Using the mobile phone as a security token for unified authentication

Steffen Hallsteinsen
Department of Telematics,
Norwegian University of Science and Technology
Trondheim, Norway
steffeng@stud.ntnu.no

Ivar Jørstad
Ubisafe
Oslo, Norway
ivar@ubisafe.no

Do Van Thanh
Telenor R&I
Oslo, Norway
thanh-van.do@telenor.com

***Abstract* - The number of different identities and credentials used for authentication towards services on the Internet has increased beyond the manageable. Still, the most common authentication scheme is based on usernames and passwords. This is a weak authentication mechanism, which can be broken by eavesdropping on the network connection or by sloppy handling by the users (e.g. re-use of the same password for different services, writing down the passwords on paper etc.). Also, management of user credentials is a costly task for most companies, estimated by IDC to around 200-300USD pr. user/year. Hence, better solutions for simplified, yet secure authentication, is required in the future. This paper proposes and describes an authentication scheme based on a One-Time Password (OTP) MIDlet running on a mobile phone for unified authentication towards any type of service on the Internet.**

Keywords-component; , authentcation, mobility

I. INTRODUCTION

More and more services are becoming available on the Internet, and many of these services require authentication. The most widespread solution for electronic authentication today is the use of a username and password. As the number of services grows so does the number of username and password pairs that the user needs to remember. Already many people are experiencing that it is impossible to remember all the combinations and therefore they use the same pairs for all their services and choose passwords that are easy to remember. This strongly reduces the security of an already weak authentication mechanism. Several more secure solutions for electronic authentication exist, such as One-Time-Password (OTP) or Smart Card PKI solutions. They solve the security problem with passwords, but they most often increase the burden for the user. Extra hardware and software is required both for the user and service provider and they are often specific for each service so the user needs to deal with many different devices and procedures.

This paper addresses this problem by proposing an authentication scheme using the mobile phone as an authentication token. The GSM system is already well understood by the user and the proposed solution does not require any extra hardware device at the user side. By using the mobile phone as hardware token, strong authentication can be realized with a system and a device that the user already has and knows how to use.

The mobile phone has several advantages which can be exploited in user authentication:

- most people already have one
- the mobile phone has computing and communication capabilities that allows automation of the authentication process, relieving the user of this burden
- there are already good security mechanisms built into the GSM system that may be exploited.

The paper starts with describing the process of electronic authentication. It continues with explaining the benefits of OTP compared to normal passwords and what is required for using the mobile as an authentication token. Then the mobile OTP solution is described and analyzed in detail.

II. RELATED WORK

The Free Auth Project has made a MIDlet version of their OTP solution [1]. Their OTP generation is based on a time factor and requires that the client and server are

Second International Conference on Systems and Networks Communications (ICSNC 2007)
0-7695-2938-0/07 $25.00 © 2007 IEEE

synchronized for the solution to work. Time synchronization is not an easy task and the solution is vulnerable if the synchronization fails.

Several solutions exist where an OTP is sent to a mobile phone and used in addition with a static password or other authentication tokens. NordicEdge AB [2] uses this approach in their OTP solution. Even though additional hardware is avoided extra user operations are needed and the OTP is sent two times over the network in clear text.

The solution presented in this paper is automated to keep the user burden at a minimum. The application is automatically launched through push registry and SMS messages and the OTP is automatically transferred back to the server through Bluetooth and Internet. A combined use of the GSM network and a secure Internet connection to exchange the authentication messages results in a secure multi-channel authentication scheme.

III. Authentication Process

Authentication is the assurance that the communicating entity is the one that it claims to be [3]. A system can authenticate a user to determine if the user is authorized to perform an electronic transaction or get access to information or a system.

The authentication process begins with a client requesting a service which requires authentication. The client is asked to present a token to prove his identity. A token binds a user's identity together with a secret and is given to the user during registration. When a user presents his token during authentication the authenticating party can verify the user's identity since the token is unique for each user.

Tokens consist of one or more of the following factors:

1. something you know, e.g. passwords
2. something you have, e.g. hardware token
3. something you are, e.g. biometrics

Single-factor authentication such as passwords, which is only based on something you know, is shown to be open for many types of attacks. Eavesdropping, dictionary attacks and replay attacks can all be used to break such a system[4].

Strong authentication schemes rely on more than one factor. This dramatically increases the security. By combining something you know with something you have, an attacker needs to physically steal something in addition to learn the secret. Since the shared secret in such solutions are not sent over the network also learning the secret becomes much harder.

Multi-channel communication is another way to further improve the security of an authentication scheme. By using several communication channels both eavesdropping and man-in-the-middle attacks are much harder to accomplish since the attacker must control all the channels.

As will be shown, authentication using the mobile phone can exploit both these two improvements.

IV. Basis for Mobile Authentication

To be able to authenticate users through a mobile phone, certain main components must be in place. Fig. 1 shows the main components and the basic architecture needed for the solutions described later to work.

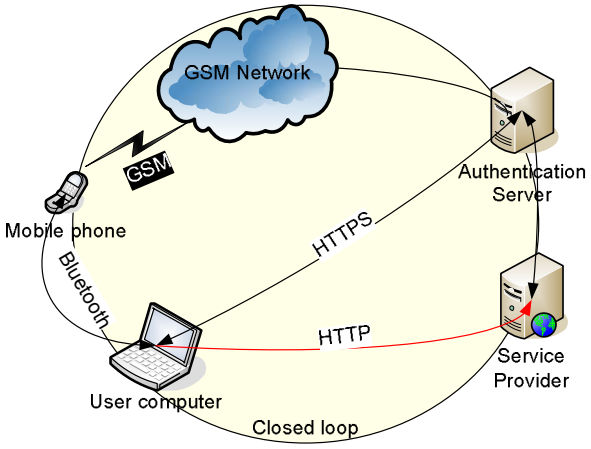

Figure 1. General architecture of mobile authentication

The user must have access to a computer connected to the Internet and be in possession of a mobile phone with a working SIM card. If the computer and mobile phone is equipped with Bluetooth higher usability can be obtained. Through the Internet browser on the computer the user can access web services provided by service providers. The service provider (SP) is connected to an Authentication Server (AS) that will handle the authentication on behalf of the SP. The AS is connected to the GSM network which enables it to communicate with the user's mobile phone and AuC. The AS is composed of two parts, an authenticator and an AAA server. The authenticator communicates with the client and relays messages to the AAA server which handles the authentication.

When designing an authentication scheme which uses two separate devices which communicate over two different networks it is very important to ensure that it is the same user that controls both devices. This is done by ensuring that there is a "closed loop" going through all the components involved in the authentication as illustrated in Fig. 1. The loop starts in the device requesting the service, goes through the network with Service Provider (SP) and Authentication Server (AS) and then via the mobile phone and back to the initial device either by user interaction or Bluetooth.

This closed loop can be realized in several ways. This paper describes how using the mobile phone as an OTP token does this.

V. One-Time Passwords

One-Time Password is one of the simplest and most popular forms of two-factor authentication today. For example, in large enterprises, Virtual Private Network access often requires the use of One-Time Password tokens for remote user authentication [5].

One-Time-Passwords solve many of the problems with user-chosen passwords. They are generated by a computer and can therefore appear totally random. In addition as the name suggests each password is only used once. That makes them invulnerable to replay attack and sniffing. By changing each time a password is needed OTP solutions introduce liveness. This is an important concept in security since it is possible to detect if someone is using old information.

The security of the OTP system is based on the non-invertability of a secure hash function. Such a function must be tractable to compute in the forward direction, but computationally infeasible to invert [6]. This hash function can be deployed in three different ways to generate the OTP. The first type applies the hash function a given number of times to generate a chain of OTPs where the next OTP is generated by taking the hash of the previous one. The second type is based on time-synchronization between the authentication server and the client. The current time is used as seed for the hash. As long as the server and client are synchronized they generate the same OTP. The third type uses a challenge, e.g. a random number, as seed to the hash function.

VI. THE MOBILE OTP SOLUTION

The mobile OTP solution combines the simple and secure OTP principle with the ubiquitousness of a GSM mobile phone. A Java MIDlet which can be installed on any Java enabled mobile phone transforms the phone into a secure OTP token which can be used to log in to any service on the Internet.

The solution is based on a simple challenge-response protocol. When the user wants to log in he presents his username, and a challenge is sent to the user's mobile phone. The OTP MIDlet installed on the phone generates an OTP from the challenge and sends it back to the server. The server verifies that the OTP is correct and the user is authenticated.

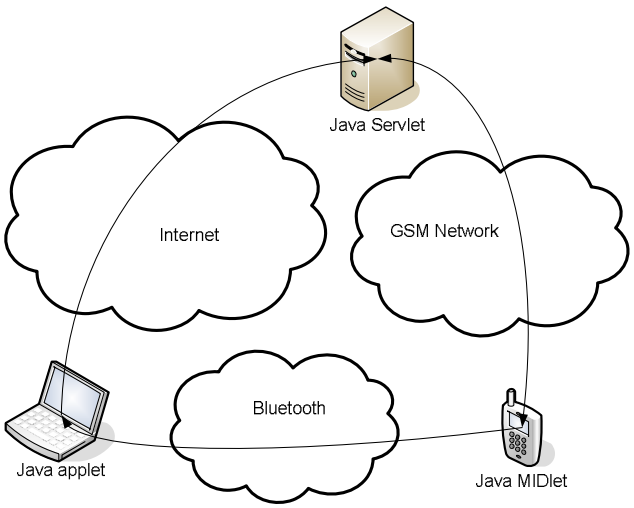

Figure 2. Component architecture

Fig. 2 illustrates the components architecture and communication interfaces used in the OTP solution. A Java MIDlet will be running on the mobile phone to handle OTP generation. A Java applet will run in the browser on the user's computer and relay the OTP from the MIDlet to the AS. The MIDlet and applet communicates over Bluetooth. A Java servlet will run in the AS and handle all client communication. The servlet will communicate with the MIDlet using short messages (SMS) over the GSM network and with the applet over a HTTPS connection.

A. Java Midlet

The java MIDlet is the core component on the client side. It enables the mobile phone to act as an OTP token and automatically exchange authentication messages with the AS and computer.

To be able to download the MIDlet to the mobile phone a user must register at the AS. This is done through an Internet page secured with TLS. When the registration is finished a push message is sent the user's phone which then automatically downloads and installs the MIDlet only requiring the user's agreement.

After installation the AS initializes a key exchange with the MIDlet by sending an SMS including some public values to the user's phone. Upon receiving this SMS the MIDlet is automatically started and the user is asked to enter two passwords.

The first password is chosen by the user and is used to ensure two-factor authentication, prevent misuse of stolen phones and act as an encryption key.

The second password is a short OTP generated by the server and displayed on the webpage when the MIDlet is downloaded. This is used once to authenticate the key exchange. The key exchange is further described later.

SMS communication

The Wireless Messaging API (WMA) [7] enables a MIDlet to automatically send and receive SMS messages. By using this together with the Push Registry function a MIDlet can be automatically activated upon receiving a SMS message on a specific port. The push registry's behavior can be described in three steps:

1. The MIDlet is registered with a port and a protocol such that, if any message arrives in the specified port with the protocol mentioned, the application management software (AMS) on the mobile phone delivers it to the MIDlet. In this case the protocol used is SMS over the GSM network. The registration is done statically using the Java ME application descriptor (JAD) file.
2. The server sends a message to the specific mobile phone using the particular protocol and port where the MIDlet application is registered to listen.
3. After the message is delivered to the mobile phone, the AMS calls the MIDlet application,

which has registered to listen to that particular port and particular protocol. Once the message is delivered to the MIDlet, it is the application's responsibility to process the message accordingly [8].

Key exchange

To make the OTP generation as secure as possible a strong secret key is used together with the challenge. This secret key must be shared between the client and server. The Simple Password Exponential Key Exchange (SPEKE) protocol [9] is used to securely exchange this key. SPEKE is an improved version of Diffie-Hellman (DH) [10] which prevents Man-In-The-Middle attacks by basing the DH generator on a hash of password. An attacker who is able to read and modify all messages between the client and server cannot learn the shared key and cannot make more than one guess for the password in each interaction with a party that knows it. A simple form of SPEKE requiring only two messages is illustrated in Fig. 3.

The AS starts the key exchange by generating a large and randomly selected safe prime p and computes

$$g = \text{hash}(s)^2 \bmod p$$

where s is the short OTP displayed in the browser after registration. Then the server computes

$$Ks = g^a \bmod p$$

where a is a secret random number and sends p and Ks to the MIDlet with an SMS.

Upon receiving the SMS from the server the MIDlet generates

$$g = \text{hash}(s)^2 \bmod p$$

$$Kc = g^b \bmod p$$

where s is the short OTP entered by the user at startup and b is a secret random number. Then the MIDlet sends Kc to the AS and computes the secret key

$$K = (Ks)^b \bmod p.$$

When the server receives Kc from the MIDlet it computes

$$K = (Kc)^a \bmod p$$

At this point the AS and the MIDlet shares a strong secret key K which can be used in OTP generation.

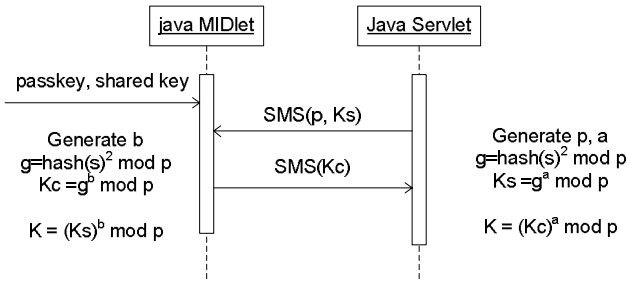

Figure 3. Key exchange

To prevent any attacker from learning the secret key it is encrypted with an expanded version of the user chosen password as an encryption key before it is stored in J2ME's Record Store[11]. The Record Store is used in private mode to deny any other MIDlet from getting access to the records. A hash of the user password is also stored so the MIDlet can verify that the correct password is entered at startup.

OTP generation

The OTP is generated from a hash of a concatenation of the challenge and the secret key.

$$OPT = \text{hash}(\text{challenge} \,||\, \text{secret key})$$

When the user needs to be authenticated the AS generates a challenge and computes the corresponding OTP. Then it computes a message authentication code (MAC) of the OTP. The challenge and MAC is sent to the MIDlet as a SMS message while the OTP is kept at the AS.

As described in the previous part the MIDlet is automatically activated when the message arrives and the user is asked to enter his password. The MIDlet generates the hash of the password and checks it against the stored value. If they match the MIDlet uses the password to decrypt the secret key. The challenge and the secret key is concatenated and fed into the hash function to produce the OTP. Then a MAC is calculated over the OTP and checked against the MAC received. If the MAC does not match the authentication is aborted and the process must be restarted. If the values match the MIDlet starts a service discovery to set up a Bluetooth connection with the applet running on the computer. When the connection is set up the MIDlet sends the OTP to the applet. Then the connection is terminated and the MIDlet exits. Fig.4 illustrates the authentication process.

If a mobile phone without Bluetooth is used the user will have to enter the OTP in the browser manually. Then the OTP will be displayed on the mobile screen if the MAC is OK. Using a hash directly as an OTP is not suitable if the user must type it in manually. Therefore the hash must be truncated into an 8 byte long string and transformed into a more user-friendly format like a character string containing only letters and numbers.

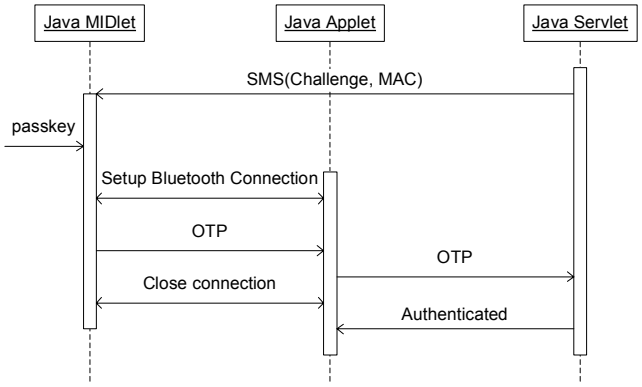

Figure 4. Authentication process

B. *Java Applet*

The java applet is necessary to enable automatic transfer of the OTP from the mobile phone to the computer. The applet will act as a Bluetooth server which the MIDlet can contact and setup a connection with. By realizing this component as an applet no additional software has to be installed on the user's computer. The mobile phone and computer must be paired before a connection can be setup. During the Bluetooth pairing a 16 byte pass phrase must be entered to secure the Bluetooth connection. When the applet has received the OTP from the MIDlet it contacts the servlet and sends the OTP. If the OTP is correct the applet is notified by the servlet and redirects the session back to the service provider.

C. *Java Servlet*

The Java servlet acts as the AS' interface towards the clients. When a user downloads a MIDlet the AS initiates the key exchange by sending its public values to the MIDlet by SMS. The servlet uses a SMS gateway to send and receive SMS messages. The gateway can be configured to send messages to specific ports. Therefore the servlet can send a SMS message to the port specified in the MIDlet. When the key exchange is finished the secret key is stored in the AS' database together with the user-profile. When authentication is requested the user is asked to enter his username. The AS gets the profile for this username and generates a challenge based on the profile. After generating the challenge the OTP and MAC is generated and the challenge and MAC is sent to the MIDlet by SMS. When the servlet receives the OTP from the applet it checks that it matches the OTP it created itself. If the values match the user is successfully authenticated. If the solution is part of a single sign-on environment an assertion is created that can be reused for several authentication procedures.

VII. SECURITY EVALUATION

This chapter contains a security evaluation of the solution. The evaluation is divided into four parts. Each part consists of the possible threats to this part and how the solution encounters these threats.

Fig. 5 illustrates the different components of the system that are vulnerable to attacks.

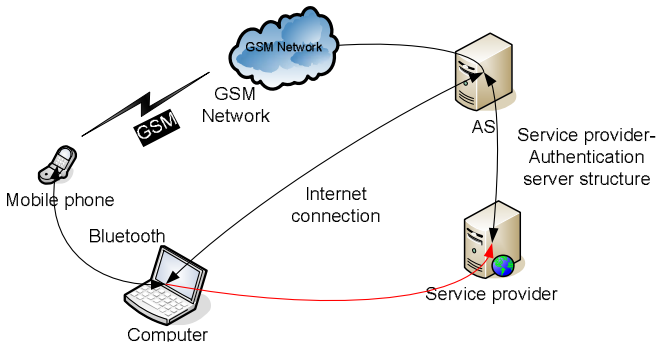

Figure 5. Threat model

A. *User devices*

- An attacker might try to steal a mobile phone to impersonate a valid user
- An attacker might try to tamper with a mobile phone to learn the secret key or password.
- Malicious software like worms and key loggers can gain access to sensitive information stored on the computer or typed on the keyboard. Malware is a big problem for everyone connected to the Internet.

Countermeasures

- The application is protected with a user chosen password. This results in a two-factor authentication preventing misuse of stolen phones.
- The secret key stored on the phone is encrypted with a strong encryption algorithm. A hash of the password is stored and not the password itself. The Java Record Store is used in private mode denying any other application access to the records.
- The user computer does not handle any secret information except when relaying the OTP. An OTP is protected against misuse by its properties.

B. *Bluetooth connection*

- An attacker might try to listen in on the Bluetooth connection to learn sensitive information

Countermeasures

- The Bluetooth connection is secured with the

use of a passkey during paring. The only information being transferred over this connection is the OTP which is in itself secure.

C. Internet connection

- An attacker might try to impersonate a user by guessing the password.
- An attacker might try to hijacking the session to get unauthorized access.
- An attacker might try to eavesdrop on the connection to learn the password or other sensitive data.
- An attacker might try to mount a Man-in-the-middle (MITM) attack.

Countermeasures

- OTPs are not possible to guess since they appear as random numbers
- Eavesdropping, hijacking and MITM attacks are prevented by securing the connection to the AS with TLS

D. GSM Network

- A weakness in the GSM system is that that the server is not authenticated. This makes it possible to mount a MITM attack on the GSM network [12].

Countermeasures

- The key exchange is authenticated with a short OTP. This prevents a MITM for being able to learn anything useful during the key exchange.
- Tampering with the authentication values sent over the GSM network will only result in authentication failure since they are protected by a MAC. The MAC is also used to authenticate the server resulting in mutual authentication.
- Picking up the challenge and MAC sent over the GSM network will not enable an attacker to learn the key. The challenge is only a random number and the MAC is a hash generated of a strong cryptographic hash function which is not possible to revert.

VIII. CONCLUSION & FUTURE WORKS

This paper proposes a novel secure authentication solution based on the OTP principle. It shows how an OTP MIDlet turns the mobile phone into a secure and user-friendly authentication token. The proposed solution eliminates weaknesses of existing similar solutions such as the need for time synchronization, possibility of man-in-the-middle attacks and substantially improves the user-friendliness.

Using the mobile phone as an authentication token provides a very flexible and secure solution. It is therefore a very good alternative to the existing authentication solutions such as username and password.

The proposed solution can also be an alternative to existing OTP solutions which require extra hardware and therefore put an extra burden on the user and increase the costs for the companies/service providers.

This solution can be securely used in most services available on the Internet today which require authentication. This can be Internet banking, corporate intranet, Internet stores and e-Government applications.

Solutions that utilize the mobile phone for authentication are very likely to become a popular choice for many users and companies in the near future.

The paper also provides a thorough security evaluation which discusses the threat model and security properties of the proposed OTP solution.

As part of future works, the solution will now be properly integrated with an identity management platform and tested in a Single-Sign-On environment.

REFERENCES

1. FreeAuthProject. The FreeAuth Project. [cited 2007 March]; Available from: http://www.freeauth.org/site.
2. NordicEdge. NordicEdge. [cited 2007 March]; Available from: http://www.nordicedge.se/.
3. Stallings, W., Network security essentials. 2003: Prentice Hall.
4. NIST, Electronic authentication guideline 2006.
5. M'Raihi, M.B., F. Hoornaert, D. Naccache, O. Ranen, RFC 4226 HOTP An HMAC-Based One-Time Password Algorithm. 2005.
6. Haller, N., et al., A One-Time Password System. 1998, IETF.
7. Ortiz, C.E. The Wireless Messaging API. 2002 [cited 2007 March]; Available from: http://developers.sun.com/techtopics/mobility/midp/articles/wma/.
8. Roy, S. Push messages that automatically launch a Java mobile application. 2006 April 17th [cited 2007 March].
9. Jablon, D., Strong Password-Only Authenticated Key Exchange. Computer Communication Review, ACM SIGCOMM, 1996. vol. 26(no. 5).
10. Rescorla, E., RFC 2631 Diffie-Hellman Key Agreement Method. 1999.
11. Microsystems, S. Java ME Technology APIs & Docs. [cited 2007 March]; Available from: http://java.sun.com/javame/reference/apis.jsp.
12. U.Meyer and S.Wetzel, A Man-in-the-Middle Attack on UMTS. In Proceedings of the 2004 ACM Workshop on Wireless Security, 2004.