

## **INTRODUCTION**

There are three major issues that have to be solved if and when, an organization wants to setup a WWW site in intranet with enough security to hold confidential material. There has to be a way to authenticate users' identities, place users into groups, and secure communication so that information doesn't fall in wrong hands.

## **USER AUTHENTICATION**

There are many reasons why you shouldn't solve this problem by creating a new user database from scratch. It's a big administrative task to handle user accounts: creating new ones, resetting passwords, removing old ones, updating their information, etc., and you probably won't receive any thanks from the users. For that reason, you should look for existing user databases, which cover your target audience. If your choice is NIS, it's quite easy to find UNIX based WWW servers, that support that authentication method. It should be noted as a security risk that it's reasonably easy to run crack against NIS passwd information. While NIS might fit some companies, it's more and more common that only intranet wide user databases are Lotus Notes and/or Windows NT domains.

One good thing in Lotus Notes and Windows NT systems is that when authentication is done against those servers, they are responsible for setting up alarms and locking accounts, if some third party tries brute force methods for cracking users passwords. As a price for this one, we also have to find ways to encrypt and minimize the traffic between WWW server and authentication server. One way to do this is to ensure that your WWW server caches successful username-password pairs for some period. This will also improve performance without compromising security.

If you wish to use the Lotus Notes authentication system, there are two choices for password: the one for Lotus Notes client and a so called http password. In Lotus Notes client passwords, you would probably face problems with the fact that the password is stored in a Lotus Notes ID file and HTTP protocols don't support the idea that we would send it along with username and password. Lotus Notes http password is also problematic, because it's yet another password for your users, unless it's widely used in your organization and that way your users won't have problems with remembering it /5/

Windows NT domains used to be inaccessible from other than Windows systems. Thanks to the great work that was done by Samba Team people, we nowadays have at least a couple of libraries for those, who wish to code the system themselves. Apache WWW server also has at least two modules that offer SMB authentication. When our project started, neither one of those modules was available, so I had to implement one of them. One issue, that some of the implementations haven't taken into account, is that in international corporations, it's unrealistic to expect that all users would be in one big domain. For this reason Windows NT domain based solutions should be able to support multiple domains and multiple domain controllers within those domains to be scalable and robust. /1/ /8/ /11/ /12/ /10/

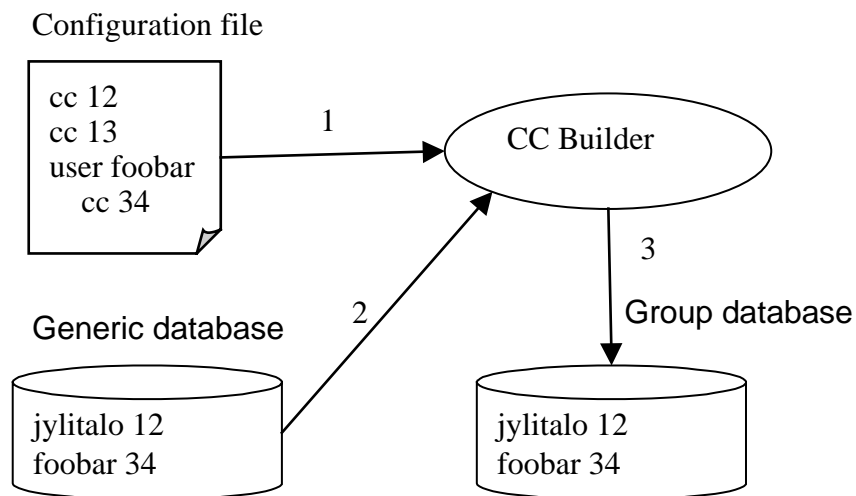
While it's nice to place responsibility for user authentication to other systems, you should have some way of creating temporary user accounts, i.e. search engines indexing routines will need temporary username and password, if you wish to index pages that require authentication. Otherwise you would need to create unnecessary user accounts, whose usernames and passwords you would have to write into files in plain text, which would definitely compromise security.

## ***USERS GROUPING***

While you can handle access to WWW pages by listing usernames, who are allowed to access them, it's practically impossible to keep those lists up to date in a dynamic organization with thousands of users, who rotate jobs. For this reason, we considered that all access lists should be based on groups and groups should be collected from an existing database, which is kept up to date by some third party. Otherwise we find ourselves with an administrative burden. One might assume that database should be the same that is used to solve username and password problems.

In a NIS based environment, this actually can be the situation. However with Lotus Notes and Windows NT domains, we were unable to find any feasible way to collect that information. Even if we would have been able to do it, we would have faced an other problem. User databases are often formed as they are needed and admins don't necessary spend much time in monitoring if some users should be removed from some groups.

Assuming that your organization has a consistent usernaming scheme in a sense that the users have the same username in all systems, i.e. Windows NT, Lotus Notes, UNIX, etc. you can have a lot of freedom in choosing the database. You only need one generic database, where



**Figure 1** Group creation. 1) configuration file is read, 2) information is collected from the database and 3) information is written into our database.

you can find username and grouping criteria information for your users, i.e. in our case the solution was found to be an electronic phonebook that has people's SMTP (Simple Mail Transfer Protocol) addresses and cost centers, which we use as our grouping criteria. The only important thing in the chosen database is that there is a group that makes sure that grouping criteria information is kept up to date and such questions as whether or not the system is normally used for user authentication is irrelevant.

Even with the best databases, you can't totally depend on them. You have to be able to alter or add grouping information for individual users for WWW use, i.e. form a special group, which has all group managers in it. To achieve this, you should create your local copy from that database on a regular basis, i.e. run it as a batch job during nights. This will also improve your system's performance and robustness, because a local database is faster to access and it shouldn't be down without your knowledge about it.

## **SECURE COMMUNICATION**

While username-password matching and grouping are important issues, secure communication is the most important out of these three due to the nature of http protocol. In http protocol, whenever some page that requires authentication, is requested, the browser has to send a username and password to the WWW server as authorization. For this reason a surfer might have to send his password tens or hundreds

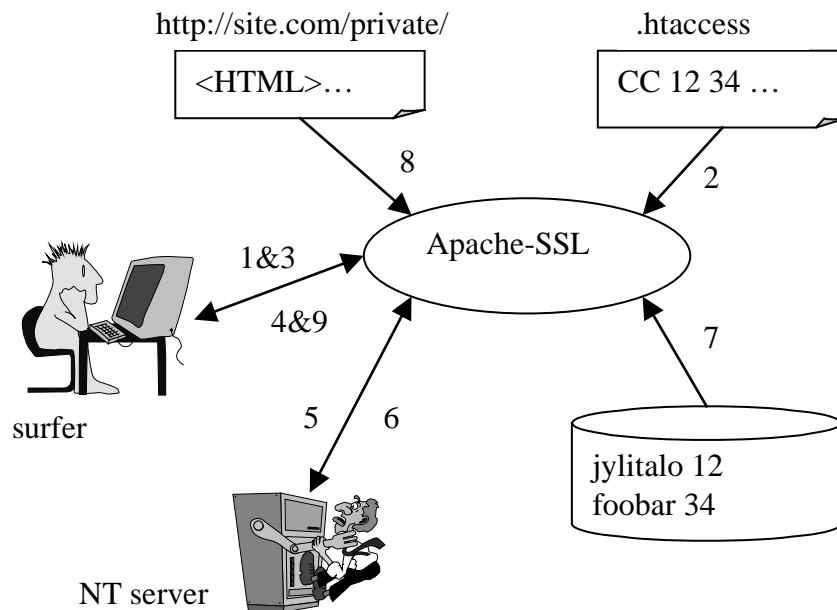
times during a day in comparison to a situation, when he opens one IMAP, telnet, X-Windows or other session, which is then kept open for the rest of the day. For that reason http is especially vulnerable to all kind of technical intrusion. /4/

The best way to get around this threat is to use an encrypted connection. Since encryption requires that used algorithms are known by the WWW server and browsers, possible solutions are pretty much limited to what SSL (Secure Socket Layer) can provide. However, there are two very different versions of SSL; the so called domestic version, which has a 128 bit key and can't be legally exported outside US and Canada, and the international version, which has only a 40 bit key.

Quite many organizations, e.g. some banks, seem to consider the international version of SSL to be secure enough for their needs, even though many cryptology people seem to have quite different view from it. Considering that decryption times without proper keys are often generalized to require a key length to 2nd power tests, the difference between versions is so huge, that administrators should search for ways to get a 128 bit key instead of a 40 bit key. While that 128 bit version can't legally be exported outside U.S. and Canada due to their export laws, it hasn't stopped Europeans and Australians from making their own implementation from it. SSLeay is one good example from such implementations; and it is a vital component in free and commercial versions of Apache WWW server with 128 bit SSL encryption. /13/ /2/

As I already mentioned, cryptology requires co-operation between the server and browsers. Since the most popular browsers, Netscape and Internet Explorer, are U.S. made, their international versions only support 40 bit keys. In Netscape, you can upgrade encryption from 40 bit keys to 128 bit keys by running Fortify, which will make necessary changes to Netscape binaries. One problem is that it has to be run for every browser in your intranet or you have to find a way to distribute fortified binary to all people and stop them from upgrading it with a non-fortified version of the software./3/

Once all the software is in place, you will have to get a SSL certificate for your WWW server. There are several CAs (Certification Authors), who can create certificates for you in exchange for a small annual fee and documentation that proves your authorization to ask for such a certificate (i.e. for your company). If you wish to avoid paper work, there are tools for creating your own certifications. The only problem in this scheme is that, when a user for the first time enters a secure WWW page in a server that has a self-made certificate, he gets a

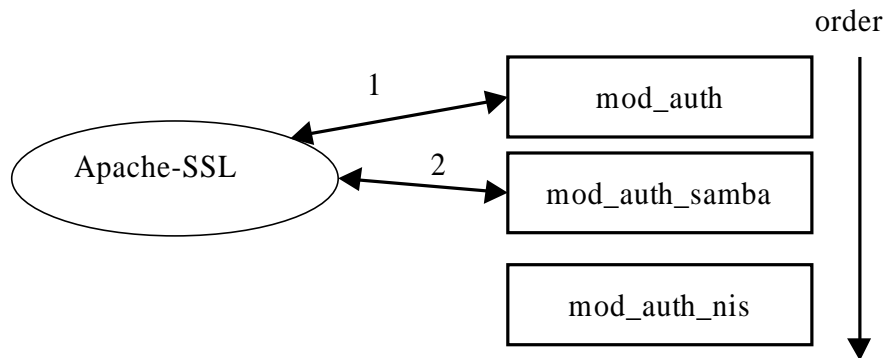


**Figure 2** Basic authentication scheme: 1) surfer sends a request, 2) Apache checks for .htaccess, 3) Apache request authorization, 4) surfer sends authorization (in SSL), 5) username and password are send for validation, 6) returns true/false, 7) group information is verified, 8) Apache reads the page and 9) sends it to the surfer

series of dialogs. If he decides to accept your certificate until it expires, it will take a long time until he faces those dialogs again.

## IMPLEMENTATION

I've already given some hints about our implementation. In past, we had been using Apache as our WWW server. Since Apache has 128 bit SSL version available, it was a logical choice for us to choose Apache-SSL as our next server. As a small R&D unit in intranet, we considered self-made certificates to be adequate for our purposes. The dialogs, that appear in the first time the user enters a secured web page with our self-made certificate are taken care as part of our normal induction to new employees. To make sure that people will use secure connections, we have placed a number of redirect directives into WWW server to ensure that all attempts to access a restricted area with http connection are redirected to https (secure http) connection. However, at the moment these redirects are updated manually, so there might be some pages that don't have proper redirects in place./1/  
/2/



**Figure 3** Order of authentication modules. 1) mod\_auth is asked for validation. It usually returns AUTH\_REQUIRED and control is passed on, 2) mod\_auth\_samba is asked for validation. It either gives OK or DECLINED.

Once security was in place, it was time to consider authentication issues. At that time, there weren't any SMB authentication modules available. However smblib did exist, and after studying how mod\_auth\_sys was built, it was a pretty straight forward operation to create my own authentication module (mod\_auth\_samba), which was based on smblib. Grouping was solved so that we have one application that every night collects usernames from those groups that we are interested in and creates one dbm database from those using the username as the key and groups as the data./9/ /10/

Afterwards we have started to take advantage of Apache's possibility to stack authentication modules. Our current implementation is such that we first check Apache's passwd and group file and if the user is not found from those, authentication responsibility is passed to mod\_auth\_samba. mod\_auth\_samba's implementation has also been changed. We've improved mod\_auth\_samba by replacing smblib with pam\_smb library and added a local cache file, where we store all successful username-password pairs with timestamps, so that we can minimize traffic between the WWW server and WINS servers. pam\_smb brings improvements in maintenance, security and portability issues, while local cache improves robustness, performance and security./6/ /11/

Based on the feedback that we've received from the users, they have been pleased about the easiness of the system and the amount of information they can get from intranet, now that security is in place.

## **FUTURE PLANS**

There are three clear tasks that we should implement at some point in future. The most urgent one is a system, which would give a graphical presentation about WWW pages access restrictions. While this would be a nice thing for system administrators, it's practically mandatory, when you negotiate access restrictions with managers, who don't necessarily have deep knowledge about WWW technologies, but who have to make the policy on these issues.

The second task, which is quite easy to implement, would be to create safeguard into the authentication module that would report all non-encryption authentications so that the webmaster can place redirect directive to avoid those cases in future.

The last task would be to make preparations for the future and implement a system that would support authentication tokens, which generate onetime passwords. This would greatly cut down the possible ways to misuse an intercepted username and password in case a new security bug is found in SSL etc.

## **REFERENCES**

- /1/ Apache WW server, <http://www.apache.org>
- /2/ Apache-SSL, <http://www.apache-ssl.org> for non-commercial and <http://www.apache-ssl.com> for commercial version.
- /3/ Fortify, <http://www.fortify.net/>
- /4/ HTTP 1.1 Proposed Standard RFC 2068, <http://www.w3.org/Protocols/rfc2068/rfc2068>
- /5/ Lotus Notes, <http://www.lotus.com/>
- /6/ mod\_auth, basic Apache authentication module
- /7/ mod\_auth\_nis, Apache authentication module for NIS, [http://med.jrc.it/~dirkx/mod\\_auth\\_nis.c](http://med.jrc.it/~dirkx/mod_auth_nis.c)
- /8/ mod\_auth\_samba, Apache authentication module for SMB authentication, [http://www.iki.fi/~jylitalo/apache/mod\\_auth\\_samba/](http://www.iki.fi/~jylitalo/apache/mod_auth_samba/)
- /9/ mod\_auth\_sys, Apache authentication for local /etc/passwd and /etc/groups, [http://www.ntb.ch/Pubs/mod\\_auth\\_sys.c](http://www.ntb.ch/Pubs/mod_auth_sys.c)
- /10/ smbllib, library which can be used as SMB API, <ftp://samba.anu.edu.au/pub/samba/smbllib/>
- /11/ pam\_smb, SMB as PAM (Pluggable Authentication Module), [http://www.csn.ul.ie/~airlied/pam\\_smb/](http://www.csn.ul.ie/~airlied/pam_smb/)
- /12/ Samba Team, <http://swamp.ios.chalmers.se/samba/samba.html>
- /13/ SSLeay, SSL package for Certificate Authorities and other SSL related activities, <http://www.psy.uq.edu.au:8080/~ftp/Crypto/>