Project Number	AC095
Project Title	ASPeCT: Advanced Security for Personal Communications Technologies
Deliverable Type	Intermediate
Security Class	Public

Deliverable Number	D12
Title of Deliverable	**Migration scenarios: first implementation**
Nature of the Deliverable	Report
Document reference	AC095/ATEA/W21/DS/P/12/C
Contributing WPs	WP2.1
Contractual Date of Delivery	May 1997 (Y03M03)
Actual Date of Delivery	2 August 1997
Editors	Geneviève Vanneste, Bart Franco

Abstract	This report describes the realized demonstration. The aim of the demonstrator is to show a migratory path for security features. After a study in depth on the migration problem, a multi-application card for GSM and UMTS is proposed as migratory path to introduce new and enhanced security features. To achieve flexible introduction of new authentication mechanisms and algorithms, a framework for authentication has been introduced, with the ability to migrate smoothly from one mechanism to another. The specifications of the demonstration : the authentication framework, the public key authentication mechanism and the UMTS UIM are described in detail. The realisation of the demo is explained. The GUI is described in detail. An overview is given of the tests of the demo. Performance measurements have been done and form part of the evaluation of the demo

Keywords	ACTS, ASPeCT, UMTS, security, migration, multi-application card, demonstration

1.Executive Summary

UMTS, Universal Mobile Telecommunications System, offering a plethora of advanced mobile telecommunication services via a variety of public and private network operators in both outdoor and indoor environments, will be introduced in the early years of the 21st century.
Enhanced security features are needed to fulfill the security requirements. Mechanisms to realise the UMTS security features are currently under development. Secret key-based mechanisms, as well as public key-based mechanisms have been proposed for UMTS, providing mutual authentication, cipher key agreement for confidentiality, anonymity and non-repudiation.
To enable migration from GSM to UMTS a multi-application card is defined, containing a GSM SIM application and a preliminary UMTS UIM application. To achieve flexible introduction of new authentication mechanisms and algorithms, a framework for authentication has been introduced, with the ability to migrate smoothly from one mechanism to another.
In order to facilitate roaming in a network with a large number of Network Operators and Service Providers, it might be desirable (or even necessary) for roaming agreements to be set-up dynamically, as and when they are required.
A demonstrator, on a PC base, has been developed, demonstrating the authentication framework with a proposed public key and secret key based authentication mechanism for UMTS. At the user's side a smart card is available, providing the authentication functionality. The demonstrator will show the feasibility of a multiapplication card for GSM and UMTS. The card will contain both a GSM SIM application and a preliminary UMTS application. Various problems arise in supporting multiple applications on a single card : application selection, independence of application. A careful approach is required to ensure both adequate security and sufficient interoperability. The functionality of the terminal is put on a PC. At the network's side an Authentication centre (AC) has been developed. This AC can function in a visited network (with the NO) or in a home network (with the SP). The procedures for automatic roaming agreement are not demonstrated.
The demonstrator has been evaluated in depth and the proposed enhancements are:

- optimisation of the GUI,
- implementation on a more realistic network environment,
- increase of the performance of the cryptographic engines,
- identification of the memory trade-offs on the smart card.

The next step, the integration of the demo on the EXODUS platform, will give a more realistic view on the demonstrated security features.

Table of Contents

2.Introduction

This report describes the realized demonstration. The aim of the demonstrator is to show a migratory path for security features. After a study in depth on the migration problem, a multi-application card for GSM and UMTS is proposed as migratory path to introduce new and enhanced security features.

To achieve flexible introduction of new authentication mechanisms and algorithms, a framework for authentication has been introduced, with the ability to migrate smoothly from one mechanism to another.

The specifications of the demonstration : the authentication framework, the public key authentication mechanism and the UMTS UIM are described in detail in section 5.

The realisation of the demo is explained in section 6. In section 7 the GUI is described in detail.

An overview is given of the tests of the demo software in section 8.

Performance measurements have been done and form part of the evaluation of the demo, described in section 9.

In section 10 a summary of enhancements is given.

This deliverable is made public. More elaborated documentation (designs, test specification, measurement files...) is available as internal ASPeCT documents.

NOTE: The following **limitations and restrictions** apply. The **public** nature of this deliverable is restricted to this document.
No general rights to the programmes and the libraries which constitute the demonstrator/prototype are given or implied. Certain components may be

- proprietary,
- subject to non-disclosure agreements,
- claimed and acknowledged as background material,
- subject to governmental controls on export or re-export.

Specific enquiries or requests for clarification may be addressed, in the first instance, to the editor.

2.1.Contributors

This is a list of all project managers involved in the ASPeCT project plus the principal editors (Bart Franco and Geneviève Vanneste) whose contact details are included.

Bart Preneel	ESAT/COSIC KU Leuven K. Mercierlaan 94 B 3001 Heverlee Belgium	Phone: +32 16 32 1148 Fax: +32 16 32 1986	bart.preneel@esat.kuleuven.ac.be
Yannis Vithynos	PANAFON 2 Mesogeon Avenue 11527 Athens Greece	Phone: +30 1 6407267 Fax: +30 1 6407039	vithynos@panafon.gr
John Shaw-Tayler	Royal Holloway, University of London Egham Surrey TW20 0EX England	Phone: +44 1784 443430 Fax: +44 1784439786	jst@dcs.rhbnc.ac.uk
Günther Horn	Siemens AG ZFE T SN 3 D-81730 München Germany	Phone: +49 89 636 41494 Fax: +49 89 636 48000	Gunther.Horn@mchp.siemens.de
Geneviève Vanneste	Siemens Atea Atealaan 34 B-2200 Herentals Belgium	Phone: +32 14 252937 Fax: +32 14 253339	p82586@vnet.atea.be
Bart Franco	Siemens Atea Atealaan 34 B-2200 Herentals Belgium	Phone: +32 14 253302 Fax: +32 14 253339	p82587@vnet.atea.be
Eric Johnson	GIESECKE & DEVRIENT GMBH Prinzregenstr. 159 D-81607 München Germany	Phone: +49 89 4119 944 Fax: +49 89 4119 905	X400: c=de; a=cwmail; p=g+d; s=johnson; g=eric Internet: 100277.1206@compuserve.com
Nigel Jefferies	Vodafone Ltd The Courtyard 2-4 London Road Newbury Berks RG14 1JX England	Phone: +44 1635 503883 Fax: +44 1635 31127	Nigel.Jefferies@ vodafone.gold-400.gb

2.2.Document History

Revision	Date	Changes
A	20/5/97	preliminary version
B	12/6/97	draft version, split up of section 9
C	17/7/97	draft version for review by WP21 partners
D	25/7/97	comments received from WP21 partners version for review by PMC partners

3.References

[1] AC095/Atea/WP21/DS/P/02/1 Initial report on security requirements, ACTS ASPeCT deliverable, Feb 1996
[2] AC095/Atea/WP21/DS/P/05/1 Migration Scenarios, ACTS ASPeCT deliverable, Aug 1996
[3] Design Patterns, Elements of Reusable Object-Oriented Software, ISBN 0-201-63361-2
[4] ITU-T Recommendation E.800 (04/94), “Terms and definitions related to quality of service and network performance including dependability”
[5] ITU-T Recommendation I.350 (03/93), “General aspects of quality of service and network performance in digital networks, including ISDNs”
[6] General authentication framework for UMTS, SMG sg 196/96
[7] AC095/G&D/WP24/DS/P/15/1 The UIM: first specification, ACTS ASPeCT deliverable, July 1997

4.Abbreviations

AC	Authentication Centre
ACTS	Advanced Communications Technologies and Services
AMV	the Agnew-Mullin-Vanstone equation
ASPeCT	Advanced Security for Personal Communications Technologies
CA	Certification Authority
ETR	ETSI technical report
ETSI	European Telecommunications Standards Institute
FSM	Finite State Machine
GSM	Global System for Mobile communications
NO	Network Operator
SIM	Subscriber Identification Module
SP	Service Provider
UIM	User Identity Module
UMTS	Universal Mobile Telecommunication System

5.Description of the demonstrator

In the following sections the technical specifications of the demo are described in detail: the currently discussed authentication framework for UMTS is presented, together with one of the proposed authentication mechanisms and the interface to the UMTS UIM is described.

5.1.The Authentication framework

The principle objective of the Authentication Framework [2] is to provide a flexible procedure for user-network authentication allowing a number of different mechanisms and algorithms to be incorporated, with the ability to migrate smoothly from one mechanism to another. This framework allows the authentication capabilities of SIMs, network operators (NOs) and service providers (SPs) to be taken into consideration for the selection of the mechanism to be used. A list of capability classes (including the mechanisms supported) will need to be maintained so that different entities (SIMs, NOs, SPs and TTPs) can permit the negotiation of the mechanisms to be used.

In order to facilitate roaming in a network with a large number of NOs and SPs, it might be desirable (or even necessary) for roaming agreements to be set-up dynamically, as and when they are required. In practice, the roaming agreement would be first requested as a result of an initial authentication request sent by the user/terminal to a network visited for the first time. A prerequisite of this procedure is that the SP and NO wishing to establish the agreement have authenticated each other.

NO-SP authentication will be carried out using a globally agreed mechanism in order to ensure that NOs and SPs world-wide have the capability to authenticate each other. Unlike the user-network authentication mechanism, flexibility to change mechanisms is not considered to be a crucial factor. Apart from being a prerequisite to a roaming agreement, NO-SP authentication will permit the SP to delegate user-network authentication to the NO. The SP would send authentication data to the NO in advance, permitting the NO to carry out authentication on behalf of the SP.

It should be noted that the identity of the User is not released until the stage of user-network authentication. The rationale for this is that the identity of the User is immaterial until the stage of authentication is reached; it is only the identity of the Service Provider which is required up until the stage of authentication. Note also that the identity of the User is never necessarily required by the Network Operator, hence temporary identities are used to provide party anonymity of the User towards the Network Operator.

A further characteristic of the Authentication Framework is the use of an authentication Capability Class, which acts to identify the particular authentication mechanisms which are supported by the UIM of a User. Each respective authentication mechanism is identified by an unique identifier. The rationale for this is that visited Network Operators may immediately identify whether they can support a particular Capability Class; unknown authentication mechanisms would be defined by the respective Service Provider upon request from the Network Operator.

5.1.1.Operational Scenario

As an example the operational scenario (described in [2], but here some enhancements are included) is described where a user, not registered in the network, initiates authentication and no roaming agreement exists between the Network and the user's service provider.

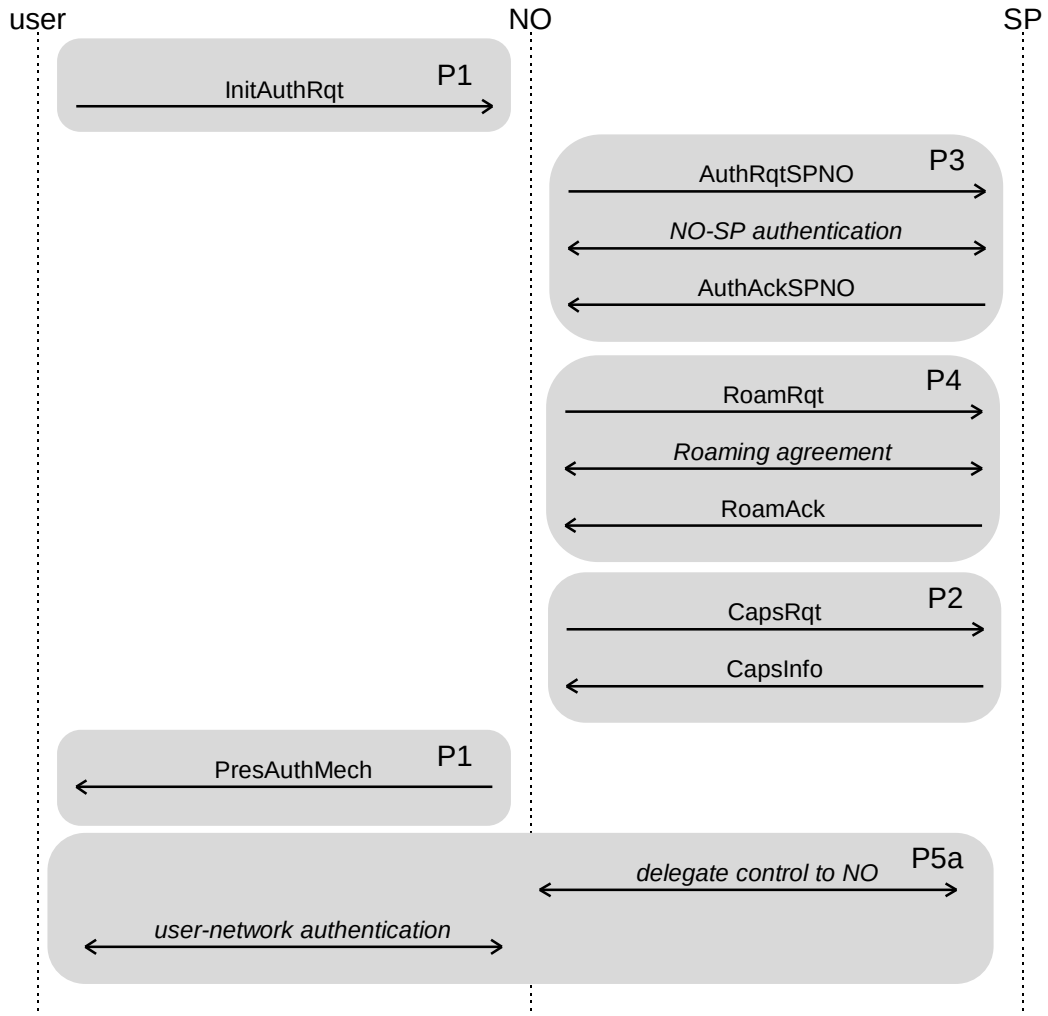

Figure 5-1Operational scenario for 'User not registered, no roaming agreement'

The user sends an initial message to a NO - this will include the user's service provider, authentication capability class, but not his identity nor his temporary identity. The NO does not have a roaming agreement with the SP so it initiates a procedure to establish one dynamically - if one cannot be established dynamically, then the request is refused. A procedure to establish a roaming agreement begins with the NO and SP authenticating each other. After authentication the NO and SP negotiate a roaming agreement which will involve each party digitally signing the agreement. Once an agreement has been established, the NO checks the

authentication Capability Class of the User to establish if it is known. If it is known, the Network operator compares the associated authentication mechanisms with its own supported authentication mechanisms. If it is not known, the Network Operator sends the user's authentication capability class to his SP. The SP will respond by providing the NO with the authentication capabilities of that particular authentication capability class - this will include the authentication mechanisms the user is capable of handling. The NO will then choose an authentication mechanism, from those of the User's Capability Class, which is both supported by the Network Operator and by the User's UIM. The NO then sends the identity of the prescribed mechanism to the user. The authentication mechanism for new registrations involving the SP, NO and user is initiated. Note, however, that the SP may choose to delegate the actual authentication to a Certification Authority (CA).

5.2.Public key Authentication mechanism

The ***Siemens protocol*** is a public key based authentication mechanism defined by Siemens AG and submitted for standardisation [6]. There exist three versions of the protocols. In the following sections only one version is described, the version allowing authentication of a user to a network, without the need that they share certificates of each other. This version is applied when 'New Registration' of a user in a network occurs. [1]

The goals of the protocol are the following:

- mutual explicit entity authentication of User and Network operator
- agreement between the user and the Network operator on a shared secret key K_S with mutual implicit key authentication
- mutual key confirmation of the User and the Network operator
- mutual assurance of key freshness
- non-repudiation by the User of data sent by the User to the Network operator and vice versa
- confidentiality of the identity IMUI of the User on the air interface
- exchange of certified public keys between U and N

The **data** used within the protocol:

AUTH$_N$: This value is calculated to authenticate the network operator (NO) to the user .

CertN : This is a valid certificate, issued by a certification authority CA, on a public key of the asymmetric signature system of the NO. It is available at the NO.

CertU : This is a valid certificate, issued by a certification authority CA, on a public key of the asymmetric signature system of the user. It is available at the user.

data1, data2 : Those are optional data fields, to illustrate the non-repudiation feature.

id$_{ca}$: : This is the identity of the Certification Authority.

id$_n$: This is the identity of the NO.

K$_S$: This is the session key .

g : generator g, known by the user, NO and SP, g is a generator of a finite group G with modulo p (p is a prime) in which the Discrete Logarithm Problem is hard .

s : This is the secret key agreement key for the NO. It is linked with g^s (the public key agreement key).

g^s : This is the public key agreement key of the NO.

IMUI : This is the International Mobile User Identity, uniquely identifying the mobile user.

PK_U : This is the public key of the user used to verify **signatures** from the user.
RND$_U$: This is random number generated by the user.
The **algorithms** used within the protocol:
h1 : This is a one way function. It is used to calculate the session key:
K_S = **h1(**(g^{RNDU})^S || RND_N**)**
h2 : This is a hash function and used to calculate $AUTH_N$.
$AUTH_N$ = **h2 (**K_S**)**
h3 : This is a hash function and used to calculate a hash value before signature calculation
Sig$_u$: This is a secret **signature** transformation owned by the user.
Ver$_u$: This is a verification algorithm corresponding the signature algorithm. This algorithm needs the public key (PK_U in this case) as input.
Enc: This is a symmetric encryption algorithm. **Enc(**K,data**)** means that data is encrypted with key K.
Dec: This is a symmetric decryption algorithm. It corresponds **Enc().**
The following list is **required** (by user and/or NO):

- the user needs the id_{no} ;
- both the user and NO possess the generator g;
- the NO has secret and public key agreement keys s and g^s ;
- the NO has a valid certificate CertN;
- the user has a signature transformation Sig_u ;
- the user has a valid certificate CertU;

The message flow corresponding to the authentication protocol:

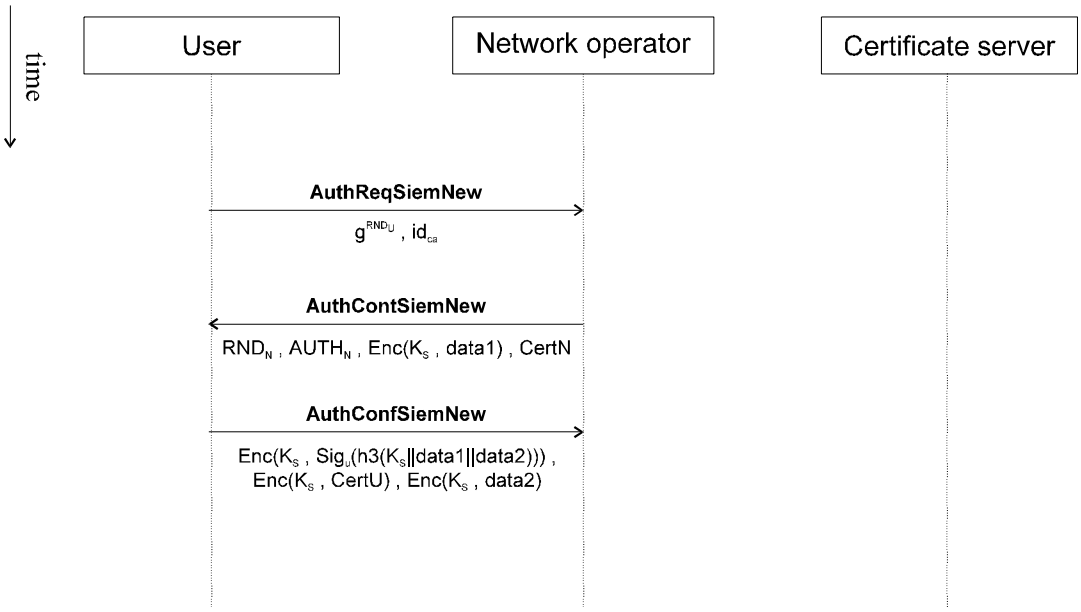

Figure 5-2 Public key authentication mechanism

The certificate server is not interrogated on-line during this version of the protocol.

5.3.The UMTS UIM

Until now, the discussion of the authentication mechanisms (in particular the Siemens protocol) has been concentrated on messages exchanged between logical entities: the user, the network operator and the service provider. This is not

sufficient to describe the interface between a UMTS mobile terminal and a UMTS UIM. The message flow across that interface is different for at least three reasons:

- the communication structure is fixed: the mobile terminal sends a card command to the UIM (together with some data) and gets data back (card response).
- the UIM cannot initiate a communication: this is not envisaged for the UMTS. For example, a framework like the SIM Toolkit in GSM does not yet exist.
- the message length is limited to 255 bytes. Longer messages can only be sent with some chaining mechanism.

The aim of this section is to give an introduction into the design process for UIM card commands - how to map the message flow across the user / network operator interface as described before to some functionality of security related card commands necessary to support that message flow.

During authentication set-up, the network operator sends a message called *InitAuthRqt* to the user's service provider in case of a new registration. The content of this message is the *authentication capability class* of the user and the *identity* of its service provider. Because the user is not yet known to the network, it is up to the UMTS mobile terminal to read out this information from the UIM plugged into it. The first two card commands issued are therefore:

- ReadBinary(File_ID[EF_{SPID}])
- ReadBinary(File_ID[EF_{ACCL}])

The elementary files EF_{ACCL} and EF_{SPID} store the respective data and are integrated into a directory DF_{UMTS}.
The service provider informs the network operator about the user's authentication capabilities. Based on this information, the network operator makes a choice among a list of mechanisms supported by the UIM and sets that mechanism for the rest of the current session with the card command

- SetAuthMechanism(AuthMechID)

The UIM responds with Yes or No depending on the network operator's choice.
In case of the Siemens protocol, it is necessary that the network knows which certification authorities are supported by the UIM:

- which CA issued the certificate on the user's public key agreement and signature verification key ?
- which CA is accepted by the UIM - can the UIM verify certificates issued by a particular CA ?

In general, it may be possible that the UIM splits this information and stores it in different files. For the rest of this section, we assume that one CA satisfies both purposes. The next card command issued by the mobile terminal is therefore:

- IdCA = ReadBinary(File_ID[EF_{CA}])

An elliptic curve and some rational point *g* on it which defines a cyclic group over that curve acceptable for cryptographic purposes, are defined in advance. The UIM can compute powers of *g* for any random integral exponent. This is the next step in the Siemens protocol and achieved by the card command

- P = GetChallenge( )

The UIM computes a random number *RND* and sends g^{RND} back as the response data for GetChallenge.
The mobile terminal sends the identity of the CA and the challenge P transparently to the network operator which looks for a certificate on its public key agreement key issued by that particular CA. If the NO can provide such a certificate, it sends it to the mobile terminal which stores it in some file EF_{CertN} and verifies its validity:

- Status = VerifyCertificate(File_ID[EF_{CertN}])

The status value is either Yes or No depending on a positive verification of the appended signature on it by the UIM.
With the received challenge and its private key agreement key, the network can compute a session key K_S shared with the UIM and an authentication token $AUTH_N$ which is a hash code of the session key:

- $AUTH_N$ = MutualAuthenticate(M)

This card command uses the message M sent by the network to the mobile terminal as a response to the challenge P:

$$M = RND_N \,||\, AUTH_N \,||\, Enc(K_S,data1)$$

The AUTH token computed by the UIM covers

$$Enc(K_S,Sig(K_S,data1)) \,||\, Enc(K_S,CertU)$$

according to the protocol. This is unfortunately not immediately possible because of the length restriction for the card's response. The encrypted user certificate is therefore read out with a second card command which is the last one for the Siemens protocol:

- EncryptedCertificate = ReadCertificate(File_ID[EF_{CertU}])

It may look surprising at first sight how the whole protocol changes by breaking down the message flow across a different interface. But this also shows that design of card commands is different from defining messages flowing across a network.

6.Realisation of the demonstrator

The demonstrator has been developed within the ACTS project ASPeCT (Advanced security for Personal Communications technologies and services). The aim of the demonstrator is to show a migratory path for security features. After a study in depth on the migration problem, a multi-application card for GSM and UMTS is proposed as the migratory path to introduce new and enhanced security features. The UMTS authentication framework is implemented in combination with a public key based authentication mechanism (see section 5.2). A version of the demonstration without smart card is also available for a secret key based authentication mechanism.

6.1.Overview

Within the demo, three logical entities (roles) are involved:

- The User: is authorised by a subscriber to make use of the telecommunication services, the subscriber subscribed to by the service provider.
- The network operator: provides the network capabilities necessary for the support of the services or set of services offered to the users.
- Service Provider : has overall responsibility for the provision of a service or set of services to users associated with a subscription and for negotiating the network capabilities associated with that service or set of services with network operators.

These roles have been mapped upon the following physical architecture:

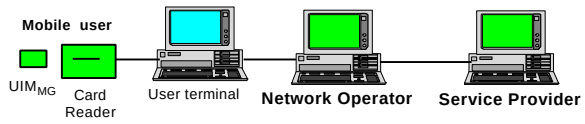

Figure 6-1 physical architecture

Description of the demonstrated features:

New registration, roaming agreement exists:
The user is not registered with the NO, the NO and user's SP do have a roaming agreement. The NO has no security related data for the user.

The user wants to register and sends a registration request to the network. The network recognises that it has a roaming agreement for the user's service provider. The NO will receive the authentication capabilities from the registered user by the SP. An authentication mechanism will be negotiated and executed between the NO and the user.

Authentication of registered user:
The user is registered with the NO, the NO has security related data for the user.

The NO or the user can initiate the authentication by sending the appropriate message. The NO has all necessary data of the user, it has a roaming agreement with the SP and knows the users authentication capabilities. An authentication mechanism will be executed between the NO and the user.

The authentication mechanism implemented on the smart card (UIM) and in the network is the one described in section5.2.

6.2.Selection of algorithms

The algorithms chosen for the mechanism described in section 5.2 are:
h1, h2, h3 : RIPEMD-128
$\boldsymbol{Sig_u, Ver_u}$: AMV signature, based on elliptic curves, .
Enc, Dec: DES-CBC
finite group G : elliptic curve, one point is represented by 40 bits.

6.3.The UIM realisation

The UIM implemented for the demonstration supports both the GSM and the UMTS. It will be a multifunctional smart card with at least two applications on it. For the demonstration, the UIM supports the SIM functionality as specified for GSM Phase 2 and the security functionality defined by the Siemens authentication protocol.
From an observer's point of view, the UIM can be used as a GSM SIM by setting up a phone call with a GSM handset that supports standard features. This shows that the card is compatible with the GSM Phase 2 functionality according to the most recent version of the GSM 11.11 specification.
The UIM functionality is demonstrated by plugging the smart card into an intelligent card reader that communicates with a PC that simulates the UMTS network. The access to the security functionality is controlled by the mobile terminal implementation in that PC.
It will also be possible to directly access data stored in the UIM that is not security relevant with the card reader's display and keyboard. This does not touch the message flow between PC and UIM and allows a convenient interface to user data stored in it.

6.4.The Network realisation

The demonstrator we have to build for the ASPeCT project consists of several entities exchanging messages to each other. Each entity acts like a finite state machine. It receives an event (a communication message over TCP/IP, serial link or message queue or a user message from the Graphical User Interface) and responses to that event by taking some actions like calculating an algorithm and sending a message. Both communication between entities in the same application (via a message queue) as well as communication between entities in different applications (via TCP/IP) are possible.
In the demonstrator, all entities are represented by finite state machines. The design of these finite state machines is based on a state pattern, and implemented in C++ [3]. We use this state pattern for the following reasons :

- An object's behaviour depends on its state, and it must change its behaviour at run-time depending on that state.
- Operations have large, multipart conditional statements that depend on the object's state. This state is usually represented by one or more enumerated constants. Often, several operations will contain this same conditional structure. The state pattern puts each branch of the conditional in a separate class. This lets you treat the object's state as an object in its own right that can vary independently from other objects.

The intent of the state pattern is to allow an object to alter its behaviour when its internal state changes. The object will appear to change its class.

The key idea in this pattern is to introduce an abstract class (TFSM_State) to represent the states of all entities in the demo. This abstract class declares an interface common to all classes that represent different operational states. The subclasses of this abstract class implement the state-specific behaviour.
The class TFSM maintains a state object (an instance of a subclass of TFSM_State) that represents the current state. The class TFSM delegates all state-specific requests to this state object. TFSM uses its TFSM_State subclass instance to perform operations particular to the state.
Whenever the state changes, the TFSM object changes the state object it uses.
An ASN.1 shareware tool is used to produce the necessary C++ routines for BER encoding and decoding of the messages exchanged in our demo. ASN.1 is a notation for describing data structures. It is an abstract representation because it does not specify how data is represented in a local computer nor does it specify how data is represented when they are communicated between systems. The tool we use is called SNACC and is freely available via the internet.
It was agreed between the ASPeCT partners to use the ACRYL library from Siemens ZT IK 3 for the provision of basic cryptographic functions. Following functions are provided by ACRYL, which stands for Advanced CRYptographic Library :

- Random number generation based on DES-OFB and triple DES-OFB
- Hash functions RIPEMD-128 and RIPEMD-160
- RSA signature generation and verification
- AMV signature generation and verification based on an elliptic curve over GF(p)
- Encryption with DES-CBC and triple DES-CBC
- Exponentiation in GF(p)
- Exponentiation in an elliptic curve over GF(p)
- Key generation for RSA, DES and elliptic curves

7.GUI of the demonstrator

7.1. Introduction

The purpose of this section is to provide design information regarding the WP2.1 demonstration GUI.
The GUI has been designed using the **Visual Programmer** Package (MFC Switch-It Module) that is offered with the WATCOM C/C++ V.10.6.

7.2. Description of the GUI

The GUI's main screen displays three bitmaps visualizing the three entities that are involved in the demonstration: User, Network Operator and Service Provider. The menu bar comprises six items, namely (see Fig.1):

- **Configuration**
- **Entity**
- **Status**
- **Demonstration**
- **Action** and
- **Help**.

The **Configuration** item refers to the communication subsystem setup. The **Entity** item enables the entity creation, destruction and registration. The **Status** menu item provides information about the entities particular features. The **Demonstration** item concerns the "MONITOR" and "TRACER" configuration and startup. The **Action** item refers to the actions that the involved entities may perform. Finally, the **Help** item will provide information regarding the GUI components. More detailed information regarding the main menu items is given below.

7.2.1. The Configuration Part

The **Configuration** pull-down menu comprises two items (Fig. 2):

- the *Setup* and
- the *Status* option.

Upon selection of the *Configuration -> Setup* item, the *TCP/IP Setup* dialog box appears on the screen. The selection of "Server" or "Client" Mode is made by clicking on the respective radio-buttons. On "Server" selection, the GUI user may define the "Number of Clients". A default value is provided (see footnote 1). On "Client" selection, the user may select among the available "Clients" (Client #1 … Client #9) and assign the Server IP address. A default value for the Server IP address is also provided[1].
A dialog box presenting status information about the communication subsystem appears by selecting the *Configuration -> Status* item.

[1] Default values are provided. The user, however, is able to assign a new value through the keyboard.

7.2.2. The Entity Part

The **Entity** pull-down (see Fig. 2) menu comprises:

- the *Create* entity,
- the *Destroy* entity
- the *Status* and

The creation and the registration of the involved entities, are enabled through the *Create entity* dialog box. The user is able to create an entity by selecting it from the displayed list and pressing the "Create" button. When an entity is created, the registration dialog box of the selected entity appears (see Figures 2-3). The registration dialog box enables:

(a) The configuration of the authentication mechanism (in case of User and Network Operator creation). There are three authentication capability classes available, namely:
   - class 1 : the symmetric key - based authentication mechanism, defined by RHUL
   - class 2 : the public key - based authentication mechanism, defined by Siemens
   - class 3 : both of the above authentication mechanisms

(b) The preferable Tracer Mode. The Tracer Window may display:
   - No messages
   - Decoded messages
   - Encoded messages, or
   - Decoded and Encoded messages

(c) The definition of the entity's identity. In case of User creation, the Service Provider's identity is also required.

Default values are provided for all of the fields described above. The user may also alter the identity value through the keyboard.

As soon as an entity is created, the background colour of its corresponding bitmap changes.

The *Entity-> Destroy* menu item enables the destruction of one or more entities (Fig. 4).

A dialog box, presenting overall information on entities status, appears when selecting the *Entity-> Status* menu item (Fig.4) .

7.2.3.The Status option

The **Status** pull-down menu item allows for the display of detailed information about the available entities (User, Network Operator, Service Provider). The particular features (see Figures 5-7) that are displayed, depending on the selected entity, are:

- the state (created or not created)
- the identities of all the involved entities (obtained by the registration process)
- the public key agreement key
- the secret key agreement key
- the public signature key
- the secret signature key
- the session key, and
- the identity of the Certification Authority.

7.2.4.The Demonstration Part

The **Demonstration** part concerns the “MONITOR” and the “TRACER” options (Fig. 8).

The *Monitor* option enables the user to have an overall view of the message flow between the involved entities, while the selected protocol is running. By highlighting the *Demonstration -> Monitor* menu item, a cascaded list appears and the user is able to select between the new or the current registration scenario to be demonstrated. In both cases, it is assumed that a roaming agreement exists between the Network Operator and the User’s Service Provider. Following, the user is able to select an authentication mechanism (Siemens or RHUL mechanism). The message flow of each mechanism is presented in Figures A, B, C and D.

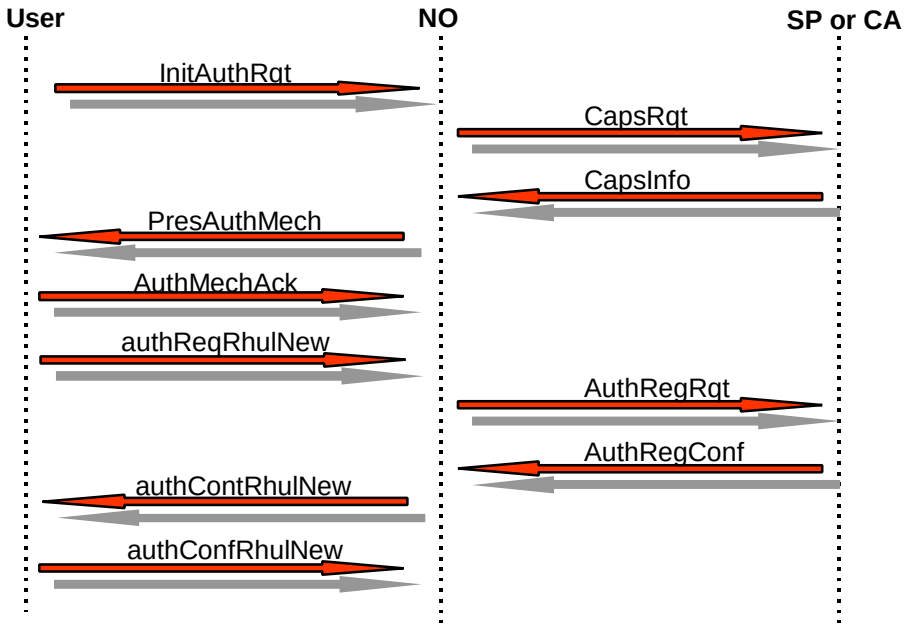

Figure 7-1 New Registration: Symmetric key authentication

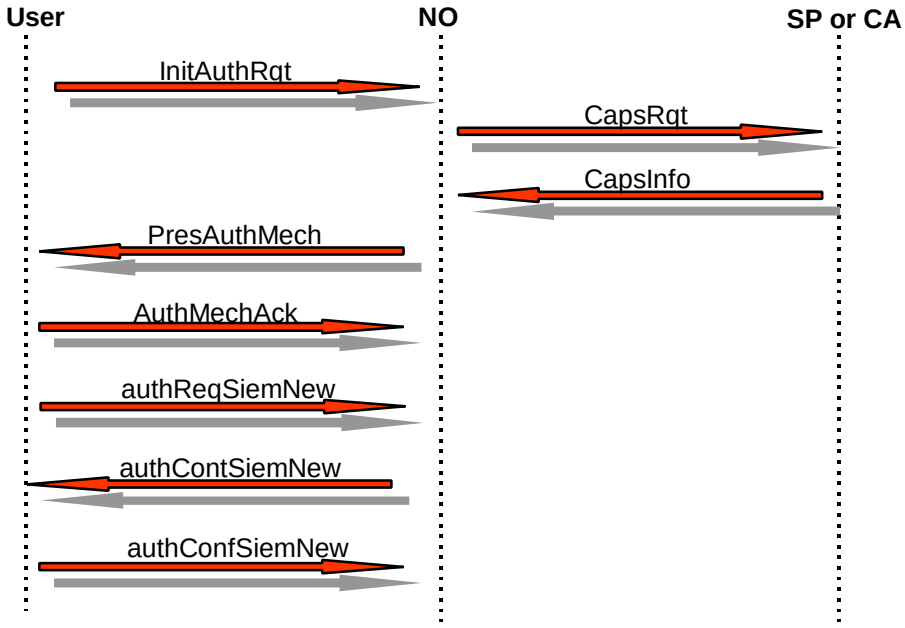

Figure 7-2 New Registration: Public key authentication

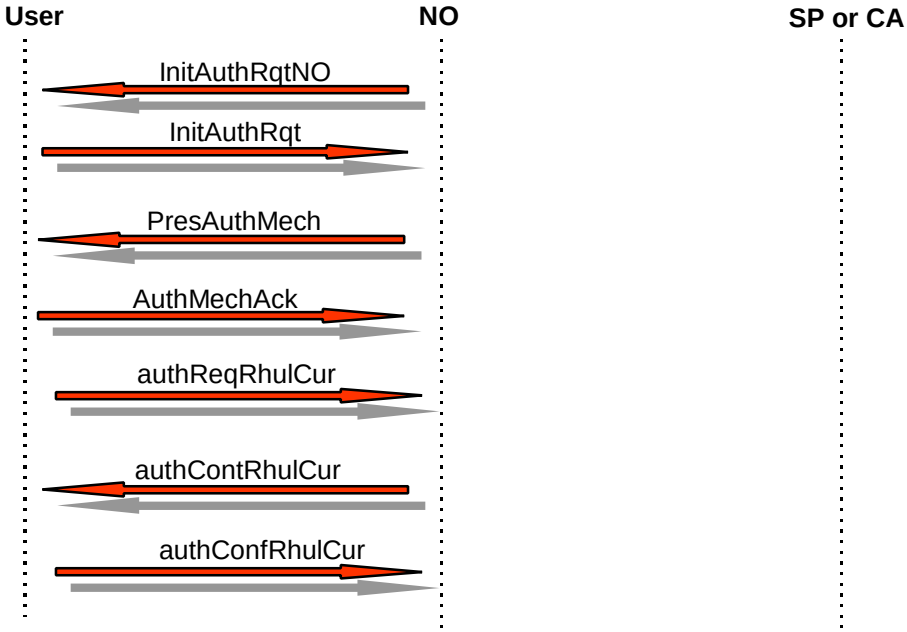

Figure 7-3 Current Registration: Symmetric key authentication

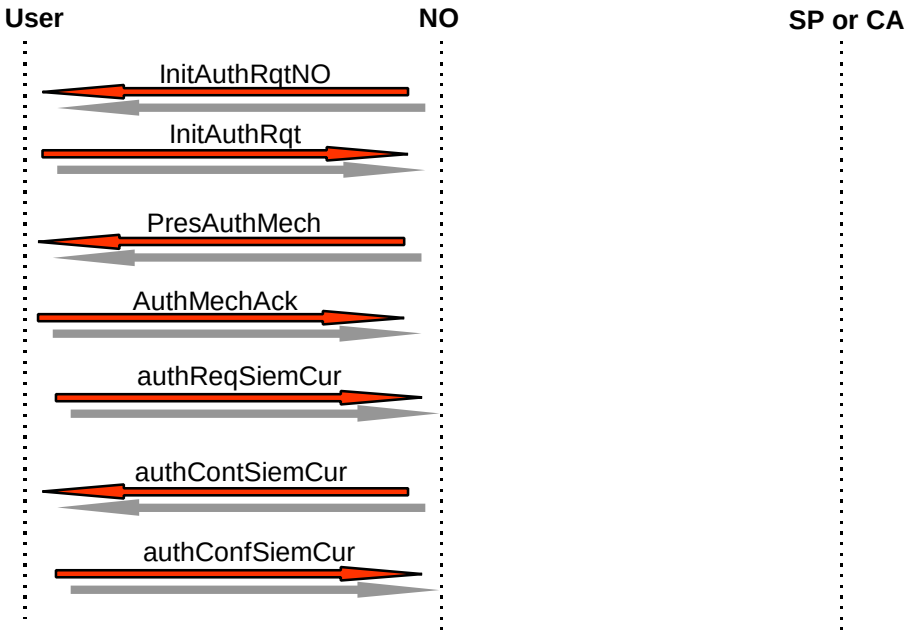

Figure 7-4Current Registration: Public key authentication

The message flow can be interrupted at any time by pressing the stop button. The entities are visually represented by their respective bitmaps and different background colour is used for the bitmaps that correspond to created entities.
The **Demonstration** pull-down menu includes also the *Tracer* option, enabling the user to observe the message exchange process as well as the status of each created entity. Upon selecting the *Tracer -> Start* option (Fig. 8), the Tracer window appears in the right side of the screen (as shown in Fig.9), giving detailed information about the three participants' actions described below.

7.2.5. The Action Part

When the process of entities selection and identification is completed, the **Action** menu item (see Fig. 9) can initiate the authentication process.
The involved entities' available actions include:

- The *User A -> Start Registration* action
- The Network Operator's actions, comprising:
  - *Start Current Registration*
  - *Deregister User*

7.2.6. The Help Part

The **Help** pull-down menu options will enable the user to search through the *Contents,* to *Search for …* a specific topic and to be informed *About* this demo. The contents of these options are not incorporated in this version.

7.3.Figures

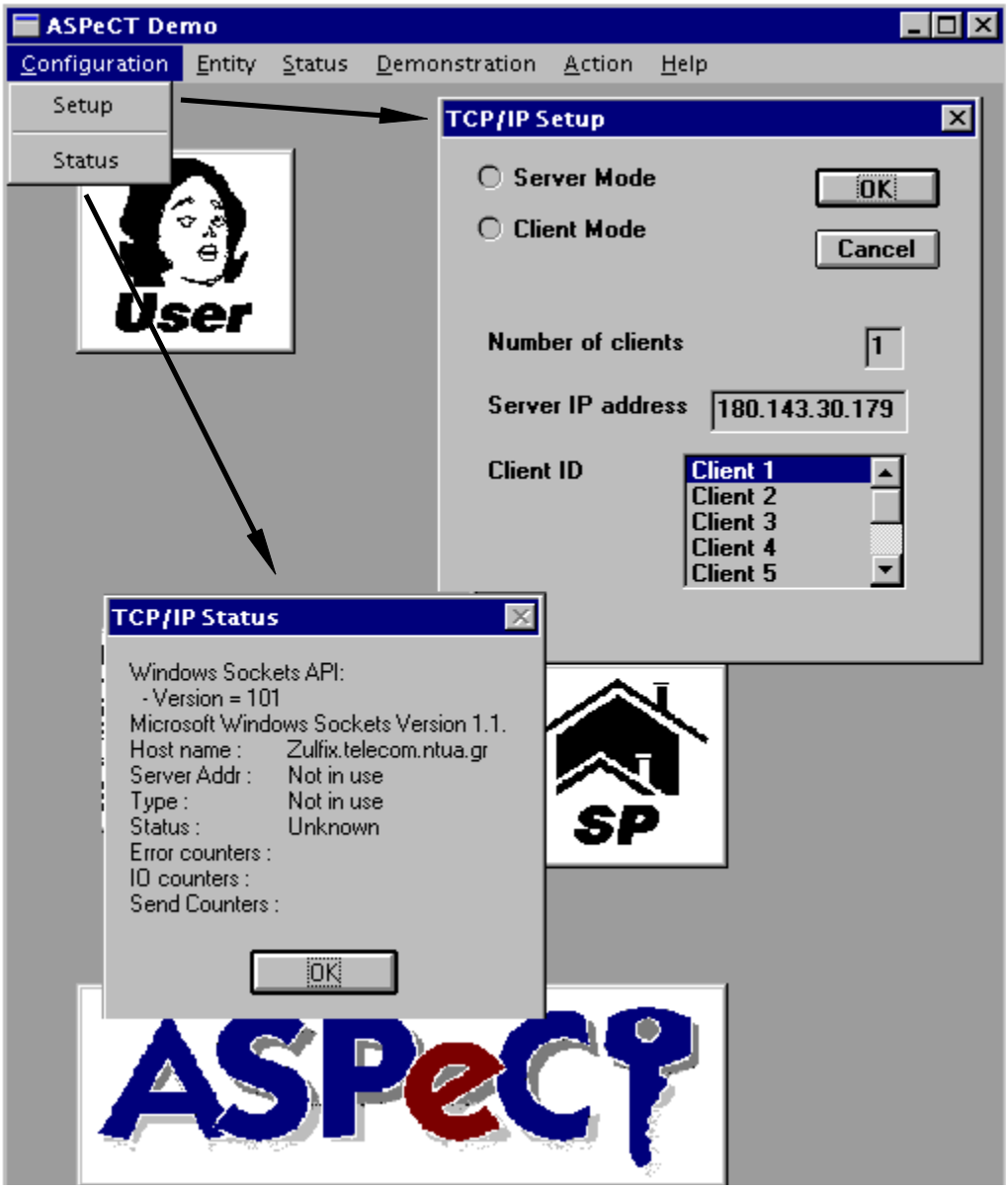

Figure 7-5The *Configuration* pull-down menu

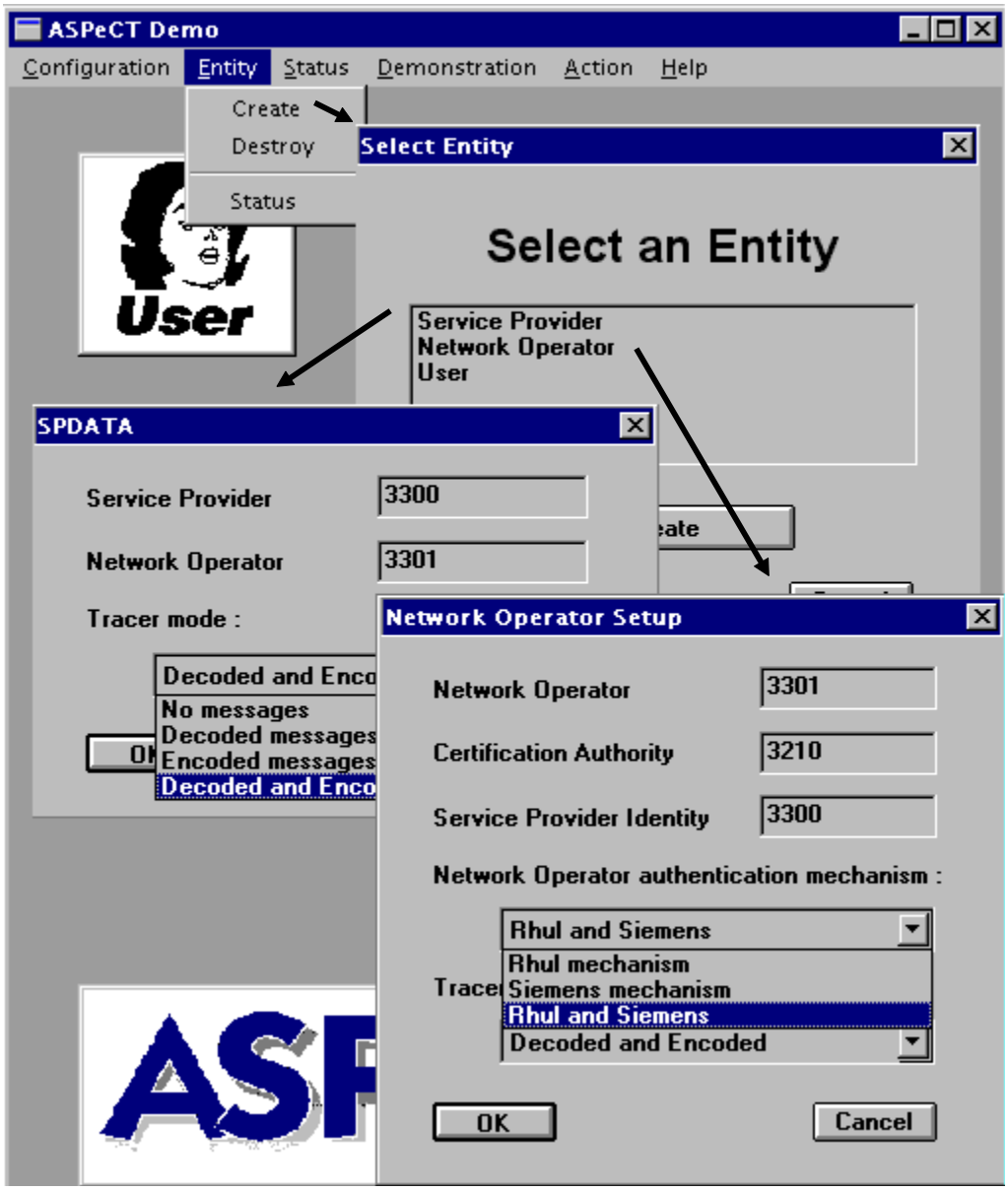

Figure 7-6The SP and NO creation process

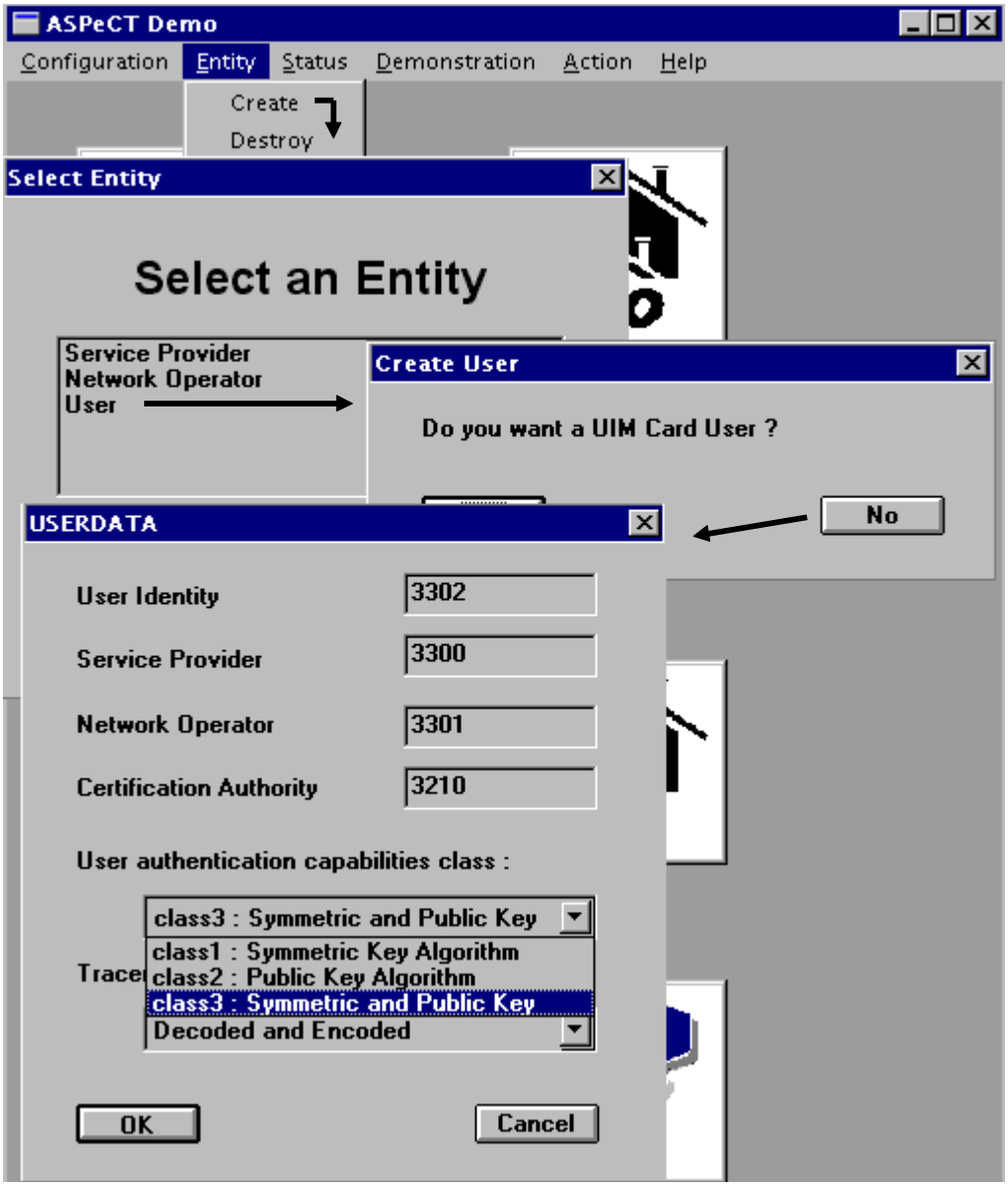

Figure 7-7 The *User's* creation process

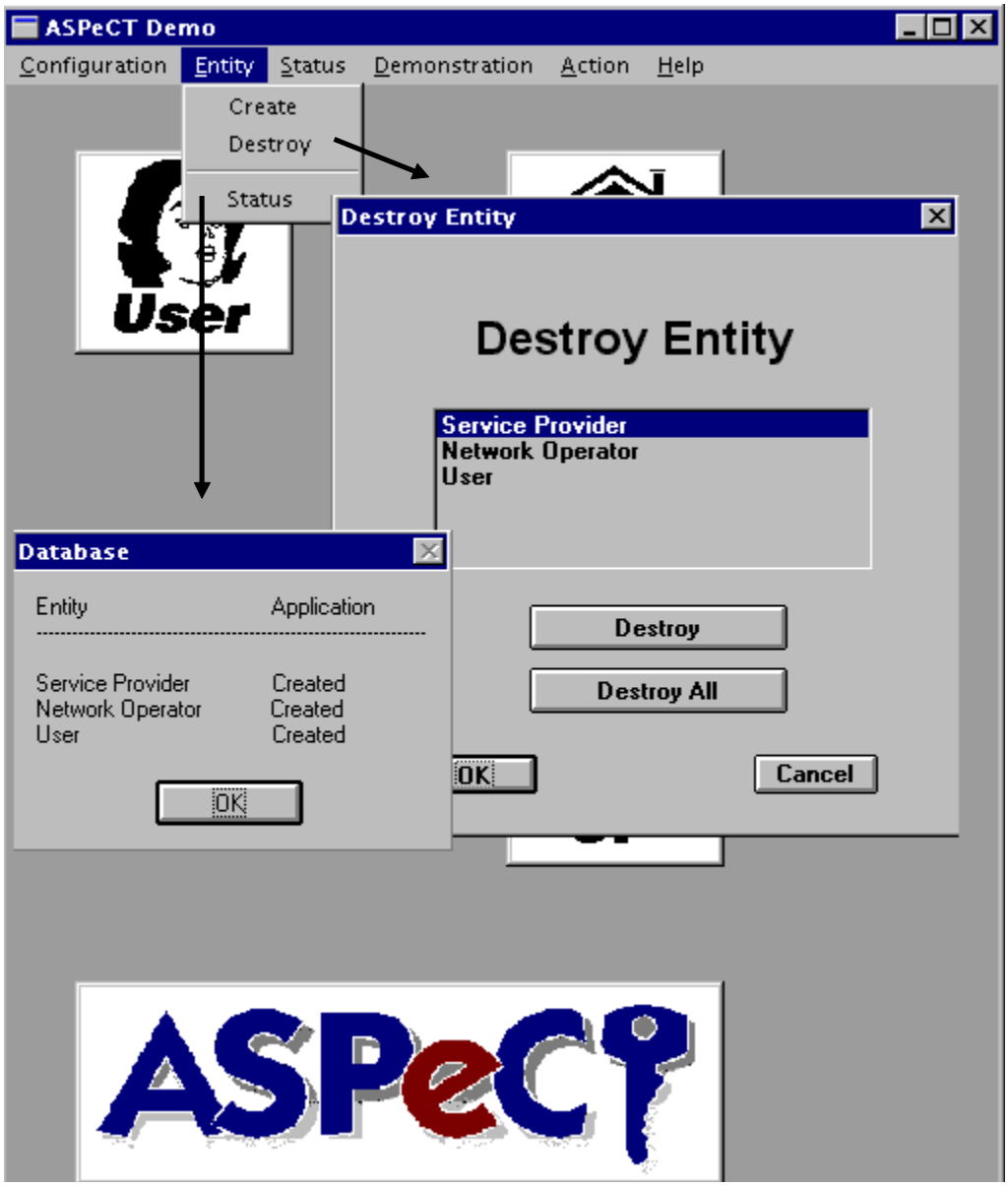

Figure 7-8 The *Entity-> Destroy* and *Status* menu items

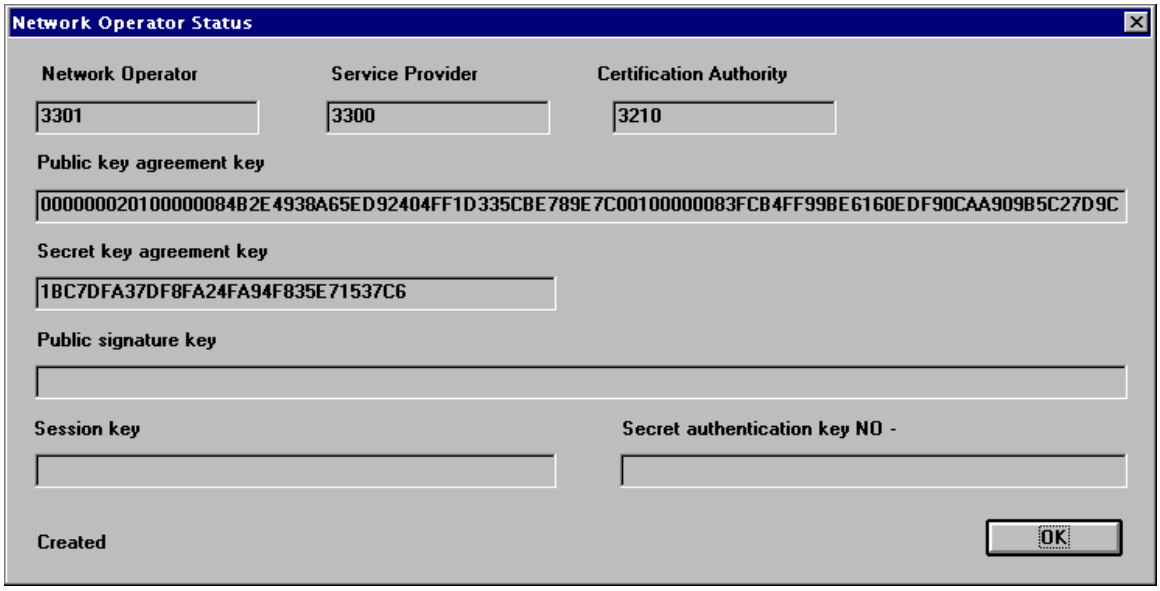

Figure 7-9The *NO* Status dialog box

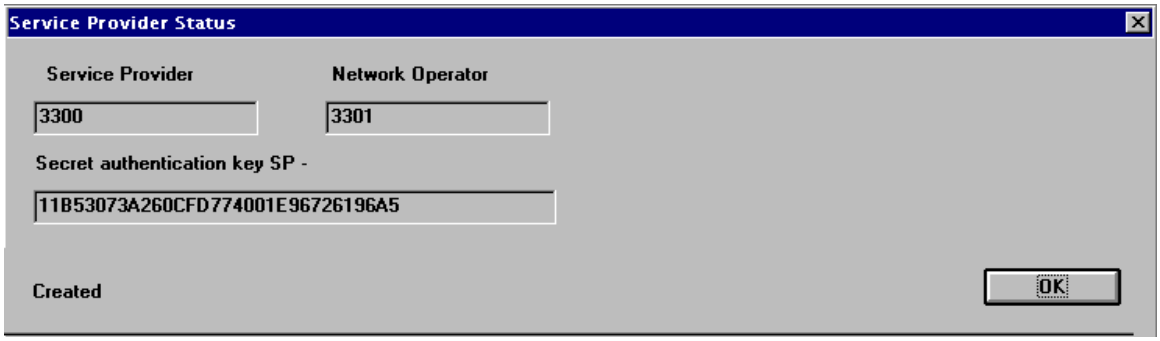

Figure 7-10 The *SP* Status dialog box

Figure 7-11 The *User* Status dialog box

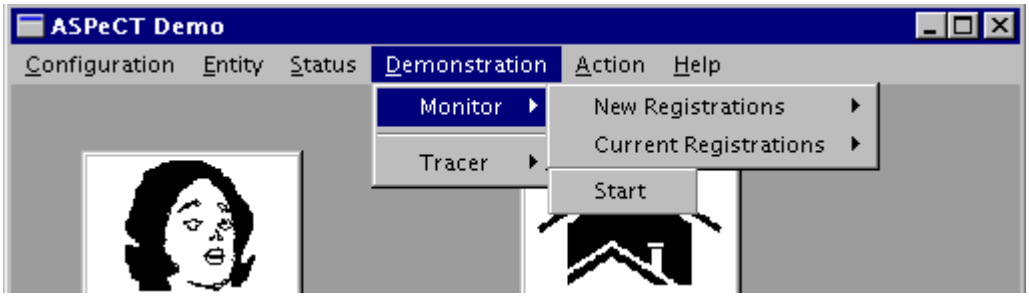

Figure 7-12The Demonstration **pull-down menu**

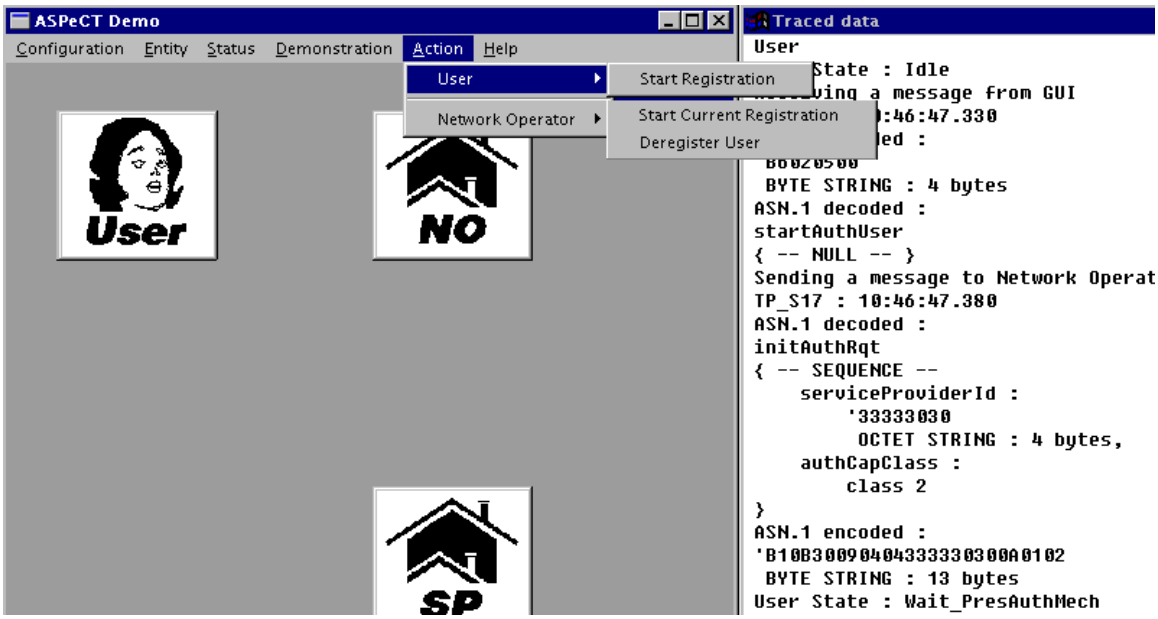

Figure 7-13The Action **pull-down menu**

8.Tests of the demonstrator

8.1.Introduction to Testing

The successful operation of either the Siemens public-key or the RHUL secret-key authentication mechanism demands that the correct messages and information are passed between either the User, the Network, or a third party which could either be a Service Provider or a Certification Authority.

It is not only necessary to ensure that the demonstrations operate correctly, but that they are also capable of correctly handling those situations where the expected message or information is not received.

8.2.Testing Performed

To this end, the following table contains details of those scenarios which have been examined. Listed below is the protocol against which the test applies and a brief description of the test.

Protocol	Description
Siemens New Registration	A wrong Certification Authority Identifier is entered via the GUI. The UIM sends a wrong Certification Authority Identifier in the first message. The Network Operator should send an Authentication Reject Message.
Siemens New Registration	The user and the network operator are given a public key agreement key and a secret key agreement key which do not correspond to each other. The verification of the authentication value at the user side fails. The user sends back an Authentication Reject message.
Siemens New Registration	A wrong Certificate is entered via the GUI in the Network Operator. The signed part of it is modified. The verification at the user-side fails. The user sends back an Authentication Reject message.
Siemens New Registration	A user signature and a certificate on the user's public key is passed to the network operator. The certificate's validity date is not valid. The Network operator notices this, and sends back an Authentication Reject message.
Siemens New Registration	The user fills in a different data1 field in message M3, than the one received in message M2. The Network Operator sends back an Authentication Reject message.
Siemens Current Registration	The same as 2.1.1.2, except that the User is already registered with the Network Operator.
Siemens	The same as 2.1.1.4, except that the User is already

Current Registration	registered with the Network Operator.
RHUL New Registration	The Service Provider is given a different Service Provider-User key. The User notices this after calculating the authentication value, and sends back an Authentication Reject Message. The tester should also verify that the new temporary identity received at the user side is not equal to the one generated by the Service Provider. This is because the ciphering key is not correct as it also depends on the Service Provider-User key. The tester should also check that the temporary identity received at the user side is not stored. The temporary identity should retain its old value.
RHUL New Registration	The service provider and user are given a different Service Provider-User key. This results in the calculation of a different Network-User key. The Network Operator notices the corrupt Authentication value, and sends back an Authentication Reject Message.
RHUL New Registration	The temporary identity sent by the User is not known to the Service Provider. The Service Provider sends an Authentication Reject Message to the Network Operator. The Network Operator sends an Authentication Reject Message to the User.
RHUL Current Registration	The Network Operator is given a different Network-User key. The User notices this after calculating the authentication value, and sends back an Authentication Reject Message. The tester should also verify that the new temporary identity received at the user side is not equal to the one generated by the Network Operator. This is because the ciphering key is not correct because it also depends on the Network-User key. The tester should also verify that the new ciphering key calculated at the user side is different than the one calculated at the network operator side. This is because the ciphering key is not correct because it also depends on the Network-User key. The tester should also check that the temporary identity received at the user side is not stored. The temporary identity should retain its old value.
RHUL Current Registration	The service provider and user are given a different Service Provider-User key. This results in the calculation of a different Network-User key. The network operator notices the corrupt Authentication value and sends back an authentication reject message.
RHUL Current	The user send a corrupt temporary identity to the network operator. The Network Operator sends back

Registration	an authentication reject message.
Authentication Framework - New Registrations	The user is given a service provider identification which is not known to the network operator. The Network Operator can't send a Capability Class Request message so it sends back an authentication reject message to the user.
Authentication Framework - New Registrations	The user is given a service provider identification which is not known by the network operator. The initial message from user to network operator contains an authentication capability class which is not known to the service provider. The network operator sends a Capability Class Request message to the service provider (which is the normal procedure). The service provider notices the unknown authentication capability class and sends back an authentication reject message to the network operator. The Network Operator sends the authentication reject message to the user.
Authentication Framework - New Registrations	The Network Operator doesn't support any of the mechanisms prescribed by the service provider. The Network Operator sends an authentication reject message to the user.
Authentication Framework - New Registrations	The network operator prescribes a mechanism which was not in the list received from the service provider. The user responds with a negative AuthMechAck message.
Authentication Framework - Current Registrations	Initiated by the User. All scenarios that can be imagined are covered under tests of the authentication framework new registrations.
Authentication Framework - Current Registrations	Initiated by the Network Operator. All scenarios that can be imagined are covered in previous tests.
Authentication Framework - Cryptographic Error	This tests the case when the Cryptographic function returns an error.
Authentication Framework - Illegal Event	This test the case when either an unexpected, or just wrong message is received.
Communication Between Entities	The protocol is executed between three entities (User, Network Operator and Service Provider) all created on the same PC in the same application. The PC-internal communication is used. The protocol should be executed successfully.
Communication Between Entities	The three entities are all created on the same PC but TCP/IP communication is used between the different entities. The protocol should be executed successfully.

Communication Between Entities	The three entities are all created on a different PC. The protocol should be executed successfully.
Communication Between Entities	A combination is used of PC-internal communication and TCP/IP communication.
Communication Between Entities	Six entities are created on 6 PC's (two users, two Network Operators and two Service Providers). The authentication protocol is executed twice. Once between the A's, once between the B's.
Communication Between Entities	One of the entities can't be reached due to an error in the communication. E.g. the entity is not created or one of the TCP/IP connections was broken down. The protocol is started but the tester receives an error indication. After the test the status of the involved entities is Idle.
Demonstration Stability	The UIM is removed from the card reader at various points throughout the message cycle. At each point, all of the involved entities should enter the idle state.
Demonstration Stability	Loss of TCP/IP connections. These can be tested in the same manner as the UIM removal.

Table 8-1 Description of Testing

8.3.Results of Testing

The above tests cover all of the areas where it was thought that possible errors could occur.
All of the above tests were successfully executed.

However, in saying this, there were some mistakes in the protocol which were highlighted by the above tests, but were then corrected, retested and passed.

eg. The Service provider was behaving wrong after the test with an unknown temporary identity
eg. The keys displayed in the GUI were the wrong information bytes

This testing was useful in ensuring that the demonstrator operated correctly in the areas defined above.

8.4.Testing the UIM

As described before, the UIM will be a dual-application smart card with both GSM and UMTS on it. Therefore, testing of such a card includes both the GSM and UMTS functionality.

- Testing the GSM side : it shall be possible to set up at least basic voice calls. This can be done with special GSM accounts over a live GSM network.

Depending on the level of supported GSM Phase 2 services, it shall also be possible to test for SMS functionality or other rudimentary services.

- Testing the UMTS side: independent of the particular authentication protocol (Siemens vs. Royal Hollway), the UIM shall support the security related card commands as specified in D15 [7]. It may not be possible to have both protocols and GSM on one smart card, but each protocol can be tested independently in the demonstration environment as described in this deliverable.

A test is passed successfully if the smart card behaves as specified in D15 and works correctly together with the simulation of the UMTS network.

9.Evaluation of the demonstrator

The remainder of this document deals with a detailed evaluation of the first UMTS authentication demonstrator. This evaluation is divided into seven subsections as follows.

1. ***Functionality.*** An overview of the authentication protocols implemented in the demonstrator, including analysis of protocol features, discussion of possible extensions and comparisons with alternative protocols.
2. ***Performance.***
3. ***Security Aspects.*** A review of the services, security functions, internal operations and interfaces of the first UMTS authentication demonstrator.
4. ***Applicability*** to other environments.
5. ***Architecture.*** An overview of the entire architecture of the first demonstrator.
6. ***Test environment***
7. ***User Friendliness.*** A discussion of the features and suitability of the Graphical User Interface.
8. ***Appearance of Demonstration.*** An analysis of appearance of the first UMTS authentication demonstrator.

9.1.Functionality

This section provides an evaluation of the functionality contained within the demonstrator. It contains a brief overview of this functionality, and what parameters are important to its operation. It then describes how an ideal system should operate, and evaluates the current demonstrator against this. Finally, enhancements to the demonstrator are considered so as to bring it closer to ideal operation.

9.1.1.Overview of the demonstrator functionality

The ‘Migration Scenario Demonstrator’ provides an environment within which the proposed ASPeCT authentication mechanisms can be tested and evaluated. The demonstrator consists of the software through which the proposed ASPeCT authentication mechanism is implemented, along with a Graphical User Interface (GUI), which controls the operation of the demonstrator.
It is implemented as a Finite State Machine so that the state of each entity within the demonstrator is always defined. The passing of a message between any two entities is reflected in a change of state for one or both of these entities.
This allows the protocol to be exactly controlled, but means that the system can only support one message at a time.
Through the GUI, the User is able to initiate the authentication mechanism, and the process continues through, using either protocol, to the point where the User is registered.

To operate the demonstrator, the GUI is first used to determine whether it should operate in server or client mode, then to create the User, Network Operator and

Service Provider entities required for operation, and finally to begin authenticating the User. In addition, the GUI allows the selection of which protocol is to be used. The messages that are then generated during authentication and their times of production are displayed on a Tracer window, which can also be printed to a file for later analysis.

The demonstrator allows the testing of the negotiation framework, the authentication mechanism, and some assessment of relative timings between the different authentication mechanisms implemented.
In addition, the demonstrator allows for a clean definition of what is expected and what is unexpected behaviour, which allows the handling of errors to be investigated.

An important component of the demonstrator is the implementation of the negotiation framework. This framework allows a negotiation to take place between the User and the Network to determine which authentication protocol to use. Hence a range of protocols can be made available to both the User and the Network, whilst leaving the Network free[2] to choose which protocol is actually used. Another advantage of this mechanism is that the User's identity is not given until after the choice of protocol has been made. Thus the negotiation can be performed with anonymity for the User.
The demonstrator is the first realisation of this framework, and is important in evaluating a real implementation of this in terms of its performance and suitability for actual use.

The demonstrator supports two distinct protocols, one proposed by Siemens and one proposed by Royal Holloway, University of London. Both have been submitted to ETSI for UMTS use [6].
The Siemens protocol uses a public-private key pair, whilst the Royal Holloway protocol uses a secret key. Detailed information on the proposed protocols can be found in D2 [1].

9.1.2.Parameters and requirements of the demonstrator

This section describes some of the parameters and variables used within the demonstrator.

The purpose of the negotiation framework is to establish which authentication protocol to use. It does this by use of the CapsClass parameter. The CapsClass parameter is a value which represents a class of protocols which are available to the User. If this representation is not known by the Network Operator, then this information is retrieved from the Service Provider. Once known, then the Network Operator chooses a protocol to use, and the mechanism continues. There are three values of CapsClass used in the demonstrator; 1 which corresponds to Siemens, 2 which corresponds to RHUL, and 3 which corresponds to both Siemens and RHUL.

[2] The network isn't entirely free; it must be a protocol that the user agrees to (otherwise a fake network could choose the null authentication protocol).

In addition to the CapsClass parameter, there also exist a number of variables which are used to record the state of the system at any given instant in time. Hence, the state of each entity within the demonstrator is always known, and is one of a finite set of states.
For example, when the demonstrator is first started, all entities are in an *idle* state. Once a request to begin authentication is sent from the User to the Network Operator, then the User's state changes to *Wait_PresAuthMech*, which indicates that the User is waiting for the Authentication Mechanism choice message from the Network Operator.

9.1.3.Desirable operation of the demonstrator

If the system is operating properly, then a request sent from the User to the Network Operator to begin authentication should produce a series of messages between the User, Network Operator and Service Provider. These should lead to the final state where the User and Service Provider are idle, and the Network Operator is in the User Registered state. In addition, this should occur with efficiency, so that no messages are redundant, and so that no useless information is passed through the network.

In addition, if the Network Operator has information about the CapsClass, then it need not contact the Service Provider. Proper operation of the demonstrator would mean that the Network Operator would only contact the Service Provider if the CapsClass information was not available.

9.1.4.Evaluation of the demonstrator

The proposed approach of using the negotiation framework to determine which protocol to use works well. For relatively little additional traffic, a high degree of flexibility has been introduced into the authentication process. Furthermore, this process can take place anonymously, which means that there are fewer unencrypted User Identities being passed around the network.
In particular, having a demonstrator allows a clear assessment of the proposed mechanisms and framework. This is invaluable in allowing people to gain an understanding of how the mechanism works.

The negotiation framework operates well in terms of a secure mechanism with which to establish an authentication protocol. However, there is one message within the negotiation framework which is redundant, and this is the acknowledgement that is sent from the User upon receiving notification of which authentication method to use. The reason why this is redundant is that the User then immediately also sends an authentication request using the stated protocol, so that if there was an error in receiving the method notification, then instead of starting the authentication process, an authentication reject message can be sent.

The demonstrator described above is the first implementation of the negotiation mechanism that will eventually be trialled and assessed with EXODUS. As the

above framework mechanism operates well, then there is no foreseeable problem to it being used in the trial, with the possible amendment that the acknowledgement message mentioned above is removed.

9.1.5.Enhancements to the demonstrator

The demonstrator operates well. However, this section discusses some possible enhancements that could be made to it.

As stated above, the authentication mechanism acknowledgement message should be removed. However, as this does not effect the overall behaviour of the demonstrator, then the authentication protocol should be changed, but the demonstrator itself left unchanged.

As above, in terms of improved functionality for the demonstrator, although more functionality could be added, the demonstrator achieves its purpose, and there seems no point in increasing the functionality of this.
However, if a second demonstrator was required as a half way point between the current specification and the trial specification, then some extra functionality could be added. This includes such things as defining and implementing messages to update the CapsClass information at both the User, and the Network Operator.

9.2.Performance

In the following sections the processing delays at the different entities is measured for new registration and current registration with the public key authentication mechanism. Measurements were done with the PC realisation of the user (the first release of the smartcard does not complete all calculations and is 4 times slower).

Measurements are done by gathering timing information in the logging file.

The data to be saved in the different entities and to be transferred is compared to the amount of data needed in GSM.

The Acryl 16 bit version is used at the network side and the user side.

The architecture for the measurements were 2 PCs , connected via a local TCP-IP connection. The PCs contain a INTEL 486 33,4 Mhz processor.

For more information on the evaluation of the UIM, refer to [7].

9.2.1.Processing delays compared to total authentication time

In the following paragraphs the processing delays are measured in each entity and compared to the total authentication time.

9.2.1.1.At the user's side

9.2.1.1.1.New registrations

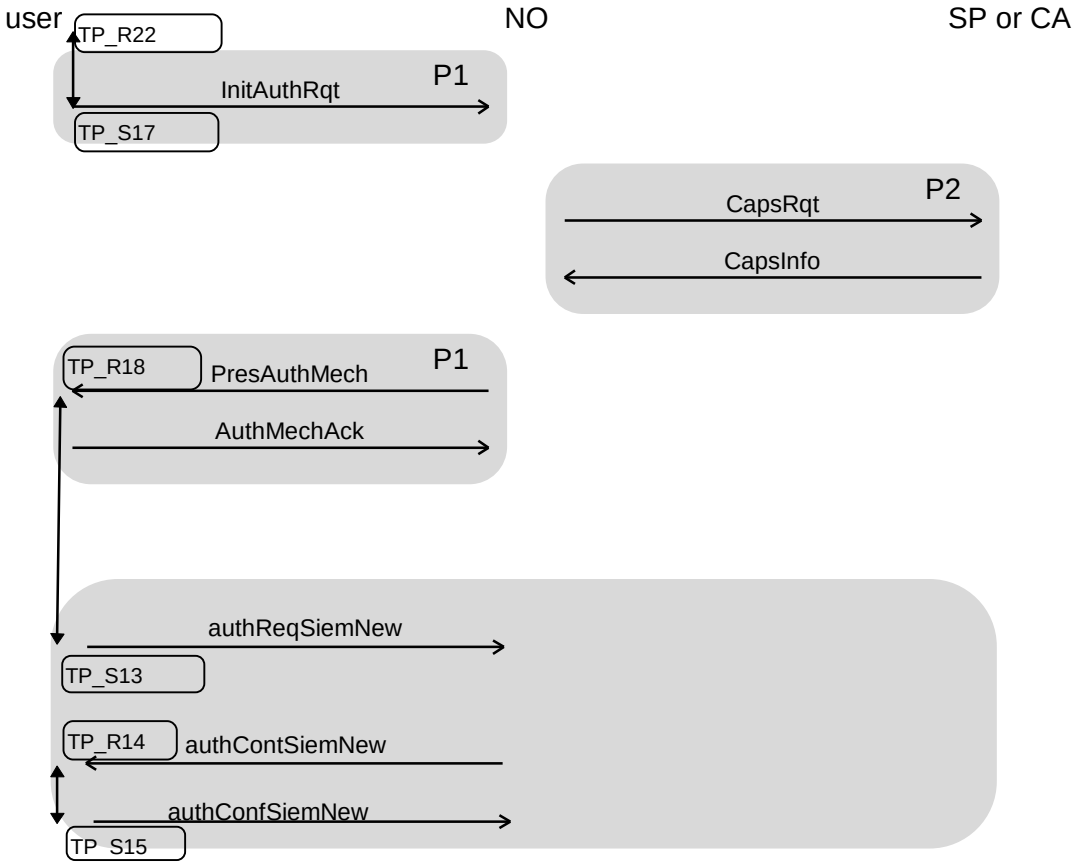

Figure 9-1 User - New registration flow

TP_S17	TP_R22	UserPD01
43.390	43.390	0.0
08.030	07.920	0.110
35.930	35.880	0.050
04.550	04.490	0.060

Table 9-1User - Start of authentication

The mean time to start the authentication is 55 msec.

TP_S13	TP_R18	UserPD02	TP_S15	TP_R14	UserPD03	Total
46.250	44.320	1.930	55.640	48.550	7.090	9.020
10.890	09.130	1.760	20.330	13.250	7.080	8.840
39.010	37.200	1.810	48.400	41.370	7.030	8.840
07.570	05.700	1.870	17.070	09.880	7.190	9.060

Table 9-2 User - Authentication

The mean time to do the authentication is 8.940 msec.

TP_S15	TP_R22	total authentication time

55.640	43.390	12.250
20.330	07.920	12.410
48.400	35.880	12.520
17.070	04.490	12.580

Table 9-3 User - Total authentication time

The mean for the total authentication time is **12.440 msec**.

The user needs **72 % processing time** of the total authentication time.

9.2.1.1.2.Current registrations

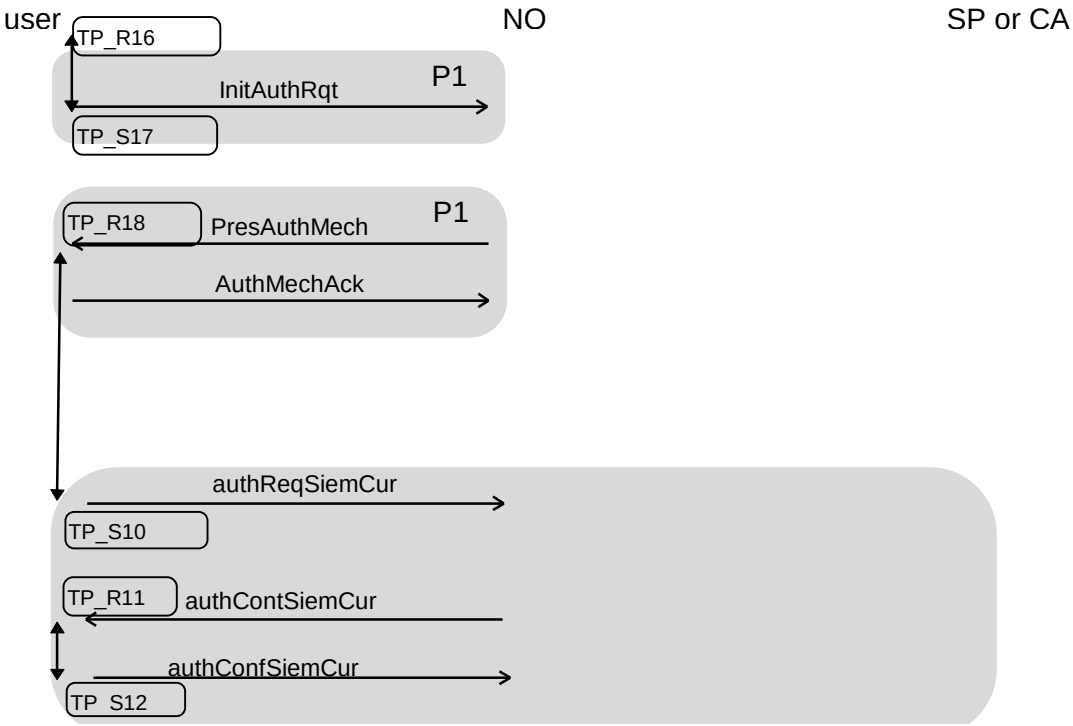

Figure 9-2 User - Current registration flow

TP_S17	TP_R16 / TP_R22	UserPD10
11.240	11.180	0.060
27.390	27.330	0.060
43.860	43.810	0.050
29.650	29.600	0.050

Table 9-4 User - Start of authentication

The mean time to start the authentication is 55 msec.

TP_S10	TP_R18	UserPD04	TP_S12	TP_R11	UserPD05	Total
13.380	11.620	1.760	19.090	15.520	3.570	5.330
29.470	27.720	1.750	35.190	31.730	3.460	5.210

46.170	44.250	1.920	51.830	48.310	3.520	5.440
32.010	30.030	1.980	37.780	34.210	3.570	5.550

Table 9-5 User - Authentication

The mean time to do the authentication is 5.380 msec.

TP_S12	TP_R16 / TP_R22	total authentication time
19.090	11.180	7.910
35.190	27.330	7.860
51.830	43.810	8.020
37.780	29.600	8.180

Table 9-6 User - Total authentication time

The mean for the total authentication time is **7.992 msec**.

The user needs **68 % processing time** of the total authentication time.

9.2.1.2.At the network operator's side

9.2.1.2.1.New registrations

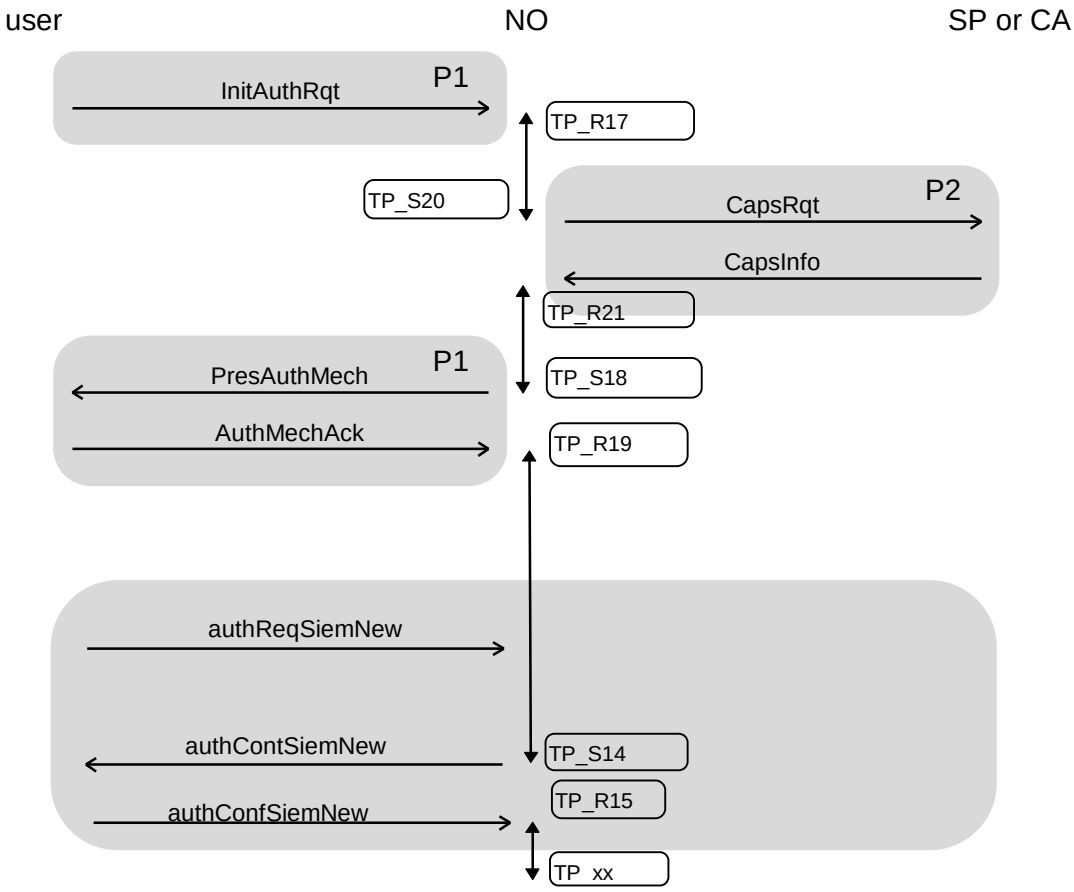

Figure 9-3 Network Operator - New registration flow

TP_S20	TP_R17	NOPD01	TP_S18	TP_R21	NOPD02	Total
43.010	43.010	0.0	43.720	43.610	0.090	0.090
07.760	07.650	0.110	08.580	08.470	0.110	0.220
35.710	35.600	0.110	36.590	36.480	0.110	0.220
04.270	04.160	0.110	05.100	04.990	0.110	0.220

Table 9-7 Network Operator - Start of authentication

The mean time to start the authentication is 188 msec.

TP_S14	TP_R19	NOPD03
47.780	45.920	1.860
12.480	08.910	3.570
40.600	36.920	3.680
09.160	05.430	3.730

Table 9-8 Network operator - Authentication

The testpoint TP_xx is not available in the demo. Therefore the time to check the signature at the network operator's side is not measured.

An estimation of the missed time is UserPD03 (7.000 msec).

The mean time to do the authentication is 10.210 msec.

TP_R15	TP_R17	total authentication time
55.420	43.010	12.410
20.110	07.650	12.460
48.180	35.600	12.580
16.800	04.160	12.640

Table 9-9 Network Operator - Total authentication time

The mean for the total authentication time (inclusive the estimated time) is **19.522 msec**.

The network operator needs **53 % processing time** of the total authentication time.

9.2.1.2.2.Current registrations

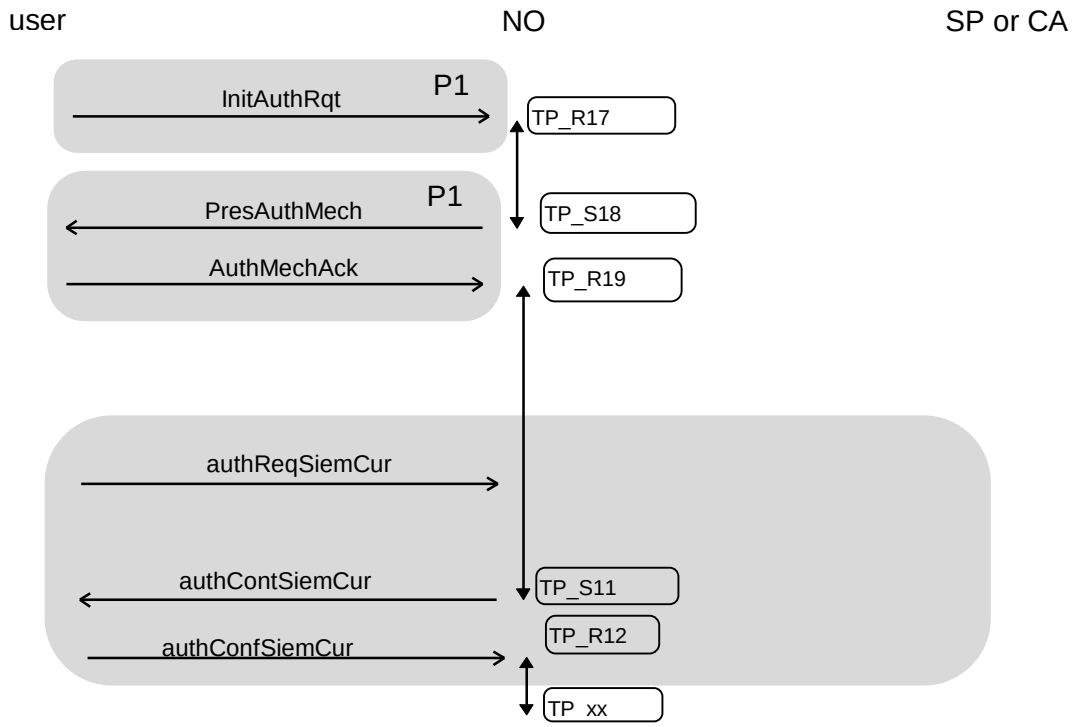

Figure 9-4 Network Operator - Current registration flow

TP_S18	TP_R17	NOPD12
11.020	10.910	0.110
27.110	27.000	0.110
43.700	43.590	0.110
29.430	29.380	0.050

Table 9-10 Network Operator - Start of authentication

The mean time to start the authentication is 95 msec.

TP_S11	TP_R19	NOPD05
14.920	11.350	3.570
31.070	27.440	3.630
47.650	43.970	3.680
33.550	29.820	3.730

Table 9-11 Network operator - Authentication

The testpoint TP_xx is not available in the demo. Therefore the time to check the signature at the network operator's side is not measured.

An estimation of the missed time is UserPD05 (3.500 msec).

The mean time to do the authentication is 7.152 msec.

TP_R12	TP_R17	total authentication time

18.760	10.910	7.850
34.860	27.000	7.860
51.500	43.590	7.910
37.450	29.380	8.070

Table 9-12 Network Operator - Total authentication time

The mean for the total authentication time (inclusive the estimated time) is **11.422 msec**.

The network operator needs **63 % processing time** of the total authentication time.

9.2.1.3.At the service provider's side

9.2.1.3.1.New registrations

The processing time of the service provider can be neglected.

9.2.1.3.2.Current registrations

The service provider is not involved in the current registrations protocol.

9.2.1.4.Conclusions

At the user's side the interface to the smartcard has to be optimised.

At the networks side the performance of the Authentication protocol has to be increased, by means of a better Acryl, a better processor, ..

The goal of the protocol, to use less calculation power at the users side is proven, the network operator needs 57 % more processing time then the user for new registrations and 43 % more for current registrations.

9.2.2.GSM - UMTS

9.2.2.1.Data to be saved

	GSM	UMTS public key before registration	UMTS public key after registration
AC/HLR or SP	Ki = 16 bytes		
MSC/VLR or NO	5 triplets = 140 bytes	secret and public key agreement keys = 16 + 48 bytes certification authority public key = 48	secret and public key agreement keys = 16 +48 bytes certification authority public key

			= 48 bytes public signature key= 48
SIM or UIM	Ki = 16 bytes	public and secret signature keys = 16 + 48 bytes certification authority public key = 48	public and secret signature keys = 16 + 48 bytes certification authority public key = 48 bytes public key agreement key = 48 bytes

Table 9-13 GSM -UMTS Data to be saved

The data in the boxes will be present for each subscriber roaming in the network.

9.2.2.2.Data transmitted over the air

	GSM	**UMTS public key new registration**	**UMTS public key current registration**
M1	rand= 16 bytes	key agreement value = 48 bytes CA id = 4 bytes	key agreement value = 48 bytes
M2	Sres = 4 bytes	randN = 16 bytes authN = 16 bytes certN = 134 bytes	randN = 16 bytes authN = 16 bytes
M3		encSign = 48 bytes encCertU = 136 bytes	encSign = 48 bytes

Table 9-14 GSM-UMTS data to be transmitted

9.2.2.3.Total authentication time

Within GSM the pure authentication message, needs less then 1 second.
In the ASPeCT demo, current version, 12,5 seconds are necessary when the user registers for the first time in the network.
For the following registrations 8 seconds are necesary.

9.2.3.Evaluation of the UIM

Different criteria are applicable to evaluate the UIM in the context of the first demonstrator:

- GSM side : the following Phase 2 services are deemed to be desirable

No.	Name	Comments	Phase
1	CHV1 disable	-	1
2	LP	One preferred language	2
3	SMS storage	One SMS record	2
4	MSISDN	One MSISDN	2
5	LND	One record	2

Table 9-15 Phase 2 services for the UIM

An evaluation of the GSM side is therefore a check which of the above services are supported.

- UMTS side: the most important criteria to evaluate the UMTS side is the execution speed of the different authentication commands. This has to be measured based on timers. It is of course also important that the UIM supports each command in its full functionality as specified in D15.

9.3.Applicability

The demonstrator is the first implementation of the negotiation framework which may be used in UMTS to establish which authentication protocol to use. This section will evaluate how applicable such an implementation is to a full UMTS implementation.

In a full UMTS framework, the negotiation framework provides a great deal of flexibility in determining which authentication protocol to use, for very little cost in terms of additional information transfer and maintenance.
In addition, the framework as implemented within the demonstrator, is concerned with establishing which authentication protocol is to be used with the messaging within the network. Hence, this framework can operate on any system where the transport layer can be considered to be invisible to the data layer.
However, the demonstrator does not differentiate between security for signalling and security for data. It is conceivable that, for some networks, encryption for data will be switched off. In this case, secure signalling is still going to be a requirement for UMTS.
The demonstrator may need to be expanded to include separately secured signalling information. In this case, authenticating switching information and authenticating data have to be considered as two separate requirements. This should be addressed later.

Hence, the demonstrator is a useful tool for studying the operation of the proposed negotiation framework, but that is the limit of its applicability.

9.4.Architecture

9.4.1.Introduction

This section describes the architecture of the first demonstration of the *migration scenarios* workpackage (WP2.1). The aim of the workpackage is to show the feasibility of the introduction of an authentication framework together with a set of authentication protocols in UMTS. Detailed specifications of the demonstrator are provided in [1,2].

9.4.2.Logical Architecture

The logical model of the demonstrator is shown in Figure 1. Three logical parts are involved namely:

- A user, represented by a UIM and a terminal, which is authorized by the subscriber to make use of the telecommunication services.
- A network operator which provides these services to the user.
- A service provider which is responsible for the provision of a service or a set of services according to the user's subscription and the negotiation of the network capabilities, associated with a particular service or a set of services, with the network operator.

Figure 9-5 ***Logical structure of the demonstration***

9.4.3. Physical Architecture

The physical structure of the demonstrator is shown in Figure 2. The physical structure consists of the smart card of the user (UIM), an intelligent card reader, and three PCs. The intelligent card reader interfaces directly with the smart card. Furthermore, it features a display and a keyboard similar to the conventional mobile phone. Additionally, there is a PC, containing the GUI and the code that realises the protocol among the network and the intelligent card reader. Finally, there are PCs for the NO and the SP. It is possible for a single PC to contain both the NO and the SP. In this case the two entities will communicate via an internal to the PC mechanism.

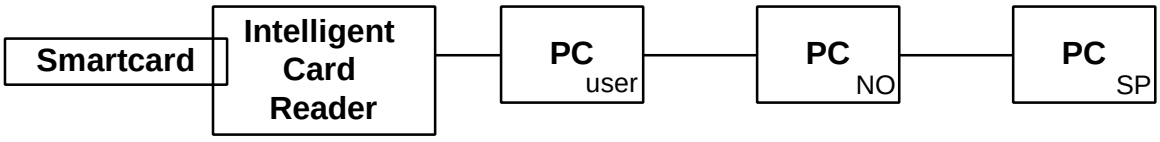

Figure 9-6*: Physical structure of the demonstration*

9.4.4. Mapping of the logical to the physical structure

The correspondence between the logical and the physical structure is depicted in Figure 3. The UIM, provided by the service provider, contains the user's profile and authentication data. The user accesses the network via a terminal which is connected with the ICR via a V24 interface. All PCs communicate via TCP/IP.

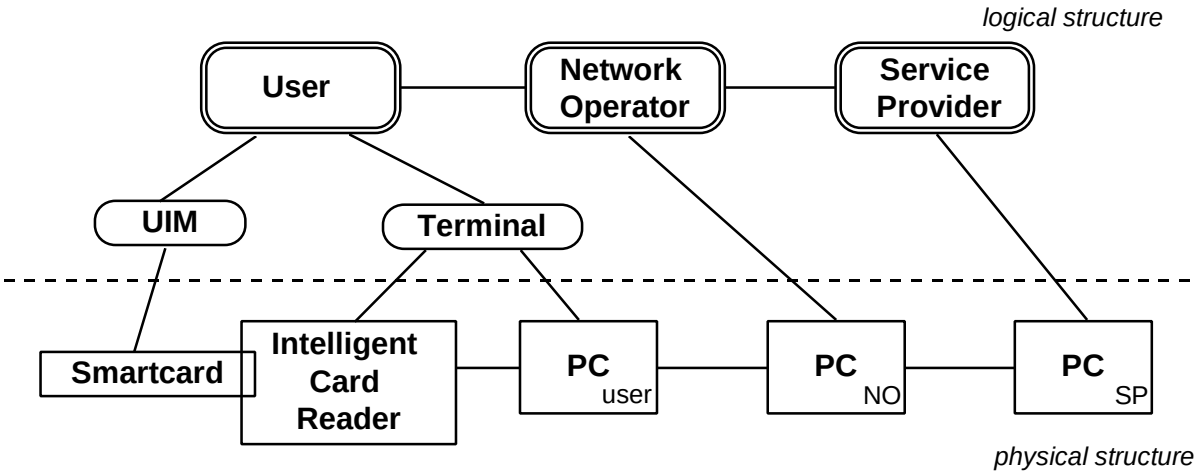

Figure 9-7 ***Mapping among logical and physical structures***

9.4.5.Software Architecture

The software architecture of the WP2.1 first implementation is shown in Figure 4. A brief description of the presented software blocks is given below. For a more detailed description of the software components the user is referred to [1,2].

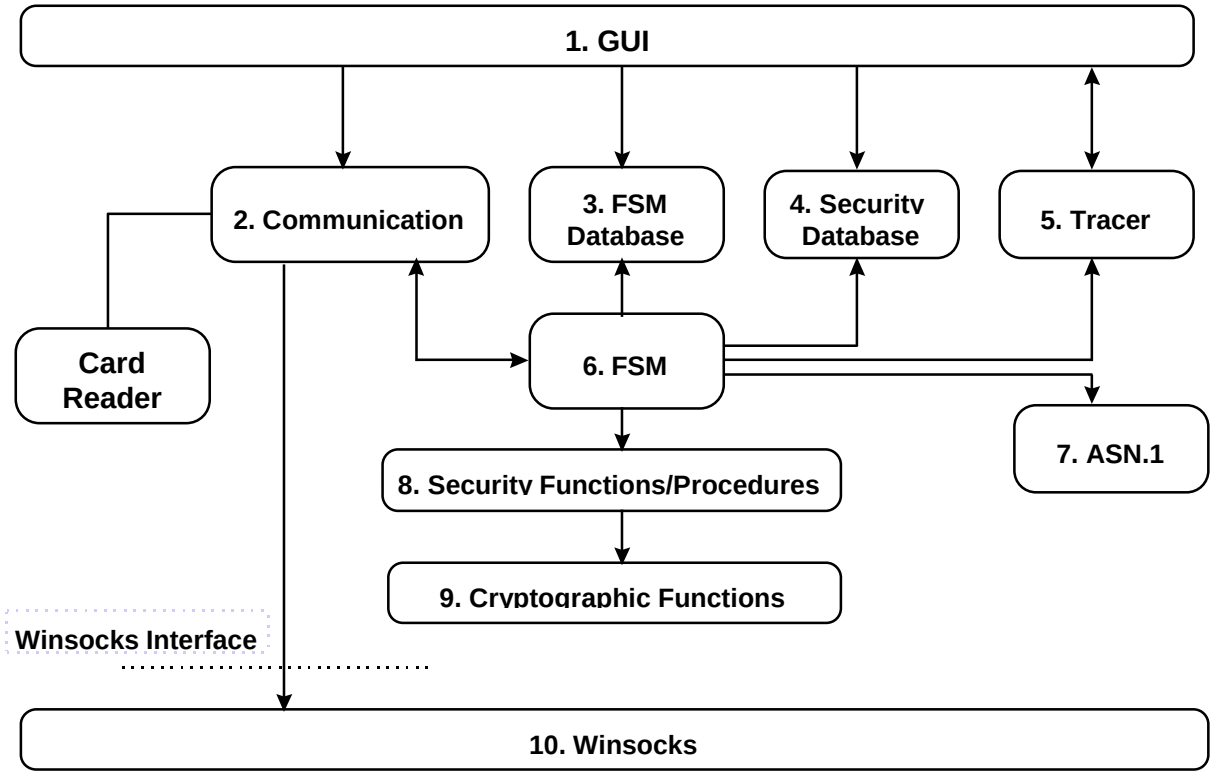

Figure 9-8 ***Software structure of the demonstrator***

9.4.5.1. Block 1: GUI

The Graphical User Interface has the following functionalities:

- The main application is defined in the GUI
- Administration of the communication database (create new entity, configure and activate TCP/IP connections)
- Allows the user to start the execution of the protocol
- Activates Tracer window

9.4.5.2.Block 2: Communication

The Communication block has the following roles:

- Includes the communication database containing all the data necessary for the communication
- Communicates to the serial port to which the intelligent card reader can be connected
- Establishes the TCP/IP connections at initialisation phase and exchanges messages with the other PCs via TCP/IP
- Enables the application-internal communication

9.4.5.3.Block 3: FSM database

All the information related to the finite state machines is stored in the FSM database block.

9.4.5.4. Block 4: Security Database

The security database holds parameters specific to the security layer and is administered via the GUI.

9.4.5.5. Block 5: Tracer

The tracer provides the user with detailed information on the experiments performed during the demonstrator. Information display in windows or logging in files is supported.

9.4.5.6. Block 6: Finite State Machines

The FSM block contains the finite state machines classes. A finite state machine instance represents one entity. One or more entities may be included in an application.

9.4.5.7. Block 7: ASN.1

The ASN.1 block contains C++ classes for ASN.1 definitions and procedures for encoding, decoding and displaying.

9.4.5.8. Block 8: Security Functions/Procedures

This software block builds an ASPeCT specific interface towards basic cryptographic functions.

9.4.5.9. Block 9: Cryptographic Functions

This block contains the basic cryptographic functions.

9.4.5.10. Block 10: Winsocks

This software block contains the standard winsocks libraries.

9.5.Test Environment

This section evaluates the test environment of the demonstrator, and in particular compares the test environment to the proposed trial environment.
The demonstrator constructs messages that it then sends between entities. In the trial system, connection between these entities may not be simple LAN connections, as, for example, the User-Network operator connection will use the EXODUS UMTS Air Interface. As the interfaces within the demonstrator are all simple TCP/IP connections, then it may appear that the final trial interfaces have not been considered. However, as the interfaces between the entities are a lower layer in the communication protocol, then these should not affect any changes to the negotiation framework.
An example of the above is that the EXODUS UMTS Air interface restricts the number of octets that can be passed across it in a single message. Although this restriction means that the messages between the User and the Network Operator will need to be cut up, the dissection and reassembly will all be visible to ASPeCT. In a full implementation, the messaging is also expected to be independent of the transport layer.

Part of the EXODUS trial consists of the measurement of certain performance criteria. The current test environment, provided by the demonstrator is not really suited to any form of performance evaluation of the negotiation framework.

However, for the restricted purposes of the demonstrator, it appears to be an adequate environment in which to operate such a system.

9.6.User Friendliness

9.6.1.Quality of Service

Among many definitions, it can be generally assumed that the Quality of Service is determined by the user's perception on the degree by which the service meets, or surpasses, the need it is designed for. It may also be defined by the level of overall user satisfaction regarding the provided service.
QoS is defined in [4] as follows: "The collective effect of service performance which determine the degree of satisfaction of a user of the service". This definition of QoS is a wide one, encompassing many areas. The QoS parameters, as user satisfaction, are subjective in nature, depending on individual perception and expectations.
As derived from [5], the user-oriented QoS parameters provide a valuable framework for design, but they are not necessarily usable in specifying performance requirements. Similarly, the performance parameters ultimately determine the user-observed QoS but they do not necessarily describe that quality in a way that is meaningful to users.

9.6.2.User-perceived QoS

The following parameters are defined concerning the QoS of the demonstration, from the user's viewpoint:

- **usability of the security features**
- **acceptability of the security features**
- **user-perceived stability of the demonstration**
- **user-perceived performance of the demonstrated protocols**
- **user friendliness of the GUI / GUI operability**
- **sufficiency and clarity of displayed information**
- **overall user satisfaction of the demonstration**

9.6.3.Achieved goals

9.6.3.1.Usability of the security features

The security functions are executed automatically. The users are not directly involved in the protocols execution. They are mainly informed about them by the Monitor and Tracer tools.

9.6.3.2.Acceptability of the security features

Since the existence of the security mechanisms is a very positive feature for the users, the degree of acceptability naturally depends only on the impact of the security functions on the transaction speed.

9.6.3.3.User-perceived stability of the demonstration

The demonstration is quite flexible regarding intentional or unintentional misuse by the users. There are messages that appear when the users make inapplicable selections, in order to help them define the proper sequence of actions.

9.6.3.4.User-perceived performance of the demonstrated protocols

There is no user-perceived impact on the transaction speed or in the general communication performance induced by the security layer.

9.6.3.5.User friendliness of the GUI / GUI operability

The GUI operability can be defined as its ability to be successfully and easily operated by a user. This goal is achieved since the protocols are executed automatically, upon selection of the authentication mechanism. Meanwhile, the tracer window provides information about the security-related message exchange. The user has to create the entities however, prior to any other action. The GUI informs the user of this task if an attempt is made to initiate a protocol without any created entities. Also, the menu bar is designed in a way that all the actions can be selected sequentially.

9.6.3.6.Sufficiency and clarity of displayed information

The user has the options to configure the communication database and the entities involved and watch the message flow in an abstract (Monitor option) or concrete way (Tracer option). The Tracer window enables the user to observe and understand the internal actions that are not presented in the main GUI window (key

generation, certificates, etc.). Also, detailed information on the involved entities' features is included in the Status windows for the user's convenience.

9.6.3.7.Overall user satisfaction of the demonstration

The user impression is that the demonstration succeeded in achieving its goal: to present the proposed security functions and prove their efficiency in a friendly and perceivable way.

9.7.Appearance of Demonstration

9.7.1.First demonstration's Graphical User Interface

The main objective of the first WP2.1 demonstration is to present alternative procedures for the authentication between user and network, in a UMTS environment. Four variations of the authentication procedure are demonstrated, corresponding to two different authentication mechanisms and their versions according to whether the user is known to the network or not. It comprises the three entities necessary for successful authentication: User, Network Operator (NO) and Service Provider (SP).
The GUI offers the possibility to the user to run one application, which simulates all the involved entities in a single PC, or to assign the entities over a number of applications, which may run on one or more PCs. In the latter case, the communication between the applications is provided via an Ethernet LAN, using the TCP/IP protocol.
The authentication is initiated by the Network Operator, when the registration is current and by the User, when the registration is new. In the former case, the respective protocols involve the exchange of seven messages between User and Network. The first four messages establish an agreement on the authentication mechanism to be used, based on the User's authentication capability class, while the last three correspond to the execution of the selected mechanism. In the latter case, the respective protocols involve the exchange of eight or ten messages among all three participants, depending on the selected mechanism. The Network queries the Service Provider on the User's authentication capabilities, upon the User's authentication request and requires additional authentication data during the symmetric key mechanism.
This is the minimum amount of information that the demonstration user needs to know, in order to understand the interactions that will be enabled.

9.7.2.The observer's view

The role of the GUI is to provide the user with the necessary means to select, initiate and observe the authentication process. In this first demonstration, the user is able to configure and view a series of important parameters about the communication database, such as:

- type mode (server or client)
- number of clients
- server IP address
- client identity

and the involved entities, namely:

- state
- identity
- authentication capabilities
- all involved keys
- certification authority
- UIM option for the User

An overall graphical representation of the message flow is provided via the Monitor option. The Monitor bitmaps, representing the involved entities and the exchanged messages, immediately provide an impression of the interactions to any viewer of the demonstration.
The Tracer window, on the other hand, has the complementary role of recording the message contents and exchanges. It provides the information needed for observing the internal process, including key generation, certificates, entities' state, etc., while the protocol is running. Thus, while the information contained in the Tracer window is intended for users that are familiar with the protocol and the scope of the demonstration, these data also enable all users to verify the successful execution of the authentication procedure. In addition, this tool provides the possibility to estimate the performance of the executed protocols and the potential delays introduced by the security functions.
The demonstration is driven by the user. First, the GUI user has to configure the communication subsystem and next, "create" the three entities involved. Then, through the entities' Action menu items, the user can select and initiate the authentication mechanisms available.
In addition, the demonstration user may always:

- destroy entities
- change the entities' profile (e.g. authentication capability class)
- view the entities status

Error messages are displayed when an improper selection is made. There are also some indicative messages that appear in order to help the user follow the proper sequence of actions. Finally, there are informative messages that confirm the successful completion of some actions. Thus, the GUI provides information on the impact of the demonstration user's actions.

9.7.3.Suggestions for enhancement

Some improvements can be made concerning the displayed messages in Monitor window, since now they are only indicative. This improvement would allow the message flow to be more intelligible for a project-unrelated user.
The Tracer window can also be enhanced to support options, for example to save or print the presented messages, so that they can be viewed off-line.
There could also be the possibility to observe the process step-by-step, in order to have an elaborate perception of the protocols, instead of just providing an overall view of the protocols execution.
Finally, the on-line Help option that will be available in the future is intended to include more extensive information about the GUI features.

10.Summary of enhancements

10.1.Enhancements to the Network side

The limitations of a more realistic platform can be simulated in the demonstration:

- the need for confirmed messages will remove an acknowledgement message,
- separation of security for signalling and data transfer,
- the cryptographic engines, the AC, can run on a more performant platform.

Performance has to be enhanced, especially at the users side (the smartcard).

Commands to update the authentication capability class can be foreseen.

In addition the GUI can be optimised:

- the monitoring mode,
- add more option to the tracer window,
- step-by-step execution,
- more on-line help.

10.2.Enhancements to the UIM

The most important enhancement for the UIM is the integration of biometric user authentication as a replacement for PIN based access control mechanisms. The amount of memory necessary to support this feature severly restricts the level of GSM services available together with biometric features. It shall be investigated how to balance the different requirements on the UIM:

- UMTS authentication protocols
- GSM voice call and Phase 2 services
- Biometric user authentication
- Storage of EXODUS data for the trials

As a major result of an enhancement process, memory trade-offs on the smart card shall be identified.