

Logging safely in public spaces using color PINs

Frank Nielsen,

Sony Computer Science Laboratories, Inc.

3-14-13 Higashi Gotanda, 141-0022 Shinagawa-ku, Tokyo, Japan

Frank.Nielsen@acm.org

April 25, 2013

Abstract

Nowadays, we are increasingly logging on many different Internet sites to access private data like emails or photos remotely stored in the clouds. This makes us all the more concerned with digital identity theft and passwords being stolen either by key loggers or shoulder-surfing attacks. Quite surprisingly, the current bottleneck of computer security when logging for authentication is the User Interface (UI): How can we enter safely secret passwords when concealed spy cameras or key loggers may be recording the login session? Logging safely requires to design a secure Human Computer Interface (HCI) robust to those attacks. We describe a novel method and system based on entering secret ID passwords by means of associative secret UI passwords that provides zero-knowledge to observers. We demonstrate the principles using a color Personal Identification Numbers (PINs) login system and describes its various extensions.

Keywords: HCI, password, PIN, shoulder-surfing attack, key logger, one time password.

1 Introduction

Nowadays, we are logging more and more to cloud services using Internet terminal in public spaces with the threat of passwords being stolen. Stealing

passwords can be done either by machine key and mouse loggers or by human shoulder-surfing attacks. Purloining passwords often opens the door to identity theft. The threat is even more important as concealed spy or surveillance digital cameras may be recording the login session, and later computer vision techniques may be used to recover the secret password. Wired magazine had in December 2012 its cover page entitled **kill the P@55W0rD** to emphasize on the emergency of rethinking the password protection systems. There is a strong need not only to rethink passwords [1] but also to reconsider the User Interface (UI) to input safely passwords. This is an important Human Computer Interface (HCI) problem that is gaining more and more attention with dedicated conferences like the Symposium On Usable Privacy and Security (SOUPS).

Ideally, we should not need to log in as the machine should recognize the unique “Me” as sophisticated computer HAL 9000 did in Kubrick’s film *2001: A Space Odyssey* (1968). That is, ultimately, it is not the user who shall authenticate oneself but the machine who shall authenticate the user, securely. Although artificial intelligence has made significant progress since that 1960’s movie release, we are yet far from such a robust non-invasive biometric system.¹ Brain machine interfaces for logging have also recently been considered [2] with the hope, one day, of ditching passwords and replace them with “pass-thoughts.” (Section 4 shall review prior work.)

Thus we ask ourselves what would be a *secure UI system* to log in public spaces in front of other people using possibly a machine with unprotected and readable input/output interfaces (I/O)? One solution is to use a One Time Password² (OTP) *physical token device* that is assumed to belong privately to the user. Each time the user logs in, it asks the token for a newly on-demand generated password (*e.g.*, typically by pushing a button on the physical token). However, this device may be inadvertently stolen or, even worse, some OTPs maybe generated by an attacker without stealing the device. The latter case is far more difficult to detect for the user that has assumed to always securely carry the device!

Notice that passwords should never be stored plainly on servers. This has dramatic consequences when servers are cracked, but unfortunately happens quite often as reported in the news. UNIX login program stores *encrypted* passwords either plainly in `/etc/passwd` file or in a privileged-access

¹Current biometric systems include 3D head scan for example.

²http://en.wikipedia.org/wiki/One-time_password

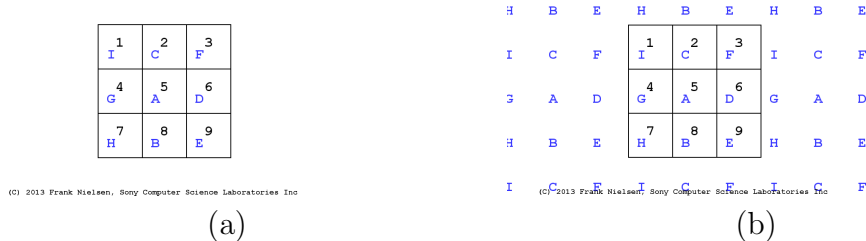


Figure 1: Snapshots of the system: The moveable 3×3 cursor letter board is displayed on top of the fixed digit keyboard. To simulate the 2D torus topology of the cursor keyboard, we display 3×3 translated (Euclidean) copies (b) of the letter cursor board with offset the board dimension, and clip the rendering to the fixed digit keyboard (a).

/etc/shadow file. It uses the user password as the key to cypher iteratively 25 times the 64-bit 0 string using the `crypt()` DES. To strengthen even more the password a 2-character “salt” is generally added [4].

Our proposed system, called *Color PINs*, is based on using an associate UI password for entering safely a secret ID password. The system only requires to have a secure login software (*e.g.*, signed with a cryptographic key for avoiding Trojan attacks) and a random generator.

The paper is organized as follows: Section 2 describes the UI login system and explain its fundamental principles. It is followed by Section 3 that presents some of its extensions. Finally, Section 4 reviews prior work and discusses on several perspectives.

2 The color PIN system

We propose a simple technique to login using a public login name based on *two associated passwords* instead of the traditional single secret password scheme. Since it is easy to record insidiously keyboard key strokes, our logging system uses a graphics keyboard and a cursor board User Interface (UI).

2.1 Description of the basic system

Once a user enters his/her login name, the system receives a *UI password*. The UI password can be defined and kept secret by the user beforehand, or generated on-line using the user profile as described later in Section 3.3. On the graphical screen, we display a fixed *key board* and allows interaction using a second moveable *cursor board* of identical dimension. This is a *two-layered* display. For example, the fixed keyboard can be a 3×3 array³ of digits labeled from 1 to 9, and the cursor board can be a 3×3 array of letters labeled from *A* to *H*. Let $n = 9$ denote the number of keys. The moveable cursor board is controlled with a mouse or touch panel finger interaction. To enter his/her secret password, the user needs to align (a task performed in his/her brain) the cursor key with the *corresponding* digit key and confirm that input using a mouse click or double finger tap event, for example. The mouse cursor of the operating system is preferably hidden, but this is not a strict requirement. The moveable cursor array, which is controlled by interaction, overlays on the fixed digit keyboard and warps on the fixed board edges to implement the 2D torus topology. This is depicted in Figure 1. Furthermore, to scroll endlessly with the 2D torus topology, we implement screen edge warping of the OS mouse cursor. Notice that in practice, it is enough to display 3×3 translated copies of the moveable cursor board with offsets the dimension of the board, as depicted in Figure 1. There is no way for an observing intruder to detect *which* key is aligned with which cursor as all pairs are visually matching. Only the user and the logging system know which key digit-letter pair shall be selected when input validation is triggered. Thus an observer needs to memorize the n potential pairs at each step if he/she wants to break the password. For passwords of length k , this ends up with an exponential number, n^k combinations, of potential passwords to test to crack the user secret password. Each time the user enters a key, an asterisk is displayed to notify the event and both the fixed key and the moveable cursor boards are randomly shuffled, thus yielding no correlation for the next digit-letter alignment task. Users can reset or validate the last entered key using either mouse middle or right clicks, or by touching the screen at virtual reset/validate button locations.

To further be resistant to mouse logger that could record and replay later the

³For sake of simplicity, we choose 2D arrays in this presentation as we are usually accustomed with this numeric pad layout. However, the array can also be 1D (linear), or even 3D (volumetric) if one wishes.

F 1	D 4	E 7
I 2	G 5	H 8
C 3	A 6	B 9

3141
CAHB

F 7	G 9	C 6
H 4	E 3	D 8
A 1	I 2	B 5

3141
CAHB

4 H	9 B	3 I
5 F	6 A	2 E
7 C	1 G	8 D

3141
CAHB

4 F	9 A	3 E
5 C	6 G	2 D
7 H	1 B	8 I

3141
CAHB

Figure 2: Login session for the associative password system using a 3×3 fixed digit board with a 3×3 moveable letter cursor. The user enters sequentially the password 3141 using the UI password *CAHB*. After each digit input is confirmed by a mouse click, both boards are randomly shuffled.

mouse events, we choose the origin of the moveable cursor pointer randomly inside the fixed board. This origin is used when displaying the UI controlled moveable cursor.

Notice that instead of displaying the full moveable cursor board with n keys on top of the fixed graphics board, it is enough to display at least $l \geq 2$ of them including the correct associative key in order to get an exponential number, l^k , of potential combinations.

Figure 2 depicts the screen captures of a session for a fixed digit-moveable 3×3 letter board implementation of the system: The user enters the ID digit password 3141 using the associated UI letter password CAHB. Note that after each digit is entered by mouse clicking both the digit and letter boards are shuffled randomly.

Figure 3 shows a similar system using a fixed board with a moveable color cursor board. The layout is 2×5 with the following moveable color cursor board:

BLACK	ORANGE	LIGHTGRAY	RED	BLUE
GREEN	PURPLE	AQUA	OLIVE	GRAY

The digits are colored with their respective cursor color currently covering the digit key. The secret password is 31413 and that password should be entered with the following color cursor combination:

Color cursor (UI PWD)	Digit cursor (ID PWD)
LIGHTGRAY	3
GREEN	1
RED	4
ORANGE	1
LIGHTGRAY	3

Using the same secret ID and UI passwords at several login sessions weakens the security. Indeed, if no shuffling of the digit and/or cursor board is done, then recording, say, at the first position of the '1' digit the color sequence will provide the password information to observers. When shuffling (at least one of the two boards), we make different permutations of digits-color cursors. However, video recording several login sessions and analyzing the common matching digit-color pairs will provide information on the passwords. Thus

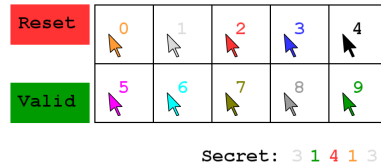


Figure 3: The Color PIN system using a 2×5 color layout. The digits are colored with the corresponding matching cursor color.

we should ideally change the UI password after at each session. This can be done by using a OTP token device for generating UI password keys. The user then keeps his/her secret password and generate a UI password when logging. The advantage of the color PIN system over the traditional OTP token device is that even if the physical token is stolen then it does not allow to properly log in as the ID password is not compromised.

2.2 Implementation details

The color PIN system has been implemented using the `Processing` language (`processing.org`), version 2.0b8. Wrapping the cursor operating system on the screen edges (for endless smooth toric motion) is done using by using JavaTM `Robot` class and `mouseMove` function.

3 Some extensions of the color PINs system

We present several extensions of the associative PIN code system.

3.1 Using legacy password systems

We can use legacy systems based on a single PIN by decomposing it into two parts. For sake of simplicity, consider the pin size of even length $2k$. Then the first k digits of the PIN are used for the traditional ID password and the last k digits are used for the UI password with the mapping $1 \rightarrow A, \dots, 9 \rightarrow H$ when considering digit-letter boards, or $1 \rightarrow \text{BLACK}, \dots, 9 \rightarrow \text{GRAY}$ when considering digit-color boards. Thus we can enter a password of length $2k$ on legacy system using only k association pairs. For example, the PIN

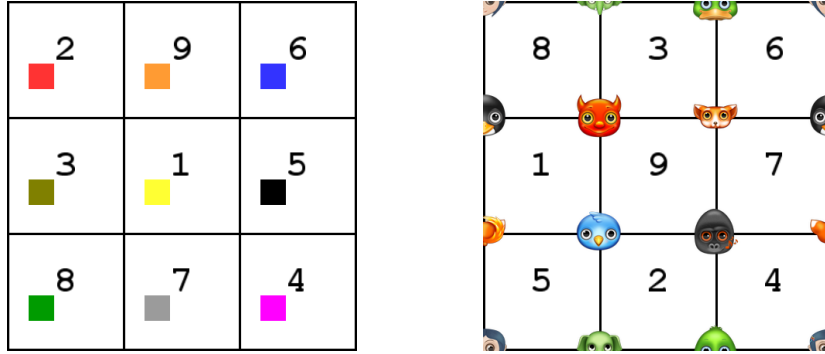


Figure 4: Snapshots of the login system using different rendering styles for the moveable cursor board: (left) color boxes and (right) icons.

1234 is entered using C and D cursors aligned to keys 1 and 2, respectively (or using LIGHTGRAY and RED color cursors). Note that by doing so, we weaken the security as we have n^k potential combinations instead of n^{2k} in a login session. In general, we can create a secure password including the UI password by juxtaposing the two parts: $\boxed{\text{ID PIN}} + \boxed{\text{UI PIN}}$. The UI PIN may be shorter than the PIN. For example, we can use letter 'A' cursor to enter the full PIN. When the UI PIN is shorter than the ID PIN (e.g., 1234AB), we choose the current cursor using the *modulo* operator (here, '1' and '3' should be entered using 'A' and '3' and '4' using 'B').

3.2 Graphics board skins

To make the system more general, we just need to define two boards of n elements: the fixed board elements F_1, \dots, F_n and the moveable cursor board elements M_1, \dots, M_n . Those elements are numbered with function $N(\cdot)$ universally using numbers 1, ..., n : $N(F_i) = N(M_i)$. This allows for various rendering styles. For example, we can use vacation images[3] for the fixed board and numbers for the cursor board, etc. We can use icon sets or color box sets as two different rendering styles for the moveable cursor array as depicted in Figure 4.

Instead of using 2D graphics boards, 1D layouts of fixed and moveable boards

may be preferred for, say, personal picture browsing, and require less graphics space to display for a login session. Thus this kind of ribbon user interface maybe appropriate to display at the bottom of other software when logging is required. We may also add *visual effects*: For example, we may have a color PIN to memorize, and have the color cursor rolling overs the digits. Each time the color is on the digit key, all digits are colored with their corresponding colors (as shown in Figure 3). Indeed, Human tend to memorize visual objects rather easily compared to juxtaposed ID-UI associated PINs. The moveable cursor may be textured (like bark, grass, or street road textures) and the digits textured with the corresponding highlighted texture cursor.

3.3 UI password generation based on user profile

To help user memorize the UI password and generate many UI passwords, we define a *user profile* by asking a set of k questions, each with n choices like what is her favorite food, favorite place, favorite color, favorite celebrity, favorite movie, favorite music, etc. Each time the user enters a key, the moveable cursor skin changes to the next mode (e.g., food→place→color→celebrity→movie→music, etc.). Furthermore, for k -length passwords, we generate a random permutation on the question orders (yielding $k!$ UI passwords) so that the user knows which moveable cursor shall be used to enter the current ID PIN digit. This profiling frees users from memorizing UI passwords but may be less secure when observers know or guess his/her preferences.

3.4 Cursor control using another device

The moveable cursor board may be controlled using events sent from another device. For example, a tablet or a game pad with a touch screen may be used to send event relative displacement orders to the login application. The orders can be absolute coordinates (rescaled to fit the display screen) or relative motion orders encoding a translation. Note that in practice, it is interesting to have a board display of relatively small size while providing a large UI pad for precise interaction. This makes the input more precise for users, say, on smart phones equipped with touch screens, where users can touch anywhere on the screen for controlling the moveable cursor array on the fixed board.

4 Prior work and discussion

We first present prior work related to our color PIN system (see references therein too): The CursorCamouflage [9] system displays a set of *dummy* cursors that makes it difficult for observers to correlate with the user hand motion. However, a careful video analysis of the recorded login session by advanced computer vision techniques may decipher the secret key by analyzing hand-mouse cursor correlations. In comparison, our color PIN system as no dummy cursor but instead use two key boards and two PINs to identify oneself. In [10], the authors present a shoulder-surfing resistant UI for entering password that uses *pass-icons* blended with other icons on a 2D layout; The user is required to pass *several challenges* where each challenge asks to click inside the convex hull of the pass icons. This system (like ours) does not require to click on the pass icons. It is however more time consuming and requires laymen to be familiar with the convex hull definition. In [7], the authors describe *cognitive trapdoor games* where the users has two select on which set the current PIN code digit is contained. After a few selections, the system knows by “intersecting” the challenge subsets which digit was entered, and proceed for entering the next digit, etc. One major drawback is that it provides limited resilience when the session login is video-taped. The closest work to our color PIN system may be found in [8].⁴ The FakeCursor system manages a fixed secret and a disposal secret: He/She enters her pin code by aligning the secret digit on the fixed disposal icons using left/right ATM-like arrow buttons. We can interpret FakeCursor as a discrete UI working on the 1D ring topology. In comparison, our system is fully graphical as it uses 2D torus topology and requires much less user input. Our system has also higher-level abstraction as it requires to define two boards generically that are associated using a universal numbering (see Section 3.2 on skins), or uses a color UI OTP token device for entering the ID password.

The main drawback of the color PIN system is to require memorizing additional information: Namely, the UI password. Another potential threat is by using gaze tracking and advanced computer vision, it might be possible to guess which part were “intentionally” aligned by observing the user’ eyes (a kind of computational mentalism). This risk is minimized by showing a small board size. Interestingly, note that conversely, there are login systems that have been designed based on eye gaze input [5].

⁴<http://www.netaro.info/~zetaka/projects/fakePointer/index.html>

We have presented a login system that considers *zero-knowledge UI* for entering passwords. In perspective, instead of using the (parallel) visual perception system for entering a ID password, we could use tactons [6]: The shuffled digits on the digit boards shall not be visually displayed, but tactilically rendered using tactons, and the user shall press the corresponding tacton corresponding to the secret PIN. This system is particularly well-suited for visually impaired people.

References

- [1] William Cheswick. Rethinking passwords. *Commun. ACM*, 56(2):40–44, February 2013.
- [2] John Chuang, Hamilton Nguyen, Charles Wang, and Benjamin Johnson. I think, therefore i am: Usability and security of authentication using brainwaves. In *Proceedings of the Workshop on Usable Security, USEC '13*, 2013.
- [3] Yutaka Hirakawa, Motohiro Take, and Kazuo Ohzeki. Pass-image authentication method tolerant to random and video-recording attacks. *International Journal of Computer Science & Applications (IJCSA)*, 9(3):20–36, 2012.
- [4] B. Kaliski. Pkcs #5: Password-based cryptography specification version 2.0 (rfc), 2000.
- [5] Manu Kumar, Tal Garfinkel, Dan Boneh, and Terry Winograd. Reducing shoulder-surfing by using gaze-based password entry. In *Proceedings of the 3rd symposium on Usable privacy and security, SOUPS '07*, pages 13–19, New York, NY, USA, 2007. ACM.
- [6] Xuân-Linh Labbé. Touchscreen accessibility - accessible and secure authentication using a haptic PIN. Master’s thesis, University of Glasgow, 2010.
- [7] Volker Roth, Kai Richter, and Rene Freidinger. A PIN-entry method resilient against shoulder surfing. In *Proceedings of the 11th ACM conference on Computer and communications security, CCS '04*, pages 236–245, New York, NY, USA, 2004. ACM.

- [8] Tetsuji Takada. Fakepointer: An authentication scheme for improving security against peeping attacks using video cameras. In *Proceedings of the 2 Second International Conference on Mobile Ubiquitous Computing, Systems, Services and Technologies*, UBICOMM '08, pages 395–400, Washington, DC, USA, 2008. IEEE Computer Society.
- [9] Keita Watanabe, Fumito Higuchi, Masahiko Inami, and Takeo Igarashi. CursorCamouflage: Multiple dummy cursors as a defense against shoulder surfing. In *SIGGRAPH Asia 2012 Emerging Technologies*, SA '12, pages 6:1–6:2, New York, NY, USA, 2012. ACM.
- [10] Susan Wiedenbeck, Jim Waters, Leonardo Sobrado, and Jean-Camille Birget. Design and evaluation of a shoulder-surfing resistant graphical password scheme. In *Proceedings of the working conference on Advanced visual interfaces*, AVI '06, pages 177–184, New York, NY, USA, 2006. ACM.