IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 5, Issue 2, February 2016

ATM Security Using Virtual Password

Renjith R[1], Arya S[1], Jasmine Yesudasan[1], Keerthy S Kumar[1], Krishnaveni S[1], Ajeesh S[2], Jooby E[3]

UG Scholar, Department of computer science, College of Engineering Perumon, Kollam, India [1]

Assistant Professor, Department of Computer Science, College of Engineering Perumon, Kollam, India[2]

Assistant Professor, Department of Computer Science, College of Engineering Perumon, Kollam, India[3]

Abstract: In this project, we implement a system for protecting user passwords from being stolen by adversaries in Automated Teller Machines (ATM).We use VIRTUAL PASSWORD MECHANISM in which a user has the freedom to choose a virtual password scheme ranging from weak security to strong security and also BIOMETRICS. ATM allows the account holder to have transactions with their own accounts without allowing them to access the entire bank's database. Traditional ATM transaction method is replaced with virtual password generation and fingerprint technology. With the use of these technologies a genuine user can be identified. The user's details such as fingerprint, 4 digit account number, phone number etc are stored in system database during registration. After identifying the user using its ID, the server generates virtual password and sends it to the user's mobile phone. The user then inputs this virtual password. If it is correct his/her fingerprint is verified and allowed to make transactions

Keywords: Phishing, Codebooks, differentiated virtual passwords, secret little functions, shoulder-surfing.

I. INTRODUCTION

Internet today has become an integral part of daily human lives. Online banking, online shopping etc are few examples of the significance internet have today. With increase in the popularity of internet the importance of providing more security and authentication for online systems has also increased. In this paper we present one such security mechanism for ATM machines.

Fingerprint recognition is an active research area nowadays. An important component in fingerprint recognition systems is the fingerprint matching algorithm. According to the problem domain, fingerprint matching algorithms are classified in two categories: fingerprint verification algorithms and fingerprint identification algorithms. The aim of fingerprint verification algorithms is to determine whether two fingerprints come from the same finger or not. On the other hand, the fingerprint identification algorithms search a query fingerprint in a database looking for the fingerprints coming from the same finger.

There are hundreds of papers concerning to fingerprint verification but, as far as we know, there is not any framework for fingerprint verification available on the web. So, you must implement your own tools in order to test the performance of your fingerprint verification algorithms. Moreover, you must spend a lot of time implementing algorithms of other authors to compare with your algorithms. This was our motivation to put our fingerprint verification framework available for everyone.

Present technology used for ATM security is OTP or one time password which is easily crackable, increases delay of time in receiving OTP via SMS and also not reliable. We use virtual password mechanism and an additional finger print verification to ensure tight security and also to overcome the time delay. Virtual password is a password that is valid for only one login session or transaction and after that it becomes obsolete.

II. RELATED WORK

Previous work on defending againstuser password-stealing attacks for the three major categories. Phishing attacks are relatively new but very effective. Thereare two typical types of phishing. First, to prevent phishingemails a statistical machine learning technologyis used to filter the likely phishing emails; however, sucha content filter does not always work correctly. Blacklists of spamming / phishing mail servers are not useful when an attacker hijacks a virus-infected PC.

A path-based verification was introduced. Key distribution architecture and a particular identity-based digital signature scheme were proposed to make email trustworthy. Second, to defend against phishing websites, the authors in and developed some web browser toolbars to inform a user of the reputation and origin of the websites which they are currently visiting. The authors implemented password hashing with a salt as an extension of the web browse, a web proxy, or a stand-alone Java Applet. Regardless of the potential challenges considered in an implementation, such password hashing technology has a roaming problem becausenot every web browser installs such an extension or sets the web proxy.

Another more important challenge is that more web browsers need to be designed in which designers are not reluctant to include specified extensions for each other. Unlike phishing, malicious Trojan horses, such as a key logger, are not attacks, and sophisticated users can avoid them. Such programs are also easy to develop and there is a great deal of freeware that can be downloaded from the Internet to prevent them. Alphanumeric password systems are easily attacked byshoulder-surfing, in which an adversary can record the user motions by a hidden camera when the user types in the password. In [22], the authors adopted a game-like graphical method of authentication to combat shoulder-surfing; it requires the user to pick out the passwords from hundreds of pictures, and then complete rounds of mouse-clicking in the Convex Hull. However, the whole process needs the help of a mouse and it takes a long time. In [23], the authors proposed a scheme to ask a user to answer multiple questions for each digit. In this way, it is only somewhat resistant to shouldersurfingbecause, if an adversary catches all the questions, then they will know what the password is. In [23], gamebasedmethod was designed to use cognitive trapdoor games toachieve a shield for shoulder-surfing. The author in filed a patent to allow a user to make some calculations based on a system generated function and random number for the user to prevent password leaking. However, the scheme in is not anti-phishing and the password can be stolen if an adversary uses a camera to record all the screens of the system and motions of the victim. Note that using SSL/TLS in websites could not prevent keylogger attacks.

A. Bank Module
The first is the login process. Initially the administrator does the account creation for a new user. Once the request is approved the user can open the account and an account number is provided by the bank at the time of registration. The login process of user leads to user page. The website provides the transaction details of the user, their account balance details about the bank and services provided by the bank

B. Virtual Password Sending
In this module the server generates a different virtual password each time by encrypting the user pin and sends it to the users registered mobile number.The user inputs this virtual password which enables him to proceed with the fingerprint authentication.

C. Fingerprint
In this module fingerprint of the registered user is enrolled and verified. Fingerprint enrolment is done by entering the unique account number of the user provided by the bank at the time of account registration. Fingerprint is verified at the transaction time and if the fingerprint verified is authorized, he/she can withdraw the money and can check the account balance.

III. VIRTUAL PASSWORD

To authenticate a user, a system (S) needs to verify a user (U) using the user's password (X) and ID (also denoted as U) which the user provides. In this procedure, S authenticates Uby using U and X, which is denoted as: S →U: X. BothU and X are fixed. It is reasonable that a password should beconstant so that it can be easily remembered. However, theprice of being easily remembered is that the password can bestolen by others and then used to access the victim's account.At the same time, we cannot put X in a randomly variantform because it would be impossible for a user to rememberthe password. To confront such a challenge, we propose ascheme using the new concept of virtual password.

A virtual password is a dynamic password that is generateddifferently each time from a virtual password scheme and thensubmitted to the server for authentication. A virtual passwordscheme P is composed of two parts, a fixed alphanumeric X(i.e., the real password, also called the hidden password) and afunction F from the domain ψ to ψ, where the ψ is the letterspace which can be used for passwords. Since we call P =(X, F) a virtual password scheme, we call F a virtual passwordfunction (VPF). The result (denoted as V) of the VPF is calleda virtual password, and F may have some hidden parameters,H, which are the secrets between the server and the user. Ifthis is the case, we denote F with FH(...). Note that the VPFcan be a secret between the server and the user. Let X =x1x2...xndenote a vector standing for the hidden password,where xi (i = 1...n) is a digit, and n is the length of the vector.Let R = r1r2...rndenote the random number provided by theserver, also called the random salt, and prompted in the loginscreen by the server. V = v1...vnis the virtual password usedfor authentication. The user input includes (U, V), where U isthe user ID. On the server side, the server can also calculate Vin the same way to compare it with the submitted password.We use V = FH(X, R) or FH(xi, ri) = vi interchangeably inthe rest of this paper.

It is easy for the server to verify the user ifF is a bijective function. If F is not a bijective function, it is also possible toallow the server to verify the user as follows. The server firstfinds the user's record from the database based on the user'sID (i.e., U), then it computes V and compares it with the oneprovided by the user. A bijective function makes it easier forthe system to use the reverse function to deduce F's virtualpassword. Therefore, we do not assume that F is a bijective function.

IV. FINGERPRINT TECHNOLOGY

With increasingly urgent need for reliable security, biometrics is being spotlighted as the authentication method for the next generation. Among numerous biometric technologies fingerprint authentication has been in use for the longest time and bears more advantages than other technologies do.

Fingerprint authentication is possibly the most sophisticated method of all biometric technologies and has been thoroughly verified through various applications .fingerprint authentication has proved its high efficiency and further enhanced the technology in criminal investigation in more than a century. Fingerprint authentication is far more accurate and efficient than any other methods of authentication. Biometrics is a technology that helps to make your data tremendously secure, distinguishing all the users by way of their personal physical characteristics.

Fingerprint technology is the most widely accepted and mature biometric method and is the easiest to deploy. It is simple to install and also it takes little time and effort to acquire one's fingerprint with a fingerprint identification device. Thus, fingerprint recognition is considered among the least intrusive of all biometric verification techniques. Fingerprint Images are captured and later encrypted as a result of which the original one cannot be recreated. This prevents hacking or any other misuse of the system.

For fingerprint recognition, a system needs to capture fingerprint and then follow certain algorithm for fingerprint matching. Improved enhancement algorithms of fingerprint image increase the security of bank account and the ATM machine. Fingerprint identification process consists of two essential procedures: enrollment and authentication. Fingerprint identification system compares the input fingerprint image and previously registered data to determine the genuineness of the fingerprint. If images of fingerprint are poor-quality images, they result in missing features, leading to the degrading performance of the fingerprint system. Thus, it is very important for a fingerprint. Recognition system to estimate the quality and validity of the captured fingerprint images.

Our framework allows performing fingerprint verification experiments for databases B of FVC2000, FVC2002 and FVC2004; and databases A of FVC2002 and FVC2004. In this experiments, we use the performance indicators of the Fingerprint Verification Competitions [2] (EER(%), FMR100(%), FMR1000(%), ZeroFMR(%), Time(ms) and ROC curves). Additionally, you can include experiments with a custom evaluation protocol or different databases.SS.
We implemented the fingerprint verification algorithms proposed by Ticoand Kuosmanen Jiang and Yau, Medina-Pérez. It is important to highlight that, despite the algorithm of Qi et al. is a combination of a minutiae matching algorithm with an orientation based algorithm, we implemented only the minutiae matching algorithm. We also implemented the feature extraction algorithms proposed by Ratha et al. and the orientation image extractor proposed by Sherlock et al.This framework allows you to include new fingerprint matching algorithms as well as new feature extraction algorithms with minimum effort and without recompiling the framework.
Extract the file “FingerprintRecognition.zip” and build the solution. Then you can debug the project “FR.FVCExperimenter” or you can execute “FR.FVCExperimenter.exe” in the directory containing the generated assemblies.In the “Resources” text box, specify the path of the database that you are going to use,. Select the proper experiment type in the combo box with label “Experiment”. Use the combo boxes “Minutia Extractor”, “Orientation Image Extractor” and “Skeleton Image Extractor“ to select the algorithms that will be used to compute the basic features (minutiae, orientation image and skeleton image).
Use the combo box with label “Matcher” to select a fingerprint verification algorithm, and the combo box with label “Feature Provider” to select the algorithm that will be used to store and retrieve the features for the selected matcher. Despite that we implemented only one feature provider for each matcher, there are possible scenarios where you may have multiple feature providers for each matcher

If you want to visualize features for certain fingerprint then you can use “FR.FeatureDisplay” project. In “Fingerprint Feature Display” form, you can change the feature extractor and feature display. In the framework, we include classes to visualize minutiae, orientation image and skeleton image.This section presents an example of using the framework to match two fingerprint images in a custom user application. It includes the three steps to compare two fingerprint images: image loading, feature extraction, and feature comparison. In this case, users need to add references from their application to the assemblies FR. Core and FR.Medina2012. Assemblies Shull Delaunay Triangulation and Image Processing Tools must be included in the output folder where the binary files appear.The first thing that you need to know is that you do not need to modify the applications of the framework in order to recognize your algorithms because we use Reflection to load all algorithms dynamically at execution time.You may create as many assemblies as you want in the directory containing the framework. For each new assembly, go to the properties and set the output path with value “..\bin\Release\”.In order to add a new feature extractor, you must inherit from the generic class FeatureExtractor<T> and implement the method ExtractFeatures(Bitmap image).For each feature extractor you must create a resource provider. Resource providers allow saving (retrieving) to (from) file the resources associated to fingerprints.
The framework includes resource providers for extractors of minutiae (MinutiaListProvider), orientation image (OrientationImageProvider), and skeleton image (SkeletonImageProvider). The users do not need to modify the framework in order to integrate custom algorithms because Reflection is used to load all the algorithms dynamically at execution time. This way, users should add new algorithms to their own custom assemblies.In order to use an existing matching algorithm within the framework, the first thing that users need to do is to create a resource provider. Resource providers allow saving (retrieving) to (from) files the resources associated with fingerprints
One of the goals that we kept in mind while developing this framework was to achieve class interfaces as simple as possible. This way, adding new algorithms is pretty straightforward.Also, fingerprint may be taken and digitized by relatively compact and cheap devices and takes only a small capacity to store a large database of information. With these strengths, fingerprint authentication has long been a major part of the security market and continues to be more competitive than others into days world.
Fingerprint recognition is an active research area nowadays. An important component in fingerprint recognition systems is the fingerprint matching algorithm. According to the problem domain, fingerprint matching algorithms are classified in two categories: fingerprint verification algorithms and fingerprint identification algorithms. The aim of fingerprint verification algorithms is to determine whether two fingerprints come from the same finger or not. On the other hand, the fingerprint identification algorithms search a query fingerprint in a database looking for the fingerprints coming from the same finger. The most closely related work to our framework is the FVC-on going web system. This system has the following limitation

V. CONCLUSION

In this project, we introduce a security system for preventing unauthorized access of a person’s bank account by an attacker when the bank card is lost or when the password is stolen. The newly introduced authentication levels such as a virtual password and finger print verification ensures tight security. This enables the authorized user to access his account securely and provides enhanced security to the ATM system.

REFERENCES

[1] T. Dierksn and C. Allen, The TLS Protocol—Version 1.0, IETF RFC 2246, Jan. 1999.
[2] [Online]. Available: http://en.wikipedia.org /wiki/Phishing
[3] Anti-Phishing Working Group. [Online]. Available: http://www. antiphishing.org
[4] [Online]. Available: http://en.wikipedia.org/wiki/Key−logger
[5] [Online]. Available: http://www.eweek.com/article2/0, 1895, 1940623, 00.asp
[6] B. Ross, C. Jackson, N. Miyake, D. Boneh, and J. Mitchell, “Stronger password authentication using browser extensions,” in Proc. 14th USENIX SecuritySymp.
[7] E. Gaber, P. Gobbons, Y. Mattias, and A. Mayer, “How to make personalized web browsing simple, secure, and anonymous,” in Proc. Financial Crypto, LNCS 1318. 1997.
[8] E. Gabber, P. Gibbons, D. Kristol, Y. Matias, and A. Mayer, “On secure and pseudonymous user-relationships with multiple servers,” ACM Trans. Inform. Syst. Security, vol. 2, no. 4, pp. 390–415, 1999.
[9] E. Jung. Password maker [Online]. Available: http://passwordmaker. mozdev.org
[10] J. la Poutr’e. Password; Composer [Online]. Available http://www.xs4all.nl/?jlpoutre/BoT/Javascript/PasswordComposer 1987[Digests9thAnnualConf.MagneticsJapan, p.301, 1982].