**RJ10334 (A0402-013) February 9, 2004**
**Computer Science**

# IBM Research Report

## Exploiting the Web for Point-in-Time file Sharing

**Roberto J. Bayardo Jr.**
IBM Research Division
Almaden Research Center
650 Harry Road
San Jose, CA 95120-6099

**Sebastian Thomschke**
IBM Deutschland GmbH
Alt-Moabit 101a
10559 Berlin
Germany

# Exploiting the Web for Point-in-Time File Sharing

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

Sebastian Thomschke
IBM Deutschland GmbH
Alt-Moabit 101a, 10559 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.

## 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, particularly in work-related environments, often lead to the need and desire for sharing other content contained within files stored on the local filesystem. We call sharing of such information *point-in-time* file sharing, since the need to share the file is not typically known apriori, and the need to share the file is immediate.

Point-in-time file sharing is not offered by the standard IM client used by most IBM employees. Many have clamored for such a feature, making it the most requested of all enhancements proposed by the developers and users of the experimental client mentioned earlier (IBM Community Tools, or ICT [3]). This situation inside IBM is not unique. On the public internet, there are several large IM networks, but only a few of which support point-in-time file transfer. Even those that do support it do so via proprietary protocols rarely implemented by alternate clients such as those implementing the Jabber protocol, a protocol intended to provide interoperability with multiple proprietary IM networks. As a result, users who are perfectly able to chat with one another across disparate networks using interoperable clients must resort to out-of-band means for sharing files, usually at great inconvenience.

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 based on exploiting 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 URL, allowing the browser to perform the download without requiring any special hooks.

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]. However, 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 are another impediment that has prevented this approach from achieving any widespread use for point-in-time sharing. Rather than suffer such complications, our experience is that users instead resort to e-mailing attachments. But e-mail can be slow, which goes against the desire for immediate sharing typical in conversational contexts, and may be limited by restrictive mailbox size quotas.

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

- Instead of requiring users upload files to a remote web space, we leverage the personal web server framework described in [2] which solves issues of firewalls and restrictive space quotes.
- Rather than require the user manually copy or upload the file and then manually formulate the URL, we allow a special URL to be immediately generated from a context-menu that allows the file to be (securely) downloaded directly 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 link generated by our approach is digitally signed to avoid tampering and time-expiring. The link itself therefore serves as the necessary authorization credentials, which expire after a brief period to minimize opportunities for misuse.

## 2. DESCRIPTION

This section illustrates how file-sharing is accomplished by end-users of our web-enabled 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. Only the file sender needs to install and run the YouServ/SecureLink software. To date, SecureLink has been installed by over 500 users.

To share a file, the sender locates the file using the host system’s file explorer, right-clicks the file, and then 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 user then pastes the clipboard contents into the IM window and sends the message to the recipient (Figure 2). When 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. 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 exchanging text messages.

To further streamline the process, we have implemented some simple integration into the ICT client, which is a popular alternative to the corporate standard client. This client is capable of communicating over the corporate IM network, among others. 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 link 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 remain the same as previous.

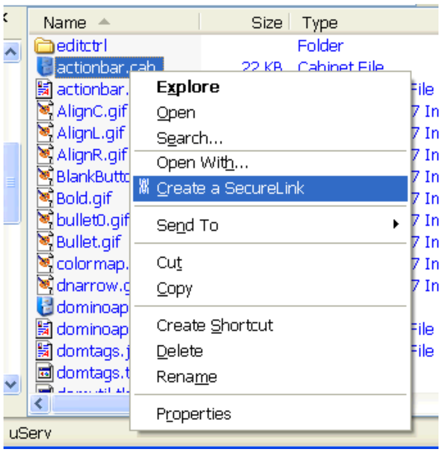
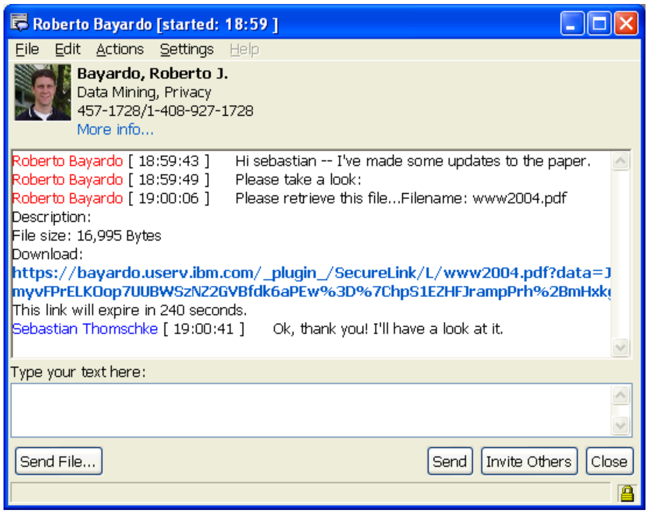


**Figure 1. User designates the file to share with the file navigator and the SecureLink right-click menu item.**



Bayardo, Roberto J.
Data Mining, Privacy
457-1728/1-408-927-1728

Roberto Bayardo [ 18:59:43 ] Hi sebastian -- I've made some updates to the paper.
Roberto Bayardo [ 18:59:49 ] Please take a look:
Roberto Bayardo [ 19:00:06 ] Please retrieve this file...Filename: www2004.pdf
Description:
File size: 16,995 Bytes
Download:
https://bayardo.userv.ibm.com/_plugin_/SecureLink/L/www2004.pdf?data=J
myvFPrELKOop7UUBWSzNZ2GVBfdk6aPEw%3D%7ChpS1EZHFJrampPrh%2BmHxk
This link will expire in 240 seconds.
Sebastian Thomschke [ 19:00:41 ] Ok, thank you! I'll have a look at it.

**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, this allows nodes to exploit proxying/relaying 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 links to specified files and requests to serve requested files. The link issuing function of the plugin allows an appropriately authenticated user or application to generate a URL to any file on the host system. Embedded within the URL are the name of the file and the time when the link expires. Security is provided by 3DES encrypting and SHA-1 signing this information with a secret key. Because all information required to retrieve the file is embedded within the link, the plugin is stateless other than the secret key required to interpret it.

If the YouServ client run by the sender supports encrypted connections (which most do), the SecureLink plugin will generate an HTTPS link. 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 our 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 link and/or transfer to be intercepted, but nothing more.

Once the link is clicked by the receiver, the SecureLink plugin on the sender's machine receives the resulting request, decrypts the filename and expiration time, verifies the digital signature, and serves the welcome page if the link is valid. The welcome page presented to the file receiver after clicking on the link displays an HTML page containing help information, and also includes a meta-refresh directive informing the browser to immediately redirect to an alternate page. This alternate page sets the HTTP content-disposition header to an attachment type, which informs the receiver's browser to display the Save/Open dialog.

For added security, we are developing an extension that embeds within the URL the user names and/or user groups of those who have permission to retrieve the file. This could be used in place of or in addition to the current link expiration mechanism. This extension will leverage the YouServ single sign-on infrastructure which supports one-click authentication, assuming login credentials have been provided once before during the user's browsing session.

## 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 sharing files through unmodified IM clients, and also his prototype implementation. Finally we thank the users of our system who have provided invaluable feedback.

## 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] J. Frank. 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.