PassPattern System (PPS): A Pattern-Based User Authentication Scheme

T. Rakesh Kumar and S. V. Raghavan

Network Systems Laboratory, Department of Computer Science and Engineering,
Indian Institute of Technology Madras
Chennai-600036, India.
{rakeshk, svr}@cs.iitm.ernet.in

Abstract. Authenticating a user online, without compromising the user comfort is an important issue. The most popular approach to authenticate a user online is Password-based authentication. Studies show that, users (always) choose very simple passwords which are often easy to guess. On the contrary, randomly generated strings are difficult to remember, especially if the user is having many passwords. In this paper we present a dynamic password scheme based on patterns, called PassPattern System (PPS), which works using the existing infrastructure. PPS is an Adaptive Authentication System, where the strength of the system can be changed depending on the need of the application without compromising the user comfort.

Keywords: User Authentication, Security.

1 Introduction

"Open, Sesame!" is a classical example of using a SecretWord/Password in the famous tale *Ali Baba and the Forty Thieves* [1] to open the door of the magical cave. Since then SecretWord/Password has been used to authenticate the user before accessing any resource. With the advancement in computers, Internet and electronic commerce, authentication has become increasingly important. One of the main problems with the username-password scheme is 'selection of the password' itself. Studies show that users will always pick passwords which are short and easy to remember [2]. Often it is very easy to break the password of the user, if the personal information about him/her is known and more often than not, it is widely known. With technological advancement, alternative authentication schemes are reported in literature, which are more secure than the conventional Password-based scheme using Biometrics, RSA SecurID® [3], and Graphical passwords [4]. Each mechanism has its own advantages and disadvantages. One of the main reasons why password is still widely used is that newer technologies entail significant changes in the infrastructure of the system. Besides, the vulnerabilities of Graphical Passwords are still not fully understood and is an active area of research [4]. It appears that the need of the hour is to design an authentication scheme that is easy and intuitive to use, strong and robust against all possible attacks, and works using the existing hardware/software infrastructure.

In this paper we present the PPS, which is a dynamic password authentication system, where the user will *prove* to the system that he/she knows the secret instead of communicating it directly. PPS can be visualized as a challenge-response system, where in the response to a challenge is hardly reusable, there by rendering the replay attack or its variant ineffective.

1.1 Related Work

There are several attempts reported in literature about authentication schemes in lieu of the traditional Password-based system. While each attempt is successful in increasing the strength of the system against some of the known attacks, they are either computationally intensive or they require additional hardware/software in the infrastructure. In this section we review the contemporary attempts, identify the gaps, and highlight the motivation for developing PPS.

Password-based authentication. Password system is the oldest and the most popular authentication scheme used in the modern world. But users tend to choose simple passwords, which are easy to remember but they are very easy to break [2,5]. One way of preventing this is to provide users with a random password. While it is difficult for the user to remember, with contemporary computing power even a computer generated random password of length 8 characters is not strong enough for bruteforce attack; With Blue Gene/L[1], it will take 1 hour to break password of length 10 characters. According to Microsoft "*A strong password should appear to be a random string of characters to an attacker. It should be 14 characters or longer, (eight characters or longer at a minimum)*" [6]. While it is recommended that user should not use same password for more than one account, it is quite common that the users have more than one online profile. Besides, users have to maintain a separate account for e-mail, social networking, e-banking, e-commerce, blogging, multimedia sharing, etc. Remembering a 14 character password for each account is difficult. The other way of increasing the strength of the password system is to make the password dynamic, i.e. the user has to enter a different password every time he/she does a login. But existing dynamic password or One-Time Password schemes either require extra hardware or complex mathematical operations.

Biometrics. Biometrics comes under the category of authentication systems which use '*What the user is'*. The main disadvantage with any biometric system is the need for a change in the infrastructure of the entire system. For example if Yahoo wants to replace its authentication system from password to fingerprint based authentication, all the users have to install fingerprint readers on their computers and Yahoo should maintain a registry. This kind of change in infrastructure may not be feasible always. Moreover, biometrics are not often secret [7], as people publicly expose their voice and fingers in various ways on a regular basis, creating the possibility of biometric spoofing. For biometrics like Retina-based authentication and Vein-based authentication, spoofing is not possible but the installation cost is very high.

[1] IBM Blue Gene is the current fastest super computer. [www.research.ibm.com/bluegene/]

Graphical passwords. The main idea of Graphical passwords [4] is that users find it easy to remember Graphical passwords than text-based passwords. Graphical passwords are more secure than text-based passwords from various attacks like dictionary attack, bruteforce attack and spyware. Some of the problems with the graphical passwords are that the users tend to choose the same sequence of images as their password making it easy to guess; Graphical passwords require much more storage than conventional text-based passwords.

RSA SecurID®. RSA SecurID® is a two-factor authentication. It is based on something you know (a password or PIN) and something you have (an authenticator). RSA SecurID® is one of the dynamic password systems, where the user password will change for every login. Issues that cause concern are the cost and the possible of loss of physical devices.

Virtual keyboard. Virtual keyboard is yet another user authentication system, where the user has to enter his/her password by clicking the characters on a virtual keyboard instead of typing the password on the conventional keyboard. While such a system is good against keyloggers and spywares, it is susceptible to other attacks like Shoulder surfing, guessing, and Social Engineering.

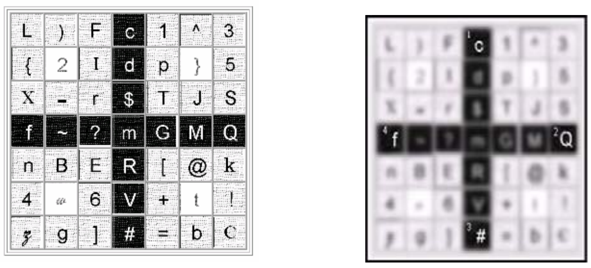
Motivation for PPS. *The discussions hitherto underline the need for an authentication system, that is simple, easy-to-use, inexpensive, incorporates the essence of challenge-response system, dynamically changing, robust against attacks such as bruteforce, dictionary, shoulder surfing, spyware, social engineering, Man-In-The-Middle attack and all above doesn't demand any special hardware/software.*

2 PassPattern System

PassPattern System is a challenge-response system and is based on the premise that 'humans are good at identifying, remembering and recollecting graphical patterns than text patterns' [8]. The core idea of PassPattern system is that, '*Instead of remembering a sequence of characters as the secret, users have to remember a shape (which is stored internally as a sequence of positions in hash form) as the secret*'.

Whenever the user wants to get authenticated, the PPS displays an NxN matrix of cells, which is known as *PatternSquare*. Each cell in the PatternSquare is an image, which represents a character as shown in Fig. 1(a). The character can be an alphabet, number or even special characters. The PatternSquare is the *challenge* that is sent by the server to the user. The PatternSquare will be generated dynamically upon each user request. Hence the characters in each cell of the PatternSquare may change or their *position* may change or both may change, for every authentication request.

At the time of registration the user is asked to choose a sequence of positions as shown in Fig. 1(b), by typing the characters in those positions. The sequence of positions *then* becomes the user's *PassPattern.* The cell sequence of the PassPattern so chosen will remain the *secret* between the user and the system.



(a) (b)

Fig. 1. (a) A sample PatternSquare. This is the sample of the image that the user will see, whenever he/she wants to login to the system. (b) A sample user choice of a PassPattern. A user can choose any sequence of positions as the PassPattern.

The user has to remember only the shape (cell sequence in the PatternSquare) as that is the only *secret*. For the user to get authenticated, he/she has to type the characters that are present in the chosen shape (PassPattern) in the PatternSquare. The cells in the PatternSquare are colored[2] in such a way that, the user can see smaller subsections of the PatternSquare (3 x 3 in our example), which makes it easy for him to remember his PassPattern. Whenever the user wants to login to the system, the user has to type the sequence of characters which appear in the user's PassPattern presented by the system as a challenge (This sequence of characters is referred to as *SecretWord*). During every presentation of the PatternSquare, the contents change and hence the user has to type a different SecretWord as shown in Fig. 2, during every authentication process such as login.

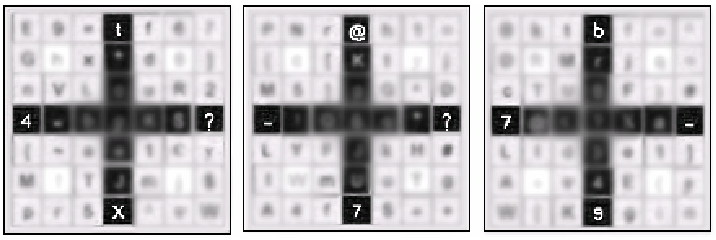


Fig. 2. The above figure shows PatternSquares that are displayed to the users at different login attempts. SecretWords corresponding to the PassPattern in Fig. 1(b) will be "*t?X4*", "*@?7-*" and "*b-97*".

2.1 Design Issues

PPS allows users to choose any set of characters as their PassPattern. The user can select the PassPattern based on some sequences familiar to him/her, for example knight's moves on a chess board or some symmetric positions. Because of the coloring[2] of the cells, remembering the PassPattern will be easy.

[2] To enhance user convenience; no relationship to security of the PPS.

The PPS comprises of 4 parts - PPS software, Pseudo-Random number generator, Image generator, and PassPattern database. PPS software controls all activities in the system, Pseudo-Random number generator and Image generator are used to create dynamic challenges and PassPattern database stores all the PassPatterns of the users. Internally the cells of the PatternSquare are indexed in row-major order (and left to right), starting from 0 to N^2-1 (where N is the size of the matrix). Hence the PassPattern of a user will be stored as a string of numbers. Both PassPattern and usernames will be stored in the database as Hash, using MD5 hash function.

Registration phase. Whenever a user sends a request for new registration, the server will create two S-seeds based on the "current clock time" (This seeds are valid only for that session at the server). Using S-seeds as the seed values, the Pseudo-Random generator will (randomly) pick two sets of 49 characters (for a 7x7 matrix) out of 94 printable characters in ASCII. The Image generator takes these characters and creates two sets of 49 images and sends the images to PPS Software. PPS Software creates two PatternSquares using the images and sends them to user as challenges. These two challenges are similar to the process of retyping the password in traditional password systems at the time of registration. The S-seed values are preserved as the session variable (in physical memory) for the verification phase**.**

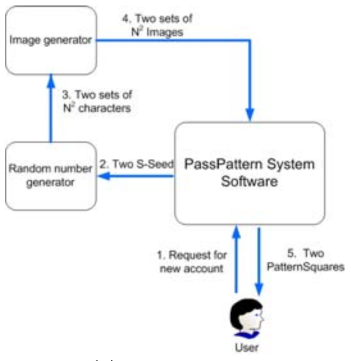
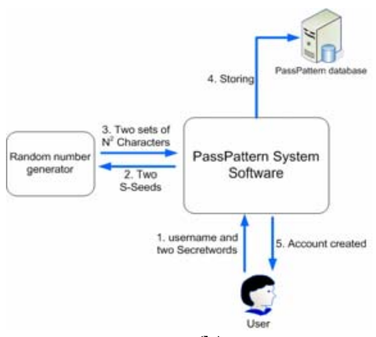


Image generator
4. Two sets of N^2 Images
3. Two sets of N^2 characters
Random number generator
2. Two S-Seed
PassPattern System Software
1. Request for new account
5. Two PatternSquares
User

(a)

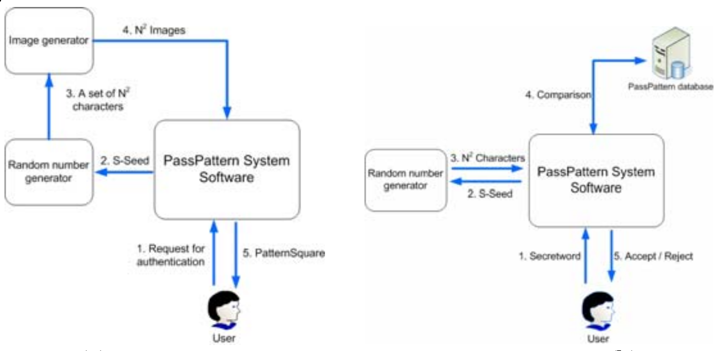

PassPattern database
4. Storing
3. Two sets of N^2 Characters
Random number generator
2. Two S-Seeds
PassPattern System Software
1. username and two Secretwords
5. Account created
User

(b)

Fig. 3. (a) Registration phase in the PPS. **(b)** Validation and Creation of the user account.

Once the user gets these two challenges, the user has to type the username and SecretWords. The SecretWords will be based on the user selection of the PassPattern. The PPS checks both the PassPatterns and if they are same, then the PPS calculates the MD5 Hash of the username and PassPattern and stores them in the PassPattern database and acknowledges the successful registration to the user.

The Authentication phase. Whenever a user makes a login or authentication request, the PPS Software creates the PatternSquare (similar to the registration phase) and sends it to the user as a challenge. Once the user gets the challenge, the user has to type the username and the SecretWord which is based on his/her PassPattern and the PatternSquare (Secret and Challenge). The PPS software calculates the MD5 Hash of the username and PassPattern and compares the result with the value stored in the PassPattern database. Based on the comparison results, the user will be either allowed or rejected.



(a) (b)

Fig. 4. (a) Challenge creation phase in the PPS. **(b)** Response validation phase in the PPS.

Other Design Issues. PPS is an adaptive user-authentication system. The strength of PPS can be adapted for various situations. The strength of the PPS can be varied by changing two parameters – the size of the PatternSquare and the type of image. The first method is to change the size of the PatternSquare, i.e. the more the size, more is the security. Number of possible patterns of length n in an NxN matrix will be $(N^2)^n$. Thus, as the size of PatternSquare (N) increases, the total possible number of PassPatterns will increase, which in turn increases the strength of the system. The second method of increasing the security is by changing the type of the image in the PatternSquare. The strength of the system can be drastically improved by changing the type of the image. Each character in the PatternSquare can be represented with different coding, compression, noise, distortion, and even as a *CAPTCHA*[3] image.

3 Security Strength of PPS

In general, several attacks are possible on an authentication system. For any authentication system, the hacker can attack at least at three places; they are *server*, *client*, and *the communication link*. The different attacks *on sever* includes Bruteforce attack, Dictionary attack, attack on PassPattern database, and compromising the server as a whole. At the *client*, the possible attacks are keylogging and shoulder surfing. Finally on the *communication link*, the possible attack is Man-In-The-Middle attack, which can be done using packet sniffers.

In terms of the data being passed from the user to the server and the data which is being stored in the server PPS is comparable with the classical Password-based authentication system. In both cases, user sends the username and a SecretWord and

[3] CAPTCHA (Completely Automated Public Turing test to tell Computers and Humans Apart) is a test used in computing to determine whether the user is human. For more information on the CAPTCHA refer [http://www.captcha.net].

this will be compared with the registry in the database. But because of the dynamic nature of the challenge-response system, PPS is more secure than Password-based scheme to attacks such as Bruteforce, Dictionary attack, Keylogger, and Shoulder surfing. For the rest of the attacks, PPS is as vulnerable as Password-based schemes (e.g. Server database compromise attack). The best known solution for such attacks is to use cryptography protocols at the server or on the communication link. In this section we analyze the impact of the four attacks mentioned here on PPS.

Bruteforce attack. The hacker can try two kinds of Bruteforce attacks on this system. The first way of attacking the system is to ignore the PatternSquare and try with some random string. For a user PassPattern of length 4, there will be a unique SecretWord for the given session. If the hacker wants to guess that SecretWord, the probability of success will be $1/(94^4) = 1.28\text{x}10^{-8}$ (Since there are 94 printable characters). If the guess is wrong the probability of success will remain same for the next guess, it is because the SecretWord will change with every attempt. Hence,

$$\text{The probability of success for every attempt} = \frac{1}{94^n} \qquad \mathbf{(1)}$$

The other way of doing Bruteforce search is to try all combinations of positions. For example, if we consider a 7x7 PatternSquare there will be 49^n (if selection of PassPattern includes reuse of positions) or ${}^{49}P_n$ (without reuse of positions) different patterns of length n.

$$\text{Number of possible PassPatterns} = \begin{cases} (N^2)^n \\ \dfrac{N^2!}{(N^2 - n)!} \end{cases} \qquad \mathbf{(2)}$$

To break the system, the hacker on an average has to break $(n\text{x}49^n)/2$ images (with reuse) or $(n\text{x}{}^{49}P_n)/2$ (without reuse).

$$\text{Number of images that are to be broken} = \begin{cases} \dfrac{n \times (N^2)^n}{2} \\ \dfrac{n \times N^2!}{2 \times (N - n)!} \end{cases} \qquad \mathbf{(3)}$$

N represents the size of the PatternSquare and n represents the length of the PassPattern.

Dictionary attack. Dictionary attack is one of the most commonly used techniques to break a Password-based system. In the case of PPS commonly used shapes, sequences can be possible candidates in a dictionary. However, as the PatternSquare changes randomly on every presentation, it approaches the behavior of one-time pad.

Keyloggers. Keylogger is a program, which captures the user’s keystrokes and sends this information to the hacker. The natural protection for an authentication system from the keylogger is to have a one-time password (or Dynamic password). PPS being a dynamic password system, is not vulnerable to keyloggers. Even if the hacker gets the SecretWord of the user of a PPS system, this SecretWord cannot be reused by the hacker to login to the system, because of the dynamic nature of the PatternSquare.

Shoulder surfing. Shoulder surfing is looking over someone's shoulder when they enter a password or a PIN code. It is an effective way to get information in crowded places because it is relatively easy to stand next to someone and watch as they fill out a form, enter a PIN number at an ATM machine, or use a calling card at a public pay phone. Shoulder surfing can also be done at a distance with the aid of binoculars or other vision-enhancing devices to know the password. Shoulder surfing can be done easily on the password system, just by seeing the keys that the user is typing. But to decode the PassPattern in the PPS, the hacker has to see both the key sequence and the PatternSquare and do a mapping before the user submits the page. So shoulder surfing is of little or no use in PPS as compared to a Password-based system.

4 Conclusion

In this paper we presented PPS, a user authentication system which can be a potential replacement to classical Password-based authentication system. The strength of the system can be changed by varying the size of the PatternSquare and the type of image associated witch each cell of the PatternSquare without increasing the length of the PassPattern. PPS is very easy for the user to use and at the same time it is more secure than conventional Password-based system. An existing Password-based system can be migrated to PPS, without any change in the infrastructure. Some of the open problems in this area are: What is the ideal length of the PassPattern? What is the ideal size of the PatternSquare? What is the general user behavior while choosing a PassPattern? *Prototype* of the PPS is available at http://netlab.cs.iitm.ernet.in/pps

References

1. Geraldine McCaughrean C.: One Thousand and One Arabian Nights, Oxford University Press, USA (1999)
2. Adams and M. A. Sasse, M.S.: Users are not the enemy: why users compromise computing security mechanisms and how to take remedial measure. In: Communications of the ACM, vol. 42, issue 12, pp. 40--46 (1999)
3. RSA Security Inc. RSA SecurID® authenticators, http://www.rsasecurity.com
4. Xiaoyuan Suo, Ying Zhuand Owen, G.S, C.: Graphical Passwords: A Survey. In: *acsac*, 21st Annual Computer Security Applications Conference (ACSAC'05), pp. 463--472 (2005)
5. K. Gilhooly, Biometrics: Getting Back to Business, in computerworld, May 09 (2005)
6. Strong passwords: How to create and use them http://www.microsoft.com/protect/yourself/password/create.mspx
7. John Brainard, Ari Juels, Ronald L. Rivest, Michael Szydlo and Moti Yung.: Fourth-Factor Authentication: Somebody You Know. In: Proceedings of the 13th ACM conference on Computer and communications security, pp. 168--178 (2006)
8. R. N. Shepard, C.:Recognition memory for words, sentences and pictures, Journal of verbal Learning and verbal Behavior, vol. 6, pp. 153--163 (1967)