

Providing Authenticated Access to Web Resources

Jonathan Esterhazy

Web Developer

University of Manitoba Libraries

Winnipeg, Manitoba R3T 2N2

1 February 1999

Abstract

Libraries are subscribing to an increasing number of web-based information resources. Most of these resources limit access based on lists of authorized IP addresses. This article explains how the University of Manitoba Libraries system uses a proxy server to provide eligible library patrons with location-independent access to such resources.

The Problem

The University of Manitoba Libraries, like other academic libraries, subscribes to a large and growing number of web-based information resources. Although we license these resources on behalf of "current students, faculty, and staff" of the University, access is typically limited to authorized sets of IP addresses. The result is that users who connect to the Internet through the University's network or dial-in modems have unfettered access to the licensed resources, while users with a non-University connection are denied access, even when they are included within the terms of the license agreement.

The number of eligible users who are excluded by the use of IP address restriction is significant. Distance education students, staff who work off-campus, faculty on study leave, and others who cannot connect through the University are the most inconvenienced users, but the large number of home users who have chosen to use an independent Internet Service Provider (ISP) are also affected.

Finding a way to allow all of these users to access IP-restricted resources was a high priority for us for several reasons. First, the University of Manitoba has a large distance education program, and we could not accept the exclusion of such a large number of users. Second, we have a large investment in web resources that use IP address restriction, and we plan to expand our offerings in this area aggressively. We wanted to be sure that the benefits of this investment would reach as many of our users as the license agreements permit.

Our Approach

The solution we developed is based on proxy server technology. First, we set up a proxy server on our campus network, and assigned to it an IP address that could access our IP-restricted resources. Then, we instructed users with external, unauthorized IP addresses to connect through this proxy server, rather than directly. The proxy server authenticates each user against our patron database, and then acts as a relay for their subsequent interaction with IP-restricted sites. This provides the users with the same access granted to users with authorized IP addresses.

The Proxy Server

The proxy software we run is called Squid (version 2.0). Squid appealed to us for several reasons: it was capable of using our catalogue's patron database as an authentication source, it could run well on the hardware we had available, and other departments at our institution already had some expertise with the program. The price was also right – Squid is free.

Squid is UNIX-based software, so we needed to find a suitable UNIX system on which to run it. We selected an available Pentium 133 PC with 32 MB of RAM and 1.0 GB of disk, and adapted it by installing Red Hat Linux 5.2. The hardware is not very powerful, but our examination of other Squid sites convinced us that it would be enough to handle the level of traffic we anticipated. We selected Red Hat over other versions of Linux because of its simple installation process and good system management utilities.

Determining appropriate cache settings was one of the most important configuration choices we faced while installing the proxy server software. Caching web pages to improve browsing performance is the primary function of most proxy server installations, but in our case it was secondary to the access-enabling capabilities. Our challenge was to find cache settings that would not cause performance problems for our low-powered server hardware. In the end, we set aside 8MB of RAM for cache memory and 100 MB of disk for cache storage – settings that have worked well, so far.

We also had to determine which users would be able to access the proxy server, and which sites they would be able to reach through it. Squid's *access control lists* provide extremely flexible and fine-grained control over access restrictions; they can allow or deny access based on authentication success, IP address, destination URL, and a variety of other criteria, and can apply these rules in any combination. We opted for simplicity: we allow access only to users who have authenticated, but we do not restrict the sites that they can access.

Custom Authentication Software

The authentication software we developed has two pieces: the *external authentication module* used by Squid, and the "Authsrvr" program that runs on our DRA catalogue server.

The external authentication module is a short, straightforward Perl script. When a user is asked to authenticate, Squid passes the username and password they enter to this script. The script opens a network connection (a TCP/IP socket) to the Authsrvr program, delivers the user's information, and then waits for a positive or negative response. Based on this response, the script tells Squid whether to grant or deny access to the requested resource.

The Authsrvr program runs on our catalogue server, and is specific to the DRA system. After receiving the username and password data from the external authentication module, it locates the corresponding patron record, verifies that the password supplied matches the PIN on file, and confirms that the user's borrower class is one of those eligible to use the proxy server. If these checks are all successful, Authsrvr sends a positive response back to the waiting external authentication module; a negative response is sent if any of the checks fail. Extending Authsrvr to look at other aspects of the patron record is not difficult – we could alter the program to deny access to users with outstanding fines or overdue items, for example.

Browser Configuration

Automatic Proxy Configuration

Web browsers must be individually configured to use the proxy server. Most users accomplish this by entering the URL of our *automatic proxy configuration* file in the appropriate spot within their browsers. Automatic configuration files tell the browser when to connect through a proxy server and when not to, and the rules they use to do this can be very specific. Our configuration file is designed to minimize the load on the proxy server by directing the browser to use it only when absolutely necessary: only when the user is connecting to an IP-restricted site, *and* their IP address is not in the authorized range. An added benefit of this approach is that users only need to authenticate when they try to access an IP-restricted resource.

Manual Proxy Configuration

Manual proxy configuration is still an option for users with browsers that do not support automatic configuration (versions of Netscape prior to 2.0, and Internet Explorer prior to 3.02). It is less convenient for users, since they must authenticate for all of their web access, even to unrestricted sites, but it means that users are not forced to upgrade their web browsers to gain access to library resources.

Conclusions

The problem of how to provide remote access to licensed web resources has provoked a great deal of discussion and research in the library community, particularly among academic libraries. Much of this discussion has focussed on the complex problems associated with providing trustworthy authentication systems at the consortium or global level. The effect seems to have been to discourage libraries from looking at the much more immediate, and imminently solvable, problem of enhancing access at the single-institution level. Our work with proxy servers has shown us that there is a practical, effective, and remarkably inexpensive way to get around the limitations of IP address restrictions.