Secure Browser-Based Instant Messaging

Christopher D. Robison

A thesis submitted to the faculty of
Brigham Young University
in partial fulfillment of the requirements for the degree of

Master of Science

Kent E. Seamons, Chair
Daniel M. A. Zappala
Sean C. Warnick

Department of Computer Science

Brigham Young University

December 2012

ABSTRACT

Secure Browser-Based Instant Messaging

Christopher D. Robison
Department of Computer Science, BYU
Master of Science

Instant messaging is a popular form of communication over the Internet. Statistics show that instant messaging has overtaken email in popularity. Traditionally, instant messaging has consisted of a desktop client communicating with other clients via an instant messaging service provider. However, instant messaging solutions are starting to become available in the web browser–services like Google Talk, Live Messenger and Facebook. Despite the work done by researchers to secure instant messaging networks, little work has been done to secure instant messaging in the browser. We present secure browser-based instant messaging overlays as a means to enable convenient, secure communication in existing browser-based instant messaging interfaces. Additionally, we present a prototype implementation of the secure messaging overlays and the results of two user studies–the first study focusing on user interest in secure chat and the second being a usability study of the prototype.

Keywords: secure instant messaging, secure chat, Facebook Chat, Private Facebook Chat

ACKNOWLEDGMENTS

Marci's love, patience, and help inspired me throughout the entire thesis process. Dr. Seamons's thoughtful advice and reviews greatly contributed to the concepts on which the thesis is built, as well as to this document itself. Internet Security Research lab provided the foundation and initial ideas on which to build. Brigham Young University - made possible by its owners and donors - made the ideal learning environment.

Thank you, all.

Table of Contents

Chapter 1

Introduction

Instant messaging is a convenient form of communication. Each year the instant messaging user base grows by 200 million. In 2004, a published report showed that the volume of instant messages surpassed email as the preferred communication medium on the Internet. The market has become saturated with various desktop clients to provide easy instant messaging access. Over the last 5 years, instant messaging usage has moved away from desktop clients to web-based clients. In 2005, Google launched its Google Talk platform, which was primarily web-based and integrated with Gmail. In 2010, Microsoft launched a new Windows Live Mail platform, which includes a web-based Live Messenger client. These solutions are convenient as they provide a one-stop shop for email and instant messaging. However, a recent report shows that users are shifting away from the email/instant messaging mash-ups and moving to online social networks as their primary communication platform.[1]

Social networks provide a handy interface for users to peruse through the happenings of others' lives and actively instant message at the same time. According to one survey, most teenagers now only instant message when logged into a social networking website. Although convenient, social networks (i.e., Facebook) have been criticized for their lax attitudes towards the privacy of user data.

[1]Email Statistics Report, 2009-2013, `http://www.radicati.com/?p=3229`

1.1 The Problem

One of the problems with browser-based instant messaging applications is the lack of true end-to-end privacy between users. Although work has been done to create end-to-end privacy for desktop clients, web-based clients remain at the mercy of those hosting the service. There are four components to web-based instant messaging systems: service providers, client proxies, browser clients and the underlying network. Each presents complex privacy issues.

Service providers facilitate the transport of messages from one user to another. A service provider can read every message traversing its system. Google Talk, for example, archives all conversations in Google's Gmail. The text of these archived messages is then mined to provide better AdSense marketing. In addition to reading messages, a service provider can also impersonate a user. For example, suppose Alice works for the messaging services division of Google and Bob is an average user of those messaging services. Alice wants to phish for information from Bob's friends. Because Alice has direct access to the internal workings of the messaging service, she can impersonate Bob and begin conversations with Bob's friends. Bob's friends will not know that it is actually Alice that is talking with them.

Web browsers are a mature technology that provide a window into the vast data repositories and applications of the Internet. All major browsers have been exploited through web sites specifically designed to take advantage of them. A browser also has access to the user data on the machine on which it is running as well as the input given by the user, which could be used for malicious purposes.

Because web browsers communicate with web servers over the stateless HTTP protocol, a client proxy is needed. A client proxy is a program that runs outside of the context of web server; it creates a persistent connection to the service provider. Client proxies are given users' credentials in order to impersonate them. There are many online, browser-based services (i.e. Meebo, ILoveIM, eBuddy, KoolIM) that aggregate and proxy the use of many

different popular instant messaging services. As with the service providers, client proxies can read every instant message it receives.

Even if we could trust the service providers and client proxies to behave properly, most instant messaging infrastructures transport messages in plain text, allowing eavesdroppers to intercept them. Many of the aforementioned client proxy services do not secure traffic between the web browser and the server, making user conversations susceptible to eavesdropping. In a web environment, communication between clients and servers is usually secured via SSL/TLS. However, even if a client could secure its connection to the client proxy or service provider to prevent eavesdropping, the client has no control over the recipient's connection.

1.2 Secure Overlays

We describe the design and implementation of the first system that provides end-to-end security in a browser-based service. The system overlays existing web-based instant messaging systems, is easy to deploy and easy to use. The system prototype will support Facebook; however, the design is extensible to other services. The advantages of a browser-based overlay approach are that users will not have to learn yet another software package, they can continue to use a browser they already understand and trust, and overlays can be layered on top of existing interfaces, removing the need to invent new protocols (e.g., [1], [2], [3], [5], [6], [10], [11]) or install third party client software (i.e. [1], [2]). Additionally, this system introduces a separation of concerns–the browser will only be concerned with sending and receiving messages, the overlays will only be concerned about encrypting/decrypting messages sent and received through the browser, the key server will only be concerned with managing and distributing keys. This separation provides a safety barrier so that no one part of the system can compromise the system as a whole. An attacker would have to compromise multiple components to compromise the entire system. Another novelty of this particular system is its use of an automated key escrow and management system, which will be described in the system description. The key escrow system removes the need for users to establish

shared secrets or obtain public keys in advance by transparently managing encryption keys. Other types of web-based messaging systems (e.g., email) suffer the same problems as instant messaging. The design and implementation we describe can be adapted to other messaging systems.

In addition, we conducted a survey to ascertain user interest and inclination to using secure chat. We also conducted a usability study that demonstrated the system is usable by current Facebook users except for the most novice computer user. The user study revealed several issues with the system. Furthermore, a threat analysis was conducted to identify potential threats to the system in order to determine how well it defends against those threats. The threat analysis identified threats inherent to the general web that affect this system–some that can be mitigated, others that cannot.

Chapter 2

Related Work

2.1 Challenges

Several studies have been conducted to identify the different challenges and vulnerabilities of instant messaging. Casey [4] talks about the risks of instant messaging from a business IT perspective. He outlines some of the high-level actions that can be done to protect businesses from threats that come from instant messaging applications. Hindocha [8], Leavitt [12] and Mannan [18] outline the security issues of instant messaging. Jennings [9] and Mannan [16] survey the different instant messaging protocols available and the available methods for securing instant messaging.

2.2 Off-the-Record Messaging

A popular area of research in secure instant messaging is Off-the-Record Messaging (OTR). OTR allows you to have private conversations over instant messaging by providing encryption, authentication, deniability and perfect forward secrecy. Borisov et al. [3] introduces an Off-the-Record Messaging protocol for secure instant messaging. The protocol uses the Diffie-Hellman key exchange protocol to establish short-term keys that are impossible to re-derive from the long-term key material. These keys are then discarded after a period of use, making any past messages permanently unrecoverable. The messages in this protocol are not digitally signed. It is thus impossible to prove who sent a message. Because of the frequent key exchanges necessary for secure communication, it is vulnerable to replay attacks that allow an attacker to impersonate the sender to any other party in the system.

Raimondo et al. [5] analyzes the key features presented in Borisov et al. [3] and examines security vulnerabilities. They propose a series of change recommendations for Off-the-Record Messaging in an attempt to fix the vulnerabilities. Their recommendations include replacing the authenticated key exchange protocol with stronger exchange protocols. Montminy [19] introduces the On the Record instant messaging protocol, inspired from the work done with Off-the-Record Messaging. On the Record messaging produces a key that allows multiple users to participate in a conversation. It also keeps a running log of all conversations and digital signatures to ensure integrity. Each conversation is recoverable. On-the-Record Messaging requires a change in the client and server technologies.

Other research in Goldberg [7] and Jiang [10] has gone into studying Off-the-Record in group conversations, such as public chat rooms, or other multi-party scenarios. Alexander [1] applies the socialist millionaire's problem to OTR to improve user authentication. OTR has also become publicly available as a Pidgin plug-in. Stedman [23] conducted a usability study of this Pidgin plug-in to determine if it is easy to use and successful at hiding computer security details from the user. He discusses flaws in the user interface that cause confusion and decreased security and further discusses possible solutions to these errors.

2.3 Key Exchange

Outside of Off-the-Record Messaging, other research focuses on the key exchanges to secure instant messaging conversations. Fong [6] presents a new protocol that overlays existing chat protocols. It uses the Diffie-Hellman key exchange protocol to establish a trust relationship with one member of an already existing chat group, who then admits the new user into the group. The protocol uses key rotation to make past messages unreadable to new chat group members. The identity of incoming members is vouched by members that are already part of the chat group.

Kikuchi [11] proposes a protocol based on the Diffie-Hellman key exchange protocol. A user wanting to chat will first register with the server by performing the Diffie-Hellman

key exchange protocol. To initiate a chat, a sending user begins the Diffie-Hellman protocol again; however, the server modifies the message with the session key established during the registration phase between the server and the receiving user. The receiving user then calculates a session key that only the sender and receiver know. This protocol requires special client and server support to perform the key exchange between clients and the server. Although the Diffie-Hellman protocol has been proven to provide a convenient way to establish strong keys between two parties, the paper provides no proof of correctness for the modifications made to the protocol.

Mannan [17] presents the Instant Message Key Exchange (IMKE) protocol, which is a password authentication and key exchange protocol. Unlike the previously mentioned solutions, which use Diffie-Hellman to exchange keys, IMKE specifies a new key exchange protocol. It allows for strong authentication and secure communication in IM systems. It provides authentication (via memorable passwords), confidentiality and message integrity with repudiation. IMKE cannot be layered on top of existing IM systems without modifications to both client and server technologies.

2.4 Other Systems

There are other systems that provide secure messaging capabilities (Bardis [2], Lee [13], Loesing [14], Lux [15], Nicholson [20] and Zhang [26]). There are even commercial products available. For example, Windows Live Communication Server is a software suite meant for large businesses that provides secure instant messaging. The issue with this product is that it is meant for internal communication in a controlled environment, whereas the power of instant messaging is in communicating with anyone, anywhere. Trillian is a software product that provides a security feature that allows for secure communications. Trillian's implementation uses keys that are too small and could easily be brute-forced. Google Talk, a browser-based solution, provides an off-the-record mode (not to be confused with Borisov [3]) that disables the storage of messages on Google servers.

Chapter 3

User Survey

3.1 Description

We conducted a three-part survey to determine user awareness and attitudes regarding the privacy of instant messaging. Part one gathers information about the chat systems users are currently using. Part two seeks to determine how users feel about transmitting sensitive information over chat. Finally, part three gathers information about opinions on the privacy of chat. The survey was distributed via word-of-mouth, email, and various social networks and resulted in 65 responses.

3.2 Chat Systems

Table 3.1: Chat system usage

System	Percent of Users
Google Talk	67%
Facebook Chat	50%
Skype	41%
Windows Live Messenger	19%
Yahoo Instant Messenger	19%
AIM	12%
IRC	3%
ICQ	2%
Others	19%

Table 3.1 lists chat systems users reported to have used regularly. The survey allowed for users to select more than one option. Most users reported that their personal preference

in chat systems is based on how easy it is to learn and use. Multiple users also said that they use a particular chat system because their work mandates it or it integrates with other online services they use regularly, such as email or social networks. In this survey Google Talk, Facebook Chat and Skype are the predominant chat platforms. Google Talk and Facebook are used primarily in the browser. The users that selected the *Others* option reported they used commercial products, like Microsoft Lync, Microsoft Communicator, and IBM Sametime, regularly.

Figure 3.1: Frequency of use

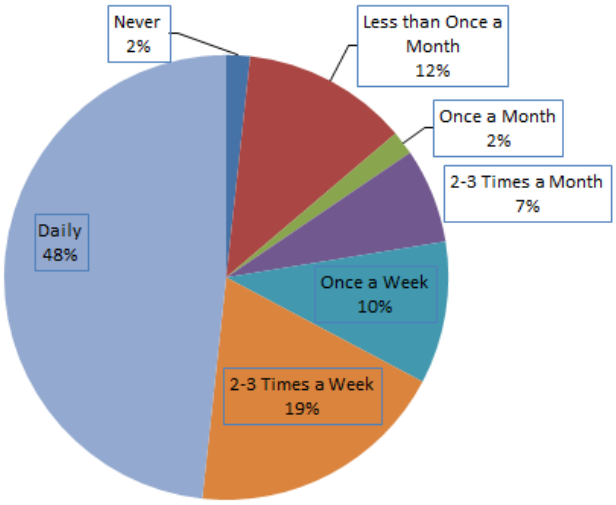


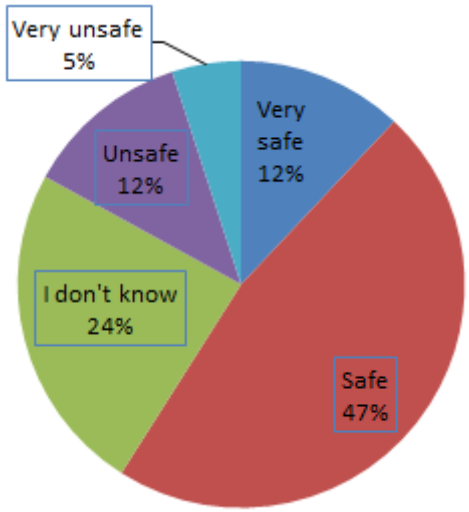
Figure 3.1 shows how frequently respondents use chat to communicate. Seventy-seven percent of respondents indicated they use chat at least once a week, with the majority of them using it multiple times a week. These users predominantly use Google Talk, Skype, Facebook Chat, and commercial products.

3.3 Transmitted Information

In this portion of the survey we asked questions to gauge how safe users feel sending information through a chat system. According to Bruce Schneier [21], there are two types of safety: the emotional feeling of safe and the actual reality of safe. He posits that most people

rely on the emotional measure for how safe they are. We assume that most respondents used emotional safety as their measure. Figure 3.2 shows the user responses.

Figure 3.2: How safe users feel using chat



We then compared those answers with what types of information respondents send via chat. We provided the users with a list of sensitivities ranging from non-sensitive to sensitive and asked them to select the options that describe the kinds of information they have ever sent via chat in both a personal and business setting (see Appendix A, question #7). Table 3.2 provides a summary of these results. Fifty-nine percent (Group 1) of respondents reported they feel safe or very safe when using chat, 24% (Group 2) admitted admitted they have never thought about how safe they felt and were uncertain and 17% (Group 3) reported they feel unsafe when they chat.

Of those in Group 1, 84% indicated they send non-sensitive personal information and 44% indicated they send moderately sensitive personal information. Fifty-six percent send non-sensitive business information and 24% send moderately sensitive business information. Fifteen percent say they send sensitive personal information (e.g., social security number, credit card, bank information). Those who send sensitive personal information report they use Google Talk, Skype and Facebook Chat to do so. This is of particular interest because Google Talk and Facebook Chat do not enforce secure connections to communicate via their

Table 3.2: Type of information sent over chat

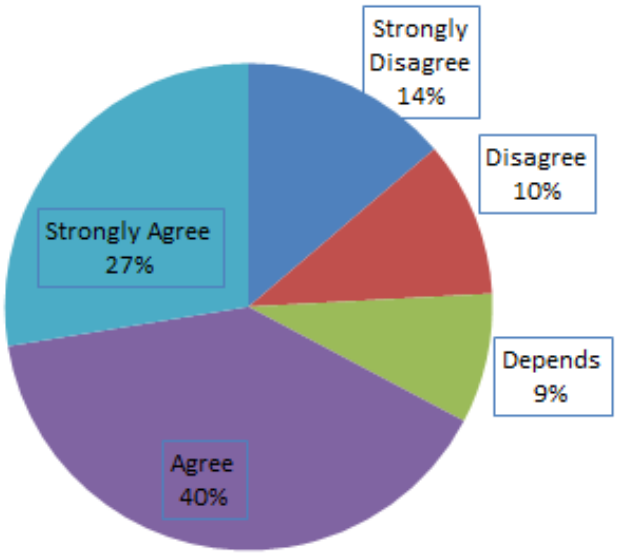
Group	Type of information	Percent
Group 1 (59%)	Non-sensitive personal info	84%
	Moderately sensitive personal info	44%
	Non-sensitive business info	56%
	Moderately sensitive business info	24%
Group 2 (24%)	Non-sensitive personal info	93%
	Moderately sensitive personal info	36%
	Any business info	<15%
Group 3 (17%)	Non-sensitive personal info	90%
	Moderately sensitive personal info	40%
	Non-sensitive business info	50%
	Moderately sensitive business info	30%

chat systems. With Google Talk, secure connections are available when connecting through Gmail, Google+ or third party software, but that does not guarantee the other party is connected securely. Facebook defaults to unencrypted HTTP connections for most of their online services. There is a greater risk of an eavesdropper seeing sensitive information sent through the Google Talk or Facebook Chat networks.

Those in groups 2 and 3 reported to have never sent highly sensitive information of any kind over chat. It is interesting that even though those in group 3 indicated that they felt unsafe or very unsafe about the security of chat, most of them still reported using chat systems on a daily basis.

The feeling of safety that a user has is not only affected by the system they are using but also the context in which it is being used. For example, a person is more likely to feel safe about investing money in a stable commodity like gold rather than in the stock market, which can fluctuate. The same can be said with sending sensitive information–a person is more likely to feel safe about sending sensitive information to an entity he or she trusts. We asked a set of questions to ascertain how users would react to communicating sensitive information to their trusted friends or family members.

Figure 3.3: More likely to send sensitive information to trusted friend or family member

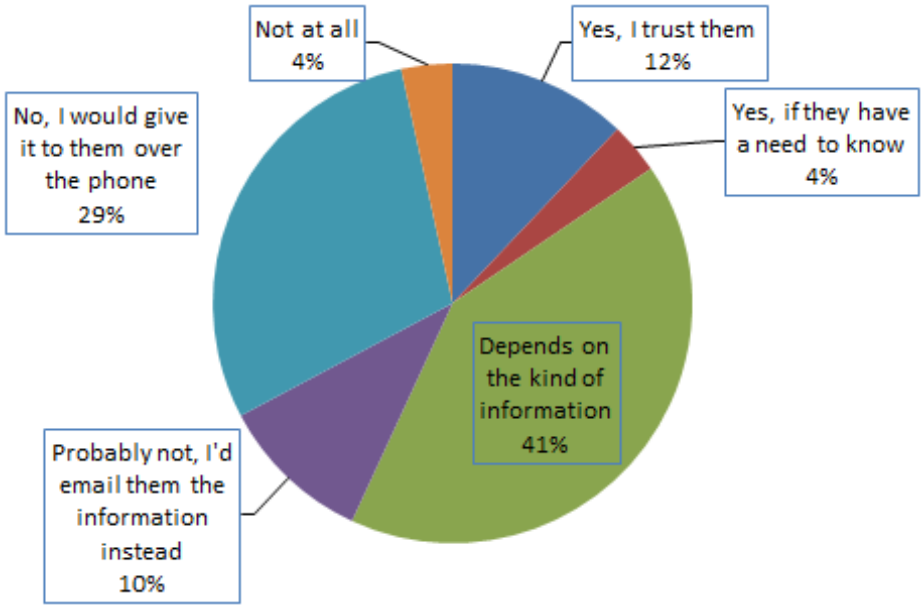


When asked if they were more likely to send sensitive information to a trusted friend or family member (see Figure 3.3), 68% answered in the affirmative, 24% answered negatively, and 9% answered that it would depend on the kind of information. Most of the 9% who answered *Depends* identified information relevance and security of the communication medium as their deciding factors. However, one respondent was more concerned with the speed of the medium rather than the security of it.

When asked if they would reply over chat with sensitive information requested by a trusted friend of family member (see Figure 3.4), 41% answered that it would depend on the kind of information, 29% answered that they would use the phone instead, 12% answered yes, 10% answered they would use email, 3% answered yes if there was a need to know and the remaining 5% responded no.

Users' responses in this part of the survey have been positive towards chat systems. Most feel safe chatting over their service of choice. This could be because their feeling of safety has never been challenged. It is apparent that users seem to have preconceived notions about which communication mediums are "safe" and which are not. The respondents mentioned email and phone systems as more secure alternatives to chat. However, email

Figure 3.4: Would reply with sensitive information to trusted friend or family member



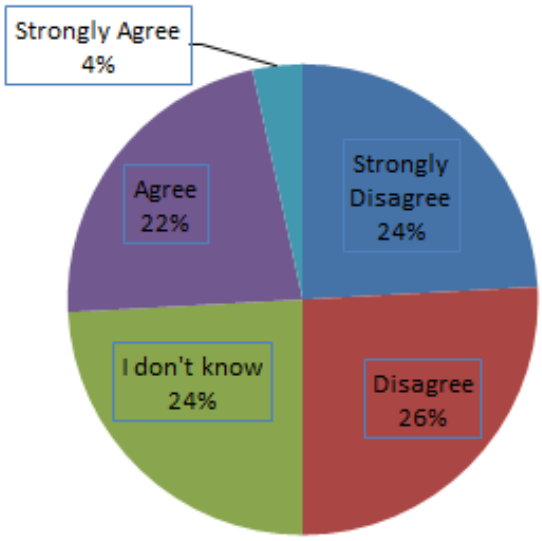
suffers from many of the same vulnerabilities as chat and the phone system has a long history of compromises [25]. The results show that users measure security by what they feel most safe using.

3.4 Privacy

Some services, such as Google Talk, offer the ability for users to connect to the service using an HTTPS connection. HTTPS allows data to be transmitted in encrypted form, preventing eavesdroppers from reading the data. While this transport security is important, it does not offer full privacy because each communique is stored unencrypted and mined for data by Google. In addition, an HTTPS connection with Google does not guarantee that the other chat party is connected to Google with an HTTPS connection. We asked a set of questions to gauge the respondent's awareness of privacy and security issues.

There was a lack of agreement among users when asked if they were confident that their chat conversations were private (see Figure 3.5). Approximately 50% disagreed or strongly disagreed that chat conversations were private, approximately 25% did not know

Figure 3.5: Question: I'm confident that my chat conversations are private



and the remaining 25% agreed or strongly agreed. These results suggest that many users lack an understanding of privacy. Those who agreed or strongly agreed were also in the group that indicated they felt safe or very safe when using a chat system. Additionally, 71% of those that answered *I don't know* indicated they felt safe or very safe when using a chat system. We further tried to assess respondents' inclination toward verifying the identity of the person with whom they are chatting beyond the facilities provided by the chat provider. Only 5% of the respondents did not feel confident in the identity of the opposite party. This suggests a trust relationship that is ripe for exploit if an attacker is able to compromise a user's account.

We then asked two questions to ascertain the respondents' awareness and concern about what chat providers do with their conversations after having sent and stored them. Fifty-seven percent of respondents showed concern that chat providers may mine the text of their chat conversations to provide better targeted advertising. The remaining 43% were either indifferent or not concerned. Forty-nine percent of respondents showed concern that chat providers permanently store their messages while 51% were indifferent or not concerned.

The final question attempted to assess how responsive users might be if their standard way of chatting was found to be vulnerable. We specifically asked about the chat client.

Fifty-five percent answered affirmatively that they would be inclined to move to another client, 29% preferred not to move but rather that the client be fixed. The remaining 16% were indifferent.

Chapter 4

Design and Architecture

This chapter describes the design goals and architecture of a secure browser-based instant messaging overlay system. The basic problem addressed by the architecture is overlaying security on top of existing interfaces to support end-to-end encryption. We also address the challenges of key management and software deployment.

4.1 Goals

4.1.1 Usability

The primary goal of the secure browser-based instant messaging system is that it must be easy to use. In previous secure communication systems, usability issues have typically been the barrier preventing widespread adoption [22][24]. The user should not have to learn new instant messaging software. We want the user to use the software and instant messaging services they are already familiar with. The user should only need to focus on composing and sending a message, whilst all other details are handled as transparently as possible in the background. The new software we require will extend the software already familiar to the user, be easy to obtain, and not add unnecessary complexity. Furthermore, this software will provide secure communication on an as-needed basis. The user will be able to select which chat sessions are secure to minimize the need for others to install the software.

4.1.2 Bootstrapping

We designed the system to spread incrementally in a grass roots fashion. Suppose Alice wants to chat securely with Bob. She does not need to contact Bob in advance to walk him through the software install. Instead, she simply indicates to her chat client to chat securely with Bob. The system sends Bob a chat message indicating that Alice wants to chat securely and helps him install the software in a matter of a few minutes. We refer to this process as bootstrapping.

Bootstrapping is the process by which a receiving user can acquire the necessary software to chat securely while the sender waits. The recipient has to obtain the software by following short instructions provided in a standard greeting. Once obtained the software will launch, authenticate the recipient and allow the secure chat to continue.

Because instant messages have short life spans, the process of obtaining the software and authenticating to the key management system needs to be quick and efficient. To help the user obtain the software easily and securely, the system will direct the new user, via a URL given in the standard greeting, to a site that will provide a small number of simple instructions and other visual aids to help the new user use the software. Our goal is to design a process with 2 or 3 simple steps that any user can follow. These steps include deployment and usage of the software. To authenticate a user, the system will use the built-in mechanisms of the service provider to avoid new account provisioning. For instance, Facebook provides an OAuth endpoint and Google exposes OpenID, both of which are open authentication protocols and can be used by applications created outside of the aforementioned organizations.

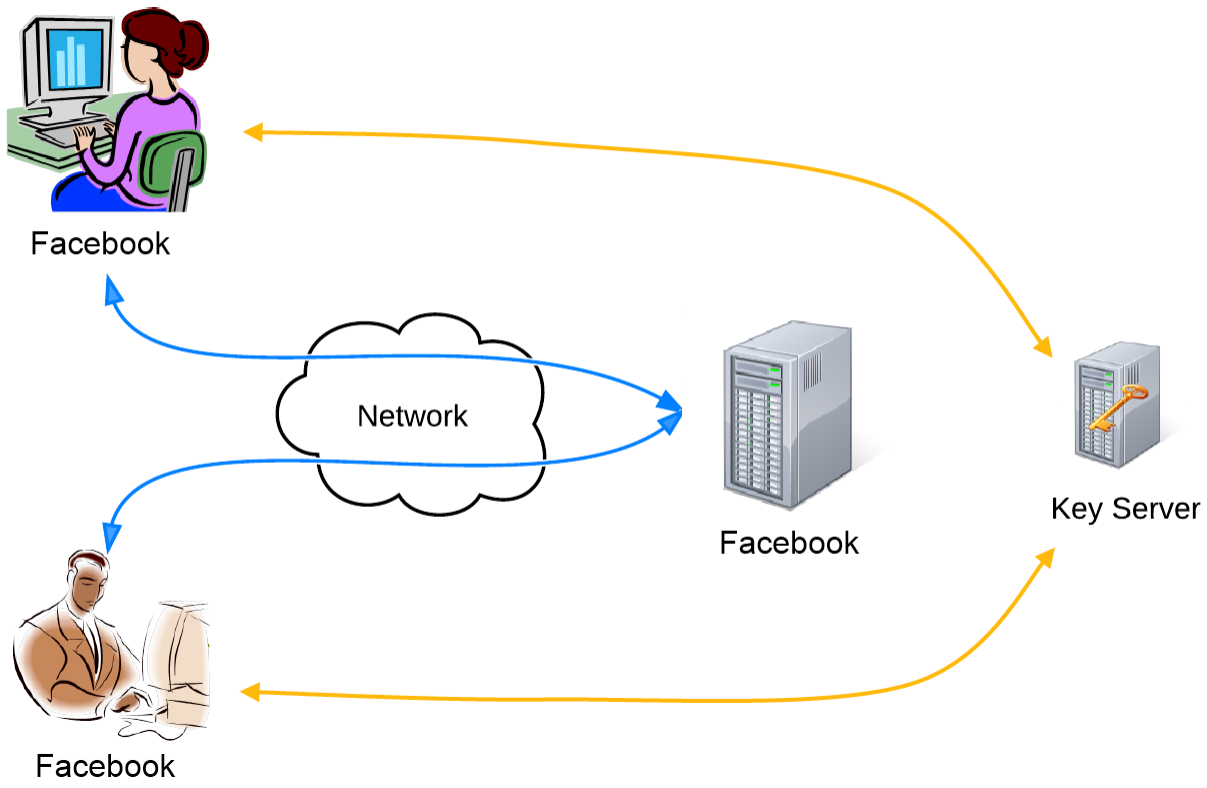
4.1.3 Confidentiality

Our goal is to provide message confidentiality through end-to-end encryption and make the underlying details and setup as transparent to the user as possible. We want to make conversations unreadable by service providers and client proxies, including archived conversations

stored on the service provider's servers. A user should be able to recover archived confidential conversations at any time.

4.2 Architecture

Figure 4.1: System architecture



4.2.1 Key Management

To address the confidentiality goal discussed previously, the secure browser-based instant messaging system incorporates a key escrow system. The purpose of the key escrow is to hide the burdensome, albeit important, key management details. It relies on a trusted server to automatically manage keys for the user. This automated key management system permits two users to establish a secure chat session without prior key negotiation. We provide confidentiality using end-to-end encryption.

4.2.2 Client Overlays

The overlay system is the software that the user acquires during the bootstrapping process. An overlay is a frame that rests directly on top of another frame. Its purpose is to hide parts of the original window to prevent the user from interacting with it. The user interacts with the overlay while the overlay interacts with the obscured parts of the original window on behalf of the user. An overlay provides the security that the original window does not. The window content in the secure overlay is served from a domain that differs from the original window. Because of modern browser same origin security policies, the difference in domain prevents content in the original window from accessing content in the overlay.

The user interface of the overlay reflects many of the same characteristics as that of the window it is obscuring from the user. The purpose is to make the user interface already familiar to the user and make the learning process easier. However, there are distinct differences so that the user can distinguish the difference between the parts of original window that need to be secured and the secure overlays.

4.2.3 Service Provider

In web-based instant messaging systems, a service provider is an entity that relays information between users. In our case, a service provider is responsible for relaying instant messages between users. The service provider may also store messages for future retrieval as a matter of convenience for the user. Some service providers support secure connections from the clients (e.g., SSL/TLS) to prevent eavesdroppers from reading messages sent across the Internet. However, most service providers do not force recipients to connect securely.

Service providers are also identity providers. The secure overlay system uses the service provider to determine the identity of the user so that the automated key management system can generate the appropriate keys to encrypt and decrypt messages.

4.2.4 Client Proxy

The client proxy maintains a persistent connection with the service provider in behalf of the user. This is necessary because of the stateless nature of the HTTP protocol that browsers use. The proxy is given credentials that allow it to impersonate the user to the service provider. This allows the proxy to send and receive messages in behalf of the user in a stateful manner. Frequently, the client proxy and service provider are the same entity. For example, Facebook Chat and Google Talk are both available to use in the browser through Facebook and Gmail respectively, thus Facebook and Google are both the service provider and client proxy in this instance. Meebo, however, provides access to the previously mentioned instant messaging services via the browser, but as a separate entity from the service provider. In this instance, Meebo is just the client proxy.

4.3 Threat Model

In general, service providers are capable of reading every message sent through their system. A malicious service provider could impersonate any of its users in a chat conversation. A malicious service provider that is also an identity provider could impersonate a user and authenticate with the automated key management system. This would allow the service provider to acquire the necessary keys to encrypt and decrypt messages as that user. In this design, we assume that a service provider is an honest-but-curious adversary–meaning, they may only try to read messages for purposes such as delivering targeted service and product offerings, but not for malicious intent. A service provider will not impersonate a user to the key management system for the purpose of acquiring the users' keys.

Similar to service providers, client proxies can read every message the user sends and receives. A malicious proxy could attempt to compose and send messages as if the user it was impersonating was sending them. Because we give our service provider credentials to the client proxy, a malicious proxy could also impersonate a user to the key management system and acquire the keys associated to that user. In this design, we will assume that a

client proxy may only try to eavesdrop on conversations. It will not attempt to impersonate a user to the key management system.

Outside of service providers and client proxies, we also assume that some users may have an insecure connection with their service provider allowing eavesdroppers to see conversations.

Chapter 5

Implementation

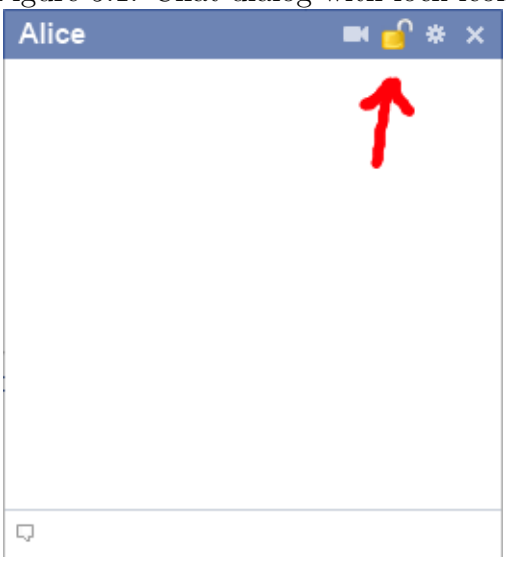
The chapter discusses a secure browser-based instant messaging implementation focusing on Facebook as the service provider. Here after, this implementation will be referred to as Private Facebook Chat (PFC). We then provide an in-depth discussion of the client software and include a walk-through of the system from a user's perspective. We finish with a summary of how the key server works.

In this chapter we refer to the Kiwi system, a collection of authentication and key distribution components, and the Secure Messaging Platform (SMP), a collection of secure packaging components, both of which are being developed in the Internet Security Research Lab at Brigham Young University. PFC directly uses or adapts components from Kiwi and SMP.

5.1 Walk-through

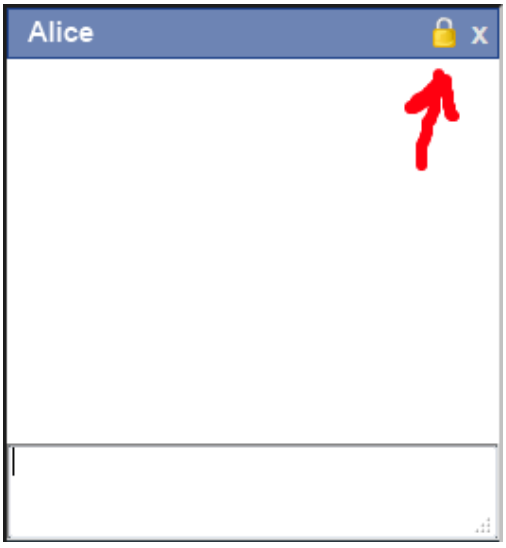
This section provides a walk-through of the system from the perspective of the user. To provide context, we will use this system through two fictitious characters: Alice and Bob. Alice wants to chat securely with Bob through Facebook Chat. She is aware of the PFC messaging system and already has the bookmarklet installed in her browser. She opens a Facebook Chat dialog to Bob and clicks on the bookmarklet to enable the secure overlay system. Upon doing so, an open lock icon appears (see Figure 5.1).

Figure 5.1: Chat dialog with lock icon



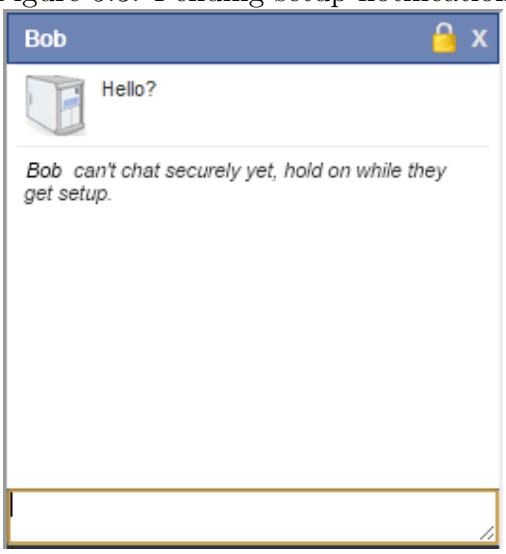
She clicks on the open lock icon to enable secure chat. This overlays the Facebook Chat dialog with a PFC secure overlay. The lock icon is changed to a closed state to signify to Alice that the session is secure (see Figure 5.2).

Figure 5.2: Secure chat window with closed lock



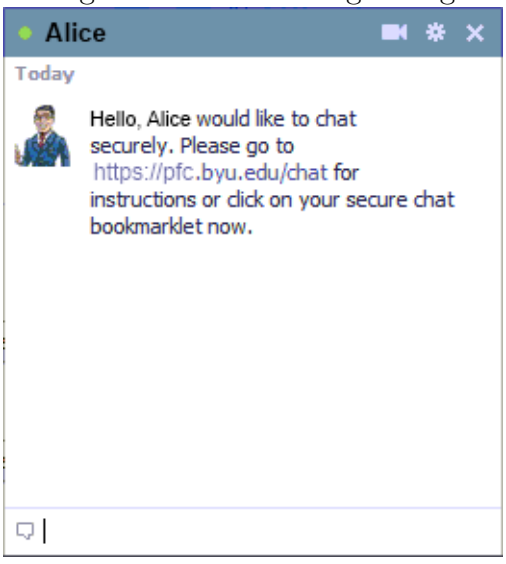
Alice now has to wait for Bob to enter secure chat before she can send a message to Bob. If Alice tries to send a message before Bob is ready, the overlays system will inform her that Bob is not yet ready to receive secure messages (see Figure 5.3).

Figure 5.3: Pending setup notification



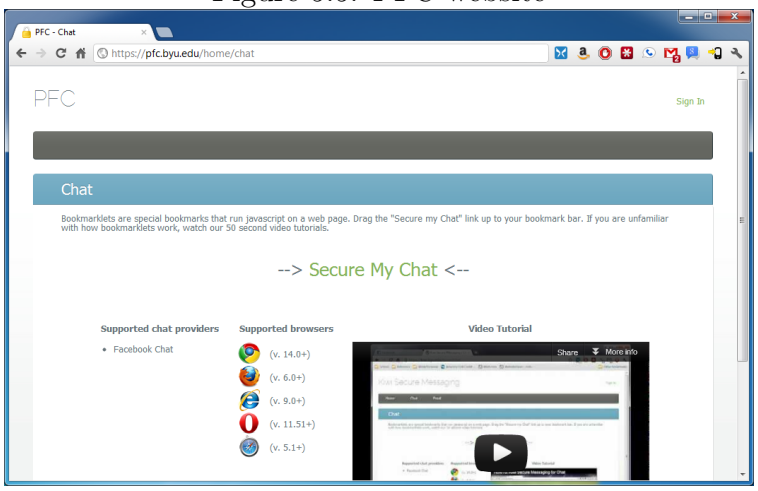
If Bob is on Facebook, a chat dialog from Alice pops up with a message stating that Alice would like to chat securely (see Figure 5.4). The message gives instructions to Bob on how he can successfully chat securely with Alice. Since Bob does not have the PFC secure overlay system he navigates to the PFC website for further instructions (see Figure 5.5).

Figure 5.4: Standard greeting



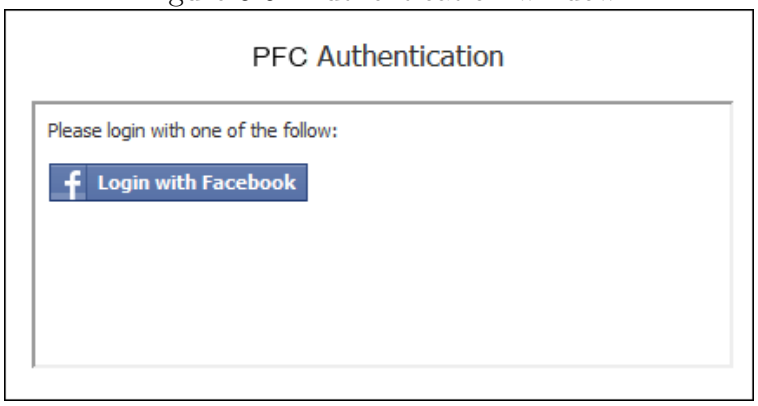
After reading the instructions and watching the video tutorial on the PFC website, Bob successfully installs the secure chat bookmarklet by dragging it to his browser's bookmark bar. He then goes back to Facebook where the chat session with Alice resides and clicks on the

Figure 5.5: PFC website



bookmarklet to secure the chat session in his browser. Alice and Bob are then both presented with a PFC authentication dialog (Figure 5.6). The dialog asks them to authenticate with Facebook.

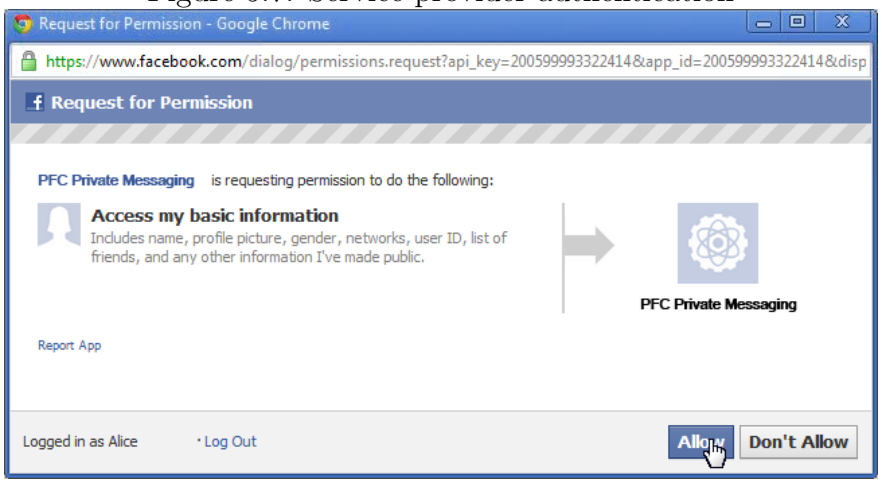
Figure 5.6: Authentication window



When they click on the *Login with Facebook* button, they are presented with a dialog from Facebook asking them if it is okay that the PFC Secure Messaging Facebook application access basic user information from their Facebook user profile (see Figure 5.7).

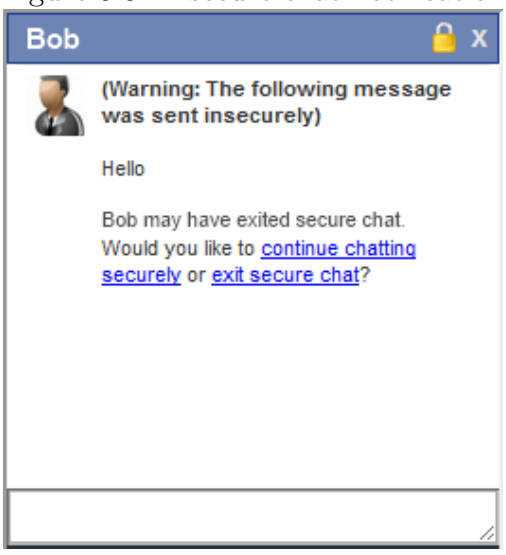
Upon choosing *Allow*, both Alice and Bob will receive a ready signal. Alice and Bob can now begin chatting securely via Facebook Chat. Suppose now that during the conversation that Bob closes his secure chat overlay but still continues to send messages.

Figure 5.7: Service provider authentication



Alice is presented with a message to help her either exit secure chat or help Bob to enter secure chat again (see Figure 5.8).

Figure 5.8: Insecure chat notification



If Alice chooses to exit secure chat, her secure overlay will close and she will again be interacting directly with the Facebook Chat dialog. However, if she chooses to continue chatting securely, the secure overlay in Bob's browser will reopen and he will be reintegrated into the secure chat session. At any time during a secure chat session, Alice and Bob can exit a secure chat by clicking on the closed lock icon in the corner of the overlay dialog. Now

that Bob has the secure chat bookmarklet installed, if at any future time Alice wants to chat securely, all he has to do is click on the secure chat bookmarklet to enter secure chat.

5.2 The Client

We now introduce the components of the client. We also introduce the PFC website which is where the client software can be obtained. The client is composed of many components: the bookmarklet, in-page services, key management system, and window services. We will describe each of these components.

5.2.1 PFC Website

The PCF website (`https://pfc.byu.edu/home/chat`) is the launch point for all new users of the secure overlay system. The goal is to make the website very simple so that any user can learn and start using the software. We provide a short textual introduction that refers the reader to a short video tutorial that shows how to install the bookmarklet. The video also shows the bookmarklet in action.

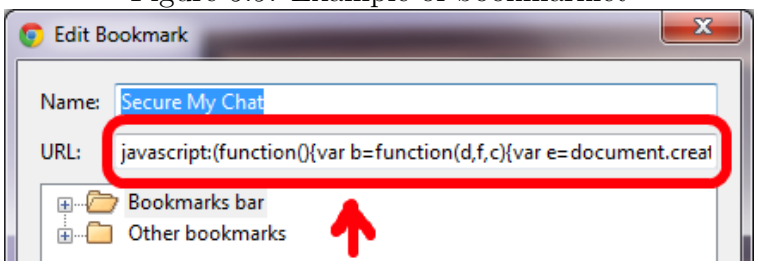
The large *Secure My Chat* link (see Figure 5.5), when bookmarked, becomes the bookmarklet. The bookmarklet is generated by a PFC web service. As we study the usability of the secure overlay system, the website may change to reflect the needs of the user.

5.2.2 Bookmarklet

A bookmarklet is a browser bookmark. The only difference is that a bookmarklet runs JavaScript in the browser window rather than navigating to a new location. JavaScript is a prototype-based, dynamic scripting language that supports object-oriented, imperative and functional programming styles. It is used predominantly in a web browser to provide enhanced user interfaces and dynamic websites.

Because the bookmarklet is just JavaScript, we can reach a greater audience because all major browsers support JavaScript. In addition, the user does not have to be an administrator

Figure 5.9: Example of bookmarklet



to install a bookmarklet since it is just a bookmark. It is also easy and highly usable because the "install" is the same as adding any other bookmark or favorite in the browser. The bookmarklet's sole responsibility is to download the latest version of the PFC overlay system and inject it on to the page. This allows software updates to be instantly available. The only stipulation of securing content in this manner is that all elements that need to be overlayed have to be accessible from the main window in JavaScript. This is because of the same origin policies browsers impose upon JavaScript code. Same origin policies permit scripts running on pages originating from the same site to access each other's methods and properties with no specific restrictions, but prevents access to most methods and properties across pages or frames on different sites.

An alternative to a bookmarklet would be a browser plugin or extension. Most modern browsers have the facilities to allow its base functionality to be extended in some way. The advantage of having a plugin is that the code is not bound by the same origin policies. This means it can access any content in any frame. One of the disadvantages of extensions is that browsers have their own unique extension platform. A different plugin would have to be built for each browser. The methods of deployment would also vary by browser and operating system.

A hybrid approach could also be taken where the plugin solely replaces the bookmarklet in its bootstrapping functions. In cases where frame content that needs to be overlayed is served from a location different from the main window, the plugin would be able to inject

the necessary software. The plugin would be very small, but still be able to overcome some of the deficiencies of the bookmarklet.

5.2.3 In-Page Services

The set of components that execute within the context of the main window are called the in-page services. Together they scan the page for elements that need to be overlaid, create and manage the overlays, and send and receive messages for and in behalf of the overlays. We will describe each component.

The controller is the kernel of the software. It is the entry point where execution begins. The controller instantiates all the other services. For a few of the service components, the controller acts as a communication medium through which messages and events are passed to other service components. The controller also updates the graphical elements in the main page to indicate areas where security overlays are possible. These areas are usually so indicated by a lock icon (see Figure 5.1).

Each window and overlay is controlled by a window manager. The controller interacts with each window through its associated window manager. Conversely, each window interacts with the page through the window manager. We will describe the services these window managers help provide in the next section.

Each service provider has unique web pages that provide their instant messaging services. The component that helps generalize these disparate systems and provide a common interface to access them is the page scanner. The page scanner notifies the controller when changes have occurred on the page that need to be addressed. For example, when a new chat window pops open, the page scanner needs to notify the controller so that a lock icon can be inserted and an overlay can be created if the dialog contains a PFC package. A PFC package is the actual ciphertext, as well as additional information needed to decrypt the message. It is often represented as base-64 encoded text. In the context of the prototype the terms "secure content" and "PFC package" are interchangeable.

5.2.4 Window Services

Window services comprise of the components that run in windows served from a domain that is different from the main window. This prevents the main window accessing the content or local storage of that domain. The first window created by the controller is the key manager window. The key manager window is invisible to the user. It is responsible for requesting and storing keys. All requests for keys are made on demand as old or new messages require them. All keys are cached in the browser's local storage, only accessible in pages served from the same domain as that which stored the data.

The key manager window also interacts with the authentication server through the authentication manager and drives the process that authenticates the user to the key server. When a user requests a key that does not exist in local storage, the key manager attempts to find a valid authentication token with which it can communicate with the key server. If a valid authentication token is not found, the user is asked to authenticate. Figure 5.6 is an example of the authentication dialog.

For the purposes of allowing this system to authenticate with Facebook, we had to register with Facebook as a third party desiring to use their authentication system. When a user clicks on the login button and it is their first time using the secure browser-based instant messaging system, they are presented with a Facebook specific login and authorization page. Figure 5.7 shows an example of the authorization page. All service providers will have similar pages.

Third party applications desiring access to users' information from a service provider have traditionally suffered from user trust issues. Although we only require a user's unique identifier, the application authorization page makes it look as though we are using more of their information than we actually are. This may be a red flag to those users that are generally cautious and could be a point of confusion for other users. The usability study will explore this in further detail.

After successful authentication, the authentication token is cached in the browser's local storage–the same storage used by the keys. The authentication tokens are reused until they expire, after which a new authentication token must be acquired.

The chat window is the window that the user interacts with securely. It is responsible for decrypting and displaying all incoming messages. It is also responsible for encrypting any outgoing messages. The chat window can be distinguished by a closed lock icon (see Figure 5.2). The chat window requests keys on demand which, in turn, drives the key and authentication managers into action. The chat window provides a few security features to help the user chat securely. The first feature prevents the user from chatting until the other party has successfully installed and opened the secure overlay. Figure 5.3 shows an example of the notification a user receives. We do this to prevent a user from prematurely sending a secure message, displaying cipher text to the other party.

The next feature helps a user distinguish secure and insecure messages sent by the other party. If the other party exits a secure chat for any reason, the user will be notified and is presented with some actionable items that will help them decide if they want to stay in secure chat or not. Figure 5.8 gives an example of this notification.

5.2.5 Messages

One of the challenges with instant messages is that unlike email, messages have a limited length. Facebook, for example, limits each message to 1000 characters. When messages are encrypted and encoded for transport, the resulting cipher text and other necessary meta data is much larger than the original message. The generic SMP package allows for multiple viewers, data integrity, timestamps, key server specification, message initialization vectors and nonces. A small message with all this extra meta data would be well over the size limitation.

To address the size limitation, we customized the package for instant messages. We make the assumption that chat messages have one sender and one receiver. As a result, we do not store any meta data about the viewer. We remove the MACs (message authentication

code) because they are simply too large to include. We make the assumption that if the message is tampered with in transit then the decryption operation will most likely fail revealing a corrupt package. We remove the timestamps and rely on the instant messaging service provider to provide that information. Facebook, for example, provides both the time a message was sent from the client and when it was received. Since the message initialization vector can be large, we omit it from the package and have a predefined value we use for all packages. The purpose of an initialization vector is add to the randomness of the cipher text, however the nonce, which we do include, performs that same function and is smaller. We include the key server URI since the key server that generated the creator key needs to be the same server that generates the viewer key. The user must shorten all messages that do not meet the size requirements.

Figure 5.10 gives an example of a PFC package for instant messaging.

Figure 5.10: Example PFC package


```
{
  "EncryptedMessage":
  {
    "Value":"/cZ3Nt10+pvgz+cHOJHzeYW0kXU7YpgdB76f2SxbUmntQ77nXPcK7vb4y6zaC0gw"
  },
  "Attributes":[{"Key":"KsUri|Nonce","Value":"https://kiwi.byu.edu/ks/|2kWVlFQOb18="}]
}
```


5.3 The Key Server

The key server is a key escrow system that automatically manages the keys for each user. The key server creates and distributes keys to the user that they use to secure their messages. With key escrow, users will not actually own or permanently store the keys they use. User will never have to worry about losing their keys and making previous messages unrecoverable. Each key server is equipped with a master key that is used to derive all other keys.

The key server provides a RESTful interface from which users can request keys. REST is a simple means of communicating between distributed systems using HTTP. A client can request two types of keys: a creator key and a creator-viewer key. Figure 5.11 is an example the key request message and Figure 5.12 is an example of a key response.

Figure 5.11: Example key request

```
{
  "AuthToken": "<truncated>",
  "CreatorId": "12378182932",
  "ViewerId": "20938429348",
  "Timestamp": "/Date(123456789)/",
  "KeySize": 32,
  "KeyType": 1
}
```

Figure 5.12: Example key response

```
{
  "{\"Start\":\"/Date(1320200351243)/\"}" :
    "GpsBp1D6WrQQXWICpvEmacZsj1NCvIXiOzbRiUoCMv4="
}
```

Each key has an expiration date that defines a key period. We expire keys to mitigate the effects of a compromised key. Our implementation sets the key period to be 1 month; however, this can be easily changed. Once a key expires, users must acquire a new key from the server.

The creator and creator-viewer keys each have their own unique function. The creator key is constructed on-demand with a hashed key derivation function (HKDF) using the master key, sender ID and key period as input. The sender derives a creator-viewer key to encrypt a message for a specific recipient. The recipient obtains the creator-viewer key from the key server in order to decrypt the message. The creator-viewer key is derived by the key server or a sender with the aforementioned HKDF using the creator key, recipient ID and key

period as inputs. The creator-viewer key is like a session key that links exactly one sender to one receiver for the duration of the key period.

To fulfill key requests, the key server must authenticate the user making the request. The key server communicates with an authentication server to verify the identity of a user. Because we use the identity mechanism of the service provider, the authentication server only redirects the user to the appropriate identity provider and verifies the validity of the token given to the user by the provider. The authentication server does not provide independent identity assertions. The token returned by the service provider's identity mechanism becomes the part of the authentication token the key server uses to authorize a key request. For example, figure 5.13 shows an example of an authentication token that is sent to the key server. A user authenticated with Facebook OAuth would fill in the *Value* attribute with the contents of the OAuth token.

Figure 5.13: Example authentication token

```
{
  "UserId":"100003145207526",
  "Period": { "End":"/Date(1320811200000)/" },
  "Value":"....",
  "IsTokenComplete":true,
  "AuthServerUri":"https://kiwi.byu.edu/auth/"
}
```

An authentication token has an expiration date specified by the identity provider with whom the user authenticated. Validating these tokens requires that the key server support OAuth. OAuth provides an interesting challenge because the tokens themselves cannot be validated. The only way to validate an OAuth token is to access the resource it gives access to. A valid token will grant access to resources.

In our secure instant messaging implementation for Facebook, we use Facebook's OAuth token to get the unique identifier that Facebook assigns to each user. We use these identifiers as the sender and receiver identities for key derivation.

Chapter 6

Usability Study

A usability study of PFC was conducted involving 17 experienced Facebook Chat users (59% male, 41% female). The participants comprised BYU students and employees from a local software company. Eight participants (47%) were students and had no technical background. Of the nine participants from the local software company, three of them (33%) had no technical background.

The goal of the usability study was to measure the usability of the PFC messaging system. The study consisted of 5 tasks designed to exercise a specific system feature along with some survey questions. The users had no advance notice that the study would focus on the security of Facebook Chat. We created 4 dummy Facebook accounts for participants to use during the study to help segregate tasks and to pose no privacy risks to the users. Each account had a friend named Steve that the participants would chat with for testing purposes. The average completion for each user was approximately 25 minutes.

6.1 Task #1

The purpose of this task is to the determine the usability of the bootstrapping process — can a participant who receives a request to chat securely successfully obtain the software and launch a secure chat session. We want the participants to be able to onboard successfully without out-of-band help. The task informs the participant that Steve (a fictional character devised to make the scenarios more believable) needs to send them some passwords and that they must login to Facebook and wait for him to contact them. We purposefully leave out

any references that imply that this must be done securely because we want to measure the effectiveness of our bootstrapping prompts with a user with no advanced notice. This task is the most difficult because this is the first time the participants encounter the secure instant messaging software and are asked to use it without human help.

In this study, the bootstrapping process begins with a standard greeting (see Figure 5.4). The greeting invites the user to the PFC website site where they are given further instruction on how to chat securely. This greeting is also used by the PFC system to trigger the overlay processes. We want to measure how successful the greeting is in getting individuals to the PFC website. We also want to measure how effective the PFC website is in instructing the user how to install and use the bookmarklet.

One of the assumptions we made in designing this task was that users would trust links being sent to them in Facebook Chat because there is already a pre-established trust relationship. We found that only 2 participants (12%) trusted links sent to them via Facebook Chat. The remaining participants either did not trust links entirely or scrutinized the sender very carefully before clicking on the link. For the purposes of this task, all but one participant clicked on the link to the PFC website. The participant who did not follow the link indicated that they do not click on links sent from people they do not know extremely well in real life. That participant did not complete the first task.

The PFC website was generally successful in helping the participants learn to install and use the bookmarklet. We found that participants' attention was drawn to the bookmarklet link before they read the short instructions or watched the video tutorial. Slightly less than half of the participants tried installing or using the bookmarklet without instruction. Six participants (33%) attempted to directly click the link to enable secure chat. Others tried dragging the link into the address bar. The term *bookmarklet* seemed to confuse a few participants. We observed them being unsure how to create the browser bookmark. Eventually these participants abandoned their efforts and proceeded to read the provided instructions and watch the video tutorial.

We observed interesting behavior during the authentication process. To use Facebook authentication, we were required by Facebook to create a Facebook application, which we call PFC Secure Messaging. Users must explicitly allow this Facebook application for authentication through Facebook to our PFC site to function properly. At least 50% of participants were very wary of the PFC Secure Messaging Facebook application we were asking them to allow. Some participants tried to continue the task without allowing that application. Thirteen users (76%) reported that they normally do not allow any Facebook applications.

Excluding the previously mentioned participant that did not complete this task, all except two participants completed this task successfully. The two that didn't complete the task were non-technical and older. They use technology on a limited basis and only feel comfortable doing things they have been repeatedly instructed to do. They use Facebook and Facebook Chat solely to stay in touch with family and maybe friends. They are unable to learn new technology without a human guiding them through it. The secure browser-based instant messaging system, as it is currently designed, cannot target users such as these.

Overall, users were successful in completing the task. Fourteen of the 17 participants (82%) found the standard greeting and the PFC website helpful in getting them started. In addition, 14 participants (82%) found the bookmarklet very easy to install. The few who did not find the task so easy provided feedback such as how to order the instructions on the PFC website, addition areas where more feedback can be given to the user and areas of further simplification. The overall success of this task shows that most users can bootstrap into the secure browser-based instant messaging with just a short greeting.

6.2 Task #2

The purpose of this task is to determine whether a user that has already installed the PFC bookmarklet is able to easily initiate a new secure chat. The task tells the participant that Steve needs their checking account number (one that we provide for them) for direct deposit

purposes. The task specifically gives instructions to securely send that number to Steve using Facebook Chat. It highlights the use the of the bookmarklet as well as entering secure chat using the lock icons described in a previous chapter.

An assumption we made during the design of this system is that users, by default, do not want secure communication enabled. As shown in a previous chapter, we provide a lock icon with which users can secure a chat session. Six participants (33%) said that clicking on the lock icon was an unnecessary step, that the bookmarklet should automatically lock communication. Fourteen of the 17 participants (82%) completed the task and reported they found the bookmarklet to be intuitive and easy to use.

The participant from the previous task that did not install the bookmarklet because of link trust concerns did not complete this task. That participant was even contacted by Steve again in an attempt to help install the bookmarklet. Another participant reported that they were very distrusting of Facebook and even if their chat session was guaranteed secure, they would never send sensitive information over Facebook channels. From conversations with this participant, we found that they would rather communicate sensitive information over the phone network.

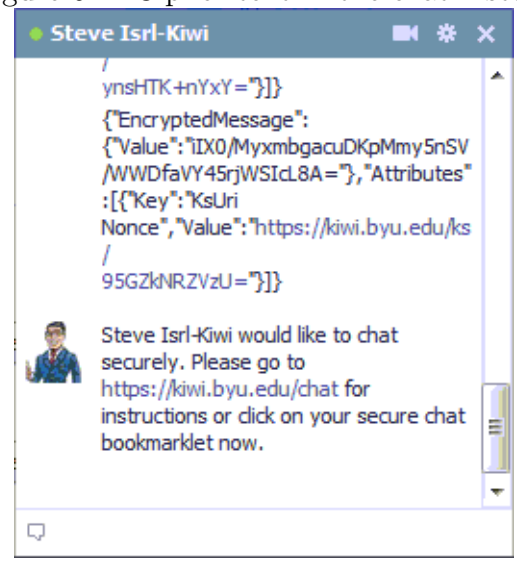
Three participants sent the checking account number insecurely reporting that they forgot the bookmarklet was there and needed reminding. This highlights a weakness of the bookmarklet approach since bookmarklets are unable to execute without direct interaction from the user. A browser plug-in approach would be able to better intervene and remind users of the need to use secure chat.

6.3 Task #3

The goal of this task is to get the reaction of the participants when they see cipher text in their chat histories and measure the usability of resuming a secure chat session. The task asks the user to re-assume the identity in task #1 and securely chat with Steve again. By resuming a conversation, the chat history in Facebook will contain the cipher text of

the previous conversation. We posit that the cipher text might be a point of confusion for participants unfamiliar with it. Figure 6.1 shows an example of the cipher text in the chat history.

Figure 6.1: Cipher text in the chat history



Five participants (32%) did not see cipher text during the task. Most of these participants did not complete the previous tasks that were needed for cipher text to appear in this task. Another 5 participants (32%) said that they understood the significance of the cipher text and it did not bother them. The remaining 7 participants (42%) who saw the cipher text had mixed reactions. One of the side effects of sending text that contains a valid URL is that Facebook converts them into links so that they are clickable. The PFC package (as seen in a previous chapter) contains a URL to the key server that was used to encrypt the message. Because of the link in the standard greeting, some participants, when they saw this new link in the cipher text, assumed they were supposed to click on it. It took them to an error page.

Other participants tried clicking on the link in the standard greeting again. Some participants reported that they were confused when they saw it or thought that the secure chat system was broken. Overall, 14 participants (82%) eventually used the bookmarklet and successfully completed the task. These participants agreed that resuming a secure chat

conversation was easy. More participants reported this task as easier than the previous task. We hypothesize that this is because the process of resuming a secure chat only requires a single click of the bookmarklet whereas the previous task also required the participants to click the lock icon.

6.4 Task #4

The goal of this task is to highlight the feature that helps a user maintain a secure chat session with the opposite party. The task instructs the participant to have a secure conversation with Steve, but that he is having problems with his computer. The instructions warn that Steve might jump in and out of secure communication. The participant is supposed to do all that they can to maintain a secure communication with Steve. This task measures the usability of the message shown previously in Figure 5.8.

Fifteen of the 17 participants (88%) agreed that it was easy to distinguish the difference between secure and insecure messages. These participants successfully completed the task. When interacting with the option to invite the opposite party to continue secure chat, some participants were expecting a *Ready!* notification when the opposite party rejoined. Overall, the participants were able to easily maintain a secure chat session with the other party.

6.5 Task #5

The goal of this task is to have participants switch in and out of a secure chat session during a conversation. The task instructs the participant to conduct a casual, insecure conversation with Steve. At some point during the conversation the participant must enter a secure chat session to send a bank account number. After sending that number, the participant must exit secure chat and finish the casual conversation they were having before. At this point in the study, the participants should be familiar with all aspects of the secure browser-based instant messaging system.

One of the features of the secure chat history is that messages are displayed as users would have received them. For example, if an insecure conversation is transitioned to a secure conversation, all previous insecure messages sent from the opposite party will be displayed as though they were received insecurely in secure mode (see Figure 5.8). Approximately half of the participants clicked on the action items presented in the insecure messages. These past messages seemed to cause a moment's confusion. However, as soon as the other party started chatting securely, all participants ignored the previous messages and continued with the task.

Fifteen participants (88%) reported that distinguishing the difference between secure and non-secure mode was easy. These participants referenced the lock icon as their primary means of demarcation. Fifteen participants (88%) said that using the lock icon to transition between secure and non-secure mode was easy and intuitive. Six participants (35%) commented that clicking the lock icon was an unnecessary step to enter secure mode because by clicking on the bookmarklet, they are already signaling their intention of chatting securely. Over half of the participants reported that once in secure chat they would continue in secure chat rather than leaving it, even if the topic of conversation becomes non-sensitive.

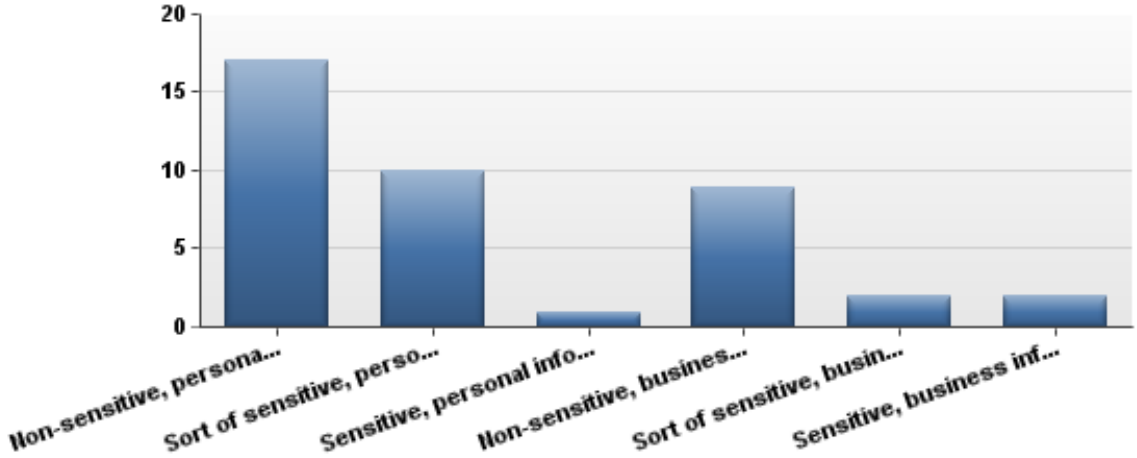
6.6 Follow-up

Following completion of the tasks, the participants answered some follow-up questions including several questions from the initial user survey. We also include questions that help us assess the participant learnings gained throughout this usability study. We try to assess the participants' inclination toward using the PFC and their understanding of why using it is important.

Twelve participants (70%) reported that they chat at least once per month. The top cited reasons for using chat are convenience and speed. Figure 6.2 shows the categories of the information participants cite sending on a regular basis.

Twelve participants (71%) said that after using PFC, they were less inclined to send sensitive information over regular chat–regular meaning insecure. The remaining 5

Figure 6.2: Categories of the information sent



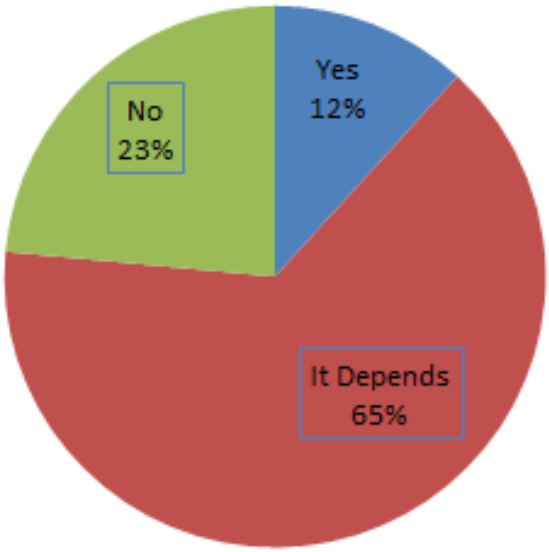
participants (29%) asserted they already do not send sensitive information over an insecure chat session. Nine participants (53%) stated they would be more inclined to send sensitive information via chat if PFC were available. Most of the other participants wanted more information about the underlying system before they would make a commitment to use it.

The success of the bootstrapping mechanism depends on users following the link sent in the standard greeting. We asked the participants if they trust links, Figure 6.3 shows their responses.

Those who answered *It Depends*, stated that their trust of the link was dependent on their trust of the party who sent it. If trust of the standard greeting link is an issue for most users the alternatives would be to change the standard greeting or rely on users to introduce the system to their contacts by giving them the link out-of-band.

Thirteen participants (76%) agreed they would be likely to start using this system with friends, family or acquaintances if it were available. The remaining 4 participants (24%) were undecided. A previous task has shown that some users forgot the bookmarklet was present. This lack of decision could be due to the lack of need or reminders to use this kind of secure system. Most participants reported that if they did use this system in the real world, they would, at the very least, enable it for sensitive topics of conversation. Eight participants

Figure 6.3: Question: Do you trust links sent to you over chat?



(47%) said their awareness about security changed towards chat systems as a result of their participation in this study.

6.7 Summary

Overall, PFC is usable. Everyone except two extremely novice users were able to successfully engage in secure chat sessions without training or human assistance. Two older, non-technical participants failed to complete any task. They use Facebook and Facebook Chat solely to stay in touch with family and close friends. They are only comfortable learning new technology with human guidance, so the current system was unable to meet their needs.

Users may be reluctant to bootstrap into the system by following the link contained in the standard greeting. In practice, this means a user wanting to initiate a secure chat session with someone unfamiliar with PFC may first have to chat insecurely and introduce them to the process.

The user study uncovered ways to improve the PFC bootstrapping web page to be more informative and easier to follow. During the study we observed that many participants were drawn immediately to the bookmarklet link before watching the video tutorial. As

a result, many assumed they were supposed to click on the link, whereas if they had first watched the tutorial, they would have created a bookmark instead. Some users ignore the video and attempt to use the system immediately. A short list of steps included on the web page that summarizes the video content will be helpful. For instance, this information can alert the users that they will need to trust the associated Facebook application in order to use PFC.

Participants quickly picked up the bookmarklet during the course of the study and recognized a way to streamline the process by having the one-click of the bookmarklet automatically opt the user into a secure chat session. Finally, the user study illustrated that the chat history is confusing when it contains links. Some participants assumed they were supposed to click on them. This can be improved by presenting the actions items in a dialog instead so that the actions do not display in the chat history.

Chapter 7

Threat Analysis

This section contains a threat analysis of the secure browser-based instant messaging architecture and the prototype implementation. We describe the most likely attacks and how well the system guards against them.

7.1 Secure Content Disclosure

End-to-end encryption prevents networks, clients, proxies and service providers from accessing my the plaintext. For secure content to be disclosed, an attacker must compromise the user's keys.

One attack vector is to compromise the key server. By obtaining unrestricted access to the key server, an attacker could obtain the master key and derive every type of key for any period. To mitigate this attack, special encryption hardware (e.g. cryptography cards) could be employed that prevents the master key from being copied off the machine. An attacker would only be able to generate keys for the periods they have access to the machine. We could also assign a key period to the master key. A rotating master key would ensure that only messages secured with keys derived from the current master key would be recoverable. Key escrow is used because it removes the need for key management from the user. If this manner of key distribution is too risky, an alternate mechanism of distributing keys could be implemented.

Another mode of the attack is to compromise the creator and viewer keys. By compromising the creator key, an attacker could read all the messages sent to any user

secured by that key during the period in which the key was valid. If the viewer key were compromised, an attacker could read all messages received from a certain creator during a specific period. To mitigate these attacks, the period for which a key is valid could be shortened considerably. This would decrease the number of messages secured by a single key and increase the frequency of key refresh.

However, in some instances, compromising the key server or the individual keys may not be enough for an attacker to discover the plain text messages. If an attacker is unable to eavesdrop on encrypted content sent across a network, the attacker would have to compromise the Facebook accounts of either the sender or receiver to acquire the encrypted content. This second layer of defense helps to mitigate the threat of secure content disclosure.

7.2 Authentication Vulnerabilities

We chose to use the authentication mechanisms built into the service provider because of their convenience and they also allow us to acquire a handle that uniquely identifies the sending and receiving parties. These unique handles are needed to derive the appropriate keys. PFC relies on Facebook's OAuth system to authenticate a user and provide us with basic identity information about the user for the aforementioned key derivation process.

A malicious service provider could impersonate a user to the key server all messages ever sent or received by this user. The system currently has no guard against a service provider maliciously impersonating a user. The system assumes that a service provider can be trusted not to impersonate the user; however, multi-factor authentication could be implemented to thwart a malicious service provider. If the risks of using a service provider's authentication mechanism is too great, PFC would have to be augmented to allow users to sign-up for PFC accounts. The PFC package would also have to be augmented to carry more identity information since it would no longer be relying on the user's identity within the service provider's network.

7.3 Content Swapping

An untrusted service provider could cause confusion by swapping secure content for secure content previous sent to the user. For example, an instant messaging provider could swap the contents of two chats, one containing the response "yes", and the other containing, "no." Or, a malicious service provider could also perform a similar attack by rearranging a web page's Document Object Model. For example, a service provider could swap the lock icons to make an insecure chat appear secure. This risk is inherent in the design of the system, because it integrates into the existing framework of the web. Note that this attack is not unique to our system: malicious service providers can already swap plaintext content. We could mitigate this attack by including messaging IDs that would aid in detecting previously sent messages.

7.4 Successful Message Modifying

One of the limitations of this system is the size of the messages that can be passed between sender and receiver. For example, Facebook enforces a limit of 1000 characters on each instant message. Consequently, message package features that would provide message integrity have been taken out to decrease the size of the secured message. Without these message integrity features, a malicious service provider could attempt to modify a message as it is being relayed through its network. Any message modification will result in a decryption failure. Message modification is also a form denial of service. To mitigate this attack, we could assign a unique ID to each instant message and provide a third party service that would store and relay message integrity information on demand.

7.5 Denial of Service

The system is vulnerable to a denial of service attack. For example, a service provider could simply delete all data it identifies as secure content, or a client proxy could simply refuse to forward secure content on behalf of the user. In addition, a key server could refuse to issue

keys or the key server could be overwhelmed with key requests preventing it from fulfilling all key requests. This type of risk is inherent in the system because it tries to co-exist with the existing Web infrastructure.

7.6 Malicious Links

One of the behaviors we saw repeatedly during the usability study was users' hesitation to click on the link provided in the standard greeting. However, once they accepted that they needed to click links, they assumed that any link presented by Facebook Chat was supposed to be clicked on. A malicious service provider could try to send malicious links to a user via the PFC standard greeting (or any other standard PFC message) since there is already an established trust with that message. The success of making this system widely available and easy to bootstrap into depends on the standard greeting. We could attempt to mitigate this attack by sending standard greetings via email, which would also allow us to provide a few more instructions in addition to the link. However, a service provider could follow the same model and send email messages to the user.

It is difficult to study trust decisions in usability studies because the studies are not real-life scenarios. During our usability study, we had one user indicate that they made the same trust decisions during the study that they make in real-life scenarios regarding clicking on links. We can assume from his response that his reactions to links in the study were genuine and something we can take learning from. However, for most other participants, being in the study may have biased their trust decision paradigm rendering any assumption about user trust reactions inaccurate regarding those users. As a result, we cannot make claims about how users' trust will change as we apply different methods of delivering the standard greeting.

Chapter 8

Conclusion

8.1 Contributions

This thesis presents the architecture of the Secure Browser-Based Instant Messaging system, a system allowing users to enjoy end-to-end privacy in instant messaging regardless of the transport medium and chat protocols. The architecture achieves this with components for Internet browsers and a key escrow system. In addition to the general architecture, this thesis provides a description of a prototype, a threat analysis and the results of two studies conducted before and after the implementation of the prototype–the first study focusing on user interest in secure chat and the second being a usability study of the prototype.

8.2 Summary of Findings

Instant messaging is a major form of communication, especially since it is integrated with major social networks and email providers. We conducted a survey of 65 users to assess their awareness and attitudes concerning the privacy of instant messaging. Thirty-eight users (59%) surveyed feel safe or very safe while communicating via instant messaging, and 10 users (15%) of that group admit to sending highly sensitive information in a chat session. Users are more likely to share sensitive information with a close friend or family member. Most users feel safe communicating via instant messaging in a casual manner. However, when discussing sensitive topics, users feel more safe communicating in person or over the phone. A number of users are wary of chat service providers; 57% are concerned that providers may scan their

messages for directed advertising purposes, and 49% have concerns with their messages being stored permanently.

When presented PFC, most users felt that it is something they would use if available. Users also felt that they would be comfortable sending sensitive information via an instant messaging system. There are a few areas of improvement that users have identified:

- A more step-oriented PFC website to make the on-boarding process easier and the system more trustworthy to the user.
  - Include tips informing users that the links will only ask them to navigate to the `pfc.byu.edu` domain.
  - Include snapshots of the different dialogs (e.g., Facebook authentication dialog) and an explanation that they are required in order for the system to function.
- Automatically secure all chat sessions after the system has been activated by the bookmarklet so that users do not have to take the extra step of clicking on the lock icon.
- Simplify the chat history and only show action items (e.g., see Figure 5.8) for the most recently received message.

The threat analysis identified a social engineering vulnerability where a service provider could take advantage of a user's established trust with PFC and attempt to send malicious links in the standard greeting. The web-based nature of the overlays precludes us from adequately addressing this issue. A service provider, like Facebook, will always be able to send links to the user. Consequently, there are trade-offs with using PFC versus other systems such as OTR. OTR provides stronger security properties than PFC, however, it is also an independent instant messaging protocol requiring adoption on service providers and clients. PFC is protocol agnostic. We conveniently overlay existing interfaces to provide end-to-end privacy. Using PFC is as simple as using a browser bookmark. By building on top of what

already exists in the web, a user can easily adopt security into their daily workflow with very few changes.

8.3 Future work

8.3.1 Formalize Secure Overlay APIs for Service Providers

There are many ways a message can be secured. It is unreasonable for a service provider to be burdened with integrating the various methods of securing a message. During the development of the prototype for Facebook Chat, we found many useful Facebook JavaScript objects available that allowed us to integrate seamlessly with their web-based chat client. These objects made it possible to receive notifications when messages arrived, query chat history and send secure messages. It would be useful to define a standardized secure messaging API that service providers could implement and expose on all web-based communication tools that would allow third-party systems to integrate and provide additional services inside the application.

8.3.2 Multi-factor Authentication

This secure messaging system leverages the service provider's identity provider. Thus, a malicious service provider could easily impersonate a user. Future research could investigate a means of incorporating a multi-factor authentication system. The other authentication factors would prevent a misbehaving service provider from impersonating a user.

8.3.3 Usability

As outlined in the summary of findings, certain deficiencies were found in PFC. Future work could include addressing the outlined deficiencies and conducting an additional usability study to validate the improvements.

References

[1] Alexander, C., and Goldberg, I. Improved User Authentication in Off-the-Record Messaging. In *Proceedings of the ACM Workshop on Privacy in Electronic Society* (New York, NY, USA, 2007), ACM, pp. 41–47.

[2] Bardis, N. G., and Ntaikos, K. Design of a Secure Chat Application Based on AES Cryptographic Algorithm and Key Management. In *Proceedings of the 10th WSEAS International Conference on Mathematical Methods, Computational Techniques and Intelligent Systems* (Stevens Point, Wisconsin, USA, 2008), World Scientific and Engineering Academy and Society, pp. 486–491.

[3] Borisov, N., Goldberg, I., and Brewer, E. Off-the-Record Communication, or, Why Not to Use PGP. In *Proceedings of the ACM Workshop on Privacy in the Electronic Society* (New York, NY, USA, 2004), ACM, pp. 77–84.

[4] Casey, D. Building a Secure Instant Messaging Environment. *Network Security 2007*, 1 (2007), 18 – 20.

[5] Di Raimondo, M., Gennaro, R., and Krawczyk, H. Secure Off-the-Record Messaging. In *Proceedings of the ACM Workshop on Privacy in the Electronic Society* (New York, NY, USA, 2005), ACM, pp. 81–89.

[6] Fong, K., Hunt, J., Liburd, S., and McLean, E. Really Secure Chat, Really! `http://www.cs.siu.edu/kfong/research/securechat.pdf`, 2002.

[7] Goldberg, I., Ustaoğlu, B., Van Gundy, M. D., and Chen, H. Multi-party Off-the-Record Messaging. In *Proceedings of the 16th ACM Conference on Computer and Communications Security* (New York, NY, USA, 2009), ACM, pp. 358–368.

[8] Hindocha, N. Instant Insecurity: Security Issues of Instant Messaging. *Security Online Focus 13* (2003), 9–15.

[9] Jennings, R. B., Nahum, E. M., Olshefski, D. P., Saha, D., Zon-Yin, S., and Waters, C. A Study of Internet Instant Messaging and Chat Protocols. *IEEE Network 20*, 4 (2006), 16–21.

[10] Jiang, B., Seker, R., and Topaloglu, U. Off-the-Record Instant Messaging for Group Conversation. In *Proceedings of the IEEE International Conference on Information Reuse and Integration* (2007), pp. 79–84.

[11] Kikuchi, H., Tada, M., and Nakanishi, S. Secure Instant Messaging Protocol Preserving Confidentiality Against Administrator. In *Proceedings of the 18th International Conference on Advanced Information Networking and Applications* (2004), vol. 2, pp. 27–30.

[12] Leavitt, N. Instant Messaging: A New Target for Hackers. *Computer 38*, 7 (2005), 20–23.

[13] Lee, W., and Lee, J. Design and Implementation of Secure E-mail System Using Elliptic Curve Cryptosyste. *Future Generation Computer Systems 20*, 2 (2004), 315 – 326.

[14] Loesing, K., Dorsch, M., Grote, M., Hildebrandt, K., Roglinger, M., Sehr, M., Wilms, C., and Wirtz, G. Privacy-aware Presence Management in Instant Messaging Systems. In *Proceedings of the 20th International Parallel and Distributed Processing Symposium* (April 2006).

[15] Lux, K. D., May, M. J., Bhattad, N. L., and Gunter, C. A. WSEmail: Secure Internet Messaging Based on Web services. In *Proceedings of the IEEE International Conference on Web Services* (2005), vol. 1, pp. 75–82.

[16] Mannan, M., and Oorschot, P. C. V. Secure Public Instant Messaging: A Survey. In *Proceedings of the 2nd Annual Conference on Privacy, Security and Trust* (2004), pp. 69–77.

[17] Mannan, M., and van Oorschot, P. A Protocol for Secure Public Instant Messaging. *Financial Cryptography and Data Security 4107/2006* (2006), 20–35.

[18] Mannan, M., and van Oorschot, P. C. On Instant Messaging Worms, Analysis and Countermeasures. In *Proceedings of the 2005 ACM Workshop on Rapid Malcode* (New York, NY, USA, 2005), ACM, pp. 2–11.

[19] Montminy III, J., and Wilson, B. On-The-Record: A Non-Repudiable, Authenticated, and Confidential Chat Client. `http://userpages.umbc.edu/~montmin1/globecomSubmission.pdf`.

[20] Nicholson, J., and Building, W. A Cryptographically Secure Decentralised Communication Protocol. In *Proceedings of the Brazilian Symposium on Information Security and Computer Systems* (2008), pp. 422–442.

[21] Schneier, B. The Security Mirage. `http://blog.ted.com/2011/04/26/the-security-mirage-bruce-schneier-on-ted-com/`, Oct 2010.

[22] Sheng, S., Broderick, L., Hyland, J., and Koranda, C. Why Johnny Still Can't Encrypt: Evaluating the Usability of Email Encryption Software. In *Proceedings of the Symposium On Usable Privacy and Security* (2006).

[23] Stedman, R., Yoshida, K., and Goldberg, I. A User Study of Off-the-record Messaging. In *Proceedings of the Symposium on Usable Privacy and Security* (2008), pp. 95–104.

[24] Whitten, A., and Tygar, J. Why Johnny Can't Encrypt: A Usability Evaluation of PGP 5.0. In *Proceedings of the 8th USENIX Security Symposium* (1999), vol. 99.

[25] Wikipedia. Phreaking, 2011. [Online; accessed 12-Nov-2011].

[26] Zhang, B., Feng, M., Xiong, H.-r., and Hu, D.-y. Design and Implementation of Secure Instant Messaging System Based on MSN. *Proceedings of the International Symposium on Computer Science and Computational Technology 1* (2008), 38–41.

Appendix A

Initial user survey

The following is the survey distributed to users to assess inclination and interest toward using secure chat.

Thank you for your interest in taking this survey about Internet chat and instant messaging. The goal of this survey is assess how you use Internet chat and some of your reasoning behind it. This survey is completely anonymous in every way. We will not ask you to provide any personally identifiable information. If you feel uncomfortable with any questions, you may quit taking the survey at any time. This survey should only take about 5 minutes of your time.

If you are okay with that, please advance and begin answering the questions.

1. Do you use a chat system (i.e., Google Talk, MSN Live Messenger, Yahoo Messenger, etc.) to communication with someone?

   - Yes
   - Not any more
   - No, I never have

2. If not any more, why?

3. Which system(s) do you use? Check all that apply.

   - Windows Live Messenger (formerly MSN Messenger or Hotmail Instant Messenger)
   - Google Talk
   - ICQ
   - AIM
   - IRC
   - Skype

- Yahoo Instant Messenger
- Facebook Chat
- Others, please specify

4. On average, how often do you communicate via chat?

    - Never
    - Less than Once a Month
    - Once a Month
    - 2-3 Times a Month
    - Once a Week
    - 2-3 Times a Week
    - Daily

5. Why do you use chat systems? Check all that apply.

    - It's convenient
    - It's fast
    - It's secure
    - Lots of people communicate that way
    - Easier than texting
    - Cheaper than texting

6. How safe do you feel when using a chat system?

    - Very safe
    - Safe
    - I don't know
    - Unsafe
    - Very unsafe

7. When you chat with another person, what kinds of information do you send? Check all that apply.

    - Non-sensitive, personal information
    - Sort of sensitive, personal information (i.e., birthday, phone number, addresses, etc.)

- Sensitive, personal information (i.e., social security number, credit card, bank info)
- Non-sensitive, business information
- Sort of sensitive, business information
- Sensitive, business information
- I've never thought about it before

8. Please agree or disagree with the following statement: I'm more likely to send sensitive information to a trusted friend or family member.

   - Strongly Disagree
   - Disagree
   - Depends
   - Agree
   - Strongly Agree

9. If depends, on what?

10. If a friend or family member asked you for some sensitive information over chat, would you reply with the requested information?

    - Yes, I trust them
    - Yes, if they have a need to know
    - Depends on the kind of information
    - Probably not, I'd email them the information instead
    - No, I would give it to them over the phone
    - Not at all

Some chat services save your messages behind the scenes so that you can go back and view them later. They also scan those messages to deliver targeted advertising.

11. Please agree or disagree with following statement: I'm confident that my chat conversations are private (e.g., no one can eavesdrop).

    - Strongly Disagree
    - Disagree
    - I don't know
    - Agree

- Strongly Agree

12. Please agree or disagree with the following statement: When I chat, I'm confident that I'm actually talking with the person I think I'm talking with.

    - Strongly Disagree
    - Disagree
    - I don't know
    - Agree
    - Strongly Agree

13. Please agree or disagree with the following statement: It concerns me that chat providers scan my messages to deliver advertisement.

    - Strongly Disagree
    - Disagree
    - Indifferent
    - Agree
    - Strongly Agree

14. Please agree or disagree with the following statement: It concerns me that chat providers store my messages behind the scenes.

    - Strongly Disagree
    - Disagree
    - Indifferent
    - Agree
    - Strongly Agree

15. Why do you use the particular chat client (i.e., Live Messenger, Google Talk, Yahoo Instant Messenger, ICQ, etc.) that you do? Check all that apply.

    - It's easy to learn
    - It's easy to use
    - It's secure
    - I've never tried anything else
    - It's the only one available for my particular chat provider
    - Other

16. If your chat client was found to be vulnerable to attacks (i.e., found to be insecure), would you be inclined to move to another client?

    - Yes
    - Only if it offered the same features I'm already used to
    - No, I'd prefer the client just get fixed
    - No, it doesn't matter to me

Appendix B

Usability study

The following is the usability study we conducted.

Thank you for your participation in this user study. Throughout this study, you will be asked to perform certain tasks in relation to Facebook Chat and then provide feedback to help us improve our software.

During the course of this study, we will record you via a web camera attached to the monitor. We will also record all acts taking place on the screen. This will help us learn whether or not our software is easy to use. None of the video content captured during the study will be released publicly or given to a third party. In addition, we will also ask you to provide some demographic information. As with the video, none of the results published as part of this research will personally tie back to you as a participant.

We will ask you to use Facebook Chat during this study. You will NOT use your own Facebook account, instead, we will provide you a list of accounts specifically created for this study. If you feel uncomfortable during the study, you may stop at any time. The time commitment is 10-20 minutes. Please advance to the next screen to begin.

B.1 Demographics

Please provide the following information.

1. What is your gender?
   - Male
   - Female
2. What is your age?
   - 18 - 25
   - 26 - 35

- 36 - 45
- 46 - 55
- 56 - 65
- Over 65

3. Are you currently a student?
    - Yes
    - No
4. What is your major?
5. What is your current occupation?

B.2 Instructions

There will be five tasks to complete. Follow the preparation instructions first before you follow the main task scenario instructions. Make sure you read the scenarios very carefully. If you don't understand the scenario, please ask the proctor for clarification. You MUST use the Google Chrome web browser. If you need to open a new browser window, open a new tab instead using the little plus icon next to the tabs. Be careful not to close the browser window as you will lose your study responses. Advance to the next section to begin.

B.3 Task #1

B.3.1 Preparation

Login to Facebook using account #1 on your account information sheet.

B.3.2 Scenario

Steve, a close co-worker and Facebook friend, said he'd get some passwords to you in order to access some online company accounts. Wait for Steve to contact you via chat. Once you have received the access information, advance to the next section to answer some questions.

B.3.3 Questions

1. How helpful was the initial greeting (e.g. "Steve Isrl-Kiwi would like to chat securely. Please go to...") in getting started?
    - Very Helpful

- Helpful
- Neither helpful nor unhelpful
- Unhelpful
- Very unhelpful

2. (Optional) Please explain you answer.

3. The bookmarklet was easy to install.

   - Strongly Disagree
   - Disagree
   - It Depends, please explain
   - Agree
   - Strongly Agree

4. (Optional) Please explain you answer.

5. How helpful were the instructions on the Kiwi website (https://kiwi.byu.edu) in setting up the bookmarklet?

   - Very Helpful
   - Helpful
   - Neither helpful nor unhelpful
   - Unhelpful
   - Very unhelpful

6. (Optional) Please explain you answer.

B.4 Task #2

B.4.1 Preparation

1. Logout of Facebook (there is a little arrow in the top right, click on it and select Logout)

2. Login to Facebook using account #2 on you account information sheet

B.4.2 Scenario

You just realized that you forgot to give Steve your checking account number. He needs it so that you will receive direct deposit payments for the work that you've done for him. Don't wait for him to contact you, you initiate the chat. Use Facebook Chat to securely send him the checking account number listed on the account information sheet. Wait for him to reply with the amount he will be depositing into your account. Then, advance to the next section to answer some questions.

B.4.3 Questions

1. The bookmarklet is intuitive and easy to use.
    - Strongly Disagree
    - Disagree
    - It Depends, please explain
    - Agree
    - Strongly Agree
    - No Opinion
2. (Optional) Please explain you answer.

B.5 Task #3

B.5.1 Preparation

1. Logout of Facebook
2. Login to Facebook using account #1 on you account information sheet

B.5.2 Scenario

Since your last chat conversation with Steve, a password to an online company account has changed. Wait for Steve to contact you via chat to give you the new password. Conduct a secure chat with him. Then, advance to the next section to answer some questions.

B.5.3 Questions

1. What was your first reaction upon seeing the jumbled text in your chat history?
    - I didn't see any jumbled text

- I understood what it was and it didn't bother me
- I wasn't expecting it and it confused me
- I thought something was broken
- Other, please explain

2. It was easy to resume a secure chat with Steve.

   - Strongly Disagree
   - Disagree
   - It Depends, please explain
   - Agree
   - Strongly Agree

3. (Optional) Please explain you answer.

B.6 Task #4

B.6.1 Preparation

1. Logout of Facebook
2. Login to Facebook using account #3 on you account information sheet

B.6.2 Scenario

Steve needs to have a secure conversation with you. He will contact you. He is having problems with his computer. During your chat you might experience some moments where he communicates insecurely. Do all that you can to keep the conversation as secure as possible. Then, advance to the next section to answer some questions.

B.6.3 Questions

1. It is easy to recognize when someone sends an insecure message while I'm in secure mode.

   - Strongly Disagree
   - Disagree
   - It Depends, please explain
   - Agree

- Strongly Agree
- No Opinion

2. (Optional) Please explain you answer.

B.7 Task #5

B.7.1 Preparation

1. Logout of Facebook
2. Login to Facebook using account #4 on you account information sheet

B.7.2 Scenario

Suppose Steve is your brother. Both your families share a cell phone family plan to save money. Since he is the primary account holder, you need to transfer your share of the monthly bill directly into his bank account. Start a brief casual family conversation. For example, you might ask him how he's doing, how his family is, or how work is going. Then switch to a secure chat to ask him for his bank account number. Once he replies, exit secure chat and briefly continue your casual family conversation. Then, advance to the section and answer some questions.

B.7.3 Questions

1. It is easy to distinguish the difference between secure mode and non-secure mode.
    - Strongly Disagree
    - Disagree
    - It Depends, please explain
    - Agree
    - Strongly Agree
2. (Optional) Please explain you answer.
3. Going between secure and non-secure mode is intuitive and easy to do.
    - Strongly Disagree
    - Disagree
    - It Depends, please explain

- Agree
- Strongly Agree

4. (Optional) Please explain you answer.

5. In real life, if you were in a secure chat and your topic of conversation moved to something casual or non-sensitive, would you take the time to exit secure chat to continue your conversation?

   - Yes
   - No
   - Maybe

6. (Optional) Please explain you answer.

B.8 Follow-up Questions

You have completed the task portion of the study. You're almost done. Please answer the following questions.

1. On average, how often do you communicate via chat?

   - Never
   - Less than Once a Month
   - Once a Month
   - 2-3 Times a Month
   - Once a Week
   - 2-3 Times a Week
   - Daily

2. Why do you use chat systems? Check all that apply.

   - It's convenient
   - It's fast
   - It's secure
   - Lots of people communicate that way
   - Easier than texting
   - Cheaper than texting

- Other

3. Which of following kinds of information have you previously sent to someone using chat? Check all that apply.

    - Non-sensitive, personal information
    - Sort of sensitive, personal information (i.e., birthday, phone number, addresses, etc.)
    - Sensitive, personal information (i.e., social security number, credit card, bank info)
    - Non-sensitive, business information
    - Sort of sensitive, business information
    - Sensitive, business information
    - I've never thought about it before

4. After using this secure chat system, I'm less inclined to send sensitive information in my REGULAR chat system.

    - Strongly Disagree
    - Disagree
    - It Depends, please explain
    - Agree
    - Strongly Agree

5. (Optional) Please explain you answer.

6. If this system were available, I would be more inclined to sent sensitive information via chat.

    - Strongly Disagree
    - Disagree
    - It Depends, please explain
    - Agree
    - Strongly Agree

7. (Optional) Please explain you answer.

8. Do you trust links that get sent to you over chat?

    - Yes

- It Depends, please explain
- No

9. (Optional) Please explain you answer.

10. How likely would you be to start using this system with your friends, family or acquaintances if it were available?

    - Very Unlikely
    - Unlikely
    - Somewhat Unlikely
    - Undecided
    - Somewhat Likely
    - Likely
    - Very Likely

11. (Optional) Please explain you answer.

12. If you were to use this system in the real world, how would you use it? Check all that apply.

    - I'd always chat securely
    - I'd use it with my closest friends
    - I'd turn it on when talking about something sensitive
    - Other, please specify

13. After going through this study, my awareness about security has changed towards the chat system(s) I use.

    - Strongly Disagree
    - Disagree
    - It Depends, please explain
    - Agree
    - Strongly Agree

14. (Optional) Please explain you answer.

15. Any other feedback, thoughts or suggestions?