A User Study of Off-the-Record Messaging

Ryan Stedman Kayo Yoshida Ian Goldberg

University of Waterloo
200 University Avenue West
Waterloo, Ontario, Canada N2L 3G1

{rstedman@cs, k2yoshid@math, iang@cs}.uwaterloo.ca

ABSTRACT

Instant messaging is a prevalent form of communication across the Internet, yet most instant messaging services provide little security against eavesdroppers or impersonators. There are a variety of existing systems that aim to solve this problem, but the one that provides the highest level of privacy is Off-the-Record Messaging (OTR), which aims to give instant messaging conversations the level of privacy available in a face-to-face conversation. In the most recent redesign of OTR, as well as increasing the security of the protocol, one of the goals of the designers was to make OTR easier to use, without users needing to understand details of computer security such as keys or fingerprints.

To determine if this design goal has been met, we conducted a user study of the OTR plugin for the Pidgin instant messaging client using the think aloud method. As a result of this study we have identified a variety of usability flaws remaining in the design of OTR. These flaws that we have discovered have the ability to cause confusion, make the program unusable, and even decrease the level of security to users of OTR. We discuss how these errors can be repaired, as well as identify an area that requires further research to improve its usability.

Categories and Subject Descriptors

H.1.2 [**Models and Principles**]: User/Machine Systems—*Human Factors, Software Psychology*; H.5.2 [**Information Interfaces and Presentation**]: User Interfaces—*Evaluation/ Methodology, Graphical user interfaces (GUI)*; H.5.3 [**Information Interfaces and Presentation**]: Group and Organization Interfaces—*Evaluation/Methodology, Synchronous interaction*; K.4.1 [**Computers and Society**]: Public Policy Issues—*Privacy*

General Terms

Experimentation, Human Factors, Security, Design

Symposium On Usable Privacy and Security (SOUPS) 2008, July 23–25, 2008, Pittsburgh, PA, USA.

Keywords

OTR, Usable Security, Instant Messaging, Think Aloud

1. INTRODUCTION

There has been much research into creating privacy-enhancing technologies, especially since the Internet has started to play an essential role in everyday life. However, not many of these technologies have seen widespread adoption. One of the reasons for this is that many of these technologies provide insufficient usability [8].

The process of evaluating and enhancing usability is important in order for a privacy-enhancing technology to provide benefits to ordinary users. Since privacy is not just intended for computer scientists or cryptographers, but for everyone, these technologies should be accessible to the general population. However, when people cannot figure out how to use those technologies, they may use them incorrectly, in which case the technologies may fail to provide the level of privacy that they are intended to provide, or give up and decide not to use them at all.

Communication across the Internet has become commonplace using instant messaging, largely for the reason that instant messaging clients are convenient and very easy to use. However, most of the instant messaging services are vulnerable to attacks that can reduce the privacy of the conversations using them. In particular, instant messages are generally relayed through a central server, which provides a convenient point of access for an attacker. To solve this problem, there have been a number of privacy-enhancing technologies developed for instant messaging, though currently the best solution is Off-the-Record Messaging (OTR), as it provides the highest level of privacy. A brief comparison of OTR and other privacy enhancing technologies is provided in section 2.1.

In the most recent redesign of OTR, one of the goals of the designers was to make it easier to use for the general public, in order to avoid the lack of adoption due to poor usability which has been seen in other privacy-enhancing technologies. To determine if this design goal has been met, we conducted a user study of the OTR plugin for the Pidgin instant messaging client. Our user study aimed to formally evaluate OTR in order to determine if the design of OTR is usable, or if usability flaws remain.

The remaining sections are organized as follows. Section 2 gives the overview of OTR and how it interacts with a user with emphasis on the authentication process. Section 3 describes the experimental design of the user study, section 4 details the result obtained from the actual experiments,

sections 5 and 6 discuss the design issues we observed and some suggested solutions, and section 7 concludes.

2. OFF-THE-RECORD MESSAGING

2.1 Overview of OTR

OTR was introduced by Borisov, Goldberg, and Brewer in 2004 as a protocol that provides privacy in low-latency online social communications [3]. The trend was to use IM for such communications, and so it was first implemented as a plugin for the Linux IM client GAIM (now called Pidgin [5]). Currently, OTR is available in many forms. There is a plugin for Pidgin, and third-party plugins are available for other IM clients, including Trillian [15] and Kopete [17]. Adium [19] for Mac OS X and the command-line client climm [14] (formerly known as mICQ) have OTR built into them. One can also set up a localhost AIM/ICQ proxy in order to enable the use of OTR with other IM clients, including Apple's iChat [2]. OTR has been modified recently by Alexander and Goldberg [1] in order to enhance its usability.

OTR provides conversations over IM with desired properties analogous to those of private face-to-face conversations. OTR encrypts messages in order to make them readable only to Alice and Bob. Alice is assured that she is talking to Bob by authentication. Alice is given deniability; that is, no one, including Bob, can prove the authorship of Alice's messages to third parties. Moreover, even if a secret key is stolen, no past messages will be decrypted, thus providing perfect forward secrecy [1, 3].

Other privacy-enhancing technologies for IM are available, but do not have the full features that OTR provides. The IM client Trillian [15] comes with a built-in encryption system called SecureIM, but it does not authenticate messages at all, and so it is vulnerable to man-in-the-middle (MITM) attacks. The pidgin-encryption plugin [20] does a little better by giving both encryption and authentication, but perfect forward secrecy and deniability are not provided.

Another alternative is Secure Internet Live Conferencing (SILC) [18], which provides all four security features provided by OTR, but it suffers from several drawbacks. One of the major drawbacks of SILC is that to use it both parties of the conversation must be using the SILC network; it does not support any other IM protocols. Also, messages sent using SILC are shared with the SILC servers, reducing the amount of privacy provided. SILC does provide a means to establish encryption private from the SILC servers, but this may not always be possible, such as in a NAT or firewall situation, and may hamper perfect forward secrecy.

If the goal is to allow your IM environment to have the same level of privacy as off-line communications, OTR is currently the best option.

2.2 An Authentication Problem

OTR uses a symmetric-key encryption scheme to encrypt messages, and a variant [13] of the familiar Diffie-Hellman key exchange protocol [4] for key sharing. In order to prevent MITM attacks, long-lived public/private key pairs are used to authenticate the initial DH key exchange. All subsequent message authentications are only valid if this initial key exchange is done properly. If Alice and Bob do not know each other's public keys in advance, however, then there is no way to detect if Eve is impersonating Alice for Bob and presenting her public key claiming that it is Alice's. This MITM attack will be easy unless some measure is taken. In fact, a plugin module for the ejabberd IM server [7] has been written, which will perform this MITM attack automatically.

Figure 1: The four privacy level indicators

One solution to this problem is to ask Alice and Bob to verify each other's fingerprints (hash values of their public keys) out-of-band. For example, Alice can make a phone call to Bob, ask Bob to read his fingerprint off the screen, and confirm that it matches the fingerprint displayed on her screen. Previous implementations of OTR used this method to authenticate buddies. One problem with this method, however, is that it is rather cumbersome. Moreover, if Alice does not know much about security or cryptography, then this method will only confuse her. If she does not understand what it means to verify Bob's fingerprint, then she may blindly accept anybody's fingerprint. In order for average users to be able to properly use OTR, a more intuitive alternative is desirable.

One of the main purposes of the recent redesign of OTR [1] was to solve this authentication problem. The key idea is to use information shared by Alice and Bob that is hard for anyone else to guess; for example, the place where they first met. When Alice and Bob want to authenticate each other, they both enter the secret into a pop-up window. If they enter the same value, Alice is convinced that she is talking to Bob, and vice versa.

The process of buddy authentication described above will be the main focus of our user study. We would like to know if average users can properly go through this authentication process without problems. For the experiment, we decided to use the OTR 3.1.0 plugin for Pidgin 2.2.1 running on Windows XP since Windows is the most popular user platform. We will describe below how OTR interacts with a user under this environment, with an emphasis on the buddy authentication process.

2.3 OTR User Interaction

2.3.1 OTR Button and the Privacy Levels

When Alice enables OTR, the OTR button appears on the lower left corner of a conversation window with Bob. The OTR button has three roles.

First, it serves as an icon indicating one of the privacy levels of the conversation: *Not private*, *unverified*, *private*, and *finished* (Figure 1). *Not private* means that the OTR plugin is enabled but it is not providing any privacy; for example, because Bob's IM client does not support OTR. *Unverified* means that the conversation is encrypted, but Alice has not authenticated Bob; this leaves Alice potentially vulnerable to MITM attacks. *Private* means that Alice has authenticated Bob. *Finished* means that Alice's buddy Bob has ended the private conversation either by closing his IM client or by choosing the option 'End private conversation' (Refer to the last paragraph of this section 2.3.1).

Second, by clicking the OTR button, it starts (when the

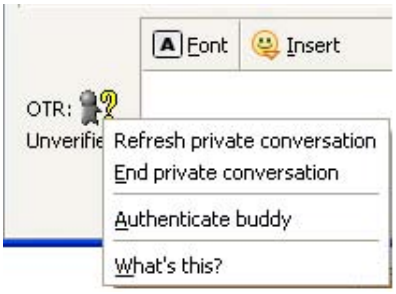

Figure 2: The OTR button in Pidgin. When right-clicked, a menu with additional options appears, as shown.

Figure 3: The OTR configuration window

current state is 'not private') or refreshes (when 'unverified', 'private', or 'finished') a private conversation.

Finally, and perhaps most importantly, right-clicking the OTR button gives four additional options to choose from: 'Start private conversation/Refresh private conversation' (the appropriate one of the two options is shown depending on the current state of OTR), 'End private conversation', 'Authenticate buddy', and 'What's this?' (Figure 2). The first option 'Start private conversation/Refresh private conversation' has the same effect as left-clicking on the OTR button. 'End private conversation' stops encryption and message authentication. 'Authenticate buddy' brings up a dialog box for authentication (Figure 4). 'What's this?' is a link to the help page 'Privacy levels' [11] that explains the four states of OTR.

2.3.2 Configuration Window

OTR's configuration window (Figure 3) is available from the 'Plugins' option of Pidgin. Among various options available, we will only describe those relevant to our user study. The configuration window has two tabs: 'Config' and 'Known fingerprints'.

Under the 'Config' tab, Alice can see her fingerprint, and can generate her public key/private key pair by clicking the button 'Generate' if she does not have one yet (although OTR will automatically generate a key for her when needed). By default, OTR automatically makes conversations private whenever possible; Alice can also choose not to do so by unchecking the option 'Automatically initiate private messaging'.

The second tab, 'Known fingerprints', shows all of the fingerprints that OTR has seen so far, and whether or not Alice has verified them. If Alice wishes to authenticate Bob by verifying his fingerprint manually, she can do so by clicking on Bob's screen name, and clicking the button 'Verify fingerprint', which brings up a fingerprint verification window (Figure 5). More details about fingerprint verification will follow in section 2.3.6.

2.3.3 Help Pages

OTR has three help pages available: 'Privacy levels' describes the four privacy levels illustrated in section 2.3.1, 'Authentication' explains how to authenticate buddies using a shared secret, and 'Fingerprints' explains how to authenticate using fingerprints [9, 10, 11]. They are available through the OTR button, hyperlinks that are embedded in the messages OTR displays in the conversation window, and also through the buddy authentication and fingerprint verification windows.

2.3.4 Starting a Private Conversation and Authentication Process

If the 'Automatically initiate private messaging' option is checked (see section 2.3.2), then starting a private conversation is as simple as sending an initial message using Pidgin. If not, simply clicking on to the OTR button will initiate a private conversation.

OTR first checks if Alice and Bob have talked with each other before. If Alice and Bob's OTR-enabled clients have seen each other before, then the buddy authentication process described below is not necessary since Alice and Bob use the same public/private authentication key pairs which were previously verified, and the conversation is automatically encrypted and authenticated. Otherwise, the OTR button indicates that the conversation is 'unverified'. OTR also injects messages into Alice's chat window and tells her that the conversation is unverified and she should authenticate Bob. She has an access to the help pages 'Privacy levels' and 'Authentication' [9, 11] through the hyperlinks embedded in the words 'unverified' and 'authenticate', respectively.

Alice and Bob can choose one of two ways to authenticate each other: improved user authentication method using a shared secret, or by manual fingerprint verification. OTR assumes that most users would prefer to use shared secret authentication, making shared secrets the default method of authentication. Fingerprint verification is still available through the 'Advanced' button on the buddy authentication window (section 2.3.6).

2.3.5 Authentication (Shared Secret)

If Alice and Bob want to authenticate each other using a shared secret, one of them, say Alice, must right-click the OTR button and select the 'Authenticate Buddy' menu option (Figure 2). This will bring up the dialog box pictured in Figure 4.

If Alice and Bob had agreed upon which secret to use for the authentication process, for instance, when they last met in person, then the next step is to enter the secret value in the dialog box. Otherwise, they need to first decide what the

Figure 4: The OTR authentication dialog box

Figure 5: Verify fingerprint window

secret information will be. They can do so using the unverified IM, but with care. That is, they must not communicate the secret value itself. For example, Alice may write, "Let's use the name of the restaurant where we first met", but she should not write, "Let's use 'Sea Side OTR' as our secret".

Once Alice and Bob know which information to use, whoever initiated the buddy authentication process, Alice in our example, will first need to enter the value in the dialog box. Once Alice hits 'OK', the same dialog box will appear on Bob's screen. At the same time, Alice will see a message window called 'Authenticating Buddy' with a progress bar (Figure 9). When Bob enters a secret value and hits 'OK', the same progress bar comes up on Bob's screen as well.

If Alice and Bob enter the same secret value, then this completes the buddy authentication process. The 'Authenticating Buddy' windows on Alice and Bob's screen will indicate that the authentication was successful, and the OTR button will change to read 'OTR: Private'. If they enter different values, however, the protocol fails and they will have to redo the process. Alice and Bob learn nothing in this case other than the fact that they entered different values; this is important so as to prevent a MITM from doing an offline attack on the secrets entered by Alice and Bob. If the process keeps failing, then they should be suspicious about the identities of their correspondents.

2.3.6 Authentication (Fingerprint Verification)

Even though OTR has developed a new way to authenticate buddies, Alice and Bob can still choose to manually verify fingerprints instead of using the new method described above. To do so, Alice should click on the 'Advanced' button once the buddy authentication window pops up. Alternatively, she can go to the OTR configuration window (section 2.3.2), choose Bob's screen name under the 'Known fingerprints' tab, and click on 'Verify fingerprint'. This will bring up a 'Verify fingerprint' window (Figure 5).

The window shows Alice and Bob's fingerprints, and asks Alice to verify Bob's fingerprint. Clicking on 'What's this' in the bottom of the window gives a brief explanation about what a fingerprint is, and further clicking onto 'More' gives more detailed description about how to properly authenticate buddies. At the bottom of the explanation, there is a hyperlink to the OTR help page 'Fingerprints' [10] (see section 2.3.3).

There are several things to note in the fingerprint verification process. As briefly mentioned in section 2.2, Alice should verify Bob's fingerprint in a secure channel other than the IM conversation they are using. Also, the fingerprint verification is one-way; that is, Alice and Bob should each verify the buddy's fingerprint if both parties wish to convince themselves that they are speaking to the right person. This is different from shared secret authentication process which does mutual buddy authentication. Of course, if Bob does not understand what fingerprints are, he can choose to do the secret sharing authentication even if Alice has already verified Bob's fingerprint. In fact, Alice and Bob have the option to authenticate each other as many times as they choose.

3. EXPERIMENTAL DESIGN

In this and the following section, we will describe the design and the results of our experiment.

The method we have chosen to test the usability of OTR is the think aloud method [16], which entails having participants perform a set of tasks and having them vocalize their thoughts as they perform these tasks. We chose this method because it allows us to understand how a user perceives the system, rather than just observe how the user interacts with the system. For instance, if a participant performs an action that causes an error, they would say why they are performing the action, giving us insight into any misconceptions that caused that action to be performed.

The assumption of OTR is that two users speaking with each other have prior knowledge of each other off of which they can base some shared secret. It was important for us to capture this assumption in this study, so that the problems observed could not be confused with problems introduced because the two users communicating did not know each other. To account for this, we decided to recruit four pairs of friends to participate in this study. We believed that this would provide a basis for the participants to establish a shared secret with one another.

One aspect of online messaging that the designers of OTR do not seem to take into account, however, is that people communicating online may not have previously been in contact. This can limit the basis on which they can establish a shared secret. Imagine, for instance, if one user gets another person's IM address from a message board and wishes to communicate privately with this person. In this scenario the two people have some basis to form a shared secret, as they are both readers of the same message board, but they do not have any personal knowledge of each other. In this

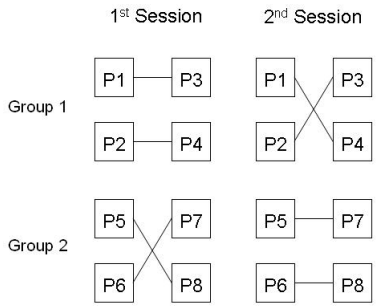

Figure 6: The layout of how the users communicated with each other across the two sessions. P1 and P3 are friends, P2 and P4 are friends, and so on.

case, the best that can be done is to authenticate the buddy as "some reader of the message board".

We wished to include this kind of scenario into our study in order to determine if there are additional usability issues influenced by the degree to which the two users know each other. To simulate this, in some of our sessions, participants interacted with their friends, and in other sessions, they interacted with other (non-friend) participants, as indicated in Figure 6. In this case the participants were all from the same university, which was a basis for them to determine a shared secret even though they did not personally know each other.

Another important aspect of any user interface that we wished to evaluate was the learnability of that interface. We assumed that the participants we recruited were unfamiliar with OTR, and there are bound to be issues when using a new system. To evaluate how learnable OTR is, we held a second evaluation session where we asked the users to perform the same tasks as in the first session. In this session we evaluated if users were still having trouble using OTR and repeating errors that they had already experienced.

To stop the participants from simply copying the steps they performed in the first session, including establishing the same shared secret, we switched who each participant was talking to (Figure 6). It is important to point out that in each session we had a group that is talking to their friends, and a group that is talking to somebody they do not know. This allowed us to identify any issues caused by users talking with someone they do not know so that they are not mistaken as learnability issues in the second session.

Participants were placed in separate rooms with an experimenter in each room to observe the users as well as provide instruction. After a short explanation of what we wanted them to do, they were asked to send instant messages to each other for a short time before enabling the OTR plugin for Pidgin. The purpose of these tasks were to get them used to the Pidgin interface, as well as to get them used to thinking aloud. After they enabled the OTR plugin they were asked to start and authenticate a private conversation using OTR, which is the main focus of this study. The last task we asked the users to perform was to end the private conversation and then re-establish it.

4. RESULTS

We ran the experiment with four pairs of friends, all students of the University of Waterloo. We will call the eight participants P1 through P8. Five of the eight participants (P3, P4, P5, P7, and P8) were male, and the remaining three were female. P5 was a graduate student and all the remaining were undergraduate students; the ages of the undergraduate participants varied. Four of the participants were from the Computer Science department, two others were from other sciences, and two were in the area of arts. All of the participants said they use IM on a daily basis. Seven of the participants had never used OTR before, and one of the participants (P4) had seen OTR before, but had only used it very briefly.

We will give the results of the first sessions (see Figure 6) in sections 4.1, 4.2, 4.3, and 4.4. In the first sessions, P1 and P3, P2 and P4, P5 and P8, and P6 and P7 talked with each other, respectively, where the first two pairs were friends and the latter two were non-friends. Section 4.5 describes the second sessions of the experiment.

4.1 Generating a Private Key and Initiating a Private Conversation

None of the participants had any problems generating private keys. Most of the participants had their keys generated automatically when initiating their first unverified private conversation, although P5 and P7 manually generated them in the plugin configuration window. In all cases the private keys were generated without any issues or confusion as to what was happening.

Figure 7: Conversation window with OTR messages

Since the default setting was to automatically start private conversations, all of the participants were able to begin unverified private conversations without any apparent problems, either by sending initial messages, by receiving messages from their buddies, or by pressing the OTR button (P6). The OTR button then turned from 'OTR: Not private' to 'OTR: Unverified' as seen in Figure 7.

However, an issue arose when P3 clicked on the OTR button to start a private conversation before P1 had turned on the OTR plugin. When this happened P3 was presented with the dialog message 'Attempting to start a private conversation...'. P1 received a message indicating that P3 was attempting to start a private conversation using OTR, and given a link to the OTR homepage, but no indication was

given to P3 that P1 had not even enabled OTR yet. This was resolved when P1 enabled OTR and a private conversation was started.

After unverified private conversations were started most of the participants realized that their 'buddy' needed to be authenticated from the dialog given by OTR, seen in Figure 7. The participants P2 and P7, however, did not immediately recognize that authentication was needed, the reason for which will be discussed later.

4.2 Starting the Authentication Process

One of the largest difficulties that the users encountered was determining how to start the authentication process in OTR. As described earlier in section 2.3.5, this process is started simply by right clicking on the OTR button and choosing 'Authenticate buddy' (Figure 2), but all of the participants had considerable difficulty figuring this out.

One of the main tools that OTR uses to guide users into the proper use of the system are help pages that are linked to from the OTR plugin [9, 10, 11]. The first thing that many of the participants (P1, P3, P6 and P8) did when a dialog from OTR seen in Figure 7 appeared was to click on the 'authenticated' hyperlink, which brought up the authentication help page [9]. We would like to note, however, that the 'authenticate' link did not appear in P2 and P4's chat windows, and that might have caused some confusion or delay in the process. P4 later saw the 'unverified' help page and followed a link from that page to get to the authentication help page.

Some of those who clicked on the 'authenticate' link thought that it would perform the authentication, and were then somewhat surprised when a browser popped up bringing them to the help page. Some of them were even put off by the size of the help page, with P3 commenting "Do I have to read all these?". Both P1 and P3 only skimmed over the help page, not reading any of it in much detail, hoping to quickly find out how to authenticate their buddies.

The apparent problem with this help page is that it only says to "click on 'Authenticate Connection' on the OTR button" in order to start the authentication process, and has no specific reference of right-clicking on the button. The participants who read this then attempted left-clicking on the OTR button, in some cases multiple times, expecting it to solve the authentication problem, which only refreshed the conversations. P8 initially thought that this had achieved authentication, but was unsure due to confusion over what was meant by refreshing the conversation. P5 and P6 were also confused as to what refreshing the conversation actually did.

After realizing that clicking on the OTR button was not accomplishing anything, they started exploring the Pidgin interface, and looking for more help sources. Three of the participants (P1, P4 and P5) clicked on the 'unverified' link (Figure 7) looking for help. However, none of them found information on this page useful.

Two of the participants, P2 and P7, decided to manually verify the fingerprints of their buddies (P4 and P6, respectively), and at first did not even attempt to initialize the 'shared secret' authentication. More about this issue will be discussed in the following section.

Eventually, some of the participants figured out to right click on the OTR button after some effort (P1, P6 and P8), but P4 needed to be prompted by the experimenter after five

The following is an example of what **not** to do:

Here you are telling the other person exactly what to do. An imposter can figure out what to type in just as easily as your friend can.

A better way to pick a secret would be something like this:

Figure 8: A part of a help page 'Off-the-Record Messaging: Authentication' describing what not to do

minutes of struggling to find the correct action, as neither he nor his buddy were able to figure it out within that time. P2, P3, P5, and P7 did not find out the 'right-click' on their own as their buddies initiated authentication, although P3 eventually learned how when P1 explained how it was done over the IM conversation.

4.3 Authentication

Once the participants figured out how to start the authentication process, properly establishing a shared secret was another hurdle that they encountered. One major issue that occurred, which severely reduces the amount of privacy that OTR can provide, is that some of the participants (P1, P4 and P8) explicitly wrote suggested secrets in the IM window as part of their chat messages. When questioned afterward, they said that it was because of the ordering of the examples in the help page (Figure 8), which shows what not to do before showing what to do. They said that they glanced through the help page, saw the example image in which one person tells the other person what exactly the secret word would be, and just copied it without reading any of the text around it. While sending the secret, P4 realized that this did not seem secure, but he nonetheless continued because he was just following the instructions. P1 initially tried to describe the secret rather than writing it explicitly. She changed her mind after failing authentication multiple times.

A couple of the participants who did not read that section of the help page (P5 and P6) encountered another problem. Because they did not read the help page, they did not know how they were meant to set up shared secrets, so they used the text box in Figure 4 to enter questions for their buddies to answer instead of secrets themselves.

Another issue arose with P3, who did not initiate the authentication process. He thought that he had caused the authentication message box (Figure 4) to appear. Because of this, he entered a random secret he thought of instead of the secret suggested by his buddy, which caused the authentication to fail.

As mentioned in the previous section, P2 and P7 authenticated their buddies (P4 and P6, respectively) by fingerprint verification in the OTR configuration window. P2 had

the configuration window open when she was asked to authenticate her buddy, and decided to verify P4's fingerprint because she thought that was what she was supposed to do to authenticate her buddy. Similarly, while turning on OTR, P7 decided to look in the plugin configuration, and then looked at the 'Known fingerprints' tab. While in here he saw that P6 was unverified, and took this to mean that messages sent to this participant were insecure, so he clicked on the 'Verify Fingerprint' button, and chose to verify the fingerprint (Figure 5).

Three problems involving this manual authentication process were observed. First, both P2 and P7 'verified' the fingerprints without bothering to ask their buddies for their fingerprints in some out-of-band means. P2 knew that it was an inappropriate way by reading the detailed description about fingerprints given under 'What's this?' in the 'Verify fingerprint' window (Figure 5). She justified her action by commenting that she could not think why anyone would impersonate P4.

Second, this process turned their OTR buttons to 'OTR: Private', and thus P2 and P7 did not initially realize that their buddies were still having trouble authenticating them. Further on in the experiment they both realized the error through communication with their buddies, as both of the other participant's OTR buttons were still set to 'OTR: Unverified'. Once this error was realized P7 began searching for how to successfully perform the authentication, though P2 just waited for her buddy to resolve the problem.

Another problem occurred when P7 was being authenticated by P6. P6 initialized the authentication process, causing the authentication message box to appear for P7. However, since he had already manually authenticated P6, the message box told him that the buddy had already been authenticated. This confused P7, so he clicked on the 'Advanced' button which brought up the message box seen in Figure 5, which he had already seen and he clicked 'OK'. Pressing 'OK' did not bring the message box seen in Figure 4 back up, so P7 assumed that the authentication had completed. However, P6 had no notification of any change that occurred on P7's side, and the progress bar remained at the same position. This caused a considerable amount of confusion, and it took some time for the two participants to realize the error.

Even when the participants had realized how to establish a shared secret, some of the pairs (P1-P3 and P6-P7) still had problems getting the shared secret right. Even though they were using secret questions for which they thought the other person should know the answers, authentication failures still occurred multiple times. These situations occurred when the person answering the secret would not know the correct answer, or would spell it differently from the other participant; for example, using capitals where the other participant did not.

After the participant who initiated the authentication entered a shared secret, the 'Authenticating Buddy' window (Figure 9) was presented on the initiator's side. The progress bar in the window would stop moving, however, until the other participant also typed in a keyword in the authentication window and hit 'OK'. There was no description around the progress bar as to what exactly was happening. P7 noted that he assumed it was waiting for the other participant to answer the question, but that he could not be sure it was not frozen.

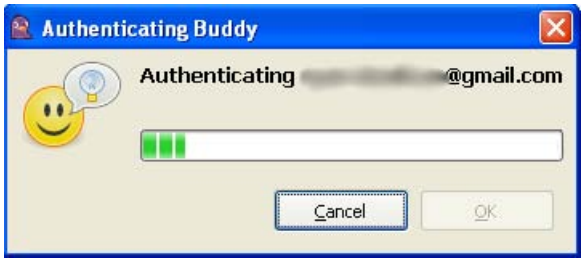

Figure 9: The progress bar shown while a buddy is being authenticated

4.4 Restarting a Private Conversation

Most of the participants except P3 had no problems when restarting private conversations with the other participants. They either sent messages or clicked on the OTR button, and noticed from the dialog messages given by OTR and the icon changes in the OTR button, that their conversations were automatically made private without authentication. P3 did not realize this first, and he initiated authentication again. The authentication went successfully, however, using the same secret as the one the pair used in the previous authentication.

4.5 Learnability Session

The second session (the right column of Figure 6) went much more smoothly than the first, with a very small number of problems as compared to the first session. P5 was the only participant that struggled to figure out how to start the authentication process, since he did not learn how to initiate authentication in the first session. This did not cause much delay, however, since after a while the participant he was talking with, P7, started it instead.

Once again, P7 first manually authenticated his buddy. He did this because he had done it in his first session, and did not realize that the fingerprint verification was not needed. He did, however, also initiate the 'shared secret' authentication with P5 since that was what happened in his first session with P6.

There were many fewer problems in setting up a shared secret. Most of the participants were able to establish a secure shared secret and authenticate on the first try. P4 no longer sent the secret over IM, and both P4 and P7 described not only the suggestion for secrets but also the format of the secrets. The one exception was P3, who sent the secret over IM. His reasoning though, was that because he did not know the participant he was speaking with, it was easier to just send her the secret.

5. DESIGN ISSUES WITH OTR

5.1 Feedback

One of the first issues that we observed is the lack of feedback on whether a buddy's IM client supports OTR at all. If a person wishes to set up a private conversation using OTR with a buddy and that buddy does not have OTR installed, it will display a message to the buddy informing them of the request, but no feedback is given to the person initiating the request. This is a relatively minor point, as a simple exchange between the two buddies can reveal that one of them does not have OTR installed; however, there is

still a possibility of confusion or frustration to the initiator as to why OTR is not working.

The messages that OTR injects into the conversation window, seen in Figure 7, are used as a tool to notify users that further authentication is needed. Although this feedback was successful in making the users realize that they needed to somehow authenticate their buddy, there was some uncertainty in the participants as to what the 'authenticate' and 'unverified' links were for. Some of the participants thought that clicking on them would perform the authentication, and were surprised when it brought up a help page. This confusion could be avoided by rewording the message to make it more clear that the 'authenticate' link leads to a help page.

Another problem that we observed was the lack of feedback given in the progress bar (Figure 9). Although no direct problems were caused by it, it was noted by a participant that he had to assume what was happening when the progress bar had stopped moving. This lack of feedback could cause confusion, possibly causing a user to think that the process had frozen. This can be solved by adding messages to the progress bar's message box indicating the state of the authentication process, particularly that it was waiting for the buddy to enter his secret.

5.2 Manual Verification

Another issue concerning feedback was encountered during the experiments when P7 manually verified his buddy. Because there was no feedback to him that manually verifying his buddy did not do mutual verification, he became confused when his buddy later initiated the authentication process. Such confusion could be avoided simply by adding a message box when a user manually verifies a buddy warning that his buddy still needs to authenticate him.

Beyond simple feedback issues, the ability to manually authenticate a user caused other significant problems. Having the ability to manually authenticate a buddy makes sense for users with a more advanced knowledge of computer privacy who may feel more comfortable confirming a buddy's fingerprint out of band. However, an average user may not realize that this is how manual verification is meant to be done, as was seen with P2 and P7. OTR does have an explanation of how the manual verification should be performed, but to see it they need to click on the 'What's this?' expansion, seen in Figure 5, and then click on an additional 'More' expansion. This message box should be changed to clearly warn of the privacy implications of manually verifying a buddy as opposed to using the shared-secret method. Further, in order to protect an average user who may accidentally open the fingerprint verification window, it should clearly indicate that there is a more intuitive alternative available. On the contrary, however, the 'What's this?' expansion on the fingerprint verification window says 'A fingerprint is a unique identifier that you should use to authenticate your buddy'. The misleading word 'should' may make a user think that the fingerprint verification is the only way to authenticate his buddy.

One other issue with manual verification was seen with P7 when his buddy initiated the 'shared secret' authentication. Presented with the authentication dialog box, P7 clicked on the 'Advanced' button and went through the manual verification message box again. This caused a lot of confusion between P7 and the buddy he was talking with, since there was no indication to the buddy about what P7 had done, even to the point that the progress bar stayed on her screen unmoving, as well as there being no indication to P7 that the fingerprint verification did not complete the authentication process that his buddy requested. This 'Advanced' button should be removed altogether from the authentication message box of the buddy being challenged with the authentication, as it does nothing toward responding to that challenge.

5.3 Help Page

We observed several issues in the help page for OTR authentication that resulted in problems for some of the participants. One was in the presentation of the instructions on the page. For example, we observed some users looking at the images on the help page depicting how to set up a shared secret and imitating what they saw. However, since the image of what not to do was shown first, many of the participants set up their shared secrets incorrectly, undermining the level of privacy that OTR offers. This problem can be solved by showing the image of the correct procedure first, and making it more visually clear that the other example is the incorrect method, possibly by putting an 'X' through the image.

More generally, we found in the experiment that most of the participants would not read the instructions on the help pages in detail, and would instead just skim them, looking mostly at the pictures in the instructions to get an idea of what to do. This could be taken advantage of by laying out more of the instructions in a pictorial format, as it would aid users in quickly understanding the content of the instructions without needing to read them.

Another issue we observed with the help page for authentication was that it did not provide adequate information to properly initiate the authentication process. Although it does tell users to click on the OTR button, it makes no mention of right-clicking on the button. This caused considerable frustration, and more precise instructions need to be given in order to provide users with all the tools necessary to use the system properly.

5.4 OTR Button

The main reason that none of the users thought to right-click on the OTR button, besides the lack of instruction in the help page, is due to the affordances inherent in buttons. The OTR button has two actions that can be performed on it: pushing it (left-clicking) and right-clicking it, but buttons are only naturally associated with the affordance of pushing [6]. This is the likely reason that the participants had so much trouble realizing that right-clicking was the correct action to perform. Because there are multiple actions that need to be performed using the OTR button, showing the menu when the button is pressed would probably cause less confusion to users. The tooltip of the OTR button should also indicate that there are multiple actions that can be performed with the button, and how to access them.

Another confusion when initiating authentication occurred when participants left-clicked the OTR button and the OTR refreshed the conversations. There should be description in the OTR dialog or a help page explaining what refreshing conversation means. Otherwise, puzzled by the unfamiliar term, some may assume that the authentication has been properly carried out.

5.5 Authentication

We observed a lot of difficulty that the participants had in completing the authentication procedure once they learned how to start it. As discussed earlier, the ordering of the instructions in the help page caused confusion in how to establish a shared secret to use, which caused some participants to set up a shared secret improperly. Others thought the text box to enter the shared secret (Figure 4) was meant to be used to enter a question to base the secret on, causing the authentication to fail. One of the participants was unaware that the buddy he was talking to had initiated the authentication. He thus thought he had caused the authentication window to come up, which lead him to use it to enter his own shared secret, causing the authentication to fail. Also, there is no clear indication in OTR as to how to set up a shared secret without referring to the help page for instruction.

We believe that all of these issues could be solved by redesigning the authentication window (Figure 4) so that it allows not only a field to enter a secret but also a field for the user initiating the authentication to enter a question that the shared secret answers. The buddy would then see a message box containing the question that the initial user entered, as well as a space to enter the shared secret. Redesigning the authentication process in this manner should guide users into the proper use of the system without the need of additional help material. Separate areas for a question and an answer should make it clear that the shared secret (answer) should be separate from the question, greatly increasing the level of privacy given by the system. Additionally, since the buddy being sent the authentication request will see a message box that is different from the challenger's, with a question to be answered, there should not be confusion as to who initiated it, or what to do with it.

One point to note is that this proposed method of authentication is only one-way; that is, only one user initiates the authentication and the other responds. As mentioned in [1], for this question-and-answer style of secret sharing to be completely secure it must be performed in both directions (once by each user), and users must be made aware of this fact.

6. DISCUSSION

One factor we wished to address was the learnability of the OTR interface. We have seen that once users have figured out how to use OTR the first time they did not have any trouble using it a second time. However, this does not mean that the usability issues we have noted in the first sessions of our experiments are less urgent. If a user learns the incorrect way of using the application, they will continue to use it incorrectly, as was seen with P7 when he used the manual authentication in the second session. This could have serious negative consequences in the level of privacy that OTR can provide to that user. Also, if a user has problems with a program the first time they use it, they are less likely to continue using it.

Another idea we wished to study, as we noted earlier, is how the secret sharing process changes when a participant is talking with a friend as opposed to a non-friend. We saw, as would be expected, that on average people who knew each other had an easier time establishing a shared secret than those who did not. Although this was the case, the people who did not know each other personally were able to establish a shared secret based on knowledge they knew they would have in common, such as "What is the name of the university we attend?". This demonstrates that OTR can have a wider audience than people who know each other personally, such as purely on-line friends. When questioning the participants after the experiment, P4 noted that he in fact had friends that he only knew on-line.

One of the informal assumptions that OTR relies on is that users know each other off-line and are able to establish a shared secret through some off-line channel or shared knowledge. However, this may not always be true, and needs to be taken into account when considering the problem of having users establish a shared secret to authenticate each other. This problem extends beyond the use of OTR, as shared secret authentication is used in other applications, such as Interac's email money transfer [12], which uses shared secret authentication to authenticate recipients of a fund transfer.

The secret sharing method of authentication is a good idea in concept, but we have seen that there are some barriers to overcome that might deter users from the use of OTR. Some of the participants stated after the experiments that the difficulty they encountered establishing a shared secret as a reason for not wanting to use OTR in the future, even though they saw some potential use in the security it provides. Further research needs to be done on how to guide users toward establishing shared secrets that are secure and that both parties can answer. "What is the name of the university we attend?" is a question that both users can answer, but it is not very secure, as a number of people could easily find the answer to that question.

7. CONCLUSION AND FUTURE WORK

In redesigning Off-the-Record Messaging, one of the intents of the designers was to make OTR easier to use for a larger user base. However, even in a relatively small user study of only 8 students, most of whom had some technical experience with computers, each one of them had some sort of problem when trying to use OTR. As a result of this user study, we have discovered several usability flaws in the design of the OTR plugin for Pidgin. Many of the flaws identified are minor and are not likely to result in too much trouble for a user, but some are much more severe, and could cause the system to be used in an unintended manner, decreasing the level of privacy that OTR can provide, or even stopping a user from being able to use the system.

Although our suggestions for improvements should resolve most of the issues that we observed, users may still have problems creating secure shared secrets that both parties can answer. We believe that our suggested improvement of the authentication message box will successfully guide users toward creating a shared secret; however, it may not help guide them toward creating a *good* shared secret. Future work needs to be done to investigate methods of helping users to create secure shared secrets.

Throughout this paper, we have discussed changes in the design of the OTR plugin for Pidgin that we believe will ameliorate the usability issues observed. At the time of the writing, a group at the University of Waterloo is implementing the suggestions that we have made, and we hope to repeat this study when the modifications are complete.

8. ACKNOWLEDGEMENTS

We would like to thank the Natural Sciences and Engineering Research Council of Canada and the Mathematics of Information Technology and Complex Systems Network for their support of this work.

9. REFERENCES

[1] C. Alexander and I. Goldberg. Improved User Authentication in Off-the-Record Messaging. *Proceedings of the 2007 ACM workshop on Privacy in electronic society*, pages 41–47, 2007.

[2] Apple, Inc. Apple - Mac OS X Leopard - Features - iChat. http://www.apple.com/macosx/features/ichat.html. Accessed February 2008.

[3] N. Borisov, I. Goldberg, and E. Brewer. Off-the-record communication, or, why not to use PGP. *Proceedings of the 2004 ACM workshop on Privacy in the electronic society*, pages 77–84, 2004.

[4] W. Diffie and M. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, 22(6):644–654, 1976.

[5] S. Egan and others. Pidgin. http://www.pidgin.im/. Accessed February 2008.

[6] W. Gaver. Technology affordances. *Proceedings of the SIGCHI conference on Human factors in computing systems: Reaching through technology*, pages 79–84, 1991.

[7] O. Goffart. mod_otr — Man in the Middle module for Off-The-Record. http://ejabberd.jabber.ru/mod_otr. Accessed February 2008.

[8] I. Goldberg. Privacy Enhancing Technologies for the Internet III: Ten Years Later. In A. Acquisti, S. Gritzalis, C. Lambrinoudakis, and S. D. C. di Vimercati, editors, *Digital Privacy: Theory, Technologies, and Practices*, chapter 1. Auerbach, 2007.

[9] I. Goldberg, C. Alexander, and N. Borisov. Off-the-Record Messaging: Authentication. OTR Help Page. http://www.cypherpunks.ca/otr/help/authenticate.php?lang=en. Accessed February 2008.

[10] I. Goldberg, C. Alexander, and N. Borisov. Off-the-Record Messaging: Fingerprints. OTR Help Page. http://www.cypherpunks.ca/otr/help/fingerprints.php?lang=en. Accessed February 2008.

[11] I. Goldberg, C. Alexander, and N. Borisov. Off-the-Record Messaging: Privacy Levels. OTR Help Page. http://www.cypherpunks.ca/otr/help/unverified.php?lang=en. Accessed February 2008.

[12] Interac, Inc. Interac Email Money Transfer. http://www.interac.ca/consumers/productsandservices_ol_emt.php. Accessed February 2008.

[13] H. Krawczyk. SIGMA: The 'SIGn-and-MAc' Approach to Authenticated Diffie-Hellman and Its Use in the IKE Protocols. In *Advances in Cryptology—CRYPTO 2003*, Santa Barbara, California, USA, August 2003.

[14] R. Kuhlmann. CLI-based Multi-Messenger. http://www.climm.org/. Accessed February 2008.

[15] K. Kurtz and S. Werndorfer. Trillian. http://www.ceruleanstudios.com/. Accessed February 2008.

[16] C. Lewis and J. Rieman. *Task-centered User Interface Design: A Practical Introduction*. University of Colorado, Boulder, Dept. of Computer Science, 1993.

[17] D. M.-V. Prett and others. Kopete Instant Messenger. http://kopete.kde.org/. Accessed February 2008.

[18] P. Riikonen and others. Secure Internet Live Conferencing. http://silcnet.org/. Accessed May 2008.

[19] E. Schoenberg and others. Adium. http://www.adiumx.com/. Accessed February 2008.

[20] B. Tompkins. Pidgin-Encryption. http://pidgin-encrypt.sourceforge.net/. Accessed February 2008.