CursorCamouflage: Multiple Dummy Cursors as A Defense against Shoulder Surfing

Keita Watanabe[*1], Fumito Higuchi[*1], Masahiko Inami[*1], Takeo Igarashi[*1]
JST ERATO Igarashi Design Interface Project[*1]
watanabe@designinterface.jp, inami@kmd.keio.ac.jp, takeo@acm.org

1. Introduction

More and more services and information are being stored on the cloud. Since anybody can access an Internet terminal, it is critical to provide appropriate security mechanisms. One popular approach is to strengthen the protocol and encryption algorithm, which is now being actively investigated in the security field. Another potentially effective approach is to enhance the user interface for security systems. Since security is ultimately a human-computer interaction problem, we believe that there are many interesting opportunities related to the latter approach.

In this paper, we present an example of applying an innovative user interface method to enhance security. Our target problem domain is shoulder surfing when an individual is typing a password or personal identification number (PIN) using a software keyboard and an indirect input device such as a mouse or track pad. Such typed key sequences are readily visible to potential attackers standing behind the user or observing the screen via video camera. A method to defend against shoulder surfing is clearly important. One of the conventional methods is to change the key assignment each time the keyboard appears on the screen and to reveal the assignment only at the beginning. However, this method does not work if a video camera is recording the screen. Several other methods have been proposed [9, 13, 15], but they are all either too complicated or require the user to memorize extra information in addition to the password itself.

Our method, called Cursor Camouflage, shows multiple independently moving dummy cursors on the screen so as to make it difficult for an attacker to identify which software key the user is actually typing (Figure 1). The user can identify the real cursor by observing the correlation between the hand motion and the cursor motion, but it is difficult for an attacker to do so because the correlation is not easy to observe. This method has a certain resistance to video recording and does not require the user to memorize any additional information.

2. CursorCamouflage

The key function of the proposed method is to make it difficult for a potential attacker to identify which key is being typed when the user enters a password using a software keyboard. We assume that the user is using an indirect input device such as a mouse or track pad; our method cannot be used with direct input devices such as a touch screen or video tablet. We also assume that an attacker can only access visible information on the screen and cannot directly steal electronic data from inside the system. Our method is most effective when the attacker cannot see the user's hand motion, but it still works well in cases when the hand motion is visible, as we show in the evaluation. Similarly, our method works best when the attacker is directly observing the screen in real time; still, it does show a certain resilience when the screen is video recorded.

The Cursor Camouflage method shows multiple independently moving dummy cursors on the screen in addition to the standard real cursor. This makes it difficult for the attacker to identify which is the real cursor, thus making it difficult to identify which key is being typed. The user, in contrast, can identify the real cursor by observing the correlation between the hand motion and the cursor motion. For example, if a user moves the input device to the left, he or she simply needs to identify which cursor is moving to the left on the screen in order to identify the real cursor.

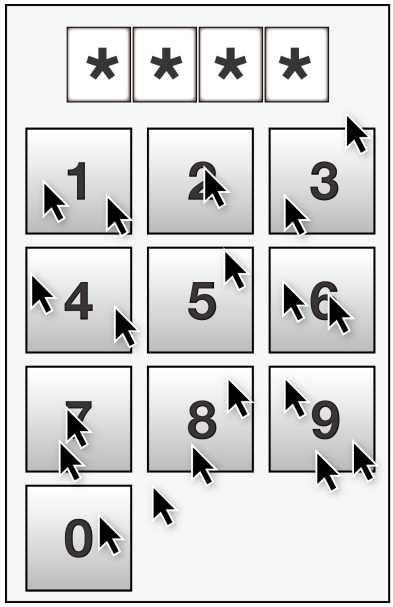

Figure 1. Cursor Camouflage in action. The system shows multiple dummy cursors to protect the real cursor from shoulder surfing.

2.1. Designing Dummy Cursor Motions

The motion design of the dummy cursors is a crucial step because the motion of the dummy cursors should not be immediately distinguishable from that of the real cursor. One possibility is to implement an automatic algorithm that synthesizes random mouse cursor motions, but we decided to record mouse cursor movement operated by a human designer because it is easier to implement and can generate more human-like motions. The system starts playing all the recorded dummy cursor motions at the beginning of the password entry session. When the recorded motion of a dummy cursor comes to an end, the system plays it backwards. The length of recorded motion is different for each dummy cursor. We implemented a system that records the motion of the real cursor on the same software keyboard to be used in the password entry. There was no dummy cursor on the screen during recording. The recorded motions were then used as the motion of the dummy cursors for password entry by the user.

Our experience designing the dummy cursor movements led us to isolate several practical guidelines. They are briefly summarized below.

1. The screen layout on which the dummy cursor movements are designed should be equivalent to the screen layout on which the user types a password using the cursor camouflage system. This ensures that the motion of the dummy cursor is similar to that of the real cursor.

SIGGRAPH Asia 2012, Singapore, November 28 – December 1, 2012.
ISBN 978-1-4503-1757-3/12/0011

2. All cursors halt temporarily when the user types a key (see the next subsection), so if a cursor is on an invalid region, it is immediately clear that the cursor is not a real cursor. Therefore, a dummy cursor should not stay long on invalid regions (gaps between software keys).
3. In the preliminary study, we identified several strategies to identify the real cursor from among many dummy cursors. The dummy cursor movement should mimic these strategies so as not to make the motion of the real cursor distinguishable from the dummy cursors. See the Preliminary Study section for details.

2.2 Other Design Considerations

When the user clicks the mouse to type a software key, the real cursor inevitably stops on the software key while the dummy cursors are continuously moving. This temporary halt of a specific cursor is clearly visible to the attacker, who can then easily identify which key is being typed as well as which cursor is the real cursor. We therefore stop all the dummy cursors when the real cursor stops. Another possible problem is that it is easy to identify the real cursor if it is the only cursor on a valid key when the key is typed. We address this problem by leaving less invalid space (gaps between keys) on the screen and using a sufficient number of cursors to make sure that multiple cursors are on valid keys at all times.

Another challenge is the screen boundary. A mouse cursor is usually blocked at the screen boundary, and users often use it to identify the real cursor. For example, the user can easily identify the real cursor by moving the input device far to the left—the cursor hitting the left screen boundary is thus identified as the real cursor. However, this also helps the attacker to identify the real cursor. We therefore implemented a torus desktop [4] connecting the left and right sides of the screen so that a mouse cursor moving into the left screen border appear from the right screen border. The top and bottom boundaries are connected as well.

Implementation Details

We implemented a prototype system and tested it on a 13-inch MacBook Air with a 1440 × 900 screen resolution (60 Hz) running Windows 7. The application was implemented using Microsoft Visual Studio C#. We used a Logicool Wireless mouse C905 on a silicon mouse pad. The speed of the mouse cursor was set to the middle of the given range.

We used a numerical keypad as the software keyboard. Each key was 120 × 120 pixels, and the gaps between keys were 30 pixels. The total size of the software keyboard was 450 × 720 pixels. An asterisk is displayed above the software keyboard to give feedback to the user when a key is typed. Password entry is completed when the user presses the space key on the physical keyboard; pressing the enter key on the screen keyboard breaks the anonymity of the real cursor, especially when the screen is video recorded. We did not support delete or cancel functions in our prototype implementation.

3.DISCUSSION

In this section, we discuss various issues related to the design of the Cursor Camouflage method.

Margins in the software keyboard. Our experiments showed that it is better not to have margins between adjacent keys in the software keyboard to achieve better protection. This is because an attacker can easily judge that a cursor on a margin at the moment when a key is typed is a dummy cursor. This strategy becomes impossible if we lay out the keys tightly, without margins. However, small margins increases the possibility of typing wrong keys, so the margin should be carefully determined considering usability-security tradeoff.

Confirmation key. In our current implementation, we use the space key on the physical keyboard as the confirmation key to signal the end of password entry. We did not include a confirmation key (OK or Enter) on the software keyboard because an attacker can judge that the cursor on the confirmation key at the moment when the password entry is finalized is the real cursor. We did not provide cancel or backspace keys on the software keyboard for the same reason. This is not a serious issue if the attacker is directly observing the typing action in real time, but it *can* become serious if the attacker is observing a video-recorded typing sequence. A possible solution is to reset the real cursor each time after a key is typed, but this requires the user to search for the real cursor every time, which results in significant overhead.

Display of asterisks. Our current implementation shows an asterisk as visual feedback when the user types a key. Whether the asterisk should be shown or not is essentially a usability and security tradeoff issue: showing it is more usable but more vulnerable, and hiding it is less usable but provides stronger protection. Showing asterisks or not is still a matter of discussion even among security experts, and there is no definitive answer. Our recommendation is to show the asterisks when using Cursor Camouflage because it is too difficult for the user to type keys without appropriate feedback.

4. Conclusion

In this paper, we presented a technique to protect password entry when using a software keyboard and indirect input device from shoulder surfing. It displays and moves multiple dummy cursors on the screen along with the real cursor so that it is difficult to identify which key is being typed. We explained how we designed the movement of the dummy cursors and presented the results of a user evaluation.

It is important to point out that no single security method can provide foolproof protection. Individual methods only reduce the success ratio of attacks, and multiple security methods must therefore be used in combination. The proposed method is advantageous because it can easily be installed on top of standard password entry systems using a software keyboard to provide additional protection. Our method can be combined with other protection methods, including those using image-based passwords, those showing the labels of the software keyboard only at the beginning of password entry, and others.

The proposed method leverages an individual's perceptual capability to identify an object being controlled by him or herself by using movement information only. Human perception covers many interesting capabilities, such as the ability to identify hidden structures in a noise pattern by using movement information. Thus far, these observations have primarily been investigated from the scientific point of view, and real-world application has been limited to artistic exploration and entertainment purposes. We hope that our work inspires more efforts to apply human perception to real-world applications such as security.

References

1. Roth, V., Richter, K. and Freidinger, R. A PIN-entry method resilient against shoulder surfing. In Proceedings of the 11th ACM conference on Computer and communications security (CCS '04). pp.236-245. 2004.
2. Takada, T. FakePointer: An Authentication Scheme for Improving Security against Peeping Attacks Using Video Cameras. UBICOMM '08, pp.395-400. 2008.
3. Wiedenbeck, S., Waters, J., Sobrado, L., and Birget, J. Design and evaluation of a shoulder-surfing resistant graphical password scheme. In Proceedings of the working conference on Advanced visual interfaces (AVI '06). pp.177-184. 2006.
4. Huot, S., Chapuis, O., and Dragicevic, P. TorusDesktop: pointing via the backdoor is sometimes shorter. In Proceedings of the 2011 annual conference on Human factors in computing systems (CHI '11). pp. 829-838. 2011.