Techniques for Visual Feedback of Security State

Tara Whalen and Kori Inkpen
Dalhousie University
Halifax, NS, Canada
{whalen, inkpen}@cs.dal.ca

Introduction and Overview

Security applications have significant challenges that make them complicated to configure and to use properly. One difficulty is that security is the result of a complex state, one that depends on many interactions. This state is hard to represent to the user clearly and completely, although it is vital that such information be available [1]. It has been recognized [2] that users require appropriate feedback so that they can assess the state of security and take appropriate actions. For example, studies conducted on browser interfaces outlined how better feedback could allow users to manage cookies in accordance with their security and privacy preferences [3].

This requirement—feedback—is what we discuss in this paper: techniques for presenting security information in clear manner. We are planning to conduct an investigation into how users can be made aware of the state of security of their networked applications and/or systems. This work is at the early development stage, which makes this workshop a valuable forum for feedback on possible research directions

As a starting point, we considered a metaphor for visualizing the state of a secure communication channel: the Security Spyglass. The Spyglass consists of a lens that represents the perspective of users along the path from sender to receiver(s). It is designed to "look inside" the channel, showing the state of confidential and/or authenticated data from various viewpoints, such as the intended recipient's view (see Figure 1). A fundamental element of this design is the simplicity of the Spyglass. It is intended to give rapid, clear feedback that can alert users to security misconfigurations. This metaphor could be used in a variety of applications, such as email or CSCW tools.

To consider a simple example, imagine that the Spyglass is integrated into a text messaging application. Alice wishes to send a secret message to Bob; we will assume that she has Bob's public key, and will use this key to configure a secure channel. If she wants to preview her settings, she can write the message, then, before sending it, use the Spyglass tool. The Spyglass will appear in the application, and can be set to show the viewpoint of various parties who have access to the data transmission. In this case, the relevant parties would be Bob, and everybody else (i.e., *world*), to whom Alice does not wish to reveal her message. If she has correctly configured the connection to Bob (using his public key), then she could set the Spyglass to "Bob", to see if Bob will be able to read her message.

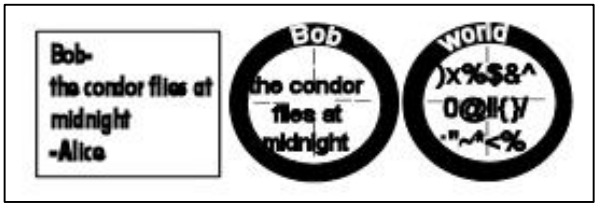

Figure 1: example of Spyglass tool

If the Spyglass shows the message text within its lens, then Alice can see that she has correctly configured a message that Bob can read. As well, if Alice wanted to know what eavesdroppers would see if they intercepted her message, she could change the perspective of the Spyglass to "world". If the connection was correctly configured, the Spyglass would show garbled, unintelligible text, to show at a glance that her message cannot be read by outsiders. Additional information, such as which key was used to encrypt the data, could be displayed at the edges of the lens.

Research Challenges

This Spyglass example is rather simplistic, used mainly for illustrative purposes. The problem of creating a clear, understandable visualization of security is a difficult one, which leaves us with a number of research challenges. One of the most important steps is to discover what sort of security information is most valuable to present to users. This could encompass data about the state of their host (including virus protection and firewall settings), as well as about other entities to whom they connect over the network. Once we have determined a useful set of data, we must then develop a means for presenting it in a meaningful way. (This presentation may involve the Security Spyglass, if this tool is shown to be helpful to users.) We welcome participants' ideas about how to meet these challenges.

References

[1] Whitten, A., and Tygar, J. D. (1999) *Why Johnny Can't Encrypt: A Usability Evaluation of PGP 5.0*. Proceedings of the 8th USENIX Security Symposium, pp. 169–184.

[2] Grinter, R. E., and Smetters, D. K. (2003) *Three Challenges for Embedding Security into Applications*. Workshop on HCI and Security Systems, CHI 2003, http://www.andrewpatrick.ca/CHI2003/HCISEC/HCISEC-papers.html

[3] Millett, L.I, Friedman, B, and Felten, E. (2001) *Cookies And Web Browser Design: Toward Realizing Informed Consent Online*. Proceedings of CHI 2001, 46–52.