NTNU
Innovation and Creativity

Mobile Supplicant for SIM Authentication

Håvard Holje

Master of Science in Communication Technology
Submission date: June 2007
Supervisor: Van Thanh Do, ITEM
Co-supervisor: Ivar Jørstad, Ubisafe

Norwegian University of Science and Technology
Department of Telematics

Problem Description

Until now, GSM SIM authentication has mostly been used in the authentication process of native mobile telecommunication services like voice telephony and SMS. Some efforts have been put into using SIM for authenticating access to Wireless Local Area Networks (WLAN). However, it should also be possible to reuse the same authentication mechanism for many other types of distributed services accessed through the mobile phone. The goals of this master thesis are to analyse, specify and implement a system which allows SIM authentication towards services that are accessed through a Web-browser or a standalone application (e.g. a J2ME MIDlet) on a mobile phone

Assignment given: 17. January 2007
Supervisor: Van Thanh Do, ITEM

Preface

This thesis is the final work of my Master's degree carried out at the Norwegian University of Science and Technology (NTNU), department of Telematics, from January to June 2007. The writing of a research thesis is a requirement during the final year for a student to be awarded the Master of Science degree in Telematics by NTNU.

I would like to thank my supervisor Dr. Ivar Jørstad, CEO Ubisafe AS, for all of his valuable comments and support during the project and for presenting our paper at the ERCIM workshop on eMobility in Coimbra, Portugal. I would also thank Professor Do van Thanh, Telenor R&I and NTNU for clarifying meetings at Telenor Fornebu.

Håvard Holje
Trondheim, Norway
June 13, 2007

Abstract

This Master's thesis proposes a solution for utilizing the GSM SIM to authenticate users to distributed services accessed through the mobile terminal. By combining the GSM SIM authentication mechanisms with the EAP-SIM framework we achieve mutual authentication between the parties. By combining the fact that the GSM SIM is a tamper resistant Smart Card, and that users have to present a valid PIN to activate the system, strong two-factor authentication is achieved fulfilling the highest security level defined by NIST [2].

The proposed system is secure, easy to use and inexpensive, because most of the components needed already exist in the GSM network today. Existing strong user authentication systems for mobile handsets require several devices to be able to offer secure services. The proposed system only requires one device, namely the mobile handset which the user is carrying anyway. The only user interaction required is typing the PIN.

The authors' major contribution to the proposed system is the Supplicant, residing on the mobile handset communicating with the SIM through the SATSA-APDU interface. By running the Supplicant as a local proxy on the mobile handset, it is able to communicate with all kinds of client applications supporting HTTP, e.g. mobile browsers, J2ME MIDlets and native applications.

A prototype implementing several of the components in the proposed system has been developed. Unfortunately, due to several reasons, the prototype cannot be deployed on a real mobile handset today's date. We are missing the necessarily certificate required to get access to the SIM and neither of today's mobile handsets support all the functionality needed. However, the prototype has been implemented successfully on a PC running the Wireless Toolkit from Sun, which simulates the SIM environment.

Based on results from this thesis, the author has written the paper "A Unified Authentication Solution for Mobile Services". The paper was accepted and published on the ERCIM workshop on eMobility in Coimbra, Portugal, on May 2007.

Contents

PREFACE ..... **I**

ABSTRACT ..... **III**

CONTENTS ..... **IV**

LIST OF FIGURES ..... **VIII**

LIST OF TABLES ..... **IX**

ABBREVIATIONS ..... **X**

DEFINITIONS ..... **XII**

1 INTRODUCTION **1**

1.1 MOTIVATION 1

1.2 PROBLEM DEFINITION 2

1.3 CHALLENGES 2

1.4 RELATED WORK 3

1.4.1 Offering SIM Strong Authentication to Internet Services *3*

1.4.2 Previous Master's thesis related to SIM authentication *3*

1.4.3 SIM authentication in WLAN *3*

1.4.4 Other similar initiatives *4*

1.5 METHODOLOGY 5

1.6 STRUCTURE OF THIS REPORT 6

2 BACKGROUND **7**

2.1 AUTHENTICATION TERMINOLOGY 7

2.1.1 Terms and concepts *7*

2.1.2 Tokens *7*

2.1.3 Authentication modes *9*

2.1.4 Authentication levels *10*

2.1.5 Authentication model *12*

2.1.6 Identity Management (IdM) *13*

2.2 GSM INTRODUCTION 14

2.2.1 GSM network architecture *15*

2.3 GSM SIM AUTHENTICATION 19

2.3.1 GSM security components *19*

2.3.2 Authentication schemes *23*

2.3.3 Security considerations *26*
2.4 EAP-SIM 28
2.4.1 Introduction *28*
2.4.2 Authentication procedure *28*
2.4.3 Security considerations *31*
2.5 JAVA 2 PLATFORM MICRO EDITION (J2ME) 33
2.5.1 Configurations *34*
2.5.2 Profiles *34*
2.5.3 Security mechanisms *34*
2.5.4 Optional packages *35*
2.6 AUTHENTICATION OF MOBILE SERVICES 37
2.6.1 Authentication for WAP-based Services *37*
2.6.2 Authentication for Web-based Services *37*
2.6.3 Authentication for Java-based Services *37*
3 ANALYSIS 39
3.1 STAKEHOLDERS AND END-USERS 40
3.1.1 Stakeholders *40*
3.1.2 End-users *40*
3.2 HIGH-LEVEL USE CASE MODEL 41
3.2.1 Use case 1 (UC1) – Use service *41*
3.2.2 Use case 2 (UC2) – Authenticate user *41*
3.2.3 Use case 3 (UC3) – Authorize user *42*
3.2.4 Actors and system parts *43*
3.3 USE CASE SPECIFICATION 45
Use Case UC1: Use service *46*
Use Case U1.1: Verify PIN *47*
Use Case UC2: Authenticate user *48*
Use Case UC2.1: Get user identity *50*
Use case UC2.2: Select EAP-SIM version *51*
Use Case UC2.3: Get GSM Triplets *52*
Use Case UC2.4: Challenge supplicant *53*
Use Case UC2.5: Compare the results *54*
Use Case UC2.6: Create SA *55*
Use Case UC3: Authorize user *56*
3.4 SUPPLEMENTARY SPECIFICATION 57
3.4.1 Functionality *57*
3.4.2 Usability *57*

3.4.3 Reliability *58*
3.4.4 Performance *58*
3.4.5 Supportability *58*
3.4.6 Existing components and interfaces *58*
3.4.7 Extensibility *58*
3.5 DOMAIN MODEL 59
4 DESIGN 60
4.1 COMPONENTS 61
4.1.1 Mobile browser *61*
4.1.2 Stand-alone application *63*
4.1.3 Supplicant *65*
4.1.4 SIM *66*
4.1.5 Service Provider (SP) *67*
4.1.6 Identity Provider (IdP) *68*
4.1.7 Authentication Server *69*
4.2 INTERFACES 70
4.2.1 Mobile handset – Service Provider interface *70*
4.2.2 Supplicant – SIM interface *72*
4.2.3 Supplicant – Identity Provider interface *72*
4.3 CLASS DIAGRAMS 73
4.4 SEQUENCE DIAGRAMS 76
5 REALIZATION OF THE PROPOSED SYSTEM 79
5.1 SIM CARD SIMULATOR 80
5.2 SUPPLICANT 81
5.2.1 SIM communication *82*
5.3 CLIENT APPLICATION 83
5.3.1 Browser and other stand-alone applications *84*
5.4 SERVICE PROVIDER (SP) 85
5.5 IDENTITY PROVIDER (IDP) 85
5.6 INTERACTION DIAGRAM 86
6 DISCUSSION 88
6.1 TECHNICAL BARRIERS 88
6.1.1 SIM communication *88*
6.1.2 Generic system *91*
6.2 SECURITY CONSIDERATIONS 93
6.2.1 Security tokens *93*

6.2.2 Securing the communication channels *94*
6.2.3 Authorization *96*
6.2.4 Client side security *97*
6.3 DESIGN CHOICES 98
7 CONCLUSION 100
7.1 ACHIEVEMENTS AND RESULTS 100
7.2 CRITICAL REVIEW 101
7.3 FUTURE WORK 102
APPENDIX A – STEPS FOR RUNNING THE SIM SIMULATOR 103
APPENDIX B – THE SIM CARD APPLET (SIM.JAVA) 105
APPENDIX C – UML CLASS DIAGRAMS 110
APPENDIX D – LIST OF CURRENT MOBILE HANDSETS 112
APPENDIX E – SUBMITTED PAPER TO ERCIM 2007 113
APPENDIX F – ENCLOSED ZIP FILE 129
REFERENCES 130

List of Figures

Figure 1 - Unified Process phases and disciplines 5
Figure 2 - Authentication model 12
Figure 3 - General architecture of a GSM network 15
Figure 4 - Mobile Equipment (ME) and SIM 16
Figure 5 - GSM security components 19
Figure 6 - Physical dimensions of ID-1 SIM and Plug-in SIM 20
Figure 7 - GSM SIM initial user authentication 23
Figure 8 - The GSM SIM authentication scheme 24
Figure 9 - Session key generation with the A8 algorithm 25
Figure 10 - COMP128 algorithm generating SRES and Kc at once 26
Figure 11 - EAP SIM full authentication procedure 29
Figure 12 - Overview of Java 2 Platform, Micro Edition 33
Figure 13 - High-level user case diagram 41
Figure 14 - Use case specification overview 45
Figure 15 – UML Domain model 59
Figure 16 - Overall architecture of the proposed SIM authentication system 60
Figure 17 – Details of the Supplicant component 65
Figure 18 – Details of the SIM component 66
Figure 19 – Details of the SIM component (Alternative solution) 66
Figure 20 – Details of the Service Provider component 67
Figure 21 – Details of the Identity Provider component 68
Figure 22 – Details of the Authentication Server component 69
Figure 23 - Service request from browser 70
Figure 24 - XML Schema defining the Supplicant interface 71
Figure 25 – Package diagram of the proposed SIM authentication system 73
Figure 26 – ServiceSupplicant – class diagram 74
Figure 27 – SIMSupplicant – class diagram 74
Figure 28 – SIMSupplicant – class diagram 75
Figure 29 – Sequence diagram – access service 76
Figure 30 – Sequence diagram – verify PIN 77
Figure 31 – Sequence diagram – authenticate user 78
Figure 32 – UML Deployment diagram 79
Figure 33 – Supplicant screenshot 81
Figure 34 – Client application – Start screen 83

Figure 35 – Client PIN authentication 83
Figure 36 – The mobile banking GUI 84
Figure 37 – Get balance results 84
Figure 38 – Interaction diagram of the realized authentication system. 86

List of Tables

Table 1 - Stakeholders of the project 40
Table 2 - End-users of the authentication system 40
Table 3 - Actors and system parts 44

Abbreviations

2G	The second generation of GSM
3G	The third generation of GSM (See UMTS)
3GPP	The third Generation Partnership Project
A3	Algorithm 3 (Authentication algorithm in GSM)
A5	Algorithm 5 (Encryption algorithm in GSM)
A8	Algorithm 8 (Cipher key generator in GSM)
AAA	Authentication, Authorization and Accounting
AID	Application Identifier
AKA	Authentication and Key Agreement
APDU	Application Protocol Data Unit
AuC	Authentication Center
BSC	Base Station Controller
BSS	Base Station Subsystem
BTS	Base Transceiver Station
CDC	Connected Device Configuration
CHV	Card Holder Verification information (See PIN)
COMP128	Algorithm combining A3/A8
CLDC	Connected Limited Device Configuration
DoS	Denial Of Service (Attack)
EAP	Extensible Authentication Protocol
EIR	Equipment Identity Register
GSM	Global System for Mobile Communications
GSM AKA	GSM Authentication and Key Agreement
HLR	Home Location Register
HTTP	Hyper Text Transfer Protocol
HTTPS	Secured HTTP
ICC	Integrated Circuit Card
IDP	Identity Provider
IP	Internet Protocol
IMEI	International Mobile Equipment Identifier
IMSI	International Mobile Subscriber Identity
ISDN	Integrated Services Digital Network
J2ME	Java 2 Micro Edition
JSR177	See SATSA

Kc	Cipher Key (Generated by A8)
Ki	Private Key (In GSM)
LAI	Location Area Identity
ME	Mobile Equipment
MIDP	Mobile Information Device Profile
MS	Mobile Station
MSC	Mobile Switching Centre
NIST	National Institute of Standards and Technology
NS	Network Subsystem
OTP	One Time Password
PC	Personal Computer
PKI	Public Key Infrastructure
PIN	Personal Identification Number
PSTN	Public Switched Telephone Network
PUK	Personal Unblocking Key
RADIUS	Remote Authentication Dial In User Service
SA	Security Association
SAML	Security Assertion Markup Language
SATSA	Security and Trust Services API (for J2ME)
SIM	Subscriber Identity Module
SP	Service Provider
SRES	Signed RESponse
SSO	Single Sign-On
TMSI	Temporary Mobile Subscriber Identity
UP	Unified Process
UMTS	Universal Mobile Telecommunications System
USIM	Universal Subscriber Identity Module
VoIP	Voice Over IP
VLR	Visitor Location Register
VPN	Virtual Private Network
WAP	Wireless Application Protocol
WIM	WAP Identity Module
WLAN	Wireless Local Area Network
WTLS	Wireless Transport Layer Security

Definitions

3GPP	A GSM based consortium advocating standardization for mobile communications
802.1x	An authentication standard for wired and wireless LANs
Authenticator	The end of the link initiating authentication
Claimant	The party to be authenticated
EAP server	A backend authentication server like RADIUS supporting EAP
Frequency hopping	Rapidly switching a carrier among many frequency channels using a sequence known to both transmitter and receiver
Handover	The passing of a call signal from one base station to the next as the user moves out of range of a cell
Hard Token	A hardware device that contains a protected cryptographic key
Identity Provider	Handles user credentials, federating identities, SSO etc. often in connection with an Identity Management system.
MIDlet	A Java program for the J2ME virtual machine
Mutual authentication	Entity authentication which provides both entities with assurance of each other's identity
NGN	A generic term used to describe the emerging Next-Generation packet-based networks
Nonce	A randomly chosen value, inserted in a message to protect against replays.
Principal	An entity whose identity can be authenticated
Protection domain	In J2ME a protection domain determines access to protected functions. There are four kinds of protection domains: minimum, untrusted, trusted and maximum
Roaming	The ability to use your cellular phone outside your local calling area
Service Provider	(Application) Service Provider is an entity providing computer-based services to customers over a network
Single Sign-on	Users sign onto a site only once and are given access to one or more applications in a single domain or across multiple domains
Smart Card	A credit-card sized tamper resistant plastic card that contains a microprocessor that can store and process data
Soft token	A cryptographic key that is stored on disk or some other media

Supplicant	In this context a supplicant is the software process on the client platform that performs the authentication negotiation
Verifier	The party verifying the identity of the claimant
Unilateral authentication	Entity authentication which provides one entity with assurance of the other's identity but not vice versa

1 Introduction

1.1 Motivation

From a simple device terminating the mobile network, the mobile phone has evolved to become a quite advanced device capable of hosting applications that are until now run only on stationary computers. The limitations in terms of processing, storage and battery life are considerably reduced, and the mobile phone will soon become a mobile computer. However, there is one major obstacle, which is the current closed architecture of the mobile terminal. Indeed, the architecture is very much telephony centric, i.e. it is built to support the traditional telecommunication services like GSM voice, SMS, WAP, etc. Other applications like browsing, Web services, P2P applications get very little support and in most cases have to manage by themselves.

Existing (strong) authentication schemes on mobile handsets suffer of serious drawbacks. Some are completely separated from the SIM and require additional elements such as a Smart Card, a one-time password generator, etc. The others access the SIM authentication functions indirectly via SMS.

Telenor has worked with the idea of using the GSM SIM as an authentication token for new applications for many years. The last contribution was the Master thesis "Using SIM for strong end-to-end application authentication" written by the former NTNU MSc students Lars Lunde and Audun Wangensteen, June 2006. They designed and implemented a solution for a SIM-based authentication system using a regular PC and a SIM-card reader so that they could communicate with the SIM-card.

A hot topic today is Internet banking on mobile phones. Such services have high safety requirements and they need strong user authentication. Using the GSM SIM as an authentication token for such purposes would be very convenient and cost-effective:

- **Secure**
  The GSM SIM is a tamper resistant device that contains strong authentication mechanisms. Hence we don't need an extra device for providing two-factor, mutual user authentication.

- **Easy to use**
  The targeted group is familiar with using the mobile phone and its features
- **Inexpensive**
  We can reuse most of the existing GSM-network components

1.2 Problem definition

Using the SIM for authentication of other services than GSM specific ones is not trivial. Detailed knowledge of a lot of technologies and components is required, all the way from the SIM itself, and towards the Authentication Centre in the GSM network. The project assignment carried out in the fall 2006 was focusing on technology research, relevant background material and analysis of the proposed authentication system. In this Master's thesis, the proposed system will be specified further and the different components will be implemented to discover eventually disagreements in the proposed system.

The major problem statement in this thesis has been:

1. Is it possible to use the SIM as a general-purpose authentication token in non-GSM services accessed directly through a mobile handset with an integrated SIM?

The following sub-statements were defined to guide the work with this project:

a. Is it possible to communicate with the GSM SIM through SATSA?
b. Is it possible to realize a local Supplicant on the mobile handset, which is able to communicate with both a WWW browser and a stand-alone application?

1.3 Challenges

The real challenge here is to understand the underlying technology and how everything can be connected together. The technology is complex and it is not trivial to combine the GSM SIM with other services than GSM specific ones.

The proposed system must adapt to other current applications and there are a lot of standards to comply with and be prepared to.

1.4 Related work

1.4.1 Offering SIM Strong Authentication to Internet Services

Telenor is involved in the SIM strong project [6] which aims to extend the use of GSM SIM authentication to internet Web Services. Telenor, Axalto, Linus and Oslo University College have implemented a proof-of-concept prototype together in Oslo. The prototype demonstrates the possibility of implementing innovative service in a heterogeneous environment using Liberty Alliance Federation Standard. [7].
The prototype is based on internet Web services on a regular PC and it supports both communicating with the SIM on a mobile phone via Bluetooth and using a SIM card-equipped dongle, card reader or 2G/3G card. This work is closely related to this assignment and both my supervisor and professor are involved.

1.4.2 Previous Master's thesis related to SIM authentication

"Using SIM for strong end-to-end application authentication" written by Lars Lunde and Audun Wangensteen spring 2006. They designed and implemented a prototype of a generic authentication system (GAS) based on GSM SIM. The GAS included a client supplicant residing on a PC and a server (authenticator) part, both developed in Java. The client supplicant communicates with the SIM via the Bluetooth SAP interface.

1.4.3 SIM authentication in WLAN

Gemalto**,** former Axalto and Gemplus [8]**,** provides a SIM-based WLAN authentication solution for mutual network-based authentication. Their solution keeps the subscribers and infrastructure protected for the provision of new high-value Internet-based services. They provide a SIM card and a USB dongle, which connects the SIM to a PC.

Their solution also supports VPN. They provide a package including software, a SIM card and a smart card reader. The system is designed to authenticate the WLAN user to the network operator and to provide secure access to the user's corporate network and data. Once installed on the PC, it manages the operator/end-user authentication over the WLAN, and then automatically runs the enterprise VPN authentication. It guarantees the most secure connection over public hotspots by linking into the GSM infrastructure.

There are several other companies offering similar solutions, but Gemalto is mentioned here because they have been working with such solutions for many years and they are the

leading company regarding SIM authentication for WLAN.

Recently some mobile phones have got a build-in EAP-SIM supplicant for use with WLAN. Nokia 9500 provides such a solution, which can be combined with the 802.1x framework to achieve strong SIM based authentication in WLAN's. Because this is quite emerging technology, there is limited documentation available regarding the usage of such a build-in EAP-SIM supplicant. But we can imagine that solutions similar to what Gemalto provides would be even easier to carry out and the users might take advantage of the WLAN capabilities on the handheld device, without using a PC at all.

1.4.4 Other similar initiatives

As far as the author knows there exists no identical solution to the one proposed in this assignment. The usage of GSM SIM for authentication is a hot topic today, but the existing solutions are either product specific or they are based on a regular PC and not a mobile phone.

If we for one moment move away from the SIM as a basis for the authentication, there are several other solutions providing strong user authentication for services accessed through a mobile handset. The major banks in Norway offer today internet banking through the mobile phone. They use bankID [9] and other PKI solutions requiring a second device, in addition to the mobile phone, in the authentication procedure. (I.e. an electronic code calculator or a code card provided by the bank). There are also other companies offering similar solutions like the Norwegian technology company enCap [10], which offers secure user authentication for any online services, by using the mobile phone as a trusted device, in connection with an electronic code calculator.

As we can see there exist several solutions concerning user authentication of services accessed both through a regular PC and through a mobile phone. The solution closest to the proposal in this assignment is probably the integrated EAP-SIM supplicant provided on the Nokia 9500 communicator.

1.5 Methodology

The methodology used is in accordance to the Unified Process (UP) and the diagrams and use case models are based on [35], "Applying UML and patterns".

An important part of the UP methodology is the early mitigation of high risk issues. This is achieved by frequent iterations. This is why the author has chosen UP as methodology for this thesis, since the development is extremely dependent of emerging and state-of-the art technology. When new requirements emerge and the assumptions are changing, an agile methodology like UP is a necessity for efficient development. The inception phase is the initial phase, but it does a lot more than just defining a vision and some high level requirements. Developers have to start looking at all the aspects of the intended system immediately. UP supports this mentality by the means of the following phases: Inception, elaboration, construction and transition. For each of these phases there are several disciplines like business (domain) modeling, requirements, design, implementation and testing as shown in figure 1 below.

This Master's thesis is based on the authors project assignment carried out in the autumn of 2006, which covered most of the inception phase of the proposed authentication system. In this Master's thesis the elaboration and construction phase has had most of the focus, but the inception phase has been revised as well, since the assumptions have changed during the scientific research.

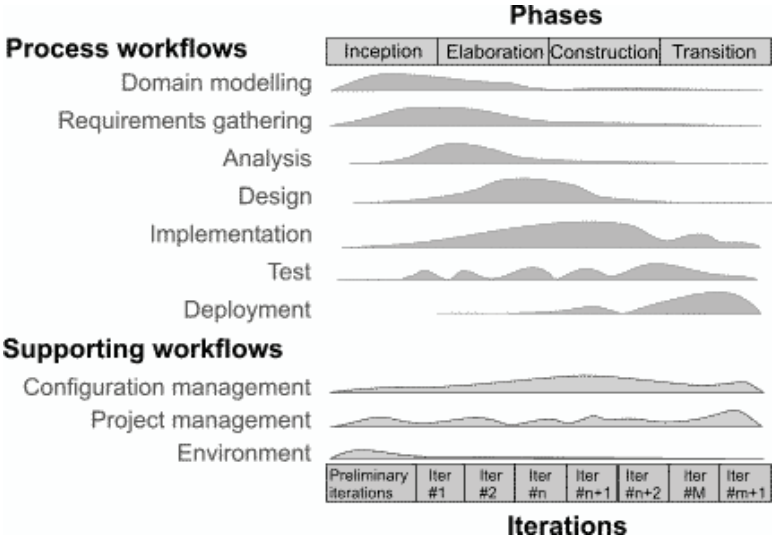


Figure 1 - Unified Process phases and disciplines

1.6 Structure of this report

Chapter 1 – Introduction
The introduction contains motivation, problem definition, challenges, related work and methodology, which form the basis of this Master thesis.

Chapter 2 – Background
This chapter contains the background material required to understand the concept of the proposed SIM authentication system.

Chapter 3 – Analysis
This chapter introduces the proposed authentication system by identifying the requirements and the concept, by means of use cases and a domain model.

Chapter 4 – Design
The design phase elaborates the concept further by detailing the components and interfaces between them. Interaction diagrams and class diagrams are also provided in this chapter.

Chapter 5 – Realization
This chapter describes the work that has been done to realize the proposed authentication system.

Chapter 5 – Discussion
This chapter provides an evaluation of the proposed authentication system and the prototype. Strengths, weaknesses and security are highlighted.

Chapter 6 – Conclusion
The conclusion gives a summary of the achievements of this Master's thesis. The results are discussed according to the problem definition, the author evaluates his work and future work is proposed.

2 Background

2.1 Authentication terminology

Before we start exploring the GSM SIM authentication schemes, we will discuss some terms and concepts regarding authentication. The theory behind is quite comprehensive and authentication often depends on complex cryptographic protocols and algorithms. Thus we need a clear understanding of the concept, as a foundation for further in-dept studies of GSM SIM and EAP authentication.

2.1.1 Terms and concepts

Authentication is the process of establishing confidence in the truth of some claim [11]. In the content of information security, authentication is comprised by two different aspects:

1) *Verification of identity* - The process of determining whether someone or something is, in fact, who or what it claims to be.
2) *Data integrity* - Ensure that no one tampers with the data.

2.1.2 Tokens

A token in this context is something used to identify the claimant's identity. Since the claimant usually authenticates to a system or application over a network, the token used must be protected and held secret. A very important aspect of authentication systems are the number of factors they are using. There are a lot of different token types available and everyone can be categorized in one of these factors:

Something you:

- know (password, PIN)
- have (ID card, smart card)
- are (fingerprint, DNA, retina pattern)

An authentication system that adopts all three factors is considered much safer than a system only adopting one or two. (See section 2.1.4 – authentication levels).

In addition there exist four kinds of claimant tokens. Each type incorporates one or more of the authentication factors mentioned above.

2.1.2.1 Password token

The most used authentication token today is the password. The claimant memorizes a secret password and uses this to authenticate his or her identity. There are several problems with the password token [6]:

- Users must remember a lot of passwords. The human brain is normally capable of memorizing about 5 different passwords combined with a username. This leads to reuse of existing tokens or maybe worse, some people will write them down on a yellow post-it note and place it beside the screen or below the keyboard.
- Might be susceptible to dictionary/guessing-attacks
- Phishing, easy to steal passwords from users. Either by asking them directly or by simulating a well known login site and make sure that the user will enter the login info there.
- With insufficient length, it is possible to use brute force attacks to uncover the passwords

Hence, stand-alone passwords as a means of authentication is not strong enough for services like e-commerce, online banking and corporate intranet.

2.1.2.2 Hard token

A hard token is a hardware device that contains a protected cryptographic key. The claimant must prove possession of the token and the token must require a password/biometric to activate the authentication key. It cannot be able to export authentication keys and it must be FIPS 140-2 validated [12] according to [2].

2.1.2.3 Soft token

This is a cryptographic key that is stored on disk or some other media. Like the hard token, authentication is accomplished by proving possession and control of the key and the key must be protected by a password/biometric only known to the claimant.

2.1.2.4 One-time password device token

This is usually a personal hardware device that generates one-time passwords for use in authentication, i.e. internet banking. The passwords shall be generated by using an

approved block cipher or hash algorithm to combine a symmetric key stored on a personal hardware device with a nonce. The one-time password must have a limited lifetime, on the order of minutes.

2.1.3 Authentication modes

There are mainly three different authentication modes according to [11]:

Individual authentication:
Verify information that is strongly linked to or that uniquely identifies an individual. (Not necessary a human being, it could be a computer, or a telephone). This might be an identifier like a person's name, e-mail or a more distinct attribute like DNA, biometrics and so on. This usually happens in to phases:

1. Identification phase where the identifier is selected
2. Authentication phase, where the required level of confidence is established based on one or more authentication challenges which are tightly attached to the individual. This kind of authentication is also referred to as "user authentication" or just "verification".

Identity authentication
This is quite similar to individual authentication, but it may not be possible to link the authenticated identity to a specific individual. Email is an example of this type of authentication. For instance, an email account requires a password, and it may be used for authentication in a certain level. But many people may have access to this account, thus we cannot tie this to a specific individual.

Attribute authentication,
This authentication type is not as strict as individual/identity authentication, but it contains two phases as well:

1. Attribute selection phase (e.g. height, weight, sex)
2. Authentication phase, where the attribute is verified against the target. This authentication is common in amusement parks and in other situations where identity doesn't matter.

2.1.4 Authentication levels

The term authentication is somewhat become “old fashion”. The new buzzword nowadays is “electronic authentication” or “e-authentication”, which is the process of establishing confidence in user identities electronically presented to an information system. [2].

The level of authentication required is heavily dependent on the situation. If accountability is involved, individual authentication is necessary and a high level of assurance is required. In other situations a lower level of assurance is sufficient and identity authentication may be used.

National Institute of Standards and Technology (NIST) has defined four authentication levels, as a technical guidance to implement electronic authentication.

Level 1

This is the lowest assurance level and the claimant is not required to proof his or her identity. It allows a wide range of different authentication mechanisms to be employed and any of the token methods from level 2, 3 and 4. But it requires that the claimant prove through a secure authentication that he or she controls the token.

Level 2

This level also supports a wide range of authentication mechanisms included all the token methods from level 3 and 4 as well as passwords and pin codes. It provides single factor remote network authentication and in addition to prove that the claimant controls the token, the claimant has to proof his or her identity as well. Eavesdropper, replay- and online guessing attacks are prevented at this level.

Level 3

Provides multi-factor remote network authentication and it requires at least two authentication tokens and a cryptographic strength mechanism to protect the primary token (a secret key, private key or a one-time password). Three kinds of tokens are allowed:

- Soft cryptographic tokens
- Hard cryptographic tokens
- One-time-passwords

Level three protects against verifier impersonation and man-in-the-middle attacks, in addition to all level 2 protections.

Level 4

This is the highest practical level of remote network authentication. It is quite similar to level 3, but only hard tokens are allowed. This means that the claimant needs a physical hardware cryptographic module, e.g. a smart card, which is tamper resistant according to FIPS 140-2 Level 3 [12].

2.1.5 Authentication model

The general model for entity authentication mechanisms is shown in figure 2. In a client/server context, which is a common way of gaining access to different resources, the users have to authenticate themselves against the server. This is called *one-way authentication* or *unilateral authentication.* The same applies if A is the claimant and B is the verifier, and A wants to communicate with B. The problem is that A in this case doesn't know whether B actually is who it claims to be. There are many examples of "traffic hijacking". For example "Man-in-the-middle-attack", fake GSM base stations and so on. Adopting *mutual authentication* is a way to avoid this. [13]

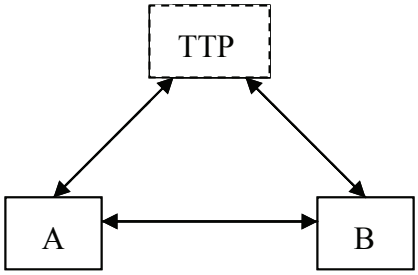


Figure 2 - Authentication model

If the user can logon to a device he or she has physically access to, it is called *implicit two-way authentication*. This is sufficient for most individuals. But sometimes even a visual inspection isn't sufficient. It is quite easy to change what's inside a computer. However, if the user is authenticating to a server accessed via a remote connection, it may be located far away from the user and we have to adopt *explicit two-ways authentication,* better known as *mutual authentication.* That is, both parts must either share a secret common cryptographic key or signature/verification key pairs. When implementing such mutual authentication mechanisms, we ensure origin authentication and data integrity. Authentication based on cryptography is secure as long as the originator's key has not been compromised. [14].

2.1.6 Identity Management (IdM)

There exist a lot of different authentication systems nowadays, and most of them have moved to the Web to automate business processes. The introduction of IdM systems is crucial for the future of authentication systems. Human beings are not capable of remembering unlimited different passwords and the amount of systems requiring user authentication is increasing. An IdM system involves the creation, access, update and storage of private user information, along with security services to protect its confidentiality [48].

The Liberty Alliance project [7] developed an open specification for a secure single sign-on (SSO) system. The specification is based on SAML [37], which is an XML-based protocol for exchanging authentication and authorization information on the Internet. The liberty specification describes a federated network identity, where a group of companies agree to work together as trusted parties. This implies the user only have to log on to one system to get access to all of these companies' Internet services. An Identity Provider (IdP) will identify and authenticate the user, and manage the identities among the different systems.

2.2 GSM introduction

GSM (Global System for Mobile Communication) is the second generation of wireless communication systems, supporting both voice and data communications. It was developed in the mid 80's by the GSM consortium and it has grown rapidly since then. In June 2006 there were 2 billion registered subscribers according to [15].

GSM was developed with security in focus. One of the security goals was to make the system as secure as the PSTN and also avoid cloning of MS's. From the operators point of view the most crucial part is to bill the correct customers, avoid fraud and protect the services from unauthorized use. The user's main concerns are privacy and anonymity. This is achieved through strong user authentication, encryption and the use of temporary identifiers.

The use of the air-interface as the transmission media causes a number of potential threats as well, i.e. eavesdropping and monitoring. This is taken care of by introducing confidentiality and anonymity on the radio path.

Security in GSM is divided in three main areas [16].

- Subscriber identity authentication
- User and signalling data confidentiality
- Subscriber identity confidentiality

We will focus on the subscriber identity authentication service. This is the core of the GSM security system allowing seamless handover and roaming. The authentication service enables the fixed network to authenticate the identity of mobile subscribers by a simple challenge-response protocol. The authentication service also establishes and manages the encryption keys needed to provide the confidentiality services.

The result of this is that no sensitive data is transmitted over the radio channel. The unique subscriber identity (IMSI) and the secret, individual authentication key (Ki) are only used in the initial authentication, in connection with the challenge-response mechanism. The actual conversation is encrypted using a random, temporary key (Kc). The IMSI is substituted with a temporary mobile subscriber identifier (TMSI) issued by the network. The TMSI may be changed once in a while, typically during handovers, for

additional security.

Every GSM network and all mobile equipment must support the GSM authentication scheme. But the operators have a free hand to implement their own algorithms within the GSM specifications. This is possible because the authentication is always going through the HLR, which is dealing with the computation of hashes and ciphers in some matter.

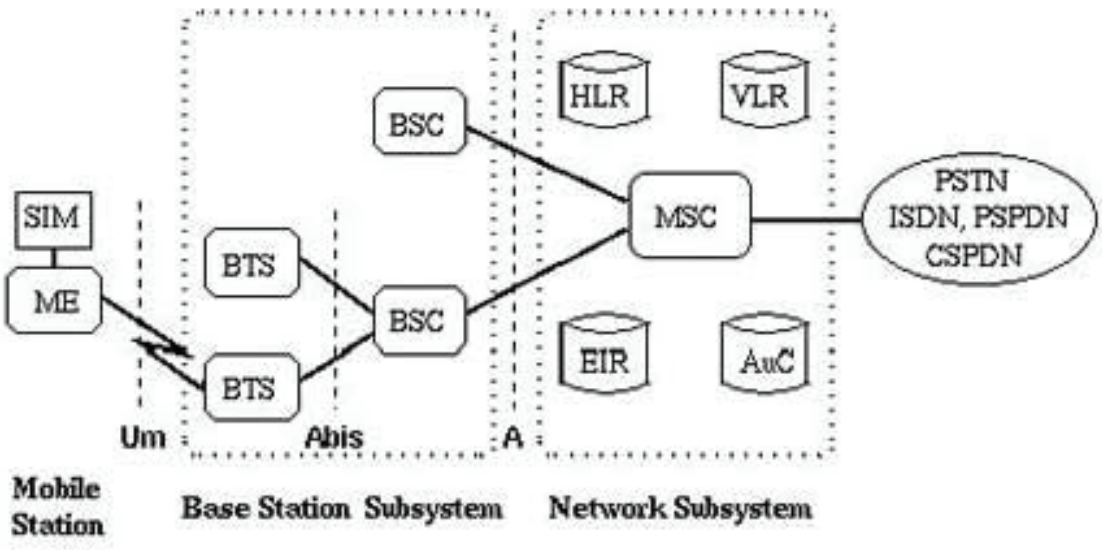


Figure 3 - General architecture of a GSM network

We will have a closer look at the security mechanisms in the next chapter, when we have looked at the general GSM network architecture, which is the foundation of all the security functions.

2.2.1 GSM network architecture

The GSM network is composed of several functional entities which can be divided into three main parts, as depicted in figure 3:

- Mobile Station (MS)
- Base Station Subsystem (BSS)
- Network Subsystem (NS)

2.2.1.1 Mobile Station (MS)

The MS is carried by the subscriber and consists of the physical Mobile Equipment (ME) and the Subscriber Identity Module (SIM). The SIM is independent of the ME, which means the SIM provides personal mobility. It allows the subscriber to switch between different mobile equipment and still have access to the subscribed services.

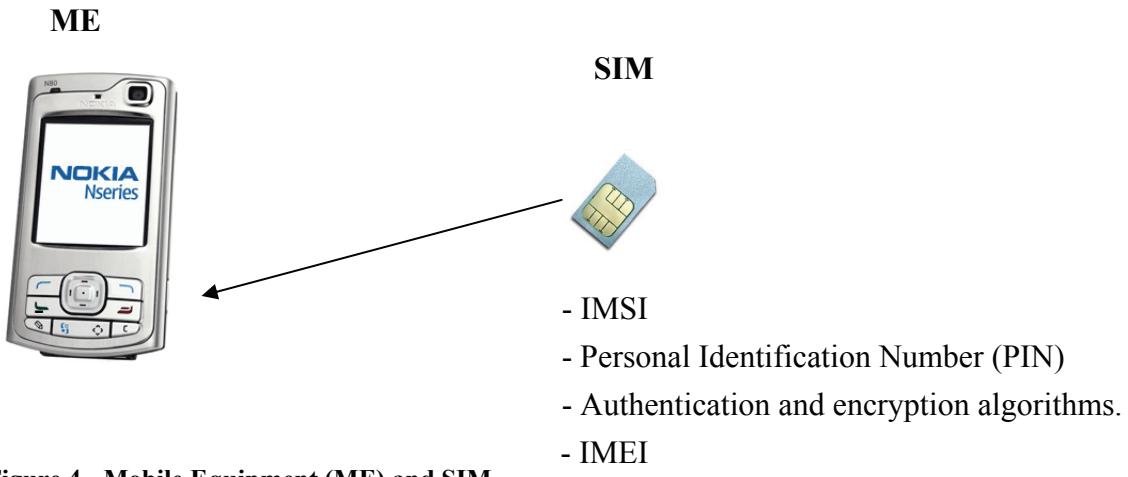


Figure 4 - Mobile Equipment (ME) and SIM

Mobile Equipment (ME)

The most common mobile equipment today is the handheld mobile telephone. The main purpose of the ME is to provide an interface to either a human user, via a microphone, loudspeaker, display and keyboard, or an interface to some other equipment such as a PC. Without a valid SIM card, GSM service is not accessible, except for emergency calls, according to 3GPP Specification 02.03 [17].

The ME is uniquely identified by the International Mobile Equipment Identifier (IMEI). The IMEI is a 15-digit number which includes information on the origin, model, and serial number of the device. It is used by the GSM network to identify valid devices and can hence be used to block stolen devices from accessing the network. The IMEI can be retrieved on most devices by typing *#06#. It is also printed underneath the battery.

Subscriber identity Module (SIM)

Similar, the SIM is uniquely identified by the International Mobile Subscriber Identity (IMSI) which is used to identify the subscriber to the system. The SIM also contains secret subscriber key and other algorithms used for authentication and encryption. We

will have a closer look at the MS security functions later on, since it is an important part of this assignment.

The SIM is originally protected by a Personal Identity Number (PIN), but the subscriber can disable this feature. The SIM card is tamper resistant, which means no one can edit or retrieve sensitive information stored in the SIM card [18].

2.2.1.2

2.2.1.3 Base Station Subsystem (BSS)

The Base Station Subsystem (BSS) is the physical equipment used to give radio coverage to a cell. It also has the equipment needed to communicate with the MS's. Figure 3 shows the relationship between the BSS and the rest of the entities in the GSM network. The BSS is not actually involved in the authentication process, so this part is only covered briefly. The BSS is composed of two parts: the Base Transceiver Station and the Base Station Controller. [16]

Base Transceiver Station (BTS)

The BTS contains the equipment for transmitting and receiving of radio signals. It houses the radio transceivers that define a cell and it handles the radio link protocols with the MS.

Base Station Controller (BSC)

The BSC manages the radio resources for one or more BTS's. It is the connection between the MS and the network subsystem and it handles the radio channel setup, frequency hopping and handovers.

2.2.1.4 Network Subsystem (NS)

The NS is performing switching functions and manages the communication between MS's and the PSTN. The central entity of the NS is the Mobile Switching Centre.

Mobile Switching Centre (MSC)

The MSC is the anchor in the GSM network (shown in figure 3). It holds all the switching functions needed for MS's located in an MSC area. It acts like a normal switching node of a regular PSTN or ISDN, and additionally it provides, in cooperation

with other functional entities, all the functionality needed to handle a mobile subscriber. This implies registration, authentication, location updating, handovers and roaming.

Home Location Register (HLR)

The HLR is a database containing information of every subscriber that is authorized against the GSM network. The HLR stores the following administrative information [19]:

- IMSI
- The Mobile Station ISDN Number } Primary keys
- The VLR address (the current location of the MS)

Visitor Location Register (VLR)

The VLR dynamically stores subscriber information when a MS is located in the area. Together with HLR and MSC, it provides the call routing and the roaming capabilities of GSM. To simplify the signaling, the VLR is usually implemented together with the MSC. This means that the geographical area controlled by the MSC corresponds to that controlled by the VLR.

Authentication Center (AuC)

The AuC is in charge of providing the authentication key used for authorizing the subscriber access to the GSM network. It is a protected database that stores the secret subscriber key (Ki) from the subscriber's SIM card. This is the only entity, except for the SIM card itself, which have access to this key.

Equipment Identity Register (EIR)

The EIR is a database of blacklisted cell phones. It contains the IMEI of all cell phones reported stolen. When a stolen handset connects to the network and the network reads the IMEI, the operator can disable it electronically. But unfortunately, not every operator is actually checking this blacklist, because it is not a requirement in the GSM specifications [20].

2.3 GSM SIM authentication

Before we dig into the GSM SIM authentication schemes, we need a clear understanding of all the security components and entities in GSM.

2.3.1 GSM security components

The security features of GSM are implemented in different parts of the GSM system as depicted in figure 5:

- SIM card
- Mobile Equipment
- GSM network

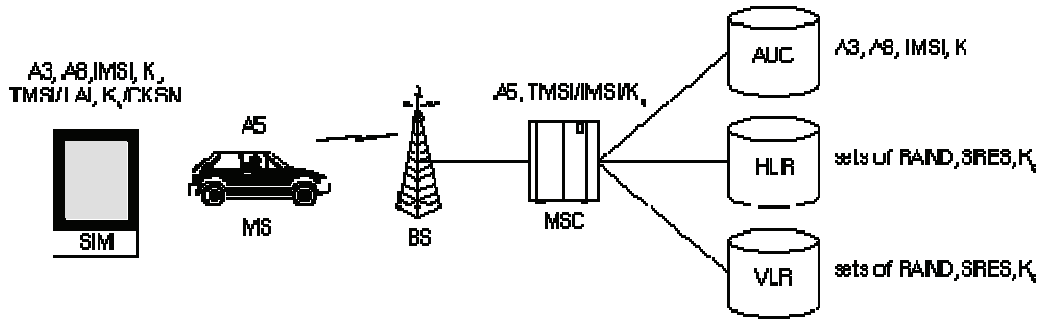


Figure 5 - GSM security components

2.3.1.1 SIM card

There are two types of SIM-cards specified in [18]: "ID-1 SIM" and the "Plug-in SIM". The physical characteristics of both types shall be in accordance with ISO/IEC 7816-1 and 7816-2. The Plug-in SIM has the exact same behavior and functionality as the ID-1 SIM. The only difference is the size. The ID-1 SIM has the dimensions of a full size Smart Card, similar to a credit card. The Plug-in SIM is the most used card nowadays and from now on when discussing the SIM card we refer to the Plug-in SIM, the smallest card to the right on figure 6 below.

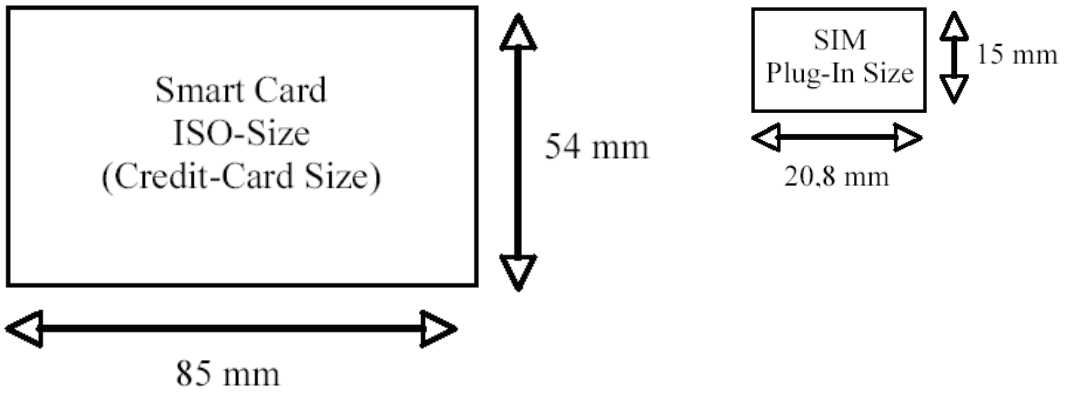


Figure 6 - Physical dimensions of ID-1 SIM and Plug-in SIM

The SIM card itself is a Smart Card containing keys, identifiers and algorithms. The Smart Card is actually a single chip-computer, an Integrated Circuit Card (ICC), containing an OS, a file system and stand-alone applications.

The SIM is the ICC defined for the second generation of GSM (2G), and is a physical and logical entity according to the 3GPP technical specifications [21]. In 3G there have been some modifications. The Universal Integrated Circuit Card (UICC) is introduced. The UICC may support both 2G and 3G networks. In a 2G network it contains a SIM application and in a 3G network it contains a USIM application.

Unlike the SIM, the USIM is not a physical entity, but a purely logical application that resides on a UICC. It does only accept 3G commands and is therefore not compatible with a 2G ME. But the USIM may provide mechanisms to support 2G authentication and key agreement to allow a 3G ME to access a 2G network.

The SIM provides storage of three types of subscriber related information:

- *Data attached during the administrative phase*; e.g. IMSI, subscriber authentication key and access control class.
- *Temporary network data*; e.g. TMSI, LAI, Kc
- *Other service related data*; e.g. Language preferences, advice of charge, telephone numbers etc.

The SIM also contains some pre-installed keys and algorithms provided by the operator:

- Subscriber authentication key (Ki)
- Authentication algorithm (A3)
- Cipher key generation algorithm (A8)
- Personal Identification number (PIN)

Subscriber authentication key (Ki)

Ki is a 128 bit key used for authentication of the subscriber by the operator. The safety of GSM depends on the secrecy of this key. If Ki is compromised it is possible to clone the SIM-card. Therefore it is only stored two places: On the tamper resistant SIM-card and in the secure AuC. (In figure 5 it is recognized as K). To keep it secret Ki is never transferred directly over the air interface. It is only used in combination with other keys and input parameters. Since no one else in the GSM network knows this key, AuC is the only one who is able to compute the triplet needed in the authentication of the subscriber.

Authentication algorithm (A3)

A3 is a one-way function and is located in the SIM card and in the AuC. It is used in the challenge-response mechanism of the SIM authentication. (See figure 8).

Cipher key generation algorithm (A8)

The A8 algorithm is also a one-way function using the same mechanism as A3, to establish a cipher key Kc for encrypting user and signaling data on the radio path. It generates a 64 bit session key (Kc) from the 128 bit RAND and 128 bit Ki.

Personal Identification number (PIN)

The PIN or Card Holder Verification (CHV) is a 4 to 8 digit code used to authenticate the subscriber against the SIM card. The PIN is provided by the operator and is stored on the SIM card.

2.3.1.2 Mobile Equipment (ME)

The ME contains a cipher A5, used for enciphering/deciphering data against the MSC over the air interface. A5 is a stream cipher, which means it is implemented very efficiently on hardware. The drawback is that the algorithm has leaked to the public, so it is not completely safe anymore. But it is not used in the authentication process.

2.3.1.3 GSM network

The MSC is the anchor in the GSM network as explained in section 2.2.1.4. Even though it serves the MS with RAND's and compares the results of different calculations, it does not store this information. RAND and SRES, together with Kc, will be stored in triplets and is kept by the HLR and the VLR. The MSC will keep the A5 ciphering algorithm and the cipher key Kc during the session, to be able to decrypt conversations.

International Mobile Subscriber Identity (IMSI)

The IMSI is stored in the SIM card but also in the AuC. It is not only a serial number identifying the MS. It also reveals the manufacturer, the country of production and type approval. The IMSI is only used when initializing the connection. Otherwise a temporary identifier is used, to protect the subscriber.

Temporary Mobile Subscriber Identity (TMSI)

TMSI is used instead of IMSI to prevent an eavesdropper from identifying the subscriber. For every location update involving a new MSC, the MS (SIM card) is assigned a new TMSI. The TMSI is also stored in the VLR, which will keep track of all the subscribers residing in the area.

2.3.2 Authentication schemes

2.3.2.1 Subscriber-SIM authentication

The subscriber is first met by a simple one-token authentication mechanism. A 4 to 8 digit Card Holder Verification (CHV), also known as Personal Identification Number (PIN). The PIN is stored on the SIM-card and is usually shipped to the subscriber independent of the SIM-card. Such a mechanism is useless in a radio environment, since listening once to this PIN is enough to break the protection. But this mechanism is only used at the client side and thus it is never transmitted via the radio path.

By authenticating the user to the SIM (See figure 7 below), the system provides a simple but effective protection against the use of stolen cards. The user is allowed to change the PIN or even remove the protection. If a wrong PIN is typed more than 3 times, the SIM-card will be locked until an 8 digit Unblock CHV / Personal Unblocking Key (PUK) is entered. If the PUK is entered wrong 10 times, the SIM will be permanently blocked and completely unrecoverable.

Depending on the requirements of the SIM issuer, and subject to the features incorporated in the SIM, a second CHV (PIN2) may be provided. Like PIN, the PIN2 shall also consist of 4 to 8 digits. There shall be no provision for the subscriber to disable PIN2. Another requirement according to the specifications is that it shall not be possible to read the PIN or PUK [18].

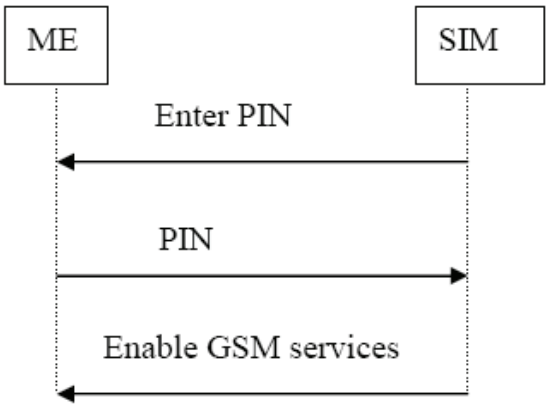


Figure 7 - GSM SIM initial user authentication

2.3.2.2 SIM-GSM network authentication

When the user is authenticated against the SIM, the SIM must authenticate against the GSM network before the subscriber is allowed to use the GSM services. The authentication is initiated by the fixed network, and it is based upon a simple challenge-response protocol. There are two different scenarios the subscriber can land in:

1) The subscriber is located in a cell which belongs to a network never visited before or at least not in the near past. The MS presents its IMSI to the serving network and the MSC contacts the MS's HLR and asks it to send a triplet containing RAND, SRES and Kc. The triplet is computed by the AuC, which is the only entity in the GSM network knowing Ki, beside the MS itself.

2) The other possible case is when the subscriber is located either in its own home network, or in a recent visited network. If an unused authentication triplet is still available in the VLR, the HLR of the MS does not need to be contacted. But if there are no unused triplets left, the AuC must be contacted nevertheless.

In both cases, the actual authentication mechanisms are equal. When the network has identified the MS by the IMSI, it sends a new RAND to the MS. The MS computes a response SRES using an algorithm A3 according to figure 8. On the network side, the MSC compares the received SRES with the SRES' computed by the AuC. If SRES and SRES' are equal, the SIM has been authenticated to the GSM network and the user is able to start using the subscribed services.

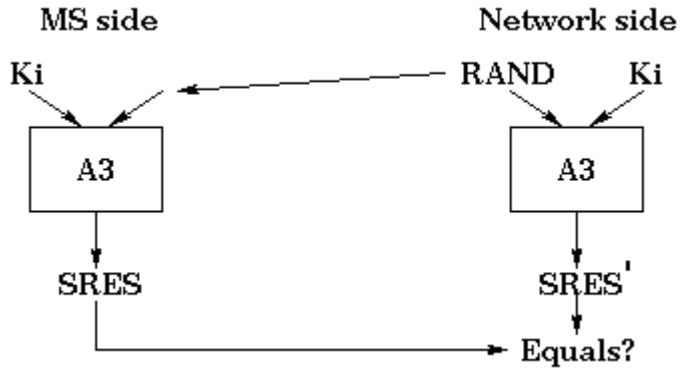


Figure 8 - The GSM SIM authentication scheme

The A3 is a one-way function, which means it should be easy to compute SRES from Ki and RAND, whereas the computation of Ki knowing RAND and SRES should be as complex as possible. Beyond this requirement, the only constraint imposed on A3 is the size of the RAND and the SRES. The RAND must be 128 bits long and the SRES must be 32 bits long. The Ki can be any format and length [16].

The same mechanism is used to generate a cipher key Kc for encrypting user and signalling data on the radio path. Ki and RAND are fed into A8 as showed in figure 9, and a 64 bit session key Kc is generated. The BTS receives the same Kc from the MSC, since the AuC knows the Ki and is able to generate the same Kc. The Kc is used until the MSC decides to authenticate the MS again.

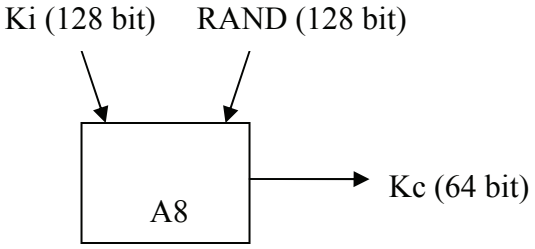


Figure 9 - Session key generation with the A8 algorithm

The Kc on the network side is precomputed by the AuC that serves the subscriber's home network. The precomputed triplets containing RAND, SRES and Kc, is passed from the AuC to the VLR on demand and is only used once.

In practice, SRES and Kc are generated together on one run. This is done with a function called COMP128 (See figure 10). COMP128 takes the 128 bit RAND and the 128 bit Ki and it generates an output of 128 bits. The first 32 bits is the SRES response and the last 54 bits become the Kc. Ten zero-bits are appended to the 128 bit key generated by the COMP128 algorithm. This means the last ten bits of Kc are zeroed out, and the actual key space is by some reason reduced from 64 to 54 bits.

The operators can choose whether they will store COMP128, or both A3 and A8 in the SIM card. Both methods will protect against tampering. They can choose algorithms independently from hardware manufacturers and other network operators. This leads to an important aspect of the GSM standard. Telenor and Netcom, two major telecom

operators in Norway, may use different algorithms in the SIM cards they provide to their customers. When a subscriber is performing roaming between the two operators, the local network will ask the HLR of the subscriber's home network for the triplets (RAND, SRES and Kc). This means the local network does not know anything about the algorithms used [16].

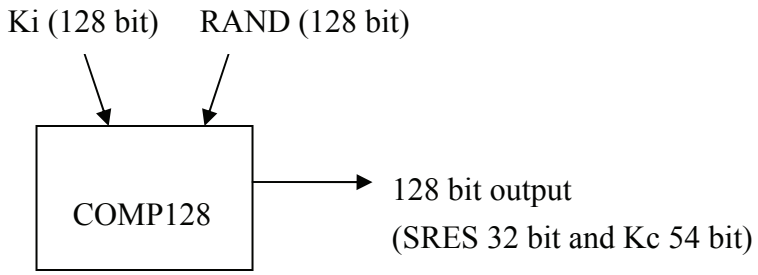


Figure 10 - COMP128 algorithm generating SRES and Kc at once

2.3.3 Security considerations

There have been numerous attacks on GSM security since 1998, when the security algorithms leaked to the public and the vulnerabilities of the system were exposed. The algorithms were originally kept secret, which is a bad idea in the means of security. It is well known that the algorithms in security systems should be open and tested by many independent security experts, and that the security should be in the key.

As long as the subscriber occupies the SIM, the possibility of fraud is low. Most of the (effective) attacks are based on physical access to the SIM. But regardless of broken security algorithms, the GSM architecture will still be vulnerable against attacks on the operator's backbone network. The link between BTS and BSC is often an unencrypted point-to-point microwave link which is a major security hole in the GSM system.

2.3.3.1 SIM attacks

Attacks on A3/8 algorithm

The security of GSM is based on the secret key Ki. If this key is compromised the security for that subscriber is lost and it could be possible to eavesdrop on calls or run calls on the original subscriber's bill.

In April 1998 the Smart Card Developer Association and the ISAAC security research group discovered a flaw in the COMP128 v.1 algorithm. This made it possible to retrieve the Ki from the SIM by a chosen plaintext attack. This method requires physical access to the SIM. Another way of obtaining Ki is to use a false BTS to send the RAND over the air interface. This would take several days, but the attacker does not need physical access to the SIM. Anyway the COMP128 and A3/8 is not considered safe anymore, due to its weaknesses. Unfortunately some operators may still be using this version of the algorithm, since the standards does not specify which algorithm to use.

Several revised versions of the COMP-128 A3/A8 algorithm have been devised after the publication of these weaknesses and the publicly specified GSM-MILENAGE algorithm [22] is not vulnerable to any known attacks within January 2006 [23].

Side channel attacks

This kind of attacks is only possible with physical access to the SIM. The "partition attack" developed by IBM researches makes it possible to obtain Ki within minutes, if there are some minor deviations from the standards or if counter measurement against differential side channel analysis have not been properly applied [23].

2.4 EAP-SIM

2.4.1 Introduction

Extensible Authentication Protocol (EAP) specified in [25] is an authentication framework supporting multiple authentication methods. Since it is a framework and not a specific method it has a wide area of application. It can be used on dedicated links, switched circuits and wireless links. It was originally designed for use with the Point-to-Point Protocol (PPP) in network access authentication, also known as 802.1X or "EAP over LAN".

EAP does only support a single packet on flight, and hence it cannot efficiently transport bulk data like TCP. But that is not the intention either. EAP is used to select a specific authentication mechanism and it permits the use of a backend authentication server (AAA-server), which may implement some or all authentication methods. Hence the authenticator does not need to be updated to support each new authentication method. The AAA-server is dedicated for this purpose.

EAP-SIM is a mechanism for authentication and session key distribution using the GSM SIM. The EAP-SIM mechanism specifies the following enhancements to GSM authentication and key agreement (AKA) [24]:

- Multiple authentication triplets can be combined to achieve greater strength than individual GSM triplets
- It enables network authentication (mutual authentication between the parties)
- Supports user anonymity
- Supports a fast reauthentication procedure.

2.4.2 Authentication procedure

The EAP-SIM authentication scheme is quite comprehensive, but at the same time it is efficient and also very user friendly. But most important, it is very secure. It enables strong and mutual authentication between the parties and it is cost effective compared to other similar technologies.

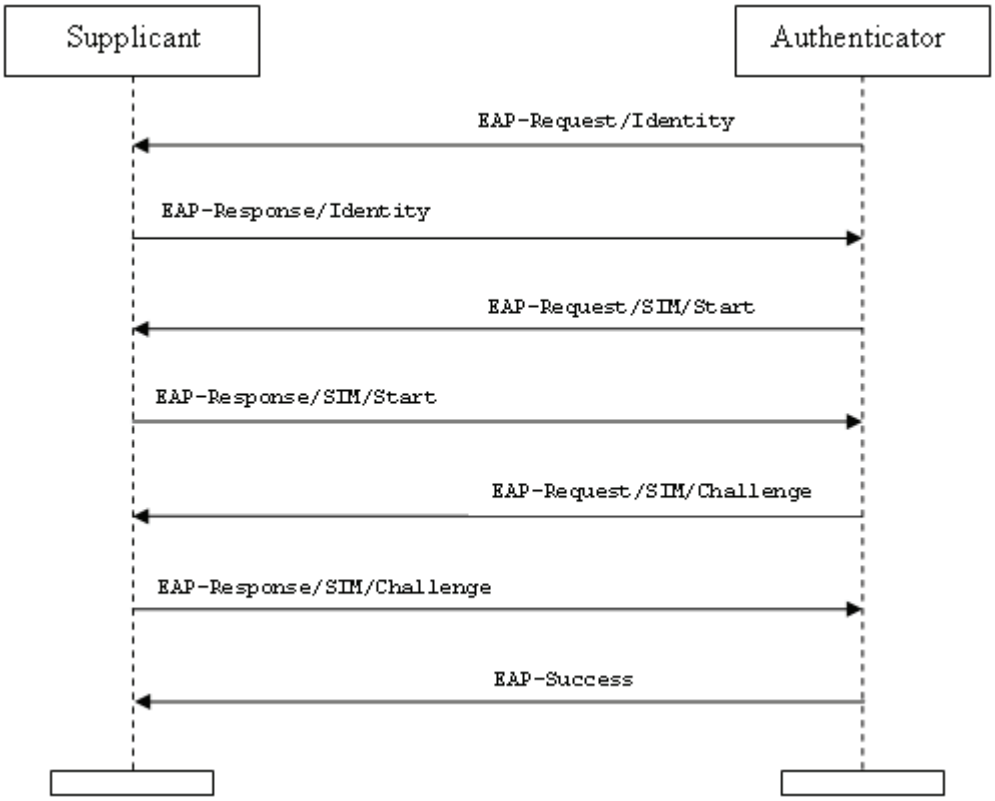


Figure 11 - EAP SIM full authentication procedure

The authenticator to the right on figure 11 is acting as a proxy between the supplicant and the authentication server. The authentication server is not shown in the figure.

The first request issued by the authenticator is EAP-Request/Identity. The supplicant's response includes either the user's IMSI or a temporary identity if identity privacy is in effect. The next request issued by the authenticator is the EAP-Request/SIM/Start packet which contains the list of EAP-SIM versions supported by the EAP server.

The supplicant responds with the EAP-Response/SIM/Start packet, which includes a selected version number and a selected random number NONCE_MT. The NONCE_MT, the version list and the selected version number is used by the authenticator to generate the master key as described below in section 2.4.2.1.

After receiving the EAP Response/SIM/Start, the authenticator obtains a number of GSM triplets for use in authenticating the subscriber. The triplets may be obtained by

contacting the HLR/AuC in the GSM network. Between 1 and 5 triplets may be obtained at a time. Triplets may be stored in the authentication server for use at a later time, but a triplet can not be used more than once.

The next request from the authenticator contains the RAND challenges and a Message Authentication Code (MAC) attribute that cover the challenges. The supplicant computes its own MAC over the received challenges, and compares that with the received AT_MAC. If the MAC's do not match, a network error may have occurred or someone has been trying to tamper with the packet. Anyway, the supplicant responds with an EAP-Response/SIM/Client-Error packet, and the authentication procedure terminates. If they do match the supplicant knows for sure that the other part possesses the valid GSM triplets, since the NONCE_MT value generated by the supplicant contributes to the MAC. This procedure is described further in section 2.4.2.1.

The supplicant will then run the GSM authentication algorithm to calculate the SRES value, based on the RAND challenge retrieved from the authenticator. The supplicant computes a new AT_MAC value, which covers the SRES, and responds with the EAP-Response/SIM/Challenge. The authenticator verifies that the MAC is correct and the authentication server compares the SRES received from the supplicant with the one retrieved from the GSM network. The procedure ends with an EAP-success message from the authenticator, if everything went well [24].

2.4.2.1 Master key generation and integrity

To ensure the integrity of the messages exchanged between the supplicant and the authentication server, a one-way hash function (SHA-1) is used to create a master key MK. The master key is used as a secret key in the generation of the Message Authentication Codes (MAC) in EAP-SIM.

The authenticator generates the master key when it gets the EAP-Response/SIM/Start packet from the supplicant, which includes the random number NONCE_MC, the version list and the selected version number. It concatenates these values with the underlying GSM session keys (Kc) and the subscriber identity retrieved in the past, and uses a one-way function to generate the master key.

MK = SHA1(Identity | n*Kc | NONCE_MT | Version List | Selected Version).

The MK is used when the authenticator calculates the MAC value over the EAP-SIM messages, to achieve mutual authentication between the parties. When the supplicant retrieves an EAP-SIM message from the authenticator, covered by the MAC, it can compute its own version of the MAC and compare with the MAC included in the EAP-SIM message. I.e. the supplicant is able to verify that the EAP-SIM message retrieved is fresh, and not a reply, and that the sender possesses valid GSM triplets for the subscriber, since the Kc's are concatenated in the master key [24].

2.4.3 Security considerations

This section outlines the security properties and the vulnerabilities in the EAP-SIM protocol, according to the EAP-SIM specification [24].

2.4.3.1 A3 and A8 Algorithms

The security of the A3 and A8 algorithms is important to the security of EAP-SIM. Some A3/A8 algorithms have been compromised as described in section 2.3.3.1, and because the operation of these functions completely falls within the domain of an individual operator, this is considered as vulnerability in EAP-SIM.

2.4.3.2 Identity Protection

EAP-SIM includes optional identity privacy support that protects the privacy of the subscriber identity against passive eavesdropping by introducing a pseudonym (Temporary identity). A client/subscriber that has not yet performed any EAP-SIM exchanges does not typically have a pseudonym available and then the privacy mechanism cannot be used unless the permanent identity is sent in clear. An active attacker that impersonates the network may use the AT_PERMANENT_ID_REQ attribute to get hold of the subscriber's permanent identity.

2.4.3.3 Mutual authentication and triplet exposure

The EAP-SIM provides mutual authentication and the security of EAP-SIM is based on the secrecy of Kc keys, which is included in the triplets. If someone gets physical access to the SIM card, it is easy to obtain any number of GSM triplets.

In GSM, the network is allowed to re-use the RAND challenge in consecutive authentication exchanges. This is not allowed in EAP-SIM. The EAP-SIM server is

mandated to use fresh triplets (RAND challenges) in consecutive authentication exchanges. This is improved in the UMTS Authentication and Key Agreement (AKA).

2.4.3.4 Flooding the AuC

A malicious EAP-SIM entity may generate a lot of protocol requests to mount a denial of service attack (DoS). The EAP-SIM server should take this into account and limit the traffic that it generates towards the AuC, preventing the attacker from flooding the AuC.

2.4.3.5 Key derivation

There is no known way to obtain complete GSM triplets by mounting an attack against EAP-SIM. An attacker may obtain n*RAND and AT_MAC values from the EAP server for any given subscriber identity, but calculating the Kc and SRES values from AT_MAC would require the attacker to reverse the keyed message authentication code function HMAC-SHA1-128. I.e. the key derivation mechanisms regarding EAP-SIM is computational secure.

2.4.3.6 Confidentiality

Confidential information must not be transmitted in EAP Notification packets.
An eavesdropper will see the EAP-Request/Notification, EAP-Response/Notification, EAP-Success, and EAP-Failure packets sent in the clear. This is because EAP-SIM is not a tunneling method and these packets are covered by the confidentiality mechanisms.

2.5 Java 2 Platform Micro Edition (J2ME)

"Java™ Platform, Micro Edition (Java ME) is the most ubiquitous application platform for mobile devices across the globe. It provides a robust, flexible environment for applications running on a broad range of other embedded devices, such as mobile phones, PDA's, TV set-top boxes, and printers" [26]

In June 1999 Sun Microsystems introduced J2ME, a collection of technologies and specifications that developers can use to construct complete Java runtime environments that closely fits the requirements of a particular range of devices. Each combination is optimized for the memory, processing power, and I/O capabilities of a related category of devices. J2ME is divided into configurations, profiles and optional packages. An overview is given in figure 12.

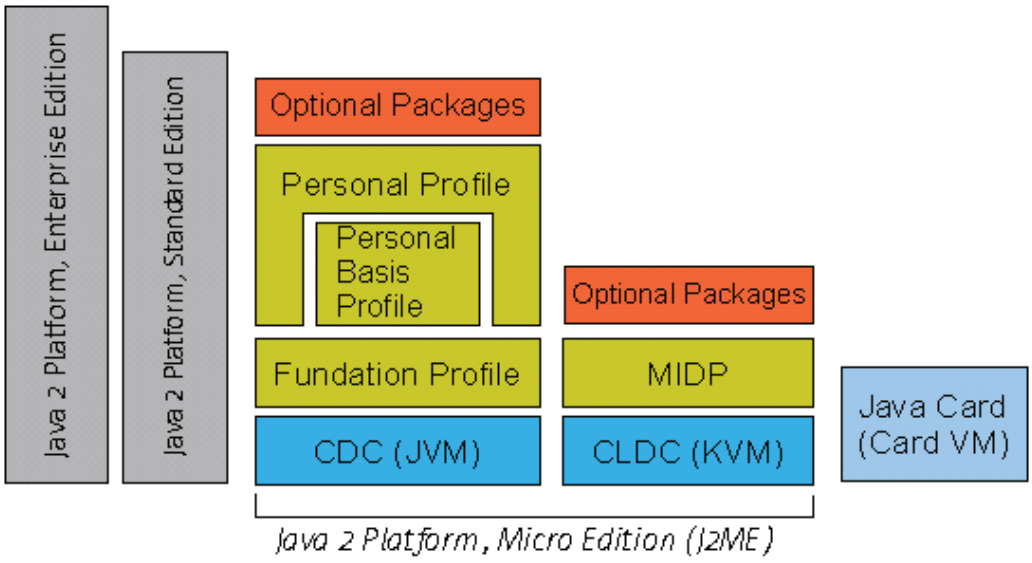


Figure 12 - Overview of Java 2 Platform, Micro Edition

2.5.1 Configurations

Connected Limited Device Configuration (CLDC) is developed for small and limited devices (in terms of memory and processing power) like mobile phones and PDA's.

The Connected Device Configuration (CDC) is for larger devices with robust network connections. Set-top boxes, Internet appliances and embedded servers, but high-end mobile devices fit this configuration as well.

2.5.2 Profiles

A profile is a set of higher-level APIs that further define the application life-cycle model, the user interface, persistent storage and access to device-specific properties. The Mobile Information Device Profile (MIDP) in connection with the CLDC provides a complete Java application environment widely adopted by mobile phones. The MIDP 2.0, which most mobile phones support today, offers a great deal of functions ranging from multimedia and 3D games, to PKI and messaging.

The development of the next generation profile, MIDP 3.0 started on March 2005 and it is currently being developed under JSR 271.

2.5.3 Security mechanisms

Any information transmitted over wireless links is subject to interception. Some of that information could be sensitive, such as credit card numbers and other personal data. To make handheld wireless devices more useful in an enterprise setting, applications must protect their users' information, using encryption, authentication, and secure communications protocols [27].

MIDP 2.0 supports the Secure Hyper Text Transfer Protocol (HTTPS), which enables Secure Socket Layers (SSL) to be used for securing wireless transactions. According to the specifications in [28], MIDP 2.0 devices are expected to operate using standard Internet and wireless protocols and techniques for transport and security.

Further the specifications say that all devices must conform to all mandatory requirements in the Wireless Application Protocol (WAP) Certificate and CRL Profiles Specifications [29]. I.e. the certificate profiles defined in the WAP specification have to

be supported by all MIDP 2.0 enabled devices.

2.5.4 Optional packages

With the optional packages it is possible to extend the technology stack with new and emerging technologies like wireless messaging, VoIP and Web Services. Since the optional packages are modular the device manufacturers can avoid implementing unnecessary functionality by including only the packages an application actually needs. Optional packages can be implemented alongside virtually any combination of configurations and profiles.

Specifications for the Java platform are developed under the Java Community Process (JCP). A java specification begins life as a Java Specification Request (JSR) and an expert group under the JCP creates the specification. J2ME specifications are usually referred to by the JSR number.

2.5.4.1 SATSA / JSR177

The Security and Trust Services API for J2ME (JSR177) extends the security features for the J2ME platform, through the addition of cryptographic APIs, digital signature service, and user credential management. SATSA also defines the methods to communicate to a Smart Card, by leveraging the APDU protocol and Java Card Remote Method Invocation (JC RMI). The specifications are developed by the JSR 177 expert group with representatives from Nokia, Sony Ericsson, Motorola, Siemens, VeriSign etc. The SATSA specification [30] defines four distinct APIs:

- SATSA-APDU allows applications to communicate with smart card applications using a low-level protocol. This API is very relevant for this assignment!
- SATSA-JCRMI provides an alternate method for communicating with smart card applications using a remote object protocol.
- SATSA-PKI allows applications to use a smart card to digitally sign data and manage user certificates.
- SATSA-CRYPTO is a general-purpose cryptographic API that supports message digests, digital signatures, and ciphers.

Requirements and status

The SATSA is a new and emerging technology and there are small amounts of public

documentation. Most of this discussion is based on different threads from the Nokia Forum and blogs from Forum Nokia expert Hartti Suomela. We have no other choice than trust this information in anticipation of official documentation from the mobile manufactures.

Unfortunately only SATSA-PKI and SATSA-CRYPTO are implemented on Symbian Series 60 devices [31]. But there are some recently announced Series 40 phones (7373, 5300 and other S40 3rd Edition Feature Pack 2 devices) which have SATSA-APDU implemented, according to Hartti Suomela, Java expert in Forum Nokia [32]. The SATSA spec says there is no Smart Card access for untrusted (unsigned) MIDlets. Hence we have to sign the MIDlet with a certificate issued by a trusted third party (The operator or the manufacturer) to get access to the SATSA-APDU.

But hopefully the device manufacturers will implement APDU as a standard in regular mobile phones in the future. The Mobile Software Architecture (MSA) specification [33] defines that APDU, CRYPTO, and PKI has to be implemented in all MSA-compatible devices. The MSA defines the next generation Java platform for mobile handsets based on the CLDC. The primary design goal of the MSA Specification is to minimize fragmentation of mobile Java environments by defining a predictable and highly interoperable application and service environment for developers. This is a very important step and many J2ME developers, including the author will appreciate this!

An alternative is to access the SIM card through a C++ API for Symbian OS's. The drawback is that this API is not part of the Nokia SDKs for Symbian OS based phones. It is more device specific implementation that requires cooperation with the manufactures. You have to be a Symbian "Platinum Partner Program" member to get access to the source code of the API. "*The Platinum Partner Program is a program for companies with a technology, service or strategic position that is key to the success of Symbian OS phones in the market"* [34].

2.6 Authentication of mobile services

2.6.1 Authentication for WAP-based Services

For WAP (Wireless Application Protocol) application a WAP Identity Module (WIM) [38] is defined and used in performing WTLS (Wireless Transport Layer Security) [39] and application level security functions, and especially, to store and process information needed for user identification and authentication. The WIM functionality can be implemented on a smart card. A smart card implementation is based on ISO 7816 [40] series of standards. The WIM is defined as an independent smart card application, which makes it possible to implement it as a WIM-only card or as a part of multi-application card containing other card applications, like the GSM SIM.

2.6.2 Authentication for Web-based Services

For Web-based services accessed through a Web browser, stronger authentication is offered by using the One-Time-Password scheme. However, this solution is not the ideal one for mobile phone. The OTP must be generated by a device, which the user must bring along, or it can be sent to the user via SMS. Anyway, the user has to enter in the OTP manually, in addition to username and password, which might be a complicated procedure on small mobile handsets with poor keyboards.

Another option is to use a PKI-solution, which requires a PKI client installed in the SIM card as separate application. To carry out authentication, the Web site has to send challenges to the PKI Client using SMS as carrier. Only when the authentication is successful, the Web server will return to the mobile browser. Quite often, the browsing session has been terminated following of the termination of the data packet session, e.g. GPRS, UMTS.

2.6.3 Authentication for Java-based Services

The Java 2 Platform, Micro Edition (J2ME) is a Java platform optimized for small devices with limited memory and processing power, such as mobile phones and PDA's. J2ME is divided into configurations, profiles and optional packages. Devices need a configuration adapted to their processing capabilities and the profile implements higher-level APIs that further define the application life-cycle model, the user interface,

persistent storage and access to device-specific properties.

For J2ME applications there is recently defined the Security and Trust Services API JSR 177 which extends the security features for the J2ME platform, through the addition of cryptographic APIs, digital signature service, and user credential management. SATSA also defines methods to communicate to a Smart Card, by leveraging the APDU protocol. The SATSA spec says there is no Smart Card access for untrusted (unsigned) MIDlets. Hence one has to sign the MIDlet with a certificate issued by the operator or the manufacturer, to be able to connect to the SIM.
JSR-248 (Mobile Service Architecture), which defines the next generation Java platform for mobile handsets, mandates the support of SATSA-APDU when a security element exists on the device, i.e. a Smart Card or a SIM card. With SATSA, the necessary security functions are offered to the J2ME applications but the architectural problem is still not solved. It is not simple for applications to make use of these security functions and there is no point to require that each application must integrate the security functions.

3 Analysis

This chapter introduces the proposed SIM authentication system for mobile handsets. In short this chapter describes the requirements and identifies the necessarily components. In a "semi-agile" development process like the Unified Process (UP), the analysis is not a phase, but a discipline as described in section 1.5. The analysis is performed in several iterations, and it is a major part of both the inception and the elaboration phase. The requirements and the presumptions are often changing rapidly through a project, and the system must be able to adapt to small and major changes through the development process.

To be able to present the analysis work in a well arranged fashion, it is gathered here in this chapter.

3.1 Stakeholders and end-users

3.1.1 Stakeholders

Name	Description	Responsibility
Telenor ASA	Employer and possible identity provider.	Provide strong user authentication for service/application providers by utilizing the existing authentication mechanisms in the GSM network
Project group	MSc student Håvard Holje from department of Telematics at Norwegian University of Science and Technology (NTNU), supervisor Ivar Jørstad (Ubisafe AS) and professor Do van Thanh (NTNU and Telenor ASA)	Propose, design and implement a solution for utilizing the GSM SIM on a mobile handset as a generic authentication token in different applications.
Service/application providers	Business or organization offering some kind of services to end-users	Give users access to services, perform billing/accounting and QoS.

Table 1 - Stakeholders of the project

3.1.2 End-users

Name	Description	Responsibility
Potentially every user of a GSM compatible mobile handset	Users who want to use a mobile handset to get access to services like internet banking and other services requiring strong authentication.	Must keep the pin code secret and keep the SIM private.

Table 2 - End-users of the authentication system

3.2 High-level use case model

This section gives an overview of the actors and the different parts of the system. The relevant use cases are identified and described in a superior manner.

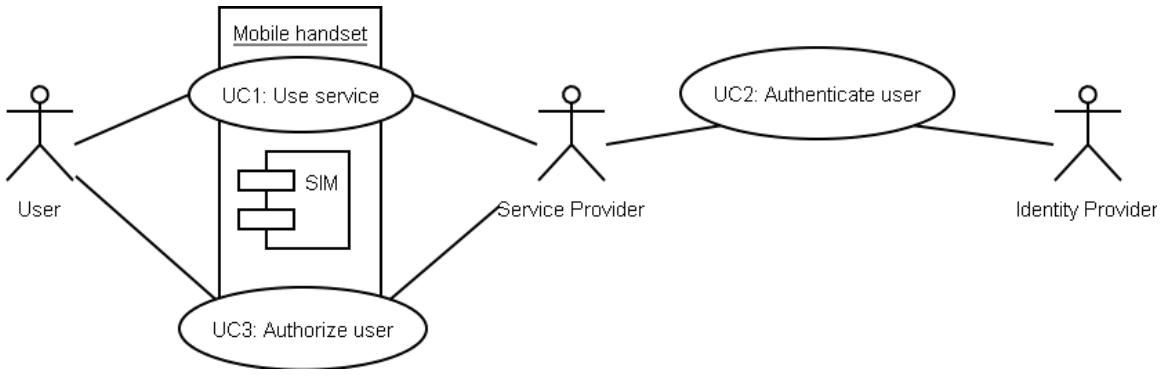


Figure 13 - High-level user case diagram

3.2.1 Use case 1 (UC1) – Use service

A user wants to access a service on a mobile phone and initializes the required application. (E.g. a WWW browser or a J2ME MIDlet). The application requests the Service Provider for the chosen service, via the GSM network. The service requires strong user authentication, and before the user is allowed to access the service he/she has to be authenticated against the Identity Provider (UC2: Authenticate user). When authentication is complete the Service Provider authorizes the client and grants access to the service, if the claimed authenticity is accepted (UC3: Authorize user).

3.2.2 Use case 2 (UC2) – Authenticate user

The SIM supplicant provides the subscriber identity and valid user credentials to the Identity Provider. The Identity Provider performs a lookup of the user in an associated authentication server and uses the existing GSM authentication mechanisms to authenticate the user.

3.2.3 Use case 3 (UC3) – Authorize user

When the Service Provider has got a service request including a Security Association from a user, it will verify the claimed authenticity against the IdP. If the provided Security Association is valid, the Service provider will create a new session ID and associate this value with the current user. If the client application performs more service requests it will be authorized immediately, if it includes the session ID received in the first service request.

3.2.4 Actors and system parts

Actor	Role	Description	Functionality and responsibility
User	Support actor	Human being or another system part who wants to get access to a service through a mobile handset	Using a mobile application to authenticate against a trusted service by the means of a build-in supplicant. Must provide a PIN to get access.
Mobile handset	System part, hardware	A GSM compatible device (usually a mobile phone)	The mobile handset is the device containing the SIM card and it communicates with the GSM network on behalf of the user.
SIM-card	System part, hardware	A small, tamper resistant smart card containing a secret key and different operator specific algorithms	The SIM makes it possible for the operator to authenticate the user against the existing rules and algorithms.
Mobile SIM Supplicant	The main actor, software	An application installed on the mobile phone which interoperates with either a WWW browser or a J2ME MIDlet.	The supplicant is responsible for performing mutual authentication between the service provider and the user. It consist of two parts: 1) Service supplicant which communicates with the service provider 2) SIM supplicant which communicates with the SIM card through a defined interface.
Service provider	System part	A company/organization offering services via a standalone application or a WWW browser on a mobile	The service provider is offering services to the user and therefore it has to keep track of users to be able to

		handset.	perform accounting
Identity provider (Authenticator)	System part	The authenticator provides mutual authentication between the user and the service provider.	Providing strong individual (identity) authentication to the service provider by combining the existing GSM SIM functionality with new and emerging technology.

Table 3 - Actors and system parts

3.3 Use case specification

This section describes the proposed system in more details. The functional requirements are outlined below in figure 14. Several of the uses cases could have been omitted by revising the structure. However, they are separated into several small use cases to emphasize all the possible combinations.

The supplementary specification follows in the subsequent section and specifies both the functional requirements not captured in the use cases and non-functional requirements. The template used to describe the use cases is based on [35] and [36].

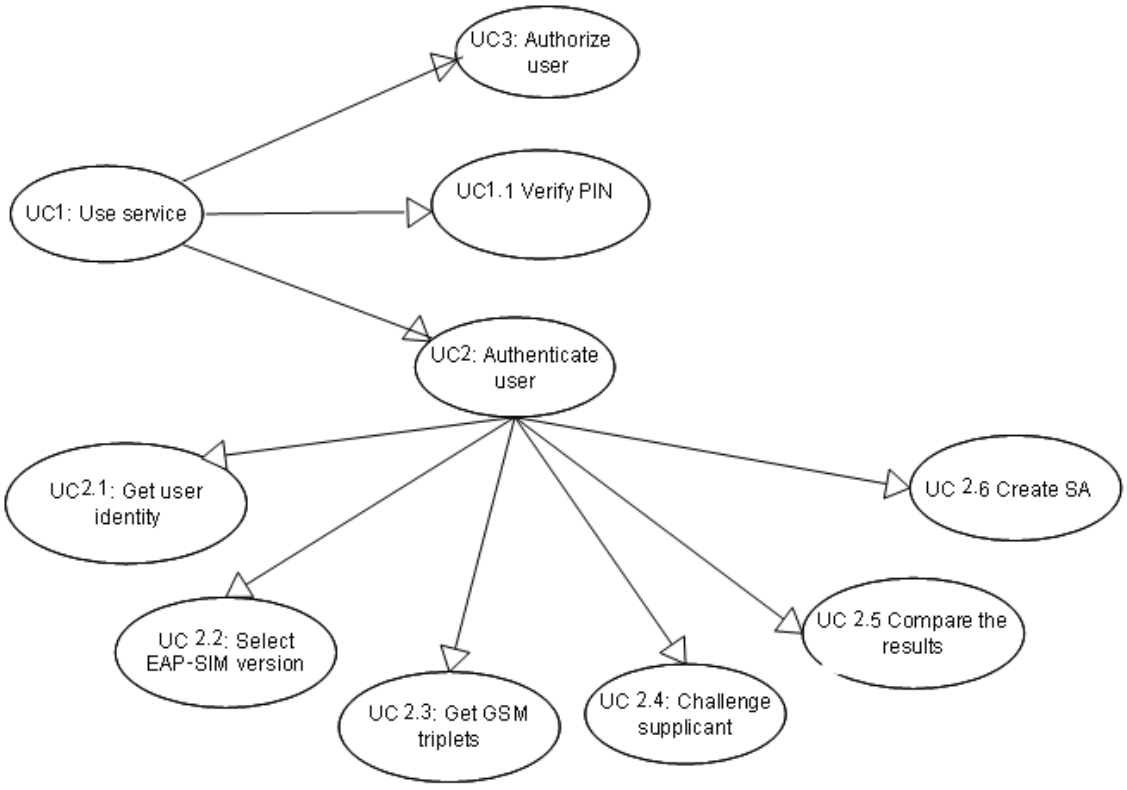


Figure 14 - Use case specification overview

Use Case UC1: Use service

Primary actor: User

Support actors:

- GSM SIM
- Supplicant
- Mobile handset (mobile phone)
- Service Provider

Stakeholders:

- Identity Provider

Precondition(s) / presumptions:

- The mobile handset has got the correct GSM packet network technology set-up.
- The user resides in an area with GSM coverage.

Postcondition(s): The user has fulfilled the task(s) he/she wanted to perform without any interruptions and the Supplicant was able to authenticate the user against the system and no one without proper rights was exploiting the system.

Main success Scenario (basic flow):

1. The user starts a client application. (It could be either a WWW browser or a stand-alone application like a J2ME MIDlet).
2. The client application will make a service request toward the Service Provider.
3. The Service Provider responds with an authentication request, i.e. a redirect to the local Supplicant.
4. The redirect activates the local Supplicant via the J2ME push registry.
5. The Supplicant is fired up in the background and requests the PIN code from the client application (Start UC1.1 Verify PIN)
6. The Supplicant initiates the authentication procedure by contacting the Identity Provider (Start UC2: Authenticate user)
7. When the client is authenticated against the Identity Provider, the client gets access to the requested service when it is authorized by the Service Provider. (Start UC3: Authorize client)

Use Case U1.1: Verify PIN

Primary actor: Supplicant

Support actors:

- User
- SIM

Precondition(s) / presumptions:

- The supplicant has got an authentication request from a client application
- PIN protection is activated on the SIM

Postcondition(s):

The SIM is opened up and ready for further communication with the Supplicant.

Main success Scenario (basic flow):

1. The Supplicant requests the PIN from the client application.
2. The client application shows the PIN form to the user and the user enters the PIN.
3. The Supplicant will open a connection to the SIM and provide the user PIN.
4. The SIM verifies the received PIN.

Extensions / alternative flows

4a. Wrong PIN, SIM can not verify it.

4b. If PIN is not verified after 3 attempts the authentication fails.

Use Case UC2: Authenticate user

Primary actor: Identity provider (IdP)

Support actors:

- Supplicant
- GSM SIM
- Mobile handset (mobile phone)
- Authentication server (RADIUS)

Stakeholders:

- User
- Service Provider (SP)

Precondition(s) / presumptions:

- The SP has an agreement with an IdP.
- The IdP communicates with an EAP server that is located on a backend authentication server (RADIUS server) using an AAA protocol.

Postcondition(s): The correct user is authenticated against the system

Main success Scenario (basic flow):

1. The IdP requests the Supplicant for subscriber identification (Start UC2.1: Get user identity)
2. The IdP will then start an EAP-SIM exchange between the authentication server and the supplicant by requesting the Supplicant for what EAP-SIM version to use (Start UC2.2: Select EAP-SIM version)
3. The IdP requests the necessarily GSM triplets from its associated authentication server (Start UC2.3: Get GSM triplets)
4. The IdP challenges the Supplicant to provide user credentials (Start UC2.4: Challenge supplicant)
5. The IdP requests the authentication server to compare the SRES retrieved from the SIM Supplicant with the SRES originating from the HLR. (Start UC2.5: Compare results)
6. The IdP creates a unique Security Association (SA) (Start UC2.6: Create SA).

7. The IdP completes the authentication procedure by sending an EAP-success message and the SA to the Supplicant.

Extensions / alternative flows

6a. The subscriber is not authenticated and the IdP will send an EAP-failure message to the Supplicant.

Use Case UC2.1: Get user identity

Primary actor: Identity provider (IdP)

Support actors:

- Supplicant
- GSM SIM
- Mobile handset (mobile phone)

Stakeholders:

- Service Provider (SP)

Precondition(s) / presumptions:

- The IdP has got a request from a SP to authenticate a user.
- The IdP knows where to find the specific user.

Postcondition(s): The IdP has got the correct use identity from the supplicant

Main success Scenario (basic flow):

1. The IdP sends an EAP identity request to the Supplicant
2. The Supplicant requests the SIM for its IMSI.
3. The Supplicant returns the extracted identity to the IdP.

Extensions / alternative flows

2a. If identity protection is used, the supplicant will get a TMSI instead of IMSI.

Issues

Extracting the identity from the SIM might be a security issue due to vulnerability in the GSM SIM specification when it comes to IMSI/TMIS. If the SIM gets a request for its IMSI it will return it no matter what, even if identity protection is in effect.

Use case UC2.2: Select EAP-SIM version

Primary actor: Identity provider (IdP)

Support actors:

- Supplicant
- GSM SIM
- Mobile handset (mobile phone)
- Authentication server (RADIUS)

Stakeholders:

- Service Provider (SP)

Precondition(s) / presumptions:

- The IdP has got a request from a SP to authenticate a user.
- The IdP knows where to find the specific user.
- The IdP has got a valid subscriber identity from the supplicant

Postcondition(s): The supplicant and the authentication server has agreed on which EAP-SIM version to use

Main success Scenario (basic flow):

1. The authentication server provides a list of valid EAP-SIM versions to the identity provider
2. The IdP requests the Supplicant for what EAP-SIM version to use and provides a list of possible EAP-SIM versions to choose from.
3. The Supplicant responds with the chosen EAP-SIM version and a random number, which is the supplicant's challenge to the network (used to achieve mutual authentication and to verify that the EAP-SIM message is fresh)

Use Case UC2.3: Get GSM Triplets

Primary actor: Identity provider (IdP)

Support actors:

- Authentication server
- GSM Gateway
- HLR

Stakeholders:

- Service Provider (SP)

Precondition(s) / presumptions:

- The IdP has got a request from a SP to authenticate a user.
- The IdP is in possession of the subscriber identity (IMSI/TMSI) and it knows where to find the supplicant (address).
- The authenticator communicates with an EAP server that is located on a backend authentication server (RADIUS server) using an AAA protocol.

Postcondition: The IdP is in possession of the necessarily GSM triplets needed to authenticate the given user.

Main success Scenario (basic flow):

1. The IdP will request the authentication server for GSM Triplets associated with the given user.
2. The authentication server will forward the request to a GSM gateway, which in turn will contact the HLR of the particular user.
3. The HLR will return a set of 1-5 triplets which is kept in the authentication server.

Extensions / alternative flows

2a. If the authentication server already possesses a valid GSM triplet for the particular user, step 2 and 3 will be omitted, and it will send a GSM challenge to the IdP immediately.

Use Case UC2.4: Challenge supplicant

Primary actor: Identity provider (IdP)

Support actors:

- Supplicant
- GSM SIM
- Mobile handset (mobile phone)

Stakeholders:

- Service Provider (SP)

Precondition(s) / presumptions:

- The IdP has got a request from a SP to authenticate a user.
- The IdP is in possession of the subscriber identity (IMSI/TMSI) and it knows where to find the supplicant (address).
- The authentication server is in possession of a set of GSM triplets belonging to the current user.
- The authenticator communicates with an EAP server that is located on a backend authentication server (RADIUS server) using an AAA protocol.

Postcondition: The IdP is in possession of the necessarily user credentials needed for authenticating the user.

Main success Scenario (basic flow):

1. The IdP will request the authentication server for a RAND challenge.
2. The authentication server responds with a RAND covered by a MAC.
3. The IdP challenges the supplicant with the RAND.
4. The supplicant combines the received RAND with the user credentials extracted from the SIM and returns SRES to the identity provider, covered by a MAC.

Extensions / alternative flows

2a. If the authentication server already possesses a valid GSM triplet for the particular user, this step will be omitted.

Use Case UC2.5: Compare the results

Primary actor: Identity provider (IdP)

Support actors:

- Authentication server

Stakeholders:

- Supplicant
- Service Provider (SP)

Precondition(s) / presumptions:

- The IdP has got a request from a service provider to authenticate a user.
- The authenticator communicates with an EAP server that is located on a backend authentication server (RADIUS server) using an AAA protocol.
- The authentication server is in possession of a set of GSM triplets belonging to the current user.
- The IdP is in possession of the subscriber identity (IMSI/TMSI) and it knows where to find the supplicant (Address).
- The Supplicant has been challenged and has provided its user credentials (SRES) to the IdP.

Postcondition: The user is authenticated against the authentication server

Main success Scenario (basic flow):

1. The IdP provides the authentication server with the SRES computed by the supplicant, and request the authentication server to approve it.
2. The authentication server will compare the received SRES with the SRES' originating from the HLR.
3. The authentication server sends an EAP-Accept message to the identity provider

Extensions / alternative flows

2a. SRES does not equal SRES' Go to step 3a.

3a. If the authentication fails, the authentication server sends an EAP-Failure message to the IdP.

Use Case UC2.6: Create SA

Primary actor: Identity provider (IdP)

Support actors:

- Supplicant

Stakeholders:

- Service Provider (SP)
- User

Precondition(s) / presumptions:

- The user is authenticated

Postcondition: The IdP possesses a Security Association (SA) proving that the user is authenticated.

Main success Scenario (basic flow):

1. The IdP generates a unique SA.
2. The IdP stores the SA together with timestamp.

Use Case UC3: Authorize user

Primary actor: Service Provider

Support actors:

- Client application
- Identity provider

Stakeholders:

- Supplicant
- user

Precondition(s) / presumptions:

- The user is authenticated against the Identity Provider

Postcondition(s): The user is authorized to use the requested service

Main success Scenario (basic flow):

1. The Service Provider receives a service request from the client.
2. The Service Provider extracts the Security Association from the service request.
3. The Service Provider may contact the Identity Provider to verify the claimed authenticity.
4. The service provider creates a new session and returns the session ID to the client application.
5. The user application is authorized to use the given service by including the received session ID.

3.4 Supplementary specification

The supplementary specification is the repository of all the requirements not covered directly in the use cases. It also covers non-functional requirements like FURPS+ requirements (functionality, usability, reliability, performance and supportability), physical environment concerns and other development constraints.

3.4.1 Functionality

Security

The SIM authentication system must fulfill the technical security requirements in level 4, which provides the highest practical remote network authentication assurance according to [2].

The communication channel between the client application and service provider must be secure.

3.4.2 Usability

The SIM authentication system must be easy to use if we don't want to exclude any users. The intended users are likely familiar with the use of mobile phones and remembering the PIN-code is a prerequisite. The problem of many authentication systems is the usage of many different authentication tokens and they have to remember multiple username/password combinations. The SIM authentication system must therefore minimize the number of authentication tokens, without compromising the security. The user must:

1. Remember the PIN code and keep it secret
2. Keep the SIM card personal

The user interface of the client application has to be intuitive and easy to use. It should be easy to navigate, find and use the relevant service. The SIM Supplicant will take care of the actual authentication mechanisms and it should never appear directly to the user. The SIM Supplicant should run as a process in the background of the client application.

3.4.3 Reliability

If the system fails it must support recovery and correct accounting (if accounting is involved). The system must ensure that all transaction sensitive state and events can be recovered from any step of the scenario. In case of failure the user must be notified.

3.4.4 Performance

Supplicant

The SIM supplicant should not cause noticeable delay. I.e. the Supplicant should not delay the overall system performance.

Bandwidth

The overall system performance should not be delayed more than what is acceptable in a GSM packet based network.

Other system components

The response time from other components, e.g. the authentication server, is a potentially bottleneck. To prevent the overall system performance suffering from delays in other system components, the authentication system must be thoroughly tested with every external component.

3.4.5 Supportability

The SIM authentication system should be based on a Java technologies solution, because most mobile phones support this platform. It is also very convenient because the project group has detailed knowledge of this platform.

3.4.6 Existing components and interfaces

The SIM authentication system must be able to collaborate with the other system components through varying interfaces.

3.4.7 Extensibility

The performance of the existing GSM network is a potentially bottleneck in the SIM

authentication system. Adapting to the 3GPP UMTS network is a possible improvement of the system. Due to delimitation of this assignment the UMTS was excluded.

3.5 Domain model

This section describes the superior domain model which involves an identification of the concept, attributes and associations between the objects. The objects are not necessarily software objects, but a visualization of concepts in the real-world domain.

The domain model in figure 15 describes the relationship and roles between the objects in a conceptual manner. Several relationships must be considered and explored further in the elaboration phase.

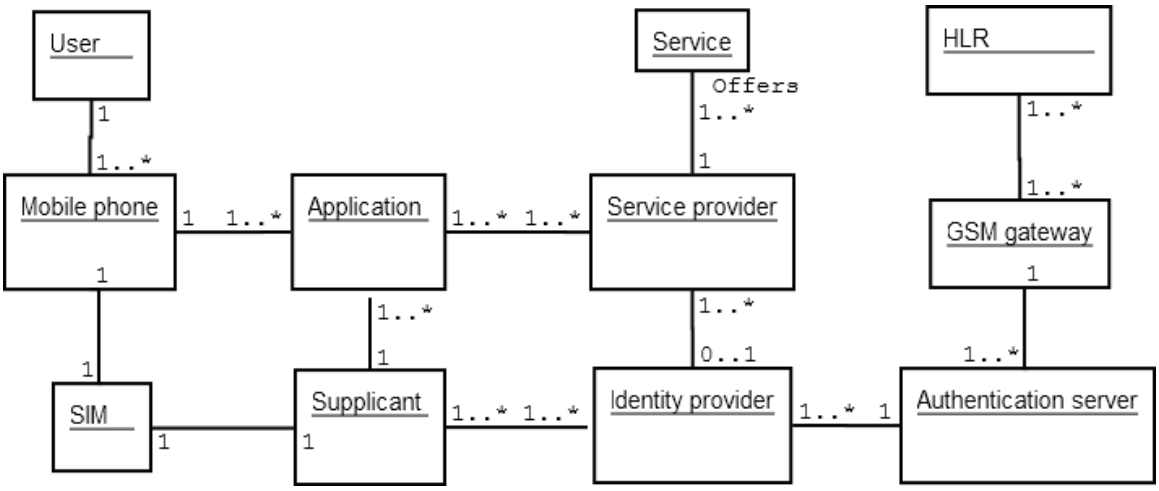


Figure 15 – UML Domain model

4 Design

In the design part of this thesis we will elaborate the components of the proposed system and look at the interface between them. The interaction diagrams are also decomposed and we will have a closer look at the communication between the different actors and components in the proposed system. The design phase is an important artifact in the elaboration phase in the UP.

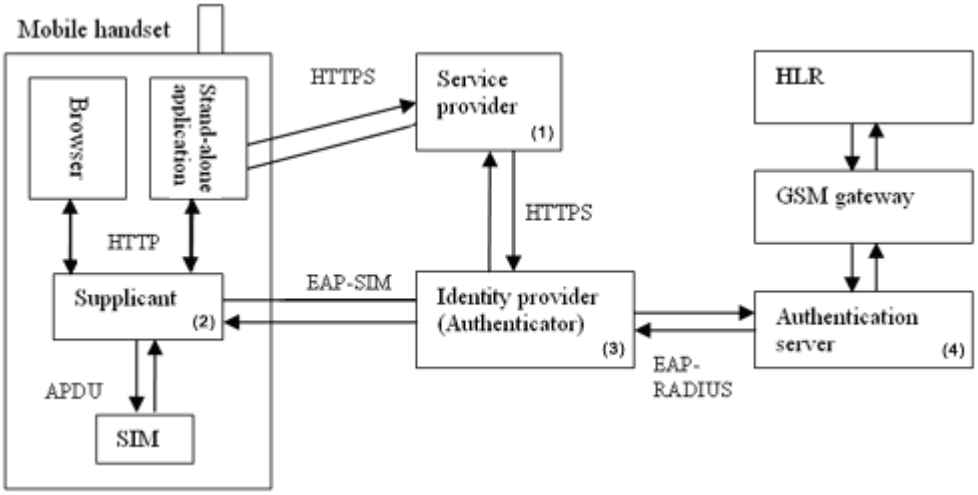


Figure 16 - Overall architecture of the proposed SIM authentication system

The proposed solution outlined in figure 16 is supposed to be a generic authentication system for mobile handsets. A generic solution implies that different kinds of client applications must be able to communicate with the Supplicant to be authenticated to the Identity Provider. The client applications of current interest are:

1. Mobile browsers
2. Stand-alone applications (i.e. a J2ME MIDlet)

In section 4.1 we will perform a detailed study of each of the components. We will highlight all the challenges and possibilities, to make a solid foundation for the realization.

4.1 Components

4.1.1 Mobile browser

There are mainly two kinds of browsers available for mobile handsets: The built-in WAP browser and a regular web browser adopted for mobile handsets. Most regular mobile phones have a lightweight browser with limited capabilities like the built-in WAP browser or a similar lightweight web browser with limited functionality like Opera Mini [41].

Other more advanced mobile handsets like Symbian Series 60-based phones and Windows mobile handsets have the opportunity to run an adequate web browser like Opera Mobile [41]. With Opera Mobile browser you can surf the same Web sites on your mobile handset as you do on your personal computer.

4.1.1.1 Server authentication

Most of the mobile browsers (at least the newest ones) support common security protocols like SSL and TLS. It does require that the browser include root certificates for commonly used certificate authorities. When a server presents a certificate or certificate chain, the device implementation must possess the root certificate so that it can verify the server's certificate.
When you enter a secure page, the browser and the web site uses public keys to agree on a secret key, which authenticates the server side. This is called a handshake. The key encrypts all the information sent and is used for this session only. Almost every major company offering secure services have an agreement with the web browser suppliers, and their server certificates are prefabricated in the mobile browsers, which means they are trusted partners. If the SP does not have such an agreement, the user must manually approve the server certificate if the web site is credible.

4.1.1.2 Client authentication

The handshake procedure described above does only authenticate the server side, i.e. the Service Provider. If the offered service involves financial transactions, credit card numbers or other sensitive information, a two-way handshake must be completed before information can be sent. The users must be authenticated through the browser to prove that they are who they claim to be. This is required to achieve mutual authentication between the parties. Unfortunately there is no support for certificate-based authentication

of the client, so the client must be authenticated at the application level by other means [42, 43].

Clients are customarily authenticated in the application layer, through the use of passwords sent over an SSL-protected channel. This pattern is common in banking, stock trading, and other secure Web applications. But providing a password is not sufficient to achieve strong user authentication. To achieve strong two-factor authentication, the user must provide something he/she: 1) knows and 2) have. The ‘knows’ might be a password combined with a PIN and the ‘have’ is a device generating OTPs. It is not very convenient to type a lot of user input on a small mobile handset. But the ‘have’ might be an external Smart Card issued by the provider. However, an external Smart Card must be read by a Smart Card reader connected to a PC, which is not very suitable from a mobility point of view.

In the proposed SIM authentication solution, the user only needs one device; a mobile handset with a GSM SIM card, to achieve strong two-factor authentication without other user interaction than typing the PIN. In the proposed solution, the ‘knows’ is the PIN associated with the GSM SIM and the ‘have’ is the SIM card itself. We utilize the fact that the GSM SIM is a tamper resistant Smart Card, which is FIPS 140-2 [12] validated. This means it fulfils the highest security requirements defined by NIST [2]. The Smart Card stores the secret key, which never leaves the card, and uses it to generate MACs that protect the authentication data.

4.1.2 Stand-alone application

Until now we have discussed how a mobile browser will cooperate with the proposed authentication system. Just as relevant is a stand-alone application e.g. a J2ME MIDlet. Most of the mobile handsets on the market today have a small Java virtual machine and is capable of running Java MIDlets. The mobile handsets are becoming more and more powerful and the screen resolution has become acceptable. This opens up for new and emerging applications.

If a stand-alone application is chosen it must be able to run simultaneously together with the Supplicant. If the stand-alone application is a native application, there should be no problems running both applications at the same time, since most mobile handsets supports multitasking. The problem arises if the stand-alone application also is a J2ME MIDlet. Even though many handsets support multitasking, only a few devices has the ability to run several J2ME MIDlets at the same time. Sony Ericsson has introduced a multitasking Virtual Machine (MVM), which enables multitasking between several MIDlets for the new Java Platform 7 (JP-7) mobile phones [44]. But for other mobile phones, and especially some of the Series 40 devices from Nokia, which fully implements the SATSA-APDU packages, this is unfortunately not possible. So a temporarily solution for stand-alone applications on devices not supporting multitasking between MIDlets, is to integrate the Supplicant within the application by running them within the same MIDlet suite.

4.1.2.1 Server authentication

The MIDP 2.0 specification requires that implementations support HTTPS. Although the MIDP 1.0 specification does not mandate HTTPS support, many MIDP 1.0 devices support HTTP over TLS and SSL. The conclusion is that server authentication works equally for browsers and stand-alone MIDP clients.

4.1.2.2 Client authentication

Because each MIDP device usually belongs to one person, user names and passwords can be stored on the device so the user doesn't have to enter them each time he or she wants to use an application. This practice significantly eases use, especially considering how hard it is to enter data on a typical MIDP device, but it entails some risk. The client can send a message digest value of the authentication secret instead of actually sending the secret over the network connection. This approach is necessary wherever the communication between client and server is not encrypted. But it does not solve the

problem with theft or misuse of the actually device.

MIDP 2.0 applications have the possibility to use client certificates to prove their identity. The client can use its private key to sign or encrypt some data and the server uses the application root certificate to verify the client's identity and uses the client's certificate to verify the signature. However, distributing the private keys securely is not trivial. It requires a huge organization to maintain and manage client certificates. If a Service Provider wants to enable services for both browsers and stand-alone applications, two different client authentication solutions are required, since mobile browsers don’t support client certificates.

The proposed SIM authentication solution is generic, which means it can be used by both browsers and stand-alone applications, since it does not require specific user certificates. The only requirement is that the mobile handset supports Java. The authentication procedure for stand-alone applications is identical to the one for mobile browsers, except for an insignificant detail. For mobile browsers the SP must redirect the authentication request to the local Supplicant because the browser doesn’t have the intelligence to do that. A standalone application can request the local Supplicant itself, when it receives an authentication request from the SP. However, it is possible to make this interface transparent, so that the authentication procedure becomes similar for both browsers and stand-alone applications.

4.1.3 Supplicant

The Supplicant is the major contribution to the proposed authentication system. It is a generic J2ME application acting as a local proxy on the mobile handset. The Supplicant provides an interface to the GSM SIM by utilizing the SATSA-APDU protocol. SATSA-APDU extends the security features for the J2ME platform, and makes it possible for the Supplicant to extract user credentials from the SIM.

The Supplicant is also implementing the EAP framework which makes it capable of exchanging EAP-SIM messages with the Identity Provider. In an authentication sequence the Supplicant retrieves GSM authentication challenges from the Identity Provider by means of EAP-SIM messages. Next the Supplicant provides the GSM authentication challenges to the SIM and retrieves its user credentials by exchanging command and response APDUs, as defined by the ISO7816 Smart Card standard.

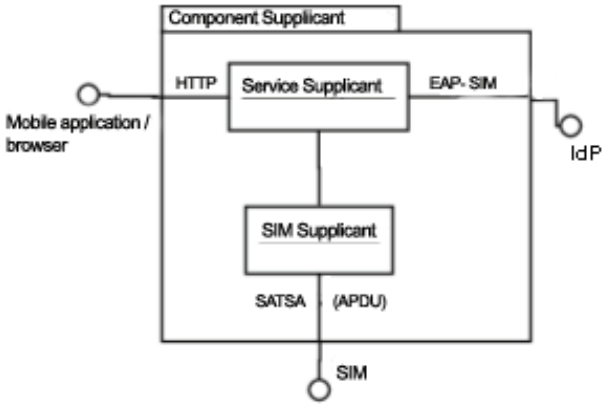


Figure 17 – Details of the Supplicant component

As depicted in figure 17 the Supplicant is composed of a Service Supplicant and a SIM Supplicant. The *Service Supplicant* is responsible for all communication with the IdP and the application requesting the service. The Service Supplicant implements the EAP framework to be able to exchange EAP-SIM messages with the IdP and it communicates directly with the SIM Supplicant. The *SIM Supplicant* implements the SATSA-APDU package, which extends the security features for the J2ME platform through the addition of user credential management, among others. It provides the GSM authentication challenges to the SIM and retrieves its user credentials by exchanging command and response APDUs, as defined by the ISO7816 Smart Card standard.

4.1.4 SIM

The SIM is an important component of the proposed system. We are not supposed to do anything with the SIM. We are only utilizing the fact that it is a tamper resistant device complying with the ISO7816 Smart Card Standard. We want to retrieve subscriber credentials from it, by exchanging APDUs. The GSM application including the A3 algorithm is the one we want to get in contact with, as depicted in figure 18. By challenging the GSM application with RAND, it will calculate a signed response and Kc by means of the A3 algorithm.

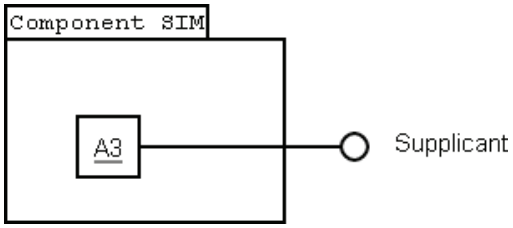


Figure 18 – Details of the SIM component

4.1.4.1 Alternative solution

Instead of communicating directly to the GSM application on the SIM card, we may change the design a bit. We do not know yet whether it is possible or not to access the SIM directly from a J2ME MIDlet.

By communicating with the GSM application through another Java Card applet, we may circumvent the system to give us access to the GSM SIM functionality including the A3 algorithm, as depicted in figure 19. This will not affect the rest of the proposed system. But this means we have to introduce another component, and this is just a backup if the proposed solution in figure 18 fails.

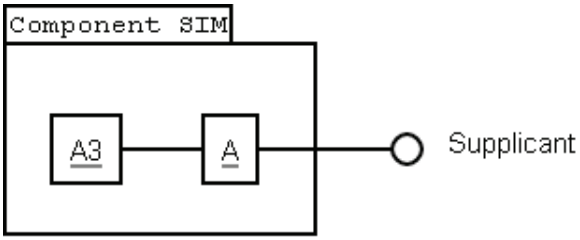


Figure 19 – Details of the SIM component (Alternative solution)

4.1.5 Service Provider (SP)

The Service Provider offers services to the users and initializes the authentication procedure. When it receives a service request from a client, it responds with an authentication request to the local Supplicant on the mobile handset, if the client is not already authenticated. If the client provides an authentication token, the SP may contact the Identity Provider to verify the claimed authenticity. If the security association is valid, the SP authorizes the client to the requested service.

As you can see in figure 20 the Service Provider communicates with the client application through standard HTTP. The (S) means that the communication channel could be protected by adding SSL/TLS on top of HTTP. For more details regarding the communication between the components, we refer to section 4.2 Interfaces.

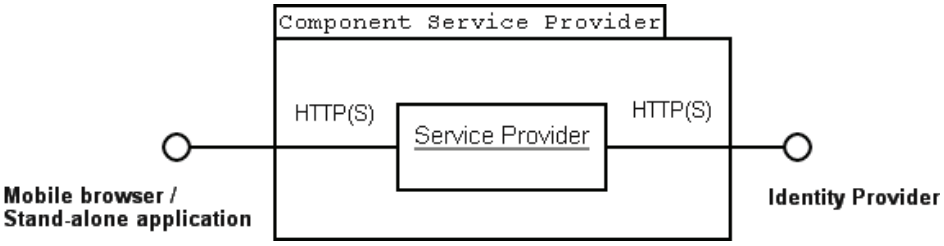


Figure 20 – Details of the Service Provider component

The Service Provider might be a part of the services offered by the operator or it might be an independent Service Provider offering different services to users. In any case the Service Provider must be able to communicate securely with an Identity Provider to exchange authentication info.

4.1.6 Identity Provider (IdP)

The Identity Provider (IdP) is responsible for locating a suitable Authentication Server and it acts as an intermediary between the Supplicant, the Authentication Server and the Service Provider. The IdP translates EAP-RADIUS messages from the Authentication Server into EAP-SIM messages and passes it to the Supplicant. It is also storing information about authorized Supplicants, so that the Service Provider can verify the Supplicant's claimed authenticity.

To get access to the Authentication Server, a RADIUS client must be implemented. A mutual trust between the IdP and the Authentication Server is required.

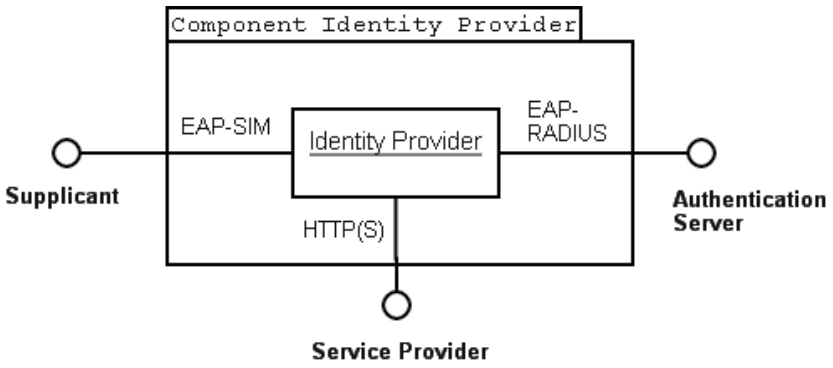


Figure 21 – Details of the Identity Provider component

4.1.7 Authentication Server

The Authentication Server is performing the actual user authentication against the GSM network. It could be realized as a RADIUS server, which is the de-facto standard for remote authentication, but other Authentication Servers like DIAMETER may also be used. To perform the GSM SIM authentication, the RADIUS server will use the GSM gateway interface to contact the HLR of the subscriber.

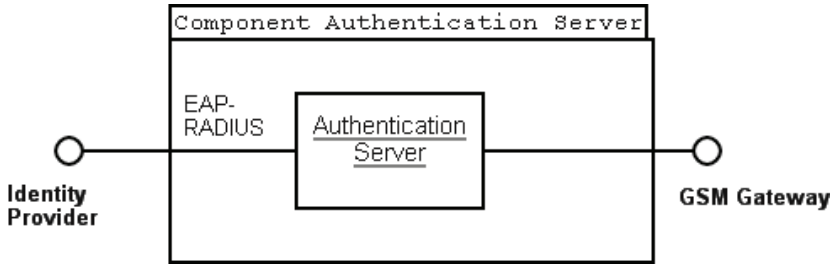


Figure 22 – Details of the Authentication Server component

The Authentication Server is provided by the operator and is not our domain. The interface between the Authentication Server and the GSM Gateway is therefore not specified further.

4.2 Interfaces

In this section the interface between the different components is specified. We have focused on the interface between the new components, i.e. the components that is our contribution to the proposed authentication system. The most important interfaces are the one between the Service Provider and browser/stand-alone application and the interface between the Supplicant and the Identity Provider. The other interfaces are also described, but we will only refer to the existing documentation.

4.2.1 Mobile handset – Service Provider interface

When a user wants access to a service offered by the Service Provider, he/she must activate a client application on the mobile handset. The client application might be either a mobile browser or a stand-alone application. In any case the client application must contact the Service Provider on the behalf of the user. If the client application is a mobile browser, the user must click on a stored bookmark. If the client application is a stand-alone application like a J2ME MIDlet, the application will take care of the request to the Service Provider.

Our challenge is to provide a transparent interface which makes the Service Provider capable of communicating with any kind of client applications. Therefore we have developed a generic interface based on the XML-standard.

When the client contacts the Service Provider, it must be redirected to the local Supplicant if it is not authenticated, as depicted in figure 23. The RedirectAuthRequest sent from the Service Provider includes the AuthenticationRequest defined in the XML Schema in figure 24.

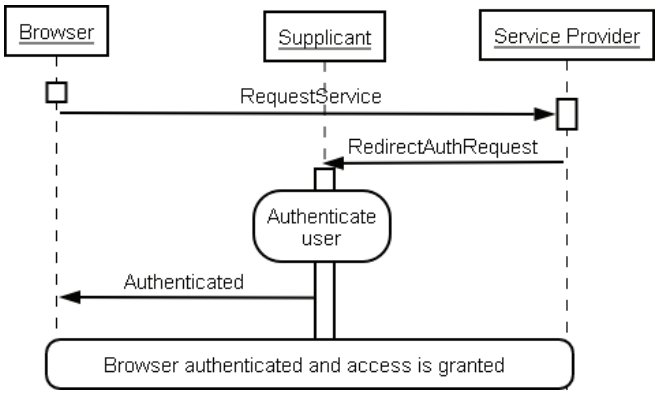


Figure 23 - Service request from browser

```
<?xml version="1.0"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <xsd:element name="Supplicant">
    <xsd:complexType>
      <xsd:choice>
        <xsd:element name="AuthenticationRequest">
          <xsd:complexType></xsd:complexType>
        </xsd:element>
        <xsd:element name="AuthenticationResponse">
          <xsd:complexType>
            <xsd:choice>
              <xsd:element name="Authenticated" type="xsd:boolean"/>
              <xsd:element name="SecurityAssociation" type="xsd:integer"/>
              <xsd:element name="ErrorDescription" type="xsd:string"/>
            </xsd:choice>
          </xsd:complexType>
        </xsd:element>
      </xsd:choice>
    </xsd:complexType>
  </xsd:element>
</xsd:schema>
```

Figure 24 - XML Schema defining the Supplicant interface

When the local Supplicant receives an *AuthenticationRequest* on the format described in figure 24, it will begin the authentication procedure. When the user is authenticated and the Supplicant has received the security association which proves the authenticity, the Supplicant will send an *AuthenticationResponse* to the Service Provider, containing the necessarily information.

4.2.2 Supplicant – SIM interface

Another crucial interface is the one between the local Supplicant and the GSM SIM card. To be able to communicate with the GSM SIM through the mobile handset it is required that the handset provides a SIM access interface. The SATSA-APDU package, described in section 2.3, provides such an interface.
By sending command APDU messages to the SIM according to the ISO7816 Smart Card Standard, the Supplicant should be able to extract the user credentials from the SIM and authenticate the user against the Identity Provider.

It is important to notice that to be able to communicate with the GSM SIM on a real mobile handset; the J2ME MIDlet must be signed by a certificate issued by the operator.
If the MIDlet is not signed, a security exception is thrown.

For more details regarding the SATSA-APDU API, we refer to SATSA Developer's Guide [30].

4.2.3 Supplicant – Identity Provider interface

When the local Supplicant receives an authentication request it will establish a connection to the Identity Provider. The Identity Provider is responsible for the exchange of necessarily user credentials between the Supplicant and the Authentication Server. To achieve mutual authentication the EAP-SIM protocol [24] must be used.

Extensible Authentication Protocol (EAP) [25] is a framework supporting multiple authentication methods. The Identity Provider implements EAP-SIM to communicate with the Supplicant. The EAP specification only discusses usage within a point-to-point protocol (PPP), which is a low layer protocol. The encapsulation of EAP messages in the higher layer protocols is not specified in the specifications. EAP is only dealing with authentication at the Application level. Hence, there is a need to secure the communication channels between the components to ensure integrity and confidentiality. To solve this problem, EAP over TCP/IP may be chosen with SSL/TLS to maintain the integrity and confidentiality between the different components.

4.3 Class diagrams

In this section we will map the components and objects into class diagrams and UML diagrams. A complete class diagram of the Supplicant is big and difficult to follow. We will not explain each of the classes in details, but give you an overview of the package structure and the main principles. If you want to look at the complete class diagrams, please have a look at appendix C.

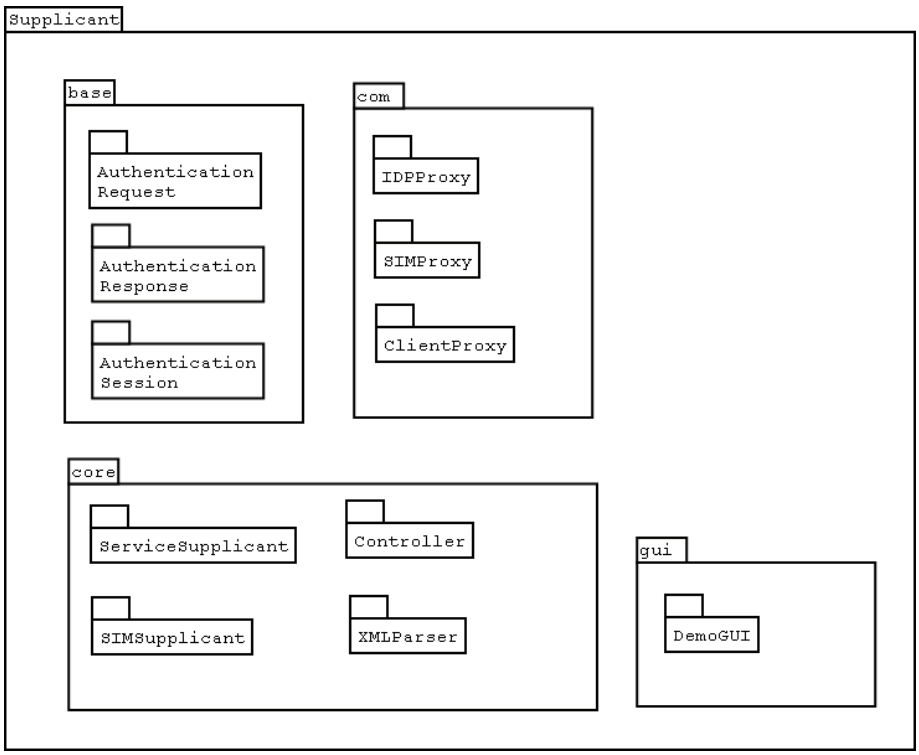


Figure 25 – Package diagram of the proposed SIM authentication system

The most of the business logic is in the core package in figure 25. The "Model-view-controller principles" is adopted to get a clean architecture that is easy to develop further. The ServiceSupplicant is responsible for the communication with both client and Identity Provider. SIMSupplicant is dealing with the SIM communication. Both classes use the Controller class to manage the communication with the different components. All I/O handling is taken care of by the proxy classes in the com package. The Supplicant does not really need a Graphical User Interface (GUI). It is only supposed to lie in the background anyway. But for demonstration purposes a simple GUI may be developed, to show what is going on.

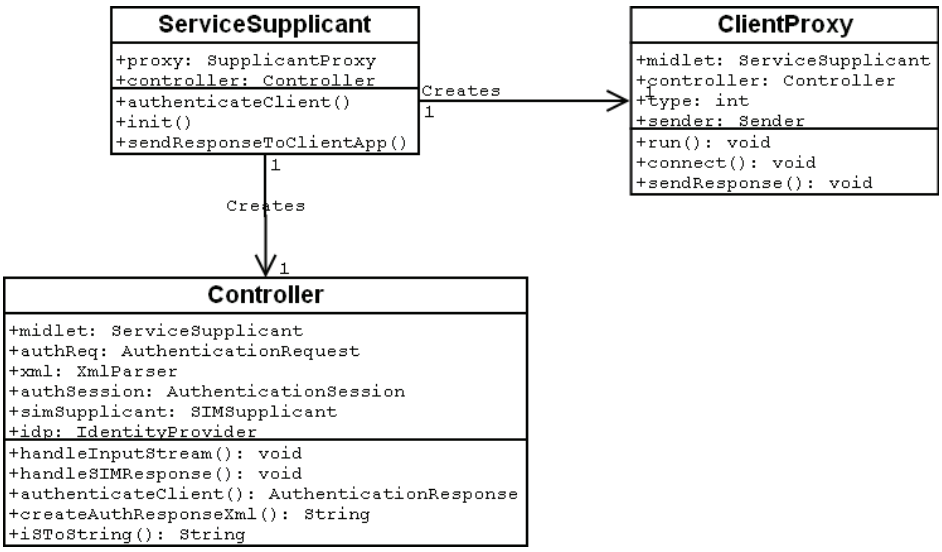


Figure 26 – ServiceSupplicant – class diagram

The ServiceSupplicant creates the Controller instance and the ClientProxy instance as depicted in figure 26. The ClientProxy is setting up a ServerSocketConnection to localhost, listening to port 4035. When an incoming request is registered on port 4035 the ClientProxy notifies the controller, which handles the incoming input stream.

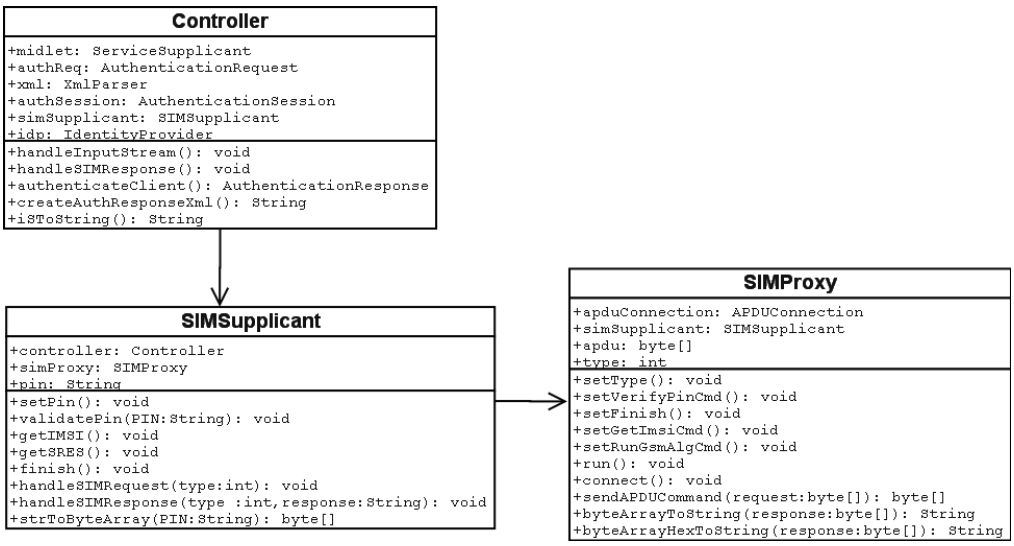


Figure 27 – SIMSupplicant – class diagram

As you can see in figure 27, it is the Controller initializing the SIMSupplicant, which is responsible for all communication with the SIM card. The actual SIM communication is taken care of by the SIMProxy class. When the SIMProxy has got an answer from the SIM, the thread will stop and the SIMSupplicant is called to handle the SIM response.

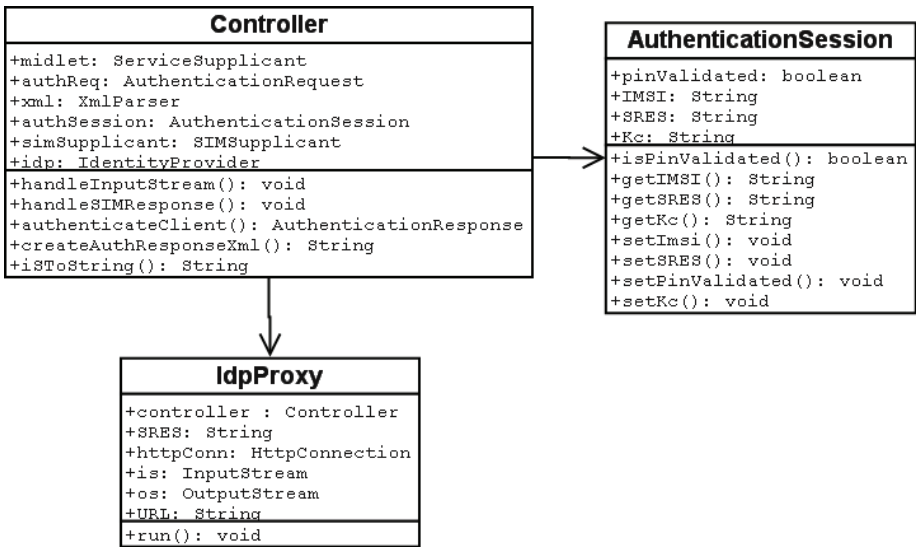


Figure 28 – SIMSupplicant – class diagram

The Controller creates an AuthenticationSession object in figure 28 that stores all the temporary data being exchanged. The IdpProxy is dealing with the communication against the Identity Provider. The Identity Provider might have different interfaces, dependent on the chosen authentication type. By introducing the IdpProxy class, we can easily adapt to other IdP types by changing the request in the IdpProxy.

4.4 Sequence diagrams

A sequence diagram in UML is used to illustrate the interactions between the actors and the operations initiated by them [35]. The main success scenario of the use case is depicted in figure 29 to 31.

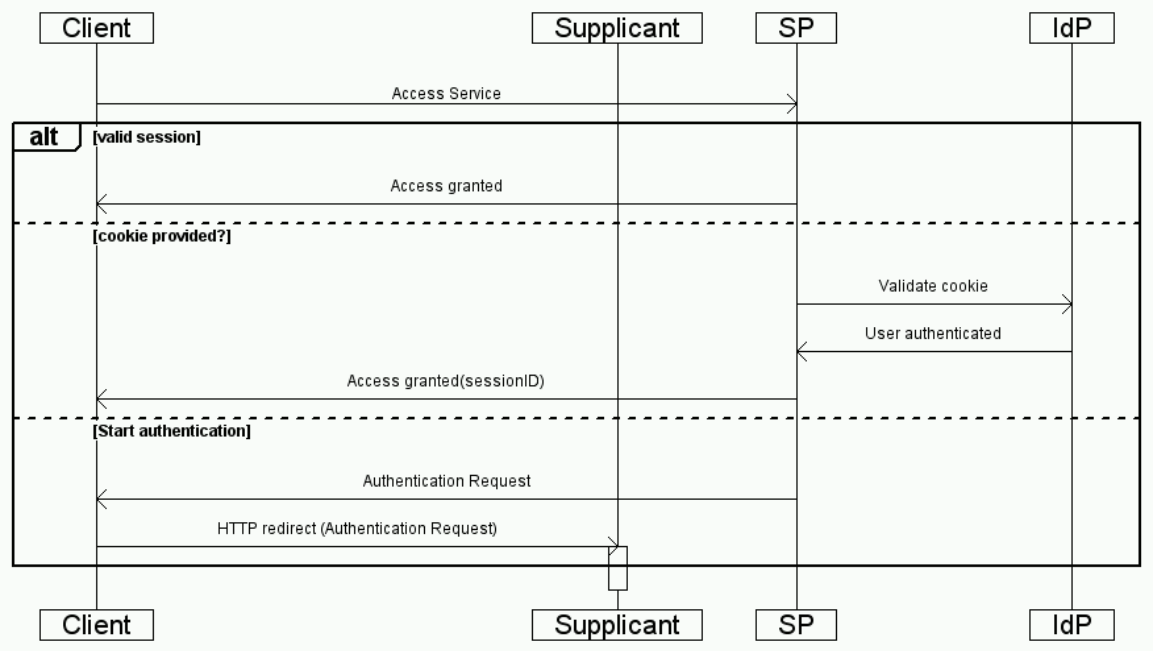


Figure 29 – Sequence diagram – access service

In figure 29 the initial actions are shown. The client will try to access a service, and the Supplicant will check whether the client provides a valid session id or a valid cookie. If the client provides a session id, it has already been authorizes by the SP and can immediately start using the service. If a cookie is provided, the client has been authenticated by the IdP and the SP can verify the claimed authenticity if desirable. If the client neither provides a session id or a cookie, the SP responds with an AuthenticationRequest, which is redirected to the local Supplicant.

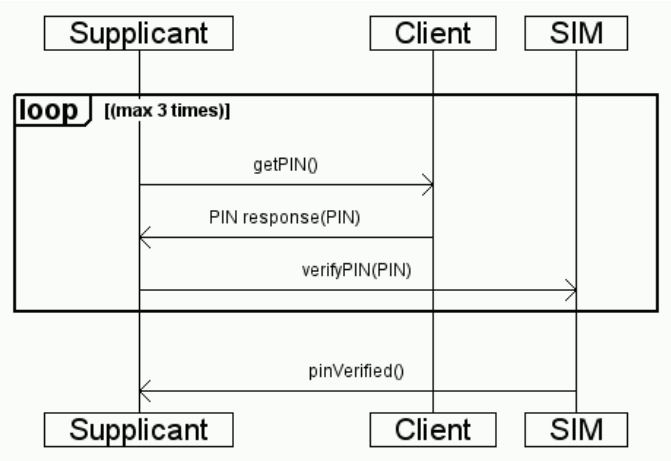


Figure 30 – Sequence diagram – verify PIN

When the Supplicant is activated and has received an AuthenticationRequest, it will start the user authentication by requesting for the PIN (figure 30). If the user types a wrong PIN 3 times, access is denied and the SIM is blocked. If the user types the correct PIN, the Supplicant will continue the authentication procedure by contacting the IdP

The IdP will respond with several EAP-requests to get the IMSI and the preferred EAP version number to use, as showed in figure 31. Finally, the IdP will challenge the Supplicant with a RAND. The Supplicant provides the RAND to the SIM and gets the SRES back, covered by a MAC. If the SRES provided in the EAP-Response/SIM/Challenge is verified by the IdP, an EAP-Success message is sent to the Supplicant and the client is authenticated. Together with the EAP-Success message sent from the IdP, a cookie is attached containing a security association that proves the authenticity of the client.

To get access to the given service, the client will repeat the steps in figure 29, but this time a valid cookie is provided and the SP can create a session between the entities.

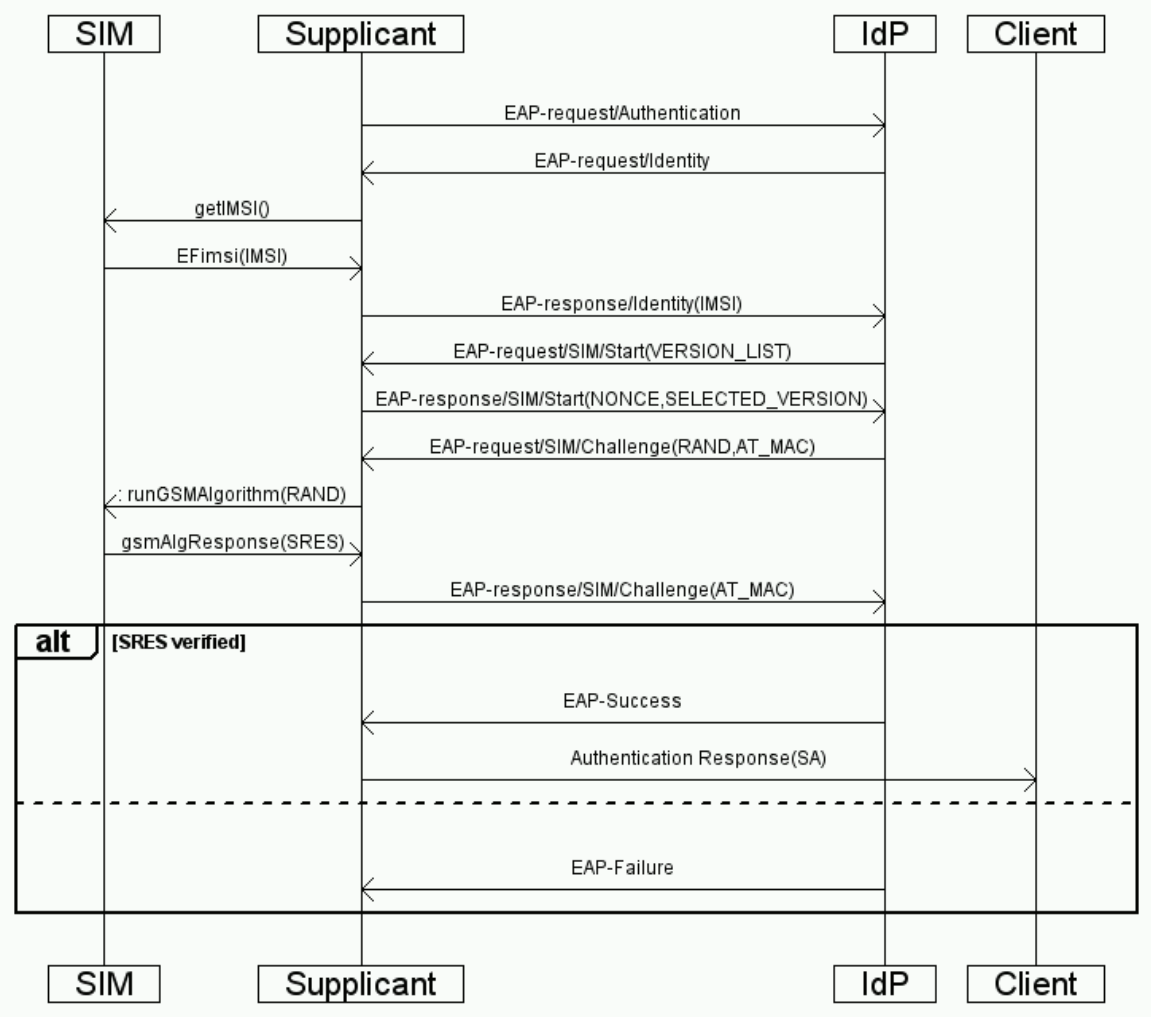


Figure 31 – Sequence diagram – authenticate user

5 Realization of the proposed system

This chapter describes the steps performed during the implementation of the proposed authentication system. There were a lot of challenges in the development phase; First of all, to be able to communicate with the GSM SIM, the application has to be signed by the operator, in our case Telenor. This is not really a technical problem, but a problem for verifying some of the contributions of the thesis. In a commercial setting, this would not have been a challenge. In the mean time, while waiting for a signed certificate from Telenor, we developed a Java Applet simulating the GSM SIM card by the means of Java Card Development Kit (JCDK) [45].

Second, some issues regarding the communication between native applications on the mobile handset and the Supplicant had to be solved. A lot of research has been done regarding the communication between the Supplicant and the client application, but some things just have to be tested in real life to discover whether it is possible or not. Even though the specification says so, a practical experiment must be carried out to prove it.

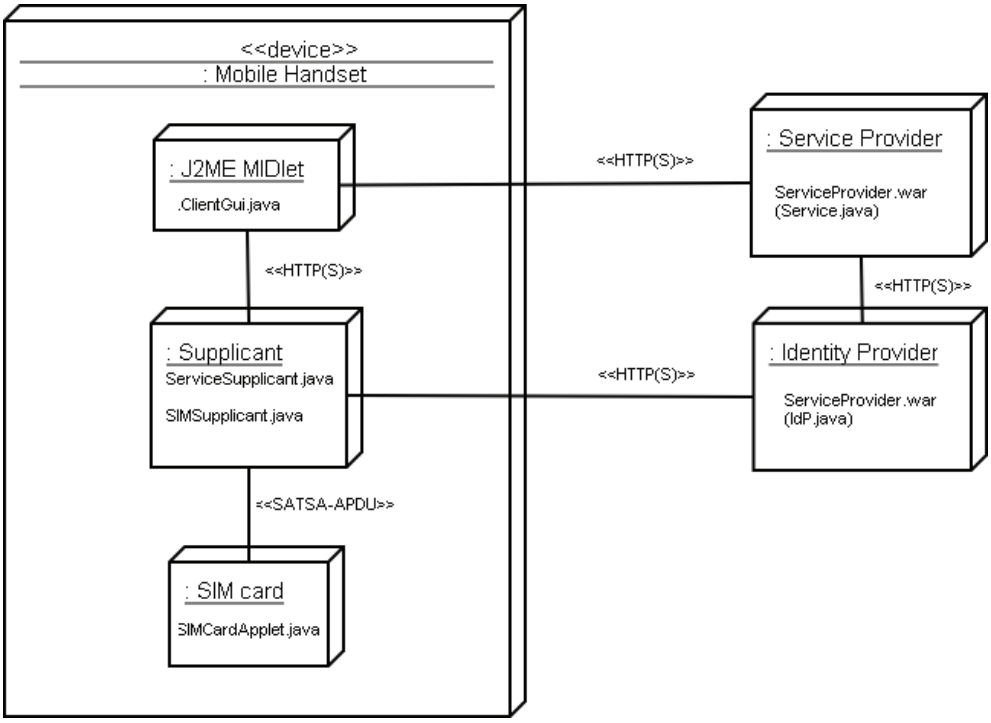


Figure 32 – UML Deployment diagram

5.1 SIM card simulator

The JCDK provides a secure environment for applications that run on smart cards. The JCDK provides a Java Card C Reference Implementation (CRef), which is a simulator that runs Java applets as if they were installed on a real SIM card. By running the mobile Supplicant through the Sun Wireless Toolkit emulator on a PC, we were able to simulate communication between the mobile handset and the SIM card.

A SIM card applet acting almost like a real SIM card was developed, to be able to test the SATSA-APDU communication. In the prototype of the SIM Supplicant we were able to verify the PIN code entered by the user, return "IMSI" by inquiries and compute "SRES", by simulating the GSM algorithm. All communication between the SIM card applet and the SIM Supplicant takes place through command and response APDUs, which is a low level byte oriented communication protocol.

As an example we can use the VERIFY_CHV command. When the user enters the PIN code to prove that he/she is who he/she claims to be, this is what is going on behind the scenes:

```
byte[] CMD_VERIFY_CHV = {
                        (byte)0xA0,(byte)0x20,
                        (byte)0x00,(byte)0x01,(byte)0x04,
                        (byte)0x00,(byte)0x01,(byte)0x02,(byte)0x03,
                        (byte)0x7F};
```

By sending this APDU command to the SIM card applet, the process method of the applet examines the received APDU header (the first 5 bytes) and calls the appropriate method, in this case verifyCHV(). If the received PIN code 1234 (byte 5-8) is valid, the applet responds with "90 00", which indicates normal processing according to ISO7816.
For more details regarding the APDU protocol and the code that constitutes the SIM card applet, we refer to appendix A and F.

5.2 Supplicant

The Supplicant is realized as a J2ME MIDlet, running simultaneously with the client application. When the client application has requested the Service Provider for a given service, a redirect is sent back to the handset's Application Management System (AMS) which fires up the Supplicant MIDlet via the J2ME push registry [47]. To show what is going on in the background, a GUI for the Supplicant has been made as you can see on the screenshot in figure 33. However, the GUI is not necessary because the user should not be aware of the Supplicant at all. It should put itself in the background and take care of the authentication procedure.

When the Service Supplicant is started, it initiates the SupplicantProxy class that creates a ServerSocketConnection listening to localhost on port 4035. When the client application sends an AuthenticationRequest on the localhost, the connection is established and the Supplicant will perform the authentication procedure.

When the ServiceSupplicant receives an AuthenticationRequest from the client, the PIN is extracted and sent to the SIMSupplicant. The SIMSupplicant will create a connection to the Java card Kit SIM card applet, and verify the SIM. This opens up the other functions on the SIM like getting the IMSI and running the GSM algorithm.

For more information regarding the Supplicant we refer to appendix C for complete class diagrams and appendix F for the source code. The SIM communication is discussed further in 5.2.1.

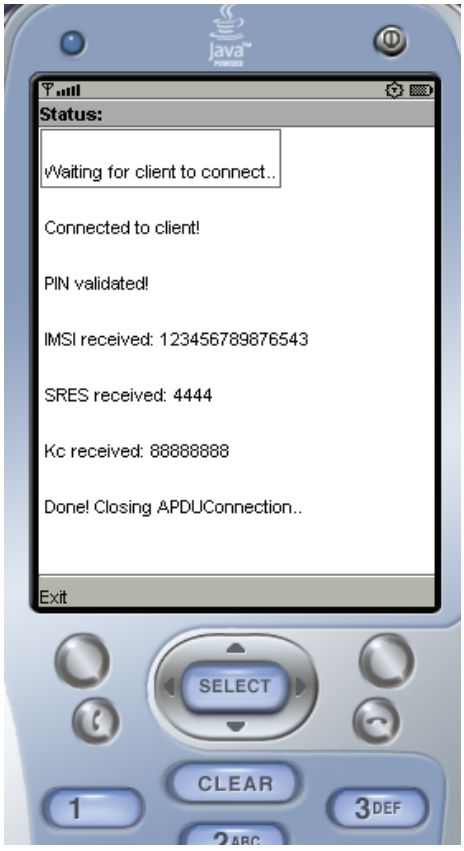


Figure 33 – Supplicant screenshot

5.2.1 SIM communication

The SIM communication is an important part of the author's contribution in the proposed system. The SIM Supplicant is taking care of the SIM communication. The SIM Supplicant is a part of the Supplicant, but it has only one fixed task; setting up a connection to the SIM. The SIM Supplicant consists of two classes: SIMSupplicant.java and SIMProxy.java. SIMSupplicant.java controls the SIM communication and handles the requests and responses. SIMProxy.java is the Thread class actually setting up the connection to SIM.

As mentioned before it is the SATSA-APDU package which makes it possible for us to communicate directly with the SIM card. Even though we do not use a real SIM card in the realized prototype, the Java Card applet simulating the SIM will act almost like a real SIM. By using the APDUConnection class provided by the SATSA-APDU, we can set up a connection to the SIM as follows:

```
String AID = "apdu:0;target=a0.00.00.00.62.03.01.0c.05.01";
apduConnection = (APDUConnection)Connector.open(AID);
```

The AID (Application Identifier) is used to reach the correct application when setting up the connection to the SIM. The SIM can have several applications in addition to the GSM application, and therefore a system is required to separate them from each other. The structure of the GSM AID is specified in [53]. The AID consists of two parts:
The first 5 bytes is the Registered application provider Identifier (RID), which is 'A000000009' according to Annex B in [53]. The next 7-11 bytes is the Proprietary application Identifier eXtension (PIX) which is defined in Annex C in [53]. The PIX consists of, among others, the ETSI application code which is '0001' for GSM. It seems like the AID of the GSM application depends on which operator the subscriber is attached to. Digit 9-12 in the PIX is supposed to be a card issuer code, coded in BCD and right justified. But this is really not a big problem. The AID can be set dynamically in connection with the installation procedure, i.e. when the Supplicant is installed on the mobile handset.

When the connection is set up, the information exchange can begin. The SIM communication is based upon command and response APDUs as explained in section 5.1. For further details regarding the SIM communication we refer to appendix F.

5.3 Client application

A client application has been developed to demonstrate the proposed authentication system. The client application is a J2ME MIDlet simulating a simple mobile banking system. The client application initializes the authentication by requesting a service from the Service Provider (figure 34). If the request does not contain a valid session id, the client is redirected to the local Supplicant. The only user interaction required is typing the PIN, which is '0000' for this application (figure 35). When the authentication procedure is completed and the user is authenticated, a new screen becomes visible (figure 36) enabling the banking services. In this case this implies getting the account balance (figure 37).

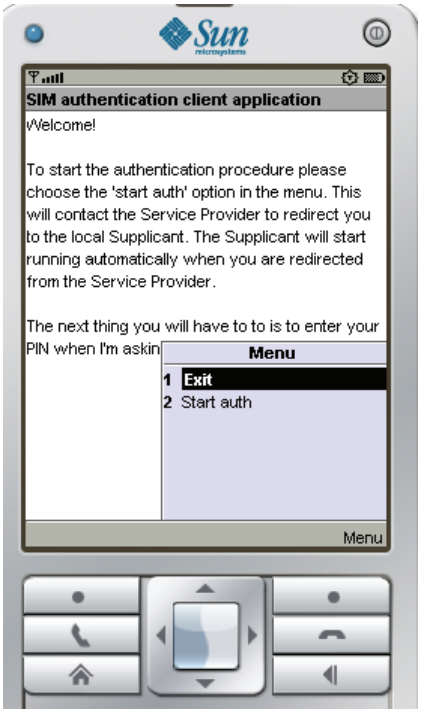


Figure 34 – Client application – Start screen

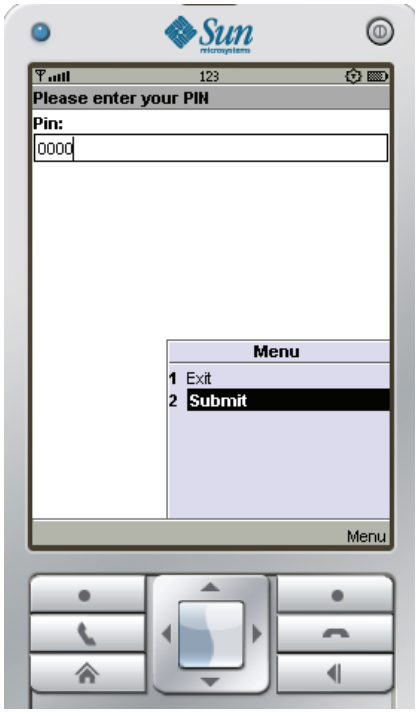


Figure 35 – Client PIN authentication

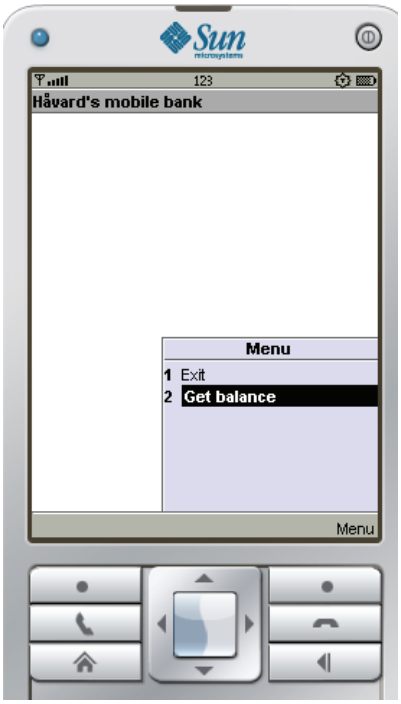


Figure 36 – The mobile banking GUI

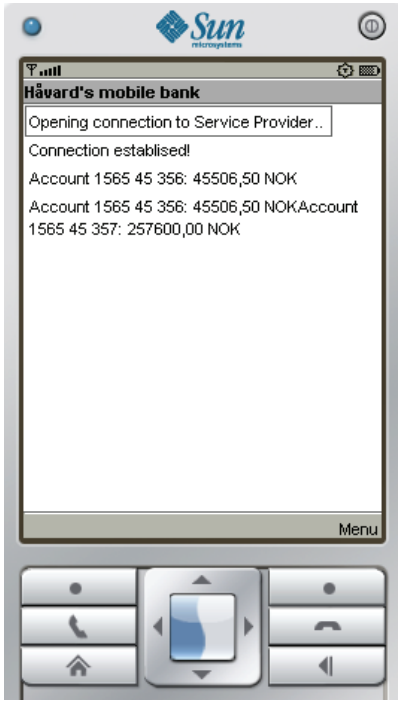


Figure 37 – Get balance results

The client application is realized as a J2ME MIDlet. It contains two different thread classes, one for communication with the local Supplicant and one for communicating with the Service Provider. For further details we refer to the class diagram in appendix C and the full source code in appendix F.

5.3.1 Browser and other stand-alone applications

The proposed system is designed as a generic authentication system, which means the Supplicant could be used in connection with different kind of client applications. I.e. it should be possible to incorporate the Supplicant with a J2ME application like the one we have developed, a browser or a native stand-alone application. In the realization we have focused on a J2ME MIDlet, because this was closest to succeed on a mobile handset. This is discussed further in section 6.1.2.

5.4 Service Provider (SP)

The SP is realized as a Java Servlet, deployed on a Gentoo Linux server running Tomcat 5.0. The SP is located at the Norwegian University of Science and Technology.

http://dellgeopos.item.ntnu.no:8080/ServiceProvider/servlet/Service

By running the SP at a different location than the client application, we have demonstrated that the SP can be anywhere and that the network communication is working well.

The SP is processing incoming requests by the following model:

1. If the incoming request has a valid session id issued by me, give access to the requested service.
2. If not, does it provide a cookie issued by the Identity Provider? If this is the case the SP should ideally verify the claimed authentication against the IDP, but this is left out of our implementation due to time constraints.

For more details regarding the deployed SP please have a look at appendix F.

5.5 Identity Provider (IDP)

The IDP is also realized as a Java Servlet and it is deployed on the same server as the SP:

http://dellgeopos.item.ntnu.no:8080/ServiceProvider/servlet/IDP

The IDP processes the incoming requests by fetching the SRES and performing a "user lookup". The user lookup process should have initiated a radius client performing the actually GSM authentication against the GSM MAP gateway. But this is left out in our implementation due to several reasons. (We will discuss this further in chapter 6). The IdP is simulating the authentication process and is always creating and returning a new cookie to the Supplicant, if the SRES is provided in the request header.

For more details regarding the deployed IdP please have a look at appendix F.

5.6 Interaction diagram

A simple interaction diagram is provided to summarize the realization of the proposed system. Figure 38 shows the most important interactions between the different components.

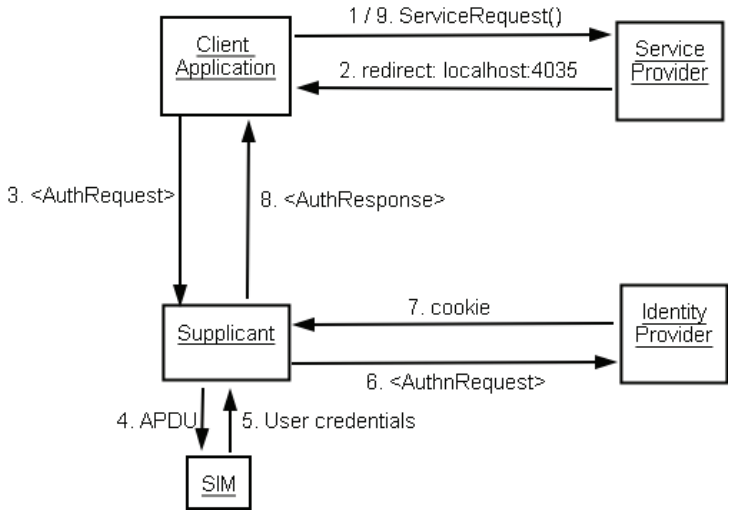


Figure 38 – Interaction diagram of the realized authentication system.

1. When the client application is started, it will automatically contact the Service Provider to request the given service.
2. Since the user is not authenticated yet, the Service Provider will respond with a redirect to the local Supplicant.
3. The redirect from the Service Provider will activate the local Supplicant via the J2ME push registry and send an AuthRequest message. The user has to enter the PIN, which is sent together with the AuthRequest.
4. The Supplicant will verify the PIN and request the necessarily user credentials from the SIM by means of SATSA-APDU messages.
5. The SIM responds with the user credentials
6. The Supplicant requests the Identity Provider to authenticate the client.
7. The Identity Provider authenticates the client and responds with a cookie, which proves the authenticity.
8. The Supplicant responds to the client with an AuthResponse containing the Security Association (provided in the cookie received in 7).

9. The client makes a new service request to the Service Provider, this time with a cookie. When the Service Provider discovers the cookie, it will verify the claimed authenticity and create a fresh session id to identity the newly authenticated user. The Service Provider sends the newly created session id back to the client application. The client application has to store this session id and provide it in all requests within this user session.

6 Discussion

This chapter will discuss the proposed mobile SIM authentication system. Strengths, weaknesses and security are highlighted.

The motivation for this thesis was to propose and implement a generic authentication system for mobile handsets. The author has spent a lot of time trying to solve the technical challenges involved in this thesis. The result of this research is discussed in section 6.1. The different security considerations of the proposed authentication system are discussed in section 6.2.

6.1 Technical barriers

6.1.1 SIM communication

To be able to run the Supplicant MIDlet on a real mobile handset, at least three requirements must be fulfilled:

1. The handset must support SATSA-APDU
2. The MIDlet must be signed by the operator
3. Establish communication with SIM application
   - Directly via AID
   - Alternatively through a proxy (Java Card applet) on the SIM card.

1) The first requirement reduces the number of handsets we can use to perform the tests on. All Nokia Series 40 3rd Edition Feature Pack 2 devices implement the SATSA-APDU package. (See appendix D for full list). However, the Mobile Services Architecture (MSA) includes SATSA-APDU as a mandatory package for the next generation mobile devices. I.e. every future mobile handset will most likely incorporate the JSR177 SATSA-APDU API.

2) The second requirement is unfortunately harder to accomplish, at least within the period of this thesis. The J2ME security architecture defines a set of permissions, usually named by the network protocol. I.e. `javax.microedition.io.Connector.socket`. This is a permission required to open a socket connection to another application. If the permission is not provided in a user certificate, the user may be asked if he/she wants to

give the application access to create a socket connection to the given application. It is not very convenient to ask the user for permissions every time the application needs to use a new network protocol. This is why the protection domain based on cryptographic signatures and certificates are introduced. The MIDlet acquires permissions through a *protection domain*. If the MIDlet is signed by a certificate issued by a trusted third party (Thawte/VeriSign) and the certificate is associated with the correct security domain, access is granted. If not, the user is either asked to give permission manually or access is denied.

In our case this leads to a big problem. To be able to install and run the MIDlet utilizing the SATSA-APDU package, the MIDlet must be signed by a certificate issued by the operator. It is reasonable though, that access to SIM functionality is restricted and hard to accomplish. Otherwise, a lot of malicious applications could have been developed by people with less honest intentions.

The Supplicant MIDlet needs the `javax.microedition.apdu.aid`-permission to be able to utilize the SATSA-APDU API. In fact, it is not possible to install the application on a mobile handset not supporting SATSA-APDU. This is why the certificate is that important for us. Without the proper certificate we cannot perform the tests we want to, on a real mobile handset.

Telenor, which is both the employer and the operator, promised they would obtain the proper certificate for us within the first months of this thesis. But time has gone and unfortunately we have not got the certificate. Therefore we had to change the plans a bit. Instead of implementing the entire proposed authentication system, we focused on the Supplicant and the communication between the client application, Service Provider and the Supplicant. We had a few challenges here as well.

Since we could not communicate with a real SIM card on the mobile handset, we had to simulate the SIM by means of a Java applet. The Java Card Development Kit [45] provides a tool called *cref* that simulates a Java Card in a card reader and enables card state saving in an EEPROM image between subsequent runs. By running the SIM applet in the cref emulator we can simulate the communication between the SIM Supplicant and the SIM by exchanging APDUs. In our prototype we can do almost the same as with a regular SIM card except for actually running the GSM algorithm with RAND as

argument.

3) The last requirement to be able to communicate with the GSM SIM is to establish communication with SIM application. The GSM SIM is a Smart Card containing a small application that constitutes the SIM functionality. We have to communicate with this application to be able to exchange the user credentials in the proposed system.

We believe this is solved, but since we can not fulfil the second requirement within the period of this thesis, we can not verify this contribution yet. However, we have another possibility if it turns out that we can not set up communication directly to the SIM application by using the GSM AID. As illustrated in section 4.1.4.1 it is possible to deploy a second application on the GSM SIM, which is acting as a proxy for our system. By communicating with the SIM application through another Java Card applet, we may circumvent the system to give us access to the SIM functionality.

6.1.2 Generic system

The goal of the proposed system is to be generic. This has been an important part of the design and it is one of the great advantages of this authentication system. A lot of tests have been carried out to be able to offer a generic interface to the Supplicant. The main problem was how to get the Supplicant to talk to mobile browsers / WAP-clients and stand-alone applications. The idea was to implement the local Supplicant as an HTTP proxy on the mobile handset and let it be in charge of all the communication between the different actors in the proposed authentication system. When the client has contacted the Service Provider for the first time (i.e. when it is not authenticated) it is redirected to the local Supplicant, which initiates the authentication procedure. These tests where performed successfully on the PC.

We adopted the J2ME push registry [47] to simplify the usage of the proposed system. The push registry enables MIDlets to set themselves up to be launched automatically, without user initiation. The push registry is part of the application management system (AMS), the software in the device that is responsible for each application's life-cycle (installation, activation, execution, and removal) [47]. By registering the Supplicant MIDlet with the push registry, the Supplicant can automatically be activated on client requests. I.e. when the mobile handsets AMS receives an incoming http request, in this case the redirect from the Service Provider, it will interpret the port number and fire up the Supplicant MIDlet without any user interaction. The user only has to open the client application, enter the PIN and the authentication procedure will proceed until the user is authenticated, with no further pains.

When similar tests were carried out on real mobile handsets (Sony Ericsson K610i and Nokia N91), some challenges showed up. When a mobile browser was used to contact the Service Provider, the redirect was sent from the Service Provider, but it never reached the Supplicant. The problem with the browser is that it can not make decisions on its own. It is just a dumb client implementing the HTTP protocol. We are not quite sure why the redirect never reaches the Supplicant. The very same tests worked properly when we simulated the mobile environment on a regular PC by means of the Sun Wireless Toolkit [49] and different WWW browsers.

Possible reasons for redirect towards Supplicant are not working with browser:

- The security model of Java (The closed architecture on the mobile handsets)
- Restrictions on ports in browser (this is not restricted in J2ME, except for ports < 1024)
- Restrictions on hostnames in browser (e.g. due to name resolution).
- Restrictions on IP-addresses in browser (e.g. does not support loopback address).

However, as mobile phones are becoming full-blown computers, the architecture and features of the phone will probably be similar to a PC in the near future. E.g. think about Linux based mobile phones, with the standard Linux network stack, name resolution etc.

To be able to proceed and get a working system, the focus was changed to a stand-alone application instead. By developing the client application as a J2ME MIDlet, we overcome the problems with the dumb mobile browser. We have totally control of the communication and are able to redirect the authentication request to the local Supplicant ourselves by interpreting the response from the Service Provider. Unfortunately we ran into more obstacles:

- Multitasking between J2ME MIDlets
- Lack of push registry socket/datagram support

The Supplicant is a J2ME MIDlet running in the background. The most recent mobile phones allow the developers to minimize the MIDlets or let them run in the background. The problem is that it is only a few devices allowing more than one J2ME MIDlet to run simultaneously. Many mobile phones allow more than one application to run at the same time, but that is only true if the other application is a native one. I.e. only one J2ME MIDlet can run at the same time. This is an obstacle right now, but within a short time this will probably become a “none-topic”, due to the technological revolution.

The other obstacle is the vague specifications regarding the Push Registry. The MIDP 2.0 specification does not mandate which inbound connection types must be available for the Push Registry. That decision is left up to the platform vendors. This is the very same problem as for the SATSA package. The vendors seem to choose the easiest way out due to the Time-to-market pressure in this competition exposed market. Unfortunately it is the users and especially the developers who suffer the most. The outcome of the vague Push

Registry specifications is that most of the mobile handsets do not support socket and datagram connections. But the Wireless Toolkit simulation environment on the PC supports all kinds of inbound connection types for the Push Registry.

Unfortunately none of the handsets that support multitasking between J2ME MIDlets is also supporting Push Registry with socket/datagram connections. Therefore we had to run the Supplicant and the client application on a PC and simulate the behavior in the Wireless Toolkit, due to lack of supported handsets per May 2007. The Wireless Toolkit supports all kinds of inbound connection types for the Push Registry and we successfully performed the tests on the PC environment. The Supplicant was launched automatically when the Service Provider sent the redirect.

6.2 Security considerations

6.2.1 Security tokens

The fundamental challenge in authentication systems is to find tokens that are secure (enough) and a way of securing them in a none-secure network. The most used token today is the password token, which have several weaknesses as outlined in section 2.1.2.1. The password token is not usable in a strong authentication system, because it only fulfills security level 1 and 2 [2]. The GSM SIM is a hard token which is FIPS 140-2 validated. This means it fulfills the token requirement of security level 4, which is the highest practical security level. Fulfilling security level 4 is also defined as a requirement in the supplementary specifications in chapter 3.5.1.

Security level 4 requires protection against reply attacks, eavesdropping, verifier impersonation, man-in-the-middle attacks and session hijacking, in addition to the basic requirements of level 1 and 2. The GSM AKA, which is the framework for distributing authentication and encryption keys in GSM, does not fulfill these requirements completely. Ergo, we have to use the EAP-SIM protocol to achieve this. EAP-SIM combines multiple authentication triplets to achieve greater strength than individual GSM triplets used in GSM-AKA. It also enables mutual authentication between the parties, which is very important to fulfill the requirements of security level 4. Even though UMTS AKA does support mutual authentication, GSM devices will be on the market in several years ahead, which means we have to consider every devices as the weakest link; namely GSM devices.

When combining the GSM SIM authentication functions with the EAP-SIM framework, we achieve strong two-factor mutual user authentication. The PIN must be used to activate the SIM, which means an attacker needs to have physical access to (i.e. steal) the SIM and possess the PIN to be able to exploit the system.

Since the security of the complete system relies on the weakest link, the GSM authentication procedure must be considered. In section 2.4.3 security properties and vulnerabilities of the EAP-SIM protocol are discussed, and it seems like the most critical vulnerabilities of the EAP-SIM are related to the basic GSM functions. But none of these vulnerabilities will affect the proposed authentication system particularly.

6.2.2 Securing the communication channels

The communication channels between the different components in the proposed authentication system have to be protected from eavesdroppers/intruders. Especially the communication between the client application on the mobile handset and the Service Provider is important to protect, since the exchanged information probably is sensitive.

When the authentication procedure is finished and access is granted, everyone listening to the "conversation" may be able to grab the authentication token and hijack the session. To prevent misuse, all information sent must be protected. For wireless communication with limited bandwidth, a tradeoff between security and performance must be carried out. Securing the communication channel with SSL/TLS is one possibility. The client application and the SP are using public keys to agree on a secret key, which authenticates the server side. The key encrypts all the information sent and is used for this session only. SSL/TLS may be useful in some cases, but it only secures the connection, not the content. To achieve end-to-end encryption, an external encryption package like the Bouncy Castle Crypto API [50] is needed. The Bouncy Castle is a lightweight crypto API perfect for MIDP applications.

The tradeoff is whether we want to use SSL/TLS, Bouncy Castle or both. HTTPS (SSL over HTTP) can be computation-intensive and it does not provide end-to-end encryption. Bouncy Castle is a lightweight cryptographic protocol adapted to J2ME devices and it provides end-to-end (e2e) encryption. Combining both SSL/TLS and e2e encryption would be a good solution from a security perspective. But unfortunately SSL is

computation-intensive and it also produces a lot of overhead, which is not ideally in a wireless environment. Encrypted data can be safely transmitted over an insecure network; hence using Bouncy Castle might be adequate for the proposed solution. But as for all other cryptographic solutions the biggest issue is the key management. If we only implements e2e encryption without securing the communication channels, the client cannot efficiently authenticate the server. With HTTPS the server must be authenticated before the session can start. This prevents possible man-in-the-middle attacks and other threats from the server side.

The proposed solution should therefore use SSL/TLS for server authentication / securing the communication channel and Bouncy Castle Crypto API for end-to-end encryption between the client and the SP. This should work well for both browser clients and MIDP clients. The issue is how to securely exchange the private user keys for e2e encryption.

Key exchange for e2e encryption

Every user must have its own private secret key to be able to encrypt the data. The server side must also know this secret key to be able to decrypt the data. How can the private key be transferred safely to the user?

It is possible to set up a server with a simple web interface, where users could enter names and numbers and the server would generate private keys. But this requires a lot of administration and the transfer of the key must be protected as well. If we look at our GSM SIM authentication solution, we might have a cutting edge solution for this problem. In the GSM authentication procedure a cryptographic key (Kc) is computed for every new session being made.

Kc, which is a 64-bit ciphering key, is generated with the COMP128 (A8) algorithm and is a part of the triplet generated by the AuC. The MS and the HLR both calculate the Kc independent of each other. The Kc is never transmitted over-the-air. If we can utilize this key and use it for encrypt/decrypt the data between the client application and the SP, we have an elegant solution for the key distribution problem.

To enhance the security several Kc's are combined to constitute stronger keying material. Kc should at least be 128-bit long to resist brute-force attacks. The issue is whether it is possible to utilize Kc in the proposed mobile SIM authentication system. There should be

no major problems retrieving Kc from the SIM via the SATSA-APDU on the mobile handset, if the Supplicant MIDlet is signed by the operator. The issue is how the IdP can retrieve Kc from the GSM network via the GSM MAP Gateway, and how the SP can retrieve this value from the IdP. The SP must be able to communicate with the IdP in any case, to be able to verify the claimed authenticity of the client application.

The SP might retrieve security attributes from the Authenticator using SAML, which is an XML standard for exchanging authentication and authorization data between security domains, in this case between an IdP and a SP. If the IdP is able to retrieve Kc from the GSM network, and send it to the SP in a SAML-message, the problem is solved. We can then use the Kc as the input key to the Bouncy Castle Crypto algorithm and encrypt the data between the SP and the mobile client.

If it is not possible to utilize Kc for encryption, another encryption key must be used. If we introduce a new ciphering key, a new challenge arises; how can we distribute this key securely? An alternative is to use SSL over HTTP, but this method does not provide end-to-end encryption, since the endpoint somehow must decrypt the content before it is transmitted to the target application. But if we fully trust the endpoint, this might be an adequate solution.

There exist other alternatives like Diffie-Hellman / RSA as well, which can be used as long as the session is authenticated mutually first. But using such asymmetric algorithms between low bandwidth wireless devices is not very convenient. The public keys must be authenticated before the key agreement can begin.

6.2.3 Authorization

The client application must provide some kind of a security token to the SP to get access to the requested service. When the IdP has authenticated a user, a security association is created and returned to the Supplicant. When the Supplicant receives this security association, it notifies the client application and sends the security association as a part of the AuthenticationResponse as defined in figure 24 in section 4.2.1.

When the client application receives the security association, it requests the SP once again, but this time it includes the security association in the service request. When the SP detects the security association, it may contact the IdP to verify the claimed

authenticity. But this is actually not necessarily. The only reason to do this extra verification is to prevent a possible hijack of the user session. But if the protection of the communication channels is adequate, as discussed in 6.2.1, this is actually not necessary. However, the SP has got the possibility and it might want to perform this extra check. If it chooses to do so, it will request the IdP and ask whether the provided security association is valid or not. If it is valid or if the SP takes it for granted, the SP will create a new unique session and return to the client application. The session is valid as long as the session is active or in a defined time span.

In other words, the proposed SIM authentication system is adapted to the SAML standard supported by the Liberty Alliance specification, and may easily be incorporated into a Single Sign-on system in the future.

6.2.4 Client side security

We have discussed different aspects of security levels and technical solutions regarding the authentication procedures. But there are other possible threats to consider as well. The client application and the SIM supplicant software resides on a mobile handset, which is a relatively new medium for systems development. Before J2ME and similar technology saw the light of day, there were mostly large telecom companies that developed applications for mobile handsets. During the last decade several interfaces and API's for developing applications on mobile handsets have been published.

This might lead to new security issues and threats like Spyware, Trojan horses and a variety of different viruses on the mobile handset. I.e. the client software must be built with security in mind and protect against malicious code attacks and insider threats. The data persistence in J2ME is handled through the Record Management System (RMS). The RMS only allows the owner, the MIDlet that created the Record Set, to access the content. So until further notice the content of the internal database is safe.

We might have to consider encrypting the information sent between the supplicant and the client application. A Trojan horse might eavesdrop on the exchange and snap important information.

6.3 Design choices

We chose J2ME as platform for the SIM Supplicant to reach as many users as possible, since most of the handheld devices implement a J2ME runtime environment. J2ME is an open standard and applications based on J2ME software are portable across a wide range of devices, yet leveraging each device's native capabilities.

One major challenge when developing for the J2ME platform is the closed architecture of the mobile terminal. It is difficult to get access to security functions like SIM access and other native features. But the SATSA-APDU package makes this possible, at least in the theory.

It seems like the proposed system is a little bit ahead of its time. But the reason is not the design, but the technology and the lack of supported features on today's mobile handsets. However, the new Mobile Services Architecture (MSA), which defines the next generation of the Java platform for mobile devices, will make it easier to develop applications for a broad range of devices in the future. The primary design goal of the MSA Specification is to minimize fragmentation of mobile Java environments by defining a predictable and highly interoperable application and service environment, which will be highly appreciated by developers.

6.3.1.1 Alternative realizations

It is also possible to implement the Supplicant in a native language like C++/C# .NET. The design would have been quite similar. The SIM card could have been accessed through a C++ API for Symbian OS's and the communication with the browser could have been solved differently. However, the SIM API for C++ is not a part of the Nokia SDKs for Symbian OS based phones and the implementation is more device specific, requiring close cooperation with the manufacturers. "*The Platinum Partner Program is a program for companies with a technology, service or strategic position that is key to the success of Symbian OS phones in the market"* [34].

Another alternative is to change the way we communicate with the SIM. The existing design is based on utilizing the SATSA-APDU package to communicate directly with the SIM, via the Supplicant. Actually we do not change the way we communicate with the

SIM, but we change the way we communicate with the GSM application, residing on the SIM. The basic idea is to install a new applet on the SIM, which takes care of the communication with the GSM application. The reason to mention this alternative design is because we do not know exactly if we are allowed to communicate with the GSM application through the SATSA-APDU package. Anyhow, we need to utilize the SATSA-APDU to be able to communicate with the SIM.

7 Conclusion

The main goal of this Master's thesis was to discover whether it is possible or not to use the SIM as a general-purpose authentication token in non-GSM services accessed through a mobile handset. This implies a lot of analyses, design and implementation throughout the whole project. The specifications of the interfaces and technologies used are somewhat vague, so everything had to be tested thoroughly.

7.1 Achievements and results

The proposed mobile authentication system combines the existing GSM SIM authentication mechanisms with the EAP-SIM framework to achieve two-factor mutual authentication that fulfills the requirements of the highest security level defined by NIST [2].

The prototype developed by the author is not a complete authentication system as specified in the design. The deployment diagram in chapter 5, figure 32, illustrates which components have been included in the prototype. A lot of minor and major obstacles have been encountered through this project. Some of them were known and others have showed up during the work. The most critical obstacle has been the lack of the certificate regarding the SIM communication. We were supposed to receive a signed certificate from Telenor within the first period of this thesis. Since this certificate never showed up, we had to overcome this obstacle by simulating the GSM SIM on a PC via the Java Card Kit emulator. I.e. we have not been able to run tests on a mobile handset to prove that the SIM communication actually works. The Supplicant receives a lot of security exceptions when we deploy it on a real mobile handset and set up an APDU connection against the SIM. But that is excepted since we do not have the proper certificate.

The Supplicant described in section 5.2 is simulated on a PC by the means of the Wireless Toolkit provided by Sun. There are several reasons that we can not run the Supplicant on a real mobile handset. The lack of certificate is one thing. Other reasons are missing support of multitasking between J2ME MIDlets and vague specifications of the push registry in MIDP 2.0.

But in spite of the many obstacles during this thesis, the future of this project looks great. Most of the problems and obstacles are caused by missing support and vague specifications on today's mobile handsets. A lot of things happen in this business, and when the mobile architecture opens up even more, the proposed system might be possible to deploy on regular mobile phones within a year or two.

The most important is that we have proposed and designed an innovative authentication system, with a huge potential. Today's authentication systems migrated to mobile handsets is bothersome and not very convenient to use, if they offer strong authentication at all. The proposed system is secure, inexpensive and easy to use. The user only have to fire up the client application and enter the PIN to get access to banking services and other services requiring strong authentication. The push registry makes this possible by automatically initialize the Supplicant when the mobile handset receives an authentication requests from the Service Provider.

We have also submitted a full paper to the ERCIM workshop on eMobility [52]. The paper was accepted for presentation at this workshop, after careful review by the Program Committee. Unfortunately I could not be there, but my supervisor Ivar Jørstad presented the paper on the workshop in Coimbra, Portugal on May 21, 2007. The paper notification is included in appendix E and the full paper in appendix F.

7.2 Critical review

Due to several obstacles, we have not been able to implement a fully functional authentication system on a mobile handset. We could have focused more on the server side and set up a functional Identity Provider and integrated the solution with an Identity Management System like Liberty Alliance But this has been done before, among others in the SIM Strong project carried out by Telenor [6]. Instead we focused on overcoming the technical barriers. The author was convinced that the signed certificate should arrive within a few weeks to a couple of months at the most. Because the certificate never showed up, the focus was pointed to other aspects of the proposed system. The most critical part of the proposed system is the SIM communication on the mobile handset and the Supplicant interface towards the mobile browser / J2ME MIDlet. Therefore we have used most of the time doing research on these topics, instead of implementing a system already existing.

7.3 Future work

The proposed system has to be tested on a real mobile handset when the proper certificate is obtained from Telenor. In addition the technology has to be more mature and we need a mobile handset which does not exist today's date; namely a handset supporting SATSA-APDU, multitasking between several J2ME MIDlets and with push registry socket support. The prototype can also be adapted to a Single Sign-On system with minor modifications. Actually it is almost ready, since we have adapted the SAML principles in the development. All the communication and I/O handling is executed in separate threads and can easily be modified. When the obstacles discussed in this thesis have been solved, the proposed system should be integrated with an Identity Management system, to accommodate to other authentication systems and the Service Providers around.

Another possible enhancement is to utilize the GSM cipher key (Kc) for encryption purposes. The information exchange between the client application and the Service Provider must be protected by some means, as discussed in section 6.2.2. By utilizing Kc in a lightweight crypto package like Bouncy Castle [50], we can achieve end-to-end encryption without the bothersome key exchange, because Kc can be derived from the SIM.

Appendix A – Steps for running the SIM simulator

The steps (and requirements) to run the Java Card Cref simulator, which simulates the SIM card:

1. Install Java Card Kit 2.2.1. Se the installation instructions in the doc-catalogue.

2. Compile SIM.java (Appendix B) with JDK compliance level 1.3.

In the java card kit-folder/bin run the following commands (of course you have to change the relative paths):

3. Generate the CAP file:

```
converter -applet 0xa0:0x0:0x0:0x0:0x0:0x62:0x3:0x1:0xc:0x5:1
SIMCardApplet.SIM -classdir
c:\utvikling\eclipse\workspace\SIMSimulation -exportpath
c:\java_card_kit_2.2.1\api_export_files SIMCardApplet
0xa0:0x0:0x0:0x0:0x0:0x62:0x3:0x1:0xc:0x5 1.0
```

4. Generate the script file (SCR):

```
scriptgen -o SIMCardApplet.scr
c:\utvikling\eclipse\workspace\SIMSimulation\SIMCardApplet\javacar
d\SIMCardApplet.cap
```

5. Edit the SCR file. The SCR file must be edited with our applet's parameters

```
powerup; //Always start with this

// Select the installer applet
0x00 0xA4 0x04 0x00 0x09 0xa0 0x00 0x00 0x00 0x62 0x03 0x01 0x08
0x01 0x7F;

//CAP begin
0x80 0xB0 0x00 0x00 0x00 0x7F;

// All the other CAP files already existing in SIMCardApplet.scr…
```

```
//CAP end
0x80 0xBA 0x00 0x00 0x00 0x7F;

// create SIM applet
0x80 0xB8 0x00 0x00 0x0c 0x0a 0xa0 0x00 0x00 0x00 0x62 0x03 0x01
0x0c 0x05 0x01 0x00 0x7F;

// select SIM applet
0x00 0xa4 0x04 0x00 10 0xa0 0 0 0 0x62 3 1 0xc 5 1 127;
// 90 00 = SW_NO_ERROR

powerdown;
```

6. Start the cref simulator and save the results in SIMCardApplet.eeprom.

```
start cref -o SIMCardApplet.eeprom
```

7. Use the apdutool to “copy” the content on SIMCardApplet.scr onto the SIMCardApplet.eeprom:

```
apdutool SIMCardApplet.scr
```

8. We are now ready to start the simulator with SIMCardApplet.eeprom as input. The simulator will run until the APDU connection is closed.

```
start cref -e -i SIMCardApplet.eeprom
```

Appendix B – The SIM card applet (SIM.java)

```
package SIMCardApplet;

import javacard.framework.*;

/**
 * This applet simulates a GSM SIM card, with the following steps:
 * - Verify CHV (PIN) must be done to get access to other services on
PIN
 * - Get IMSI
 * - Run A3 alghoritm with a Challenge(RAND)
 *
 * Runs in the Java Card environment (Java_card_kit_2.2.1)
 * @author Håvard Holje
 * @since 12.feb.2006
 */

public class SIM extends Applet {

  //standard APDU input offset values
  public final static byte THIS_CLA = (byte)0xA0;

  //INS value for dummy response (echo)
  public final static byte INS_DUMMY = (byte)0x10;

  //INS value for verify CHV according to GSM 11.11
  public final static byte VERIFY_CHV = (byte)0x20;

  //INS value for retrieving IMSI
  public final static byte GET_IMSI = (byte)0x30;

  //INS value for RUN GSM ALGORITHM according to GSM 11.11
  public final static byte RUN_GSM_ALGORITHM = (byte)0x88;

  //signal that the PIN verification failed
  final static short SW_VERIFICATION_FAILED = 0x6300;

  //The user PIN (0000)
  private final static byte[] pinCode =
{(byte)0x00,(byte)0x00,(byte)0x00,(byte)0x00};

  //maximum number of incorrect tries before the PIN is blocked
  final static byte PIN_TRY_LIMIT =(byte)0x03;

  //maximum size PIN
  final static byte MAX_PIN_SIZE =(byte)0x04;

  //For offset computations
  final static short START = 0;

  private OwnerPIN cardPIN;
  private UserPIN userPin;
```

```
  //sendResponse
  private byte[] echoBytes;
  private static final short LENGTH_ECHO_BYTES = 256;

  /**
   * Constructor.
   * Only this class's install method can create the applet object.
   */
  public SIM(byte[] installationData, short offset, byte length) {

        cardPIN = new OwnerPIN(PIN_TRY_LIMIT, MAX_PIN_SIZE);
        //pin - the byte array containing the new PIN value
        //offset - the starting offset in the pin array
        //length - the length of the new PIN.
        cardPIN.update(pinCode,(short)0, MAX_PIN_SIZE);
        echoBytes = new byte[LENGTH_ECHO_BYTES];
        register();
  }

  /**
   * Installs this applet.
   * @param byteArray the array containing installation parameters
   * @param offset the starting offset in byteArray
   * @param length the length in bytes of the parameter data in
byteArray
   */
  public static void install(byte[] byteArray, short offset, byte
length) {
    new SIM(byteArray, offset, length);
  }

  /**
   * Implementation  of the standard method for processing an incoming
APDU.
   * @param apdu the incoming APDU
   * @exception ISOException with ISO 7816-4 response bytes
   */
  public void process(APDU apdu) {
    byte buffer[] = apdu.getBuffer();

    //Examines the first two bytes of the APDU header, the CLA byte and
INS byte.
      //If the value of the CLA byte is 0 and the value of the INS byte
is 0xA4,
      //it indicates that this is the header of a SELECT APDU command.
      //In this case, the process method returns control to the JCRE:
      if ((buffer[ISO7816.OFFSET_CLA] == 0) &&
         (buffer[ISO7816.OFFSET_INS] == (byte)(0xA4)) ) {
        return;
      }

    if (buffer[ISO7816.OFFSET_CLA] == THIS_CLA) {

      switch (buffer[ISO7816.OFFSET_INS]) {
            case VERIFY_CHV:
```

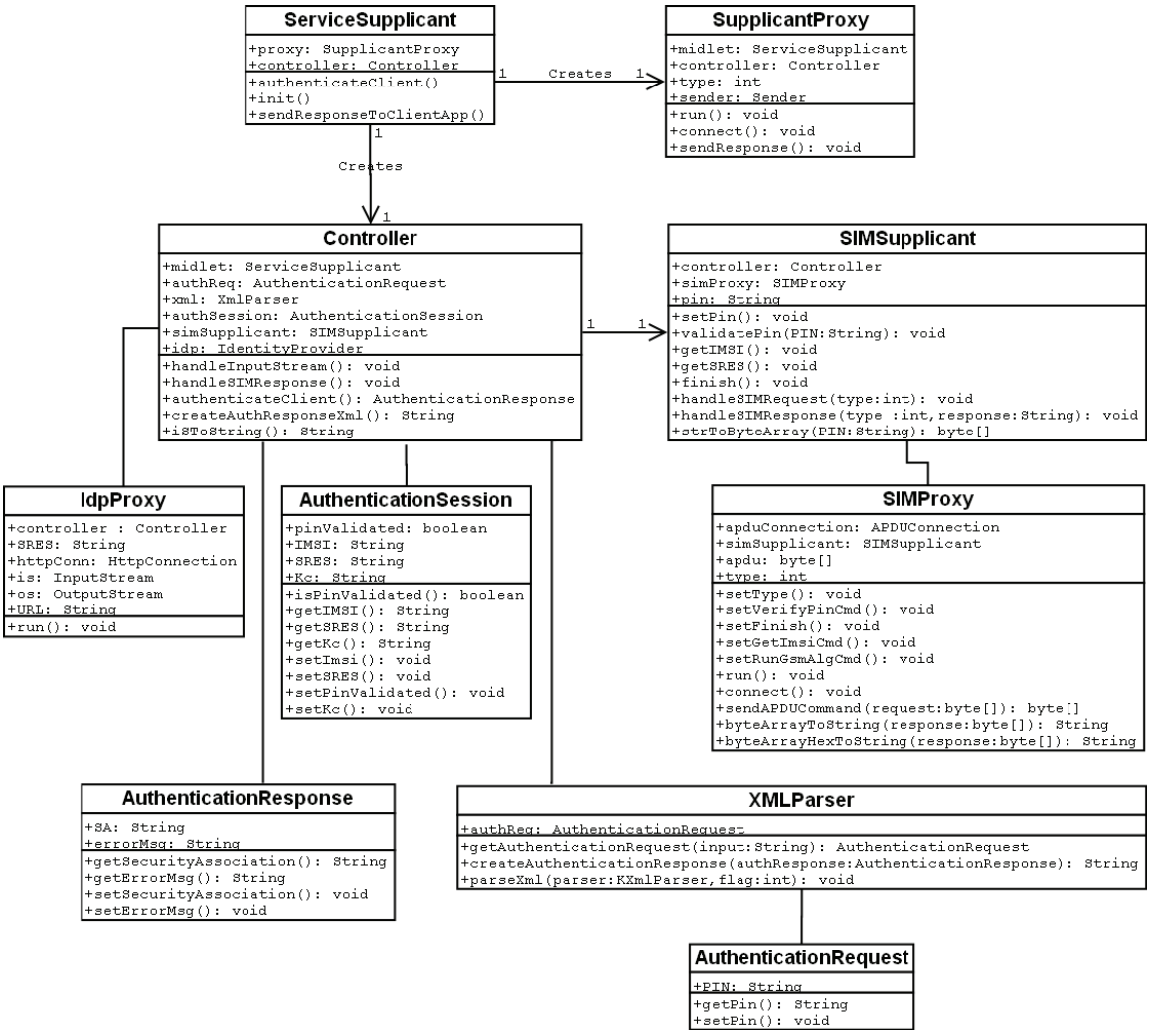
```
                    verifyCHV(apdu);
                    break;

              case GET_IMSI:
                    getImsi(apdu);
                    break;

              case RUN_GSM_ALGORITHM:
                    runGSMAlgorithm(apdu);
                    break;

              case INS_DUMMY:
                    sendResponse(apdu);
                    break;

              default:
                    ISOException.throwIt(ISO7816.SW_INS_NOT_SUPPORTED);
        }
      }
    }


    /**
     *
     * This method verifies the PIN in the APDU buffer against a PIN
established
     * when the applet was installed. If the values match, the method
returns
     * a the response code '9000'.
    */
    private void verifyCHV(APDU apdu) {

          //Obtains a reference to the APDU buffer (which contains the
message)
        //Only the first five APDU header bytes are available in the APDU
buffer.
        //CLA, INS, P1, P2, and P3 bytes
        //(Byte P3 denotes the Lc byte, if the command has optional data)
          //Lc byte denotes the number of bytes in the
          //data field of the command APDU

          byte[] buffer = apdu.getBuffer();
          byte pinLength = buffer[ISO7816.OFFSET_LC];


        //retrieve the PIN data for validation.
        //(indicate that this APDU has incoming data
        //and receive data starting from the offset
        //ISO7816.OFFSET_CDATA following the 5 header bytes).
        short bytesRead = apdu.setIncomingAndReceive();
        short echoOffset = (short)0;

        //Copies the incoming bytes to echoBytes
        while ( bytesRead > 0 ) {
          Util.arrayCopyNonAtomic(buffer, ISO7816.OFFSET_CDATA, echoBytes,
echoOffset, bytesRead);
```

```
        echoOffset += bytesRead;
        bytesRead = apdu.receiveBytes(ISO7816.OFFSET_CDATA);
    }

      // check pin
      // the PIN data is read into the APDU buffer
      // at the offset ISO7816.OFFSET_CDATA
      // the PIN data length = byteRead

      if (cardPIN.check(apdu.getBuffer(), (short)5, (byte)4) == false) {
          ISOException.throwIt(SW_VERIFICATION_FAILED);
      } else {

        //1. short bOff - the offset into APDU buffer
        //2. short len - the bytelength of the data to send
        //Send nothing but SW1 and SW2, which is '90 00'
        apdu.setOutgoingAndSend((short)0, (short) 0);

      }
    } // end of validate method


  /**
   * Simulates the get IMSI function and returns an imaginary  IMSI
   * @param apdu
   */
  public void getImsi(APDU apdu) {

        byte IMSI[] =  {(byte)1,(byte)2,(byte)3,(byte)4,(byte)5,
                        (byte)6,(byte)7,(byte)8,(byte)9,(byte)8,
                        (byte)7,(byte)6,(byte)5,(byte)4,(byte)3};

        apdu.setOutgoing();

        apdu.setOutgoingLength( (short) (IMSI.length) );

        apdu.sendBytesLong( IMSI, (short) 0, (short)(IMSI.length) );
  }

  /**
   * Simulates the run GSM Algorithm and returns SRES and Kc.
   * (In real life the response is covered by a MAC)
   * @param apdu
   */
  public void runGSMAlgorithm(APDU apdu) {

        //Byte 1-4 = SRES
        //Byte 5-12 is Kc according to GSM 11.11 version 5.0.0
        byte SRES_KC[] =  {(byte)4,(byte)4,(byte)4,(byte)4, //SRES
                                    (byte)8,(byte)8,(byte)8,(byte)8,
//Kc
                                    (byte)8,(byte)8,(byte)8,(byte)8};
//Kc

        apdu.setOutgoing();
```

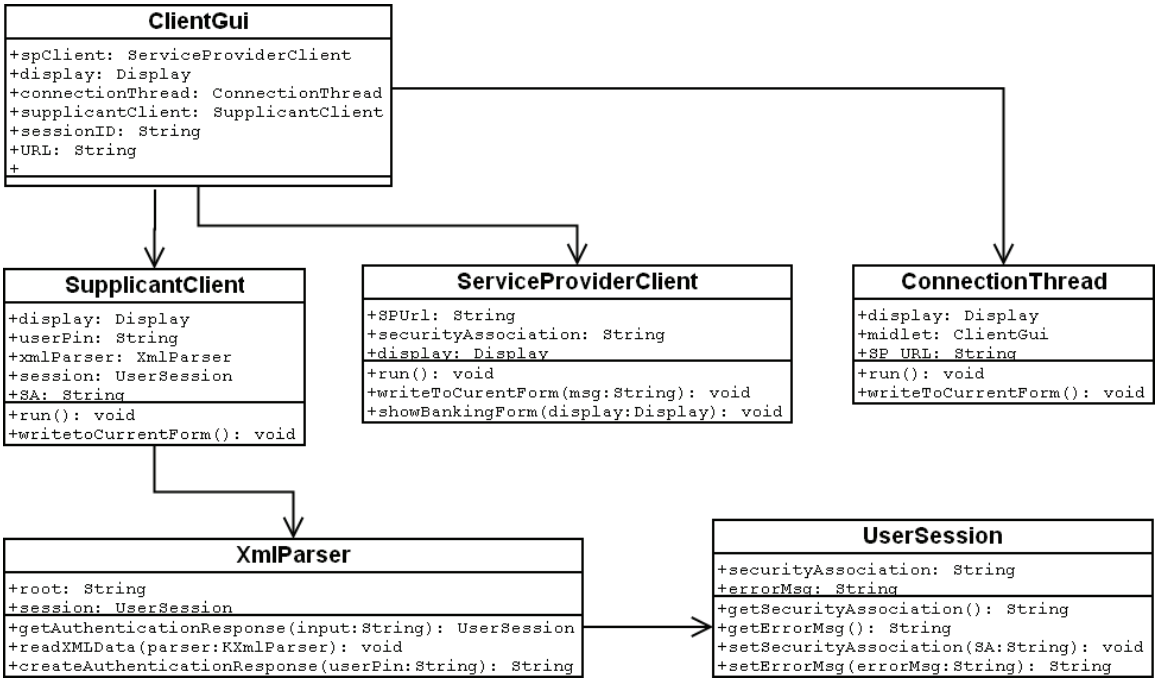
```
        apdu.setOutgoingLength( (short) (SRES_KC.length) );

        apdu.sendBytesLong( SRES_KC, (short) 0, (short)SRES_KC.length );
  }


  /**
   * This method echo's the received bytes
   * @param apdu
   */
  protected void sendResponse(APDU apdu) {
            byte buffer[] = apdu.getBuffer();

            short bytesRead = apdu.setIncomingAndReceive();
            short echoOffset = (short)0;

            while ( bytesRead > 0 ) {
          Util.arrayCopyNonAtomic(buffer, ISO7816.OFFSET_CDATA,
echoBytes, echoOffset, bytesRead);
          echoOffset += bytesRead;
          bytesRead = apdu.receiveBytes(ISO7816.OFFSET_CDATA);
      }

      apdu.setOutgoing();
      apdu.setOutgoingLength( (short) (echoOffset + 5) );

      // echo header
      apdu.sendBytes( (short)0, (short) 5);
      // echo data
      apdu.sendBytesLong( echoBytes, (short) 0, echoOffset );
  }



}
```

Appendix C – UML class diagrams

The Supplicant class diagram



The client application class diagram



Appendix D – List of current mobile handsets

SATSA-APDU enabled mobile phones:

All Nokia Series 40 3rd Edition Feature Pack 2 devices [32].

- Nokia 5200
- Nokia 5300
- Nokia 6085
- Nokia 7360
- Nokia 7373
- Nokia 7390

Phones supporting push registry with socket/datagram inbound connections

Probably all Nokia Series 60 3rd Edition devices. The author has successfully tested on Nokia N91. One have to run a real life test to discover whether the device supports push registry with socket/datagram inbound connection or not. The MIDP 2.0 specification does not specify this sufficient.

Phones supporting multitasking between J2ME MIDlets

All Sony Ericsson Java Platform 7 (JP-7) phones [51].

- K610i
- K790
- K800
- S500
- T650

Appendix E –Paper published on ERCIM 2007

The paper in appendix E adheres to the formatting standard for a 12-page manuscript of Springer-Verlag LNCS. The paper was published on the ERCIM workshop on eMobility in Coimbra, Portugal on May 21, 2007.

Below is the paper notification received from the program committee chairmen on April 11, 2007. The full paper is attached after the notification.

Dear Mr. Haavard Holje,
We have the pleasure to inform you that your paper entitled A Unified Authentication Solution for Mobile Services, submitted to the 1st ERCIM Workshop on eMobility, has been accepted for presentation at this workshop as full paper, after careful review by the Program Committee.

Authors are requested to adhere to the formatting standard for a 12-page manuscript of Springer-Verlag LNCS in preparing the camera ready version of their contribution. Also, the review comments included below must be taken into account.

Thanking once again for your active participation, we remain at your disposal for any further information, and we look forward to welcoming you to Coimbra.

The review reports are given below.

Yours sincerely,

Torsten Braun and Dimitri Konstantas
Program Committee Chairmen

Saverio Mascolo and Markus Wulff
Program Committee Co-Chairmen

%%%

% BEGINNING OF THE REPORTS PROVIDED BY THE REVIEWERS %
%%

Overall recommendation (1=Definite Reject, 2=Likely Reject, 3=Accept if room, 4=Likely Accept, 5=Definite Accept): 3

Reviewer's familiarity with the subject (1=Outside area of research, 5=Expert): 2

What are the contributions of the paper (major issues addressed, novelty and potential impact)?:
The authors propose a authentication solution to access any service based on the GSM authentication mechanism

What are the main reasons to accept this paper?:
The solution avoids duplicated authentication mechanisms.

What are the main reasons NOT to accept this paper?:
I think there are some open questions. I think that the user is asked to write something in order to ensure he/she is the person that says to be. With this procedure, only the handset can be identified. What happens if somebody uses my telephone without my permission?

Summary and comments to authors: Please provide maximum possible feedback towards paper improvement.:

.....

Overall recommendation (1=Definite Reject, 2=Likely Reject, 3=Accept if room, 4=Likely Accept, 5=Definite Accept): 4

Reviewer's familiarity with the subject (1=Outside area of research, 5=Expert): 3

What are the contributions of the paper (major issues addressed, novelty

and potential impact)?:
The problem is well defined, and the approach of using SIM-based authentication for applications other than mobile telephony is interesting.

What are the main reasons to accept this paper?:

What are the main reasons NOT to accept this paper?:

Summary and comments to authors: Please provide maximum possible feedback towards paper improvement.:
Perhaps the authors can discuss in more detail their design choices, justifying them and identifying other design alternatives. My understanding is that the proposed solution uses the J2ME platform. How about the other related work (authentication for WP/Web)? How does the proposed solution relate to them?

Also, I am a bit surprised that there doesn't exist related work following a similar approach: utilize SIM authentication capabilities.

.....

Overall recommendation (1=Definite Reject, 2=Likely Reject, 3=Accept if room, 4=Likely Accept, 5=Definite Accept): 3

Reviewer's familiarity with the subject (1=Outside area of research, 5=Expert): 4

What are the contributions of the paper (major issues addressed, novelty and potential impact)?:
The paper presents an architecture for Authentication Service on mobile devices.
The paper threats an important problem. Recent evolution of mobile devices has raised the problem of exploiting SIM authentication functionalities by "not-telephony" applications.

What are the main reasons to accept this paper?:

The paper describes a solution for Unified Authentication in mobile devices. The architecture hinges on the supplicant, a J2SE application running on the mobile device. This component is responsable of the authentication by exploiting the SIM functionality.

The architecutre is simple and functional.

What are the main reasons NOT to accept this paper?:

There is no prototype even if the authors claim that "the proposed system should be reasonable to implement".

The paper is hard to read, in particular Section 3.2 and Figure 3 is not well explained.

Summary and comments to authors: Please provide maximum possible feedback towards paper improvement.:

....

%%

% END OF THE REPORTS PROVIDED BY THE REVIEWERS %

%%

A Unified Authentication Solution for Mobile Services

[1]Håvard Holje, [2]Ivar Jørstad & [3]Do van Thanh

[1] Norwegian University of Science and Technology, O.S. Bragstads plass 2, NO-7491 Trondheim, Norway, holje@stud.ntnu.no
[2]Ubisafe AS, Bjølsengata 15, NO-0468 Oslo, Norway, ivar@ubisafe.no
[3]Telenor R&I - Norwegian University of Science and Technology, Snarøyveien, NO-1331 Fornebu, Norway, thanh-van.do@telenor.com

Abstract. Current mobile phone architecture does not provide adequate security support for applications. This paper contributes to the opening of the mobile terminal architecture by presenting a solution that enables non-telephony applications to make use of the strong authentication functions located on the SIM card. The solution is based around a supplicant which provides services with access to the native authentication mechanisms of a GSM/UMTS network. All communication between the supplicant and the network side is performed using standards protocols.

Keywords: mobile service, authentication, SIM, mobile handsets, Java, web services

1 Introduction

From a simple device terminating the mobile network, the mobile phone has evolved to become a quite advanced device capable of hosting applications that are until now run only on stationary computers. The limitations in terms of processing, storage and battery life are considerably reduced, and the mobile phone will soon become a mobile computer. However, there is one major obstacle, which is the current closed architecture of the mobile terminal. Indeed, the architecture is very much telephony centric, i.e. it is built to support the traditional telecommunication services like GSM voice, SMS, WAP, etc. Other applications like browsing, Web services, P2P applications get very little support and in most cases have to manage by themselves.

This paper contributes to the opening of the mobile terminal architecture by presenting a solution that enables non-telephony applications to make use of the strong authentication functions located on the SIM card.

The SIM card, which is a tamper resistant Smart card, utilizes ISO-standardized Application Protocol Data Units (APDU) to communicate with host devices via PIN codes and cryptographic keys.

Existing (strong) authentication schemes on mobile handsets suffer of serious drawbacks. Some are completely separated from the SIM and require additional

elements such as a Smart Card, a one-time password generator, etc. The others access the SIM authentication functions indirectly via SMS. The paper starts with a summary of related works. The Unified Authentication is then presented thoroughly. All the components are described in detail.

2 State-of-the-art in authentication on mobile phones

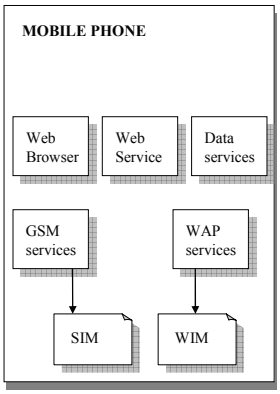


Figure 1. Mobile phone service architecture

The mobile phone is originally intended only for voice communication service, i.e. telephony. The mission of the SIM card is to carry the subscriber's identity and to provide security functions like encryption and authentication. In fact, the SIM card is acting as a slave executing order only from the GSM services which include telephony and SMS (Short Message Service) as shown in Figure 1.

The GSM network authenticates the identity of the subscriber through the use of a challenge-response mechanism. The GSM security model is based on a shared secret between the SIM and the AuC (Authentication Center) of the subscriber's home network. The shared secret (Ki) is a unique 128 bit key.

The authentication is initiated by the fixed network and it is based upon a simple challenge-response protocol. First the network identifies the MS (Mobile Station) by retrieving the IMSI (International Mobile Subscriber Identity). Next the MSC (Mobile Switching Center) contacts the MS's HLR (Home Location Register), and asks it to send a triplet containing RAND, SRES' and Kc. The triplet is computed by the AuC, which is the only entity in the GSM network knowing Ki besides the MS itself, and it is sent back to the MSC by the HLR.

When the MS retrieves the triplet, it will use the built-in authentication mechanisms in the SIM to generate a signed response (SRES). This is the unique part of the GSM authentication scheme, which makes it so strong. The private key is never

sent over the network. Instead the MS computes its own SRES by feeding the COMP128 algorithm with the local version of Ki and the newly retrieved RAND.

When the mobile phone evolves other services start to appear. Unfortunately, they usually do not have access to the security functions on the SIM card, but have to implement their own solutions. As shown in figure 1 there is no connection between the SIM card and emerging services like Web browser and Web services.

2.1 Authentication for WAP-based Services

For WAP (Wireless Application Protocol) application a WAP Identity Module (WIM) [1] is defined and used in performing WTLS (Wireless Transport Layer Security) [2] and application level security functions, and especially, to store and process information needed for user identification and authentication. The WIM functionality can be implemented on a smart card. A smart card implementation is based on ISO 7816 [3] series of standards. The WIM is defined as an independent smart card application, which makes it possible to implement it as a WIM-only card or as a part of multi-application card containing other card applications, like the GSM SIM.

2.2 Authentication for Web-based Services

For Web-based services accessed through a Web browser, stronger authentication is offered by using the One-Time-Password scheme. However, this solution is not the ideal one for mobile phone. The OTP must be generated by a device, which the user must bring along, or it can be sent to the user via SMS. Anyway, the user has to enter in the OTP manually, in addition to username and password, which might be a complicated procedure on small mobile handsets with poor keyboards.

Another option is to use a PKI-solution, which requires a PKI client installed in the SIM card as separate application. To carry out authentication, the Web site has to send challenges to the PKI Client using SMS as carrier. Only when the authentication is successful, the Web server will return to the mobile browser. Quite often, the browsing session has been terminated following of the termination of the data packet session, e.g. GPRS, UMTS.

2.3 Authentication for Java-based Services

The Java 2 Platform, Micro Edition (J2ME) is a Java platform optimized for small devices with limited memory and processing power, such as mobile phones and PDA's. J2ME is divided into configurations, profiles and optional packages. Devices need a configuration adapted to their processing capabilities and the profile implements higher-level APIs that further define the application life-cycle model, the user interface, persistent storage and access to device-specific properties.

For J2ME applications there is recently defined the Security and Trust Services API (SATSA/JSR177) which extends the security features for the J2ME platform, through the addition of cryptographic APIs, digital signature service, and user credential management. SATSA also defines methods to communicate to a Smart Card, by leveraging the APDU protocol. The SATSA spec says there is no Smart Card access for untrusted (unsigned) MIDlets. Hence one has to sign the MIDlet with a certificate issued by the operator or the manufacturer, to be able to connect to the SIM.

JSR-248 (Mobile Service Architecture), which defines the next generation Java platform for mobile handsets, mandates the support of SATSA-APDU when a security element exists on the device, i.e. a Smart Card or a SIM card.

With SATSA, the necessary security functions are offered to the J2ME applications but the architectural problem is still not solved. It is not simple for applications to make use of these security functions and there is no point to require that each application must integrate the security functions.

2.4 Related work

As far as the authors know there exists no similar solution to what is proposed in this paper. However, related work regarding SIM authentication do exist, but the existing solutions are either product specific or they are based on a regular PC and not a mobile phone. An example of a product specific solution is the built-in EAP-SIM supplicant provided on the Nokia 9500 communicator. It is a native application provided by the manufacturer and can be combined with the 802.1x framework to achieve strong SIM based authentication in WLAN's.

Another similar approach is the SIM Strong project, which aims to extend the use of GSM SIM authentication to internet Web Services. Telenor, Axalto, Linus and Oslo University College have implemented a proof-of-concept prototype together in Oslo [4]. The prototype demonstrates the possibility of implementing innovative services in a heterogeneous environment using Liberty Alliance Federation Standard [5].

3 The Unified Authentication solution

The goal of the Unified Authentication solution is to provide a unified authentication mechanism for any type of application and service on the mobile handset. In addition, there are the following requirements:

- The authentication mechanism must be strong
- The authentication mechanism must be mutual
- The authentication mechanism must be user-friendly
- The authentication mechanism should be simple to add to existing and future mobile services
- The authentication mechanism should be cost-effective to establish for service providers

The main idea is to utilize the fact that the GSM SIM is a tamper resistant Smart Card accomplishing the ISO8716 Smart Card specifications. By utilizing the existing GSM SIM authentication mechanisms for IP based services, we want to achieve strong two-factor authentication, without other user interaction than typing the PIN.

To ensure that the user is who he/she claims to be, the PIN function on the SIM must be enabled and the user have to provide a valid PIN to get access to the service. This mechanism prevents misuse if the mobile handset is stolen, since the SIM is blocked after 3 invalid PIN attempts.

As depicted in figure 2, the SIM Supplicant is located in the Mobile handset. The SIM Supplicant is an important component of the proposed system. It is responsible for all communication between the SIM, the browser and the Identity provider. To be able to get access to the SIM functions, the supplicant utilizes the Security and Trust Services API (SATSA) for J2ME. The SATSA-APDU package allows the supplicant to communicate with the SIM by leveraging the APDU protocol, which is an ISO7816 compliant low-level protocol for Smart Card communication.

When a user wants to access a service on a Mobile handset, he/she uses the browser to contact the Service Provider. If the service requires authentication, an authentication request is returned. The Supplicant is contacted and initializes the authentication procedure.

The SIM Supplicant provides the subscriber identity and valid user credentials to the Identity Provider. The Identity Provider performs a lookup of the user in an associated Authentication Server and uses the existing GSM authentication mechanisms to authenticate the user. The Authentication Server uses a GSM gateway to communicate with the GSM network during the authentication.

When the subscriber is authenticated against the Identity Provider, a security association (SA) is created and returned to the browser. When the Service Provider has verified the claimed authenticity, the user is authorized to use the requested service

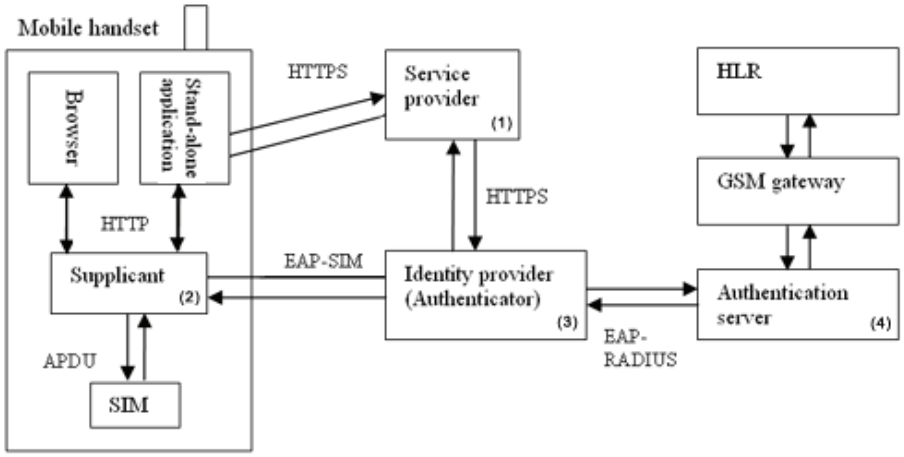


Figure 2. Overall architecture of generic SIM authentication system

3.1 Components

In this section the most important components in figure 2 will be explained in more details. The interface between the components is explained further in section 3.2

3.1.1 Service Provider (SP)

The Service Provider, component (1) in figure 2, offers services to the users and initializes the authentication procedure. When it receives a service request from a client, it responds with an authentication request to the local Supplicant on the mobile handset, if the client is not already authenticated. If the client provides an authentication token, the SP may contact the Identity Provider to verify the claimed authenticity. If the security association is valid, the SP authorizes the client to the requested service.

3.1.2 Supplicant

The Supplicant (2) is the major contribution to the proposed authentication system. It is a generic J2ME application acting as a local proxy on the mobile handset. The Supplicant provides an interface to the GSM SIM by utilizing the SATSA-APDU protocol. SATSA-APDU extends the security features for the J2ME platform, and makes it possible for the Supplicant to extract user credentials from the SIM.

The Supplicant is also implementing the EAP framework which makes it capable of exchanging EAP-SIM messages with the Identity Provider (Authenticator).

In an authentication sequence the Supplicant retrieves GSM authentication challenges from the Identity Provider by means of EAP-SIM messages. Next the Supplicant provides the GSM authentication challenges to the SIM and retrieves its user credentials by exchanging command and response APDUs, as defined by the ISO7816 Smart Card standard.

3.1.3 Identity Provider

The Identity Provider (3) is responsible for locating a suitable Authentication Server and it acts as an intermediary between the Supplicant, the Authentication Server and the Service Provider.

The Identity Provider translates EAP-RADIUS messages from the Authentication Server into EAP-SIM messages and passes it to the Supplicant. It is also storing information about authorized Supplicants, so the Service Provider can verify a Supplicant's claimed authenticity.

To get access to the Authentication Server, a RADIUS client must be implemented. A mutual trust between the Identity Provider and the Authentication Server is required.

3.1.4 Authentication Server

The Authentication Server (4) is performing the user authentication against the GSM network. It could be realized as a RADIUS server, which is the de-facto standard for remote authentication, but other Authentication Servers like DIAMETER may also be used. To perform the GSM SIM authentication, the RADIUS server will use the GSM gateway interface to contact the HLR of the subscriber.

3.3 Interfaces

3.3.1 SIM interface

To be able to communicate with the GSM SIM through the mobile handset it is required that the handset provides a SIM access interface. The SATSA-APDU package, described in section 2.3, provides such an interface. For more details regarding the SATSA-APDU API, we refer to SATSA Developer's Guide [6].

3.3.2 Authenticator interface

Extensible Authentication Protocol (EAP) [7] is a framework supporting multiple authentication methods. The Authenticator implements EAP-SIM [8] to communicate with the Supplicant, and EAP-RADIUS [9] to communicate with the Authentication Server. The EAP specification only discusses usage within a point-to-point protocol (PPP), which is a low layer protocol. The encapsulation of EAP messages in the higher layer protocols is not specified in the specifications. EAP is only dealing with authentication at the Application level.

Hence, there is a need to secure the communication channels between the components to ensure integrity and confidentiality. To solve this problem, EAP over TCP/IP may be chosen with SSL/TLS to maintain the integrity and confidentiality between the different components.

3.3.3 Supplicant interface

The Supplicant is a J2ME MIDlet running independent of the other applications on the mobile handset. It acts as an HTTP proxy and is listening to incoming request on a

local port. The interface between the Supplicant and the client application is defined by the XML schema in figure 5.

```
<?xml version="1.0"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <xsd:element name="Supplicant">
    <xsd:complexType>
      <xsd:choice>
        <xsd:element name="AuthenticationRequest">
          <xsd:complexType></xsd:complexType>
        </xsd:element>
        <xsd:element name="AuthenticationResponse">
          <xsd:complexType>
            <xsd:choice>
              <xsd:element name="Authenticated" type="xsd:boolean"/>
              <xsd:element name="SecurityAssociation" type="xsd:integer"/>
              <xsd:element name="ErrorDescription" type="xsd:string"/>
            </xsd:choice>
          </xsd:complexType>
        </xsd:element>
      </xsd:choice>
    </xsd:complexType>
  </xsd:element>
</xsd:schema>
```

Figure 5. XML Schema defining the Supplicant interface

The defined Supplicant interface is very simple and consists of either an AuthenticationRequest or an AuthenticationResponse. If the authentication succeeds, a security association is returned inside the AuthenticationResponse. Otherwise, an error description will be returned.

The defined protocol provides a generic interface to the Supplicant, which means that any kind of client applications supporting HTTP can communicate with the Supplicant, with none or minor modifications.

3.3.4 Client application – Service Provider (SP) interface

The client application communicates with the SP by opening a standard secured HTTP connection. When the SP receives a service request, it checks whether the client is authenticated or not. If the client is not authenticated, the SP will respond with an authentication request. If the requesting client is a browser, the SP will redirect the authentication request to the local Supplicant on the mobile handset.

When the client is authenticated via the Supplicant, it requests the SP again, but this time it provides an authentication token (security association) together with the service request. The SP verifies the claimed authenticity and if everything is ok, the client is authorized to use the requested service.

3.3.5 Service Provider (SP) – Identity Provider (IDP) interface

The SP must be able to communicate with the IDP as well. When the client application is authenticated, the SP must be able to verify the claimed authenticity. The SP sends a request to the IDP via a secured HTTP connection, and the IDP responds with the corresponding security association. The communication channel between the SP and the IDP must be secured with SSL/TLS and both parties must be

authenticated to each other. I.e. they have to exchange certificates to each other before they can communicate.

3.3 Design choices

We chose J2ME as platform for the SIM Supplicant to reach as many users as possible, since most of the handheld devices implement a J2ME runtime environment. One major challenge when developing for the J2ME platform is the closed architecture of the mobile terminal. It is difficult to get access to security functions like SIM access and other native features. But the SATSA-APDU package makes this possible

Another obstacle is the communication between the local Supplicant and the mobile browser. In principle not-native applications does not have access to other native applications because of the closed architecture. But since the mobile terminal is opening up, e.g. with the MIDP 2.0 Push registry [10], which enables MIDlets to set themselves up to be launched automatically without user initiation, it is just a matter of time before most of the mobile handsets will act as a small PC. The new Mobile Services Architecture (MSA) [11] defines the next generation of the Java platform for mobile devices, which will make it easier to develop applications for a broad range of devices.

3.4 Sequence diagram of successful authentication

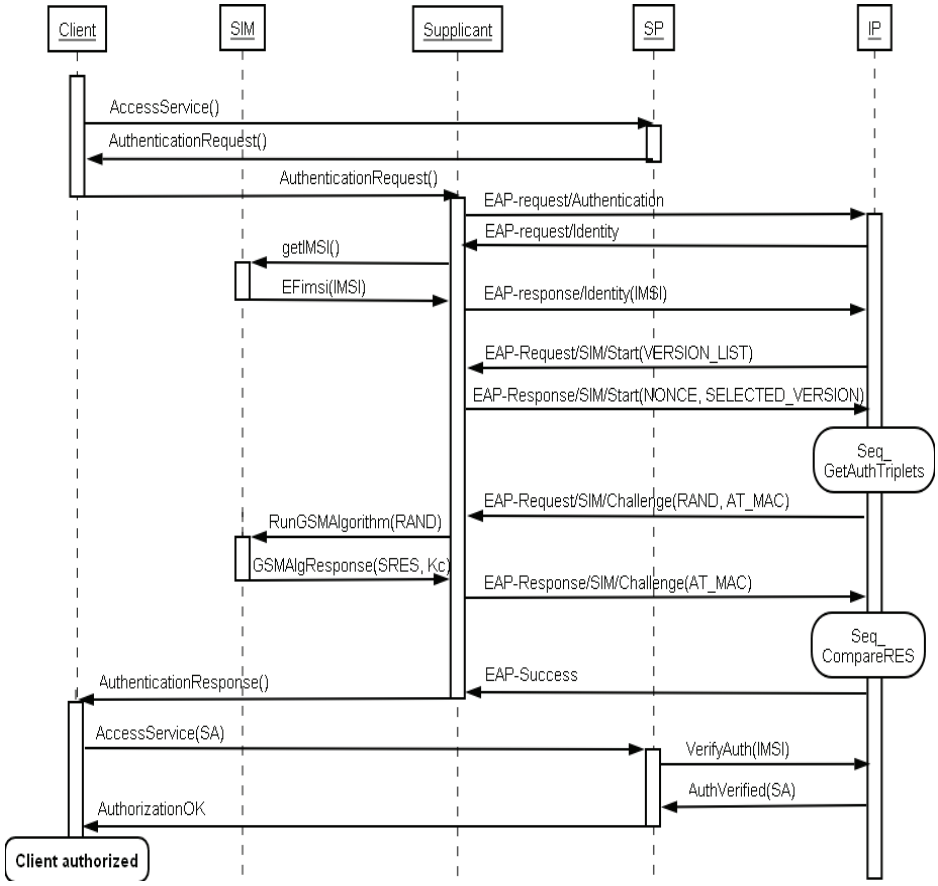


Figure 3. Sequence diagram for successful authentication of a subscriber

Figure 3 shows a sequence diagram for a successful authentication procedure. The client requests a service from the Service Provider (SP) and the SP responds with an AuthenticationRequest if the client doesn't provide a valid authentication token. The AuthenticationRequest is redirected to the Supplicant, which is responsible for authenticating the client against the Identity Provider (IDP). The IDP utilizes the existing GSM mechanisms for authenticating the client. The details of the GSM authentication procedure are hidden in this sequence diagram.

4 Conclusion and future works

This paper proposes the novel design of a unified authentication system for services accessed through a mobile handset. Specifically, the proposed solution makes use of the already ubiquitous authentication mechanism provided by existing GSM/UMTS networks.

By combining the GSM SIM authentication mechanisms with the EAP-SIM framework, we achieve mutual authentication between the parties. Since the SIM is a tamper resistant Smart Card, and the user has to present a valid PIN to activate the SIM, we have also achieved strong two-factor authentication, which fulfils the highest security level defined by NIST [12]. So we got a secure system, which is also easy to use, since the user only needs one single device, the mobile handset, to access the secure services. One of the greatest benefits is that the proposed system is generic, which means it can be used by both mobile browsers and stand-alone applications on the mobile handset.

Parts of the proposed system have already been implemented, and most of the external components needed already exist in the GSM network today. One challenge might be to realize the Supplicant as a local proxy on the mobile handset and to be able to run it on a broad range of mobile handsets. Another possible challenge is the communication with the SIM through the SATSA-APDU package. According to the specifications, there should be no major problems to extract the user credentials from the SIM, as long as the Supplicant MIDlet is signed with a certificate issued by the telecom operator. However, due to vague specifications, some challenges might arise which have not been predicted yet.

Integrating the solution with an Identity Management System (e.g. Liberty Alliance) [5] could be the next step for this project. To be able to offer security services in cooperation with many different types of Service Providers with proprietary technologies that do not interoperate easily, a standardized methodology for exchanging authentication data between security domains is required.

By adopting the SAML framework [13] or similar, it will be possible to offer the proposed system as a part of a single sign-on system (SSO) in the future.

Another possible enhancement is to utilize the GSM cipher key (Kc) for encryption purposes. The information exchange between the client application and the Service Provider must be protected by some means, and by utilizing Kc in a lightweight crypto package like Bouncy Castle [14], we can achieve end-to-end encryption without the bothersome key exchange, since Kc can be derived from the SIM.

References

1. WAP Forum, WAP-260-WAP Identity Module (WIM), 2001 http://www.wapforum.org/tech/documents/WAP-260-WIM-20010712-a.pdf
2. WTLS Specification WAP-199, http://www.wapforum.org/tech/documents/WAP-199-WTLS-20000218-a.pdf
3. ISO/IEC 7816, Smart Card Standard, http://www.iso.org
4. Do, T.V. et. al. (2006). "Offering SIM Strong Authentication to Internet Services", whitepaper, 3GSM 2006, http://www.simstrong.org/resources.php
5. The Liberty Alliance, http://www.projectliberty.org
6. Sun Microsystems, SATSA Developers Guide 1.0, 2004 http://sw.nokia.com/id/2e279a27-26ef-4435-8492-9ebae977aa9c/MIDP_SATSA_APDU_API_Developers_Guide_v1_0_en.pdf
7. B. Aboba ET.AL., IETF, RFC3748 - Extensible Authentication Protocol (EAP), 2004 http://www.faqs.org/rfcs/rfc3748.html
8. H. Haverinen ET.AL.,IETF, RFC4186 – Extensible Authentication Protocol Method for GSM SIM, 2006 http://www.faqs.org/rfcs/rfc4186.html
9. B. Aboba ET.AL., IETF, RFC3579 - Remote Authentication Dial In User Service (RADIUS) support for EAP, 2003. http://www.faqs.org/rfcs/rfc3579.html
10. Ortiz Enrique, The MIDP 2.0 Push Registry, January 2003 http://developers.sun.com/techtopics/mobility/midp/articles/pushreg/
11. Kay Glahn ET.AL, JSR 248: Mobile Service Architecture, 2006 http://jcp.org/en/jsr/detail?id=248
12. Burr, E William ET.AL, NIST (2006) Electronic Authentication Guideline, http://csrc.nist.gov/publications/nistpubs/800-63/SP800-63V1_0_2.pdf
13. OASIS Security Services, Security Assertion Markup Language (SAML), http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=security
14. Bouncy Castle Crypto APIs for Java, http://www.bouncycastle.org

Appendix F – Enclosed ZIP file

Appendix F is the enclosed ZIP file, containing the source code of the prototype.

The content of the ZIP file is:

- Supplicant
- Client application
- Service Provider
- Identity Provider
- SIM card applet
- Javadoc
- Tools
- Libraries

Please see the index.html file in the enclosed ZIP for further information.

References

1. Lars Lunde, A.W., *Using SIM for strong end-to-end application authentication* 2006, NTNU.
2. NIST *Electronic Authentication Guideline*, 2006
3. JCP, *JSR 248: Mobile Service Architecture,* 2006.
4. Lawrence, O., *IT Solutions Series: Information Technology Security: Advice from Experts*, 2004.
5. NorSIS, *Trusselbildet,* 2006.
6. Telenor (2006) O*ffering SIM Strong Authentication to Internet Services*.
7. Alliance, L. *Liberty Alliance Federation Standard*, 2001 (url: http://www.projectliberty.org/).
8. Gemalto. *Gemalto*. 2006 (url: http://www.gemalto.com).
9. BankID. *BankID*. 2006 (url: http://www.bankid.no/).
10. enCap, *The mobile phone as a security security device,* 2006.
11. Kent, S.T. *Who Goes There?: Authentication Through the Lens of Privacy*, 2003 (url: http://site.ebrary.com/lib/ntnu/Doc?id=10046903).
12. NIST. *FIPS PUB 140-2 Security requirements for crypthographic modules*, 2001
13. ISO/IEC. *ISO/IEC 9798-1: Security techniques - Entity authentication*, 1997 (url: http://www.pronorm.no/default.asp).
14. Dent, A. *User's Guide to Crytography and Standards*, 2004 (url: http://site.ebrary.com/lib/ntnu/Doc?id=10082005&ppg=190).
15. GSMA, *GSM Association Press Release, 2006*.
16. Hideki, I., *Wireless Communications Security*, 2006.
17. 3GPP, *Teleservices Supported by a GSM Public Land Mobile Network (PLMN),* 1999.
18. ETSI, *Subscriber Identity Modules (SIM); Functional characteristics*, 1999 (url: http://www.3gpp.org/ftp/Specs/archive/02_series/02.17/0217-800.zip).
19. ETSI, *General description of a GSM Public Land Mobile Network (PLMN)*. Volume, DOI: GSM 01.02 V6.0.1, 1998.
20. Redl, S.M., *An introduction to GSM*, 1995.
21. 3GPP. *SIM/USIM internal and external interworking aspects*, 2006 (url: http://www.3gpp.org/ftp/Specs/archive/31_series/31.900/31900-710.zip).
22. 3GPP. *3GPP TS 55.205 V6.0.0 Specification of the GSM-MILENAGE Algorithms*,

2002.

23. Josyula R Rao, P.R., Helmut Scherzer and Stefan Tinguely, *Partitioning Attacks: Or how to rapidly clone some GSM cards,* 2002.
24. IETF. *RFC4186 - Extensible Authentication Protocol Method for Global System for Mobile Communications (GSM) Subscriber Identity Modules (EAP-SIM)*, 2006 (url: http://tools.ietf.org/html/rfc4186)
25. IETF. *RFC3748 Extensible Authentication Protocol (EAP)*, 2004 (url: http://tools.ietf.org/html/rfc3748).
26. SUN. *The Java ME Platform*, 2006 (url: http://java.sun.com/javame/index.jsp).
27. Mahmoud, Q.H., *Secure Java MIDP Programming Using HTTPS with MIDP,* 2002.
28. JSR118, *MIDP 2.0 Specifications*, 2004 (url: http://jcp.org/aboutJava/communityprocess/final/jsr118/index.html).
29. WAPForum, *WAP Certificate and CRL Profiles*, 2001.
30. SUN, *Security and Trust Services API for J2ME (SATSA)*, 2004
31. Nokia, F. *Security using SATSA and SIM-card*, 2006 (url: http://discussion.forum.nokia.com/forum/showthread.php?t=75464&highlight=SATSA).
32. Suomela, H., *SATSA API on Nokia devices*, 2006. (url: http://blogs.forum.nokia.com/view_entry.html?id=225)
33. JSR248, *Mobile Service Architecture (MSA) Specification, 2006*.
34. Symbian. *Platinum partner program*, 2006 (url: http://www.symbian.com/partner/index.html).
35. Larman, C., *Applying UML and patterns*, 2002.
36. Cockburn, A.A.R. *Basic use case template*, 1998 (url: http://www.usecases.org/).
37. OASIS, *OASIS Security Services (SAML) TC*, 2006 (url: http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=security).
38. WAP Forum, WAP-260-WAP Identity Module (WIM), 2001 (url: http://www.wapforum.org/tech/documents/WAP-260-WIM-20010712-a.pdf).
39. WTLS Specification WAP-199, 2000 (url: http://www.wapforum.org/tech/documents/WAP-199-WTLS-20000218-a.pdf)
40. ISO/IEC 7816, Smart Card Standard, http://www.iso.org
41. Opera mobile, http://www.opera.com/products/mobile/
42. Kjell J. Hole ET.AL., Challenges in Securing Networked J2ME Applications (url: http://ieeexplore.ieee.org/iel5/2/4085604/04085618.pdf?tp=&arnumber=4085618

&isnumber=4085604&code=2

43. MIDP Application Security 3: Authentication in MIDP (url: http://developers.sun.com/techtopics/mobility/midp/articles/security3/)
44. Hanz Hager, Sony Ericsson. *Introducing multi-tasking and improved gaming performance for new Java Platform 7 (JP-7) mobile*. (url: http://developer.sonyericsson.com/site/global/newsandevents/latestnews/newsjune06/p_multitasking_improved_gaming_performance_jp7phones.jsp)
45. Sun Developer Network, Java Card Development Kit 2.2.1, 2003. (url: http://java.sun.com/products/javacard/dev_kit.html)
46. B. Aboba ET.AL, PPP EAP TLS Authentication Protocol, 1999. (url: http://www.ietf.org/rfc/rfc2716.txt)
47. Enrique Ortiz, The MIDP 2.0 Push Registry, 2003 (url: http://developers.sun.com/techtopics/mobility/midp/articles/pushreg/)
48. Dargan, P.A, Open Systems and Standards for Software Product Development, 2005.
49. Sun Developer Networks, Sun Java Wireless Toolkit for CLDC, 2007 (url: http://java.sun.com/products/sjwtoolkit)
50. Bouncy Castle Crypto APIs for Java (url: http://www.bouncycastle.org/)
51. Sony Ericsson JP-7 supported phones (url: http://developer.sonyericsson.com/site/global/newsandevents/latestnews/newsmar06/p_new_javaplatform7.jsp
52. ERCIM Workshop on eMobility, 2007 (url: http://emobility.unibe.ch/workshops/2007-05-21/workshop-cfp.html)
53. ETSI EG 201 220, Integrated Circuits Cards (ICC) - ETSI numbering system for telecommunication; - Application providers (AID), 1999.