A two-Factor Authentication System using Mobile Devices to Protect against Untrusted Public Computers

Abdullah Alqattan, Nima Kaviani, Patrick Lewis, Nicholas Pearson
Department of Electrical and Computer Engineering
The University of British Columbia, Canada

Abstract— **This paper discusses a two-Factor authentication system that allows both the user and the server to assure that the other is the correct entity. This system also allows a user to use unprotected computers with no concern of revealing information to attacks such as phishing, key logging and wire listening. This is accomplished by placing the entering and encrypting of personal information on a trusted handheld device and by encrypting the data with an asymmetric key that only the correct server would be able to decrypt.**

Index Terms—**two-Factor Authentication, Assymetric Encryption, Privacy, Confidentiality, Password Repository.**

I. Introduction

The current situation for users of the web is a constant barrage of attacks, trying to steal users' confidential information. These include, but are not limited to, phishing scams, key loggers, DNS hijacking, transmission interception and spyware. Unfortunately, typical users are most often neither aware of these dangers nor able to defend against them correctly.

The current protection systems are mostly reactionary, only protecting from previously known attacks. The protection only arrives after a new threat has been detected and analyzed. Anti-malware, anti-virus, anti-spam, anti-phishing, and intrusion detection systems all suffer from this aspect. This time difference leads to gaps in defenses as the protections catch up to new attacks. This period between when a security vulnerability is first found and when users are protected against it can be long enough for massive damage to be done, and for nefarious entities to have stolen personal user information from thousands of individuals.

Quite often the critical personal information includes user credentials used for authentication to websites and other services. Passwords and user names are the usual desirable targets since they lead to the greatest monetary proceeds for the smallest effort. With an ever increasing amount of daily life moving towards or onto the internet, public confidence in the safety of information must also increase. The current reactionary protection only defends against attacks that have happened, but does little to protect against the thousands of new attacks that are written each year. All of this overhead of protection takes a toll on overall speed of a system. A user's system with all of these protections will become inalienable slower.

On a system where a user is in complete control of what is installed or running, they can assure to themselves that their system is safe, but what about when that is out of their hands? Many people use public computers, e.g., when traveling, at a library, at the airport, or at an internet café. A user is left to trust that the owner of the public system has been diligent in protecting the system from outside attacks, and also from fellow users who may have installed something malicious on the public system. Most of the existing protection tools can be easily circumvented when the attacker has physical access to a system.

In this paper we propose the use of handheld mobile devices combined with one-time passwords to guarantee the integrity and confidentiality of user information, including usernames and passwords, while dealing with public computers. In our approach, we treat the handheld device as a digital wallet to securely store the confidential information of the user. Having access to the password protected cell phone is considered one of the two factors during the process of authentication to a Web server. This digital wallet will be further protected with a one-time password which acts as the second factor for the purpose of authentication. This provides users with a more secure way of storing personal information, such that a user can be assured that only he or she would be able to access it. Because two factor authentication is used, even if the cell phone is lost there would be of no effect to the confidentiality, integrity, or availability of user's information. This system is also designed so that users are not trapped to only use their personal computers. This approach provides security against both the unauthorized viewing of information and allows users to confidently use public systems without fear of security being compromise.

The paper is organized as follows: in the next section we will discuss some of the related works in this area, Section III explains our solution for two-Factor authentication, Section IV analyzes our approach against some existing attacks, followed by a conclusion in Section V.

II. RELATED WORK

Security assurance through using handheld devices is attracting more and more attentions due to increasing number of users adopting mobile technologies. Security related researchers have started thinking about employing handheld and mobile devices to devise approaches that may increase the level of assurance in accessing critical information by the end users. The Guardian [4] framework has been designed with an elaborate threat model in mind. Its focus is to protect the privacy of a mobile user, including securing long-term user passwords and protecting sensitive information, e.g., personal data from being recorded (to prevent identity profiling). Guardian works as a personal firewall but placed on a trusted PDA. In effect, the PDA acts as a *portable privacy proxy*. Guardian keeps passwords and other privacy sensitive information out of the reach of key loggers and other malware installed on an untrusted PC.

In their work, Wu et al. [3] use a mobile phone as a hand held authentication token and a security proxy which allows the system to be used with unmodified third party web services. In this method, a user who wishes to use an internet kiosk to access a remote service requiring authentication would instead connect to a trusted security proxy. The proxy store's the user's passwords and can use them to login to the remote service. It also stores a mobile phone number for each user, to which a short text message (SMS) is sent to complete the authentication. Once the user responds to this message, the user's connection from the kiosk is authenticated. The proxy then operates as a traditional web proxy and mediates all aspects of the user's communication with the remote service, but preventing long-term authenticators (e.g., cookies) from touching the kiosk.

Oprea et al. [1] proposed a three-party protocol to provide secure access to a home computer from an untrusted public terminal. A trusted device (PDA) is used to delegate temporary credentials of a user to an untrusted public computer, without revealing any long-term secrets to the untrusted terminal. Two SSL connections are established in the protocol: one from the trusted PDA and another from the untrusted terminal to the home PC using a modified Virtual Network Computing (VNC) system. The PDA authenticates normally (using a password) to the home PC, and forwards temporary secret keys to the untrusted terminal. A user can control how much information from the home PC is displayed to the untrusted PC. Control messages to the home VNC, e.g., mouse and keyboard events, are only sent from the PDA.

Finally in the closes method to that of ours, Mannan and van Oorschot [1] propose a simple approach to counter the attacks we mentioned in Section I. Their method cryptographically separates a user's long-term secret input from (typically untrusted) client PCs; a client PC performs most computations but has access only to temporary secrets. The user's long-term secret (typically short and low-entropy) is input through an independent personal trusted device such as a cell phone. The personal device provides a user's long-term secrets to a client PC only after encrypting the secrets using a pre-installed, "correct" public key of a remote service (the intended recipient of the secrets). The proposed protocol (MP-Auth) realizes such an approach, and is intended to safeguard passwords from key loggers, other malware (including rootkits), phishing attacks and pharming, as well as to provide transaction security to foil session hijacking.

III. OUR SOLUTION

Our approach proposes a two factor authentication system that employs a handheld device, particularly a Nokia N80 smart phone, to authenticate a user to a Web server (possibly a bank server) through an untrusted computer (i.e., a client). The first factor is the combination of username and password that are usually required by Web servers. The second factor is brought to the system as a one-time pass that would be typed into the browser in the untrusted computer and sent from the user side to the Web server (See Fig. 1. for the system architecture).

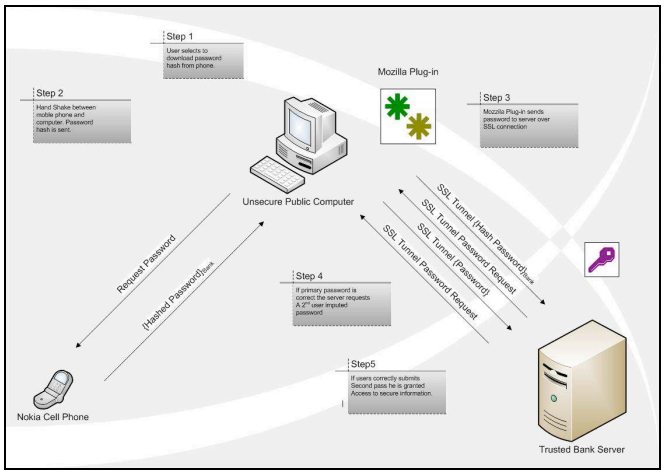


Fig. 1. Architecture of the Designed System

First, the system is initialized by storing the public key of the Web server on the handheld device using an out-of-band channel. During the process of authentication, the user enters the username and the password to the handheld device as it is a less hostile environment than a computer in terms of existing malware and attacks. The password is hashed and stored on the handheld device. Hashing the password provides another layer of security so that even if someone obtains the hash of the password from the handheld device, he or she would not be able to obtain the original form of the password. Furthermore, storing the password is more comfortable than having to enter the username and the password into the handheld device every single time. Also, as we mentioned earlier, the client computer, used to log-in to the Web server, is assumed to be untrusted, and thus it is more prone to attacks. Therefore, it is safer to keep the username and the password away from the untrusted client.

The second factor is a one-time password provided to the user from a set of stored one-time passwords in a notebook.

The notebook is basically a table of passwords and indices with each password corresponding to an index value (i.e., an integer). After the username and the password are passed to the Web server and are correctly authenticated by the Web server, the Web server will pop up a window asking for one of the passwords by displaying its index on the screen. Once the user enters the password into this window, it gets evaluated by the server and, in case of a successful authentication, the user is authenticated to access the information that he or she had asked for. We allow the user to directly enter the one-time password to the untrusted client as we believe even in case it is obtained by the attacker, no confidential information will be revealed and user's information won't be compromised. The reason behind adding the second factor authentication to the design is to add a second layer of security that will protect the user's confidential information in case the smart phone is lost or stolen.

Stepwise, we can define our system through the points below:

1. The public key of the Web server (possibly a bank server) is stored on the user's handheld device.
2. During a login, the user enters her username and password into the handheld device, which is then hashed and stored into the device.
3. The handheld device adds a time-stamped token as well as a random nonce to the username and hashed password, encrypts it using the public key of the bank, and sends it to a bridging client on the untrusted computer. Meanwhile it shows the random nonce on its screen.
4. The untrusted computer receives the message through a handshake on a TCP port and immediately forwards it to a Firefox extension that is also open on a TCP port.
5. The Firefox extension receives the cipher text, opens an SSL connection with the server, and sends the cipher to the server
6. The server decrypts the cipher using its private key and extracts the username, the password, the timestamp, and the nonce. The timestamp is checked to protect against replay attacks, the hashed password for the user is checked against the existing usernames and passwords, and the random nonce is shown to the user on the browser.
7. The user checks the nonce shown on the browser page with the nonce shown on the screen and in case of a match, enters the one-time password to the server.
8. In case of a successful identification of the long term username and password, the validity of time stamp, and the correctness of the one-time password, the server authenticates the user to the system.
9. The user is also assured of a correct connection with the server, first off, because the cipher text cannot be accessed without the correct private key which is kept on the server for sure. Also the equality of the nonce number shown on the browser page with the one shown on the screen of the handheld guarantees against session hijacking. Fig. 2 shows the step by step authentication method in our system.

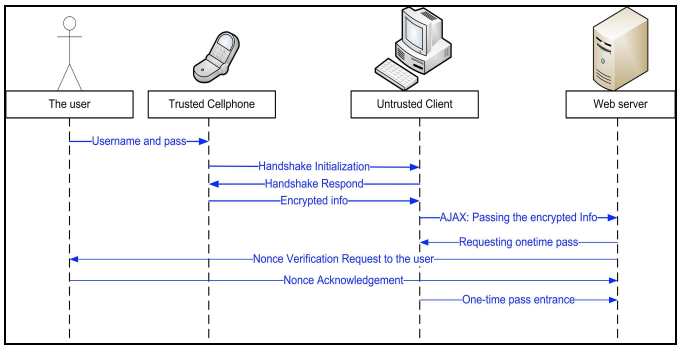


Fig. 2 Step by Step authentication in our two-Factor authentication method

Our prototype system consists of four main pieces of application; a MIDLET application installed on the handheld device, a Java application on the untrusted client, a Mozilla Firefox plug-in as a web browser tool installed on the untrusted client, and a Web server with a PHP script application to behave as the Web server for our prototype.

As for the first part on the handheld device, we have programmed a MIDLET application using Java 2 Mobile Edition (J2ME) [5] to obtain the username and the password from the user, store it on the handheld device, and later send the combination to the untrusted client and from there to the Web server. We have used a simple handshake protocol to establish the connection between the smart phone and the untrusted computer. The connection uses the TCP/IP protocol on port 1234 and WiFi technology. Fig. 3 shows a screen shot of the handheld device communicating with the client application on the untrusted computer. The format of the handshake can be found in Appendix I. Also Appendix II shows the internal structure of the application developed on the cell phone.

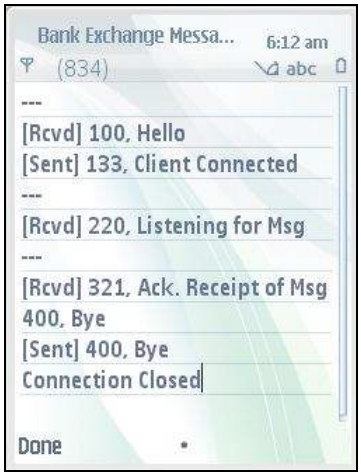


Fig. 3: The screenshot for handshake executed on the cell phone

The second part in the system is a Java application developed to be placed on the untrusted client shown in Fig. 4. It would be used to receive the encrypted username and password. The encryption assures the confidentiality of the

message while it is being sent from the smart phone to the Web server. After receiving the message, the application on the untrusted client will pass it to the web browser used to access the Web server. In our approach we have developed a plug-in for Firefox to achieve this goal. The connection between the application and the web browser is a TCP/IP connection on the local host using port 7055 with a handshake protocol similar to the one used between the handheld device and the Java application on the untrusted computer.

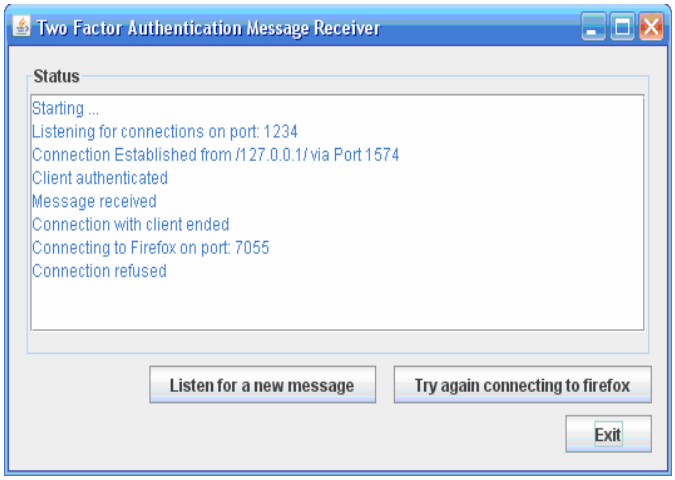


Fig. 4. Screenshot of the Java application developed for the Untrusted Computer

The third part of our system is a Firefox plug-in that we have developed using JavaScript and AJAX technology to receive the encrypted message from the application and then pass it along to the Web server. The Web browser plug-in establishes a connection with the server over the Internet using the HTTPS protocol on port 443.

The fourth part of our design is the receiving end which is a Web server with a PHP script application to ensure security. The application is developed using the OpenSSL library for security protocols. The server will receive the message and decrypt it using the previously stored private key.

The second stage of our approach is to send the one-time password to the server. Once the data is decrypted on the serve side, the server will show the nonce and asks the user to verify its correctness. Having the nonce verified, the server will ask for a one-time password by displaying the corresponding index number for the password. The user will find the password that has the same index number from the provided notebook and enter the one-time password directly to the web browser which will, in turn, send it to the web server through the same SSL channel.

IV. Discussion

Our approach solves multiple problems in one proactive solution. Instead of waiting for new attacks to be created, we are going to sidestep a whole assortment of them. The main advantages are that we do our encryption on a trusted device and that we authenticate the server. It should be noted that many of the choices of how this was implemented was to protect the user, even from the most dangerous individual out there; themselves. By assuming that the worst will happen, (users will give information to attackers freely, the system they are using is compromised, their internet connection is being tracked, etc) we can better protect from these possibilities. We are also assuming our approach has been accepted, and necessary software is installed and available to use.

Our approach defends the user against many forms of attack. We take how well the computer is protected out of the situation by turning that system into a messenger, unable to view the encrypted data. If our communications were replayed at a later date, the first authentication has a time stamp embedded into the encrypted part, allowing the server to determine how long of a time window is acceptable. Spoofing and man in the middle attacks are also negated since the data is public key encrypted. We have designed this approach with the worst possible circumstances in mind so that all of them may be avoided if any attacks were tried against a user. It is also safe against shoulder surfing as the index number for our one-time password method is chosen randomly, thus even having a look at the notebook will most likely give no clue to the attacker about what the next password is going to be as there is very little chance that the next password to be questioned by the server is going to be the one seen by the attacker. The method is also safe against session hijacking as the user has to approve the validity of the nonce shown by the server and the one shown on the screen of the cell phone, thus assuring the validity of the delivered packet and the communicating server. Furthermore, the system is safe against key logging attacks as no long time credential is typed into the untrusted public terminal. The only authenticating interaction with the untrusted terminal is entering the one-time password. However, it will be of no harm to the user even in case it gets stolen, as it is not going to be valid any longer after the first time it is used.

Personal computers have steadily become more vulnerable to and come under more frequent attacks. We decided to assume that many have fallen into the realm of untrustworthy. To help the user securely pass credentials to the web service, encryption must be employed. Since we are assuming the system cannot be trusted, another trusted element must be employed. With the prevalence and power of modern cell phones, they seem to be a likely candidate to assist us. If we properly use the phone as an encryption device, we have taken away all attacks on the personal computer. The computer, if compromised, will have no way of recovering the encrypted password. This turns the computer into a middleman, just there to pass on the message.

At the other end, the server side, they are used to receiving plain text passwords (ignoring transmission encryption). If we allow the current approach to continue, anybody who asks for and receives a password can make use of it. This is where phishing scams come from. By faking their credentials, they

can trick users into freely giving over personal information. If we were to encrypt the user's information before transmission with a known public key of the actual institute, we can neutralize the possibility of the information falling into the wrong hands. This public key encryption is the encryption done on the cell phone.

Unfortunately, cell phones lack in ease for inputting text. If we allow the cell phone to store the authentications in a secure manner, we can ask the user to only enter them once. The cell phone now becomes a key storage, storing the user information and the server's public key. Entry could be handled in various ways, from manual with the phone's keypad, or from a transmission protocol that establishes a new entry.

The cell phone also allows the user to move from system to system, without having to worry about a newly encountered system's trustworthiness. It may be full of malware, but the malware will get nothing of use. The ability to be mobile leans on our assumption that the new system has the ability to communicate with the cell phone. A future expansion on this would be the ability to transfer seamlessly between computers by taking your cell phone with you, but this is beyond the scope of this report.

With a new mobile trusted device that has our personal information, and the ability to dispense that information so that only the trusted party can understand it, we seem to have solved many of the problems by shifting the weight of the attacks from the personal systems to our cell phones. This will put our cell phones at greater risk. To continue with our approach of assuming the worst has happened, we have also planned if a cell phone is stolen. A second authentication beyond what the cell phone provides has been added to the authentication system. Now if the cell phone is broken or compromised, there is still one more piece that an attacker is missing. This notebook will have a list of passwords for each site indexed for easy reference. After accomplishing the first authentication, the server responds by requesting a random password from its list. This second factor of authentication protects against attacks if the cell phone is stolen.

V. CONCLUSION

The system is designed to protect the user from a wide range of attacks comprising computers in general. The second factor is added to eliminate the risk of the handheld device itself being compromised. Therefore, the user can log-in using public computers without the fear of their passwords being exposed, and be safe even if the handheld is lost or stolen. The system uses layers of security in an effort to eliminate as many threats as possible. However, it is impossible to eliminate risk all together since using the password in itself means that there is a chance of identity theft. The only way to eliminate risk all together would be not to log-in at all, but, obviously, that is unacceptable. The design protects the user by making it harder for the attacker to gain access of the components used to authenticate the user.

REFERENCES

[1] A. Oprea, D. Balfanz, G. Durfee, and D. Smetters. Securing a remote terminal application with a mobile trusted device.In *ACSAC*, 2004.

[2] M. Mannan, P.C. van Oorschot. Using a Personal Device to Strengthen Password Authentication from an Untrusted Computer. Financial Cryptography and Data Security (FC'07), Lowlands, Scarborough, Trinidad and Tobago, Feb.12-15, 2007. *Extended version: Technical Report TR-07-11* (Mar 2007).

[3] M. Wu, S. Garfinkel, and R. Miller. Secure web authentication with mobile phones. In *DIMACS Workshop on Usable Privacy and Security Systems*, July 2004.

[4] N. B. Margolin, M. K. Wright, and B. N. Levine. Guardian: A framework for privacy control in untrusted environments, June 2004. *Technical Report* 04-37 (University of Massachusetts, Amherst).

[5] Sun Microsystems (2007, November). Java platform micro edition [Online]. Available: http://java.sun.com/javame/index.jsp

APPENDIX I

Handshake protocol:

Initialization
100, Hello – sent by the server
133, Client Connected - sent by the client

Request
220, listening for message - sent by the server

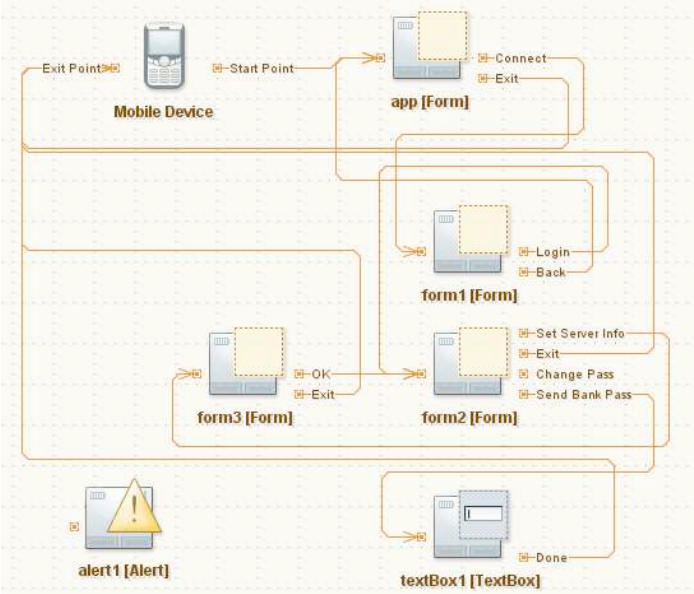
Replies
320, message: "The Cipher" - sent by the client
321, acknowledge receipt of message - sent by the server

400, Bye - sent by the server

Errors
500, Handshake Failed - sent by the server
510, Unknown protocol - sent by the client
520, Unknown protocol - sent by the server
530, Unknown Request - sent by the client
540, Message missing - sent by the server

APPENDIX II

The internal architecture of the application that we developed on the cellphone is shown below. The architecture is generated using NetBeans MIDLET development tool.


Exit Point
Start Point
Mobile Device
app [Form]
Connect
Exit
Login
Back
form1 [Form]
OK
Exit
form3 [Form]
Set Server Info
Exit
Change Pass
Send Bank Pass
form2 [Form]
alert1 [Alert]
Done
textBox1 [TextBox]