

SPY-RESISTANT KEYBOARD: MORE SECURE PASSWORD ENTRY ON PUBLIC TOUCH SCREEN DISPLAYS

Desney S. Tan, Pedram Keyani*, Mary Czerwinski
Microsoft Research
One Microsoft Way, Redmond, Washington 98052, USA
desney@microsoft.com, pkeyani@cs.cmu.edu, marycz@microsoft.com

ABSTRACT

Current software interfaces for entering text on touch screen devices mimic existing mechanisms such as keyboard typing or handwriting. These techniques are poor for entering private text such as passwords since they allow observers to decipher what has been typed simply by looking over the typist's shoulder, an activity known as shoulder surfing. In this paper, we outline a general approach for designing security-sensitive onscreen virtual keyboards that allow users to enter private text without revealing it to observers. We present one instantiation, the Spy-Resistant Keyboard, and discuss design decisions leading to the development of this keyboard. We also describe the results of a user study exploring the usability and security of our interface. Results indicate that although users took longer to enter their passwords, using the Spy-Resistant Keyboard rather than a standard soft keyboard resulted in a significant increase in their ability to protect their passwords from a watchful observer.

KEYWORDS: *Touch screen, keyboard, input technique, visual search, selective attention, password.*

1. INTRODUCTION

Touch screens are becoming increasingly common, appearing on devices such as digital whiteboards, tablet PCs, as well as ATM and debit card machines. Many of these devices assume that the touch screen is the primary input mechanism and make using traditional text input mechanisms such as physical keyboards inconvenient. For example, using touch screens on public displays aimed at facilitating ad hoc conversations provides much more fluid interaction than using a physical keyboard. In fact, it is sometimes unfeasible to include a keyboard in such a setup. As a result, many of these devices employ alternative mechanisms for text input, including soft keyboards and handwriting recognition.

The soft keyboard functions like a hardware keyboard except that users touch an onscreen image-map to type (see Figure 1). With handwriting recognition, users enter text by writing on the touch screen. These alternative input interfaces are intrinsically observable. That means that someone watching the typist use these interfaces can fairly easily reconstruct text that has been entered, an activity known as *shoulder surfing*. This is undesirable when typing any private text, but is especially problematic for passwords.

Users typing on these public touch screen displays can take precautions that make it harder for a casual observer to obtain their password. For example, they may physically obscure the display so that observers cannot see the interface feedback or the results of their actions. However, this can be difficult on touch screens that are placed in locations that make blocking it inconvenient or that are larger than the user can physically block with their body. Users may also adopt other strategies, such as quickly adding and

* Currently at Carnegie Mellon University, 5000 Forbes Avenue, Pittsburgh, Pennsylvania 15213, USA



Figure 1. User typing with a soft keyboard on a publicly observable touch screen.

deleting characters that are not in the intended password in order to confuse observers. Unfortunately, our observations show that this strategy usually leads to increased levels of mistyped passwords and does not add much security against alert observers. Additionally, taking intentional actions to protect one's password may be socially awkward as it conveys a lack of trust in the observers.

In our work, we aim to design an onscreen virtual keyboard that increases security against shoulder surfers without requiring typists to take explicit precautions. In fact, we assume that observers can openly watch any part of the typist's interaction with the keyboard. Hence, the keyboard must not only ensure that the typist's actions cannot be easily converted into knowledge of the password, it must also prevent observers who do not explicitly know the password from repeating the typist's actions in order to enter the password, a security violation commonly referred to as a replay attack.

In this paper, we present a novel approach to designing keyboards for entering private text on public touch screen displays. This approach introduces indirection by utilizing an auxiliary mapping that allows typists to focus their attention on a particular part of the keyboard, while observers have to pay attention to and memorize the entire keyboard. We should stress that this approach does well to protect against casual observers, but does not necessarily guard against malicious attackers equipped to record and review the entire interaction, for example with video cameras. We describe one particular instantiation, which we call the Spy-Resistant Keyboard, as well as the main design decisions leading to its development. We present results from a user study evaluating both the usability as well as the additional security offered by the Spy-Resistant Keyboard. Finally, we discuss future work extending these ideas.

2. BACKGROUND

In many computer systems, users have to authenticate themselves to access sensitive data and services. There are two basic components to any user identification scheme: the interaction with which the user implicitly or explicitly provides identifying data to the system; and the ensuing check by the system to ensure that this data matches some prior piece of information it has about the user. There exists a large body of work on the latter, examining possible attacks (Neumann, 1994) as well as how to protect against them as the system either sends the data to be authenticated, or does the comparison locally (for a review, see Halevi & Krawczyk, 1999).

Technical issues, while important, are not the only component of secure authentication. Hitchings (1995) asserts that treating security as a purely technical issue has led to mechanisms that are less effective than they could and should be. Davis and Price (1987) add that since security necessarily involves people, human factors should be carefully considered in designing effective security mechanisms. In our work, we focus on creating interaction techniques for users to securely provide private data in public circumstances.

Currently, users have three basic methods to provide private data and authenticate themselves: tokens, biometrics, and private knowledge such as passwords. Token-based methods utilize something a user possesses, such as an identification card, to verify their identity (Brostoff & Sasse, 2000). Such methods often require costly construction and distribution of tokens, as well as installation of specialized sensing hardware. Additionally, possession of a token does not necessarily imply ownership, and theft or forgery is a serious threat to these systems.

Biometric methods identify individuals based on distinguishing physiological or behavioral characteristics. These methods include signature, keystroke pattern recognition, voice, vein geometry, as well as eye-based, facial, finger, and palm imaging (for a detailed review, see Jain, Hong, & Pankanti, 2000). Just as with token-based methods, biometric methods involve costly hardware, and characteristics can be stolen or forged. Furthermore, since these characteristics cannot be easily replaced, theft is more costly than it is with other methods.

The third class of methods, which remains dominant on many computing systems, verifies access privileges with pieces of knowledge such as passwords known only to the user. Historically, the choice of passwords has been such a prevalent problem that the National Institute of Standards and Technology has published a document advising users of proper password selection and use (National Institute of Standards and Technology, 1995). They recommend picking random strings of characters and keeping different passwords for different accounts. Unfortunately, this places a large cognitive strain on users, who have to remember an increasing number of passwords (Adams, Sasse, & Lunt, 1997).

To alleviate this problem, researchers have proposed alternatives and augmentations to standard text passwords. For example, numerous researchers have proposed graphical passwords, in which users have to select predetermined positions within an image (Blonder, 1996), recognize images such as faces (Brostoff & Sasse, 2000), or sketch drawings recognized by the system (Jermyn, Mayer, Monrose, Reiter, & Rubin, 1999). Others have explored the use of cognitive passwords, which use a question and answer session prompting for personal details or word associations (Zviran & Haga, 1990) known only to the user.

All these knowledge-based methods assume that no one other than the user knows the password. In fact, these methods are only as secure as the user's ability to keep the password secret. However, as Sasse, Rostoff, and Weirich (2001) point out, users are typically the 'weakest link in the security chain', and any truly secure system must be designed with this in mind. This importance of human factors in the design of security systems has been increasingly recognized recently (e.g. Patrick, Long, & Flinn, 2003).

With the introduction of large touch screen displays that utilize onscreen soft keyboards, learning someone's password has become as easy watching them type it in, or shoulder surfing. Even apart from adversarial observers, people are more likely to peek at private content on large public displays, making unintentional viewing of password entry more likely than before (Tan & Czerwinski, 2003). The use of one-time passwords (Haller, Metz, Nesser, & Straw, 1998) is the closest general-purpose method we have found that might protect against such attacks. However, this method usually requires that users learn new passwords, which is typically impractical. Also, it cannot be applied to generic private text entry.

Recently, Roth, Richter, and Freidinger (2004) have identified a similar set of problems that we have with entering passwords on public displays. They devised the cognitive trapdoor game, a pin-entry method that consecutively displays digits to the user in two distinct sets. After multiple user selections, the system determines the intended digit by intersecting the chosen sets, and the algorithm is repeated for each digit. Unfortunately, tests of their system showed that users took about ten times as long to enter simple PINs on a number pad, a usability hit we believe can be improved.

In our work, we present general design principles for creating interaction techniques and interfaces that allow users to enter private text such as passwords on public touch screen displays without the risk of revealing them to casual observers. We then describe the Spy-Resistant Keyboard, an alternate interface derived directly from applying these principles. Finally, we present results from a user study evaluating the effectiveness of this interface.

3. GENERAL APPROACH

Our approach to designing virtual keyboards that add security against shoulder surfing involves breaking the typing interaction into two phases, the mapping phase and the selection phase. It should be noted that this approach does not generally alleviate dangers of attackers with video cameras who are able to record, rewind, and review the entire interaction.

In the mapping phase, the keyboard presents the typist with some method of uniquely mapping each character to a property, which serves as an alternate representation of the character. This property could be anything ranging from another character to a color to the shape of the button to the spatial location of the

button, and so on. While each keyboard should use as simple a mapping as possible, it should always re-randomize the mapping after each character is typed so that observers cannot learn the mapping over time. The typist locates the character they wish to type and mentally notes the specific property that represents this character. They need not focus any attention on properties representing other characters. Once they have done this, they signal to the keyboard that they are ready to move into the selection phase.

In the selection phase, the keyboard removes the mappings, usually by blanking characters from the keyboard. The typist completes their interaction by specifying the property that represents the character they wish to type. They then repeat this process for each character in their password.

3.1. Justification of Approach

This approach provides two mechanisms that make it hard for the observer to derive the character that has been typed simply by watching the actions of the typist. First, since the keyboard is randomized for each character, there is no way for the observer to repeat the typist's actions to reproduce the correct password. Second, the explicit two-phase interaction conceals the typist's intention until pertinent mapping information is hidden. This makes it difficult for the observer to derive the character typed even when they can watch the entire interaction. In fact, the observer has little information to help focus their attention in the mapping phase, and is forced either to guess which characters they should focus on or to memorize mappings for all keys. The former is unreliable in deciphering what has been typed and the latter is difficult without recording equipment.

4. SPY-RESISTANT KEYBOARD

We instantiated this approach in an interface we call the Spy-Resistant Keyboard. This keyboard randomizes the spatial location of all characters as each password character is entered.

The Spy-Resistant Keyboard is composed of 42 Character Tiles, two Interactor Tiles (labeled “Drag Me...”), a feedback textbox, a backspace button, and an enter button (see Figure 2). Each Character Tile is randomly assigned a lowercase letter, an uppercase letter, and either a number or a symbol, all positioned vertically on top of each other. Lowercase letters are always on the top row of each tile and have a red background; uppercase letters are placed in the middle and have a green background; numbers and symbols are positioned on the bottom and have a blue background. Since there are exactly 42 numbers and symbols combined, but only 26 letters, some letters are repeated. Just as each button on a standard keyboard represents two characters, depending on the state of the caps lock or shift keys, each Character Tile represents three characters, depending on the state of shifting. Rather than having a fixed shift state for the entire keyboard, as traditionally done, each tile has a randomly assigned shift state, indicated by the red line under the active character.

In order to select a character on the Spy-Resistant Keyboard, the typist first locates the tile that contains the character to be typed. They remember the mapping by noting the location of this tile. Next, the typist



Figure 2. In the first phase of typing, the mapping phase, the user first finds the character they would like to type and notes its location. For example, if they are trying to type the capital letter “Z”, they would scan the three green rows, finding the letter on the seventh tile on the second row.

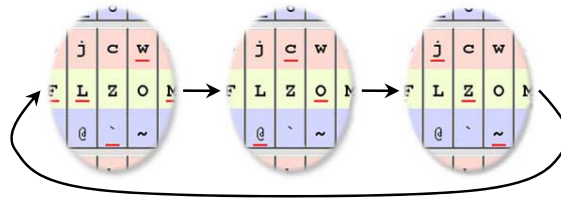


Figure 3. Each Tile on the keyboard begins with a random shift state, indicated by the red underline. Hitting the Interactor moves the shift state on all Tiles. In our example, the user taps the Interactor to cycle through and get the underlining under the letter “Z”.

clicks on one of the Interactors at the bottom of the keyboard to cycle through shift states and move the red underline to the desired character (see Figure 3). Clicking on the Interactor moves the underline to the next character on each tile. Note that since the underlines start on different types of characters on each tile, knowing that the typist has clicked on the Interactor but not knowing which tile to focus on gives the observer no useful information about the kind of character being typed.

Finally, the typist drags the Interactor towards the Character Tile on which the desired character resides. Upon the start of the drag interaction, the system knows that the user has located the character and moves into the selection phase, blanking all Character Tiles (see Figure 4). Hence, without knowing where the Typist is going to drop the Interactor, adversarial observers have to memorize the location of all characters on the keyboard so that they can reconstruct the typed character from the location of the drop. We anticipated that this would be a difficult, if not impossible. Each tile highlights as the typist drags over it. The typist drops the Interactor on the desired tile and the character is entered. The keyboard re-randomizes characters and the typist repeats the process to select the next character. After beginning the drag, the typist may also drop the Interactor on anything other than a Character Tile to reset the board and get a new set of characters, in case they lose track of their target.

4.1. Design Rationale

The Spy-Resistant Keyboard is a product of iterative testing and design. Specifically, it has evolved in three regards: the basic interaction mechanism, the design and layout of tiles, and the mechanism for specifying the shift state.

4.1.1. Basic Interaction Mechanism

We had initially used a tapping gesture for the interaction mechanism. In early prototypes, typists first tapped the Interactor to indicate that they had located the desired character and were ready for the tiles to be blanked. They then tapped on the appropriate tile to type that character. Most typists quickly realized that they were much faster using one hand to tap the Interactor and the other to tap the tile. Unfortunately, most typists would anticipate their action by positioning their typing hand directly over the tile to be typed even before tiles were blanked. This gave observers the opportunity to look at the character before it was blanked, negating any benefit of using the interface. Rather than having typists consciously sequence their

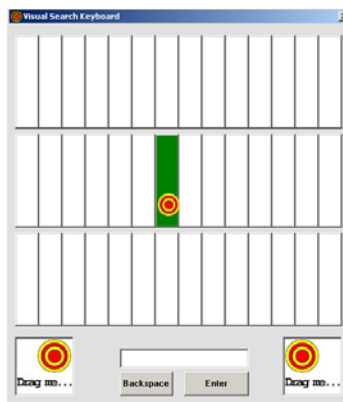


Figure 4. To complete the selection, the user drags the Interactor to the appropriate Tile. It is hard for an observer to reconstruct what has been typed (a “Z”).

actions, we decided to build this requirement into the interaction technique. The current drag-and-drop mechanism provides an intuitive interaction that forces users to perform the two phases one at a time, first blanking the tiles by starting to drag the Interactor and then selecting the desired character by dropping it.

4.1.2. Design and Layout of Tiles and Interactors

The layout of Character Tiles significantly affects how quickly users are able visually search the keyboard and find specific characters. In early prototypes, tiles were designed with the set of three characters running across each tile. This meant that similar characters (i.e. lowercase, uppercase, as well as numerals and symbols) were grouped in vertical columns. Performing the visual search with this design required typists to scan across all tiles, stopping to look at every third character. Alternatively, the typist could scan the characters from top to bottom, looking only at every third column. Unfortunately, early observations showed that this was an extremely difficult task, even for experienced users.

Since we are more adept at scanning contiguous blocks of text running from left to right, grouping letters horizontally by stacking the three characters on each tile made the search task easier. Also, color coding each type of character aided in identifying the appropriate rows to scan. These adjustments are consistent with suggestions by Wickens and Hollands (2000) for directing attention in display space. Furthermore, we explored several different configurations of these tiles, for example a single row of 42 columns or 2 rows of 21 columns. We settled upon the current layout of 3 rows of 14 columns, as it provides a relatively small number of rows but maintains an aspect ratio that fits nicely on most displays.

Another layout decision we made was to include two Interactors, one on either side of the keyboard. Although this was initially done to make the interaction comfortable for both right and left-handed users, we found that both groups seem to use both Interactors interchangeably, depending on where they are standing in relation to the interface and where the target character lies. We assert that this is a useful design element to include in large, touch screen displays such as the SmartBoard™.

4.1.3. Specifying Shift State

Finally, we had initially used one tile to hold each character, eliminating the need for a shift key. However, since we wanted to separate the different character types to help users constrain their visual search, knowing the layout of the keyboard as well as where a typist dropped the Interactor would necessarily reveal the kind of character entered. While this information does not reveal the entire password, it drastically reduces the search space of all possible passwords. We could have also used a single shift state for the entire keyboard, much as traditional soft keyboards do. However, since this state would have to be visible, knowing that the typist had hit the Interactor to change the shift state would also reveal the kind of character typed. Hence, we decided to randomize the shift state for each character. This adds little additional load on the typist as they still only have to focus on one particular tile and its shift state, but adds significant complexity for the observer who has to monitor all possible shift states in addition to all possible characters.

5. USER STUDY

We compared the Spy-Resistant Keyboard to a standard soft keyboard in order to examine usability as well as additional security it provides. To ensure equivalent visibility by observers, we used the same font, 16-point Courier bold type, for all characters in each interface.

5.1. Participants and Setup

Six pairs of Microsoft employees (8 males, 4 females) volunteered to participate in the study. All users had normal or corrected-to-normal eyesight, and all were right-handed. The average age of users was 28.8, ranging from 21 to 38 years old. Users received a small gratuity for participating.

We ran the study on a SMART Board™ 3000i, which provides a physically large rear-projected touch screen display. The display was approximately 53" (~134.6 cm) tall by 40" (~101.6 cm) wide and ran at a resolution of 1024 x 768. Users stood in front of the display and interacted with the interfaces by touching the display with their fingers (see Figure 1).

5.2. Task and Procedure

Before beginning the test, we gave users paper-based instructions on how to type with the soft keyboard as well as with the Spy-Resistant Keyboard. Both users took turns practicing each interface by typing in a password we provided. All users completed each practice password in less than two and a half minutes.

For each trial in the test, one user played the role of Typist while the other was the Observer. The Typist used one of the two interfaces to type in passwords. The Observer watched the Typist to decipher the passwords. Typists were allowed to use any technique they wished to prevent the Observer from figuring out the password. However, in order to simulate public visibility of the display, they were not allowed to explicitly physically obstruct the Observer's view of the keyboard. Observers were also allowed to use any technique they wished to watch the Typist and figure out the password. For example, they could move around to get the best view of the screen and many took notes to help them reconstruct the passwords. After each entry, the Observer recorded what they thought the password was. The pair performed each entry twice for each password.

5.3. Design

We assigned each Typist one easy, moderate, and difficult password for each interface. All passwords were 8 characters long. We randomly chose the easy passwords from the set of English words having Kucera-Francis (1967) familiarity and concreteness ratings between 300 and 700 (e.g., contract). Moderate passwords contained 3 to 5 letter English words surrounded by random characters (e.g., #back\$Jr). The difficult passwords were completely random sequences of 8 characters (e.g., s%g7^Lp=).

We used a 2 (Interface: Soft Keyboard vs. Spy-Resistant Keyboard) x 3 (Password: Easy vs. Moderate vs. Difficult) within-subjects, dyadic design with repeated measures. Each user performed each of the 6 conditions twice, once as the Typist and once as the Observer. We balanced the order of Interface across pairs, with each member using the interfaces in the same order, and randomized the order of Password.

We collected the following dependent measures from the Typist in order to compare usability of the two interfaces: completion time, number of backspaces, and error rates for each password entry. In order to determine the level of security provided by the interfaces against watchful observers, we collected the Observer's guesses from each password entry. Finally, users filled out a post-test questionnaire indicating their preference for each of the interfaces.

5.4. Results

5.4.1. Usability Evaluation: Typist Performance

We analyzed the average completion time required to enter each password with a 2 (Interface: Virtual Keyboard vs. Spy-Resistant Keyboard) x 3 (Password: Easy vs. Moderate vs. Difficult) repeated measures analysis of variance (RM-ANOVA). We found a significant main effect of Interface ($F(1,11)=114.11$, $p<0.0001$), with the Soft Keyboard resulting in faster completion times on average (see Figure 5). It should be noted that the two time slowdown with our interface over traditional typing methods was a significant improvement over prior attempts (e.g. Roth et al., 2004).

We found no significant difference in the number of backspaces hit for each of the conditions. In fact, Typists seemed to hardly ever use the backspace key (average of about 1 backspace every 20 passwords typed). We also found no significant differences in the error rate for entering passwords. In fact, only 9 out of a total of 144 passwords were entered incorrectly, and most were off by a single character.

5.4.2. Security Evaluation: Observer Performance

We compared each guess made by the Observer to the password typed and generated two metrics representing the level of security offered by the interface: a strict metric, the number of characters in each guess that did not match its typed counterpart exactly; and a loose metric, the Levenshtein (1966) distance, or number of deletions, insertions, and substitutions required to transform the guess into the typed password. This loose metric accounted for characters that were shifted in position (Jermyn et al., 1999). Both these metrics produced ratings on a scale of 0 to 8, with 0 indicating poor level of security and 8 indicating strong level of security offered by the interface.

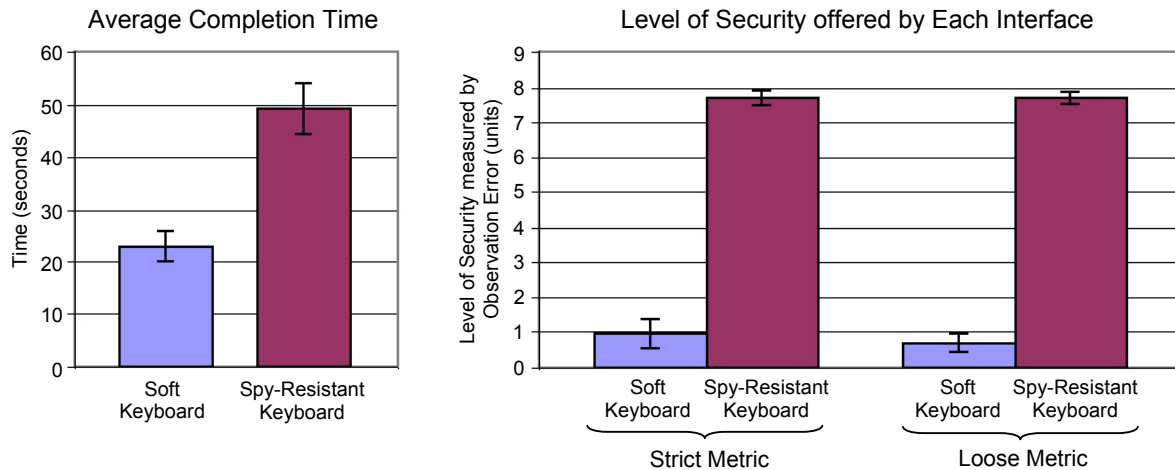


Figure 5. (left) Main effect of Interface for average time to type each password. (right) Main effects of Interface for the level of security, measured by errors in guessing the password. Error bars represent standard error.

We performed similar 2 x 3 RM-ANOVAs for the level of security offered by the interfaces. This analysis revealed a significant main effect of Interface for both the strict metric ($F(1,11)=641.47, p<0.0001$) as well as the loose one ($F(1,11)=1250.68, p<0.0001$), with the Spy-Resistant Keyboard resulting in far stronger security, on average. In fact, although most observers were able to fairly accurately guess entire passwords with little error on the Soft Keyboard, they were not able to get even one of the eight characters correctly with the Spy-Resistant Keyboard. Additionally, the loose metric revealed a significant main effect of Password ($F(1,11)=7.31, p=0.004$), with progressively higher security using the more difficult passwords (3.85 vs. 4.31 vs. 4.44, on average). These results, illustrated in Figure 5, indicate the drastically improved level of security offered by the Spy-Resistant Keyboard against shoulder surfers.

5.4.3. Subjective Ratings

In addition to performance data, we gathered user preference data on 5-point Likert scales after the study. Users found the Soft Keyboard ($M=4.92$) significantly easier to use than the Spy-Resistant Keyboard ($M=2.42$), ($t(11)=16.58, p<0.0001$). However, users indicated that they were also significantly less comfortable with using the Soft Keyboard to enter their passwords ($t(11)=-13.01, p<0.0001, M=1.17$ vs. $M=4.50$). This sentiment was further supported by users feeling like they had much more difficulty acquiring useful information when observing someone using the Spy-Resistant Keyboard ($M=4.50$) as opposed to the Soft Keyboard ($M=1.67$), ($t(11)=-10.47, p<0.0001$). Additionally, most users agreed that the extra security was worth the extra effort, especially since most passwords are relatively short. This is important since it has been shown that security measures that are not compatible with user perceptions often end up being circumvented, thereby undermining system security (Adams et al., 1997).

6. DISCUSSION AND FUTURE WORK

Study results suggest that the Spy-Resistant Keyboard imposes a trade-off between efficiency of entering text and the security of text entered against observers. Using the Spy-Resistant Keyboard takes about twice as long as a soft keyboard, but distinctly makes the perceived as well as actual level of security provided against one-time casual observers stronger. This validates the assertion that our general approach to designing such text entry systems may yield interesting interfaces that improve upon prior attempts.

In future work, we will explore schemes to make the selection task easier, while maintaining similar levels of security against observers. One improvement would be to make remembering the location of a tile easier by providing landmarks within the keyboard. These landmarks could simply be spaces in between sets of tiles, or they could be more complex background images or tiles of different shapes. While this would make selection of the character easier, this would not significantly speed up the visual search task.

One promising alternative is to completely eliminate the visual search task by not randomizing the characters on the keyboard later in the process. Initially, we present a standard keyboard. However, when

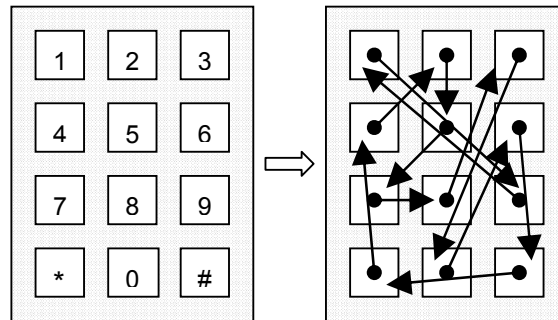


Figure 6. (left) We can eliminate the visual search task by laying out the keys as they normally would be. (right) The mapping is performed by blanking the keys and animating them into new positions. After mapping, the sequence of keys from left to right, top to bottom, is 9,4,8,*,2,0,5,7,1,#,3,6. To type a character, the user specifies its new position.

the user starts the drag, we hide the characters and animate each key into a new position, thus providing a one-to-one mapping of a character to a new spatial location (see Figure 6). With this mapping, the typist has to watch the changing location of one key, but the observer has to know where all keys started and ended in order to later reconstruct what has been typed. Additionally, this interface might not need an explicit shift state because of the way randomization happens after the user has found the character. In fact, this allows all characters to be displayed at once and grouped in whatever manner is most convenient.

We found in the study that observers who devised strategies either tried to monitor the typist's gaze or concentrated on only a small region of the display hoping that the desired character lay there. Although these may be more effective than other strategies, it would still take many observations before gaining access to the full password. In future work, we will explore schemes that provide feedback on the remaining safe lifetime of a password based on the number of times it has been entered and the types of interfaces used. In such a scheme, a password's level of safety would decay much more rapidly when entered on a public touch screen keyboard than on a private desktop machine in the user's personal office, and users would be warned accordingly.

Finally, we must stress that the Spy-Resistant Keyboard does not do well to protect against observation that may be rewound and replayed, for example from an observer with a video camera. We are currently exploring techniques that protect against this kind of attack.

7. CONCLUSIONS

In this paper, we have presented a novel approach for designing virtual keyboards that protect typists from revealing private text to watchful observers. By breaking the typing interaction into two distinct phases, we provide a level of indirection which allows the typist to focus on a specific part of the keyboard while the observer has to memorize the entire keyboard to reconstruct the character typed. We described the Spy-Resistant Keyboard, one instantiation of such a keyboard, as well as important design decisions we made in building this interface. We ran a study showing that the Spy-Resistant Keyboard provides a significantly higher level of security over a traditional soft keyboard at the cost of text entry time. However, the text entry time for this interface was significantly shorter than prior attempts aimed at solving similar problems. The study also showed that users thought the extra level of security was worth the additional effort. Finally, we presented several ideas that could potentially speed up typing on such a keyboard while maintaining similarly high levels of security.

8. REFERENCES

- Adams, A., Sasse, M.A., Lunt, P. (1997). Making passwords secure and usable. *Proceedings of HCI on People and Computers XII*, 1-19.
- Blonder, G. (1996). Graphical passwords. *United States Patent 5559961*.

- Brostoff, S., Sasse, M.A. (2000). Are Passfaces more usable than passwords? A field trial investigation. *Proceedings of HCI on People and Computers XIV*, 405-424.
- Davis, D., Price, W. (1987). *Security for Computer Networks*. Wiley: Chichester, UK.
- Halevi, S., Krawczyk, H. (1999). Public-key cryptography and password protocols. *ACM Transactions on Information and System Security*, 2(3), 230-168.
- Haller, N., Metz, C., Nesser, P., Straw M. (1998). A one-time password system. *Network Working Group Request for Comments* 2289.
- Hitchings, J. (1995). Deficiencies of the traditional approach to information security and the requirements for a new methodology. *Computers and Security*, 14, 377-383.
- Jain, A., Hong, L., Pankanti, S. (2000). Biometric identification. *Communications of the ACM*, 43(2), 90-98.
- Jermyn, I., Mayer, A., Monroe, F., Reiter, M.K., Rubin, A. (1999). The design and analysis of graphical passwords. *Proceedings of the 8th USENIX Security Symposium*.
- Kucera, H., Francis, W.N. (1967). *Computational analysis of present-day American English*. Brown University Press: Providence, RI.
- Levenshtein, V. (1966). Binary codes capable of correcting deletions, insertions and reversals. *Soviet Physics Doklady*, 10, 707-710.
- National Institute of Standards and Technology. (1995). Password Usage. *NIST FIPS PUB 112*.
- Neumann, P.G. (1994). Risks of passwords. *Communications of the ACM*, 37(4), 126.
- Patrick, A., Long, C., et al. (2003). *Workshop on human-computer interaction and security systems at CHI Conference on Human Factors and Computing Systems*.
- Roth, V., Richter, K., Freidinger, R. (2004). A PIN-entry method resilient against shoulder surfing. *Proceedings of 11th Conference on Computer and Communications Security*, 236-245.
- Sasse, M.A., Brostoff, S., Weirich, D. (2001). Transforming the 'weakest link' – a human/computer interaction approach to usable and effective security. *BT Technology Journal*, 19(3), 122-131.
- Tan, D.S., Czerwinski, M. (2003). Information voyeurism: Social impact of physically large displays on information privacy. *Proceedings of CHI Conference on Human Factors and Computing Systems*, 748-749.
- Wickens, C.D., Hollands, J. (2000). *Engineering Psychology and Human Performance*, 3rd edition. Prentice Hall: New Jersey, 69-118.
- Whitten, A., Tygar, J.D. (1999). Why Johnny can't encrypt: A usability evaluation of PGP 5.0. *Proceedings of the 9th USENIX Security Symposium*.
- Zviran, M., Haga, W.J. (1990). Cognitive passwords: The key to easy access control. *Computers and Security*, 9(8), 723-736.

9. ACKNOWLEDGEMENTS

We thank Patrick Baudisch, George Robertson, Brian Meyers, Greg Smith, Sumit Basu, Darko Kirovski, Jeffrey Nichols, and Tara Whalen for their insightful comments and discussion of this work.