

# A mobile single sign-on system

Mats Byfuglien



Master's Thesis  
Master of Science in Information Security  
30 ECTS  
Department of Computer Science and Media Technology  
Gjøvik University College, 2006

Institutt for  
informatikk og medieteknikk  
Høgskolen i Gjøvik  
Postboks 191  
2802 Gjøvik

Department of Computer Science  
and Media Technology  
Gjøvik University College  
Box 191  
N-2802 Gjøvik  
Norway

## Abstract

Today users have to manage a set of usernames and passwords for every service they are using. As the number of passwords grow people start writing them down, use easily guessable passwords or use the same password on different accounts. This severely reduces the security passwords provide and a better password managements system is needed. Single sign-on (SSO) is proposed as a solution to the password management problem. These solutions allow users to store their passwords in one place, and the user only has to remember one master password. Most of the SSO solutions available on the market today are either too expensive and complex for the common user or they lack mobility.

The goal of this thesis is to propose a new mobile single sign-on system that automates logins for the user. The passwords will be stored on a mobile phone and transferred via Bluetooth to a USB unit connected to the PC. This unit will be configured as a keyboard and emulate keystrokes as if the user was typing them on a conventional keyboard.

The main contribution from this thesis will be to perform a technical feasibility study to show if it is possible to implement the proposed concept. The result is a prototype of the solution with only the functional features implemented. A security analysis has been conducted on the prototype. The aim of the analysis was to find what security measures should be implemented to assure the security of the solution.

Also a user test was conducted to see how the concept was received by the potential users of the system. Results from the test show that a majority of the participants liked the concept and would like to use it on a daily basis.

**Keywords:** Single sign-on, Bluetooth, Adversary modeling, usability testing, USB, Java.



## Sammendrag

Brukere må i dag ha et sett med brukernavn og passord for hver tjeneste de vil benytte. Etter hvert som antall passord de må holde styr på øker, blir mange nødt til å skrive dem ned, velge passord som er lette å huske eller bruke samme passordet på flere kontoer. Dette er med på å kraftig redusere sikkerheten som passordene gir, og det er tydelig at et bedre system for passordhåndtering behøves. Single sign-on (SSO) er foreslått som en løsning på problemet rundt håndtering av passord. Slike løsninger tillater at alle passordene til en bruker kan lagres på et sted, og brukeren trenger kun å huske et hovedpassord. Mange av dagens SSO systemer er enten altfor dyre for en vanlig bruker, eller så mangler de mobilitet.

Målet med denne oppgaven er å legge frem et forslag til en mobil SSO løsning som tilbyr automatisk innlogging for brukerne. Brukerens passord lagres på en mobiltelefon og overføres via Bluetooth til en USB enhet på datamaskinen. Denne enheten vil være konfigurert som et tastatur og sende tegn inn til maskin akkurat som om de ble tastet inn av brukeren ved hjelp av et vanlig tastatur.

Hovedbidraget fra denne oppgaven vil være å vise i hvilken grad det er mulig å implementere den foreslåtte ideen, og resultatet vil være en prototyp av løsningen. Prototypen er kun en teknisk gjennomførbarhetsanalyse og inneholder kun de funksjonelle delene av systemet. Derfor har en sikkerhetsanalyse blitt gjennomført. Målet med denne analysen var å avdekke hvilke sikkerhetsmekanismer som må implementeres for at dette skal bli en sikkert løsning.

Det har også blitt gjennomført en brukertest for å finne ut hvordan brukerne likte dette SSO konseptet. Resultater fra denne testen viste at et flertall av brukerne likte konseptet og kunne tenke seg å bruke system til daglig.

**Nøkkelord:** Single sign-on, Bluetooth, Adversary modeling, Brukertest, USB, Java.



## Contents

<b>Abstract</b> . . . . .	<b>iii</b>
<b>Sammendrag</b> . . . . .	<b>v</b>
<b>Contents</b> . . . . .	<b>vii</b>
<b>List of Figures</b> . . . . .	<b>ix</b>
<b>List of Tables</b> . . . . .	<b>xi</b>
<b>1 Introduction</b> . . . . .	<b>1</b>
1.1 Topic . . . . .	1
1.2 Problem description . . . . .	1
1.3 Motivation . . . . .	1
1.4 Research questions . . . . .	2
1.5 Contributions . . . . .	2
1.6 Outline of the report . . . . .	3
1.7 Scope . . . . .	3
1.8 Acknowledgements . . . . .	4
<b>2 Related work</b> . . . . .	<b>5</b>
2.1 Single sign-on solutions . . . . .	5
2.1.1 Local pseudo SSO . . . . .	5
2.1.2 Proxy-based pseudo SSO . . . . .	7
2.1.3 Local true SSO . . . . .	7
2.1.4 Proxy-based true SSO . . . . .	7
2.2 Security in the Bluetooth protocol . . . . .	8
2.3 Methods for testing usability . . . . .	10
2.3.1 User tests . . . . .	11
2.3.2 Heuristic Evaluation . . . . .	12
2.3.3 Cognitive Walkthrough . . . . .	14
<b>3 Methods</b> . . . . .	<b>17</b>
3.1 Methodology discussion . . . . .	17
3.2 Test protocol . . . . .	18
3.2.1 Scenario . . . . .	18
3.2.2 Debriefing . . . . .	20
<b>4 Prototype design</b> . . . . .	<b>23</b>
4.1 Requirements . . . . .	23
4.2 Choice of technologies . . . . .	24
4.2.1 Hardware . . . . .	24
4.2.2 Software . . . . .	26
4.3 Design of the SSO prototype . . . . .	27
4.3.1 Design of the Bluetooth/USB device . . . . .	27
4.3.2 Design of the MIDlet . . . . .	30
<b>5 Security analysis</b> . . . . .	<b>35</b>
5.1 Security analysis methods . . . . .	35

5.1.1	Threat modeling	35
5.1.2	Attack modeling	35
5.1.3	Protocol analysis	35
5.1.4	Adversary modeling	36
5.1.5	Choosing a method to use in the security analysis	37
5.2	Security analysis of the SSO system	37
5.2.1	Principals	38
5.2.2	Channels	38
5.2.3	Protected assets	39
5.2.4	The adversarial setting	39
5.2.5	Intra-adversary channels	45
5.3	Security measures to implement	46
5.3.1	Secure the Bluetooth channel	47
5.3.2	Properly authenticate the devices involved	48
5.3.3	Protect the data stored on the mobile phone	49
5.3.4	Confirm the integrity of software/firmware updates	51
5.4	Final evaluation of the solution's security features	51
<b>6</b>	<b>Results from the user test</b>	<b>53</b>
6.1	Background information on the participants	53
6.2	The participants experience with other SSO solutions	54
6.3	The participants impression of the SSO concept	55
6.4	Commercial aspect	58
6.5	Comments from the participants	59
<b>7</b>	<b>Discussion</b>	<b>63</b>
<b>8</b>	<b>Further work</b>	<b>65</b>
8.1	Practical further work	65
8.2	Theoretical further work	67
<b>9</b>	<b>Conclusion</b>	<b>69</b>
	<b>Bibliography</b>	<b>71</b>
<b>A</b>	<b>Technical problems</b>	<b>75</b>
A.1	Setting up the USART	75
A.2	Finding services and connecting to the f2m Serial Port Plug	76
<b>B</b>	<b>Descriptor making the USB demo board work as keyboard</b>	<b>77</b>
<b>C</b>	<b>Scenario used in the usability test</b>	<b>79</b>
<b>D</b>	<b>Survey used in the usability test</b>	<b>81</b>
<b>E</b>	<b>Overview of the participants</b>	<b>87</b>



## List of Figures

1	Bluetooth protocol architecture . . . . .	9
2	Overall system design . . . . .	24
3	PICDEM FS USB Demo board . . . . .	25
4	J2ME platform . . . . .	27
5	Use Case . . . . .	28
6	Sequence diagram . . . . .	28
7	MIDlet screenshots . . . . .	31
8	Notation used in adversary modeling . . . . .	37
9	Adversary modeling diagram . . . . .	38
10	Price survey . . . . .	58



## List of Tables

1	Single sign-on products . . . . .	6
2	Security vulnerabilities in the prototype . . . . .	46
3	Number of user accounts . . . . .	54
4	Knowledge of SSO . . . . .	55
5	Rating of the SSO solution . . . . .	55
6	Summary of statements . . . . .	56
7	Configuration bits . . . . .	76
8	Overview of the participants in the user test . . . . .	87



# 1 Introduction

## 1.1 Topic

The topic of this thesis is password management. A new single sign-on concept that makes it possible for users to store passwords on a mobile device and use this device to sign in on every computer they use is proposed. The aim is to conduct a technical feasibility study. This will result in a functional prototype. In addition a security and usability analysis will be conducted on the prototype. The aim of the usability analysis is to find out what users think of this new single sign-on concept.

## 1.2 Problem description

People today have a multitude of different accounts such as online-shops, work accounts, home, Internet Service Providers, e-mail, instant messaging, online banking etc. As a result people have to remember a lot of passwords. This is a problem because people start writing down their passwords, choose easily guessable passwords or use the same password on different accounts. These approaches severely reduces the security passwords provide. For this reason there is a need to find a better way to manage passwords that doesn't require the user to remember passwords or write them down.

## 1.3 Motivation

There are mainly three types of authentication i.e. what you know, what you are and what you have. Passwords is an example of something you know. Something you are means biometric authentication such as fingerprint scanning, voice recognition etc. You use a physical object such as a smart card when authenticating with something you have. A combination of these authentication types are often used.

Biometric authentication is becoming more and more widespread. Fingerprint scanners have been installed on laptops. Face recognition data is stored on all newly issued passports. Even though biometrics are getting popular, there are several security issues to consider. A main problem is that a biometric feature is not secret. Keeping your face, voice or fingerprints secret is an impossible task. This means that an impostor can easily collect your biometric information. Several approaches for defeating biometric systems with relatively simple approaches has been pointed out in [41] and [33]. Some of these attacks have been successfully completed without the legitimate users cooperation. An attacker could for example lift a fingerprint from a used glass when the user is finished with it, and use it to trick the fingerprint sensor.

Another problem with biometrics is that they cannot be changed. This means that if the fingerprints are compromised, they can no longer be deemed as secure.

Tokens, or something you have, is rarely used as a single means of authentication. They are often used for storing other authentication data, such as biometric data or passwords, in a secure manner.

According to [33] passwords is one of the most secure means of authentication. However, this requires that passwords are managed correctly. This is often not the case. For

a password to be secure it should consist of a random mix of upper case letters, lower case letters, numbers and special characters. According to common security guidelines such as [16] and [48] a password should be at least eight characters long. Also the user should have different passwords for each account, and they should all be changed regularly. For most users, it's impossible to meet these requirements. As a result users choose passwords that are easy to remember or they write them down. It's also common to have the same password on several accounts. This severely reduces the security passwords provide.

Even though biometric and other alternative authentication systems are on the rise, passwords is still the most common mean for authentication. One reason for this is because password systems are simple and cheap to implement. Also most users are comfortable using passwords. Because of this, it is likely that passwords will continue to be a popular authentication method for time to come. This will require some modifications to how passwords are managed today. Weaknesses in current password management systems also introduce a substantial cost to businesses. Large companies spend a lot of money on help desk resources. Handling forgotten passwords takes up a significant part of the help desk's available resources. Also users with many different accounts waste time authenticating themselves over and over again.

By introducing single sign-on some of these problems can be mitigated. A SSO solution could manage the users' passwords. With a solution like this, the user doesn't have to remember his passwords allowing the passwords to be more complex. The SSO solution could even generate secure passwords, to assure their security.

Since the solution proposed in this thesis stores passwords on mobile devices, it would benefit users who are not confined to a single computer. If the user has the USB device and his mobile phone, he can sign in automatically to every computer he uses. Users who use only a single computer, but have several passwords, could also benefit from this solution because it will provide them with a tool for managing passwords in a secure manner. Essentially, almost everyone who uses a computer can benefit from this solution.

Finding a better way to manage passwords is the main motivation for this thesis.

## **1.4 Research questions**

The following research questions are identified:

1. What types of single sign-on solutions are available on the market?
2. How secure is the Bluetooth protocol for transferring sensitive data?
3. Is it possible to implement the proposed single sign-on concept?
4. What security mechanisms need to be in place to assure the security of this system?
5. How will this SSO concept be received by the users?
6. Will this SSO concept increase the users' security level?

## **1.5 Contributions**

This thesis will consist of a technical feasibility study showing to what extent it is possible to implement the proposed concept.

The solution will make it possible to automatically complete login forms on a computer from a mobile device without the need to install any drivers on the computer. This

will be done by configuring the USB/Bluetooth device as a keyboard. When the username and password are transferred to the unit, the characters will be received on the computer just as if they were typed on a conventional keyboard by the user. In addition a security and usability analysis will be conducted. The security analysis will identify what security measures that needs to be implemented to make the system secure. And the usability analysis will give an indication on what the users think of this new single sign-on concept.

Another important contribution from this thesis will be to show to what extent use of the prototype will result in more secure user behavior than if users were to manage their passwords as they do today.

## **1.6 Outline of the report**

This thesis contains five main parts. First there is a chapter about related work. Here other single sign on solutions are presented. It also includes a section on security in the Bluetooth protocol. Different methods for usability testing are also discussed in this chapter.

Following this chapter is a description of the methods used to conduct the user test. Next there is a chapter that describes how the SSO prototype is designed and implemented. The choice of technology and hardware is also described here.

In chapter five different methods for conducting security analysis are discussed. The method best suited is used to conduct a security analysis on a fairly high level of abstraction. Based on the results from the test, several security measures that should be implemented in the system are presented. Finally, in chapter six, the results from the user test are discussed.

In addition to the five main parts, the thesis also has a chapter that discusses the results obtained. Also there is a further work chapter that suggests what be done in the future to improve the system. The conclusion of the thesis is found in chapter eight. It is also included a couple of appendices that describe technical issues and other information that is relevant but does not fit in the main body of the thesis.

## **1.7 Scope**

An important part of this thesis will be to implement the SSO solution proposed as a technical feasibility study. Due to time limitations, the prototype will be a rough first version. Functionality is the main focus of this version. The prototype will have all the functionality required to successfully perform a usability test. Since the security of the solution is transparent for the users, this is not implemented. Simply because it would take too much time. A preliminary security analysis will be conducted. Suggestions on what security measures should be implemented in the system will be made based on the results from this analysis. The suggestions will focus mainly on measures for securing the wireless link, authenticating the devices and protecting the data in the MIDlet. Detailed security analysis is beyond the scope of this thesis, because it would require specific security measures to be implemented.

Also a user test will be conducted to get a feel for what users think of the solution. A large scale user test is however beyond the scope of this thesis. So is a usability test where the users are supposed to find problems with the user interface and other details of the implementation. This is because most of these details will be changed in later

versions.

## **1.8 Acknowledgements**

First I would like to thank my supervisor Einar Snekkenes for excellent guidance and for helping me to acquire the hardware I needed in order to complete the prototype;

Torkjel Sønderol for giving me an introduction on how to use the MPLAB IDE and MPLAB ICD2;

Vegar Johansen for providing me with literature on USB device development;

The support team at free2move for helping me make the f2m serial port plug work;

The staff at GUC's library for helping me acquire relevant literature;

Frode Volden for helping me with planning the usability test;

The people volunteering for the usability test;

My opponent Tor Erik Buvarp and Tommy Egeberg for reading my thesis and providing useful comments;

And finally my girlfriend Vibeke for showing patience during this hectic period.



## 2 Related work

This chapter will present some of the single sign-on solutions available on the market today. Also some proposed solutions that are not yet developed are presented. A discussion on the security in the Bluetooth protocol is also included in this chapter. The chapter is ended with a description of different methods for conducting usability analysis.

### 2.1 Single sign-on solutions

A taxonomy of single-sign on systems has been proposed by Pashalidis in [36]. This taxonomy identifies four main categories that SSO products fit into:

- Local pseudo SSO
- Proxy-based pseudo SSO
- Local true SSO
- Proxy-based true SSO

Table 1 shows where a selection of SSO solutions belong in the taxonomy.

Pseudo SSO systems use a component that automatically executes the authentication mechanism at the different service providers, such as usernames/passwords, X.509 certificates etc. At the beginning of a session the user authenticates once to the pseudo SSO component. For pseudo SSO systems a separate authentication occurs between the SSO component and the service each time the user requires it. Local pseudo SSO means that the SSO component is stored locally at the users machine. Proxy-based pseudo SSO means that the SSO component is located on an external proxy and authentication is performed between the proxy and the service provider. In the latter case, the local machine never has access to the access credentials.

In a true SSO system the user authenticates to an Authentication Service Provider (ASP). The ASP has a relationship with every service provider it provides SSO to. The user authenticates himself to the ASP once. Service providers are notified of the status of the authentication via authentication assertions. Usernames and passwords will not be passed to the service provider as is the case with pseudo SSO. A local true SSO system uses a trusted component within the user system which acts as an ASP. In a proxy-based true SSO system, the ASP is located on an external server. This server acts as a broker between the external user and the service provider.

Pseudo based SSO solutions are transparent and don't require any changes to the applications/service providers. True SSO solutions on the other hand, are not transparent.

#### 2.1.1 Local pseudo SSO

The simplest SSO solution is a password manager that does nothing but store your passwords securely, usually in an encrypted database. These solutions act as a digital safe, where the user has to type a secret master password in order to access the password database. When the master password is confirmed, all the passwords stored in the database are available in clear text.

SSO solution	Local pseudo SSO	Proxy-based pseudo SSO	Local true SSO	Proxy-based true SSO
Password Director [19]	X			
Secure Word.Mobile [2]	X			
MobiPassword [12]	X			
Online Password Manager [7]		X		
MyPasswordManager [30]		X		
SSO from untrusted devices [37]		X		
SSO using trusted platforms [38]			X	
Microsoft passport [28]				X
Kerberos [23, 32]				X
Mobile SSO using GSM/ UMTS [39]				X
Protocom SecureLogin [40]				X
Utimaco - Safeguard [54]				X
SSO architecture with dynamic tokens [45]				X
SSO using cookies [44]		X		

Table 1: A selection of single sign-on products for each category based on the taxonomy in [36].

Some of these solutions also provide additional tools such as testing the strength of the passwords, generating secure passwords for you and automatically capture passwords from password fields shown on the screen. However, they do not provide functionality for automating the login procedure.

A problem with this kind of SSO solution is that it depends on a master password. If you forget this password, there is no way to retrieve the secret information. Also it's important to choose a secure master password, that is easily remembered. The use of a single master password acts as a single point of failure. If an attacker manages to get the master password he will have access to all your passwords.

An example of this kind of single sign-on solution is Password Director [19]. Password Director also provides support for storage of the password database on external devices such as smart cards, USB-sticks etc. This makes it possible to carry your passwords with you at all times, and use the passwords on every computer where this software is installed.

SSO products that store passwords securely are also developed for mobile devices, such as SecureWord.Mobile [2]. Mobile SSO products work in a similar way as the SSO products described above. They provide secure storage, but no automated logins. Some mobile solutions can communicate with similar products on a computer, making sure the data is synchronized.

One of the better SSO solutions available for personal use is MobiPassword [12]. This is a program that stores all your passwords in a database and it has the possibility to provide automated logins. The user loads the website he wants to access. Next the user clicks in the user name field to identify where the login form is. Finally the user presses the login button in MobiPassword and the program enters the credentials. Next time the site is loaded, the login is completed automatically. Unfortunately, this solution only works on web forms.

### 2.1.2 Proxy-based pseudo SSO

An example of proxy-based pseudo SSO is web based single sign-on. Products such as Online Password Manager [7] and MyPasswordManager [30] are examples of this. Web based SSO works in almost the same way as SSO products confined to a single computer. The main difference is that the encrypted database with the passwords are available online on an external proxy server. This makes it possible to access the passwords from any computer connected to the Internet. When using web based SSO products it's even more important to use a secure master password. Since the database is available online, the only thing preventing an attacker from viewing your passwords is the master password. MyPasswordManager [30] also provides a function to automate the login procedure on web forms by using auto complete.

Another proxy-based pseudo SSO solution is proposed by Pashalidis and Mitchell [37]. This scheme is designed to be used in untrusted networks, such as a terminal at an Internet café. Since the user doesn't trust the network, it's not safe to provide passwords over it. Therefore a trusted proxy is introduced. The user authenticates himself to the proxy once. And the proxy, which stores copies of the users access credentials, makes the real authentication to the service provider.

In [44] Samar proposes a SSO solution using cookies. This solution uses a centralized cookie server for performing the authentication. When the user has successfully authenticated to the cookie server, he is granted access to all services the solution supports for as long as the cookie is valid. However, since this is a cookie based solution, it will only work on web applications.

### 2.1.3 Local true SSO

Pashalidis and Mitchell describes a local true SSO solution using a Trusted Computing Platform Alliance (TCPA) compliant platform in [38]. The scheme requires a trusted component and a public key infrastructure (PKI) to be in place.

### 2.1.4 Proxy-based true SSO

A popular web based or proxy-based true SSO solution is Microsoft passport [28]. Users register with a valid e-mail and password. Once they have logged in they have access to all the services on the Passport network. This is achieved by storing a ticket in the form of an encrypted cookie in the users browser. The functionality is quite similar to Kerberos [23] (described below).

A mobile proxy-based true SSO solution is also proposed by Pashalidis and Mitchell [39] using GSM/UMTS<sup>1</sup> operators as the authentication service provider (ASP). This solution authenticates subscribers to service providers in a way that is transparent to the users. The proposed protocol requires some minor changes to the existing GSM infrastructure.

The SSO solutions on the market with the best functionality are the ones made for businesses market.

Two good SSO solutions for the business market are Protocom SecureLogin [40] and Utimaco - Safeguard [54]. Both these systems implement a centralized manager. The user only authenticates to this manager once. After that the manager automates all logins to

<sup>1</sup> The Global System for Mobile Communications (GSM) is the most popular standard for mobile phones in the world. Universal Mobile Telecommunications System (UMTS) is one of the third-generation (3G) mobile phone technologies. Definitions from wikipedia.org

services the user has access to. Companies that implement these solutions get an effective access control system in addition to a SSO system. These solutions support a large number of login types ranging from simple web forms to database servers and terminal emulators. The security of these solutions are also quite good. Since the database that stores all the passwords are on a centralized server, it can be well protected behind firewalls and other network perimeter security devices. It's possible to allow mobile users access to the database via secure channels such as VPN<sup>2</sup>. These products also provide support for other authentication methods than master passwords, such as a smart card in combination with a PIN. This makes the authentication procedure more secure.

Another solution proposed is to implement a SSO solution using Kerberos V [32]. The user has to authenticate himself to the key distribution center (KDC) using the strong authentication protocol in Kerberos. You are issued a ticket after authenticating once to the KDC. As long as the ticket is valid, usually about 8 hours, users have access to every service they are allowed to use. Kerberos also provides access control in addition to SSO.

The downside to these solutions is that they are not an out-of-the-box product that you simply install and run. The configuration can be very complex and requires knowledge and experience in order to make it work correctly. Also they require specific hardware and operating system platforms. Because of this, these solutions are mostly implemented in large companies, smaller organizations simply cannot afford or justify the cost.

Finally Satoh and Itoh [45] describes a different approach to SSO. Instead of relying on static tokens such as user name and passwords, this scheme introduces dynamic tokens. These tokens can describe data such as payment history etc. The architecture also introduces a new server called circulator. This server is responsible for distributing the latest token values to the service providers. Most SSO schemes today requires an extra process when payment is required, such as a pay-per-view service. The solution proposed in this scheme will incorporate SSO and payment in a single process.

## 2.2 Security in the Bluetooth protocol

Bluetooth [1] is a radio based cable replacement technology that allows a number of different devices to connect to each other in a simple way. The Bluetooth protocol architecture is shown in figure 1. When two or more Bluetooth devices connect, they form a personal area network (PAN). The low price and small size of Bluetooth chips, means it possible to incorporate Bluetooth support in to almost any electronic device. Everything from mobile phones, PDAs, laptops to headsets, printers and mice can support Bluetooth. One reason for the popularity of Bluetooth is because of the simple discovery and connection of devices. The first activated Bluetooth unit in the area acts as the master device. All the other devices will be slaves. The master unit is constantly polling to see if any new Bluetooth enabled devices is within its range. That is between 10 and 100 meters depending on which class the device supports. Bluetooth devices in the area can be set to general discoverable mode or non-discoverable mode. Devices in general discoverable mode receives a poll from the master device to start pairing sequence. This sequence will perform a mutual authentication of the devices using a challenge response protocol. This process allows the devices to trust each other. Once the pairing is complete the service discovery application profile (SDAP) starts looking for service profiles supported by the

---

<sup>2</sup> A virtual private network (VPN) is a private communications network usually used within a company, or by several different companies or organizations, to communicate over a public network.

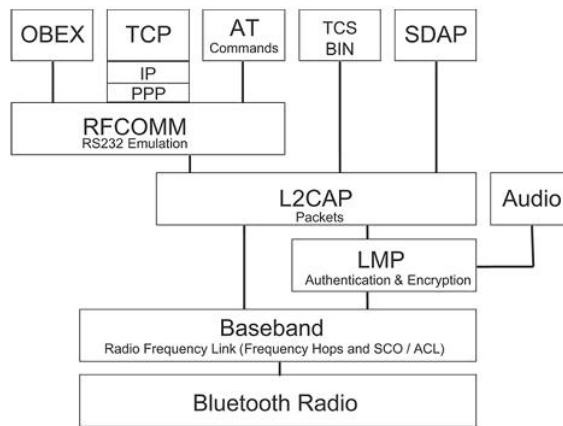


Figure 1: Bluetooth protocol architecture. Image from [13]

other device.

Bluetooth has some built-in security features. During pairing of the devices, a unit key is used. This key consists of a 48-bit Bluetooth address and a randomly generated number. These values are sent in clear text to the other device and forms an initialization key. This key is combined with a secret PIN or Passkey to form a link key. The secret has to be entered by the user and needs to be identical on both devices and is used for mutual authentication. If the Bluetooth connection requires encryption as well, the encryption key is derived from the link key. This approach is convenient, fast and requires little memory and processing overhead. However, this method introduces some security concerns. The only thing secret in the link key and encryption key is the PIN which due to its short length, usually only four digits, is vulnerable to brute force attacks. Also many devices without a user interface use a default PIN (0000).

Bluetooth also supports 128 bits link level encryption. However this is not default and has to be defined when setting up the connection. It is also important to note that data is not encrypted before the devices have successfully authenticated, i.e. all the authentication data is sent in clear text.

In [29] [9] the Bluetooth security architecture and the Bluetooth security are described. Papers such as [13], [14] and [55] describe several vulnerabilities in the protocol. It's for example mentioned that there are a multitude of Bluetooth hacking tools available on the Internet. One such tool is Bluestumbler which is a sniffing software that can monitor and log different Bluetooth devices. There is also a tool called Bluebrowse which basically functions as a conventional port scanner.

One of the more serious attacks in Bluetooth is BlueSnarfing [10]. This attack makes it possible to access data on the victim's device such as contact information, calendars, images, business card etc. All this can be achieved without ever alerting the owner of the device. BlueSnarfing also makes it possible for the attacker to send SMS, initiate calls and access the Internet from the compromised device. The experiment in [10] shows that a large number of the most popular mobile phones are vulnerable to this kind of attack. BlueSnarfing is usually only possible if the mobile device is set to discoverable

mode. However, additional experiments mentioned in [10] shows that some phones are vulnerable even when in undiscoverable mode.

Another attack on the Bluetooth protocol comes from exploiting the trust already established between two devices. Since all key exchange data is sent in clear text, it's possible for an attacker to record this transmission. Later he can use the data obtained to recreate the secret PIN code. In [47], a detailed method for cracking the Bluetooth PIN is presented. The Bluetooth protocol has no integrity checking. This makes man-in-the-middle attacks and packet modification possible. One such attack is described in [18]. Relay attacks is a special kind of man-in-the-middle attack that is also possible in the Bluetooth protocol. The goal of this attack is impersonation. Where the attacker talks to victim A posing as victim B and to victim B posing as victim A. All packets sent from A are captured by the attacker before they are forwarded to B. In order for this attack to work, victim A and victim B have to be in two different piconets. One example of a relay attack is discussed in [20].

Bluetooth only supports device authentication. User authentication is not within the scope of this protocol. As a consequence, an unauthorized user can gain access using a lost or stolen device.

In addition to the Bluetooth vulnerabilities mentioned above, it's also important to note that the user can choose to switch off the security completely. This means that the Bluetooth enabled device is not protected at all and open to any kind of attack.

Even though Bluetooth has several known security issues, this protocol will be used for transferring data between the mobile phone and the USB device. There are several reasons for this. First, the device developed at NISlab<sup>3</sup> that the mobile phone is communicating with is going to support Bluetooth. Second, this makes it simple to implement the basic communication between the devices because Java has a Bluetooth API. Also the Bluetooth protocol is supported on a large number of different devices. It is also user friendly and flexible. In addition, a lot of people are already using Bluetooth on a daily basis. This means that the prototype will use a technology user already know. This will reduce the skill level required for using the prototype and hopefully make it more user friendly.

### 2.3 Methods for testing usability

Testing an application's usability is extremely important. This is especially true for security applications because a security application is not something a typical user will want to learn. When users sit down by their computers, they usually want to perform typical user tasks, such as browsing the web, sending e-mails, working with a word processor etc. As stated in [58], users rarely sit down at the computer wanting to manage their security. They expect the security to be in place while they perform their user tasks. Because of this, users must understand the reason for using the application, i.e the added security it offers. Since using a security application is not a main user task, it's important that the application is easy and intuitive to use.

There are several different ways of conducting usability tests. In the sections below, three common types of usability tests are described:

---

<sup>3</sup> NISlab is Norwegian Information Security laboratory. This is the information security community at Gjøvik University College.

- User tests (experiments)
- Cognitive Walkthrough
- Heuristic Evaluation

In [58] a complete usability test where these three methods are used is described.

### 2.3.1 User tests

User tests is the practical approach to usability testing. These tests can mainly be conducted in two different ways. The test can be performed as practical and as close to real life usage as possible. This means that the user is given a case scenario that includes the use of security. In order to complete the case, users have to utilize the security application being tested. When the test is performed in this way, the user will use the application just as in a real setting. The goal of the scenario will be something not related to security, making security tasks the secondary goal. Just as it is in real life. In [58] user tests are one of the methods for evaluating the usability of PGP. The volunteers were told that they were a manager for an election campaign, and the objective of the scenario, was to send a sensitive e-mail to members of the campaign. Since the information is sensitive, users will understand the importance of encrypting the information.

An approach like this will be much closer real user scenarios than if the test subjects were directly told to use PGP for encrypting an e-mail. If they had only been told to encrypt an e-mail with PGP, this would have become their main goal. And when it becomes their main priority, they approach the problem in a whole other way, making the test more unrealistic. A well designed security application should make it easy for the user to understand how to use the security when he or she needs it.

Practical user tests can either be performed under controlled circumstances in the laboratory or in a real environment where it's likely that the application will be used. Either way, it's important to let the user work without interference from the test team. The most important information obtained from a user test is to see where the user gets stuck and what he does to get unstuck.

The other way to conduct a user test is to give the user specific tasks. This is actually the opposite of practical user tests described above. The tasks given to users are often designed to test specific functions of an application. A test like this is usually supplementary to the practical user test. Some parts of a program can be difficult to include in the practical test. Still, it's important that these functions are tested by users as described in [58].

A user test that combines practical user test with direct analysis provides the application designers valuable feedback about the usability of the system.

Selecting participants is also an important part of user testing. It's important that the selection of participants are representative for the population expected to use the application. If the application being tested is going to be commercially available - i.e. used by a large number of people with different backgrounds - it's important to select participants that represent the different backgrounds, such as different genders, ages, educational background, IT knowledge etc. The wider and more evenly distributed the range of people, the better.

On the other hand, if the application is only going to be used by a limited number of



people, only people representing this user group should participate in the test.

The number of people participating in the user test is also something to consider. The more people participating in the test, the better. A large number of people will result in more information gathered. However, user tests are time consuming. Because of this, the resources available often forces a maximum limit of participants to be set. It is also common to let a limited number of people test the earliest versions of the application. Based on feedback from these users, the application can be developed further and the next test can include more people. This cycle will continue for as many iterations as the developers find necessary.

There are several ways to gather data during a user test. Some common techniques include:

- Observation
- Record user action with a video camera
- Let the system log every action performed by the user
- Interview
- Questionnaire
- Instruct the user to think aloud
- A combination of the above

To have test personnel observing the participants or recording them with a video camera, is usually only possible if the number of participants is small. This is because watching each user directly or reviewing tapes of the session is time consuming. However, this approach will usually result in more detailed data than interviews and questionnaires. When the user is observed continuously either directly or via a video camera, the designers of the application can see every action the user performs. This will obviously result in more detailed information than if the users only were asked some questions after the test session. When just using interviews and questionnaires to gather the data, important information can be lost. For example, the questions asked might not touch all the relevant areas. Also, the user could run into some problems during the test that he forgets to inform the test personnel about. It is also possible for misunderstandings to occur if the user does not know how to explain a particular problem. Continuous observation directly, or via a video camera, will capture this information. However, interviews and questionnaires also have their strengths because they can uncover the user's thoughts. This is not easy to catch during observation or video recording. To get the most of a user test, a combination of observations and interviews should be used.

### **2.3.2 Heuristic Evaluation**

Heuristic evaluation is a cheap and simple method for uncovering usability problems in an application. A description of how to conduct a heuristic evaluation is shown in [31]. When performing a heuristic evaluation, a small number of evaluators are used. These evaluators will examine the user interface and compare it with recognized usability principles, called the heuristics. An example of a collections of heuristics is shown in [3].



It's not recommended that the evaluation is performed by more than one person. The reason being that it's difficult for one person to uncover all the usability problems in an application. The number of people evaluating an application during a heuristic evaluation, is recommended to be somewhere between three and five people. Research has shown that different people find different problems with an applications user interface [31]. This means that multiple evaluators will increase the effectiveness of the evaluation process. However, it is not recommended to use more than five evaluators. Because the number of usability problems found are not proportional to the number of evaluators.

A heuristic evaluation is performed by having each evaluator use the application alone. During the evaluation, he or she compares the user interfaces to the heuristics. The evaluator takes notes of other issues he finds problematic or difficult to understand. Commonly the evaluator is asked to go through the user interface at least twice. The first time is for getting a feel for the design and the overall flow of the application. On the next iteration, the evaluator is asked to go through the interface more thoroughly and point out any problems encountered. When all the evaluators have completed the evaluation, they are allowed to communicate and possibly aggregate their findings. The evaluators are not allowed to communicate before they all have completed the evaluation. The reason for this is make sure the feedback from each evaluator is unbiased and independent.

There are mainly two different ways of recording the evaluation. The results can be recorded as a report written by the evaluator or the evaluators think aloud during the evaluation and an observer records the information.

The way heuristic evaluation is described in this section, it can seem similar to a standard user test. This is not the case. In a user test, it is the responsibility of the test personnel to somehow record the actions performed by the users. The actions recorded have to be interpreted into issues about the usability and design of the application being tested. Since it is up to the observer to transform user actions into usability issues, it is possible to perform the test even if the users know nothing about interface design. In heuristic evaluation, this responsibility is in the hands of the evaluators. This means that the person evaluating the application must have knowledge and experience about interface and usability design. The heuristic evaluation reduces the workload for the observer as opposed to user tests. Since the observer only needs to record the evaluators comments. He does not have to map different user actions into usability issues.

The output of a heuristic evaluation is a list of usability problems related to the application being tested. Each of the usability issues found in the evaluation must be referenced to one or more of the heuristics. It is also important a reason is given for why each issue is deemed problematic. The reasons should be as specific and detailed as possible. As an extension of the evaluation, the evaluators can propose solutions for each of the issues pointed out.

There are two main advantages in doing a heuristic evaluation as opposed to user testing. First, the evaluators are not using the system to perform a task such as when the test user has a scenario he has to complete. A result of this, is that it is possible to perform a heuristic evaluation on simple mock-ups or even on systems that only exist on paper. This makes heuristic evaluation a good tool for uncovering errors or weaknesses in the usability design early in the development process. This is important because it is easier to fix problems at an early stage. Second, heuristic evaluation is intended as a

"discount usability engineering" method [31]. Which means that it provides a fast and simple evaluation of the usability of an application.

Heuristic evaluation can also be used as a first step in an evaluation process. When the issues pointed out in the heuristic evaluation are fixed, a user test can be performed. This way, a lot of design issues can be fixed before a user test is initiated.

### 2.3.3 Cognitive Walkthrough

Cognitive walkthrough is another method for evaluating the usability of an application. The common way to measure usability is by ease of use. Cognitive walkthrough has a different focus. It focuses on ease of learning by exploration. Ideally the application should be user friendly enough so that a user who never has used it before can use it without any training or introduction. The user interface should inform the user about what actions are available and what the user needs to do in order to successfully complete the task.

The following list gives an overview of the cognitive walkthrough process (from [57]):

1. Define inputs to the walkthrough
  - Identification of the users
  - Sample task evaluation
  - Action sequences for completing the tasks
  - Description or implementation of the interface
2. Convene the analysts
3. Walk through the action sequence for each task
  - Tell a credible story
  - Will the user try to achieve the right effect?
  - Will the user notice that the correct action is available?
  - Will the user associate the correct action with the effect that the user is trying to achieve?
  - If the correct action is performed, will the user see that progress is being made toward solution of the task?
4. Record critical information
  - User knowledge requirements
  - Assumptions about the user population
  - Notes about side issues and design changes
  - The credible success story
5. Revise the interface to fix the problems.

A cognitive walkthrough can be performed on different stages in the development process. The approach can be applied to everything from a paper mock-up to a working prototype. As shown in (1) in the list above, there are four factors that are input to the cognitive walkthrough process.

Identification of the users is an important input parameter. The reason for this is that

this is crucial for the result of the walkthrough. An application that is designed for use by novice users with little or no prior computer experience, the usability demands are higher than if the application is designed for computer experts.

When performing a cognitive walkthrough, user scenarios are the core of the process. These scenarios describe a sequence of actions a user has to do in order to successfully complete the task. Before a cognitive walkthrough process is initiated, different scenarios have to be developed. As stated in [15], this can be the hardest step of the walkthrough process. Developing good case scenarios that will uncover as many usability problems as possible is not easy.

The main goal of a cognitive walkthrough is to uncover errors that interfere with the philosophy of "learning by exploration". A walkthrough will also find mismatches between users' and designers' conceptualization of a task, poor choices of label names, menu titles, buttons and error messages. Inadequate feedback about an action and its consequences will also be uncovered by the walkthrough.

For each scenario, a credible success story has to be drafted. This story should explain why it's expected that users would be able to perform the action. For a credible success story to be fulfilled, all four of the questions above needs to be successfully completed. If the analysts find a problem with one of the questions, we have a failure story.

A cognitive walkthrough can be performed either by a single person or by a group. However to uncover the largest amount of usability problems, it's recommended that the walkthrough be performed by a group [57]. Preferably the group should consist of people with different backgrounds. For example one with knowledge of cognitive psychology, one with knowledge about the domain the application is being developed for (e.g accountants), one user interface expert, one system developer etc. The reason for this is that people with different backgrounds think differently and might therefor uncover different usability problems.

For each scenario, it should be noted what the user must know before performing the task, and what the user should learn from the interface while performing the task.

The cognitive walkthrough method is a good method for evaluating user interfaces. However, there are a couple of drawbacks to the method. For example, the method is focused on one single aspect of usability, namely easy of learning. This means that the scope of the method is quite narrow. A common problem with the cognitive walkthrough is that it only focuses on specific problems. General problems will not be found. This means that if only cognitive walkthrough is used for usability evaluation, problems that concern the entire application will not be found. Instead a single general problem would probably be recorded as many small specific problems. A result of this could be that the designers try to fix each of the small problems separately instead of fixing one general problem. This is also the reason why it's recommended to use a combination of multiple usability inspection methods.



## 3 Methods

There are three issues this thesis aims to uncover. First, a technical feasibility study to prove that it is possible to implement the proposed SSO concept. A detailed description of the solution is included in the next chapter.

The second issue is to conduct a security analysis to identify what security measures need to be implemented to make the system secure. Different methods for conducting such an analysis are presented in chapter 5 together with the security analysis.

This chapter will focus on the third issue, usability testing. The following sections will discuss how the user test will be conducted.

### 3.1 Methodology discussion

To answer the questions above, a usability test will be conducted. In section 2.3 three methods for conducting usability tests are presented. To test the SSO solution developed in this thesis, user tests are chosen as the preferred method. This method was chosen because it's likely to gather the most relevant data for this case. The two other methods mentioned were heuristic evaluation and cognitive walkthrough. These methods are used for finding problems with a particular user interface, such as confusing labels on soft keys etc. This kind of information is not relevant in this thesis because we only want to find what the users think of this new single sign-on concept. How the current interface looks and feels is irrelevant because it will most likely be changed in later versions.

Usability testing is also important in terms of security. In Saltzer and Schroeder's paper "The Protection of Information in Computer Systems", [43] eight design principles are stated. The final design principles is:

Psychological acceptability: It is essential that the human interface be designed for ease of use, so that users routinely and automatically apply the protection mechanisms correctly. Also, to the extent that the user's mental image of his protection goals matches the mechanisms he must use, mistakes will be minimized. If he must translate his image of his protection needs into a radically different specification language, he will make errors.

This principle is especially important in this case because users must find the system easy and convenient to use. If not, they won't use it, even if it provides added security and enhanced password management.

When it comes to recording data from the user test, a combination of surveys and interviews is chosen. Originally it was also planned to videotape the participants. For example [56] recommends a user test where six participants are chosen to be in the test. Each participant is video taped while he or she uses the application. However, a staff member at GUC who is an expert on human - computer interaction did not advise this. The reason is that such a test will also focus on problems with a particular interface. The video tape will show what kind of problems to participants have with the interface they are using. As described above, this information is not relevant for the experiment in this thesis.

The best way to obtain this information is by letting each user test the prototype for a while, and when he or she is done, ask questions about how they felt about the concept and if they would use this system on a daily basis.

Each user to participate in the test will be asked to complete a specific scenario. This scenario is described in detail in section 3.2. By giving the users a scenario they have to complete, there will be a good structure to the test. Because every one will go through the same tasks giving them as equal conditions as possible.

The number of users to participate in the test is also an important issue to consider. Somewhere between 25 and 30 users would be ideal. A larger number of users will of course provide more data, but it is important to find a compromise between the amount of data acquired and the time spent on user tests. Each user test will take about 20 minutes to go through. A test with 25-30 people should be possible to complete within two working days. Another important issue is to choose participants with different backgrounds. The group of people chosen to participate in the test should be as representative for the expected target group as possible. 20 to 30 people is not a large number, so the selection is not large enough to say that is totally representative for such a large target group as this prototype is expected to have. Since this master thesis has to be completed within reasonably short time limits, there will simply not be enough time to conduct a large scale user test. Still a selection of 20 to 30 people will be a good trade off between the accuracy of the data and the amount of time spent on usability testing.

## 3.2 Test protocol

The participants will be called in to test the SSO solution one at the time. Before the test begins, the users will be given a short introduction to single sign-on in general and the purpose of this system. The users will also be given an introduction on how to use the w550i mobile phone if necessary. As mentioned above, we want to find out what the users think of the single sign-on concept and not specific issues in this particular application. To start the MIDlet, the user is required to authenticate. In this prototype, the authentication is achieved by using a master password the user has to type in. Typing passwords on a mobile phone is a slow and frustrating task and this authentication scheme will probably be replaced with a scheme more adapted for use on a mobile phone in later versions of the MIDlet<sup>1</sup>. If the user gets frustrated by a temporary login function that is likely to be replaced, the results might be negatively biased. To prevent this from happening, the authentication function of the application is disabled.

### 3.2.1 Scenario

When the initial introduction is complete, the users are asked to complete the scenario presented to them. A copy of the scenario is printed on a sheet of paper and made available to the users. During the test, the users will be instructed to follow this scenario. Also we will be present in the room to aid the participants if they encounter any problems during the test.

In normal usability studies where you test a specific interface for usability problems, the observer would not step in to help if the participants encounters problems. This is because the observer wants to see where the user gets stuck and what she does to get

---

<sup>1</sup> Tommy Egeberg is writing a master thesis this year about security on the mobile phone. His thesis will probably include different authentication schemes suited for mobile devices

unstuck. The experiment conducted in this thesis only aims to test what the users think about this new SSO concept, and not how they feel about a details in the interface. Therefore assistance will be provided if they get stuck. However, the user will not get any help before he or she really is stuck.

The accounts the participants are asked to use during the test are made up especially for this occasion for two different reasons. First, the solution developed in this thesis is only a functional prototype, i.e. no security measures are implemented. For example the passwords are stored on the mobile phone in clear text. Because of this, it would be an unnecessary risk for the users to provide their real usernames and passwords. The second reason for using dummy access credentials is for making the experiment more effective. It would take a lot of time if each participant had to enter several of their own usernames and passwords before the test could begin. Also the access credentials would have to be deleted before the next participant uses the system. Otherwise he or she would be able to see the usernames and passwords of the previous participants. Also each of the participants are using different services. If their real access credentials was to be used, it would not be possible to create a scenario every participant could follow.

When the MIDlet is run for the first time, a search for available Bluetooth devices is conducted. The user selects the device he or she wants to use. The address of this device is written to the RMS. Next time the user starts the application, the address will be read from the RMS and no Bluetooth device search is required. This will reduce the time required for each user test.

The scenario presented to the user is meant to be as realistic as possible. That is, the scenario should cover common tasks a user is expected to perform during a normal day. At the same time, it is important to keep the test short, due to time constraints. In order to meet both these requirements, a scenario where the user logs on to several accounts to check status have been designed. This could be something a user does routinely every day or even several times a day. Also, the fact that the users are using a working prototype of the system, will help increase the feeling of realism.

In the introduction to the scenario, the user is presented with a description of its setting. The setting describes that the user is traveling. While on this journey, the user wants to check the status of some user accounts. To do this, the user walks into an Internet café. The computer he or she sits down by have never been used by this user before. This means that no usernames or passwords are stored on the computer. Also the user does not remember any of his or hers passwords. However, he or she has are using the SSO solution proposed in this thesis. This means that the user has both the mobile phone with the SSO MIDlet and the Bluetooth/USB device in his or hers pocket<sup>2</sup>.

When the participant sits down by the computer, he or she is presented with the Windows XP login form. The user is asked to log on to the PC using the MIDlet on the mobile phone. Next the user is asked log on to Hotmail and check for new e-mails. After he or she is finished checking their e-mail, they are asked to visit an Internet auction where there are posted some items for sale under their user account. The participant is asked to log on to the auction and check status on each item, such as the value of the highest bid.

---

<sup>2</sup> The prototype of the Bluetooth/USB device used in this thesis is large and not possible to carry in a pocket. However, the participant is explained how this device will look in later versions, and that he or she should not take the size of the device into consideration during the evaluation.

In addition to testing how the participants like to log in using this SSO solution, we wanted to test how they felt about adding a new account to the MIDlet. Typing on a mobile phone is a time consuming and cumbersome task<sup>3</sup>. There are two reasons for testing how the users like to enter usernames and passwords on a mobile phone. First, we wanted to find out if the users like the process of storing access credentials to the MIDlet. Second, we wanted to find out if the participants have any suggestions on how to improve this feature so that it is more user friendly and effective.

To verify that the participant has entered the new access credentials correctly, he or she is asked to log on to a forum and check if there are any new replies to posted topics posted by the user. Finally, the participant is asked to delete the account he or she just added. This is because we wanted to test this feature as well. A copy of the user scenario as presented to the user is included in appendix C<sup>4</sup>.

### 3.2.2 Debriefing

After the user has completed the scenario, a debriefing will be conducted. The purpose of the debriefing is to get feedback from the participant on what he felt about the system. This debriefing will be conducted in the form of a survey.

The survey has two purposes. First, it will help gather information about the users background. This includes generic background information such as age, gender, education level etc. Also, the persons occupation will provide information about his or hers technical background. The prototype developed in this thesis have a very broad target group. This means that everyone from novice users with no prior computer experience to computer experts should find the system useful. The background information can help determine what kind of people liked the system best.

Additionally, it can be useful to know how much experience the user has with mobile phones. To find this out, the participant will be asked if he or she owns a mobile phone and how often he or she uses different services such as SMS, MMS, taking picture, running programs, accessing the Internet etc.

The users will also be asked if they have heard of the term single sign-on before and if they have used any other single sign-on solutions. If they have, a follow up question about what they thought of the single sign-on solution developed in this thesis as opposed to the other SSO solutions will be asked.

The survey will also gather information on how many usernames and password the user has to manage now, and how he or she manages them. Are they written down? Is the same password used on several different accounts?

Another important piece of information is the users perception of a secure password. The users will be asked if they believe their own passwords are secure. This background information is important to gather because it will provide statistics on how passwords are handled today.

Next the participant will be asked what he or she felt about the single sign-on concept they just tested.

In addition to asking what they felt about the concept as a whole, they will be asked what they felt about the two main components separately. First they will be asked how

---

<sup>3</sup> Several usability tests conducted on mobile devices conclude that users don't like the input mechanism on mobile phones. In the book *Handheld usability* [56], it is stated that user generally like to scroll and select and not type on a mobile phone.

<sup>4</sup> Since all the participants are Norwegians, the survey is also written in Norwegian



they liked the login functionality. Second, they will be asked the same questions about the password management part of the system, i.e. entering a new sets of username and password on the mobile device. Also they were asked how they felt about deleting accounts from the device.

The participant will be asked if he or she would use this type of SSO solution on a daily basis.

Toward the end of the survey, the participant is asked to give a rating of some statements. The rating has five values ranging from totally disagree to totally agree. These statements ask the user what he or she thinks about the security aspect of this solution, if he or she is willing to trust a mobile device with his or hers usernames and passwords. There will also be a question about whether or not the user would like a backup solution in case the mobile device is lost or stolen. In addition the user will be asked if he thinks this system will make his or hers password management more secure.

Finally the user is asked how much he or she would be willing to pay for this solution if it was commercially available. When conducting a user test and a number of potential user of the system are available, it's a good idea to take advantage of this to perform some marketing research. If this SSO solution is to be further developed and perhaps also commercialized, it's important to find out what the consumers are willing to pay for the system. To answer this, a small price survey was included.

A few of the participants (about 5 to 10) will also be selected for a more in depth interview. This is an open interview where the participant can speak freely about what he or she felt about the SSO solution. Any suggestions on how to improve the system, both in terms on functionality and security, will be greatly appreciated.

The analysis conducted will produce both qualitative and quantitative results. Quantitative results from the survey and qualitative results from the interview with the chosen participants.

The survey used in the debriefing is included in Appendix D<sup>5</sup>. Results from the usability test are discussed in chapter 6.

---

<sup>5</sup> Since all the participants are Norwegians, the survey is also written in Norwegian.



## 4 Prototype design

This chapter describes the requirements for the prototype. Also the hardware and software used to developed the solution are described. Finally a description on how the Bluetooth/USB device and the MIDlet on the mobile phone work.

### 4.1 Requirements

The main purpose of this prototype is to make password management easier and more comfortable for the user. Hopefully, the user will have a better overall computer experience by using this prototype since he can focus on user tasks and not spend time on authentication tasks. In addition this prototype will hopefully introduce a more secure way of handling passwords since the users are no longer required to remember them.

In order to offer the user a more comfortable and secure way of managing password, the following requirements have to be met:

- The system has to be mobile, i.e it should be easy to use on different computers and small enough for the user to carry at all times
- The system should be more than a just a digital safe that stores your access credentials. It should automate the login procedure so that the user doesn't need to type anything.
- The system should be simple, intuitive and effective to use.
- The system should be secure enough for the users to trust it with their passwords

If the system is going to be easy to use on different computers, the installation procedure has to be simple. To supply drivers on a CD that the user has to install on every computer before he uses the system, will be much too cumbersome. Also, this could limit the number of computers where the system can be used. Because the user might not have privileges to install software on every computer he uses. For example at an Internet café or the library.

To meet this requirement, it was decided to use a USB device and a mobile phone. The USB device will identify itself as a generic desktop keyboard. This means that when the device is plugged in, the operating system will think that a conventional keyboard is plugged in and a generic keyboard driver will be loaded automatically. A design like this will work on every operating system that supports USB, i.e the same device will work on Windows, Mac OS, Linux etc without any modifications. Also the design excludes the need for an installation CD.

The access credentials (user name and password) will be stored on a Bluetooth enabled mobile device, such as a mobile phone or a PDA. Since most people carry their mobile phone at all times, the requirement about mobility is partly fulfilled. The next challenge is to make the USB device small enough so that the user can carry it with him without any trouble. An ideal solution would be if the device could fit on a key ring. As described in section 4.2.1 Nislab is developing a Bluetooth enabled USB device which is about the same size as a USB pen drive. However, this device is not ready yet and



Figure 2: Shows how the hardware components in the prototype are connected.

replacement hardware had to be used in order to implement the solution. This hardware is quite large, and not very mobile. Because of this, the mobility requirement for the USB device is not met for the prototype developed in this thesis. However once the NISlab device is ready, the requirement will be met.

Automated logins, is also very important. If this is not implemented, the system will only act as a digital safe where the users access credentials are stored. This would mean that the user will have to look up the username and password for an account, and manually enter them into the login form. It's obvious that this approach is not very user friendly. By configuring the USB device as a keyboard, logins can be automated. The only thing required by the user is to place the cursor in the username field on the screen. Once he chooses an account on the mobile device, the credentials are sent to the PC via the USB device as emulated keystrokes.

If this system is to be widely used by a large number of users, it has to be user friendly. It's important that the system is simple enough so that novice users understand it intuitively. Otherwise it is unlikely that the system will be used even if it provides a better and more secure password management scheme.

The overall design of the prototype is shown in figure 2. The username and password is sent wirelessly over Bluetooth to the f2m serial port plug as key codes. Data received at the f2m device is sent to the FS USB board through a nullmodem adaptor. This is needed in order to cross the TX (Transmit) and RX (Receive) signals because both the f2m and the FS USB device have female RS232 connectors. The FS USB board is connected to the PC with a standard USB cable.

## 4.2 Choice of technologies

### 4.2.1 Hardware

As mentioned in the previous section, the prototype was supposed to be implemented on a device developed at Nislab. However during fall 2005 problems with the hardware on this device were discovered. Because of this, a backup solution had to be created. My supervisor Einar Snekkenes came up with a solution that requires some extra hardware components, but should preserve the basics of the proposed concept. A description of the hardware used in this prototypes follows below.

#### *PICDEM FS USB Demo board*

My main hardware device is the PICDEM FS USB Demo board [26] from Microchip shown in figure 3<sup>1</sup>. This is a demonstration board for the PIC18F4550 family with support for full speed USB and RS232 serial communication. Some of the technical specifications of

<sup>1</sup> [www.microchip.com](http://www.microchip.com)

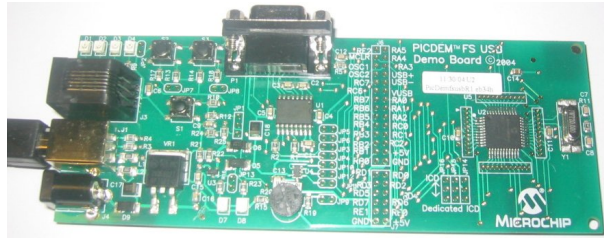


Figure 3: The PICDEM FS USB Demonstration board.

this card are<sup>2</sup>:

- 48Mhz operating speed
- 32 Kb of Enhanced Flash memory
- 2 Kb of of RAM
- 256 bytes of data EEPROM
- Full speed USB 2.0 interface
- 20 MHz crystal
- Serial port connector
- Connection to the MPLAB ICD 2 in circuit debugger
- 2 LEDs for status display

The microchip on this board is the same one as on the device developed at NISlab. This means that the same code can be used on both these devices. The demo board is connected to the computer via an USB cable. When the card is plugged in, it will identify itself as a normal USB keyboard. At the back-end it will receive data (user name and password) through the serial port which is connected to the "Bluetooth serial port plug" - F2M01 <sup>3</sup>.

#### ***Bluetooth serial port plug - F2M01***

The Bluetooth serial port plug - F2M01 is a Bluetooth device with a RS232 interface. A complete data sheet is found in [8]. This device will communicate with the mobile phone, and receive the usernames and passwords a the Bluetooth channel. The username/password data will be received in the form of key codes. In section 4.3.1 a description of key codes are and how mapping between ASCII characters and key codes works is described. The key code data is read from the USART input buffer, and sent to the PC via the USB interface as an emulated keystroke.

#### ***Mobile device***

The MIDlet developed in this thesis have to be deployed on a mobile device to work. It can be installed on any device that supports Bluetooth, Java MIDP and the Java Bluetooth API JSR-82 (see 4.2.2). The phone used for this prototype is the SonyEricsson w550i.

<sup>2</sup>A complete list of all the specifications are found at the Microchip website.

<sup>3</sup> www.free2move.se

### 4.2.2 Software

To develop this prototype, two different programming languages have been used. The C18 compiler v3.02<sup>4</sup> together with the MPLAB IDE v7.30 was used to program the microchip. The MPLAB IDE is a free development environment for employing Microchip's micro controllers. It is a windows based program with a users friendly interface and contains a lot of different tools for helping to speed up the development process. The IDE also provides support for several hardware debuggers and programmers, such as the MPLAB ICD 2. MPLAB IDE is originally a tool for writing assembly code, but the IDE supports the integration of the C18 compiler. This is a C compiler optimized for PIC18 series micro controllers. Once it's installed, it's possible to program standard ANSI C in the MPLAB IDE. In [27], a complete reference of all library functions for the C18 compiler is given.

The Java MIDlet has been developed using NetBeans IDE 5.9. This is a free cross platform visual development environment for making applications for the Java 2 platform. In addition to the editor for writing Java code, the following software packages are needed for this prototype<sup>5</sup>:

- Java 2 Standard Edition (J2SE) v1.4.2(or greater) SDK <sup>6</sup>, the basic development kit for the Java 2 platform. This is needed for running the Java compiler (javac) and for creating Java archive files (JAR) and Javadoc files.
- Connected Limited Device Configuration (CLDC) v1.1. This package contains a virtual machine which is optimized for running on devices with limited memory and processing capacity (the K virtual machine - kvm). The package also contains a set of Java libraries (classes). The number of classes in the CLDC are somewhat less than in the J2SE platform, due to constraints in storage space on mobile devices.
- Mobile Information Device Profile (MIDP) v2.0 is the most popular Java runtime environment for mobile devices today. This profile is placed on top of the CLDC as shown in figure 4. This means that MIDP application has access to the libraries of both CLDC and MIDP.
- J2ME Wireless Toolkit. This is a toolkit designed to help create and manage J2ME projects. It also provides emulators for different mobile devices.
- Bluetooth API JSR-82. This API consists of two independent packages; the Bluetooth API and the Object Exchange (OBEX). The JSR-82 exposes the Bluetooth software stack to developers working on the Java platform. Usage of the JSR-82 is described in [35] and [22].
- NetBeans mobility pack. Needed in order to make the Bluetooth API work with NetBeans.

---

<sup>4</sup> [http://microchip.com/stellent/idcplg?IdcService=SS\\_GET\\_PAGE&nodeId=1475&category=devSoftware](http://microchip.com/stellent/idcplg?IdcService=SS_GET_PAGE&nodeId=1475&category=devSoftware)

<sup>5</sup> J2SE SDK is found at <http://java.sun.com/j2se/1.4.2/download.html>. The other packages are found at <http://java.sun.com/j2me>

<sup>6</sup> <http://java.sun.com/j2se/1.4.2/download.html>

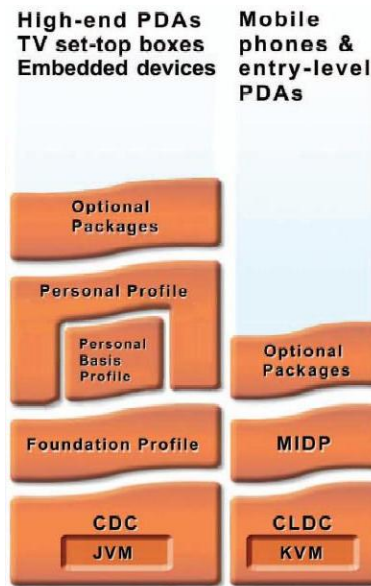


Figure 4: Java 2 Platform, Micro Edition (J2ME). This figure is a modified version of the one found in [50]

### 4.3 Design of the SSO prototype

In the following sections, the design of the prototype will be discussed. From a users point of view, there are several actions he or she wants to perform on the system. The main actions a user wants to perform are shown in the overall use case diagram in figure 5.

The single sign-on solution developed in this thesis consists of two design phases. The design of these phases will be described separately in the next two sections. How the works together as a whole is shown in figure 6.

#### 4.3.1 Design of the Bluetooth/USB device

The Bluetooth/USB device acts as a bridge between the mobile phone and the PC. It will receive data over the Bluetooth interface and send this data through the USB port to the PC as an emulated keystroke. The first thing a user needs to do to start using the single sign-on system is to plug the Bluetooth/USB device into the USB port on the PC. The devices is among other things equipped with two LED lights. These lights are used to inform the user of the devices state. When the two LED's starts blinking alternately, the USB device is configured properly and ready to receive data.

Any USB device connected to a PC needs to go through an enumeration process<sup>7</sup> During the enumeration process the demo board will tell the PC that it is a device in the major class Human Interface Device (HID) and the minor class Generic Desktop Keyboard. Also the device will inform the PC on how the format of the reports will be. Reports are the packets of data the USB device sends to the PC. For the communication between the PC and USB device to work properly, the operating system must know the

<sup>7</sup>This is a process where the USB devices tells the PC what type of a device it is. If the device belongs to any of the predefined device classes, the computers operating system will automatically load the correct driver. The enumeration process is described in detail in [11].

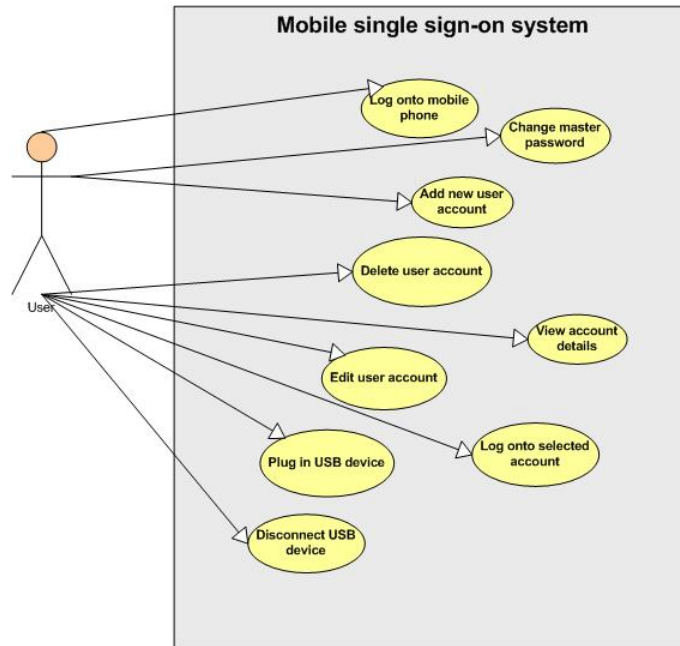


Figure 5: Overall use case for the prototype

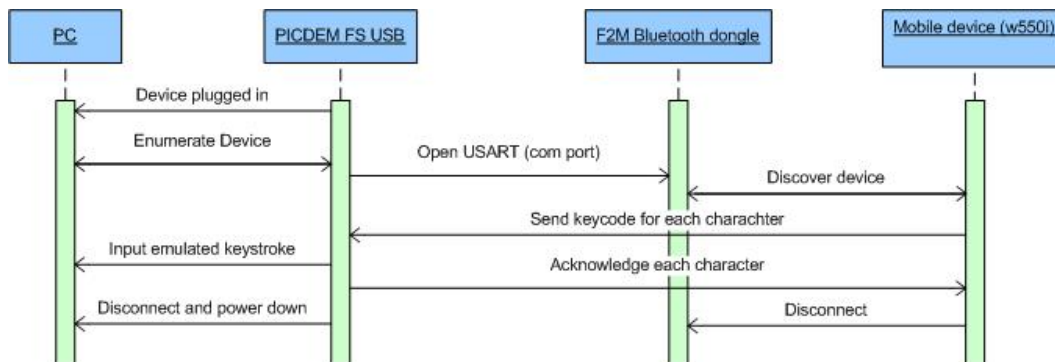


Figure 6: Sequence diagram for the prototype

length of the report being sent. This information is passed to the PC in the form of a device descriptor. The descriptors used to configure the device as a keyboard are included in appendix B. According to the HID specification [52] the recommended length of a report containing keystrokes is eight bytes. The report has the following format [52]:

- 0 Modifier Keys
- 1 Reserved
- 2 Keycode 1
- 3 Keycode 2
- 4 Keycode 3
- 5 Keycode 4
- 6 Keycode 5
- 7 Keycode 6

In the first of element of this eight byte report the modifier keys are stored. Modifier



keys are keys such as CTRL, ALT SHIFT etc. Each of these modifier bits are assigned a bit value<sup>8</sup>. If no modifier keys are pressed, this field is set to 0. If one modifier key is pressed, the bit value will be stored in the modifier keys field. However, if several modifier keys are pressed simultaneously, for example CTRL and ALT, the two bit values are ORed together and the result is stored in the modifier keys field.

Field 2 to 7 (field 1 is not used) can contain 6 different keystrokes. The report supports more than one simultaneous keystrokes. This is because a user is allowed to press several keys at the same time, something which is common in games. In the prototype developed in this thesis, only one keystroke will be sent in each report. This is because the input should work just as if a user was typing the characters directly on a keyboard. And when a user types, he presses one key, releases it and then presses another key. However some characters can only be displayed by pressing multiple keys at the same time. For example to type the } character on a Norwegian keyboard, the user has to press CTRL, ALT and the 0-key simultaneously. To emulate this character, the bit values of CTRL and ALT have to be OR-ed together and added to the modifier keys field. Additionally the Keycode 1 field will contain the keycode of the 0-key. The fields not in use will have the value 0. This means that if for example the character 'e' is to be sent to the PC, all the fields would be set to zero except the Keycode 1 field that would contain the keycode for 'e'. Also it is important to send an empty report (with all fields set to zero) to the PC after each character is sent to indicate a key up event. If this key up report is not sent, the PC will think that the character is still pressed and display it to the screen continuously. It will not stop until it receives a report telling it that no keys are pressed.

Mapping between an ASCII character and a keycode is done on the MIDlet. How this mapping is performed is described in the next section.

The functionality of the USB device is actually quite simple. Once the device is correctly enumerated the board sets up the USART interface. This interface communicates with the RS232 serial COM port. The f2m device described in section 4.2 is connected to the this port. As described in the next section, the MIDlets sends data over the serial port profile in Bluetooth. This is a protocol that emulated RS232 communication over Bluetooth. Which means that the Bluetooth communication is transparent to the USB board. The f2m device handles everything that has to do with Bluetooth. The data forwarded to the USB board over the RS232 interface looks just like any other serial communication traffic. This means that the USB demo board does not see any difference between this emulated serial traffic and normal serial traffic passing through a standard RS232 cable. When this interface is set up, the card registers an interrupt function which is set up to listen on the RX (receive) pin of the RS232 interface. Each time a byte is received on the RX pin, the interrupt function is evoked. This function reads the byte and stores it in a globally available circular buffer. The MIDlet sends one byte at the time. When the ring buffer contains 8 bytes or more, another function is invoked. This function reads 8 bytes from the buffer and places them in the input report. When the report is ready, it is sent to the PC, and the character is displayed on the screen. This process will continue until the buffer is empty and all the characters are displayed.

---

<sup>8</sup>For a complete overview of all modifier keys see [52] page 56.

### 4.3.2 Design of the MIDlet

The MIDlet is responsible for most of the functionality in this prototype. All the user interaction is handled by the MIDlet. It is also the MIDlet's responsibility to manage the users different usernames and passwords. This includes creating, storing, altering and deleting these access credentials.

When the MIDlet is started up, the user is prompted to enter the master password. If the users enters the correct password he or she is allowed access to the MIDlet. The implementation of the authentication procedure in this MIDlet is quite simple. This is because Tommy Egeberg is doing a master thesis on the security in the mobile phones [6]. In his thesis he will thoroughly evaluate different authentication methods for mobile devices. When his thesis is complete, he will probably recommend an authentication procedure which is a good compromise between security and usability that works well on mobile devices. The simple password authentication procedure implemented in the prototype is only a temporary solution and will most likely be replaced with the authentication procedure recommended in Tommy's thesis.

Once the user has successfully authenticated, he is presented with the MIDlet's main menu as shown in figure 7 a)<sup>9</sup>. From this menu, he has the possibility of choosing an existing account or creating a new account. Additionally he can change the master password, change the MIDlets settings and view information about the MIDlet.

#### *Adding a new account*

First let's start with a user who wants to add a new account to the MIDlet installed on his mobile phone. From the main menu she chooses the new account command, as highlighted in figure 7 a). When this action is selected, a wizard for creating a new account is started. The wizard consist of three steps. First the account name is entered, figure 7 b). This is the name that will be displayed in the list of stored accounts, figure 7 e). Second, the user is asked to enter the username for the account, figure 7 c). Finally she is asked to enter the password, figure 7 d). When these three values have successfully been added, the account is created and stored in a record store in the devices memory (the record store management RMS).

#### *Logging in*

When the user wants to log in with the account she just created. To do this, all she has to do is to select the top choice of the main menu in figure 7 a). A list of all accounts in the record store (RMS) will be displayed as seen in figure 7 e). The accounts are sorted alphabetically on the name of the account name.

If the user wants to log in to an account, all she has to do is to select the account from the menu (for example Hotmail) and hit the select button on the mobile phone. When an account is selected to log in to, the MIDlet checks if this is the first time a login is performed. If that is the case, a search for Bluetooth devices is conducted. In order to make this as easy as possible for the user, the MIDlet will first search for devices with the friendly name "sso device". If a device with this friendly name is found, the MIDlets recommends that this device is used. The friendly name "sso device" is the name that the Bluetooth module on the USB device uses as default. A user can change the default

---

<sup>9</sup> The MIDlet is implemented in norwegian in order to avoid any language confusion during the test. This is because the usability is performed on participants that have norwegian as their first language. If the MIDlet was in english, and some participants didn't know the english language, it could bias the test results negatively.

friendly name, but in order to do this, she has to install the configuration software from f2m and write a new configuration to the chip. This is probably too cumbersome for the average user.

However, if no devices with this name are found the user is presented with a list of other accessible Bluetooth units in the area. This list is also displayed if the user does not want to use the Bluetooth device suggested.

Bluetooth device and service search usually takes quite a bit of time. When a device search and service search is complete, a Bluetooth URL<sup>10</sup> is returned. The same URL is used each time someone wants to connect to the device. In order to make the login procedure as fast as possible for the user, this device search will only be performed once. When the users selects the Bluetooth device she wants to use, the Bluetooth URL is stored in a record store in the phones memory. The next time a user wants to log in, the MIDlet will read the connection URL from the record store and immediately set up a connection with the device. No device search have to be conducted this time.

Once the connection between the two devices is established, the MIDlet will fetch the username and password from the record store and send it to the Bluetooth/USB device over the encrypted Bluetooth connection.

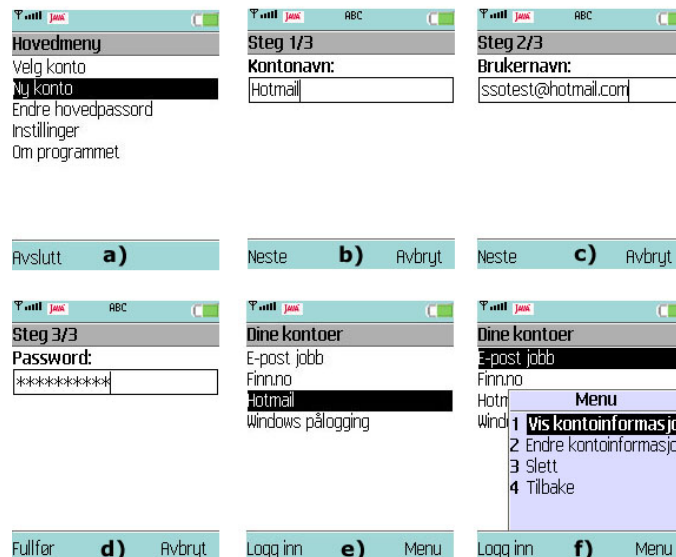


Figure 7: a) The main menu of the MIDlet. b) Adding a new user account step 1. c) Adding a new user account step 2. d) Adding a new user account step 3. e) Displaying stored accounts. f) List of options that can be applied to the accounts.

### Mapping between characters and keycodes

As described in section 4.3.1 each character in the username and password are sent to the Bluetooth/USB device as an 8 byte input report. A complete table of the mappings

<sup>10</sup> The Bluetooth URL is what is needed to connect to Bluetooth device. The URL can for example look something like: `btsp://0123456789AF:1;master=false;encrypt=true;authenticate=true[22]`. `btsp` means that the serial port profile is to be used as the communication protocol. This is a protocol which emulates serial communication (RS232). `0123456789AF` is the address of the Bluetooth device. `1` is the channel number to be used. The parameters following gives information on how the connection should be set up. In this example, the devices must successfully authenticate before the encrypted connection is set up.

between ASCII characters and their corresponding keycodes are found in the HID usage tables document chapter 10 [53]. The mapping between ASCII and keycodes could also be done on the USB device, but this will not be as effective as doing it on the MIDlet. The USB device has limited processing power and very small amount of internal memory available. Because of this, it makes more sense to do the mapping on the device with the most resources available.

An important issue is the format used to send data sent from the mobile phone. The USB device connected in to the PC is configured as a standard keyboard. This means that each character in the username and password is sent from the USB device to the PC as an emulated keystroke using a look up table. The index of the table is the ASCII code of the character. For example if the character 'a' is in the username, the MIDlet will find the ASCII value of 'a' (97). Next it will look up element 97 in the table and get the corresponding keycode (4). One important thing to note about keycodes is that keycodes are mapped to a specific key on the keyboard and not a particular character. For example will 'a' and 'A' have the same keycode. It is the "Modifier keys" field, described above, that makes it possible for the PC to determine if it is to display an 'a' or an 'A' character. If an 'A' is to be displayed, the "Modifier keys" field contains the value of SHIFT indicating that SHIFT and 'a' are pressed simultaneously displaying 'A' on the screen.

One problem with the mapping between ASCII characters and keycodes is that the mapping will only work for a specific keyboard layout (norwegian in this case). Say for example that a user has a password which contains a special character, such as '?'. In the look up table, the ASCII code would map to SHIFT and the keycode 45. However if the operating system is configured for the English (US) keyboard layout the character '\_' will be displayed.

### ***Account options***

There are several options available for each account. The user can view a list of the options by selecting an account in the list and pressing the menu button as shown in figure 7 f). The user has three options to choose from, in addition to going back to the main menu. First she can view the account information, i.e. she can see her username and password in cleartext. This feature can be useful if the user has forgotten the USB device at home and needs to log in somewhere. Since this SSO system stores the users password, there is no need for the user to remember the passwords. This means that if the user has forgotten the USB device, she will not be able to log in manually because she probably does not remember her password. To prevent this from happening, the feature of displaying the username and password is implemented. so that she simply can look up the access credentials and enter them manually.

Some might argue that this feature is a security risk because unauthorized people may be able to see the password. For example a person standing behind the user and reading over her shoulders. However the risk of somebody reading this information can be mitigated if the user is careful and hides the display when she looks up the password. There is also the risk of the user leaving the phone unattended with the MIDlet application open. However if this happens, the system is completely compromised. Even if the MIDlet did not have the option of displaying the username and password, the attacker could simply open a notepad file on the computer and "log in" with all the accounts. This would write all the usernames and passwords to notepad. For the safety of the user, she should exit

the application as soon as she is finished using it. It could also be an idea to implement a timeout function that exits the application automatically after some minutes of inactivity as suggested in Egeberg's thesis [6].

The user also have the option of editing the account information. For example if she needs to change the password on an account. Additionally, the user is presented with the option of deleting a user account.

### ***Changing settings***

From the main menu, the user can change the Bluetooth device that will be used by default. When the user selects this option a search for Bluetooth devices is initiated. When the search is complete, the user is presented with a list of available devices in the vicinity. From this list the user can select the new device she wants to use when logging in. The address of the device will replace the existing one in the record store.



## 5 Security analysis

The single sign-on system developed in this thesis is only a prototype to test to what extent it's possible to implement the proposed concept. This means that no security measures are incorporated in the system. In the following sections, the security requirements of the system will be investigated and discussed.

The security analysis will be conducted on a fairly high level of abstraction and the aim is to find where the most serious vulnerabilities are. Based on the results from the test, measures that will mitigate these vulnerabilities will be suggested. This means that the scope of the analysis is to identify the main areas of the system where security measures need to be implemented. A detailed analysis cannot be conducted until it is determined what measures will be used in the system, hence it's beyond the scope of this thesis.

### 5.1 Security analysis methods

There are three methods commonly used for uncovering security vulnerabilities and threats in a system: threat modeling, attack modeling, protocol analysis. Additionally, Ole Kasper Olsen proposed a framework for adversary modeling in his master thesis from 2005 [34].

#### 5.1.1 Threat modeling

Threat modeling [51] is an approach much used in software engineering and system developments projects. This method focuses on uncovering and understanding the goals an attacker might have for attacking the system. There are several different ways of conducting threat modeling each using different notations for visually modeling the system. One such framework is CORAS. This is a fully based UML framework which aims to provide a good overview of the system's threats and vulnerabilities.

#### 5.1.2 Attack modeling

Attack modeling [46], [49] and [24] is somewhat different from threat modeling. This method tries to identify the attackers full attack path. The attack path will be broken into small attack goals. Threat modeling as described in the previous section is primarily used in the software development phase. Attack modeling on the other hand, is performed when the system is complete. This method is used to do penetration tests on the finished system. This penetration test will asses the level of security in the system when it is complete. The main focus of this method is on analyzing the capabilities and resources of the adversary.

#### 5.1.3 Protocol analysis

Protocol analysis is somewhat different from the other two approaches because it focuses on the details of the communication between entities in the system and not the system as a whole. Also, this method uses a totally different level of abstraction. For example a system may look secure enough in a threat model analysis, but specific communication issues might be uncovered in a protocol analysis.

Several formal methods for conducting protocol analysis have been developed, such as [25], [4] and [17]. The protocol analysis method is very good at analyzing possible attacks on a specific protocol. However for the prototype developed in this thesis, no specific protocol is chosen yet. Therefore it is not possible to use this method.

#### 5.1.4 Adversary modeling

The adversary modeling [34] method also focuses on information flow, but on a higher level of abstraction than the protocol analysis approach. This approach uses a formal framework consisting of the key properties shown below.

**Principals** are persons, computers and processes that are participating and interacting with the system. Every principal in the system can be subject to attacks from adversaries.

**Channels** are the means of information flow between the principals. The principals interact with each other through the channels. There are several different types of channels ranging from speech and written messages to data flow in a network cable. The channels are described with parameters such as bandwidth and direction of the information flow.

**Protected Asset** is what the security measures in a system are supposed to protect. That is, the asset is what the adversary is trying to get access to. The asset is something that has a certain value for the owner. A breach in the security measures will usually lead to a decreased value of the asset. If the confidentiality is compromised, the adversary will have access to the protected information. If the integrity of the asset is compromised, the adversary has tainted the information. The adversary can also prevent legitimate users timely access to the asset by compromising its availability.

#### The adversarial setting:

**Adversaries.** An adversarial setting may consist of several adversaries.

**Channel operations.** Each adversary has its own set of operations he can perform on the channel itself or on the information traveling within the channel. There are three main operations specified, intercept (eavesdrop), write and block<sup>1</sup>. Intercept means that the adversary can monitor data being transmitted over the channel, i.e. he intercepts a package and reads its contents. An adversary with write capabilities is able to inject data on to the channel. Block means that the adversary has the ability to block data transmitted on a channel. Often, an adversary has the ability of combining these operations. E.g. combining the intercept, write and block operations to perform a man-in-the-middle attack.<sup>2</sup>

---

<sup>1</sup> The adversary modeling approach used in this thesis is somewhat different than the method proposed in Olsen's thesis [34]. The changes made to the method was proposed by professor Einar Snekkenes and were meant to clarify some issues that could be subject to misunderstanding in Olsen's thesis. For example the channel operations in his thesis are read, write and intercept.

<sup>2</sup> "A man-in-the-middle attack (MITM) is an attack in which an attacker is able to read, insert and modify at will, messages between two parties without either party knowing that the link between them has been compromised. The attacker must be able to observe and intercept messages going between the two victims." taken from Wikipedia.org



**Capabilities** dictates what the adversary may compute or deduce. The capabilities of the adversary falls in to one of these categories; guaranteed capabilities, probabilistic capabilities or possible capabilities.

**Resources** are the objects an adversary controls or have in his possession. This includes everything from exploitable persons to access to computational hardware and time available to the adversary.

**Intra-adversary** channels are channels between cooperating adversaries. The adversary produced when combining adversaries through intra-adversary channels will be much more dangerous then an adversary operating alone.

When creating an adversary model, the notation shown i figure 8<sup>3</sup> is used.







adversaries	
principals	
channels	
adversary: intercept	
adversary: write	
adversary: block	

Figure 8: Notation used in the adversary modeling framework

### 5.1.5 Choosing a method to use in the security analysis

The attack modeling and threat modeling methods are mostly used in the development stages of the system. These methods are not so well suited for evaluating existing systems resilience against adversaries. The protocol analysis method can be applied to existing systems. However, this method does not consider the system as a whole. The method will only analyze the strength of a specific protocol and in this system, no specific protocol is chosen. Taking these facts into consideration, adversary modeling is the most logical choice of method for the security analysis of the solution developed in this thesis. Also the framework is considered fairly light weight and rapid to use. Adversary modeling will help identify and visualize potential adversaries the system may face.

## 5.2 Security analysis of the SSO system

In this section, a preliminary analysis of the system will be conducted using the adversary modeling approach. As mentioned in the previous sections, the SSO system developed in this thesis is only a functional prototype. This means that the necessary security measures are not yet implemented. The aim if this preliminary analysis is to determine where

<sup>3</sup> Figure taken from [34].

adversaries can attack the system and what measures need to be in place to mitigate the risks.

### 5.2.1 Principals

There are several principals that come into play in this single sign-on system:

- The mobile device with the SSO MIDlet
- The Bluetooth/USB device
- The computer
- The user
- The firmware provider for the Bluetooth/USB device
- MIDlet servers where updates of the SSO MIDlet can be downloaded
- The firmware provider for the mobile phone
- Other Bluetooth enabled devices

The principals involved are also shown in figure 9.

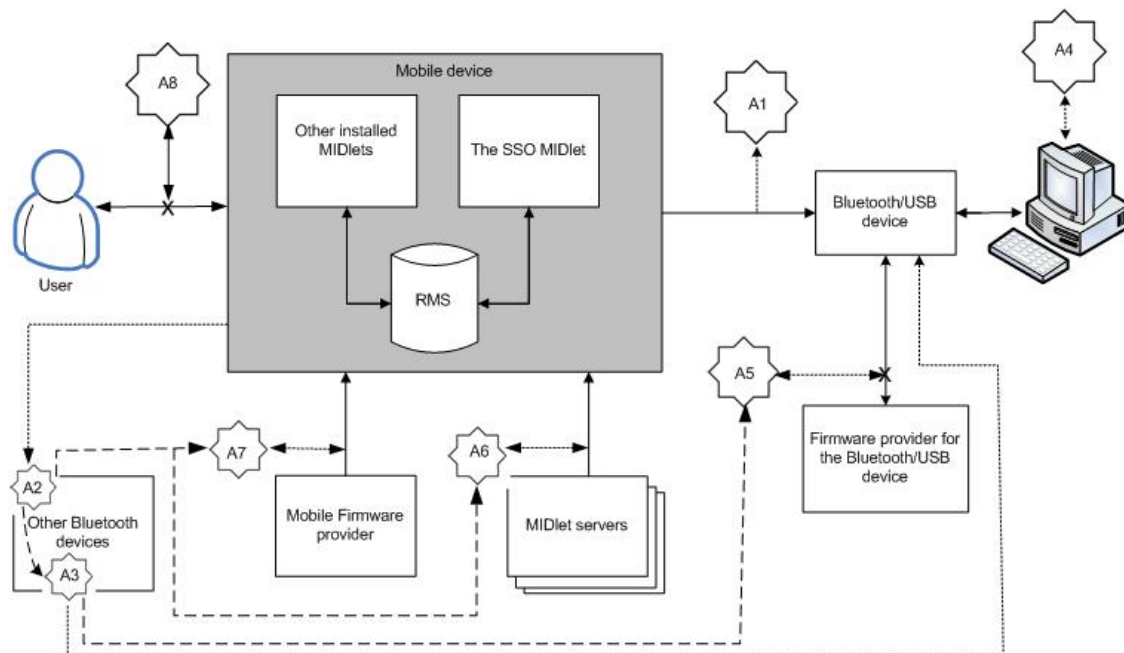


Figure 9: The adversary model for the single sign-on system developed in this thesis.

### 5.2.2 Channels

As seen in figure 9, there are channels between the different principals in the system. They are shown as solid arrows in the figure. The channels between the SSO MIDlet and the Bluetooth/USB device and between the Bluetooth/USB device and the PC are the most important ones for the system to function. These channels are also where all the confidential data is sent. This means that these are the channels that must be protected. Also there are channels between the firmware provider for the Bluetooth/USB device and the device, between MIDlet servers and the mobile phone, between the firmware

provider for the mobile phone and the phone. There is also a channel between the user and the mobile phone. This is not a data channel where information is flowing. This is a channel where the user types data on the phone's keyboard and reads information from it's display.

### 5.2.3 Protected assets

In information security, there are three main aspects that need protection: Confidentiality, integrity and availability. In this SSO system, confidentiality is the most important issue. The aim of the adversary will be to learn the secrets the systems protects, i.e. the passwords. However an attacker might use techniques that compromise integrity or availability in order to learn the secrets. For example, the attacker could launch a man-in-the-middle attack on an upgrade package of the MIDlet's software. This means that the attacker can make the MIDlet server where the upgrade is stored unavailable and trick the MIDlet into downloading a version of the MIDlet modified by the attacker. This means that the integrity of the MIDlet is compromised which in turn might lead to the compromise of confidentiality. Even though protecting against breach of confidentiality is the main focus, it's also important to consider if breach of integrity or availability can be used as a means for compromising the confidentiality of the system.

There are three protected assets in this SSO system. Firstly and most importantly there are all the usernames and passwords stored in the SSO MIDlet.

It is also very important to protect the master password. If an adversary manages to compromise the master password, he will have "the keys to the entire kingdom". That is, the adversary will have access to every username and password stored on the mobile phone.

Another asset which is important to protected is the PC the USB/Bluetooth device is connected to. This is because the Bluetooth/USB device is configured as a keyboard which means that if an attacker is able to connect to the device he might be able to use keyboard commands to remotely control the computer. With almost full access to the computer, the adversary can extract all kinds of sensitive information from it. However protection of the PC is beyond the scope of this thesis.

### 5.2.4 The adversial setting

*Adversary  $A_1$*

An adversary with the ability to monitor the encrypted Bluetooth connection between the SSO MIDlet and the Bluetooth/USB device.

*Channel operations:* Intercept.

*Capabilities:* The encryption scheme used in Bluetooth communication has been thoroughly tested for security weaknesses by the cryptographic community. Results from these tests shows that the encryption scheme is not as secure as anticipated [14, 21]. Research has also shown that there are weaknesses in the protocol for pairing Bluetooth devices [47]. If the PIN code is short enough (usually about 4 digits) and the adversary is within range of the Bluetooth devices during the pairing process, he might be able to compromise the PIN. This will make it possible for the adversary to eavesdrop on the communication and view usernames and passwords being sent between the phone and the USB device.

When two devices are being paired the user has to enter the same PIN on both

devices. This is a problem because there is no way to enter a PIN on the Bluetooth/USB device without installing software on the PC. If we do not want to have software installed on the PC, a default PIN has to be used. This will reduce the security of the system. Because if the adversary finds out what the default PIN is, it's quite simple for him to decrypt the information sent on the channel.

For this attack to be possible, the adversary must have the capability of executing cryptographic operations in order to compromise the PIN used in the pairing process.

*Resources:* In order to complete such an attack, some sophisticated radio equipment is needed for intercepting and processing Bluetooth signals. Also the adversary must be within range of victim's Bluetooth device.

### **Adversary $A_2$**

An adversary with control over another Bluetooth device.

*Channel operations:* Intercept

*Capabilities:* This is an adversary that has his own Bluetooth enabled device, and is within range of the mobile phone with the SSO MIDlet. The adversary can obtain the sensitive information by setting his device name to the same as the default name the MIDlet searches for ("sso device", see section 4.3.2 for more details). This way, the adversary can trick the user into thinking she is choosing the correct device. But she is in fact instructing the MIDlet to connect to the adversary's device each time she wants to log in. This means that, the username and password will be sent directly to the adversary's device. The attack described here will probably only let the adversary get one set of access credentials. Because the user will not be able to log in, she will probably disconnect and try it again. This means that the connection to the adversary's device will be broken. However, also this approach requires that the adversary has knowledge of the Bluetooth PIN. Without this the adversary will not be able to pair his device with the SSO MIDlet and the connection will not be set up.

*Resources:* The adversary only needs a Bluetooth enabled device with the possibility to set a friendly name. This is a feature almost any Bluetooth device has. In order to crack the Bluetooth PIN, the adversary will also need the resources described for adversary  $A_1$ . However if a default PIN is used, obtaining it will require little effort from the adversary.

### **Adversary $A_3$**

This is also an adversary with control over a Bluetooth enabled device.

*Channel operations:* Write

*Capabilities:* This adversary has almost the opposite capabilities as adversary  $A_2$ . Instead of tricking the MIDlet into connecting to a Bluetooth device controlled by the adversary, the adversary will connect to the Bluetooth/USB device. Also in this attack will the adversary need to crack the Bluetooth PIN in order to establish a connection. This is because the Bluetooth/USB device is configured for accepting

encrypted communication only. The encryption key is derived from the Bluetooth PIN. As soon as the adversary has knowledge of the PIN, he can connect to the Bluetooth/USB device. Since this device is configured as a keyboard, the adversary can use it to remotely control the computer. For example, the adversary can write a program on his Bluetooth device that converts ASCII characters to key codes. With this program, he can enter keystrokes as if he actually was sitting at the computer. If the Bluetooth/USB device is connected to a PC running a Linux distribution, the adversary could easily make an image of the hard drive and send the content to a server of his choosing, for example over FTP. He could also use the program to quickly download a trojan to the computer, for example a key logger to log everything being typed on the PC including emulated keystrokes from the Bluetooth/USB device. This information can be used to capture sensitive information such as usernames and passwords

However, there are some conditions that need to be met in order to successfully complete this attack. First, the user needs to be away from the PC. If the user is at the computer, she will notice strange things happening and probably take action, such as rebooting the PC. Second, the user needs to leave the computer without locking it. Otherwise the adversary would require knowledge of the user's password. Third, the adversary must be within range of the device. Usually this means within a radius of 10 to 100 meters. Forth, the Bluetooth/USB device must be connected into the computer.

*Resources:* The adversary will need the same resources as adversary  $A_1$  and  $A_2$ . Additionally he will need a program that converts ASCII characters to key codes (emulated keystrokes).

#### **Adversary $A_4$**

This is an adversary that has compromised the computer.

*Channel operations:* Intercept, Write

*Capabilities:* The adversary has control over the computer. This means that he can read information passed to it. He may for example be able to install a key logger that monitors the data being sent from the Bluetooth/USB device to the computer. However this threat is not unique to this SSO system. The adversary can just as easily monitor everything a user types on a conventional keyboard, including usernames and passwords. The adversary also have write capabilities. He can for example install malicious code that reads sensitive information. Even though this threat is not special for the SSO system being evaluated, it is important to be aware that the threat is there so that security measures can be implemented on the computer as well.

*Resources:* The adversary will exploit a vulnerability in some of the software running on the PC in order to compromise it. To do this, the adversary will probably need tools such as password crackers, port scanners, trojans and other relevant hacking tools. The adversary will also need technical knowledge and knowledge about vulnerabilities it is possible to exploit.

### **Adversary $A_5$**

An adversary with the ability to pass himself off as the firmware provider for the Bluetooth/USB device.

*Channel operations:* Write or block, intercept and write.

*Capabilities:* The chip that handles Bluetooth communication in the Bluetooth/USB device is developed by a third party. Occasionally the developers release firmware upgrades for the chip. The adversary described here, has the ability to masquerade as the original firmware developer. If the adversary can trick the user into thinking he is the firmware provider, he only needs to be able to write to the channel. This can for instance be achieved by sending an e-mail to the user containing a link to the latest firmware upgrade for the device. The e-mail will contain instructions on how to perform the upgrade, and the user will be strongly advised to do so.

Another, and more complicated way to perform this attack is by launching a DOS attack at the original web server making it unavailable. To be able to complete such an attack, the adversary must be able to block the original provider, intercept information from the provider and write new data to the users. When the original site is unavailable (blocked), the adversary can launch a fake web site identical to the original. Next he needs to trick the user into visiting this site and downloading the modified firmware package (write). The adversary will also need access to the code of the original firmware (intercept). This is because the firmware needs to have all the original functionality in place, otherwise the user will notice something is wrong.

The upgrade can for example contain a modified version of the firmware back door functionality. The code can for example send a copy of all the received information unencrypted to a Bluetooth device of the adversary's choosing. Or the code can monitor what Bluetooth PIN is in use, and send it to the adversary. In order to get the most out of an attack like this, the adversary will need to cooperate with other adversaries. This will be described in more detail in the "Intra-adversary channels" section below.

Another approach is that the adversary downloads the firmware, decompiles it and studies the code to see if there are any flaws in the programming. If it is a flaw there, the adversary can take advantage of this without making any changes to the firmware.

*Resources:* In order to successfully complete this attack, the adversary needs to block the original website for example by a DOS (denial of service) attack. There are several ways to perform such an attack. The adversary can either use special tools designed for this, or compromise multiple computers and use them as zombies to launch a distributed DOS attack. The adversary will also need access to a web server where he can host the fake website. Additionally he needs to spoof the IP address of the original server or add his own IP address to a DNS server to trick the user into visiting the fake website.

Sending an e-mail to the user containing a link to where she can download the firmware or attach the package directly in the mail message is probably the easiest way to perform this attack. This method also requires the fewest resources. All the

attacker has to do is make it look as if the e-mail was sent from the company providing the firmware. Spoofing an e-mail address is easy. For example the adversary could set up a PHP site and use the `mail()` function to send the e-mail. This function let's the user specify what the from address of the message will be, for example `upgrade@company.com`.

Getting hold of the original firmware is quite simple. All the adversary needs to do, is to download it from the developers website. Making changes to the firmware is the hard part. The adversary will need to disassemble the code, find out what it does, make the appropriate changes and compile it again. He will also need to compute new checksums if that is used. In order to pull this off, the adversary must have deep knowledge of programming. He must also have tools for decompiling and compiling the firmware.

#### **Adversary $A_6$**

This is an adversary who has control over a server where users can download MIDlets.

*Channel operations:* Write, Intercept

*Capabilities:* This attack can be performed by creating a MIDlet where the vendor and name values in the JAD file are identical to that of the SSO MIDlet. The phone will then treat the installation of this MIDlet as an update of the existing MIDlet. This means that the original SSO MIDlet will be replaced by the version created by the adversary. The modified MIDlet can contain code that reads the content of the RMS where all the usernames and passwords are stored. In this prototype the passwords are stored unencrypted. This means that the adversary's MIDlet can access all the users passwords. It is also possible for the adversary to instruct the MIDlet to send the RMS content to a computer of his choosing for example over GPRS. This type of attack is described in more detail in Tommy Egeberg's thesis [6].

*Resources:* The adversary described here will need a lot of the same resources as adversary  $A_5$ . He can either use the DOS approach, or send an e-mail or an SMS with a link to the modified MIDlet as described above. Also in this case the adversary will need to get a hold of the original code (intercept the channel), or at least he will need to know how the SSO MIDlet looks and feels. This is so that the user don't notice the unauthorized changes made to the MIDlet.

#### **Adversary $A_7$**

This is an adversary with the ability to pass himself off as the firmware provider for the mobile phone.

*Channel operations:* Write and intercept.

*Capabilities:* The capabilities for this adversary is the same as for adversary  $A_5$  and  $A_6$ . Only this time, the adversary has the ability to modify the firmware in the mobile device. If he manages to get access to this code, he can inspect it and look for flaws in the implementation and take advantage of them. The adversary can also make changes to the code, such as installing a back door he can use at a later time to extract information from the phone. Firmware upgrades on the mobile phone is usually performed by technicians and not by the user. The adversary can for

example pose as a technician. He can call up the user and say that she will need to send the phone or deliver it at a specific address because a production error has been discovered. The adversary will inform the user that the update will be free of charge. Once he has the phone, he is free to install whatever software he needs on it.

*Resources:* To trick the user into upgrading the firmware, the kind of resources as described for adversary  $A_5$  and  $A_6$  is needed. He needs write capability to install software on the phone and the intercept capability to obtain the original firmware package from the provider. If the adversary is going to pose as a technician he will need to infiltrate the business that handles support. He can for example get a job there as a real technician or he can pay a technician already employed there to do the job.

#### **Adversary $A_8$**

An adversary in possession of the phone with the SSO MIDlet installed.

*Channel operations:* Block, Intercept, Write

*Capabilities:* If the user has lost her phone or if the phone is stolen, the adversary described here might be in possession of it. If adversary  $A_7$  passes himself off as a technician, he will also have the phone in his possession. Therefore  $A_7$  and  $A_8$  can be the same person. This adversary uses a special form of denial of service. When an adversary has physical access to the phone, the user is not in possession of it. This means that the adversary has blocked her from using the phone, causing a denial of service.

If the adversary manages to get the phone while the SSO MIDlet is running, he will have access to all the usernames and passwords with no extra effort.

However if the adversary has access to the phone but the MIDlet is not running, he must crack the master password first. If the adversary has observed the user for some time, he might have been able to look over her shoulder and read the master password (intercept).

The fastest and easiest way to decrypt the information is by copying the content of the RMS over to a PC (intercept), and use password cracking tools to crack the password. Different techniques for cracking the master password are described in [6]. It is also possible to run a cracking tool on the phone, but it's faster to do it on the PC due to superior processing power. The adversary can also try to guess the correct master password by trying different password combinations directly on the phone (write capability).

The password in this prototype are stored in plain text. This means that there is actually no need for the adversary to crack the master password. If he manages to extract to contents of the RMS, he will have access to all the usernames and passwords directly. Extracting the RMS content can for example be done with the modified MIDlet described under adversary  $A_6$

*Resources:* Once the adversary has his hands on the phone, few resources are required. A computer with normal processing capabilities will do just fine. However, a faster computer will speed up this process. The tricky part is to get the mobile phone.



The adversary can for example watch the user and steal the phone when the users leaves it unattended. This will however require a lot of time just watching the user. Another approach is to steal the phone using pickpocketing skills. A third approach is to do as adversary  $A_7$  and pose as a technician needing to perform an update on the phone.

### 5.2.5 Intra-adversary channels

Intra-adversary channels are created when two or more adversaries start working together. By joining forces, they will achieve a more effective attack than they would have been able to alone. These channels are shown in figure 9 as strong dashed lines.

In this single sign-on system, four such intra-adversary channels have been identified. The first one is between adversary  $A_2$  (an adversary in possession of a Bluetooth device) and  $A_7$  (an adversary posing as the firmware provider for the mobile phone). One adversary ( $A_7$ ) is somehow able to inject malicious code to the phone, for example through a fake firmware update. The code injected could for example contain a back door making it possible for adversary  $A_2$  to access the phone via Bluetooth without entering a PIN number and without the user being notified that the connection has been made. Another possibility is that the code automatically makes a connection to  $A_2$ 's device and sends sensitive information to it.

The same kind of attack can be performed if  $A_6$  (an adversary in control of a server where MIDlets can be downloaded) and  $A_2$  (an adversary in possession of a Bluetooth device) cooperate. Instead of injecting malicious code through a fake firmware update,  $A_6$  tricks the user into installing a modified version of the SSO MIDlet. Once this MIDlet is running, it can for example extract data from the RMS and send to  $A_2$ . However, such an action would prompt the user and ask her permission before establishing the connection. Many users often answer yes to these questions, so the attack may still be possible.

Another intra-adversary channel is created if  $A_3$  (an adversary in possession of a Bluetooth device) and  $A_5$  (an adversary posing as the firmware provider for the Bluetooth/USB device) start working together.  $A_5$  can inject malicious code onto the Bluetooth/USB device. This code can for example make the device accept a default PIN of the adversary's choosing.  $A_3$  is then able to connect to this device. Once they are connected, the malicious code can send a copy of the data received to  $A_3$  through this connection. Since data is decrypted on the device,  $A_3$  will receive everything in clear text. The data received will also be sent to the PC as normal. Therefore the user will not notice this attack taking place. The attack will require that  $A_3$  is within range of the Bluetooth/USB device.

The final intra-adversary channel identified through this analysis is between  $A_2$  and  $A_3$ . This is actually quite a sophisticated attack.  $A_2$  manages to trick the user into connecting to his device instead of the Bluetooth/USB device. Then he will record all the data received, i.e. all the usernames and passwords, and relay the information to  $A_3$  through another Bluetooth connection.  $A_3$  is able to connect to the Bluetooth/USB device. The information  $A_3$  receives from  $A_2$  is forwarded to the Bluetooth/USB device. This device will receive the information and send it to the PC as if it came directly from the SSO MIDlet on the mobile phone. In this attack the adversaries are able to record all communication without the user knowing the attack is taking place. To successfully complete this attack both  $A_2$  and  $A_3$  need to be within range of the mobile phone and

the Bluetooth/USB device.

### 5.3 Security measures to implement

In this section security measures to protect against the attacks described above are proposed. Table 2 gives an overview of the attacks the system is vulnerable to. The table also shows how serious the attack is. Also the effort required to complete the attack is taken into account. L is Low, M is Medium and H is High. The required effort for each attack is estimated based on time required to complete the attack, the technical knowledge required, equipment and software required etc. An attack with high significance and low effort is very critical to the system.

For each of these attacks, one or more security measures will be proposed in order to mitigate or prevent the attack from happening.

Number	Possible attack	Significance	Effort	Adversary involved
1	Eavesdropping on the Bluetooth channel	H	M-H <sup>1</sup>	A <sub>1</sub> , A <sub>2</sub> and A <sub>3</sub>
2	Trick users phone into connecting to the adversary's Bluetooth device	H	M	A <sub>2</sub>
3	Connect to the users phone	H	M	A <sub>2</sub> and A <sub>7</sub> , A <sub>2</sub> and A <sub>6</sub>
4	Connect to Bluetooth/USB device and remotely control the PC	M	H	A <sub>3</sub>
5	Trick Bluetooth/USB device into connecting to the adversary's Bluetooth device	H	H	A <sub>2</sub> and A <sub>5</sub>
6	Compromise the PC	M-H <sup>2</sup>	L-H <sup>3</sup>	A <sub>4</sub>
7	Inject malicious code into the Bluetooth/USB device	M	H	A <sub>5</sub>
8	Trick user into installing malicious update of the MIDlet	H	L	A <sub>6</sub>
9	Install a malicious firmware update on the user's mobile phone	M	H	A <sub>7</sub>
10	Direct physical access to the user's mobile phone	H	L	A <sub>8</sub>

Table 2: An overview of the possible attacks on the SSO system.

The attacks described in the table above, can be summarized as four main vulnerabilities:

1. Bluetooth channel not secure enough for sensitive data
2. Devices involved are not authenticated
3. The confidentiality of the data on the mobile phone is not protected
4. The integrity of update packages is not confirmed

There are some security issues this SSO system face that are not within the scope of this thesis. For example attack number 6, compromise the PC. The system developed

<sup>1</sup>If a default PIN is used the effort required will be lower then if the PIN is entered on both devices by the user.

<sup>2</sup>The significance of a compromise depends on the sensitivity of the information on the PC.

<sup>3</sup>The effort of a compromise can vary from low to high depending on the strength of the security measures on the PC. However, this attack is beyond the scope of this thesis.

here can connect to any PC with an USB port. It is not possible for this system to check the security of each PC it's connected to. The user must on her own decide if she trusts the PC enough to enter usernames and passwords on it.

Issues concerning how the user can retrieve her passwords if the phone is lost or stolen is also beyond the scope of this thesis. However these are important issues and some suggestion will be made in the further work section (chapter 8).

### 5.3.1 Secure the Bluetooth channel

The encryption protocol used by Bluetooth is not secure enough for protecting sensitive data such as passwords. A discussion about the security in Bluetooth is found in section 2.2. If the adversary is able to crack the Bluetooth PIN he is able to view all the sensitive information sent on the Bluetooth channel. This is extremely critical in this system because that would make it possible for the adversary to learn the secrets the system is trying to protect, i.e. the usernames and passwords. However, the attacker must be pretty lucky to pull this off. This is because the pairing of the devices is only performed the first time the SSO MIDlet is used. After that the Bluetooth connection URL will be stored in the RMS, and there is no need to search for or re-pair the devices again. This means that the adversary must be within range of the user when she starts the SSO MIDlet for the first time, otherwise it will be much harder for him to eavesdrop on the communication. The adversary would then need to monitor all the traffic sent between the devices and use cryptanalyst techniques to try to break the encryption. Breaking encrypted information this way, usually requires large amounts of data. Considering that there is very little data being sent between the devices, the adversary will probably not be able to complete this attack.

The effort to complete such an attack estimated to be medium or high depending on how the PIN code is used, but the significance is high if the attack is successful because the adversary will have compromised the confidentiality of the system.

Since the communication channel between the mobile phone and the Bluetooth/USB device is where all the sensitive information is traveling, it is crucial that it's secure. As mentioned above and in section 2.2, the security in the Bluetooth protocol has several known weaknesses. An application level protocol on top of the Bluetooth protocol would be an effective means to secure this channel adequately. Evaluating different protocols in terms of security and efficiency is a time consuming and difficult task. Therefore choosing a specific protocol is beyond the scope of this thesis. However, a discussion on the requirements of the protocol follows.

Some might argue that it would be enough for the user to make sure that there are no other Bluetooth devices in the vicinity. The problem is that it is not possible for the user to determine this. An adversary can simply turn off the visibility of his device to prevent the user from detecting it. This means that the adversary will know of the user's device, but the user cannot see the adversary's device.

One of the goals of this SSO system is that it should work on any computer without having to install any drivers. A result of this design choice is that it's hard to configure the Bluetooth/USB device because it will require software on the PC. This means that it will be difficult to generate master encryption keys to store on the device. It is possible to ship the card with a pre-stored key, but this is not a good approach because if the key somehow is compromised it will be difficult to install a new one. This is why the protocol

used should support automatic key generation such as the Diffie Hellman Key generation protocol [5].

Another important issue to consider is performance. The Bluetooth/USB device has limited processing powers. A protocol that requires a lot of overhead and heavy processing will significantly increase the time it takes to log on with the system. If the log on procedure takes too long, it will affect the usability of the system.

It is also preferable if the protocol chosen has been thoroughly tested by the cryptographic community and been in use commercially. This will mitigate vulnerabilities and errors present. However it is not a guarantee. There may be serious errors or vulnerabilities in a protocol that have not yet been detected, even though the protocol has been in use for a long time.

Developing protocols from scratch is challenging. Newly developed protocols are usually vulnerable. It is also common that errors are made during implementation of a protocol. This will introduce additional vulnerabilities an adversary can take advantage of. This is why using a well known protocol where both the theoretical principals and the implementation has been thoroughly tested, will be the most secure choice.

### 5.3.2 Properly authenticate the devices involved

Attack number 2, 3, 4 and 5 in table 2 are possible because there is no proper authentication between the Bluetooth/USB device and the mobile phone. For this single sign-on system to be secure enough for commercial use, it is important to prevent adversaries from connecting to either of the devices. Finding a way for these devices to authenticate without compromising the usability of the system, is challenging.

A possible method for authenticating the devices, is to use public key cryptography [42]. Both the Bluetooth/USB device and the SSO MIDlet is assigned a key pair. One public key and one private key. The private key is secret and is only known to the device it is stored on and the public key is publicly available. The first time two devices connect, they send each other their public key. When for example the SSO MIDlet wants to authenticate the Bluetooth/USB device, it will connect to it and request the device's signature. The device will then use its private key to encrypt a message. The message can for example consist of a replay of the message sent by the SSO MIDlet together with the device's Bluetooth address. The message should also include a timestamp to assure freshness so that the same package can not be retransmitted by an adversary at a later time. The message can only be decrypted with the public key of the device. Since the Bluetooth/USB device is the only one with access to the corresponding private key, the SSO MIDlet can be sure it is connected to the correct device. The process can also be repeated the other way so that the Bluetooth/USB device know that the correct mobile device is connected to it.

It is also possible to enhance this scheme by using digital certificates. A digital certificate contains a digital signature and the signed public key of the device in addition to other relevant information. The certificate can also have been signed by a certificate authority to ensure its authenticity. A certificate signed by a well known authority is more trustworthy than a certificate signed only by the owner. This is because a certificate signed by a trusted third party ensures the authenticity of the public key associated with the certificate. If the certificate is only signed by the owner, it is possible for the adversary to get his public key accepted as the key of the correct device.

To authenticate the devices using certificates, the devices simply have to send their certificates to each other accompanied by a message encrypted with the private key. If the devices successfully decrypt the message with the public key enclosed in the certificate, they are authenticated. It is also important that the message contains a timestamp so that the freshness of the message can be validated. This is an important measure to protect against reply of old messages.

In order to use this authentication scheme, there are two important issues to consider. The first is the generation and management of private keys. It is crucial that these keys remain secret, otherwise the security of the scheme is compromised. One way to do this is by generating the private key to be used on the Bluetooth/USB device during production, and store it on the device before it is shipped to the customer. Also a signed digitally certificate could be installed on the device. On the SSO MIDlet the key could be generated and stored in the RMS in a secure manner when the MIDlet is executed for the first time. This way the user does not have to bother creating private keys and installing them on the devices.

The second thing to consider is performance. Public key cryptography requires a lot of processing power. As mentioned above, the Bluetooth/USB device have limited resources available. Even though this scheme requires some extra processing overhead, it will only be performed once for each transaction. The authentication will take place each time the SSO MIDlets connects to the Bluetooth/USB device. This means that only the time it takes to set up the connection between the devices will increase slightly.

Another approach is to use key generation protocols for authentication as well as securing the channel. A number of these protocols are able to perform device to device authentication in addition to providing encryption. If a key generation protocol is chosen for additional security of the Bluetooth channel, it might be a rewarding to choose a protocol that also supports authentication of devices.

### 5.3.3 Protect the data stored on the mobile phone

Attack number 10 in table 2 is a very critical attack. When the attacker has physical access to the mobile phone, there is almost no limit to what the adversary can do. The prototype developed in this thesis is especially vulnerable because all the information the SSO MIDlet uses is stored in clear text. Clearly it is important to protect the sensitive data the SSO solution handles. The most obvious measure to protect the confidentiality of the data, is by encrypting the confidentiality of the information.

The standard edition of Java (J2SE) supports the Java Cryptography Architecture (JCA) and the Java Cryptography Extension (JCE). A wide range of cryptographic functions ( e.g. digital signatures, message digests, symmetric and asymmetric cipher) and algorithms (e.g. RSA, AES, DES) are supported by these packages. However, the packages are not directly supported by Java Microedition. To make the packages work on a mobile device, API's from third party vendors have to be used. One such API is the Bouncy Castle crypto package<sup>4</sup>.

Choosing an encryption algorithm for the SSO MIDlet is challenging. There are mainly two aspects to consider, security and efficiency. If a long cryptographic key is chosen, it will take more time to compute the cipher, but it will be harder for the adversary to break the encryption. On the other hand, if a short cryptographic key is used, it's faster to

---

<sup>4</sup> <http://www.bouncycastle.org>

compute, but it is easier for the adversary to break the encryption. A mobile phone has limited processing powers and memory capabilities as opposed to a conventional computer. This is why it is important to find a cryptographic algorithm with the best compromise between computing time and security. Tommy Egeberg suggests in his thesis [6] that Elliptic curve cryptography might be the best choice for encryption on a mobile device because it is computationally friendly and provides reasonably good security. Elliptic curve cryptography is also supported by the Bouncy Castle package.

It is also a good idea to use the master password as part of the encryption key. An adversary who doesn't know the master password will not be able to read the information stored in the RMS without using techniques for breaking the encryption. It is also important that the encryption key is kept secret. This means that it will not be possible to store the key on the phone. Because if an adversary has physical access to the phone, he will probably be able to extract the key and use it to decrypt the sensitive information. By using the master password in the process of generating the key, the adversary must know this password to decrypt the data.

Encrypting the usernames and passwords is important to protect the confidentiality of the system. However, there are also other data values that need to be encrypted in order to assure the security of the SSO solution. The integrity of the connection URL to the Bluetooth/USB device is important to protect. If this is stored in clear text, an adversary with access to the phone might be able to change the URL so that the SSO MIDlet will connect to a Bluetooth device of the adversary's choosing. If the URL is encrypted it will be much harder for the adversary to accomplish this.

As mentioned above, public key cryptography is suggested as a mean for mutually authentication of the devices. This will require a way of protecting the private key. Because if the private key is stored in the RMS in clear text, it is possible for the adversary to access the key. The adversary can then use this key on his own Bluetooth device to fake the authentication to the Bluetooth/USB device. As a result, the adversary is able to connect to the device. To prevent this, the private key used for the digital signature should also be encrypted before it is stored in the RMS.

Another measure that could improve the security of this SSO system is to split the data. This means that data that is important for the system to work is split up so that both the phone with the SSO MIDlet and the Bluetooth/USB device must be present. An adversary who has physical access to the phone will not be able to extract any useful information if he doesn't have the Bluetooth/USB device too.

This measure will also make it harder for the adversary to extract data by installing an altered version of the SSO MIDlet. Because the data he will be able to extract from the RMS is not complete. Also he will have to break the encryption to see how much of the data he has. To successfully complete the attack, the adversary will also need access to the information stored in the Bluetooth/USB device. This will make it harder for the adversary to succeed with his attack. Using this splitting approach, will also authenticate the devices. Because it will not be possible to sign in if only one device is available. If an adversary manages to trick the mobile device into connecting to his device, he will only get fragments of the usernames and passwords. However, there is a downside to this approach. If the Bluetooth/USB device is lost, stolen or broken the data fragments stored on this device will be lost. This means that it will not be possible to completely reconstruct the usernames and passwords.

### 5.3.4 Confirm the integrity of software/firmware updates

A common approach for confirming the integrity of software packages is by using checksums. The checksums are computed by applying a cryptographic hash function to the installation file. This will create a message digest. The value of the digest will change significantly if as much as one bit is changed in the installation file. When the user receives the installation file, she will run the same hash function, if the result from this function is identical to the message digest included in the update package the software is deemed authentic. However, if the adversary has the ability to change the code, he will most likely also be able to compute a new checksum and send it with the installation file. The user will not be able to tell that the software package has been tampered with.

To prevent this from happening, public key cryptography can be used. Each vendor supplying software updates to the SSO system are assigned as set of keys. When the vendor has developed a new software package they want to distribute, a message digest will be computed. The software vendor can then sign the message digest using its private key. To verify the signature, the user has to use the vendors public key to decrypt the digest. The user will also compute a digest of the package. This digest will be compared to the digest that came with the package. If they match, the package is deemed authentic. Because only the vendor is in possession of the private key, it is not possible for an adversary to alter the digest so that it will match his modified version of the package.

Signed MIDlets can also be used to prevent malicious updates of the SSO MIDlet. If the MIDlet is signed the adversary cannot create an unsigned version of the MIDlet to replace the signed one. This is because the J2ME platform prevents this. More details about signed MIDlets is found in Egeberg's thesis [6].

## 5.4 Final evaluation of the solution's security features

The analysis conducted above does not take all elements into consideration. For example, the analysis does not consider what information the adversary can get access to if he uses forensics techniques to extract data directly from the phones memory by removing the memory chip from the phone's motherboard. This kind of attack requires expensive and sophisticated electronic equipment and will most likely only be used by an extremely determined adversary. For example if this single sign on was used to store passwords with high value to the adversary such as administrator passwords for a large company or passwords used by the military. However, the main target group for this system are people who want a system for managing their personal passwords. Using the system for passwords with high value to the adversary, will require a different design of the system where the security measures are much stricter. High security will often effect the usability of the system and also make the system more expensive for the consumer. For this system to be accepted and used by a large number of people, it's important that the right balance between usability and security is found. The security measures suggested in this chapter are all thought to be as user friendly as possible. That is, the system should perform the security checks with as little involvement from the user as possible. This is because many of the potential users of the system will be novices with little or no knowledge of security. Requiring these users to set up the security of the system correctly can be a risk because it will place the responsibility of the security in the hands of the user. This means that the system is only secure if the users configures it correctly. A result of this can be that the user thinks the system is secure, but in reality it is an easy target for an adversary.



If all the security measures suggested above are implemented in the SSO system, it's reasonable to believe that the system will be well protected against attacks from adversaries. That is, the security measures are strong enough to protect the usernames and passwords of normal users. However, one cannot be sure of this until the features are implemented and thoroughly tested. This is because a system may look secure in theory, but the specific implementation can contain flaws such as programming errors. To assure the security of this system, all the security measures should be implemented and a detailed security analysis should be conducted. When this analysis is complete, an assessment of the possible remaining vulnerabilities have to be made. This assessment will determine if additional measures have to be implemented or if the vulnerabilities are insignificant enough so that it is acceptable to live with these risks.



## 6 Results from the user test

In this chapter results from the user test are presented. The user tests were conducted over a period of two weeks and a total of 28 people participated.

In chapter 3.1 it was estimated that the user test should be completed within two working days. However when it was time to set up appointments with the participants, it turned out to be difficult to get all the participants to meet on these two days. The participants who were not able to attend the test during these days were given a choice of any other day within a two week time frame. Also, most of the participants who are not connected to Gjøvik University College (GUC) either as a student or as a staff member, did not have the time to attend the test at GUC's premises. To get these people to participate, the tests were conducted at their offices.

Common for all the test was that the participant was in a room alone so that he or she was not disturbed or stressed. Also this made conditions as equal as possible for the participants. Giving the participants equal conditions is important in order to minimize bias and uncertainty in the results. For example if the test was conducted in a crowded room for some participants and in a quiet room for other participants, this might affect the results.

### 6.1 Background information on the participants

A total number of 28 people participated in the user test, 64% (18) male and 36% (10) female. The age of the participants range from 22 to 53 with an average of about 30 years. However the selection of users is not equal for each age group. The bulk of participants are between 23 and 29 years<sup>1</sup>. A full overview of the participants can be seen in Appendix E.

Due to time constraints the selection of participants were mainly done through acquaintances. This will not result in a representative group because the participants are not selected randomly. A consequence of this is that the results from the user test cannot be generalized to larger population such as the expected target group of the system. Random selections can for example be conducted by choosing people from random pages and lines in the phone book. The problem is that this kind of selection takes a lot more time, because individual appointments have to be set up with each of the participants. Also when you call people randomly, there is a big chance that some of them say no. To gather 28 participants randomly, it's likely that a larger number of potential participant have to be contacted. Also if people are selected randomly, they usually expect some kind of compensation for participating in the test. This project did not have the budget to do this.

Since the single sign-on concept being tested have a large target group, it is important to get a mix of both people with technical knowledge and people with little or no technical knowledge. The selection of participants is slightly biased toward people with technical knowledge. 64% (18) of the participants work with information technology on

---

<sup>1</sup> 71% (20) of the participants are in this range.

a daily basis <sup>2</sup>. While the other 36% (10) are people with limited technical knowledge. Common for all the participants is that they have a number of usernames and passwords they need to manage.

Of the 28 participants, only one does not own a mobile phone (P3). All these participants use their phones on a daily basis. The phone is most commonly used for making calls or sending SMS. There are also a couple of participants who use it to send MMS, take pictures and use the calendar on a regular basis. Few people use the more advanced services on the phone such as playing games, browsing with WAP, checking e-mail etc. Only three participants reported that they use these services regularly (P6, P12 and P16).

User accounts	Participants
1-5	25%
6-10	28,6%
11-15	17,9%
16-20	7,1%
> 20	21,4%

Table 3: Shows the distribution of the number of user accounts the participants have.

In the survey the participants were asked to provide some information on how they manage their usernames and passwords today. Such as how many different accounts they have and if they use the same password on multiple accounts. The participants were also asked how they handle their passwords today. Do they write them down and store them securely, do they write them down and store them insecurely or do they simply remember them. The distribution on how many user accounts the participants have is shown in table 3. In appendix E it is listed how the participants manage their passwords. It is also interesting to note that every single one of the participants use the same password on multiple accounts<sup>3</sup>. This means that if an adversary manages to obtain one of the users passwords, it's likely that he will have access to a number of the user's accounts.

The survey also asked the participants if they believe their passwords are secure. That is, will an adversary have a hard time guessing them. 46% (13) of the participants stated that they believe their passwords are secure, 29% (8) said believed their passwords are not secure and 25% (7) was not sure if their passwords are secure or not. However, it is a bit problematic to interpret this data because it is unlikely that all the participants have the same definition of a secure password. For example a novice user can believe that a password set to the date of birth of his wife is secure. Due to the sensitive nature of passwords, it would not be ethical to let participants elaborate more on information about their passwords because it could compromise their security. Still it is interesting to note that at least 29% of the participants have easily guessable passwords which they use on multiple accounts.

## 6.2 The participants experience with other SSO solutions

It is interesting to uncover the participants previous experience with other single sign-on solutions. First the participants were asked if they have heard the term single sign-on before. As seen in table 4 most of the participants with technical knowledge have heard

<sup>2</sup> P1, P6, P7, P8, P9, P10, P11, P12, P13, P14, P15, P16, P17, P19, P20, P21, P23 and P26 are people with a technical background.

<sup>3</sup> That is everyone except P17 who refused to answer this question.

of single sign on before, and most of the people with limited technical knowledge have not heard of single sign on. This makes sense, since single sign-on is not a term commonly used in everyday speech.

	Heard of SSO	Not heard of SSO
<b>Technical background</b>	50%	14,3%
<b>Non-technical background</b>	10,7%	25%
<b>Total</b>	60,7%	39,3%

Table 4: Shows the distribution of user who have heard of SSO and who have not heard of SSO. It is also taken into consideration if the participant have technical knowledge or not.

The survey also asked the participants if they have tried any other SSO solutions. Only 6 of the 17 who have heard of SSO, had tried other solutions. And only one of these six is a user with a non technical background.

Three participants had tried Microsoft passport (described in section 2.1.4 and three participants had tried the Wand tool in the Opera browser <sup>4</sup> This solution only works on a single computer. If she wants to log in at another computer, she will have to enter the passwords manually again to store them in the Opera Wand on this computer.

MS Passport will only work on services that are registered with the Passport service. This means that the solution is even more limited that Opera Wand because the solution will only work on the limited amount of services connected to Passport.

### 6.3 The participants impression of the SSO concept

In the survey, the participants were asked to give a rating of the SSO system as a whole, and an individual rating for the login function and the creation and deletion of user accounts. The rating can have the following values: very bad, bad, average, good and very good. A summery of the ratings are shown in table 5 below.

Rating	Entire system	Login	Password administration
<b>Very good</b>	39,3%	60,7%	42,9%
<b>Good</b>	53,6%	35,7%	32,1%
<b>Average</b>	7,1%	3,6%	21,4%
<b>Bad</b>	0%	0%	3,6%
<b>Very bad</b>	0%	0%	0%

Table 5: The table shows how the participants rated the system as a whole, and how they rated the login part and the password administration part individually.

The participants were also asked their opinion on several issues concerning the SSO solution. They were presented with 10 statements. For each statement they were asked in what degree they agreed with it. The participants had the following options to choose from: Totally disagree, partially disagree, neither, partially agree and totally agree. Table 6 below summarizes the participants assessment to the statements. To make the table more readable, the options totally disagree and partially disagree have been merged. The same has been done with the options partially agree and partially disagree.

As seen in table 5 the majority of the participants rate the system as good or very good. Only 7% users rate the system as average. No users rate it lower then average.

<sup>4</sup> More information about Opera Wand can be found at <http://www.opera.com/features/wand/>.

Statement	Disagree	Neither	Agree
I could use this system on a daily basis	7,1%	25%	67,9%
Adding user accounts works to my satisfaction	17,9%	7,1%	75%
Deleting user accounts works to my satisfaction	3,6%	3,6%	92,8%
I have no reservations in storing passwords on my mobile phone	21,4%	14,3%	64,3%
I like having all my password in one place	14,3%	3,6%	82,1%
I worry about the security aspect of the system	28,6%	28,6%	42,8%
The system will help me manage my passwords	3,6%	10,7%	85,7%
For me to use this system, I will need a backup solution in case the phone is lost or stolen	0%	0%	100%
The system will increase my security	7,1%	35,7%	57,2%
I'm willing to live with the added risk of storing the user-names and passwords in the web browser instead	60,7%	10,7%	28,6%

Table 6: Shows the participants opinions of the different statements.

The fact that almost 93% of the participants rate this single sign on solution better than average is very good. This is also supported in the first statement in table 6 where almost 68% of the participants said they would like to use this system on a daily basis.

Based on table 5, it seems obvious that the participants liked the login part of the system better than the password management part. 75% (21) users rated the password administration as above average, and 25% (7) users rated it as average or below. But 96% (27) user rated the login part as above average and only about 4% rated it as average, and none rated it below average. This clearly indicates that the login part is more appreciated by the participants than the password administration. From table 6 it is also possible to see that the participants liked the method for deleting user accounts better than adding new accounts. However the difference is marginal. Even though only one user disliked the password management part of the system, it seems obvious that this part is not as good as the login part of the system. One factor that might have biased the results, is the fact the participants did not use their own phone. The phone used in the test was the Sony Ericsson w550i. This is a relatively new phone. For most users it takes a while to get used to a phone, especially if they are used to phones from an other manufacturer such as Nokia.

On the other hand, the fact that almost every one of the participants are used to typing on the keyboard of a mobile device through frequent sending of SMS messages is probably an advantage. However one of the participants (P3) had never used a mobile phone to type characters before. After giving him a short introduction to the triple tap method, he had no trouble entering the usernames and passwords on the phone.

Some of the participants who rated the password administration as above average commented that it was cumbersome to enter passwords in this manner. They also said that even though it is cumbersome it did not matter much because this function will only be used rarely. For most user it is not often they add a new user account or change the password of an existing one. Because of this they don't mind it being a bit cumbersome.

It is also interesting to see that 64% (18) of the participants said they were comfortable with storing sensitive information on a mobile phone. Storing usernames and passwords on a mobile phone is one of the key features of the single sign-on solution developed in this thesis. If the users had not been comfortable about storing this kind

of sensitive information on the mobile phone, this SSO concept will not have much of a future.

A majority of the users also stated that they liked that all their access credentials were kept in one place. Also, a large majority 86% (24) of the users stated that using this system will help them manage their passwords.

Even though the participants liked this SSO concept, it is obvious they are conscious about the security of the system. Less than half stated they were not worried about the security aspect, while the rest expressed some kind of worry. Of course this is natural if they are to trust the system with sensitive information such as their passwords.

For this SSO system to have any chance on the commercial market, it is imperative that there is a backup solution available. This is also confirmed in the survey. All of the 28 users said there needed to be a backup solution in place before they would use this system on a daily basis. Since the users will be trusting the system with their passwords, there is no longer any need to remember them. If the mobile phone is lost, stolen or broken, the user will lose all her passwords if there isn't a backup solution available.

Another good sign from the survey is that 57% (16) believe that this SSO system will improve their security as opposed to managing their access credentials manually. The main reason for this is that there is no need for the user to choose easily memorable passwords or write down the passwords. This means that the user can create more complex passwords, or better yet let the system generate secure passwords. More complex passwords will make it much harder for an attacker to guess the passwords.

The participants were also asked if they were willing to live with the risk of storing their passwords in the Internet browser, by using the auto complete function, instead of this SSO system. 61% (17) of the participants said they were not willing to live with this risk, while only 29% (8) participants said this risk was acceptable. The people who are not willing to live with this risk have to find another way of managing his or her passwords. Creating passwords that are easy to remember or writing down the passwords is just as risky as storing them in the browser.

As mentioned earlier in this section, only 21% (6) of the participants have tried other SSO solutions (P1, P6, P10, P14, P17 and P26). Four of these people (P1, P6, P17 and P26) liked the other SSO solutions they have tried better than the SSO solution developed in this thesis. The main reason for this was that the other solutions did not require any hardware to work. Participant P10 thought the single sign-on solution developed in this thesis was much better than the other solution he has tried (MS Passport) and P14 thought this solution to be equal to Opera Wand.

The other 79% (22) participants have never used any other SSO solution. Therefore they do not have anything to compare this SSO solution with other than manual password management. This might have an effect on the results. Since the users do not have anything to compare the system with, it is hard to determine if it is this SSO concept they liked or simply single sign-on in general. Still a large number of the participants said they liked this solution because it was mobile and could be used on a computer without having to install software or drivers first. However, the only way to make sure the users liked this SSO concept and not just single sign-on in general, is to conduct a user test where the participant are asked to go through the same scenario with different SSO products and rate them after how they function in terms of usability and functionality. Such a test will require more time from each participant and a lot more work to set up

the test. First different SSO products have to be tested and a few different have to be selected to be in the test. Next the solutions have to be installed and configured properly before the testing can begin. Conducting such a test would require too much resources to be performed within the time limits of this thesis.

## 6.4 Commercial aspect

In the survey the participants were also asked their opinion on how much they were willing to pay for this single sign-on system if it was commercially available. The package the users buys will include the Bluetooth/USB device and the software to be installed on the mobile phone.

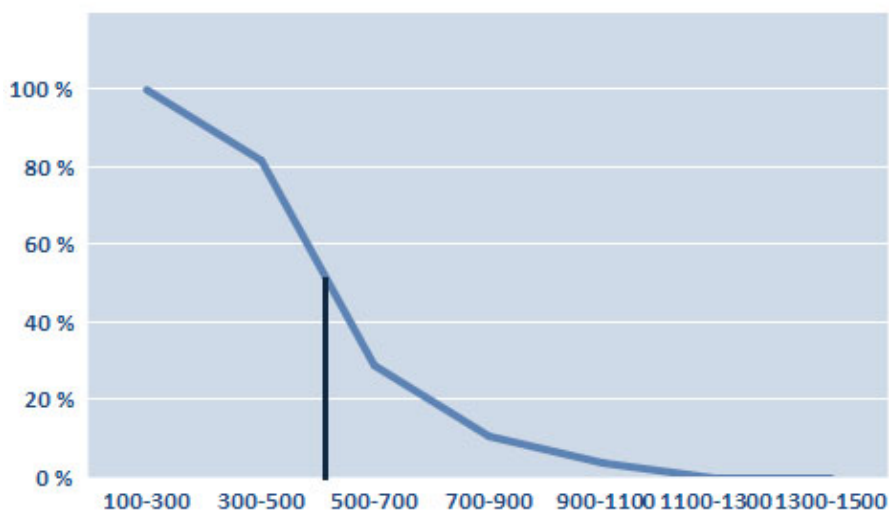


Figure 10: Is an graphical representation of table

To find the limit on how much they are willing to pay, the participants were presented with 7 different price ranges. For each range they were asked in what degree they would agree to pay this price. Results from this part of the survey are shown graphically in figure 10. The participants should give a rating for each of the price ranges. This means that the same participant can for example both agree to pay 100-300 and 300-500 for the system. The reason for doing it this way, is to try to find where the users feel the price limit is. Based on this figure it is possible to find how much the users are willing to pay for this solution. Doing this kind of research before the system is ready for the commercial market is important. This is because it is important to know that the production cost will be less then the price the users are willing to pay.

The line in figure 10 shows how many users would agree to pay the different price ranges. For example 100% of the participants were willing to pay between 100 and 300 NOK for the system, while only 4% were willing to pay 900 to 1100 NOK.

The dark vertical line in the figure represents the critical limit. This is where more then 50% of the users feel the price is too high. If a product is priced so high that over half of the potential customers feel it's too expensive, the price is too high. A large majority of the users were willing to pay between 300 and 500 NOK for this system. Based on these results it seems wise to set the price somewhere between the values. For example 399

NOK seems like a sensible price for this system.

## 6.5 Comments from the participants

During the survey, some of the participants had comments that the survey did not cover. Also some of the participants were chosen for an interview where they were asked to speak freely about what they liked and disliked about the system. Any comments they had on improving the system was also appreciated.

### *Participant P1:*

This participant stated that he thought the system was too slow for him. He prefers having two different passwords, one that he uses on secure sites and one he uses on more insecure sites. With this approach he only has to remember two different passwords.

He also thought it was too cumbersome to use the mobile every time he wanted to log in. He said that he often forgets to charge his phone. If there is no power on the phone he will not be able to log in. This is the main downside to the system for him. However he thought it was nice to have all passwords stored in one place. He also stated that he would probably use this system when he is traveling. He was also concerned about what will happen if the phone is lost or stolen, and what will happen if the Bluetooth/USB device is lost, stolen or broken.

### *Participant P2:*

The SSO system was very well received by this participant. She liked how the concept worked. The participant had some trouble using the MIDlet because she was not used to the w550i phone. This participant is used to working with a Nokia phone. If the MIDlet had been installed on the phone she usually uses, it would probably be easier for her to use the it.

Also this participant stated it would be nice to use this system while traveling. She also stated that she would much rather use this system daily than storing the passwords in the web browser. She also feels that using this system, will enable her to use more secure passwords.

### *Participant P3:*

This was the only participant in the test that did not own a mobile phone. Naturally he needed an introduction on how to use the phone before he could start the test. When he had completed the user test, he was so happy with this system that he said if this system will be commercially available, it will be a reason for him to buy a mobile phone.

### *Participant P13:*

Participant number 13 also liked this SSO concept. He is the system administrator for a local company. He thought this system would be useful in his line of work because there are several people that need access to the different servers in the network. Each of these employees could be issued a mobile phone with all the passwords together with a Bluetooth/USB device.

He also suggested that there could be a central service, such as a web server the system could use to synchronize with. For example if the password on any of the servers are changed all the MIDlets could update their local RMS with the new password automatically without the need of manually entering it on all the phones. Using this system will

also make their password management more secure. This is because passwords updates are today issued on Excel spreadsheets. It would be a lot more secure if password updates was done on the central server, and the passwords were downloaded to the phone securely.

He also made a suggestion on the Bluetooth/USB device. He said it would be really great if it was possible to make this device small enough so that it would fit on the mobile phone. For example by installing a magnet on both devices. This is a very good idea, but it requires that the manufacturer of mobile phones are willing make such a feature on the phone. If this is possible and mobile phones are shipped with the Bluetooth/USB device and the SSO MIDlet already installed, it is likely this system will be a hit.

***Participant P14:***

This participant had some trouble finding the delete function on the MIDlet. In order to delete a user account, it has to be selected. The user can then press the "More" soft key and choose delete from the menu. P14 would have preferred to have the delete command in the main menu instead.

When a user enters a password on the MIDlet, the character will be visible for about one second. After that it will be replaced by the '\*' character so that the password is masked. P14 thought this feature was confusing on a mobile phone. This is because you have to tap the same key multiple times to get the correct character. It is very easy to enter the wrong character. And since the password is masked it's not easy to detect if the wrong password is entered. The user will then need to edit the user account and correct the password. This is frustrating for the user. As suggested by this participant, it might be a good idea not to mask the passwords as they are entered on the MIDlet. Not masking passwords on a mobile phone is not big security risk as it is on a PC. This is because an eavesdropper have to stand close to the user and look over his shoulder in order to see the password being entered. This is quite easy for the user to notice.

***Participant P16:***

Also this participant had problems with the masked password. At the first attempt, he entered the password incorrectly. He therefor had to edit the user account, and enter the password again. Besides this, the participant really liked this SSO concept. he especially liked that everything is done from the mobile phone. This is because he always have the mobile phone with him. He also liked that finally there are some useful programs for the mobile phone, not just games and such.

This participant also suggested that the Bluetooth/USB device should work as a storage unit for digital certificates as well. So that this device can also be used for creating digital signatures. By adding several features into one unit, the user wouldn't need to carry a number of devices for different purposes.

He also expressed worry about what would happen if the Bluetooth/USB device is lost, stolen or broken. Because if the device is not present, it is not possible to sign in automatically.

***Participant P17:***

Did not care too much for this solution. He thought it was cumbersome that he needed the mobile phone each time he wanted to log in. Also he said that he liked the MS Passport solution better than this one because it did not require any extra hardware. He also did not like that the password was masked while being entered.



**Participant P21:**

This participant had some problems when he used the solution to log in. He forgot to place the cursor in the username field before logging in. This meant that the address bar in the browser was in focus. When he sent the username and password to the Bluetooth/USB device, the data was typed in the address field causing the password to be shown in clear text. If there are people standing near the computer when this happens, he might be able to see the user's password. P21 expressed some worry about this and he would like some logic that makes sure the cursor is in the username field before any data is sent. This is a good idea, but it is difficult to implement because it will require a driver of some kind to be installed on the PC. It is possible to do this, but it will make it harder to use the system on any computer because it would require a certain level of user privileges to do this. For example if this system is to be used to log in on a PC on an Internet café, the user might not have the proper rights to install the required software.

**Participant P26:**

P26 felt the delay on the login function was too long<sup>5</sup>. She would much rather prefer using Opera Wand where the delay is close to zero. She also thought it was too cumbersome to add new access credentials. When a set of username and password is entered into a login form on a website, the Opera Wand will ask the user if she wants to store this information. If the user clicks yes, the data is automatically stored. The next time she enters the site, she only have to click the wand icon to log in. P26 missed this way of automatically storing the usernames and passwords without the need for entering them manually.

---

<sup>5</sup> It takes somewhere between 2 and 4 seconds for the SSO MIDlet to establish a connection with the Bluetooth/USB device.



## 7 Discussion

The SSO solution developed in this thesis uses standard sign in components, i.e the user-name and password field, to authenticate directly to the services. There are no services that performs the authentication on behalf of the user such as the ASP (Authentication Service Provider). Also the user only has to authenticate once. That is when the MIDlet is launched. When the MIDlet is running, all the authentication to the services will be handled by the MIDlet. This makes the prototype a pseudo single sign-on solution by the terms used in Pashalidis' taxonomy [36]. Also the only components involved are the PC, the Bluetooth/USB device and the mobile phone. There is no external proxy involved, which makes the prototype developed in this thesis a local pseudo single sign-on solution.

There were three main goals identified with this thesis. First it was to find out weather or not it was possible to implement the SSO concept proposed. A working prototype is now in place, so this goal is reached.

The second goal of this thesis was to conduct a user test where we wanted to find out what the users think about this SSO concept. It is important to note that we only want to test the concept of storing the access credentials on the mobile phone and automating the login procedure. We do not want to test the details of the prototype, such as the design of the user interface because this will most likely be changed in later versions. Results from the user test were good. Almost 68% of the participants said they would like to use this kind of SSO system on a daily basis. A majority of the user liked the fact that all their usernames and passwords were gathered in one place. They also beleived that using this SSO system, their overall security level will increase. Most of the participants also liked that the system is mobile and that it will work on almost any computer without having the install any drivers or software.

However, there are a couple of issued that may have effected the results from the test as discussed in section 6.3. The main concern is that very few of the participants have any experience with other SSO solutions. This means that they were not able to compare it to other SSO concepts. A consequence of this might be that some of the participants simply liked the idea of single sign-on, and not this solution in particular. Still, this issue is probably not effecting the results dramatically. Some of the participants who had tried other SSO systems, also stated they liked this solution better. Mainly because it is mobile and very easy to use. The mobility feature of the system was also something a majority of the participants, who has not tried any other SSO solutions, emphasized. Hardly any other SSO solution available on the market today, support this kind of mobility.

The selection of participants in the user test is not representative for the Norwegian population or even the target group for the SSO system. There are too few participants to generalize the selection so that it is representative of the Norwegian people or the target group<sup>1</sup>. A selection that is to be representative for such a large population, has to be much larger. A large scale user test would not be possible to successfully complete

---

<sup>1</sup> The target group for this SSO solution is very large. Everyone who uses a computer and has more then one set of usernames and passwords to manage, are in this group.

within the time limits of this thesis. It takes a lot of time and resources to conduct a user test of this size. Also the selection of participants should be random, so that the selection is as representative of the population as possible.

Because the population is small and not representative, it is of little value to perform statistical analysis, such as correlations, on the data. This is because the statistical results will probably be much too inaccurate. It is also likely that results from a user test where the selection is larger will differ from the results obtained in this thesis. For example it is possible that the results from the user test are too positive because we were lucky with the selection of participants. However it is difficult to know the accuracy of the results before a larger number of participants have tested the system.

Still the results are important and relevant. Because all the people who participated in the test are users who belong in the expected target group of the system. This means that the information the participants provided is information from a subset of the target group. The results from the test provide an important pointer on how this single sign-on concept will be accepted by the users.

Finally the third goal of this thesis was to conduct security analysis. The analysis uncovered where possible adversaries could attack the system and what capabilities they have. It was also given an estimate on the severity of the attack and the effort required to successfully complete the attack. The aim of the analysis performed was to uncover where the system is vulnerable and make suggestions on what security measures that needs to be implemented in order to mitigate the risks. Results from the security analysis show that it is likely that the system will be secure enough when the proposed measures are implemented. When the security measures are implemented, a more detailed security analysis can be conducted. This analysis should use other methods with a lower level of abstraction, because the aim of this analysis will be to find out if the system is still vulnerable after the measures have been implemented.

## 8 Further work

The prototype developed in this thesis is basically just a functional prototype which shows that it is possible to implement the proposed SSO concept. Based on the results in chapter 6, this solution has the potential to become a commercial product. Before the product is ready for the commercial market, there is a lot of work that needs to be done. The remaining work can be split into two categories, practical and theoretical. These two categories will be described separately in the following sections. The practical section includes the issues that has to do with the implementation. While the theoretical section includes suggestions on what can be done on a more theoretical level to improve the overall security and usability of the system, such as security analysis and usability tests.

### 8.1 Practical further work

#### **Implement security measures:**

The most important thing that needs to be done, is to implement security measures in the system. For example some or all of the security measures suggested in section 5.3 could be implemented to make this a secure prototype. At least measures to protect against the four main vulnerabilities found in section 5.3 should be implemented.

#### **Port the solution to different hardware:**

As described in section 4.2.1 the solution was supposed to be implemented on a device developed at NISLAB. However, this device was not ready when the programming in this thesis started. Porting the code to this device should not be too big of a job since it is using the same microchip as the USB device used in the prototype. This means that most of the code used in the prototype can be reused on the NISLAB device. The only part of the code that has to be rewritten is the communication between the Bluetooth and USB units. In the current prototype, these are on separate devices and they communicate through the RS232 interface. On the NISLAB device, the chip from the USB device and the Bluetooth device are soldered on the same circuit board. This means that the communication between these chips can be done internally.

**Implement a backup solution:** Based on the results from the usability test, a backup solution needs to be in place for this system to be used on a daily basis. Every participant in the user test stated that a backup solution must be in place before they would consider using the system. Implementing a backup solution in this SSO system, is something that needs to be done before the system is ready for the commercial market. A backup solution can for example be created on the web. Each person using the system will have a user account on a web server. This account will contain a copy of the usernames and passwords stored on the mobile phone. It is also possible to automate the process of backing up the access credentials. For example each time the user adds a new set of username and password or alters an

existing account, the content of the RMS will be written to the account on the web server over an Internet connection such as GPRS. The contents of the RMS will be sent encrypted over the Internet and stored securely on the web server. If the user's phone is lost, broken or stolen, all she has to do is connect to the account on the web server and retrieve the usernames and passwords.

A solution like this could also be used to make it easier to enter new usernames and passwords. When the user wants to add an account, she can log on to the web site and enter the information there. Next time the SSO MIDlet is executed, it will connect to the server and download the updates. This way the user do not have to go through the cumbersome process of entering usernames and passwords on the mobile phone.

#### **Support different login screens:**

When the user wants to log in to an account, the username is sent to the Bluetooth/USB device followed by a TAB character, the password and then the ENTER key. This approach will not work on for example a SSH login, because it expects username, ENTER, password, ENTER. On Windows XP machines it's not necessary to enter the username. The user simply click on the username, and she is prompted for the password.

It is quite simple to make the system support different types of login screens. For example it can be achieved by presenting the user with a list of different login screen supported when a new account is created. The user only has to select the correct login type from this list.

#### **Redesign the MIDlet's user interface:**

The prototype MIDlet has been created using standard components included in J2ME. The problem with these components is that it's not Java that decides their look and feel, but the manufacturer of the phone. This means that input boxes etc will look different on a SonyEricsson phone and a Nokia phone. Also it is the layout of the phone that decides where the labels and soft keys should be placed. This makes it difficult for the developer to have any control over how the user interface will look and feel. It might also be confusing for the user if she changes from one phone to another.

To avoid this, the MIDlet should be implemented with low level user interface elements instead, i.e not the standard Java components. With this approach, the developer have better control over the look and feel of the user interface elements so that it will look the same on different phones.

#### **Support multiple keyboard layouts:**

The current prototype is only able to map between ASCII characters and key codes using the Norwegian keyboard layout. Later version should implement a more dynamic mapping so that multiple keyboard layouts are supported.

#### **Let the system generate secure passwords:**

When the the user wants to create a new user account, he can run a password generator on the MIDlet. The secure password generated will be set as the current password for this account. This way, the user doesn't have to create the passwords

herself. Also this will mitigate the risk of using the same passwords on several accounts.

## 8.2 Theoretical further work

### **Conduct a detailed security analysis:**

When the security measures are implemented, a new security analysis should be conducted. First a theoretical analysis can be performed to uncover possible vulnerabilities remaining in the system. It could also be rewarding to perform a practical penetration test where security professionals are allowed to try to break the system with any means necessary.

If a backup solution as described above is implemented, it's important to conduct a thorough analysis on this as well. Because the backup solution will contain the same sensitive data as the SSO MIDlet which makes it an attractive target for adversaries. Actually it is even more important to protect this solution, because it is available over the Internet. This means that any adversary with an Internet connection can try to attack it. Because of this, it is important that this part of the solution does not become a weak link in the overall security of the SSO system.

In addition to making sure the data on this solution is secure, it is important to have good procedures for authenticating the user and the MIDlet so that an adversary cannot use his own phone to download the victim's backed up access credentials.

### **Conduct a more thorough user test:**

When all the necessary security measures are implemented, it is possible to conduct a more thorough user test. Since the system is secure, the participants can trust the system with their real usernames and passwords. This will create a more realistic scenario, because the users can use the system to sign in to the accounts they normally use.

When the NISlab device is complete, it will also be possible to let the participants use the system for a longer period of time. Each participant could for example test the system for about a week. This will give the user a more accurate impression of the system.

It could also be interesting to conduct a comparative study where the participants go through the same scenario using a number of different SSO solutions. Allowing the participants to directly compare the solutions and rate them on functionality, efficiency and usability. This will provide information on how the users like this system as opposed to other SSO solutions. It could also be interesting to get a broader and larger selection of participants so that it will be more representative for the expected target group of the system.

A more formal usability test should also be conducted if the user interface is re-designed. The participants should test every aspects of the interface during formal settings. During this test the participants should be videotaped and timed. This will provide information on where the user has trouble understanding the interface. Such a test could be conducted on a simple mock-up of the interface early in the design process, when it is easy to make changes based on the feedback. A second test could also be conducted near the end of the design process.





## 9 Conclusion

In section 1.4 the research questions for the thesis were defined. It is now time to return to these questions and determine to what extent they have been answered.

1. **What types of single sign-on solutions are available on the market?**

A number of SSO solutions are presented in section 2.1. After studying these solutions, it was clear that little work has been done on mobile SSO. Most of the solutions presented only work on a single computer, or in large computer networks. There are a few solutions that allow the users to store their usernames and passwords on a mobile phone, but they do not provide any functionality for automating logins. This means that the solution developed in this thesis presents a new SSO concept.

2. **How secure is the Bluetooth protocol for transferring sensitive data?**

The Bluetooth protocol is an important part of the SSO solution developed in this thesis because sensitive data is over sent it. If the users are to trust the system with their usernames and passwords, it's important that the communication channel is secure. After studying several papers on Bluetooth security, a number of vulnerabilities in the protocol were discovered. Due to the sensitive nature of the information the system handles, the Bluetooth protocol alone is not secure enough to properly secure the communication channel.

3. **Is it possible to implement the proposed single sign-on concept?**

The prototype developed shows that it is possible to implement the SSO concept.

4. **What security mechanisms need to be in place to assure the security of this system?**

The prototype developed in this thesis only contains the basic functionality needed to prove that it is possible to implement the proposed SSO concept. This means that no security measures are implemented. In chapter 5, a security analysis is conducted. The aim of this analysis was to determine what security measures that needs to be in place. When some or all of the security measures suggested in section 5.3, the SSO system should have a security level that is acceptable for most users.

Analysis show that there are mainly three aspects of the system that are vulnerable. First it is the communication channel between the mobile phone and the Bluetooth/USB device. Second the data stored on the mobile phone must be protected, for example by using encryption. Finally the devices involved must be certain that they are not communicating with a device controlled by the adversary. This can be achieved by properly authenticating the devices.

5. **How will this SSO concept be received by the users?**

To find out what the users think about the SSO solution developed in this thesis, a user test was conducted. Even though the selection of users is not representative of the expected target group, the results provide a good pointer to how this SSO solution will be received. Results from the test show that a majority of the participants liked the solution, and would be willing to use it on a daily basis. Because this is a device

they carry with them at all times. It is also a plus that the solutions works on almost any computer without the need to install drivers or software. Based on the positive feedback from the participants, it is inspiring to continue work on the solution, and hopefully make it available on the commercial market one day.

**6. Will this SSO concept increase the users' security level?**

The main goal of the SSO system is to help the users to increase their level of security. Conventional password management is usually not very secure. Often easily guessable passwords are chosen or the password is written down. This was also reflected in user test. Several participants stated that they do not think their passwords are secure. Also every participant in the test said they used the same password on more than one account.

As described in section 1.3 it is likely that passwords will be a common method for authentication for a long time to come. Password based authentication is a very good security mechanism if the passwords are secure enough. Normal password management is by no means secure enough due to the fact that the users have to remember or write down their passwords. Single sign-on solutions offer a way to manage passwords more securely by taking responsibility for the password management. When the user choses to use a single sign-on solution, it is no longer any need to remember the passwords. Therefor it is possible to choose complex password that are hard to guess or crack with brute force attacks.

The single sign on system developed in thesis will increase the user's security if the user manages his or hers passwords manually. However this system provides about the same security level as other SSO solutions available. That is, if the user is using another SSO solution, his or hers security level will, in most cases, not be increased by switching to the solution developed in this thesis. One exception is if this solution is used instead of for example a web based SSO solution on an insecure computer. The web based solution would require the user to enter the master password on this PC. This might lead to a compromise of the master password. If the SSO solution developed in this thesis is used, only the one password used to sign in will be compromised. The master password is only entered on the mobile phone and will remain protected.

The SSO solution developed in this thesis also has some functional advantages. It seems like this is the only mobile SSO solution available on the market today that supports automated sign ins. It is also very easy to use. No software or drivers are required, only a mobile phone and the USB device connected to the PC.

The mobility feature of this system is it's greatest strength and what makes it stand out from other SSO systems. This feature might also make more users convert from conventional password management to single sign-on, something that will increase their overall security.

To summarize, the single sign-on solution developed in this thesis will increase the user's security level if his or hers passwords are managed manually. In addition the system is user friendly and requires very little technical knowledge to use. Also the users have access their passwords everywhere as long as they have their mobile phone. When the system also has a function for automating logins to any PC with a USB port, it's likely that the system has potential of becoming a commercial success.

## Bibliography

- [1] CDW.G. How IT works what it decision makers need to know, 2005.
- [2] CEZEO Software. SecureWord Mobile. <http://www.cezeo.com/products/secureword-mobile>. (Last visited november 2005).
- [3] Danino Nicky. Heuristic evaluation - a step by step guide. 2001. <http://www.sitepoint.com/article/heuristic-evaluation-guide> (Last visited April 2006).
- [4] DeMillo Richard A., Nancy A. Lynch and Merritt Michael. Cryptographic protocols. In *Proceedings of the fourteenth annual ACM Symposium on Theory of Computing*, pages pages 383–400, 1982.
- [5] Diffie Whitfield and Hellman Martin E. New directions in cryptography. *IEEE Transactions on Information Theory*, IT-22(6):644–654, November 1976.
- [6] Egeberg Tommy. Storage of sensitive data in Java enabled cell phone. Master's thesis, Gjøvik University College (GUC), 2006.
- [7] ESoftPRO. Online password manager. <http://www.handyarchive.com/Utilities/Password-Recovery/12343-Online-Password-Manager.html>. (Last visited november 2005).
- [8] Free2Move. Bluetooth serial port plug - F2M01C1 Datasheet. 2004. [http://free2move.se/pdf/Datasheet\\_F2M01C1.pdf](http://free2move.se/pdf/Datasheet_F2M01C1.pdf) (Last visited April 2006).
- [9] Gehrman Christian. Bluetooth security whiteaper, Bluetooth SIG Security Expert Group. 2002.
- [10] Herfurt Martin. Bluesnarfing @ CeBIT 2004, detecting and attacking bluetooth-enabled cellphones at the Hannover Fairground. *Salzburg Research Forschungsgesellschaft mbH, Austria*, 2004. [http://trifinite.org/Downloads/BlueSnarf\\_CeBIT2004.pdf](http://trifinite.org/Downloads/BlueSnarf_CeBIT2004.pdf) (Last visited April 2006).
- [11] Hyde John. *USB design by example : a practical guide to building I/O device 2nd edition*. Intel Press, 2001.
- [12] Inc Com Consulting. Mobipassword. <http://www.mobipassword.com>. (Last visited November 2005).
- [13] Insight Consulting. How can bluetooth services and devices be effectively secured. *Computer fraud and security*, pages 4 - 7, January 2006.
- [14] Jakobson Markus and Wetzel Susanne. Security weakness in Bluetooth In D. Naccache, editor, *Proceedings of the Cryptographers' Track at RSA Conference*,. *Lecture Notes in Computer Science*, volume 2020:pages 176–191, April 2001.

- [15] John Bonnie E. and Mashyna Matthew M. Evaluating a multimedia authoring tool. *Journal of the American Society of Information Science*, 48(11):pages 1004–1022, 1997.
- [16] Kaletka Mark O. Easy things users can do to improve security: Recommendations of "best practices" for securing individual user's accounts. <http://security.fnal.gov/UserGuide/password.htm>, 1998. (Last visited November 2005).
- [17] Kemmerer Richard A. Analyzing encryption protocols using formal verification techniques. *IEEE Journal on Selected Areas in Communication*, pages 448–457, May 1989.
- [18] Kügler Dennis. Man in the middle attack on Bluetooth. *Lecture Notes in Computer Science*, 2742:pages 149 – 161, Aug 2003.
- [19] Lastbit Software. Password director. <http://www.passworddirector.com>. (Last visited November 2005).
- [20] Levi Albert, et.al. Relay attacks on Bluetooth authentication and solutions. *Lecture Notes in Computer Science*, 3280:pages 278– 288, January 2004.
- [21] Levy Ophir and Wool Avishai. A uniform framework for cryptanalysis of the Bluetooth  $E_0$  cipher. *Cryptology ePrint Archive*, Report 2005/107, 11 April 2005.
- [22] Mahmoud Qusay H. The java APIs for Bluetooth wireless technology, part II. *Sun Developer Network*, 2003. <http://developers.sun.com/techtomics/mobility/midp/articles/bluetooth2/> (Last visited April 2006).
- [23] Massachusetts Institute of Technology. Kerberos. <http://web.mit.edu/kerberos/www/>. (Last visited november 2005).
- [24] McDermott John P. Attack net penetration testing. *In Proceedings of the 2000 workshop on New Security Paradigms*, 3.2:pages 15–21, September 2000.
- [25] Meadows Catherine. A system for the specification and analysis of key management protocols. in. *Proceedings of the 1991 IEEE Synopsium on Reaserch in Security and Privacy*, pages 182-195, May 1991.
- [26] Microchip. PICDEM FS USB demonstration board users guide. 2004. <http://ww1.microchip.com/downloads/en/devicedoc/51526a.pdf> (Last visited January 2006).
- [27] Microchip. MPLAB C18 C compiler libraries. 2005. [http://ww1.microchip.com/downloads/en/DeviceDoc/MPLAB\\_C18\\_Libraries\\_51297f.pdf](http://ww1.microchip.com/downloads/en/DeviceDoc/MPLAB_C18_Libraries_51297f.pdf) (Last visited May 2006).
- [28] Microsoft Coperation. Microsoft passport. <http://www.passport.com>. (Last visited November 2005).
- [29] Muller Thomas. Bluetooth security architecture version 1.0. 1999.
- [30] MyPasswordManager. Your powerful password keeper. <http://mypasswordmanager.com/index.htm>. (Last visited November 2005).

- [31] Nielsen Jakob. How to conduct a heuristic evaluation. 1994. [http://www.useit.com/papers/heuristic/heuristic\\_evaluation.html](http://www.useit.com/papers/heuristic/heuristic_evaluation.html) (Last visited April 2006).
- [32] Niemi Antti T. Single sign-on with Kerberos V, Helsinki University of technology. <http://www.tkk.fi/cc/docs/kerberos/sso.html>, 2002. (Last visited November 2005).
- [33] O’Gorman Lawrence. Comparing passwords, tokens, and biometrics for user authentication. *Proceedings of the IEEE*, 91:pages 2019+, 2003.
- [34] Olsen Ole Kasper. Adversary modelling. Master’s thesis, Gjøvik University College, 2005.
- [35] Ortiz C. Enrique. Using the Java APIs for Bluetooth Wireless Technology, Part 1- API Overview. <http://developers.sun.com/techtopics/mobility/apis/articles/bluetoothintro/>, 2004. (Last visited November 2005).
- [36] Pashalidis Andreas and Mitchell Chris J. A taxonomy of single sign-on systems, information security group, Royal Holloway, University of London. In *Lecture Notes in Computer Science*, volume 2727:pages 249–264, July 2003.
- [37] Pashalidis Andreas and Mitchell Chris J. A single sign-on system for use from untrusted devices, information security group, Royal Holloway, University of London. 2004.
- [38] Pashalidis Andreas, Mitchell Chris J. Single sign-on using trusted platforms. technical report RHUL-MA-2003-3, Mathematics Department, Royal Holloway, University of London. 2003.
- [39] Pashalidis Andreas, Mitchell Chris J. Using GSM/UMTS for single sign-on, information security group, Royal Holloway, University of London. 2003.
- [40] Protocom. Securelogin. [http://www.protocom.com/html/securelogin\\_single\\_sign\\_on.html](http://www.protocom.com/html/securelogin_single_sign_on.html). (Last visited November 2005).
- [41] Qinghan Xiao. Security issues in biometric authentication. *Systems, Man and Cybernetics (SMC) Information Assurance Workshop, 2005. Proceedings from the Sixth Annual IEEE*, pages 8-13, June.
- [42] Rivest Ron, Shamir Adi and Adleman Len. A method for obtaining Digital Signatures and Public Key Cryptosystems. *Communications of the ACM*, pages 120–126, February 1978.
- [43] Saltzer Jerome H. and Schroeder Michael D. The protection of information in computer systems. *Proceedings of the IEEE*, volume 63:pages 1278–1308, September 1975.
- [44] Samar Vipin. Single signon using cookies for web applications. in. *Proceedings of IEEE 8th International Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises*, pages 158–163, June 1999.

- [45] Satoh Fumiko, Itoh Satoh. Single sign-on architecture with dynamic tokens, IBM Research, Tokyo Research laboratory, Japan. *Saint Symposium on Applications and the Internet*, page 197, 2004.
- [46] Schneier Bruce. Attack trees. *Dr. Dobb's Journal*, December 1999.
- [47] Shaked Yaniv and Wool Avishai. Cracking the Bluetooth PIN In. *Proceedings of the Third Annual International Conference on Mobile Systems, Applications and Services: MobiSys*, pages 39-50, June 2005.
- [48] Sonnenberg Alan. SSO: Enabling an effective password policy, Imprivata inc. 2003.
- [49] Steffan Jan and Schumaker Markus. Collaborative attack modeling. *In proceedings of the 2002 ACM Symposium on Applied Computing*, pages 253–259, 2002.
- [50] Sun Microsystems. Java 2 Platform, Micro Edition. 2002.
- [51] Swiderski Frank, Snyder Window. *Threat modeling*. Microsoft professional Microsoft press, 2004.
- [52] USB implementers' forum. USB Device class definition for human interface devices (HID) Firmware specification version 1.11. 2001. [http://www.usb.org/developers/devclass\\_docs/HID1\\_11.pdf](http://www.usb.org/developers/devclass_docs/HID1_11.pdf) (Last visited April 2006).
- [53] USB implementers' forum. USB HID Usage Tables v1.12. 2005. [http://www.usb.org/developers/devclass\\_docs/Hut1\\_12.pdf](http://www.usb.org/developers/devclass_docs/Hut1_12.pdf) (Last visited April 2006).
- [54] Utimaco Software. SafeGuard Advanced Security. <http://www.utimaco.com/C12570CF0030C00A/CurrentBaseLink/W26K9K2R4060BELUK>. (Last visited April 2006).
- [55] Vanio Juha T. Bluetooth security. 2000. <http://www.niksula.cs.hut.fi/~jiitv/bluesec.html> (Last visited April 2006).
- [56] Weiss Scott. *Handheld usability*. John Wiley & Sons, Ltd, 2002.
- [57] Wharton Cathleen, et al. The cognitive walkthrough method: A practitioners's guide. *In J. Nielsen and R.L Mack (eds.) Usability Inspection Methods*, New York: John Wiley, 1994.
- [58] Whitten Alma, Tygar J. D. Usability of security: A case study. *Technical Report CMU-CS-98-155, Carnegie Mellon University*, December 1998.

## A Technical problems

In this appendix, some of the technical problems encountered during implementation of the prototype are discussed. This discussion might be useful for others who are going to recreate what is done in this thesis. The information could also be useful if somebody wants to further develop the prototype.

### A.1 Setting up the USART

One problem that took some time to figure out, was how to configure the USART (serial port). The C18 compiler has several useful libraries. Full documentation of the C18 library functions is found in [27]. For serial communication, the `usart.h` file has to be included in the source code. In the prototype developed in this thesis, the following parameters are used in the `openUSART()` function:

```
OpenUSART( USART_TX_INT_OFF & Do not transmit interrupt
           USART_RX_INT_ON   & Receive interrupt
           USART_ASYNC_MODE  & Run in asynchronous mode
           USART_EIGHT_BIT   & 8-bit transmit/receive
           USART_CONT_RX     & Continuous reception
           USART_BRGH_HIGH,   High Baud rate
           77 );              Value to baud rate generator
```

The main problem was to find out what value to pass to the baud rate generator. This value is calculated from the following formula in high speed mode (`USART_BRGH_HIGH`):  $spbrg = FOSC / (\text{Baud\_rate} * 16) - 1$ . Where `FOSC` is the operating frequency of the device and `Baud_rate` is the desired baud rate.

In the documentation from Microchip, there are listed several operating frequencies such as 20 MHz crystal, 48 Mhz high speed mode, 96 MHz PLL module, 6 MHz low speed mode and several other combinations of this. Finally we put the card under an oscilloscope. Here we were able to isolate on bit, and use this to calculate the frequency the card was running on. After some calculations, we found the frequency to be 48MHz. Finally we chose a baud rate of 38 400 bps. When we plug all these numbers in to the formula, we get 77 as the vale to be passed to the baud rate generator.

Also, we had to set the correct configuration bits in the MPLAB IDE found in the Configure menu under Configuration Bits. The six first values are important to configure correct. Table 7 shows how these bits needs to be set.

Any program/device that needs to communicate with this device over RS232 (ex HyperTerminal in MS Windows) have to be set up with the following values:

```
Bits per second: 38400
Data bits:      8
Parity:         none
Stop bits:      1
Flow control:   none
```

Address	Value	Category	Setting
300000	24	Full-Speed USB Clock Source Selection	Clock src from 96MHz PLL/2
		CPU System Clock Postscaler	[OSC1/OSC2 Src: /1][96MHz PLL Src: /2]
		96MHZ PLL Prescaler	Divide by 5 (20MHz input)
300001	0F	Oscillator	HS: HS+PLL, USB-HS
		Fail-Safe Clock Monitor Enable	Disabled
		Internal External Switch Over Mode	Disabled

Table 7: The first six configuration bits that have to be altered. The remaining bits can keep their default value Found under Configure->Configuration Bits in the MPLAB IDE

## A.2 Finding services and connecting to the f2m Serial Port Plug

When implementing the Bluetooth module of the MIDlet, we had some trouble discovering services on the device. The device was discovered, but no services were found on it. After consulting with the support at free2move informed us that the f2m device can not be in configuration mode while a service search is being performed. The device is in configuration mode when the configuration software is running. In order to make service discovery work properly, the configuration software must be closed.

We also had some trouble sending data to the f2m device from the mobile phone. The connection was established fine, but no data were received. There were two things that had to be done in order to solve this problem. First, I had to install new firmware on the device. The new version is 3.04. The files needed for this upgrade will be made available when contacting the support at f2m<sup>1</sup>.

Second, a delay had to be added after the output stream connection was established, but before any data is sent. Below is a snippet of the code that shows where the delay needs to be placed in the code:

```
String data = "Hello World";
StreamConnection connection = (StreamConnection)Connector.open( url );
OutputStream os = connection.openOutputStream();
try{
    Thread.sleep( 1000 ); // Sleep before sending data to make sure connection is ready
catch( Exception e ) { e.printStackTrace(); }
os.write( data.getBytes() );
os.flush();
os.close();
connection.close();
```

If this delay is not present, the MIDlet will send data immediately after the the output stream is open.

---

<sup>1</sup> support@free2move.se



## B Descriptor making the USB demo board work as keyboard

```
/* Device Descriptor */
0x0D,    // Size of descriptor in bytes
0x01,    // DEVICE descriptor type
0x0200,  // USB Spec Release Number in BCD format
0x00,    // Class Code
0x00,    // Subclass code
0x00,    // Protocol code
0x08,    // Max packet size for EP0 8 bytes
0x04D8,  // Vendor ID
0x0001,  // Product ID
0x0001,  // Device release number in BCD format
0x01,    // Manufacturer string index
0x02,    // Product string index
0x00,    // Device serial number string index
0x01     // Number of possible configuration

/* Interface Descriptor */
0x09,    // Size of this descriptor in bytes
0x04,    // INTERFACE descriptor type
0,       // Interface Number
0,       // Alternate Setting Number
1,       // Number of endpoints in this interface
0x03,    // Class code, Human Interface device (HID)
0x01,    // Subclass code, Boot interfaces subclass
0x01,    // HID protocol keyboard
0        // Interface string index

/* HID Class-Specific Descriptor */
0x07,    // Size of this descriptor in bytes
0x21,    // HID descriptor type
0x0101,  // HID Spec Release Number in BCD format
0x00,    // Country Code
0x01,    // Number of class descriptors
0x22,    // Report descriptor type
0x3B     // Size of the report descriptor

/* REPORT DESCRIPTOR */
0x05, 0x01, //Usage Page (Generic Desktop)
0x09, 0x06, // Usage (Keyboard)
0xA1, 0x01, // Collection (Application)
```

```
0x05, 0x07, // Usage Page (key codes), use keycodes
0x19, 0xE0, // Usage Minimum
0x29, 0xE7, // Usage Maximum
0x15, 0x00, // Logical Minimum (0)
0x25, 0x01, // Logical Maximum (1)
0x75, 0x01, // Report Size (1)
0x95, 0x08, // Report Count (8), 8 byte report
0x81, 0x02, // Input (Data, Variable, Absolute)
0x81, 0x01, // Input (Constant)
0x19, 0x00, // Usage minimum(0)
0x29, 0x65, // Usage maximum(101)
0x15, 0x00, // Logical Minimum
0x25, 0x65, // Logical Maximum
0x75, 0x08, // Report Size (8)
0x95, 0x06, // Report Count (6)
0x81, 0x00, // Input(data,array)=Keycode Bytes
0x05, 0x08, // Usage Page (LED)
0x19, 0x01, // Usage minimum(1)
0x29, 0x05, // Usage maximum(5)
0x15, 0x00, // Logical Minimum (0)
0x25, 0x01, // Logical Maximum (1)
0x75, 0x01, // Report size(1)
0x95, 0x05, // Report Count(5)
0x91, 0x02, // Output (Data, Var, Abs)=LEDs
0x95, 0x03, // Report Count(3)
0x91, 0x01, // Output (Constant) = Pad(3 bits)
0xC0      // End Collection
```

## C Scenario used in the usability test

### Mobil Single sign-on

Masteroppgave HiG våren 2006

Mats Byfuglien

Takk for at du tar deg tid til å delta i dette eksperimentet. Forsøket vil ta mellom 20 til 30 minutter. All data behandles anonymt. Eksperimenter er fullstendig frivillig, og du står fritt til å trekke deg når som helst.

Mobil single sign-on er en ny måte å håndtere passord på. Alle dine brukernavn og passord lagres sikkert på mobiltelefonen. Når du vil logge inn et sted trenger du bare å plugge inn en USB enhet i PCen og velge aktuell konto på mobilen. Systemet vil da sende brukernavnet og passordet trådløst til datamaskinen og logge deg på automatisk.

**Bakgrunn:** Du er på reise og går inn på en nettkafé for å sjekke status på noen av brukerkontoene dine.

- 1) Logg deg inn på PCen ved hjelp av mobilen. Kontoen for å logge inn på Windows er: WinXp
- 2) Du ønsker å sjekke om du har fått noe ny e-post på Hotmail kontoen din. Bruk Internet Explorer til å gå inn på [www.hotmail.com](http://www.hotmail.com) og logg deg inn via mobilen.
- 3) Under favoritter i Internet Explorer ligger det en nettauksjon hvor du allerede er registrert som bruker. Velg Nettauksjon fra Favoritter. Når siden er oppe, kan du logge inn ved hjelp av mobilen. Gå så inn på brukerprofilen og sjekk status på auksjonene som er lagt ut.
- 4) Du har nylig opprettet en brukerkonto på [www.diskusjon.no](http://www.diskusjon.no). Denne kontoen har du ikke lagret på mobilen ennå. Lagre denne kontoen på mobilen med brukernavn: **ssoHiG** og passord: **32z3aihm**
- 5) Gå inn på [www.diskusjon.no](http://www.diskusjon.no) og logg deg inn ved hjelp av mobilen og kontoen du nettopp lagret. Når du har logget inn, sjekk status på dine innlegg ved å gå inn på kontrollpanelet på høyre side.
- 6) Dette forumet var kanskje ikke like bra som du hadde tenkt deg. Derfor ønsker du ikke å ha denne kontoen lenger. Prøv å slette den kontoen du nettopp lagret.
- 7) Ta fem minutter og svar på spørreundersøkelsen på neste side

Takk for hjelpen



## D Survey used in the usability test

### Spørreundersøkelse Mobil Single sign-on

1 Alder: \_\_\_\_\_

2 Kjønn:

Mann

Kvinne

3 Skolegang:

Ungdomsskole

Videregående Allmenn

Videregående Yrkesfag

Høgskole 3 år

Høgskolen 5 år

Universitet

4 Yrke: \_\_\_\_\_

5 Har du mobil?

Ja

Nei

6 Hvor ofte bruker du følgende tjenester på mobilen:

	Daglig	Ukentlig	Månedlig	Sjeldnere	Aldri
Ring	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
SMS	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
MMS	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Spill	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Ta bilder	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Kalender	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Laste ned ringetoner	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Laste ned bilder	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Værmelding via WAP	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Nyheter via WAP	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Andre WAP-tjenester	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
E-post	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

7 Har du hørt begrepet single sign-on før?

- Ja
- Nei
- Vet ikke

8 Har du brukt andre single sign-on løsninger?

- Ja
- Nei

Hvis ja:

Hvilke?

---

---

Hvordan fungerte de løsningene i forhold til den du har testet nå?

- Mye bedre
- Bedre
- Omtrent likt
- Dårligere
- Mye dårligere

9 Anslagsvis hvor mange brukerkontoer har du som krever brukernavn og passord?

- Ingen
- 1 - 5
- 6 - 10
- 11 - 15
- 16 - 20
- Mer enn 20

**10** Hender det at du bruker samme passord på flere brukerkontoer?

Ja

Nei

**11** Hvordan håndterer du passordene dine i dag?

Husker dem

Skriver de ned og oppbevarer dem på et sikkert sted

Skriver de ned og lagrer dem usikret

Annet

**12** Mener du at dine passord er sikre?

(Vil en angriper klare å gjette dine passord uten for store anstrengelsers?)

Ja

Nei

Usikker

**13** Hva synes du om single sign-on konseptet du nettopp har testet som helhet?

Svært bra

Bra

Middels

Dårlig

Svært dårlig

**14** Hvordan synes du log-in biten av systemet fungerte?

Svært bra

Bra

Middels

Dårlig

Svært dårlig

**15** Hvordan syns du passordhåndteringen fungerte (legge til/slette konto)?

Svært bra  
 Bra  
 Middels  
 Dårlig  
 Svært dårlig

**16** Kunne du tenke deg å bruke en slik løsning til daglig?

Ja  
 Nei  
 Vet ikke

**17** Gi en vurdering av følgende utsagn:

	Helt uenig	Delvis uenig	Hverken eller	Delvis enig	Helt enig
Denne løsningen kunne jeg tenke meg å bruke til daglig	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Opprettelsen av brukerkontoer fra mobilen fungerer tilfredsstillende	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Sletting av brukerkontoer fra mobilen fungerer tilfredsstillende	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Jeg er komfortabel med å lagre brukernavn/passord på mobilen	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Jeg syns det er bra å ha alle brukernavn/passord samlet på et sted	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Jeg er bekymret for sikkerhetsaspektet rundt løsningen	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Systemet vil hjelpe meg å holde orden på mine brukernavn/passord	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
For å kunne bruke denne løsningen fast, vil jeg ha en backup løsning tilgjengelig i tilfelle telefonen blir borte eller stjålet	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Systemet vil gjøre min hverdag sikrere	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Jeg er villig til å leve med den økte risikoen med å få nettleseren til å ta vare på mine passord i stedet	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>



**18** Hvor mye ville du eventuelt være villig til å betale for en slik løsning?  
(Pakken vil inneholde en USB enhet med Bluetooth og programvare til mobilen)

	Helt uenig	Delvis uenig	Hverken eller	Delvis enig	Helt enig
100 – 300 kr	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
300 – 500 kr	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
500 – 700 kr	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
700 – 900 kr	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
900 – 1100 kr	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
1100 – 1300 kr	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
1300 – 1500 kr	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>



## E Overview of the participants

	Age	Gender	Education	Occupation	Accounts	Password management today
P1	23	Male	5 year college	Student	16-20	Remembers them
P2	22	Female	3 year college	Social worker	> 20	Writes down, stores insecurely
P3	51	Male	3 year college	Teacher	1-5	Writes down, stores insecurely
P4	49	female	3 year college	Dental secretary	1-5	Remembers them
P5	29	Male	University	Dentist	1-5	Remembers them
P6	23	Male	5 year college	Student	6-10	Remembers them
P7	38	Male	5 year college	Student	16-20	Writes down, stores securely
P8	24	Male	5 year college	Student	11-15	Remembers them
P9	25	Female	5 year college	Student	6-10	Remembers them
P10	29	Male	5 year college	Student	11-15	Remembers them
P11	25	Male	5 year college	Student	6-10	Remembers them
P12	29	Male	5 year college	Student	1-5	Remembers them
P13	35	Male	3 year college	Sys. administrator	6-10	Remembers them
P14	28	Male	University	Web developer	> 20	Remembers them
P15	26	Male	3 year college	Business owner	> 20	Remembers them
P16	34	Male	University	Senior adviser	> 20	Remembers them
P17	30	Male	University	Researcher	> 20	Remembers them
P18	53	Female	3 year college	Financial adviser	1-5	Writes down, stores securely
P19	23	Male	3 year college	Student	> 20	Remembers them
P20	24	Male	5 year college	Student	11-15	Writes down, stores securely
P21	29	Male	3 year college	Military officer	11-15	Remembers them
P22	27	Female	University	Nurse	6-10	Remembers them
P23	26	Male	5 year college	Student	1-5	Remembers them
P24	43	Female	University	Teacher	6-10	Remembers them
P25	25	Female	3 year college	Student	6-10	Remembers them
P26	25	Female	3 year college	Support technician	11-15	Writes down, stores securely
P27	28	Female	3 year college	Nurse	1-5	Remembers them
P28	26	Female	3 year college	Seller	6-10	Writes down, stores insecurely

Table 8: Shows an overview of the participants in the user test with information about their background and how they manage their usernames and passwords today. The number of user accounts each participant have, is also shown in the table.