International Journal of Wireless Communications and Mobile Computing
2013; 1(1): 26-34
Published online June 10, 2013 (http://www.sciencepublishinggroup.com/j/wcmc)
doi: 10.11648/j.wcmc.20130101.15

Two aspect authentication system using secure mobile devices

S. Uvaraj[1], S. Suresh[2], N. KannaiyaRaja[3]

[1]Arulmigu Meenakshi Amman College of Engineering
[2]Sri Venkateswara College of Engineering
[3]Defence Engineering College, Ethiopia

Email address:

ujrj@rediffmail.com(S. Uvaraj), ss12oct92@gmail.com(S. Suresh), kanniya13@hotmail.co.in(N. KannaiyaRaja)

To cite this article:

S. Uvaraj, S. Suresh, N. KannaiyaRaja, Two Aspect Authentication System Using Secure Mobile Devices. *International Journal of Wireless Communications and Mobile Computing.* Vol. 1, No. 1, 2013, pp. 26-34. doi: 10.11648/j.wcmc.20130101.15

Abstract: Mobile devices are becoming more pervasive and more advanced with respect to their processing power and memory size. Relying on the personalized and trusted nature of such devices, security features can be deployed on them in order to uniquely identify a user to a service provider. In this paper, we present a strong authentication mechanism that exploits the use of mobile devices to provide a two-aspect authentication system. Our approach uses a combination of one-time passwords, as the first authentication aspect, and credentials stored on a mobile device, as the second aspect, to offer a strong and secure authentication approach. By Adding an SMS-based mechanism is implemented as both a backup mechanism for retrieving the password and as a possible mean of synchronization. We also present an analysis of the security and usability of this mechanism. The security protocol is analyzed against an adversary model; this evaluation proves that our method is safe against various attacks, most importantly key logging, shoulder surfing, and phishing attacks. Our simulation result evaluation shows that, although our technique does add a layer of indirectness that lessens usability; participants were willing to trade-off that usability for enhanced security once they became aware of the potential threats when using an untrusted computer.

Keywords: Computer Network Security, Mobile Handsets, One-Time Password, Smart Mobile Phones

1. Introduction

Today security concerns are on the rise in all areas such as banks, governmental applications, healthcare industry, military organization, educational institutions, etc. Government organizations are setting standards, passing laws and forcing organizations and agencies to comply with these standards with non-compliance being met with wide-ranging consequences. There are several issues when it comes to security concerns in these numerous and varying industries with one common weak link being passwords. Mobile-OTP was introduced in 2003. As of 2012 there are more than 40 independent implementations of the Mobile-OTP algorithm making it a de facto standard for strong mobile authentication. Most systems today rely on static passwords to verify the user's identity. However, such passwords come with major management security concerns. Users tend to use easy-to-guess passwords, use the same password in multiple accounts, write the passwords or store them on their machines, etc. Furthermore, hackers have the option of using many techniques to steal passwords such as shoulder surfing, snooping, sniffing, guessing, etc.

Several 'proper' strategies for using passwords have been proposed [1]. Some of which are very difficult to use and others might not meet the company's security concerns. Two aspect authentication using devices such as tokens and ATM cards has been proposed to solve the password problem and have shown to be difficult to hack. Two aspect authentication also have disadvantages which include the cost of purchasing, issuing, and managing the tokens or cards. From the customer's point of view, using more than one two- aspect authentication system requires carrying multiple tokens/cards which are likely to get lost or stolen.

Mobile phones have traditionally been regarded as a tool for making phone calls. But today, given the advances in hardware and software, mobile phones use have been expanded to send messages, check emails, store contacts, etc. Mobile connectivity options have also increased. After standard GSM connections, mobile phones now have infra-red, Bluetooth, 3G, and WLAN connectivity. Most of us, if not all of us, carry mobile phones for communication

purpose. Several mobile banking services available take advantage of the improving capabilities of mobile devices. From being able to receive information on account balances in the form of SMS messages to using WAP and Java together with GPRS to allow fund transfers between accounts, stock trading, and confirmation of direct payments via the phone's micro browser [12].

Installing both vendor-specific and third party applications allow mobile phones to provide expanded new services other than communication. Consequently, using the mobile phone as a token will make it easier for the customer to deal with multiple two aspect authentication systems; in addition it will reduce the cost of manufacturing, distributing, and maintaining millions of tokens. We have developed a scheme that enables the use of mobile devices for authenticating users to a web service provider. The approach provides a two- aspect authentication mechanism by combining one-time passwords (OTPs), as the first authentication aspect, together with encrypted user credentials stored on a mobile phone as the second authentication aspect. In this approach, we treated the mobile device as a trusted digital wallet to securely encrypt and store users' long term credentials. These credentials (i.e., her username and password) are encrypted using the public key of the service provider, stored on the mobile device, and are transferred to the service provider when needed. The storage of long term credentials on the mobile device enables users to use stronger passwords for their accounts, as they don't need to remember and retype the passwords for each and every login time. One-time passwords further protect the stored credentials on the cell phone, if stolen or lost, by requiring additional information at the time of log in. In addition to a description of this authentication protocol, we present the results of security and usability evaluations of our two-aspect mobile authentication system. The security analysis evaluates the mobile authentication mechanism against an adversary model. Our analysis shows that the security of our devised method is improved over similar authentication approaches that use mobile devices [MvO07, MWL04, PKP06, WGM04], due to the addition of the OTP which leads to having a strong authentication mechanism. Furthermore, the results of our usability study show that our participants were willing to adopt the new technology once became aware of the potential threats to their passwords when using untrusted computers. Participants indicated they would accept a lower level of usability in return for the higher level of security of the mobile technology. However, for this new technology to be a complete replacement to conventional username/password based systems, it should be signicantly simpler.

In this paper, we propose and develop a complete two aspect authentication system using mobile phones instead of tokens or cards. The system consists of a server connected to a GSM modem and a mobile phone client running a J2ME application. Two modes of operation are available for the users based on their preference and constraints. The first is a stand-alone approach that is easy to use, secure, and cheap. The second approach is an SMS-based approach that is also easy to use and secure, but more expensive. The system has been implemented and tested.

In the next section we provide a related works about authentication aspect s and existing two aspect authentication systems. Section 3 & 4 describes the proposed system design, the OTP algorithm, client, server, and the database. Section 6 & 7 provides simulation setup and results of testing the system. Section 8 concludes the paper and provides future work.

2. Related Works

By definition, authentication is the use of one or more mechanisms to prove that you are who you claim to be. Once the identity of the human or machine is validated, access is granted.

Three universally recognized authentication aspect s exist today: what you know (e.g. passwords), what you have (e.g. ATM card or tokens), and what you are (e.g. biometrics). Recent work has been done in trying alternative aspects such as a fourth aspect, e.g. somebody you know, which is based on the notion of vouching [10]. Two aspect authentications [4] is a mechanism which implements two of the above mentioned aspects and is therefore considered stronger and more secure than the traditionally implemented one aspect authentication system. Withdrawing money from an ATM machine utilizes two aspect authentications; the user must possess the ATM card, i.e. what you have, and must know a unique personal identification number (PIN), i.e. what you know. Passwords are known to be one of the easiest targets of hackers. Therefore, most organizations are looking for more secure methods to protect their customers and employees. Biometrics are known to be very secure and are used in special organizations, but they are not used much in secure online transactions or ATM machines given the expensive hardware that is needed to identify the subject and the maintenance costs, etc. Instead, banks and companies are using tokens as a mean of two aspect authentication. A security token is a physical device that an authorized user of computer services is given to aid in authentication. It is also referred to as an authentication token or a cryptographic token. Tokens come in two formats: hardware and software. Hardware tokens are small devices which are small and can be conveniently carried. Some of these tokens store cryptographic keys or biometric data, while others display a PIN that changes with time. At any particular time when a user wishes to log-in, i.e. authenticate, he uses the PIN displayed on the token in addition to his normal account password. Software tokens are programs that run on computers and provide a PIN that change with time. Such programs implement a One Time Password (OTP) algorithm. OTP algorithms are critical to the security of systems employing them since unauthorized users should not be able to guess the next password in the

sequence. The sequence should be random to the maximum possible extent, unpredictable, and irreversible. Aspects that can be used in OTP generation include names, time, seed, etc. Several commercial two aspect authentication systems exist today such as BestBuy's BesToken, RSA's SecurID [6], and Secure Computing's Safeword [2]. BesToken applies two-aspect authentication through a smart card chip integrated USB token. It has a great deal of functionality by being able to both generate and store users' information such as passwords, certificates and keys. One application is to use it to log into laptops. In this case, the user has to enter a password while the USB token is plugged to the laptop at the time of the login. A hacker must compromise both the USB and the user account password to log into the laptop.SecurID from RSA uses a token (which could be hardware or software) whose internal clock is synchronized with the main server. Each token has a unique seed which is used to generate a pseudo-random number. This seed is loaded into the server upon purchase of the token and used to identify the user. An OTP is generated using the token every 60 seconds. The same process occurs at the server side. A user uses the OTP along with a PIN which only he knows to authenticate and is validated at the server side. If the OTP and PIN match, the user is authenticated [8]. In services such as ecommerce, a great deal of time and money is put into countering possible threats and it has been pointed out that both the client and the server as well as the channel of communication between them is imperative [1]. Using tokens involves several steps including registration of users, token production and distribution, user and token authentication, and user and token revocation among others [6]. While tokens provide a much safer environment for users, it can be very costly for organizations. For example, a bank with a million customers will have to purchase, install, and maintain a million tokens. Furthermore, the bank has to provide continuous support for training customers on how to use the tokens. The banks have to also be ready to provide replacements if a token breaks or gets stolen. Replacing a token is a lot more expensive than replacing an ATM card or resetting a password. From the customer's prospective, having an account with more than one bank means the need to carry and maintain several tokens which constitute a big inconvenience and can lead to tokens being lost, stolen, or broken. In many cases, the customers are charged for each token. We propose a mobile-based software token that will save the organizations the cost of purchasing and maintaining the hardware tokens. Furthermore, will allow customers to install multiple software tokens on their mobile phones. Hence, they will only worry about their mobile phones instead of worrying about several hardware tokens.

3. System Design

In this paper, we propose a mobile-based software token system that is supposed to replace existing hardware and computer-based software tokens. The proposed system is secure and consists of three parts: (1) software installed on the client's mobile phone, (2) server software, and (3) a GSM modem connected to the server. The system will have two modes of operation:

• Connection-Less Authentication System: A onetime password (OTP) is generated without connecting the client to the server. The mobile phone will act as a token and use certain aspects unique to it among other aspects to generate a one-time password locally. The server will have all the required aspects including the ones unique to each mobile phone in order to generate the same password at the server side and compare it to the password submitted by the client. The client may submit the password online or through a device such as an ATM machine. A program will be installed on the client's mobile phone to generate the OTP.

• SMS-Based Authentication System: In case the first method fails to work, the password is rejected, or the client and server are out of sync, the mobile phone can request the one time password directly from the server without the need to generate the OTP locally on the mobile phone. In order for the server to verify the identity of the user, the mobile phone sends to the server, via an SMS message, information unique to the user. The server checks the SMS content and if correct, returns a randomly generated OTP to the mobile phone. The user will then have a given amount of time to use the OTP before it expires. Note that this method will require both the client and server to pay for the telecommunication charges of sending the SMS message.

3.1. OTP Algorithm

In order to secure the system, the generated OTP must be hard to guess, retrieve, or trace by hackers. Therefore, its very important to develop a secure OTP generating algorithm. Several aspects can be used by the OTP algorithm to generate a difficult-to-guess password. Users seem to be willing to use simple aspects such as their mobile number and a PIN for services such as authorizing mobile micropayments [9]. Note that these aspects must exist on both the mobile phone and server in order for both sides to generate the same password. In the proposed design, the following aspects were chosen:

• IMEI number: The term stands for International Mobile Equipment Identity which is unique to each mobile phone allowing each user to be identified by his device. This is accessible on the mobile phone and will be stored in the server's database for each client.

• IMSI number: The term stands for International Mobile Subscriber Identity which is a unique number associated with all GSM and Universal Mobile Telecommunications System (UMTS) network mobile phone users. It is stored in the Subscriber Identity Module (SIM) card in the mobile phone. This number will also be stored in the server's database for each client.

• Username: Although no longer required because the IMEI will uniquely identify the user anyway. This is used together with the PIN to protect the user in case the mobile

phone is stolen.

• PIN: This is required to verify that no one other than the user is using the phone to generate the user's OTP. The PIN together with the username is data that only the user knows so even if the mobile phone is stolen the OTP cannot be generated correctly without knowing the user's PIN. Note that the username and the PIN are never stored in the mobile's memory. They are just used to generate the OTP and discarded immediately after that. In order for the PIN to be hard to guess or brute-forced by the hacker, a minimum of 8-characters long PIN is requested with a mixture of upper- and lower-case characters, digits, and symbols.

• Hour: This allows the OTP generated each hour to be unique.

• Minute: This would make the OTP generated each minute to be unique; hence the OTP would be valid for one minute only and might be inconvenient to the user. An alternative solution is to only use the first digit of the minute which will make the password valid for ten minutes and will be more convenient for the users, since some users need more than a minute to read and enter the OTP. Note that the software can modified to allow the administrators to select their preferred OTP validity interval.

• Day: Makes the OTP set unique to each day of the week.

• Year/Month/Date: Using the last two digits of the year and the date and month makes the OTP unique for that particular date. The time is retrieved by the client and server from the telecommunication company. This will ensure the correct time synchronization between both sides.

The above aspects are concatenated and the result is hashed using SHA-256 which returns a 256 bit message. The message is then XOR-ed with the PIN replicated to 256 characters. The result is then Base64 encoded which yields a 28 character message. The message is then shrunk to an administrator-specified length by breaking it into two halves and XOR-ing the two halves repeatedly. This process results in a password that is unique for a ten minute interval for a specific user. Keeping the password at 28 characters is more secure but more difficult to use by the client, since the user must enter all 28 characters to the online webpage or ATM machine. The shorter the OTP message the easier it is for the user, but also the easier it is to be hacked. The proposed system gives the administrator the advantage of selecting the password's length based on his preference and security needs.

3.2. Client Design

A J2ME program is developed and installed on the mobile phone to generate the OTP. The program has an *easy to-use* GUI that is developed using the NetBeans drag and drop interface. The program can run on any J2ME-enabled mobile phone. The OTP program has the option of (1) generating the OTP locally using the mobile credentials, e.g. IMEI and IMSI numbers, or (2) requesting the OTP from the server via an SMS message. The default option is the first method which is cheaper since no SMS messages are exchanged between the client and the server. However, the user has the option to select the SMS-based method. In order for the user to run the OTP program, the user must enter his username and PIN and select the OTP generation method. The username, PIN, and generated OTP are *never* stored on the mobile phone.If the user selects the connection-less method the username and PIN are used to locally generate the OTP and are discarded thereafter. The username and PIN are stored on the server's side to generate the same OTP. If the user selects the SMS-based method, the username and PIN, in addition to the mobile identification aspects, are encrypted via a 256-bit symmetric key in the OTP algorithm and sent via an SMS to the server. The server decrypts the message via the same 256-bit symmetric key, extracts the identification aspects, compares the aspects to the ones stored in the database, generates an OTP and sends the OTP to the client's mobile phone if the aspects are valid. The advantage of encrypting the SMS message is to prohibit sniffing or man-in-the-middle attacks. The 256-bit key will be extremely hard to brute-force by the hacker. Note that each user will have a pre-defined unique 256-bit symmetric key that is stored on both the server and client at registration time.

3.3. Database Design

A database is needed on the server side to store the client's identification information such as the first name, last name, username, pin, password, mobile IMEI number, IMSI number, unique symmetric key, and the mobile telephone number for each user. The password field will store the hash of the 10 minute password. It will not store the password itself. Should the database be compromised the hashes cannot be reversed in order to get the passwords used to generate those hashes. Hence, the OTP algorithm will not be traced.

3.4. Server Design

A server is implemented to generate the OTP on the organization's side. The server consists of a database as described in Section 3.C and is connected to a GSM modem for SMS messages exchange. The server application is multithreaded. The first thread is responsible for initializing the database and SMS modem, and listening on the modem for client requests. The second thread is responsible for verifying the SMS information, and generating and sending the OTP. A third thread is used to compare the OTP to the one retrieved using the connection-less method. In order to setup the database, the client must register in person at the organization. The client's mobile phone/SIM card identification aspects, e.g. IMEI/IMSI, are retrieved and stored in the database, in addition to the username and PIN. The J2ME OTP generating software is installed on the mobile phone. The software is configured to connect to the server's GSM modem in case the SMS option is used. A unique symmetric key is also generated

and installed on both the mobile phone and server. Both parties are ready to generate the OTP at that point.

4. Existing OTP Algorithm

Certain types of encryption in the networks, by their mathematical properties cannot be easily overcome by brute force. An example of this is the one-time password algorithm (OTP) [5], where every clear text bit has a corresponding key bit. One-time passwords rely on the ability to generate a truly random sequence of key bits. A brute force attack would eventually reveal the correct decoding, but also every other possible combination of bits, and would have no way of distinguishing one from the other. A small, 100-byte, one-time-password encoded string subjected to a brute force attack would eventually reveal every 100-byte string possible, including the correct answer, but possibly low chance. Here we have analyzed one-time password algorithm for a secure network [6] available today based on mobile authentication and we have listed the possible attacks [7] to the one-time password algorithm.

4.1. One-Time Password Algorithm

In existing one-time password algorithm, Java MIDlet is a client application and we assume that this runs in client mobile phones which can be able to receive one time passwords. A MIDlet is an application that uses the Mobile Information Device Profile (MIDP) of the Connected Limited Device Configuration (CLDC) for the Java ME environment. Typical applications include games running on mobile devices and cell phones which have small graphical displays, simple numeric keypad interfaces and limited network access over HTTP. This whole design describes the two main protocols used by Java MIDlet system. Initially, the user downloads the client (Java MIDlet) to his mobile phone. Then the client executes a protocol to register with both server and a service provider utilizing server system for user authentication. After the successful execution of the activation protocol the user can run the authentication protocol an unlimited number of times.

4.2. Activation Protocol

After the user has downloaded the client software from a service onto his mobile phone, he must activate the phone as an OTP receiver before it can be used for authentication to a web-based secure service [8]. The activation protocol takes place between five parties: the user, the user's mobile phone, the user's PC, the Server (SE), and the service provider (SP). The main steps of the protocol are summarized below.

1. The user authenticates himself to SP using credentials already known to SP.

2. When the user asks to activate his mobile phone as an OTP receiver, SP redirects the user's browser to SE with a URL that contains an activation request and a Secure Object [9].

3. SE verifies that the Secure Object comes from SP, and gets the user's phone number and other unique identification number like IMEI.

4. SE sends an activation code to the user's PC and an SMS message to the user's phone asking it to start the client software.

5. The mobile phone asks the user to enter the activation code, available on his PC, and transmits the code to SE.

6. SE verifies that the activation code is the same as the one sent to the PC, and sends a challenge to the mobile phone together with an encryption key K0 (The role of K0 is explained separately).

7. The user chooses a personal identification number (PIN) and enters it on the mobile phone, which generates a security code and a response. The response is the encryption of the challenge using the security code as key. The security code and response are sent to SE, and SE stores the security code.

8. SE verifies that the response and the security code correspond to the challenge, and if so, the user has activated the mobile phone as an OTP receiver for use with SP.

These steps should ensure that the PC and the mobile phone are in the same location, or at least that there exists a communication link between the person using the PC and the holder of the phone. Since the person using the PC is authenticated and has transferred the activation code to the phone, we can assume that this person really wants to activate the mobile phone as an OTP generator.

4.3. Authentication Protocol

The OTP-based authentication protocol takes place between five parties: the user, the user's mobile phone, the user's PC, the SE, and SP. The main steps of the protocol are described below.

1. The user enters the identity he shares with SP on its login page.

2. SP asks the user for an OTP, and sends a request to ES to generate an OTP for the user.

3. SE first sends an SMS to the user's mobile phone to start the client software. It then sends a challenge to the phone together with two encryption keys Ki and Ki+1 (The role of Ki and Ki+1 is explained separately).

4. The user enters his PIN on the phone, and the phone computes the same security code generated at the time of activation. The phone then encrypts the challenge with the security code as key and sends the cipher text as a response to SE.

5. ES verifies that the response from the mobile phone corresponds to the challenge, and sends an OTP to the phone.

6. The user enters the OTP on the SP's login page, and SP contacts ES to verify that the OTP is indeed the correct one for this user.

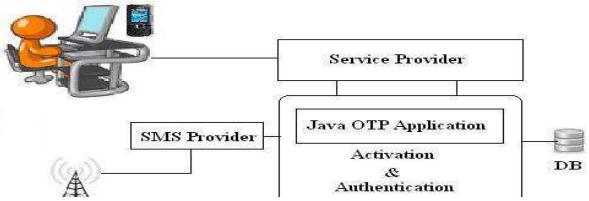

Fig 1. *Existing OTP Generation Mechanism*

Fig1 shows architecture of existing OTP mechanism. The authentication protocol's main goal is to ensure that only the legitimate user can obtain an OTP from SE. The goal is not achieved fully because the phone's response to SE challenge is the encryption of the challenge using the key (security code) made during activation. The answer for this challenge may be known to non-legitimate users also. The correct generation of this key requires the correct PIN and other unique information, which possibly other person who are not legitimate also is supposed to have. This person was in turn authenticated at the time of activation; hence we cannot be confident that he is the legitimate user.

5. Proposal

Here we designed a PassText based authentication scheme in order to produce a security code instead of using a challenge. Our proposed idea explores the usage of PassText [14] which is impossible to break with brute force attack and stay remains as unpredictable. Proposed works are listed below.

5.1. Proposed Activation Protocol

1. The user authenticates himself to SP using credentials already known to SP.
2. When the user asks to activate his mobile phone as an OTP receiver, SP redirects the user's browser to SE with a URL that contains an activation request and a Secure Object.
3. SE verifies that the Secure Object comes from SP, and gets the user's phone number and other unique identification number like IMEI. Here users have to specify PIN.
4. SE sends an activation code to the user's PC and an SMS message to the user's phone asking it to start the client software.
5. The mobile phone asks the user to enter the activation code, available on his/her PC, and transmits the code to SE.
6. SE verifies that the activation code is the same as the one sent to the PC, and sends a Passphrase to the mobile phone together with an encryption key K0.
7. The user chooses a personal identification number (PIN) and enters it on the mobile phone, which generates a security code and a PassText response. The response is the encryption of the PassText using the security code as key. PassText is known only to the legitimate user. The security code and response are sent to SE, and SE stores the security code.
8. SE verifies that the response and the security code correspond to the PassPhrase send, and if so, the user has activated the mobile phone as an OTP receiver for use with SP.

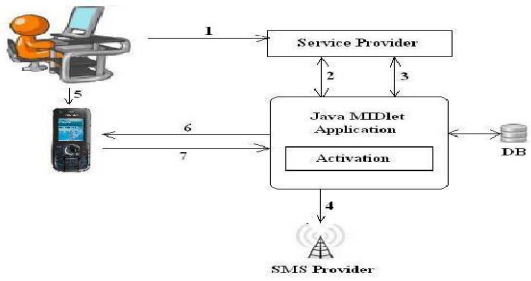

Fig 2. *Proposed Layout of Activation Protocol*

In the above proposed protocol, Passphrase is a simple passage given by either a user or SP and it sent from SP to user while activation process. User converts this Passphrase into PassText by some remember able changes. These changes are known to and done by only legitimate users. So this leads to maximum security and so security level for this scheme becomes unpredictable and proposed security level of this measure described later.

5.2. Proposed Authentication Protocol

The proposed protocol steps are as follows. This ensures the necessary authentication steps for recognizing legitimate user.

1. User requesting SP for an OTP at the first step, rather than user enters the secure login page.
2. SP verifies the mobile request with existing database and if it's approved request, SP request SE to generate an OTP for the user.
3. SE sends a PassPhrase to the phone together with two encryption keys Ki and Ki+1.
4. The user enters his PIN and PassText from Passphrase on the phone, and so the mobile computer security code. The mobile then encrypts the PassText with the security code as key and sends the cipher text as a response to SE.
5. ES verifies that the response from the mobile phone corresponds to the passphrase, and sends a sms to the users mobile to login using the identity shares with SP.
6. Now session cookie is activated by SE and it will be send to user's PC after completing login session by the user.
7. If session cookie has arrived to user's PC, secure login page will be redirected automatically by session cookie after a successful logon process finished by a user. If user logins already without mobile authentication, secure page will not be redirected as there is no session cookie. This method completely avoids malicious user to login with the secure system.
8. Redirected URL will request OTP for a secure authentication. Now SE will send an OTP to user mobile through SP.
9. User can authenticate them by the OTP received through mobile. Successful users only can receive OTP

from SP after completing a strong mobile authentication.

This proposed protocol ensures that only legitimate users are accessing the secure system and it also avoids malware based attacks. It's proven that PassText is a string which is known neither only to the legitimate users nor to unauthorized. So brute force attack turns to be zero and its measures are discussed later.

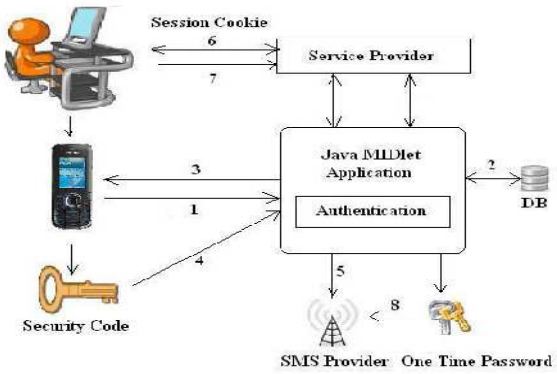

***Fig 3**. Proposed Layout of Authentication Protocol*

5.3. Generation of Security code and Responses

The hash function SHA-1 and the encryption algorithm AES with a 16-byte key are used to generate the security code and the responses. Hashing is denoted by H () and encryption with key K is denoted by EK (). The security code, SC, is computed by the following hash, truncated to 16 bytes,

$$SC=H(PIN||IMEI||CR||SPID)16 \quad (1)$$

where || denotes concatenation. PIN is a secret number with at least four digits entered by the user; IMEI is a 14 digit code uniquely identifying the mobile phone where the Java MIDlet client is installed; CR or client reference is a 40-byte random string generated on the mobile phone during the activation protocol; and SPID is a public value identifying the SP to whom the user wishes to authenticate. The client reference needs to be stored on the mobile phone for later use. It is only stored in encrypted form. During the activation protocol it is encrypted using AES with the key K0 which is sent by ES together with the PassPhrase. When the client reference is needed in the authentication protocol it is first decrypted using Ki, and when it goes back into storage it is encrypted using Ki+1. The keys Ki and Ki+1 are sent from SE together with the challenge in the authentication protocol. The generation of a 16-byte response, R, to a Passphrase PP, is defined by the expression,

$$R = ESC (PP) \quad (2)$$

where SC is the 16-byte security code defined by (1) and PP is a 16-byte PassPhrase received from ES.

6. Simulation Setup

It has been analyzed that in order to thwart the possible attacks in a secure system, three countermeasures have to be considered. To protect against malware-based replay attack, the proposed activation protocol needs to be secured against the replay of old requests. SE must ensure that each secure object is only used once. Also it should not be allowed to activate another mobile phone as OTP generator for an account, if there already exists a mobile phone activated for the specific account. In order to protect against malware-based impersonation attack and phishing attack there is a need for a tighter control over the transition from old OTP generator to a new mobile phone. Simulation was done between a system and a mobile phone which is having the capacity of receiving and storing the secure object (Fig 4). Mobile phone can be replaced with any device which should have an enough capacity to manipulate a secure object. While analyzing the existing one-time password algorithm, obtained results shows that brute force attack is possible in most of the secure networks. Even though an OTP giving more reliability, according to the cryptographic policy, brute force attack will try all possible combinations of passwords until it finds the correct one. It is very difficult to defend against brute force. First when a user requests SP to provide an OTP, SP verifies that whether the mobile request has been registered already. If not, the request will be discarded immediately. If the requested is approved, SE sends a Passphrase which can be considered as a challenge to the user along with the keys. By the help of personal identification number and a passtext, mobile phone will generate a response to the challenge. Here we have designed a schema which will ensure that same challenge and response should not be given to different users. In this way a secure system can be protected safely from a number of user's. If the response is approved to ES, a message will be sent to user's mobile.

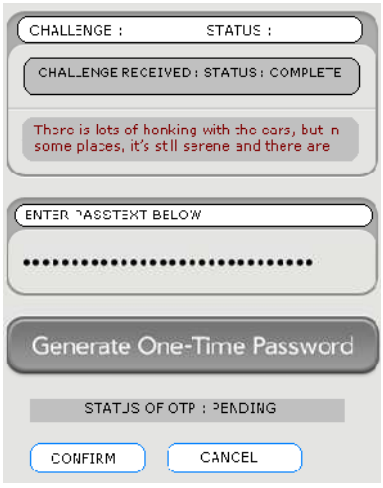

***Fig 4.** Simulation setup (Making of OTP)*

After getting the confirmation message from ES, user will logon by having his/her own credentials. This mechanism completely avoids malware based impersonation attack since user will not advised to use

URL at the first step itself. Fig6. User logon process when the authenticated user logon into the system, SE will send a session cookie to the user's PC and this will redirect to the secure system. If the user have an OTP token, SE will never send session cookie and this will be stop the malicious users to access the secure system.

7. Simulation Results

It seems unfair to say that any set of alphanumeric characters are equally easy to commit to memory. For example "A7Jo0" and word "commit" are not both equal to six units of memory. We have compared password space with different password schemas we can identify the most secure approaches with respect to brute force attack. Table1 demonstrates comparison of password space and password length for popular user authentication schemas. Table1 shows that the approach presented by us is both more secured and the easiest to remember. At the same time, it is relatively fast to produce during an authentication procedure.

Table 1. *Password space comparison.*

Authentication System	Alphabet	Password Length	Password Space Size
Password	64	8 Char	2.8 x 1014
Pin Number	10	4 Numbers	1 x 104
Text with Graphical Assistant	10 spaces	8 Char	2 x 106

7.1. System Testing

The server was implemented using Java. A Siemens MC35i GSM modem was used for sending and receiving SMS messages on the server side. The smslib3.2.0 [17] library was used to send the messages and the SHA 4j [16] library was used to hash the password. An Oracle 10g was used as a database. The client was implemented using J2ME and installed on a Nokia E60 and Nokia E61 phone. Both methods, the connection-less and SMS-based, were tested. In the first method, fake user accounts were setup on both the mobile phone and server. The mobile phone was used to generate 5000 random OTPs at various times of the day and all 5000 generated OTPs matched the OTPs generated on the server side. The use of date and time from the telecommunication company helped solve the synchronization problem.

8. Conclusions

Today, single aspect authentication, e.g. passwords, is no longer considered secure in the internet and banking world. Easy-to-guess passwords, such as names and age, are easily discovered by automated password-collecting programs. Two aspect authentications has recently been introduced to meet the demand of organizations for providing stronger authentication options to its users. In most cases, a hardware token is given to each user for each account. The increasing number of carried tokens and the cost the manufacturing and maintaining them is becoming a burden on both the client and organization. Since many clients carry a mobile phone today at all times, an alternative is to install all the software tokens on the mobile phone. This will help reduce the manufacturing costs and the number of devices carried by the client. This paper focuses on the implementation of two-aspect authentication methods using mobile phones. It provides the reader with an overview of the various parts of the system and the capabilities of the system. The proposed system has two option of running, either using a free and fast connection-less method or a slightly more expensive SMSbased method. Both methods have been successfully implemented and tested, and shown to be robust and secure. The system has several aspects that make it difficult to hack. Future developments include a more user friendly GUI and extending the algorithm to work on Blackberry, Palm, and Windows-based mobile phones. In addition to the use of Bluetooth and WLAN features on mobile phones for better security and cheaper OTP generation.

References

[1] A. Jøsang and G. Sanderud, "Security in Mobile Communications: Challenges and Opportunities," in Proc. of the Australasian information security workshop conference on ACSW frontiers, 43-48, 2003.

[2] Aladdin Secure SafeWord 2008. Available at http://www.securecomputing.com/index.cfm?skey=1713

[3] A. Medrano, "Online Banking Security – Layers of Protection," Available at http://ezinearticles.com/?Online-Banking-Security---Layers-of-Protection&id=1353184

[4] B. Schneier, "Two-Aspect Authentication: Too Little, Too Late," in Inside Risks 178, Communications of the ACM, 48(4), April 2005.

[5] D. Ilett, "US Bank Gives Two-Aspect Authentication to Millions of Customers," 2005. Available at http://www.silicon.com/financialservices/0,3800010322,39153981,00.htm

[6] D. de Borde, "Two-Aspect Authentication," Siemens Enterprise Communications UK- Security Solutions, 2008. Available at http://www.insight.co.uk/files/whitepapers/Twoaspect%20authenticatio n%20(White%20paper).pdf

[7] A. Herzberg, "Payments and Banking with Mobile Personal Devices,"Communications of the ACM, 46(5), 53-58, May 2003.

[8] J. Brainard, A. Juels, R. L. Rivest, M. Szydlo and M. Yung, "Fourth-Aspect Authentication: Somebody You Know," ACM CCS, 168-78.2006.

[9] NBD Online Token. Available at http://www.nbd.com/NBD/NBD_CDA/CDA_Web_pages/Internet_Banking /nbdonline_topbanner

[10] N. Mallat, M. Rossi, and V. Tuunainen, "Mobile Banking

Services,"Communications of the ACM, 47(8), 42-46, May 2004.

[11] "RSA Security Selected by National Bank of Abu Dhabi to Protect Online Banking Customers," 2005. Available at http://www.rsa.com/press_release.aspx?id =6092

[12] R. Groom, "Two Aspect Authentication Using BESTOKEN Pro USBToken." Available at http://bizsecurity.about.com/od/mobilesecurity/a/twoaspect.htm

[13] Sha4J. Available at http://www.softabar.com/home/content/view/46/68/

[14] SMSLib. Available at http://smslib.org