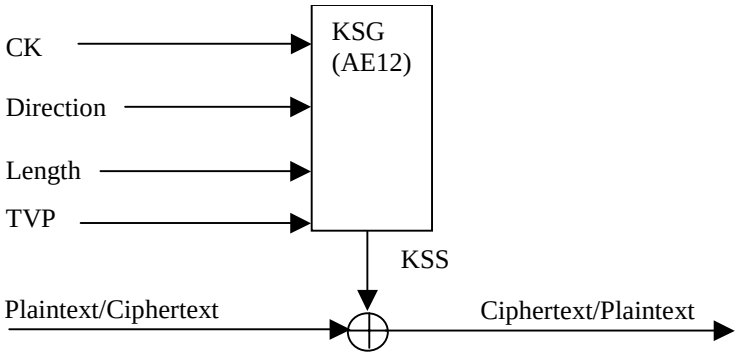


Figure 19: Location of encryption in generic TIPHON model

The encryption device shall be a streaming cipher device operating in a bit replacement mode (i.e. every bit of plaintext is replaced with a bit of ciphertext).

The SpoA should only have one encryption algorithm. A terminal may have more than one algorithm but shall use the algorithm indicated by the SpoA at the time of registration.



CK: Cipher Key
KSG: Key Stream Generator
KSS: Key Stream Segment
TVP: Time Variant Parameter

Figure 20: Speech and control information encryption

11.2.2 Encryption mechanism

The KSS bits shall be modulo 2 added (XORed) with plain text bits in traffic and signalling Service Data Units (SDUs) to obtain encrypted cipher text bits. KSS(0) shall be XORed with the first transmitted bit of the first SDU, and so on. Any unused bits of KSS shall be discarded.

11.3 Confidentiality services E3 and E7 step B specification

11.3.1 Description

The confidentiality services E3 and E7 are provisioned by means of an encryption service that lies between the service encoding unit and the transport service. This means that any transport service will be able to carry the encrypted Service Data Unit.

In the case of service E3 there can be no guarantee of a pre-provisioned encryption key (or other secret). In this instance some form of zero-knowledge secure exchange may be required.

In the case of service E7 a crypto-key is made available as a direct result of authentication services A5 and A6. This key should be used to encrypt any signalling between SpoA and registrar. Where both A5 and A6 have been invoked as part of a mutual authentication the key to be used for encryption should be created as a cryptographic product of the individual cipher keys. E7 shall use an encryption algorithm EA7 (Encryption Algorithm for service E7) defined by its boundary conditions.

11.3.1.1 Algorithm requirements for EA7

The security shall be maintained for "n" messages.

NOTE: SAGE to assist in the definition of EA7 and of "n".

Annex A (normative): Boundary conditions of algorithms

A.1 Authentication algorithms

A.1.1 A1-1

A1-1: shall be used to compute Intermediate Response from K and Challenge. The algorithm shall have the following properties:

Input 1: Bit string of length |K|;

Input 2: Bit string of length |Challenge|;

Output: Bit string of length |Intermediate Response|.

The algorithm should be designed such that it is difficult to infer any information about Input 1 from the knowledge of Input 2 and the Output (even if the details of the algorithm are known).

A.1.2 A1-2

A1-2: shall be used to compute (X)Response as well as CryptoKey from Intermediate Response and Seed. The algorithm shall have the following properties:

Input 1: Bit string of length |Intermediate Response|;

Input 2: Bit string of length |Seed|;

Output 1: Bit string of length |(X)Response|;

Output 2: Bit string of length |CryptoKey|.

The algorithm should be designed such that it is difficult to infer any information about Input 1 or Output 2 from the knowledge of Input 2 and Output 1 (even if the details of the algorithm are known).

A.1.3 A1-3

A1-3: shall be used to compute Result by bit comparison of Xresponse and Response. A1-3 has no special cryptographic properties.

A.1.4 A2-1

A2-1: shall be used to compute IntermediateResponse from K and Challenge. The algorithm shall have the following properties:

Input 1: Bit string of length |K|;

Input 2: Bit string of length |Challenge|;

Output: Bit string of length |IntermediateResponse|.

The algorithm should be designed such that it is difficult to infer any information about Input 1 from the knowledge of Input 2 and the Output (even if the details of the algorithm are known).

A.1.5 A2-2

A2-2: shall be used to compute (X)Response as well as CryptoKey from IntermediateResponse and Seed. The algorithm shall have the following properties:

Input 1: Bit string of length |IntermediateResponse|;

Input 2: Bit string of length |Seed|;

Output 1: Bit string of length |(X)Response|;

Output 2: Bit string of length |CryptoKey|.

The algorithm should be designed such that it is difficult to infer any information about Input 1 or Output 2 from the knowledge of Input 2 and Output 1 (even if the details of the algorithm are known).

A.1.6 A2-3

A2-3: shall be used to compute Result by bit comparison of Xresponse and Response. A2-3 has no special cryptographic properties.

A.1.7 A34-1

A34-1: shall be used to compute SealK from K_S and RSO. The algorithm shall have the following properties:

Input 1: Bit string of length | K_S |;

Input 2: Bit string of length |RSO|;

Output 1: Bit string of length |SealK|.

The algorithm should be designed such that it is difficult to infer any information about Input 1 from knowledge of input 2 and the output (even if details of the algorithm are known).

A.1.8 A34-2

A34-2: shall be used to compute SK34 from K34, Rand, and SealK. The algorithm shall have the following properties:

Input 1: Bit string of length |K34|;

Input 2: Bit string of length |Rand|;

Input 3: Bit string of length |SealK|;

Output: Bit string of length |SK34|.

The algorithms should be designed such that it is difficult to infer any information about Input 1 or Input 3 from the knowledge of Input 2 and the Output, provided that Input 3 is unknown (even if the details of the algorithm are known).

A.1.9 A34-3

A34-3: shall be used to compute K34 and Rand from SK34 and SealK. The algorithm shall have the following properties:

Input 1: Bit string of length |SK34|;

Input 2: Bit string of length |SealK|;

Output 1: Bit string of length |K34|;

Output 2: Boolean;

Output 3: Bit string of length |Rand|.

The algorithm should be designed such that it is difficult to find for a fixed Input 2 values for Input 1 that result in Output 2 assuming the value FALSE, provided that Input 2 is unknown (even if the details of the algorithm are known). Moreover, it shall be difficult to derive (parts of) Input 2 from the observation of various matching values of other inputs and outputs (known plain text attack).

A.1.10 A34-4

A34-4: shall be used to compute SealK' from K and RSO'. The algorithm shall have the following properties:

Input 1: Bit string of length | K|;

Input 2: Bit string of length |RSO'|;

Output 1: Bit string of length |SealK'|.

The algorithm should be designed such that it is difficult to infer any information about Input 1 from knowledge of input 2 and the output (even if details of the algorithm are known).

A.1.11 A34-5

A34-5: shall be used to compute SK34' from K34, Rand', and SealK'. The algorithm shall have the following properties:

Input 1: Bit string of length |K34|;

Input 2: Bit string of length |Rand'|;

Input 3: Bit string of length |SealK'|;

Output: Bit string of length |SK34'|.

The algorithms should be designed such that it is difficult to infer any information about Input 1 or Input 3 from the knowledge of Input 2 and the Output, provided that Input 3 is unknown (even if the details of the algorithm are known).

A.1.12 A34-6

A34-6: shall be used to compute K34 and Rand' from SK34' and SealK'. The algorithm shall have the following properties:

Input 1: Bit string of length |SK34'|;

Input 2: Bit string of length |SealK'|;

Output 1: Bit string of length |K34|;

Output 2: Boolean;

Output 3: Bit string of length |Rand'|.

The algorithm should be designed such that it is difficult to find for a fixed Input 2 values for Input 1 that result in Output 2 assuming the value FALSE, provided that Input 2 is unknown (even if the details of the algorithm are known). Moreover, it shall be difficult to derive (parts of) Input 2 from the observation of various matching values of other inputs and outputs (known plain text attack).

A.1.13 A34-7

A34-7: shall be used to compute CryptoElement from RSO, RSO' and K34. The algorithm shall have the following properties:

Input 1: Bit string of length |RSO|;

Input 2: Bit string of length |RSO'|;

Input 3: Bit string of length |K34|;

Output 1: Bit string of length |CryptoElement|.

The algorithm should be designed such that it is difficult to infer any information about Input 3 from the knowledge of the other inputs and the Output (even if the details of the algorithm are known).

A.1.14 A34-8

A34-8: shall be used to compute token T34-A from SK34' and SK34. A34-8 has no special cryptographic properties.

A.1.15 A34-9

A34-9: shall be used to compute token T34-B from SK34' and CryptoElement. A34-9 has no special cryptographic properties.

A.1.16 A34-10

A34-10: shall be used to compute token T34-C from RSO, RSO', K34 and the ClientAuthorizationToken. The algorithm shall have the following properties:

Input 1: Bit string of length |RSO|;

Input 2: Bit string of length |RSO'|;

Input 3: Bit string of length |K34|;

Input 4: Bit string of length |ClientAuthorizationToken|;

Output 1: Bit string of length |T34-C|.

The algorithm should be designed such that it is difficult to infer any information about Input 3 from the knowledge of the other inputs and the Output (even if the details of the algorithm are known).

A.2 Dimensioning of the cryptographic parameters

Table A.1 shows the lengths of the cryptographic parameters given in annex A.

Table A.1: Dimensioning of cryptographic parameters

Abbreviation	No. of Bits
K	196
Intermediate result	196
Challenge	128
Response	128
Seed	128
CryptoKey	128

A.2.1 Terminal-identity

The terminal identity used is required to uniquely identify the authentication key. In cases where access to the authentication key is required globally then the terminal identity is required to be globally unique.

A.3 Encryption algorithms

A.3.1 EA12 - Confidentiality algorithm

A.3.1.1 Overview

The input parameters to the algorithm are the Cipher Key (CK), a Time Dependent Input (TVP), the direction of transmission (DIRECTION) and the length of the keystream required (LENGTH). Based on these input parameters the algorithm generates the output keystream block (KEYSTREAM) which is used to encrypt the input plaintext block (PLAINTEXT) to produce the output ciphertext block (CIPHERTEXT).

The input parameter LENGTH shall affect only the length of the KEYSTREAM block, not the actual bits in it.

A.3.1.2 Use

The function EA12 shall only be used to protect the confidentiality of user data and signalling data as required for services E1 and E2.

EA12 shall be found in the terminal and in the SpoA.

A.3.1.3 Extent of standardization

The function EA12 shall be fully standardized.

A.3.1.4 Implementation and operational considerations

The algorithm should be designed to accommodate a range of implementation options including wholly hardware and wholly software implementations, and mixed hardware and software implementations.

A.3.1.5 Type of algorithm

The function EA12 should be a symmetric synchronous stream cipher.

A.3.1.6 Interfaces to the algorithm

A.3.1.6.1 CK

CK: the cipher key

CK[0], CK[1], …, CK[127]

The length of CK is 128 bits. In case the effective key length should need to be made smaller than 128 bits, the most significant bits of CK shall carry the effective key information, whereas the remaining, least significant bits shall be set zero.

CK should be derived during the authentication cycle and be computed from the combination of the SessionKey outputs from authentication services A3 and A4 (i.e. combination of key K34 and K43).

ASN.1 example: CK-Type ::= BIT STRING (SIZE(KeyLength))

A.3.1.6.2 TVP

TVP: a time dependent input.

TVP[0], TVP[1], …, TVP[31]

The length of the TVP parameter is 32 bits. The exact structure of the TVP parameter is implementation dependent and has to be specified in the step D process.

ASN.1 example: TVP-Type ::= BIT STRING (SIZE(TVPLength))

A.3.1.6.3 DIRECTION

DIRECTION: the direction of transmission of the transport to be encrypted.

DIRECTION[0]

The length of DIRECTION is 1 bit. The same cipher key may be used for uplink and downlink channels (i.e. from the terminal and to the terminal) simultaneously. To avoid using the same keystream to encrypt both uplink and downlink transmissions, the algorithm shall generate the keystream based on the direction of transmission.

An explicit direction value is required in preference to splitting the keystream segment into uplink and downlink portions to allow for asymmetric bearer services.

ASN.1 example: Direction-Type ::= BIT STRING (SIZE(DirectionLength))

A.3.1.6.4 LENGTH

LENGTH: the required length of keystream.

LENGTH[0], LENGTH[1], …, LENGTH[18-1]

The length of LENGTH is 18 bits. The length of the plaintext block that is transmitted during a single transport frame may vary. The algorithm shall generate a keystream block of variable length based on the value of the length parameter.

The input parameter LENGTH shall affect only the length of the KEYSTREAM block, not the actual bits in it.

ASN.1 example: Length-Type ::= BIT STRING (SIZE(LengthLength))

A.3.1.6.5 KEYSTREAM

KEYSTREAM: the output keystream segment.

KS [0], KS [1], …, KS [LENGTH-1]

The length of a keystream segment block equals the value of the input parameter LENGTH.

ASN.1 example: Keystream-Type ::= BIT STRING (SIZE(Length))

A.3.1.6.6 PLAINTEXT

PLAINTEXT: the plaintext.

PT[0], PT[1], …, PT[LENGTH-1]

The length of a keystream block equals the value of the input parameter LENGTH.

This plaintext block consists of the payload of the particular PDUs/SDUs to be encrypted in a single transport frame. It may consist of user traffic or signalling data.

ASN.1 example: Plaintext-Type ::= BIT STRING (SIZE(Length))

A.3.1.6.7 CIPHERTEXT

CIPHERTEXT: the ciphertext.

CT[0], CT[1], …, CT[LENGTH-1]

The length of a keystream block equals the value of the input parameter LENGTH.

ASN.1 example: Ciphertext-Type ::= BIT STRING (SIZE(Length))

Annex B (informative): Bibliography

- GSM 03.20: "Digital cellular telecommunications system (Phase 2); Security related network functions".
- GSM 02.09: "European digital cellular telecommunications system (Phase 2); Security aspects".
- GSM 12.03: "Digital cellular telecommunications system (Phase 2); Security management".
- ETSI EN 300 392-7: "Terrestrial Trunked Radio (TETRA); Voice and Data; Security".
- ETSI TS 101 303: "Telecommunications and Internet Protocol Harmonization Over Networks (TIPHON) Release 4; Service Independent Requirements Definition; Service and Network Management Framework; Part 1: Overview and Introduction".
- ETSI TS 101 882-2: "Telecommunications and Internet Protocol Harmonization Over Networks (TIPHON) Release 4; Protocol Framework Definition; part 2; Registration and Service Attachment service meta-protocol definition".
- ITU-T Recommendation H.225.0 Version 2: "Call Signalling Protocols and Media Stream Packetization for Packet Based Multimedia Communications Systems".
- ITU-T Recommendation H.245 Version 3: "Control Protocol for Multimedia Communication".
- ITU-T Recommendation H.323 Version 3: "Packet Based Multimedia Communication Systems".
- ITU-T Recommendation H.323 Annex F: "Simple Endpoint Types".
- ITU-T Recommendation H.323 Annex J: "Security for H.323 Annex F".
- ITU-T Recommendation H.235 Version 2: "Security and Encryption for H Series (H.323 and other H.245 based) multimedia terminals".

History

Document history		
V4.1.1	February 2003	Publication