Graphical Passwords as Browser Extension: Implementation and Usability Study[1]

Kemal Bicakci[1], Mustafa Yuceel[1], Burak Erdeniz[2], Hakan Gurbaslar[2], Nart Bedin Atalay[3]

[1] TOBB University of Economics and Technology, Ankara, TURKEY
{bicakci, myuceel}@etu.edu.tr
[2] Middle East Technical University, Ankara, TURKEY
{burakerdeniz, gurbaslar}@gmail.com
[3] Selcuk University, Konya, TURKEY
nartbedin@gmail.com

Abstract: Today, most Internet applications still establish user authentication with traditional text based passwords. Designing a secure as well as a user-friendly password-based method has been on the agenda of security researchers for a long time. On one hand, there are password manager programs which facilitate generating site-specific strong passwords from a single user password to eliminate the memory burden due to multiple passwords. On the other hand, there are studies exploring the viability of graphical passwords as a more secure and user-friendly alternative. In this paper, we present GPEX, a password manager program implemented as a web browser plug-in to enable using graphical passwords to secure Internet applications without any need to change their authentication interface. Experimental results show that GPEX has security and usability advantages over other password manager plug-ins. specifically; we find that with the visual interface of GPEX, users have a more complete and accurate mental model of the system and incorrect login attempts causing security exposures can easily be avoided.

Keywords: graphical password, password manager, usable security, authentication

1 Introduction

User authentication is a central component of almost all security applications. The weaknesses of using text based passwords for authentication are well known

[1] This research is supported by TUBİTAK (The Scientific and Technological Research Council of Turkey) under project number 107E227.

and there is a significant body of recent research exploring the feasibility of graphical approaches to provide a more secure and usable alternative. Based on the studies showing that human brain is better at recalling images than text [1], graphical passwords are intended to solve memory burden and small password space problem of classical passwords [2].

Another solution to generate strong passwords is password managers. These manager programs can be implemented as plug-ins to web browsers and they translate easy to remember and low-entropy passwords into stronger passwords which are more immune to dictionary attacks [3-5]. Password managers also provide a solution to "password reuse" problem and are capable of generating site-specific passwords from the same password entry to protect victims against phishing attacks [3-5].

While password managers have attractive security properties, independent usability studies found that they suffer from major usability problems [5]. In particular, one critical finding is that users have inaccurate or incomplete mental models of the password manager software which potentially causes security exposures [5].

In this paper, we join the forces of graphical passwords and password manager programs in order to obtain a more usable and secure authentication system. For this purpose, we use click-based graphical passwords as the new interface of the password manager browser plug-in. More precisely, in the proposed system user clicks several times on an image as a password and the browser extension converts the graphical password into a site-specific text-based password. We think that this system provides a clearer mental model since the translation of graphical password to a text-based one and inserting it to the password field is an easier process to understand. The extension implemented is user-friendly and provides a more secure user experience. For instance, in our system it is obvious when the plug-in has been activated and is awaiting input, thus the solution alleviates the problems associated with incorrectly assumed state of the system. We also argue that our work will ease quick adoption of graphical passwords since it does not require any change on the server side.

Implementing a browser extension for click-based graphical passwords was not as straightforward as we initially thought. We discovered that a design allowing arbitrarily chosen sequence of clicks on an image without predefined click regions could not be a portable solution. Offset values are required to be initially generated and stored in the system in order to convert chosen click locations in acceptable regions to the same text password in each trial [9][10]. However, carrying these offset values to a different machine is apparently not practical. Thus, we first conduct an experiment to compare security and usability of click-based graphical passwords with and without visible grids. An earlier work [11] claimed that visible grids limit user's freedom of choice, without conducting an experiment to justify this argument. We found that using images with visible predefined click regions improve the security and do not degrade the usability of graphical passwords. This is an important finding in its own right. In our implementation, we used images with visible grids due to the portability advantages.

The rest of our paper is organized as follows: The procedure and the design of the first experiment are detailed in Section 2. Section 3 provides the analysis of the data collected in this first experiment. Section 4 discusses other specifics in our implementation of GPEX (Graphical Password as Browser EXtension) software. Section 5 provides brief information on PwdHash, an earlier browser extension to strengthen text based passwords. Section 6 explains the methodology for our second experiment, which compares usability and security of GPEX and PwdHash. Section 7 provides the analysis of the data collected in the second experiment and Section 8 gives concluding remarks.

2 Experiment I: Click-based Graphical Passwords with and without Visible Grids

There are numerous graphical password schemes proposed in the literature (e.g., Passpoints [11], Cued Click Points [12], and Persuasive Cued Click Points [13]) which share the same basic principle: image(s) to be clicked are displayed without any visible grids. User is free to choose his password by clicking to any point(s) on the image(s) and for a successful authentication he is expected to re-enter his password within a tolerable region centered on the original click location.

The method we have described above would have portability problems when implemented as a browser plug-in instead of the authentication method for a specific Internet application. In order to facilitate using the browser extension from any computer the user wishes to use, clicked points in a tolerable region must always be converted to the same text password since obviously the Internet application expects the same text password to be entered in the password field. Previous work [9] [10] proposed solutions to this problem for a different reason and in a different context (i.e. to be able to store the hash of the password not the password itself). However their solutions do not satisfy the portability requirement in our application scenario because of the need to pre-calculate and store offset values to translate clicks in tolerable regions to a single text based password. On the other hand, converting click locations on a pre-partitioned image to a text password is straightforward. Drawing visible (and maybe ugly) grid lines over an image is adequate to obtain a one-to-one mapping between click locations and a text based password (more detailed information on generating site-specific text-based passwords from graphical passwords will be provided in section 4).

We conducted a laboratory experiment to evaluate whether drawing grid lines over an image changes the effectiveness, efficiency, user satisfaction and security of click-based graphical passwords. This experiment as well as our second experiment was approved by the ethics committee of Middle East Technical University. Performance memory and data entry speed were compared for grid and non-grid conditions. Each participant either saw the stimuli in Figure 1a, which was first used in [11], or Figure 1b. In both cases, users click on five points as their click-based passwords.

If Figure 1b is used, participants have to choose exactly the same grids to confirm their password. Each grid is a square with 20x20 pixels in size. As a convention, clicks on the grid line are accepted inside the below and right of that line. For Figure 1a, a tolerance square of 19 pixels centered on the original click-point is allowed to confirm the password. In both cases, participants are required to click in the correct order. The image is 451x331 pixels in size.

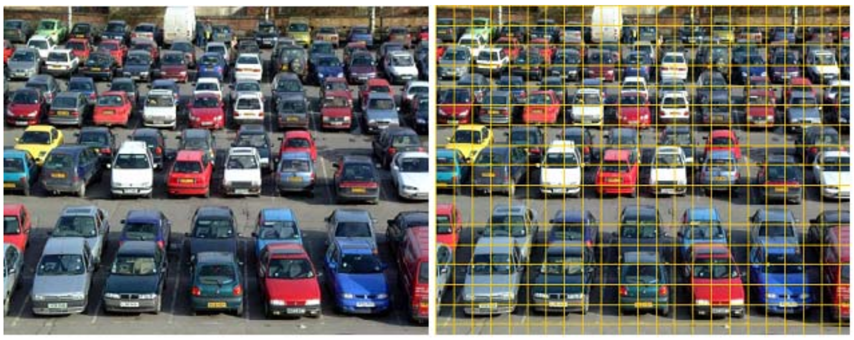


Fig.1. The stimuli used in Experiment I: An image with (a) grids (right) and (b) without grids (left).

Effectiveness of click-based graphical passwords was measured with the number of participants who forgot their passwords, number of attempts to remember the password, and the amount of time to enter the correct password. Efficiency was measured with the amount of time to generate and confirm passwords. User satisfaction was evaluated with a questionnaire. Security was evaluated with a hot spot analysis.

Forty six subjects participated in the experiment. All of the participants were students or employees of TOBB University of Economics and Technology (average age: 22.5 years). There were twenty four males and twenty two females. Participation was voluntary. Participants were randomly allocated to one of the two experimental conditions: password image with grid (n=23, 11 males, 12 females) and without grid (n=23, 13 males, 10 females).

The data was collected with a desktop computer. Each participant sat facing a computer screen. Participants gave responses using mouse. They were informed about click-based graphical passwords, and they were introduced to the stand alone version of the graphical password manager system. The experiment began with a practice session in which participants practiced generating and confirming click-based graphical passwords. Passwords were generated by clicking five points on the picture and confirmed by re-clicking these points on the same order. The number of clicks to go was presented at the bottom of the image. Participants who failed to confirm their passwords repeated the practice session. The images of the practice session were different from the experiment session. Participants in the grid group practiced with images with grid lines, and participants in the non-grid group practiced images without grids.

Following the practice session, participants generated and confirmed a click-based graphical password which they had to remember later. Then participants left the laboratory. They were re-invited to the laboratory after at least four days. In the test session participants were asked to remember their click-based graphical password. Participants continued their attempts until they remembered the correct password or they decided that they could not remember the password. Participants decided themselves that they forgot the password. Finally, a questionnaire (a 5-point Likert scale) on usability of click-based graphical passwords was completed.

3 Results and Discussion of Experiment I

Effectiveness, efficiency, user satisfaction and security were investigated. Efficiency was measured with the amount of time to generate (Figure 2) and confirm password (Figure 3). There was no significant difference between grid and no-grid group for either of these measures (t(44) = -0.41, p>0.05 for generation; t(44) = 0.83, p>0.05 for confirmation).

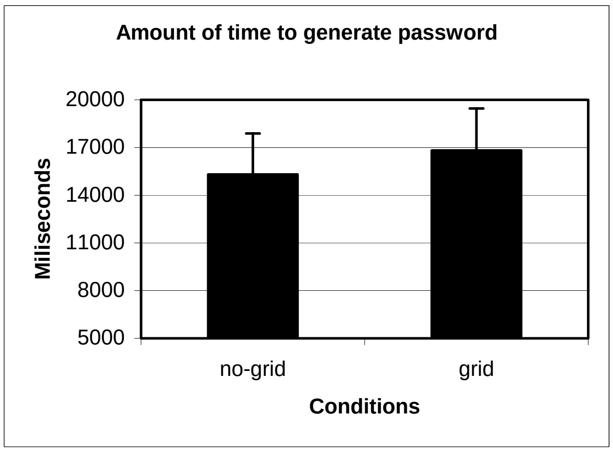


Fig. 2. Amount of time to generate a click-based graphical password for grid and no-grid conditions.

Effectiveness of click-based graphical password was measured with the number of participants who forgot their passwords, number of attempts to remember the password, and the amount of time to enter the correctly remembered password. Four participants in the grid group (%17) and five participants in the no-grid group (%22) forgot their passwords. There was no significant difference between groups ($X^2(1) = 0.138$, $p>0.05$). Number of attempts to remember the correct password is presented in Figure 4, which was not significantly different between groups ($t(35) = -0.52$, $p>0.05$). Amount of time to enter the correctly remembered password is presented in Figure 5. There was no significant difference between groups either ($t(35) = -0.19$, $p>0.05$).

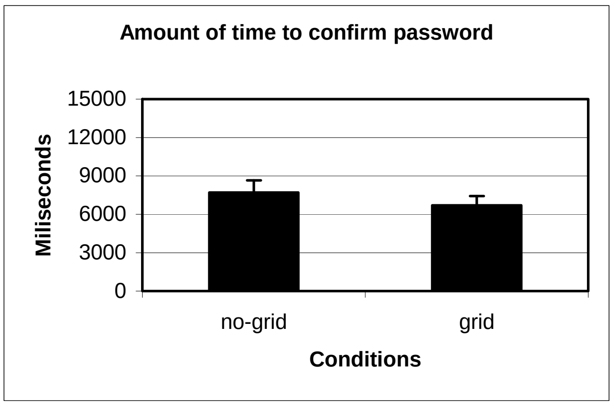


Fig. 3. Amount of time to confirm a click-based graphical password for grid and no-grid conditions.

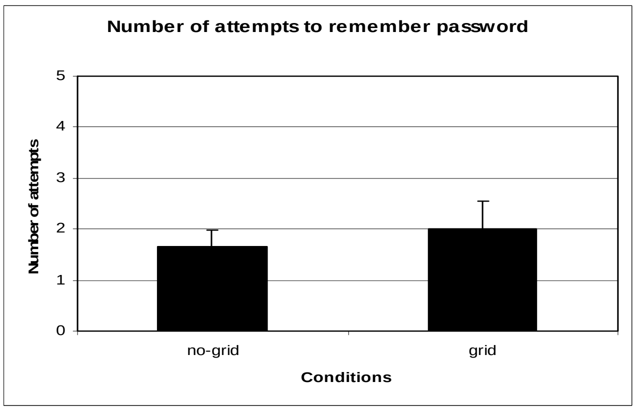


Fig. 4. Number of attempts to remember a click-based graphical password for grid and no-grid conditions.

User satisfaction was evaluated with a questionnaire (5-point Likert). Answers were compared with non-parametric Mann-Whitney U test. None of the comparisons revealed a significant difference between grid and no-grid groups with respect to user satisfaction. As a result, we can conclude that drawing grid-lines on an image did not change usability and user satisfaction of click-based passwords.

Hot-spots are regions on images that are clicked with higher probabilities compared to other regions. Hot-spots create security vulnerabilities for click-based graphical password systems since they reduce the effective size of password space. In other words, graphical passwords can be broken by less effort by conducting a hot spot attack similar to dictionary attacks possible in text-based passwords [2].

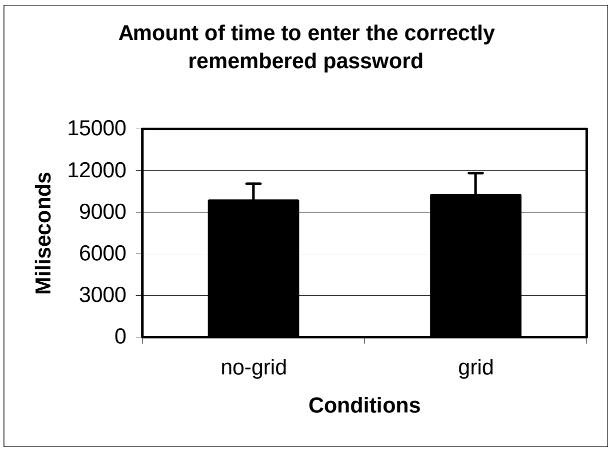


Fig. 5. Amount of time to enter the correctly remembered password.

To compare the security of graphical passwords for grid and non-grid conditions, we conduct a hot-spot analysis of passwords chosen by subjects in the experiment. For each condition of Experiment I, 23 participants clicked 115 points in total.

Figure 6 presents histograms showing number of regions with respect to number of times clicked by subjects in grid and non-grid conditions. For instance, in grid condition there are 45 regions clicked only one time, 16 regions clicked two times and so forth. In non-grid case, when a click is in the tolerance region of another click, we enroll them in the same region.

The method we compare these two histograms is described as follows: First of all, we devise the following formula to calculate the expected number of clicks per region.

$$E_r = \binom{115}{r} \times \left(\frac{390}{391}\right)^{115-r} \times \left(\frac{1}{391}\right)^{r} \times 391$$

E_r denotes the number of regions chosen “r” times. Since we have a total of 115 points, the probability of choosing the given region for “r” times is $\left(\frac{1}{391}\right)^{r} \times \left(\frac{390}{391}\right)^{115-r}$. We insert $\binom{115}{r}$ in order to consider the order of the choice. Finally, we multiply it with total number of regions (391) in order to find the expected value.

Secondly, using this formula, we determine that if each user click is an independent random event we expect 86 regions clicked for one-time, 13 regions clicked two-times and 1 region clicked three times. There are 100 different regions in total expected to be clicked in the idealized case.

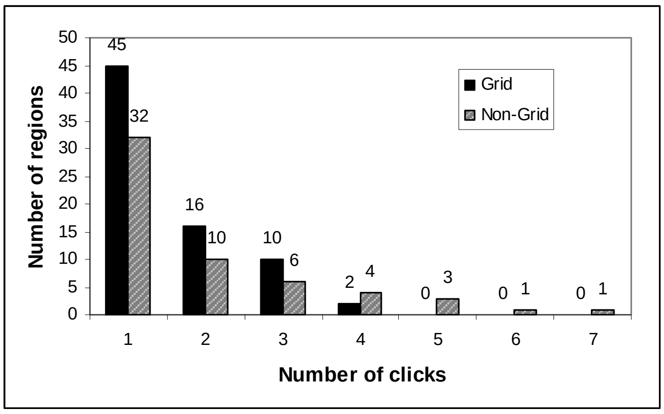


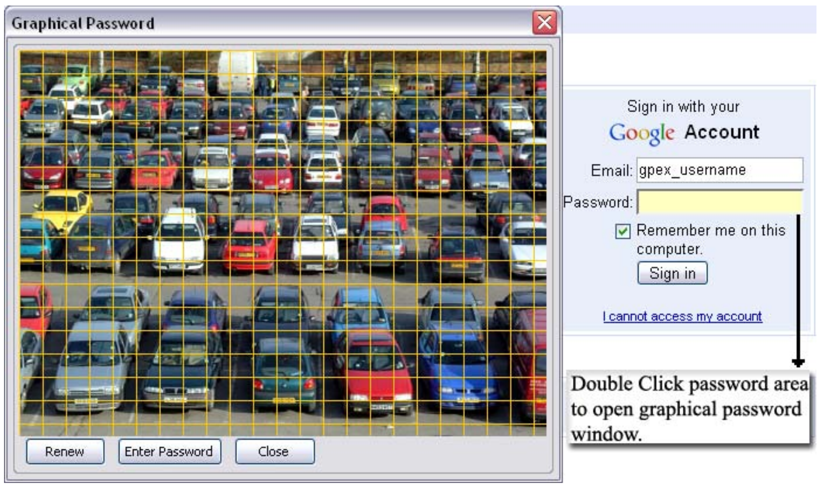
Fig. 6. Number of regions versus number of clicks for grid and no-grid images.

And finally, we decide whether grid or non-grid image is more vulnerable to a hot-spot analysis by comparing the deviations of the corresponding histogram from the expected values calculated. Using figure 6, we calculate that 73 and 57 different regions were clicked with grid and non-grid images, respectively. Since with non-grid image, clicks were concentrated more on specific regions, we conclude that visible grids help to alleviate the hot-spot problem[2]. One explanation for this difference is that with visible grids in addition to other image features (cars, plates, colors, etc.) users can also benefit from grid lines to choose and memorize their passwords. In other words, visible grid lines help to enrich the image to select among more choices.

As a result of Experiment I, we conclude that drawing visible grid lines over an image does not affect usability and user-satisfaction of click-based graphical passwords. More than that, it helps to reduce the number of hotspots.

4 GPEX: Graphical Passwords as Browser Extension

In this section we introduce GPEX as a novel method to experience more secure and user-friendly web interactions. GPEX is an authentication system implemented as a plug-in to the Firefox web browser. The implementation is publicly available at http://myuceel.etu.edu.tr/gpex. It uses click-based passwords based on selection of at least five points on a picture which has horizontal and perpendicular grid lines with 20 pixel gaps between each adjacent parallel line. User interface of GPEX is very simple and includes a picture (451 x 331) and three buttons namely "Renew", "Enter Password" and "Close" (Figure 7).



Email: gpex_username
Password:

Double Click password area to open graphical password window.

Fig. 7. The screenshot of the GPEX browser extension activated while *gmail.com* is visited (the box and the arrow showing it is not part of the GPEX interface)

[2] This hot-spot analysis is our preliminary work. As a future work, we plan to collect more data and make a more elaborate investigation on this topic.

In order to activate graphical password extension, a user should double-click the password field on a web page. If the clicked field is actually a password field, extension window pops up and the color of the password field turns to yellow indicating that a password is expected (Figure 7). Grid lines on the image constitute 391 squares and all clicks in the same square are regarded as equivalent entries. To create a password, user should select at least 5 points by clicking inside squares. Clicking on the grid line is accepted as clicking right and/or bottom square of the line. For security reasons, less that five clicks is not accepted as a legitimate password entry.

There are three buttons on the pop-up window: renew, enter password and close (see Figure 7). The functions of the buttons are as follows:

Renew: If a user selects a point by mistake or decides to change the selected points he can click this button to initialize the process. After clicking this button, the system resets selected points and user must click all five points again.

Enter Password: After five points are selected and user is sure that correct points are clicked, user should click on this button. This button activates the hashing process, which will be explained shortly, and generates a unique password for the visited web site. The selected (double-clicked) password field is filled with the generated password, color of the field turns to its original color and pop-up window disappears.

Close: This button is used to close the window when the user decides to give up entering his password. Password field turns to its original color.

Implementation details of GPEX are summarized as follows. GPEX first detects all password fields in the visited web page and listens for double click events. When a password field is double clicked, extension window appears. If any location other than password fields is clicked, extension window does not appear. This is an effective countermeasure against so called “mock password field attack” by which an attacker obtains each pressed key in a hidden field and writes an asterisk in the mock password field.

In the extension window, the parallel lines on the 451 x 331 resolution picture forms 391 squares in total (Figure 7). When a user clicks inside a square, the extension internally records the chosen square with respect to its row and column numbers. This recording repeats for all clicks and row and column numbers are concatenated until the “Enter Password” button is clicked. The domain name is extracted from the full web site address and when “Enter Password” button is clicked, a secure hash function takes the concatenated coordinate values and the domain name of the web site as input parameters and generates a strong site-specific password like “xC4rEnjW7v”[3].

There are several types of JavaScript attacks such as keyboard and mouse monitoring, domain rewriting, etc. [3] aiming to steal user’s password while it is typed or during its submission. Graphical user interface of GPEX allows a user to enter his password by mouse clicks instead of pressing keys. This means that malicious codes that listen keyboard events can not succeed in stealing the password if GPEX is preferred.

It is true that malicious JavaScript codes can also listen to mouse events. However using GPEX when a user double-clicks a password field, a pop-up window opens and the user enters his password from this new window. Up to our best knowledge, a mouse listener event implemented as JavaScript code and embedded in the HTML of the visited web page can not listen to mouse events in another window. Thus, we can say that GPEX is also superior to other password manager systems with respect to its immunity against JavaScript attacks. Note that any mouse event can be observed by all the extensions installed on the browser hence all extensions installed on the browser should be trustworthy to protect the password.

5 PwdHash

In this section, we provide brief information on PwdHash [3,4], a browser plug-in that automatically generates strong site-specific passwords from a user’s text based password and the domain name of the visited web site. After installation, in order to use PwdHash, a user should either type @@ or press F2 before entering his password. When F2 is pressed if the active field is not a password field, PwdHash warns user not to enter his password. PwdHash uses the domain name of the website as a parameter of a hash function which automatically generates a site-specific password from the typed password. Then, the system updates the target password field accordingly. By this procedure, the system provides a solution to password reuse problem and prevents phishing attacks [3, 5].

PwdHash has no visual interface and users can be confused since it is difficult to see the difference between entering password with or without PwdHash. We will discuss usability problems of PwdHash in section 7. To login using a browser without the extension, users can go to www.PwdHash.com and generate their site-specific passwords there.

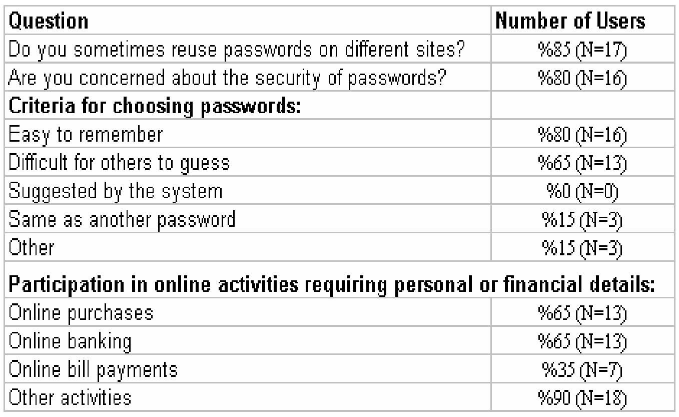
[3] We have not considered tricky implementation issues such as multiple domain names and different rules for password syntax yet. We refer interested readers to [3] for the discussion of these issues.

6 Experiment II: Usability Comparison of GPEX and PwdHash

Before used by end users, performing a usability analysis of software products is a best practice [8]. There are numerous usability inspection methods [7] that can be conducted by experts in the field (e.g., user interface designers). Another method to validate the usability of end products is to make a usability study in a laboratory environment with participants who are potential users of the product. On this line of argument, we designed an experiment to compare the usability of graphical (GPEX) and text based (PwdHash) password manager systems. We choose PwdHash [3] instead of Password Multiplier [4] as the text-based password manager to compare with our system because an earlier work [5] has found that PwdHash is more user-friendly and perceived to be more secure. Our experimental design and methodology is similar with the authors of [5] and we compare our findings with their results in section 7.

Twenty participants (10 male and 10 female) who did not participate in Experiment I participated in the Experiment II. Participants were undergraduate and graduate students of Middle East Technical University (average age: 24.8 years). Participation was voluntary. The design of the experiment was within-subject; participants both tested with GPEX and PwdHash password manager systems. The order of tests was balanced between subjects.

Table 1. Results represent the number of participants (out of 20) responding yes to each question.



Question	Number of Users
Do you sometimes reuse passwords on different sites?	%85 (N=17)
Are you concerned about the security of passwords?	%80 (N=16)
Criteria for choosing passwords:	
Easy to remember	%80 (N=16)
Difficult for others to guess	%65 (N=13)
Suggested by the system	%0 (N=0)
Same as another password	%15 (N=3)
Other	%15 (N=3)
Participation in online activities requiring personal or financial details:	
Online purchases	%65 (N=13)
Online banking	%65 (N=13)
Online bill payments	%35 (N=7)
Other activities	%90 (N=18)

At the beginning of the experiment, participants filled a questionnaire about their Internet usage and perception of web security. In this questionnaire all participants reported visiting the web daily and their initial attitude toward web security are shown in Table 1. For us, the most striking result is that 80 percent of participants are concerned about the security of their passwords. This result is an indication for the need to develop new password security systems.

In the experiment, the participants completed a set of four tasks (login, migrate, update and second login) for both GPEX and PwdHash. These were same as in [5] except [5] also included the Remote Login task (login from a computer that does not have the necessary plug-in installed). We excluded the Remote Login task because this task would be very similar for GPEX and PwdHash.

Participants completed the tasks for a specific online email system (www.gmail.com). The order of the tasks was also balanced in the experiment. In the Log in task, participants have to login to the email account by using one of the plug-ins. In the Migrate Password task, participants have to login to the email account by using the given unprotected password and then they have to change the password to a protected one by using one of the plug-ins. In the Update Password task, participants have to change the password which was previously created by the plug-in. Finally, in the Second Login task, participants have to login to the email account for a second time with their changed protected password. After finishing a particular task, participants continue with other tasks without any break. After all tasks were completed, participants answered eight questions about perceived security, comfort level, ease of the task, and perceived necessity of password manager systems.

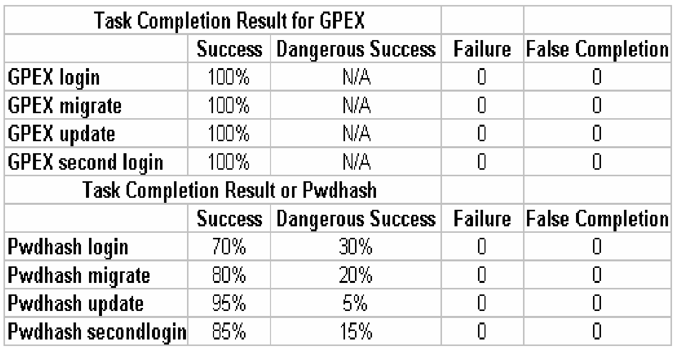
GPEX and PwdHash software were pre-installed in different notebooks and subjects performed the four tasks on these computers. Before starting the experiment, participants were given instruction sheets providing very brief information on GPEX and PwdHash. Similar to [5], users were given fixed graphical and text passwords to minimize the effect of learning new passwords on the experiment. We also asked participants to think-aloud while they were completing the tasks.

7 Results and Discussion of Experiment II

We classify each task performance into one of the four categories: success (completing the task in the first attempt), dangerous success (completing the task in more than one attempt which causes security exposures), failure (not completing the task) and false completion (believing the task is completed but actually it is not). Security exposures are an issue for instance when users forgot to invoke the mechanism and the raw password is sent to the server instead [5]. Apparently, completing the task in more than one attempt does not cause any security exposure with GPEX.

The results are given in Table 2. With GPEX, all 20 participants completed all of the tasks in their first attempt. With PwdHash plug-in, the success rate was below 100% for all tasks. But none of the participants failed to complete the task or made a false completion. The results indicate that participants' password usage performances are better with GPEX plug-in compared to PwdHash.

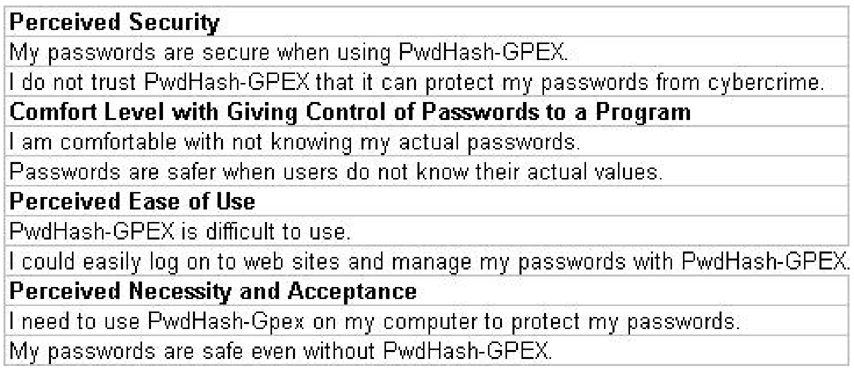
Table 2. Task performances for GPEX and PwdHash



Task Completion Result for GPEX				
	Success	Dangerous Success	Failure	False Completion
GPEX login	100%	N/A	0	0
GPEX migrate	100%	N/A	0	0
GPEX update	100%	N/A	0	0
GPEX second login	100%	N/A	0	0
Task Completion Result or Pwdhash				
	Success	Dangerous Success	Failure	False Completion
Pwdhash login	70%	30%	0	0
Pwdhash migrate	80%	20%	0	0
Pwdhash update	95%	5%	0	0
Pwdhash secondlogin	85%	15%	0	0

The questionnaire was as same as in [5]. Participants responded their level of agreement with various statements (see Table 3). A 5-point Likert scale was employed. Half of the questions were inverted to avoid bias. All participants completed the questionnaire for both browser extensions.

Table 3. Questions in the survey



Perceived Security
My passwords are secure when using PwdHash-GPEX.
I do not trust PwdHash-GPEX that it can protect my passwords from cybercrime.
Comfort Level with Giving Control of Passwords to a Program
I am comfortable with not knowing my actual passwords.
Passwords are safer when users do not know their actual values.
Perceived Ease of Use
PwdHash-GPEX is difficult to use.
I could easily log on to web sites and manage my passwords with PwdHash-GPEX.
Perceived Necessity and Acceptance
I need to use PwdHash-Gpex on my computer to protect my passwords.
My passwords are safe even without PwdHash-GPEX.

The results are shown in Figure 8. Participant believed that their passwords are secure with GPEX and PwdHash but the perceived security of two systems did not differ significantly ($t(19) = 1.677$, $p = .110$). Both systems were perceived easy to use but there was no significant difference between them ($t(19) = 0.335$, $p = .741$). Participants were not comfortable by giving control and there was no difference between GPEX and PwdHash ($t(19) = 1.748$, $p = .097$). On the other hand, Participants did not find password manager systems necessary, but the perceived necessity of GPEX was significantly higher than of PwdHash ($t(19) = 2.668$, $p < .05$).

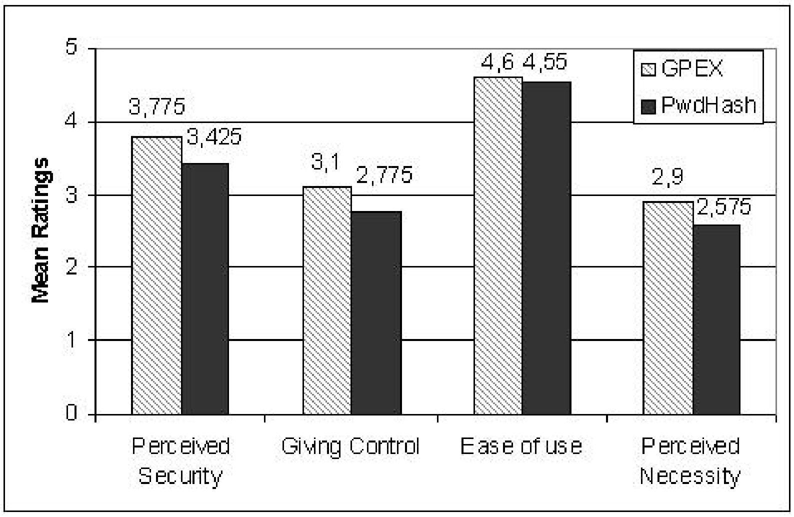


Fig. 8. Results of the survey

Some results of Experiment II are in paralel with [5]. The previous work [5] concluded that major problem of PwdHash is the invisible user interface. They reported that participants did not understand how the program works and this was due to users' inaccurate or incomplete mental model of the system. With PwdHash we replicated their findings. On the other hand, with the help of visual password interface of GPEX a correct mental model of the system is easily generated and it prevents failure in login attempts.

It is important to note that GPEX is not free of usability problems, either. Some participants found it strange to click on the empty password field. Others reported that they prefer entering a password with a keyboard.

The most important difference between the results of our usability study and results of [5] is on the task completion results for PwdHash. In our experiment, the success rates for all of the four tasks were significantly higher than the rates given in [5]. The most noticeable disagreement is that while only 16% of participants completed successfully the Update Password task in [5], the ratio for the same task was 95% in our experiment. In our experiment, participants were more familiar with using the web and they reported that they are more concerned about security of their passwords. We also observed that the participants showed great willingness to be a part of our study. However we still think that the huge difference between these two experimental results can not be explained only by change of participants and needs further research.

8 Conclusion

In this study we developed and tested GPEX, a password manager program implemented as a web browser plug-in to enable using graphical passwords to secure Internet applications without any need to change their authentication interface. The design of GPEX necessitates one-to-one mapping between click locations and a text based password. We achieved this by drawing grid lines over an image. First we conducted a laboratory experiment to understand whether drawing grid lines over an image changes the effectiveness, efficiency, user satisfaction and security of click-based graphical passwords. Results showed none of the variables degrade with drawing grid lines over an image. The usability comparison of GPEX with a text based password plug-in (PwdHash) showed that it is easier to use GPEX because users develop correct mental model of the system easily.

As a follow-up study, we will test usability and security of GPEX out of laboratory. We are also planning to extend GPEX by utilizing other graphical password methods.

Acknowledgments. We thank Osman Emre Kaya for his help on usability experiments. We thank Orhun Kara for his help on analyzing the experimental results. We also thank anonymous reviewers for their suggestions for improving the paper.

References

1. S. Madigan, Picture Memory, In John C. Yuille, editor, Imagery, Memory and Cognition, pages 65–89. Lawrence Erlbaum Associates, N.J., U.S.A., 1983.
2. J. Thorpe, P.C van Oorschot, Human-Seeded Attacks and Exploiting Hot-Spots in Graphical Passwords, pages 103-118. In 16th Usenix Security Symposium, Boston, USA, 2007.
3. B. Ross, C. Jackson, N. Miyake, D. Boneh, J. Mitchell, Stronger password authentication using browser extensions, In Proceedings of the 14th USENIX Security Symposium, Baltimore, USA, 2005.
4. J. Halderman, B. Waters, E. Felten. A convenient method for securely managing passwords. In Proceedings of the 14th International World Wide Web Conference, 2005.
5. S. Chiasson, P.C. van Oorschot, R. Biddle. A Usability Study and Critique of Two Password Managers. In 15th USENIX Security Symposium 2006, Vancouver, Canada, 2006.

6. R. Likert, A technique for the measurement of attitudes. Arch. Psychol, 140 (1932) I-5.5.
7. J. Nielsen and R.L. Mack, Usability Inspection Methods, John Wiley & Sons, Inc, 1994
8. L.F. Cranor and S. Garfinkel. Security and Usability:Designing Systems that People Can Use. O'Reilly Media,edited collection edition, 2005.
9. K. Bicakci, Optimal Discretization for High-Entropy Graphical Passwords, 23rd International Symposium on Computer and Information Sciences, IEEE ISCIS 2008, October 27-29, 2008, Istanbul, Turkey.
10. J.C. Birget, D. Hong, N. Memon, Graphical Passwords Based on Robust Discretization, IEEE Transactions on Information Forensics and Security 1(3) (Sept. 2006) 395-399.
11. S. Wiedenbeck, J. Waters, J.C. Birget, A. Brodskiy, N. Memon, PassPoints: Design and longitudinal evaluation of a graphical password system', International J. of Human-Computer Studies (Special Issue on HCI Research in Privacy and Security), 63 (2005).
12. S. Chiasson, P.C. van Oorschot, R. Biddle, Graphical Password Authentication Using Cued Click Points. ESORICS, Sept.24-27 2007, Dresden, Germany. Springer-Verlag, LNCS 4734 (2007).
13. S. Chiasson, A. Forget, R. Biddle, P.C. van Oorschot, Influencing Users Towards Better Passwords: Persuasive Cued Click-Points. HCI 2008, September 1-5 2008, Liverpool, U.K.