The impact of length and mathematical operators on the usability and security of system-assigned one-time PINs

Patrick Gage Kelley*, Saranga Komanduri, Michelle L. Mazurek, Richard Shay, Tim Vidas, Lujo Bauer, Nicolas Christin, and Lorrie Faith Cranor

University of New Mexico* and Carnegie Mellon University

Abstract. Over the last decade, several proposals have been made to replace the common personal identification number, or PIN, with often-complicated but theoretically more secure systems. We present a case study of one such system, a specific implementation of system-assigned one-time PINs called PassGrids. We apply various modifications to the basic scheme, allowing us to review usability vs. security trade-offs as a function of the complexity of the authentication scheme. Our results show that most variations of this one-time PIN system are more enjoyable and no more difficult than PINs, although accuracy suffers for the more complicated variants. Some variants increase resilience against observation attacks, but the number of users who write down or otherwise store their password increases with the complexity of the scheme. Our results shed light on the extent to which users are able and willing to tolerate complications to authentication schemes, and provides useful insights for designers of new password schemes.

1 Introduction

Personal identification numbers (PINs), or short numeric passwords, are commonly used at automated teller machines and to restrict entry into secure physical spaces. Both scenarios are potentially vulnerable to observation attacks, in which an attacker observes a user entering her password in order to learn about it. Attackers may be physically present and witness the password by looking over a person's shoulder (shoulder surfing) [21], or through recording devices (e.g., keyloggers or cameras) [17].

One solution to the problem of shoulder surfing is a *one-time PIN*, which is valid for only a single authentication. An attacker who observes a one-time PIN cannot replay it to gain access. Large numbers of one-time PINs may be computed in advance and shared between the system and the user. However, the user must have the next PIN on the list with her every time she wishes to authenticate, which requires carrying the list or having the PINs delivered on demand.[1] Alternatively, the system may display a challenge to the user and prompt her to use a shared secret to compute a response that demonstrates that she knows the secret. Such challenge-response systems typically involve cryptographic functions that require the use of a computational device. However, it may be more convenient for a user if she can compute a response in her head.

Some graphical-password schemes allow the user to derive the correct one-time response from a combination of the screen display and their knowledge of the secret [7,

[1] http://gmailblog.blogspot.com/2011/02/advanced-sign-in-security-for-your.html

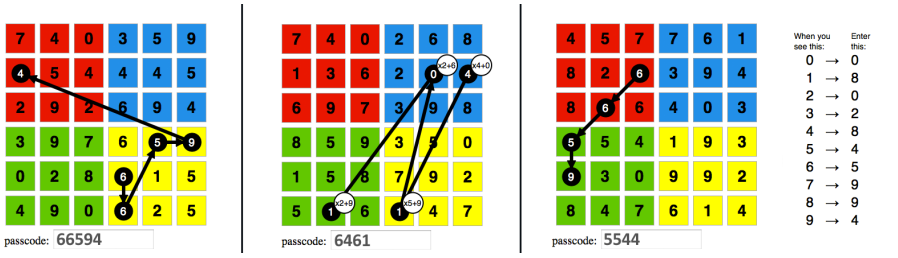

Fig. 1. The first image shows PGlength5, a standard length-five PassGrid. The second shows PGx+4, a length-four PassGrid with four different multiplication-addition rules. The third shows PGcodecard, a length-four PassGrid where each element must be translated with a single function, always shown to the right of the grid (The first grid element is a 6, so a user would enter a 5).

26]. These schemes offer some convenience, but only modest advantages over PINs in terms of resistance to observation attacks. These schemes can often gain observation attack resistance by requiring the user to remember a longer secret or perform simple mathematical operations. However, the tradeoffs between usability and security that such schemes may present have not been studied previously.

In this paper we explore the usability and security benefits of enhancing system-assigned one-time PIN systems with longer secrets or mathematical operators. We present a usability case study in which we analyze *PassGrids*, an implementation of a one-time PIN authentication mechanism in which users memorize a secret pattern on a 6x6 grid. Each time a user attempts to authenticate, she is presented with a grid filled with random digits, and she enters the digits that correspond with the elements of her pattern. While our study is limited to a specific authentication system, our approach allows us to examine security and usability tradeoffs that are generally applicable to a range of system-assigned one-time PIN systems as well as other authentication systems. Rather than testing the PassGrid scheme per se, we primarily use it to assess the relative security gains and usability impacts associated with adding various complications to its base design.

Surprisingly, we find that neither increased length nor mathematical operators greatly impacts usability. Although added complexity generally reduces user enjoyment, it does so far less than could be expected. However, added complexity does increase the tendency of users to write down or otherwise store their passwords. Using mathematical operators provides larger security gains than lengthening the pattern, while achieving similar usability. We also find that users are able to perform basic modular arithmetic operations as part of the authentication process, but dislike having to remember and perform multiple operations. More generally, our results shed some light on the extent to which users are willing and able to tolerate complications to authentication schemes, which in turn could be useful to designers of new schemes.

The remainder of this paper is organized as follows: In Section 2 we review related work and present the PassGrids case study in detail. Section 3 explains our methodology and reports on our study participants. In Section 4 we present our security analysis and main usability results. Section 5 concludes with a discussion of the implications of our findings.

2 Background and Related Work

In this section we review related work on graphical one-time PINs and graphical passwords, and introduce PassGrids, the scheme whose variants we focus on in this paper.

Graphical one-time PINs Graphical one-time PIN systems are a subset of one-time PIN systems, in which the authentication challenge is presented graphically. One example is the GrIDsure system,[2] studied by Brostoff et al. in an 83-participant user study [7]. GrIDsure is similar to PassGrids, with a five-by-five grid and user-selected patterns. Since users tend to select somewhat predictable patterns, this reduces the effective password space [22]. This erosion of practical entropy, along with other security issues related to graphical one-time PINs, is detailed by Bond [6].

Other examples of graphical one-time PINs are PassFaces and the commercial GridPIN system [8, 20]. In one variation of PassFaces, the user enters a one-time PIN calculated by locating previously selected pictures of faces within a grid. GridPIN displays a keypad in which each digit is surrounded by eight smaller digits; the user selects a direction (e.g., bottom left), and enters a one-time PIN calculated by locating her original PIN digits and then selecting the associated smaller numbers based on her direction.

We expand on previous studies of graphical one-time PINs by conducting a large online user study that examines a larger six-by-six cell grid while varying pattern length as well as the use of mathematical operators. Our findings provide insight into the tradeoffs between usability and resistance to observation attacks for this class of systems.

Graphical password schemes For more than a decade, various graphical password schemes have been proposed to combat weaknesses of text passwords. Surveys by Biddle et al. [5] and earlier by Suo et al. [19] provide a comprehensive discussion of the breadth and history of graphical passwords. We focus here on a subset of graphical password schemes that Biddle et al. refer to as *recall-based systems*, in which users reproduce a secret.

Recall-based systems, or authentication through "what you know," are a general class that also includes text passwords. Many types of recall-based, single-factor authentication are subject to observation attacks (e.g. shoulder-surfing). When a user provides input to the authenticator, an attacker can observe the secret, effectively allowing the attacker to impersonate the user in the future. The quintessential example of this attack is during PIN entry at an ATM [2]. Sophisticated attacks may infer passwords from keypad acoustics or electromagnetic emanations from computer displays.

Many graphical recall-based systems require users to draw a sketch or pattern of their own creation, normally on a grid [5, 14]. A simple version of this is the current Android phone-unlock screen, where users trace a pattern of their own choosing on a three-by-three grid. The Android system is susceptible to shoulder-surfing attacks and to "smudge" analysis [4].

Weiss and DeLuca introduce a highly memorable graphical-password system using shapes [24]. Unfortunately, this scheme provides no more security against observation attacks than do traditional passwords [10]. This is not atypical, as many recall-based

[2] http://www.gridsure.com/

graphical passwords are vulnerable to single-observation attacks, and others are resistant to attacks after a small number of observations [12].

One example of a graphical password system designed to resist observation attacks is the convex hull click system, in which users must click a point within the convex hull created by locating the correct icons within a field of other icons [25]. This system has been shown to be vulnerable to repeated observation attacks based on the frequency with which the secret icons appear as compared to other icons [3]. To address this, previous work has looked at obscuring part of the challenge-response from an attacker [10, 18]. When successful, these systems are resistant to any number of observation attacks, but become vulnerable when an attacker can observe all parts of an authentication.

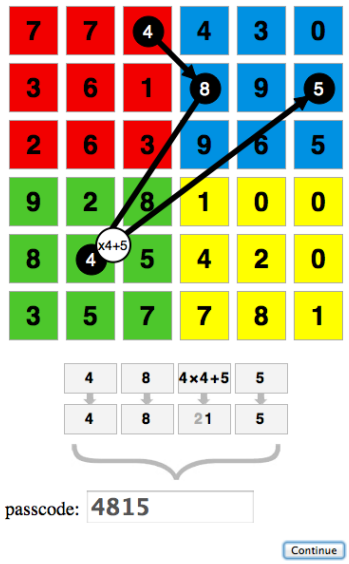

Fig. 2. An excerpt from the tutorial we developed to explain patterns with operators. Here, a participant assigned the PGx+1 condition must modular multiply the third digit in their pattern by the constant '4' and then add '5.' If the result is greater than 10, they should enter only the digit in the ones place, as shown in the example.

PassGrids For this study, we implemented the user interface for a graphical one-time PIN system based on designs provided by PassRules US Security LLLP, creators of the "It's Me!" graphical one-time PIN system. We call our implementation "PassGrids," a name we made up for the study. We also created video-based tutorials and a JavaScript animation to teach study participants how to use PassGrids. PassGrids examples can be seen in Figures 1 and 2.

The PassGrids user interface contains a six-by-six grid and a password text-entry field. The grid displays the challenge: 36 colored squares, each containing a single randomly generated digit. Each quadrant of squares is a different color: red, blue, yellow, or green. The user's secret is a *pattern*, formally an n-tuple, of locations on the grid, that users memorize. The grid shape and color are intended to aid the user in remembering the pattern and are the same for every user.

To authenticate, a user identifies the digits that correspond to the grid locations in her pattern and enters them in the text field using a keyboard or number pad. We will refer to the resulting one-time PIN entered by the user as a *passcode*. For example, a pattern of length five would require a user to memorize five locations in order, as shown on the left grid in Figure 1. On the left grid, the user would enter 66594 to authenticate.

The random digit generation is constrained so that each digit appears either three or four times in each six-by-six grid.[3] This means any user input matches multiple patterns in the grid. For example, in the center grid in Figure 1, "8440" matches 144 permuta-

[3] Six digits appear four times, four digits appear three times.

tions of points. If grids were randomly generated without this constraint, some digits might appear only once in some grids greatly reducing the number of permutations.

In this study, we tested several variations on system-generated PassGrid passwords, selected to represent a range of resistance to observation attacks. These variations, which include varying the length of the pattern and requiring users to apply mathematical operators to elements of the pattern, are described in detail in Section 3.1.

3 Methodology

To test multiple variations of PassGrids and PINs, we conducted an online study using Amazon's Mechanical Turk service.[4] Mechanical Turk facilitates the recruitment of workers to complete short online tasks for small payments. Despite concerns about blindly relying on Mechanical Turk [1], several studies have found that properly-designed MTurk tasks provide high-quality user-study data, with much more diverse participants than are typically available in lab-based studies [9, 11, 13, 15, 23].

We conducted a two-part study with 1600 participants, using an experimental protocol similar to that used previously to study password-composition policies for text passwords [16].

In part one, we assigned each participant a four-digit numeric PIN or a PassGrids variant, described in the Conditions subsection below. Throughout this paper, and in all communications with participants, we refer to a participant's assigned PIN or PassGrids pattern as her password. We told users to imagine their password was assigned to them for use with their main email account after their previous password was compromised, and we asked them to behave as if this were their real password. While this hypothetical scenario may not produce the same results as a situation in which users actually use their passwords to protect high-value accounts, any behavioral bias introduced by this hypothetical scenario would likely impact all experimental conditions similarly, and thus the impact on our comparative analysis should be small.

3.1 Conditions

Participants were assigned to one of the following eight conditions, selected to represent a range of resistance to observation attacks.

- **PGbasic.** Participants were assigned a randomly generated four-element PassGrids pattern. In all conditions no location in the grid appeared more than once in a pattern.
- **PGlength5.** Participants were assigned a randomly generated five-element PassGrids pattern.
- **PG+1.** Participants were assigned a random length-4 PassGrids pattern. A randomly generated addition operator (for example, add 4) was applied to a randomly selected element of the pattern.
- **PG+4.** Participants were assigned a random length-4 PassGrids pattern. Separate randomly-generated addition operators (for example, add 4, add 1, add 4, and add 9) were applied to each of the four pattern elements.

[4] http://mturk.amazon.com

– **PGx+1.** Participants were assigned a random length-4 PassGrids pattern. A randomly-generated multiplication/addition operator (for example: multiply by 3, then add 4) was applied to one of the pattern elements, also randomly selected.
– **PGx+4.** Participants were assigned a random length-4 PassGrids pattern. Separate randomly-generated multiplication/addition operators (for example: multiply by 3, then add 4; multiply by 2, then add 8; multiply by 3, then add 5; multiply by 5, then add 2) were applied to each of the four pattern elements.
– **PGcodecard.** Participants were assigned a random length-4 PassGrids pattern. Participants were told that a "swap" function would be applied to the numbers they typed in. For example, whenever they saw a 3, they must enter a 0. They were also told the entire translation would always be shown to the right of the grid, in a table we call a *codecard*. This condition roughly simulates providing each user with a paper codecard, which could be carried in her wallet and used for each login. The protocol design assumes a best-case scenario, one where the participant cannot lose the card or leave it at home. Note that if the codecard is also observed, this condition has security properties similar to PGbasic.
– **PIN.** Participants were assigned a randomly generated 4-digit PIN.

In each of the above conditions, the user of the system must memorize the pattern (the location of cells in the grid) as well as any operators (including type and quantity) that they must apply. The exception is PGcodecard which always displays the function box beside the grid.

In any condition involving mathematical operators, the math is modulo 10: once the result has been calculated, only the one's place digit is retained, so that the final passcode has the same number of digits that the pattern has cells. From left to right, Figure 1 illustrates conditions PGlength5, PGx+4, and PGcodecard. Figure 2 illustrates modular math for PGx+1.

3.2 Protocol details

Participants in each condition were first shown an introductory video. The video welcomed them to the study and showed two examples of a basic password (a length-4 PassGrid or a length-4 PIN). Participants in the non-basic conditions were shown a third example, in the style they would be assigned, to demonstrate how the operator(s) were used or how the codecard function worked. The videos themselves ranged in length from 28 seconds for PIN to 117 seconds to PGx+4

Immediately after the video, participants were assigned their password. For PassGrids participants, the password was animated on the screen, and math, if present, was detailed below the grid. A screenshot of this is shown in Figure 2. We then told participants they would need to successfully authenticate three times using the password they had just been assigned. After each attempt, we prompted them to enter the password again, until three successful authentications were achieved. After any three consecutive incorrect attempts, we displayed the password again. Throughout the process, a counter at the top of the screen reminded participants how many more authentications were needed. This set of authentications is considered the "practice" period.

After three successful authentications, participants were presented with 24 randomly selected, single-digit arithmetic problems: eight addition problems, eight multiplication problems, and eight hybrid multiplication/addition problems. These problems served as a distractor task between password entry attempts, as well as to measure the speed and accuracy with which participants could perform simple arithmetic problems like those used in some PassGrids variants. To mirror passcode entry, we instructed participants to perform modular arithmetic, saying: "For example, if you were to see the following addition problem: 4 + 9 = 13, Enter only '3' into the box."

Next, participants completed an online survey about demographics, password habits, and opinions of the password system used in the study.

Finally, participants were required to authenticate successfully one more time to complete the first part of the study. (Again, we displayed the password after three unsuccessful attempts.) We told participants we would contact them for follow-up surveys and displayed a completion code that they entered into Mechanical Turk to receive a 55-cent payment.

Two days after a participant completed part one, we sent an email asking her to return for part two of the study for a 70-cent bonus payment. URL customized for each participant. Participants who returned were asked to recall their passwords. Those who failed to recall their password after three tries were shown their password. Participants were then presented with a second survey, which included additional questions about password creation, storage, and usage.

3.3 Participants

Over a five-week period in August and September 2011, 4731 participants began our study. Of those, 3250 (68.7%) completed part one; the other 1481 (31.3%) are discussed in Section 4.6. Of the 3250 who completed day one, 2000 returned and successfully completed day two. From each condition, we selected the first 200 participants that successfully completed both days; unless stated otherwise, our analysis focuses on those 1600 participants.

The mean age of these participants was 30; 843 (52.7%) reported being male and 739 (46.2%) female. 449 (28.1%) reported studying or working in a technical field, and 195 (12.2%) in art or design. With Kruskal-Wallis and χ^2 tests, there were no significant differences between conditions for any of these characteristics.

4 Results

Across the PIN condition and PassGrids variations we tested, we first evaluate the security properties of each variation and then explore how successfully participants authenticated (accuracy), whether they memorized or stored their passwords (memorability), how they felt about the system (perception), the rate at which potential participants dropped out, and how much time was required to successfully log in. Our results show that all PassGrids conditions are more resistant to a single observation attack than PINs, but that with multiple observations PassGrids variants can also be compromised. We found that most variations of PassGrids are entered less accurately on first use by

users than PINs, though users quickly comprehend how to authenticate with the system. Users report PassGrids to be a little bit more difficult but considerably more fun than PINs. We found that although users can generally authenticate surprisingly accurately even with arithmetic operations, adding such operations to PassGrids increases the rate of dropout from the study, decreases enjoyment, and greatly motivates people to write down information about their PassGrid password.

4.1 Security

We examined several security metrics for evaluating PassGrids: password space, passcode strength, and resistance to observation attacks. We focus most on observation attacks, and consider a particular threat model in which the attacker is given three chances to authenticate after n observations.

Password space and passcode strength One simple measure of security is *password space*, or the set of all possible passwords that could be assigned. In PassGrids, the password space can be increased by increasing the length of the pattern, increasing the size of the grid, or introducing mathematical operators. Conversely, the password space could be reduced by pruning patterns from the password space, for example patterns with points far apart from each other might be removed in an attempt to improve usability. Table 1 quantifies the password space for each of our conditions.

Another security metric is *passcode strength*. With a randomly assigned PIN, all passcodes are equally likely. If an attacker with no knowledge of the user's password guesses a PIN at random, he has a 1 in 10,000 chance of gaining access. As a result, the password space and passcode strength are the same for randomly assigned PINs. With PassGrids, however, the probability of success from guessing a random passcode (with no knowledge of the user's password) can increase if the attacker analyzes the grid presented at login time. Since some digits are repeated more than others in the grid, some passcodes might be more likely to grant access than others. We can measure this effect by examining the distribution of passcodes produced by each PassGrid scheme.

condition	possible passwords	**% guessed after n observations** 1	2	3	4	5	6	7
PIN	1.0E+4	100						
PGbasic	1.4E+6	6.0	96.3	100				
PGlength5	4.5E+7	2.2	93.9	100				
PG+1	5.7E+7	2.7	64.5	99.2	100			
PG+4	1.4E+10	0.3	7.6	90.3	99.9	100		
PGx+1	5.7E+8	0.2	11.2	51.5	82.3	93.6	97.4	98.9
PGx+4	1.4E+14	1.2	3.32	18.6	67.9	91.2	97.0	99.0
PGcodecard	1.4E+16	0.4	0.8	1.0	1.4	2.1	5.0	21.6

Table 1. Experimental conditions, shown in order of resistance to observation attacks, with number of possible passwords. Note that if the codecard is also observed, PGcodecard behaves similarly to PGbasic.

Our analysis finds that the attacker's benefit from this kind of grid analysis is negligible. The weakest of the conditions we considered was PGbasic, in which an attacker has a 1.8 in 10,000 chance of gaining access with this kind of educated guess. Therefore, an attacker would still require a large number of guesses to gain access. Some

of the PassGrids conditions we tested — PG+4, PGx+4, and PGcodecard — have the same passcode strength as a randomly assigned PIN. The other three PassGrids conditions have more passcode strength than PGbasic but less than a corresponding PIN. As a result, we don't consider passcode strength a very useful security metric for comparing these systems.

Estimating observation resistance With traditional PINs, only a single observation is required for an attacker to learn the password, because a user's PIN is always the same. With PassGrids, the passcode is a function of the randomly generated grid and the user's password (pattern and operators), where each passcode maps to multiple unique passwords. Nevertheless, PassGrids are not immune to observation attacks.

With each observation, the attacker can reduce the space of possible passwords (ignoring degenerate cases where the same grid is observed multiple times). To provide an intuition of how this works, imagine that the attacker observes a victim in the PGbasic condition enter a passcode of "1234." If the digit "1" appears in four different cells in the grid, then the attacker knows that the first element of the victim's pattern must be in one of those four cells. If the attacker observes the victim again, he can eliminate any cells that don't correspond to the first digit the victim enters on the second observation. After a sufficient number of observations, the space can be reduced to a single password.

In our threat model, though, we allow the attacker to make three guesses before being locked out. Because passcodes map to multiple passwords, this gives the attacker more power than one might expect — for example, each incorrect guess can eliminate multiple passwords. After developing an algorithm which makes optimal guesses in this way, we used simulations to estimate the strength of PassGrids against observation attacks. This allowed us to quickly test many PassGrid variants. Our threat model assumes the attacker can see the complete grid and the victim's passcode in each observation, as might be available in an ATM skimming attack. We also assume the attacker knows the victim's password policy, i.e. the space of possible patterns and operators from which the password was assigned.

In each simulated observation attack, we randomly select a password from the password space S and generate n observations, i.e. random grids and corresponding passcodes. Our algorithm receives this data and removes any passwords from S which could never have produced the given data. The algorithm is then given the chance to authenticate with a new grid. It selects the most likely passcode to guess based on the remaining passwords in S. If this guess fails, the algorithm uses this failure to prune the space before guessing again on a new grid. The simulation is counted as successful if the algorithm's passcode is accepted within three guesses.

Observation attack results Table 1 presents the results from 980,000 simulations run on the PassGrids conditions. For each condition and number of observations, we report the percentage of the 20,000 simulated attacks that were successful.

Overall, we see that all of the PassGrids conditions are better than PINs against a single observation, as might occur in an opportunistic shoulder-surfing attack. However,

even PGcodecard is compromised after six observations. This is realistic if, for example, an attacker uses a hidden camera to record the same victim multiple times.

To compare between PassGrid conditions, we can select a threshold for success probability. For example, if we consider a condition compromised when 5% or more of the attacks succeed, then both PIN and PGbasic are compromised after a single observation, although PGbasic poses a greater challenge to the attacker. Conditions PGlength5, PG+1, PG+4, and PGx+1 are compromised after two observations, PGx+4 is compromised after three observations, and PGcodecard is not compromised until six. Choosing different cutoff points would result in different equivalence classes among the conditions.

A possible variation on PassGrids restricts the space of possible patterns by choosing only cells that are relatively close to one another, in an effort to increase usability. A similar approach is to allow users to select their patterns, which we expect them to do non-uniformly. This leads to a reduction in password space and increased vulnerability to observation attacks. To evaluate this, we simulated observation attacks on a variant of PGbasic in which patterns were selected such that the Euclidean distance between cells did not exceed a given threshold.[5] Our analysis indicates that this scheme provides little to no benefit over traditional PINs against observation attacks, which were successful more than 50% of the time after just one observation. We did not test this variant further.

It is important to note that these attacks require a relatively powerful attacker who can record both the grid and the victim's passcode, and calculate the optimal passcode to try on each attempt. Many realistic attackers, for example, an opportunistic shoulder-surfer, may be weaker.

4.2 Math Results

We analyzed participants' responses to the 24 random modular arithmetic problems to determine whether any of the operators we tested would be particularly problematic. Overall, participants completed these problems accurately, getting 96%, 94%, and 92% of addition, multiplication, and combined multiplication addition problems correct, respectively. Each problem type differs significantly in the proportion participants got right (Holm-corrected FET, $p < 0.05$). The mean number of incorrect problems per participant was 1.4; only 83 participants (5%) got nine or more problems wrong (two standard deviations above the mean).

The mean completion time was 5.0 seconds per addition problem, 5.6 seconds for multiplication, and 8.0 seconds for combined problems.[6] These results suggest that including simple modular arithmetic does not pose a significant barrier to authentication for our population (Mechanical Turk workers); it could be more problematic for others.

[5] The threshold was set to $(3 * \sqrt{2}) = 4.24$. This number allows for a pattern that has all four points on a diagonal and reduces the total number of possible patterns to $10,060$.

[6] Measured from page load to submission of answer, which includes the time needed to load the page, navigate the mouse/cursor to the answer field, type the answer, and submit.

4.3 Accuracy

Participants authenticated with our system five times: three times on day one immediately after being assigned a PIN or pattern, once more on day one after the survey and math questions, and a fifth time when they returned for day two. We consider a participant to have successfully logged in if she can authenticate within three tries (before she is shown her password again). Many participants performed poorly in the first trial, as they first used the system, but by the end of the third trial most participants seemed to grasp password entry. Figure 6 (Appendix A) shows the percentage of participants who successfully logged in per condition, per trial.

Surprisingly, by the final authentication all conditions show similar accuracy, despite the large differences in success in the previous trials. This indicates that after some practice, users can authenticate just as successfully with the complicated conditions as with the simpler ones. In this final trial, no differences between conditions were found using Fisher's exact test at the $\alpha = 0.05$ significance level. (Seven pairs of conditions were selected a priori and tested without correction; all other pairs were Holm-corrected.)

We also examined the types of mistakes participants made, finding that many mistaken entries were very close to correct. Across the PassGrids conditions, 308 of 1400 participants (22%) made errors in the day two recall. In 7% of these cases, the participant entered a passcode with too many digits; 16% used too few. 65% percent of participants who made errors used only one wrong digit in their passcode; 33% of these got the last digit wrong. 13% of mistaken participants entered the right digits in the wrong order, and 6% entered a passcode that appears to result from transposing their pattern to an incorrect starting cell. Forty-three of 1000 participants in conditions with operators (4%) entered a passcode for the correct pattern with no operators. Note that individual participants may exhibit more than one type of error.

4.4 Storage and memorability

We use storing behavior as a rough proxy for perceived memorability; that is, conditions in which participants wrote down their passwords more often can be considered harder to remember. During the survey at the end of day two, we asked, "Did you write down or store any information to help you remember your password pattern? (please be honest, you get paid regardless, this will help our research)." The results of this question are shown in Figure 3.

Unsurprisingly, patterns in conditions that are intuitively more difficult, specifically those where participants needed to memorize more information, were more frequently written down. PGx+4 (83%) and PG+4 (75%) were each stored significantly more often than all the other conditions.[7] PGcodecard, while much more resistant to observation attack, was not significantly different in storage frequency from a standard length-4 PassGrid (44% and 43% respectively). This is not surprising, because we always showed the codecard on the screen; in practice, the user would need to store it.

[7] All comparisons in this paragraph HFET, $\alpha = 0.05$.

Surprisingly, some PassGrids conditions, such as PGbasic, were stored significantly less frequently than PINs (43% and 60% respectively). However, that difference may be caused, at least in part, by the fact that writing down a pattern is less straightforward and familiar than writing down a PIN, rather than by differences in perceived memorability.

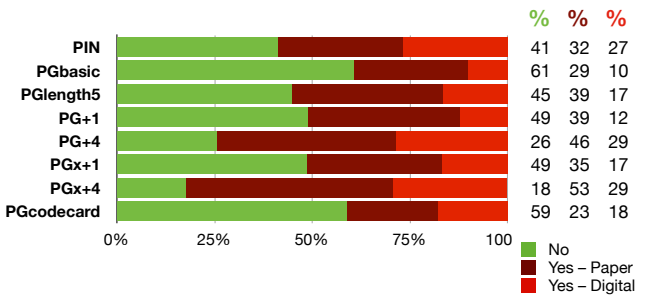

Fig. 3. Percentage of participants who stated they did or didn't store their password, by condition.

We also compared accuracy between participants who said they did not write down their password and those who said they did. We show the results in Figure 4. Across all conditions, 87% of participants who wrote down their passwords authenticated successfully on day two. The success rate for those who did not write down was only 76%, a significant difference (FET, $p < .001$). Within conditions, we saw no significant difference in accuracy between writers and non-writers for PIN and PGbasic. In PG+4 and PGx+4, writers were significantly more accurate than non-writers (FET, $p < .006$). (These conditions were selected a priori for significance testing.)

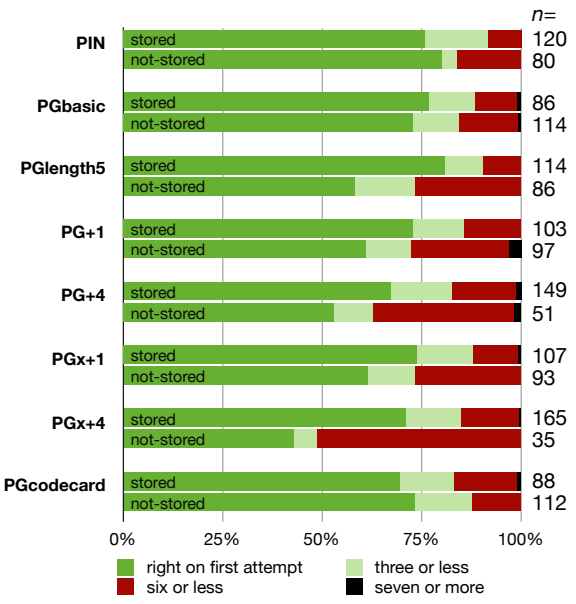

Fig. 4. For this analysis, we split the participants in each condition into two groups, those who stored their password (top) and those who did not (bottom), and compared their day 2 login success rates. Note sample sizes are different, and noted on the right. We a priori selected PIN, PGbasic, PG+4, and PGx+4 for significance testing, only the latter two showed significant differences (FET, $p < .006$).

As a rough estimate of memorability, we also consider how many participants did not write down their password, yet authenticated successfully on day two. Just 33.5% of our 200 PIN participants didn't write down their password and still logged in; this is similar to PGx+1, at 34.0%. Excepting PGcodecard, where participants were told they did not need to memorize or store the translation, the condition that performed

best was PGbasic, where 48% of participants successfully authenticated on day 2 without password storage. In the worst case, PGx+4, only 8.5% of participants were able to successfully log in without password storage, suggesting it may not be tractable by memory alone.

One further note is that some participants indicated they were not storing the entire PassGrid password, but only the operators they needed to apply.

4.5 User Perception

To explore user perception of PassGrids, we asked a series of Likert questions on day two (1-5 from strongly agree to strongly disagree). Three of these were: "Using PassGrids was ..." annoying, difficult, and fun. A fourth asked the participant if remembering their password was difficult. We graph the responses in Figure 7 in Appendix A.

Our results indicate that most PassGrids variations are more fun than PINs. Some are also more difficult, but in some cases additional complexity can be achieved without decreasing usability.[8] PGbasic and PGlength5 performed best, being rated significantly more fun than PIN but not significantly different in annoyance or difficulty. The other PassGrids conditions were significantly worse than PIN in annoyance and difficulty.

Comparing PassGrids conditions to each other, PGx+1 was not significantly different in user perception than PG+1, but users rated it significantly easier to remember and use than PG+4. PGx+4 was the worst overall.

4.6 Dropouts

While there are many reasons for participants to leave a study, it is likely the dropout rate can be abstracted as a rough metric for difficulty and user frustration. We examined the number of participants who accepted the task from Mechanical Turk, but then left the study before completing day one, and found that the rate at which users dropped out of our study increased roughly in line with the number of operators added in the PassGrids conditions. The results are shown in Figure 5.

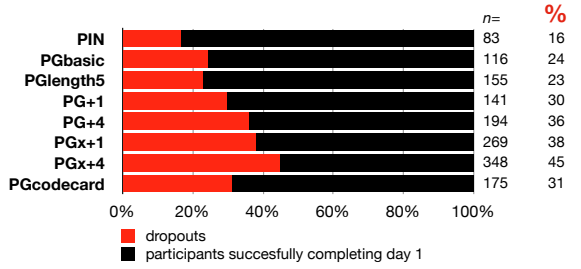

Fig. 5. Percentage of participants who upon accepting the MTurk task successfully completed day one. Ranges from 16.4% for PIN to 44.8% for PGx+4.

The dropout rate among participants assigned to PassGrids was significantly higher than among PIN participants (33% and 16% respectively).[9] Within PassGrids conditions, those in PGlength5 were not significantly different from those in PGbasic, but were significantly less likely to drop out than those in PGcodecard. Similarly, PGx+1

[8] All comparisons in this paragraph HFET, $\alpha = 0.05$.

[9] All comparisons in this section HFET, $p < 0.001$.

was not significantly different from PG+4, but had significantly more dropouts than PG+1. Lack of a significant difference does not mean that dropout rates were the same, but it does indicate that the size of any difference is small. On the other end of the spectrum, PGx+4 participants were nearly as likely to drop out as they were to complete day one (44.8%).

We did find that in conditions with the highest dropout rates (PG+4, PGx+1. PGx+4) accuracy on the math problems was highest (94-96%, compared to 91% in PIN). This may mean that those participants who dropped out were not as confident at mathematics. While we might expect demographic differences between conditions due to the differences in dropout rates, we did not see that.

4.7 Timing Information

Password authentication includes many subtasks, such as reading the web page, remembering the password, entering the password, and retrieving written notes. Identifying and timing these subtasks in an online study is infeasible. Here, we present instead two measures of timing: *entry* time and *login* time. Entry time estimates the amount of time spent entering the password and is taken from the first successful attempt of the third authentication. We used the third practice entry here, to attempt to reduce the impact of memory-based recall, focusing just on time to actually authenticate, as the participants had just done this twice prior. Login time encompasses an entire authentication, including unsuccessful attempts, and is taken from the final authentication. All times were measured from server-side events, which do not account for client-side delays like page loading. Therefore, these times might be overestimates. The timing data is shown in Table 2.

In entry time and login time, PIN is the clear winner, as expected. Among PassGrids conditions, the cost of additional complexity in authentication time is clearly illustrated. Median entry times range from 12 sec for PGbasic to 38.5 sec for PGx+4. Login times are almost three times longer, even though the mean number of attempts required on day two was 1.8. This suggests that authentication takes longer when users haven't used their password in several days.

4.8 Tutorial

A potential problem with comparing PassGrids to PINs is pre-existing user familiarity with PINs. People have seen and used PINs many times, but are likely completely unfamiliar with one-time graphical PIN systems in general, as well as with our specific implementation. To attempt to address this, we created a series of video tutorials, based on PassRules system specifications but with some modifications for our conditions. Since this was an online study, we have no way of knowing if a video was actually watched. Users may have covered the video with another application, muted the audio, or otherwise underutilized the tutorial.

We recorded the number of times participants returned to the video tutorial after being shown their password. Of the 1400 participants across the seven PassGrids conditions, 363 participants (26%) returned to watch the tutorial at least once, with most returning only once (299 participants), and one participant returning 5 times.

Despite the tutorials, we found there was still some confusion about PassGrids. In the free-response portion of the day 1 survey, many participants described the concept as both "interesting" and "new." Some found it confusing, and it is unclear whether they understood that the passcode would be different each time. When asked how they remembered their pattern, participants gave responses such as, "just keep retrying the combination" or "I had a very hard time with remembering due to the fact that you changed the numbers around on the side and I had to put different numbers for each number." From such comments, it seems that improving the tutorial so that more participants truly understand how PassGrids work could prove beneficial.

	entry time (s)	login time (s)
PIN	6.0	20.0
PGbasic	12.0	35.0
PGlength5	15.0	51.0
PG+1	15.0	51.5
PG+4	23.0	74.5
PGcodecard	20.0	56.0
PGx+1	17.0	50.0
PGx+4	38.5	96.5

Table 2. Median login times in seconds per condition.

5 Discussion

Our results show that a system-assigned one-time PIN system such as PassGrids is a viable PIN replacement for systems where observation attack prevention is a priority. While not invulnerable against observation attacks, attackers must be technologically assisted, with complete knowledge of the grid and a non-trivial algorithm for determining passcodes with a high likelihood of success.

We found that several methods can be used to increase the security of PassGrids, including increased length, mathematical operators, and codecards. Modular arithmetic is not difficult for our participants when explained in straightforward terms.

While mathematical operators do provide additional security without suffering a loss in ability to authenticate, there are substantial usability drawbacks as the complexity increases. Participants considered our most difficult condition, PGx+4, substantially more difficult to remember and use. We also saw greatly increased rates of password storage, and we lost nearly half of our incoming participants in that condition, more than any other. All this together suggests math should be used in moderation, with as few constants as possible and a minimal number of pattern elements affected. Additionally, we must keep in mind that increased complexity here will lead to more password storage, which depending on the threat model may be more harmful than the benefits gained.

Increasing length did result in slight observation resistance gains, with little difference in accuracy, reported enjoyment, or timing, and only a slight increase in storage; however, it is likely that length cannot continuously be increased without more substantial usability losses.

Finally, the codecard functionality allowed us to examine a much greater observation resistance, by increasing the space of the translation in a more diverse way than simple operators will ever allow. Overall, PGcodecard performed well, but the requirement of a written source that must be kept secret in order to achieve the observation resistance benefits may not in practice be usable. Additionally, this requirement may not align with the motivation behind one-time PIN systems, especially if the goal is to allow participants to log in, simply from memory, as with a standard PIN.

Applying similar types of modifications to other systems, such as the closely related GrIDsure system is straightforward; users could select longer patterns, or select operators in combination with their patterns. GridPIN could be extended in a similar way, after using the displayed keypad and selected direction to map the original PIN digit to a one-time digit, an additional mathematical operator or codecard could be applied. A similar modification to PassFaces might require users to find a starting number associated with the correct face, then modify that number using mathematical operators or a codecard.

We tested modifications on a basic one-time PIN system, increasing the length, adding mathematical operators or a digit translation, each designed to increase the observation resistance of the system. We measured how these modifications affected the usability of a system in various ways including memorability, storage, enjoyability, and accuracy. We believe that these modifications and the results we described here are not unique to this system, but give password system designers an understanding of how these techniques can enhance the security of other one-time PIN systems.

6 Acknowledgments

This research was supported by NSF grants DGE-0903659 and CNS-1116776, by CyLab at Carnegie Mellon under grants DAAD19-02-1-0389 and W911NF-09-1-0273 from the Army Research Office, by Air Force Research Lab Award No. FA87501220139, and by gifts from PassRules US Security LLLP and Microsoft Research.

References

1. E. Adar. Why i hate mechanical turk research (and workshops). In *Proc. CHI Workshop on Crowdsourcing and Human Computation*, 2011.
2. R. Anderson. Why cryptosystems fail. In *ACM CCS'93*, pages 215–227, 1993.
3. H. J. Asghar, S. Li, J. Pieprzyk, and H. Wang. Cryptanalysis of the convex hull click human identification protocol. In *ISC'10*, pages 24–30, 2011.
4. A. J. Aviv, K. Gibson, E. Mossop, M. Blaze, and J. M. Smith. Smudge attacks on smartphone touch screens. In *WOOT'10*, pages 1–7, 2010.
5. R. Biddle, S. Chiasson, and P. van Ookrschot. Graphical passwords: Learning from the first twelve years. In *ACM Computing Surveys (to appear)*, 2011.
6. M. Bond. Comments on gridsure authentication. `www.cl.cam.ac.uk/~mkb23/`, 2008.
7. S. Brostoff, P. Inglesant, and M. A. Sasse. Evaluating the usability and security of a graphical one-time PIN system. In *BCS Conference on HCI*, 2010.
8. S. Brostoff and A. Sasse. Are passfaces more usable than passwords? a field trial investigation. In *HCI'00*, pages 405–424, 2000.
9. M. Buhrmester, T. Kwang, and S. D. Gosling. Amazon's Mechanical Turk: A new source of inexpensive, yet high-quality, data? *Perspectives on Psychological Science*, 6(1):3–5, 2011.
10. A. De Luca, M. Denzel, and H. Hussmann. Look into my eyes!: Can you guess my password? In *SOUPS'09*, pages 1–12. ACM, 2009.
11. J. S. Downs, M. B. Holbrook, S. Sheng, and L. F. Cranor. Are your participants gaming the system? Screening Mechanical Turk workers. In *Proc. CHI*, 2010.
12. P. Golle and D. Wagner. Cryptanalysis of a cognitive authentication scheme. In *IEEE SP'07*, 2007.

13. M. Jakobsson. Experimenting on Mechanical Turk: 5 How Tos. http://blogs.parc.com/blog/2009/07/experimenting-on-mechanical-turk-5-how-tos/, July 2009.
14. I. Jermyn, A. Mayer, F. Monrose, M. K. Reiter, and A. D. Rubin. The design and analysis of graphical passwords. In *USENIX Security Symposium*, pages 1–1, 1999.
15. A. Kittur, E. H. Chi, and B. Suh. Crowdsourcing user studies with Mechanical Turk. In *Proc. CHI*, 2008.
16. S. Komanduri, R. Shay, P. G. Kelley, M. L. Mazurek, L. Bauer, N. Christin, L. F. Cranor, and S. Egelman. Of passwords and people: Measuring the effect of password-composition policies. In *CHI 2011*, 2011.
17. B. Krebs. ATM skimmers: Hacking the cash machine. http://krebsonsecurity.com/2011/04/atm-skimmers-hacking-the-cash-machine/, 2011.
18. H. Sasamoto, N. Christin, and E. Hayashi. Undercover: authentication usable in front of prying eyes. In *SIGCHI'08*, pages 183–192. ACM, 2008.
19. X. Suo, Y. Zhu, and G. S. Owen. Graphical passwords: A survey. In *ACSAC'05*, pages 463–472, 2005.
20. SyferLock. Syferlock technology. http://www.syferlock.com/day1/demovidpin.htm.
21. F. Tari, A. A. Ozok, and S. H. Holden. A comparison of perceived and real shoulder-surfing risks between alphanumeric and graphical passwords. SOUPS'06, pages 56–66, 2006.
22. J. Thorpe and P. C. van Oorschot. Human-seeded attacks and exploiting hot-spots in graphical passwords. In *USENIX Security Symposium*, pages 8:1–8:16, 2007.
23. M. Toomim, T. Kriplean, C. Pörtner, and J. Landay. Utility of human-computer interactions: toward a science of preference measurement. In *Proc. CHI*, 2011.
24. R. Weiss and A. De Luca. Passshapes: Utilizing stroke based authentication to increase password memorability. In *5th Nordic conference on HCI*, 2008.
25. S. Wiedenbeck, J. Waters, J.-C. Birget, A. Brodskiy, and N. Memon. Authentication using graphical passwords: effects of tolerance and image choice. SOUPS'05, pages 1–12, 2005.
26. S. Wiedenbeck, J. Waters, L. Sobrado, and J.-C. Birget. Design and evaluation of a shoulder-surfing resistant graphical password scheme. In *AVI '06*, pages 177–184, 2006.

A Additional User Study Results

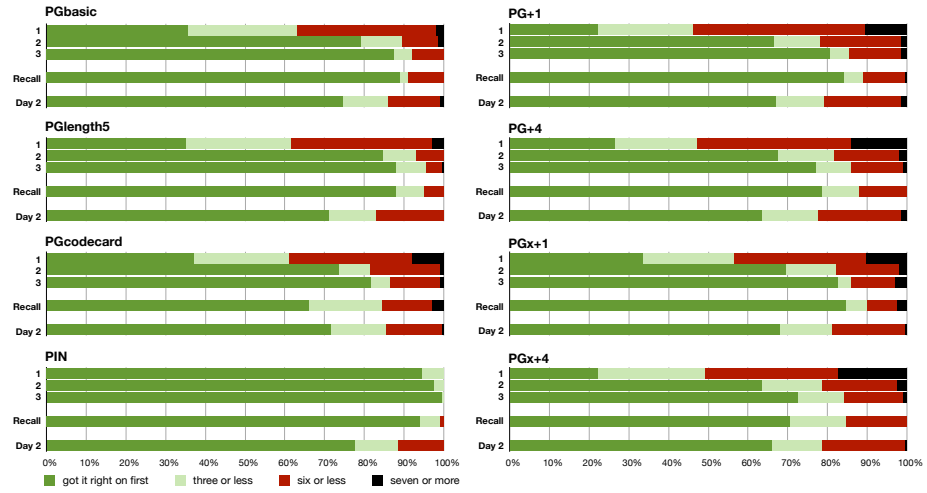

Fig. 6. Percentage of participants who logged in successfully (either on their first attempt or within three), across three practice authentications, day 1 recall, and day 2 entry, by condition.

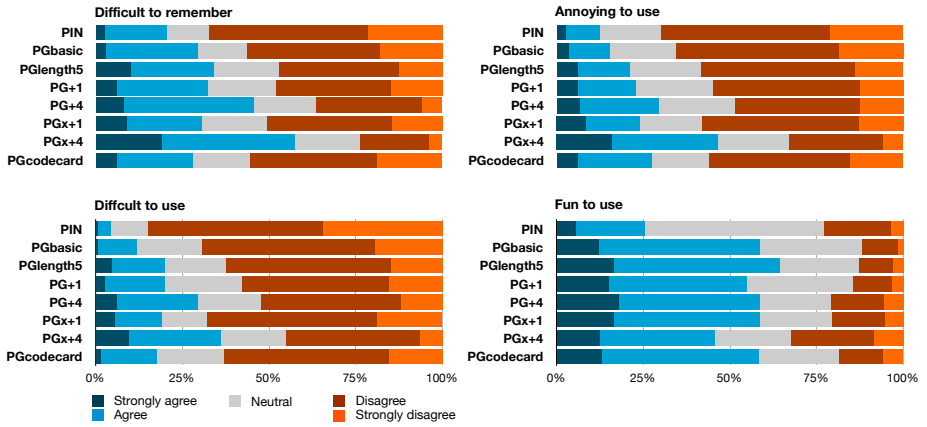

Fig. 7. Likert responses graphed by response, by condition. All participants answered four standard questions on day two about difficulty to learn, difficulty to use, annoyance, and fun to use, for each password system.