Exploiting the Web for Point-in-Time File Sharing

Roberto J. Bayardo
IBM Almaden Research Center
San Jose, CA 95120 USA
bayardo@alum.mit.edu

Sebastian Thomschke
IBM Deutschland GmbH
Nahmitzer Damm 12, 12277 Berlin
sebastian.thomschke@de.ibm.com

ABSTRACT

We describe a simple approach to "point-in-time" file sharing based on time expiring web links and personal webservers. This approach to file sharing is useful in environments where instant messaging clients are varied and don't necessarily support (compatible) file transfer protocols. We discuss the features of such an approach along with a successfully deployed implementation now in wide use throughout the IBM corporation.

Categories: H.4.1 [**Office Automation**]: *Groupware*
General terms: Human Factors, Security
Keywords: Instant messaging, file sharing, personal web server

1. INTRODUCTION

Instant messaging (IM) is already a fundamental communication mechanism both inside and outside the workplace. Its use within the IBM corporate intranet is pervasive: even an experimental internal messaging client attracts over 20,000 users every month [3], and the officially supported corporate instant messaging client (IBM Lotus Instant Messaging) is used by substantially more. Instant messaging conversations regularly lead to the need for sharing content contained within files stored on the local file system. We call such sharing of file system content *point-in-time* file sharing, since the need to share the file is immediate and often unanticipated.

A common solution for IM-based point-in-time file sharing is to develop and include a proprietary transfer protocol within the IM client itself. The resulting proliferation of proprietary and often unreliable protocols has created a situation where users who are perfectly able to chat with one another regularly find themselves unable to share files. Rather than propose yet another proprietary file transfer method for addressing the point-in-time file sharing problem, we have developed and deployed a solution that exploits the ubiquity of the world-wide web. The advantage of this approach is full interoperability: it allows file sharing between disparate clients regardless of their (lack of) support for proprietary transfer protocols. With our solution, the file sender simply sends a special URL to the receiver, which can be clicked by the receiver to invoke the transfer. Almost every IM client will launch the default browser in response to clicking on a link, allowing the browser to perform the download without requiring any special hooks. The approach naturally supports distributing files to multiple people participating in multi-user chats, and can leverage the proven security of existing web standards such as HTTPS for authentication and encryption.

File sharing through personal or centralized web servers has been around since the advent of the web. The idea of using "out of band" file transfer methods such as HTTP within an instant messaging framework is also well known, having even been formalized as part of the Jabber instant messaging framework [4]. But most users find the approach too cumbersome for use in the point-in-time context, as it involves (1) copying or uploading the file to the appropriate location, (2) forming the appropriate URL (3) sending the URL to the recipient, and (4) removing the file from the web server once it is downloaded. The security conscious must in addition set appropriate access permissions to avoid having the file disclosed beyond the intended recipient. This burdens the recipient, who must obtain and remember login information, and provide it before retrieving the file. Firewalls and network address translators (NATs), which can render local webservers inaccessible, are another impediment that has prevented this approach from achieving any widespread use.

Our solution to the point-in-time file sharing problem involves simplifying, streamlining, and securing the cumbersome webserver-based approach as follows:

- Rather than require the user manually copy or upload the file and then manually formulate the URL, our solution provides context-menu support for immediately generating a special URL that allows the file to be downloaded from its existing location.
- Instead of requiring that users remove shared files once they have been downloaded and/or manually assign appropriate access permissions, the URL generated by our implementation is tamper proof and time-expiring. The URL itself therefore serves as the necessary authorization credentials, which expire after a brief period to minimize opportunities for misuse.
- We eliminate the firewall/NAT issue and avoid any dependence on a large and expensive centralized infrastructure by leveraging the personal web server framework described in [2].

2. DESCRIPTION

This section illustrates how file-sharing is accomplished by end-users of our web-based point-in-time file sharing tool. The tool is implemented as a plugin (called SecureLink) for the YouServ personal web-serving system [1], which runs on a variety of operating systems. The plugin has been available within the IBM intranet on an experimental basis for the last 13 months, during which it has been used by over 600 unique users to send over 5500 files ranging in size from a few bytes to hundreds of megabytes.

Only the file sender needs to install and run the YouServ/SecureLink software in order to share files. To share a file, the sender first locates the file using the host system's file system navigator, right-clicks the file, and selects "Create a SecureLink" (Figure 1). The result of this operation is a dialog indicating that the necessary information has been copied to the clipboard. The sender then pastes the clipboard contents into the IM window and sends the message to the recipient (Figure 2). Once the message is received, the receiver can simply click the link to retrieve the file. Note that the IM containing the link includes information such as the file's name and size, along with a description should the sender chooses to provide one. The message also indicates the expiration period after which the link becomes invalid. Clicking the link launches the default browser to a welcome page that immediately pops up a Save/Open dialog for retrieving the file. The welcome page also provides help information in case the receiver has concerns about the process, and a link to restart the transfer in case the dialog is dismissed or the transfer fails.

Because each step of this file transfer process uses only standard operating-system provided features, this approach can be used with any IM or chat client both on the sending and receiving sides. The only requirement is that the sender and receiver are capable of

WWW 2005, May 10-14, 2005, Chiba, Japan.
1-59593-051-5/05/0005

exchanging text messages.

To further streamline the process, we have implemented some simple integration into the IBM Community Tools (ICT) client, which is a popular alternative to the IBM corporate standard client. The IM window provided by this client is depicted in Figure 2. The "Send File" button at the bottom of the window can be clicked to bring up a file-select dialog. The dialog provides a standard file system navigator and also supports drag and drop. Once a file is selected by either of these methods, a URL to the file is generated and sent to the receiver, bypassing the intermediate clipboard step required of the non-integrated approach. We note that this feature requires only the file sender to use an integrated client. The requirements of the receiver's client are as before.

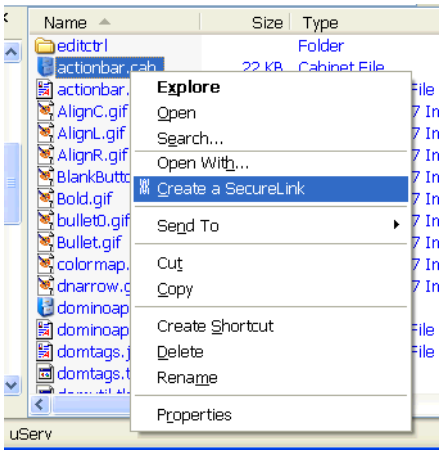

Figure 1. User designates the file to share with the file navigator and the SecureLink right-click menu item.

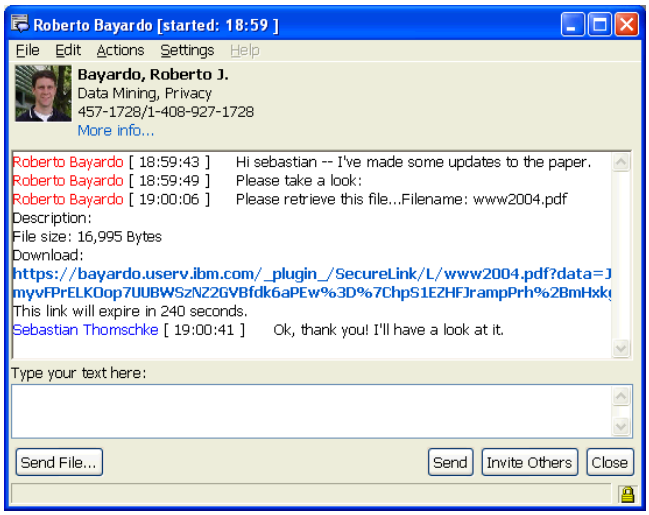

Figure 2. IM partner receives a link that can be clicked to retrieve the file.

3. IMPLEMENTATION

SecureLink is implemented as a plugin for the YouServ web-hosting system. Each YouServ node is an HTTP server with additional features that allow nodes to form a web serving "grid." Among other things, participating nodes can exploit peer-to-peer proxying and relaying in order to circumvent firewalls and NATs. YouServ plugins are delegated all HTTP requests that are appropriately prefixed (http://[user's domain]/_plugin_/SecureLink), allowing them to implement dynamic content as well as more interesting features by leveraging other aspects of the YouServ infrastructure.

The SecureLink plugin handles requests for issuing URLs to specified files and requests to serve the specified files. The URL issuing function of the plugin allows an appropriately authenticated user or application to generate a URL to any file on the host system. Each such URL contains a 3DES encrypted payload consisting of the file's path name F, the time the URL expires T, and the value SHA1($F|T$) where SHA1() is the SHA1 cryptographically secure hash function and "|" is the concatenation operator. Before serving any content, the plugin (1) decrypts the URL's payload, (2) computes SHA1($F|T$) where F and T are taken from the decrypted payload, and (3) compares the value computed by step 2 to the hash embedded within the payload. If the two hash values fail to match, the plugin will return an error page as it knows the payload has been tampered with or is otherwise corrupted. Note that the only state the plugin must maintain is the 3DES key used to generate and decrypt the URL payload.

If the YouServ node run by the sender supports encrypted connections (as most do), the SecureLink plugin will issue an HTTPS URL for downloading the specified file. An HTTPS encrypted transfer prevents the request as well as the transfer from being intercepted or replayed. If the IM client itself uses an encrypted channel for message exchange (as is the case for most corporate IM clients), there is no unencrypted information exchanged that could allow eavesdropping on or hijacking of the transfer. For YouServ sites or IM clients that do not support encrypted connections, it is possible for the URL and/or transfer to be intercepted by eavesdroppers, but the eavesdroppers cannot use this information to retrieve other files on the server it does not already have access to.

Once the link is clicked by the receiver, the SecureLink plugin on the sender's machine receives the resulting request, decrypts and verifies integrity of the payload as described earlier, and serves the welcome page if the URL is valid and unexpired. This welcome page includes a meta-refresh directive that redirects to another payload protected URL in order to prompt the browser to display the Save/Open dialog via the HTTP content-disposition header. The response body for this redirected request delivers the designated file if the request payload passes the integrity and expiration tests. A connection between two end users may be unreliable and will sometimes break, but most browsers will transparently resume broken transfers through appropriate specification of HTTP range headers in subsequent requests. Our implementation extends the expiration period of URLs gradually based on the first byte offset of the range request in order to prevent expiration from blocking attempts at transfer resumption.

Acknowledgements

We are indebted to the ICT development team for their assistance in developing and deploying an integrated file-transfer solution. We also thank Hanspeter Jochmann who originally suggested the concept of a right-click context menu for supporting file sharing via unmodified IM clients. Finally, we are grateful for the invaluable feedback we have received from the users of our system, and the IBM corporation for allowing intranet users access to our experimental tools.

4. REFERENCES

[1] R. J. Bayardo Jr., A. Costea, and R. Agrawal. *Peer-to-Peer Sharing of Web Applications*. IBM Research Report RJ 10268, Nov. 2002.

[2] R. J. Bayardo Jr., A. Somani, D. Gruhl, and R. Agrawal. YouServ: A Web Hosting and Content Sharing Tool for the Masses. In *WWW-2002*.

[3] F. Jania. Broadcast Messaging: Messaging to the Masses. In *ACM Queue* 1(9), Nov 2003.

[4] P. Saint-Andre. *JEP-066: Out of Band Data*, version 1.0. Jabber Enhancement Proposal, Oct 8, 2003.