Nowadays, Web security is as big a concern as ever -- so why not take advantage of Lotus Domino's many security features to ensure that your Web data is as safe and secure as possible?

Many customers use Lotus Domino for their intranet or Internet sites. Securing the Domino server in these environments is very important to ensure both the integrity of the data and the availability of the Web site, especially on the Internet.

In this article, we discuss how you can use Domino security features to enforce security in your Web environment. We begin with a quick overview of the Domino security model. Then we look at securing a Web server through Web authentication, server security, and data security. This article assumes that you are an experienced Domino system administrator.

Domino security model

Let's start with a brief review of the multi-layered Domino security model. (Previous developerWorks: Lotus articles have discussed Notes/Domino security in more detail, for example, "Overview of Notes/Domino security." But a quick overview will help lay the groundwork for some of the concepts we describe later in the article, particularly concerning Domino Web server security.)

In Lotus Domino, there are several different layers in which a user's access to data and/or resources can be restricted. Within each layer, there is a user authorization challenge. For example, to gain access to a Domino database, you must be authorized to access the Domino server on which the database resides. These access layers can be grouped into five more general categories:

- Server access
- Database access
- View access
- Document access
- Field access

The following sections review each of these groups.
Server access

Server access includes:

- **Network access**
  Network access is typically controlled by a firewall or a router. A firewall or router can block network ports, restricting access to the Domino server. For example, a firewall that blocks port 1352 blocks Notes client access.

- **User authentication**
  User authentication validates that the user name and password are correct. This is different from user authorization, which is performed by the Domino server and the Domino database.

- **Domino server**
  Domino server access is defined in the Server document in the Domino Directory. If a user is listed in a deny access list or is not listed in an allowed access list, then the user cannot be granted access to the server. In Lotus Domino 5, server access settings in the Server document were not applied to users accessing the server with a Web browser. However, Lotus Domino 6 allows the enforcement of the server permission for Web users (see figure 1).

**Figure 1. Enforce server access settings for Web users**
Database access
Database access is controlled by the Access Control List (ACL) of each database. The ACL determines whether or not users have access to the database and which level of permission they are granted. It is also possible to assign roles to users and to user groups. Applications can use these roles to grant users more specific levels of access. For example, you can give a user author access to the Domino Directory, but assign him the Group Creator role. This allows the user to create groups without editor access to the database (see figure 2).

Figure 2. Domino database Access Control List

View access
View access is controlled by the view properties. You can define which users can see the view; however, users may have access to other views that display the same documents. Mostly, view access provides user interface management rather than true data security (see figure 3).

Figure 3. View access control
Document access

Document access is controlled by one or more Readers fields that are saved as part of a document when it is created or modified. Allowed readers are listed in the field, explicitly naming the users and/or groups who can read documents created from the form. Without Reader access to a document, a user cannot see the document in a view or in the database.

Field access

Individual fields can be hidden or restricted by modifying the field properties (see figure 4).

Figure 4. Field hiding properties

![Field hiding properties](image)

Remember that when you use field hiding properties, the data items can be accessed through other mechanisms. In this case, field hiding provides user interface management rather than true data security. Also, fields can be encrypted with a private key. Any user that does not have the key cannot access the encrypted data. This provides security for the field as opposed to hiding the field for interface management purposes (see figure 5).

Figure 5. Field security options

![Field security options](image)
Web security

Now let’s look at a Web-based example using all five security levels. An ordering and shipping company wants users to place orders over the Internet. To securely set up their Domino server in their DMZ, they do the following:

1. Restrict Internet access to ports 80 (HTTP) and 443 (HTTPS).
2. Require all users to authenticate with the server and to verify server access before checking database access. Users are not allowed to read any Web pages on the server until they authenticate successfully.
3. Disable server browsing by setting "Allow HTTP clients to browse databases" to No.
4. Modify the ACLs on the application databases to give users author access only. This allows them to create new orders and to view their past orders.
5. Restrict view access. There are several views in the database, but users are restricted to the Catalog view and to the Orders view.
6. Add a computed Readers field that lists the user that places the order. The orders form contains several private fields that are only to be used by the shipping company to track orders. These fields are created with hide/when formulas to be hidden from the user placing the order.

Now that we have looked at an overview of Domino security, let’s talk about security concerns that overlay this multi-layered approach. A Web server that is accessible from the Internet is open to potential threats. These threats are specific to the nature of the server being accessible via the Internet. As a result, there are three main security topics that should be addressed:

- Web authentication
- Server security
- Data security

We now turn our attention to these topics, the associated threats, and how to eliminate the threats.

Web authentication

Web authentication is the process by which users identify themselves to a Web server. This is usually through a user name and password dialog prompt. Let’s suppose you set up a Web site, and you force all users to authenticate. A hacker determines a user name and continually hits your site with various passwords until discovering the right one, thus gaining access to your site. This is called a dictionary attack. The best way to prevent this type of attack is to enforce policies that make passwords difficult to guess. A password policy should include security measures, such as account lockout after failed password attempts, minimum password length or "strength," password aging, and password history. For instance, your password policy may be similar to the following:

- After three unsuccessful password attempts, the account is locked for 15 minutes. This makes it difficult to use login routines that quickly cycle through hundreds of different passwords until the right one is entered.
• Minimum password length of eight characters.
• Password must contain numbers and symbols.
• Password must be changed every 60 days.
• A user cannot use one of his or her previous 10 passwords.

Lotus Domino does not provide the password policy capabilities with Web authentication out-of-the-box, although Lotus Domino 7 provides the ability to enforce Internet password changes after a specified length of time. This enforcement of password changes is available in current versions of Lotus Domino by customizing the login form. Lotus Domino 7 also allows you to lock out a user account and to force a password change on the next authentication.

To enforce a stronger Internet password policy, you can develop a custom authentication DSAPI filter, or authentication can occur against an LDAP directory. Most LDAP directories provide the capability to enforce an adequate password policy.

**Types of authentication**

There are three types of authentication, each with its own unique security concerns. Basic authentication is simple user name and password authentication. Users are prompted to enter in their user name and password. There is also forms-based or token-based passwords. A user is prompted for a user name and password and given a session cookie or session header. This cookie or header is then passed onto that system (and sometimes several other systems), and the user is no longer prompted for authentication. This session cookie or header is given a timeout value. After the timeout value has been exceeded, the user is prompted for his user name and password once again. This can lead to some security issues should the token generation mechanism or header generation mechanism become easily counterfeited. All a hacker would need to do is pass the correct token or header, and the hacker would not be challenged for authentication.

A third type of authentication mechanism is certificate-based. This can be done by the user supplying an x.509 certificate to the Web site. This certificate is specific to that individual, and the user is not challenged for authentication. If the user loses the certificate or the certificate is stolen, hackers can access this information without being challenged for authentication. It is possible to use x.509 certificates to accept the user name, but still prompt the user for a PIN or a password.

The Domino server can allow a Web user to authenticate with many variations of the user name. There is a setting in the Domino Server document that controls what information can be used to identify a user for authentication. The default setting is "Fewer name variations with higher security." This setting allows a user to authenticate with any of the following: full hierarchical name, common name, any alias in the user name field of the Person document, Internet address, and uid (see figure 6).
The other setting is "More name variations with lower security." This setting allows a user to authenticate with the following: last name only, first name only, short name, common name, full hierarchical name, any alias in the user name field, Internet address, and uid.

The number of ways that a user can authenticate against the Web server has a direct impact on security. The more name variations that are allowed, the more open the Web site is to a dictionary attack. For example, if last name is a valid user name, in a large company it would be fairly easy to find a "Smith" or an "Adams." A hacker could easily try to gain access to the site with any number of common last names.

It is possible to restrict the allowed user names even further. By adding the NABWebLookupView parameter to the server's Notes.ini file, you can restrict which view is used to identify users. For example, you can create a custom view named ($web) in the Domino Directory that is sorted by full name. You can then set the NABWebLookupView=($web) parameter in Notes.ini and restart the HTTP task. After the task is started, the Web site is only accessible by full name. Similar functionality is possible with an LDAP server by modifying the authentication filter to list only the allowed LDAP field names. The default LDAP authentication filter is (|(cn=%*)(|(&(sn=%a)(givenname=%z))(&(sn=%z)(givenname=%a))))). This filter typically translates to authentication by full name (for example, John Doe).

**Web server security**

An important aspect of security is the physical location of the server. The server should be stored in a secure room. This room should have limited access controlled by electronic badges. The electronic badges serve two purposes: providing the method to actually lock and unlock the door and keeping track of who enters the room and when they enter the room. If an unauthorized user is allowed to gain physical access to the server, there is a potential for him to obtain secured data or to do severe damage to the server. ISO 17799 (or BS 7799) is a detailed security standard. It is organized into ten major sections; one of these sections is for physical security.

The server should also be secured from network access. This is just as important as physical access. The server should be behind a firewall or a router capable of blocking ports. Only ports 80
and 443 should be accessible to the server from the Internet. Internal users should only be able to access the server on ports 80 HTTP, 443 HTTPS, and 1352 NRPC. You may also need to open port 22 for SSH on Unix-based systems or port 3389 for Windows terminal services. However, it is not recommended that you open these ports; instead, perform administration tasks directly on the server. Both of these protocols are encrypted and use the operating systems method of authentication. This prevents most attacks that can occur directly on the operating system. By limiting the port access, it stops attacks on all the other services (such as file sharing) and FTP.

Now that we have secured the server, we should discuss the placement of the server in the network topology. An external Web server can be placed on the extranet or in the DMZ. The location of the server is dependent upon whether or not the server needs access to internal servers to replicate data or to authenticate users. For maximum security, the server should only be placed on the extranet. If hackers gain access to the server, they cannot gain access to any internal servers. If the server needs to access internal data and be accessible from the Internet, it is recommended that the server be placed on the internal network and that you use a reverse proxy. A reverse proxy protects the server from virus attacks and most hacking attempts because the reverse proxy server is the only server accessible from the Internet. Also, the use of reverse proxies can make the solution scalable and fault-tolerant.

The operating system should be continually updated. Administrators should monitor which security patches have been released and whether or not they apply to their systems. An important aspect of security patches is that depending upon the operating system, downtime may need to be scheduled. It is recommended that you have a weekly window for downtime to apply security patches. This maintenance window is necessary on Windows servers, but you can apply patches to most Unix-based operating systems while the Domino server is running. The only patches that cannot be applied while the server is running are patches that update the kernel or the TCP/IP stack. These patches require a reboot of the Unix operating system.

The server should contain operating system and Domino virus scanning capabilities. This ensures that any emails being sent or received do not contain viruses. Any files being stored on the server will not contain viruses; and any attachments stored in a Domino database will not contain viruses. This is very important because viruses can be destructive, limit availability of the server, and cause privacy concerns.

**Data security**

Now that we have secured the operating system and the server, we can talk about securing the databases on the Domino server. It is important to set up the ACL to databases with the proper access. Administrators should be listed with manager access and all users should have editor access or lower. The levels of access to databases that are available are shown in the following table.

<table>
<thead>
<tr>
<th>Access level</th>
<th>Allows users to</th>
</tr>
</thead>
<tbody>
<tr>
<td>Manager</td>
<td>Modify the database ACL.</td>
</tr>
<tr>
<td></td>
<td>Encrypt the database.</td>
</tr>
<tr>
<td></td>
<td>Modify replication settings.</td>
</tr>
<tr>
<td></td>
<td>Delete the database.</td>
</tr>
<tr>
<td>Access Level</td>
<td>Permissions</td>
</tr>
<tr>
<td>--------------</td>
<td>-------------</td>
</tr>
<tr>
<td>Designer</td>
<td>Perform all tasks allowed by lower access levels. Modify all database design elements. Create a full-text search index. Perform all tasks allowed by lower access levels.</td>
</tr>
<tr>
<td>Editor</td>
<td>Create documents. Modify all documents, including those created by others. Read all documents unless there is a Readers field in the form. If an editor is not listed in the Readers field, the user with editor ACL access cannot read or edit the document.</td>
</tr>
<tr>
<td>Author</td>
<td>Create documents if the user or server also has the Create documents access level privilege. When you assign author access to a user or server, you must also specify the Create documents access level privilege. Edit the documents in which there is an Authors field in the document and in which the user is specified in the Authors field. Read all documents unless there is a Readers field in the form.</td>
</tr>
<tr>
<td>Reader</td>
<td>Read documents in which there is a Readers field in the form and in which the user name is specified in the field.</td>
</tr>
<tr>
<td>Depositor</td>
<td>Create documents, but otherwise has no access with the exception of options to Read public documents and Write public documents. These are privileges that designers may choose to grant.</td>
</tr>
<tr>
<td>No Access</td>
<td>Has no access with the exception of options to Read public documents and Write public documents. These are privileges that designers may choose to grant.</td>
</tr>
</tbody>
</table>

It is also possible to restrict access to system databases. You can enable the "Don’t allow URL open" database property (see figure 7). This prevents the database from being opened from the Web. This is a feature introduced in Lotus Domino 6. If this is enabled in the Domino Directory, then the users can still authenticate. This is very useful when registered users on a server need to remain anonymous to other users of the server.

**Figure 7. Don’t allow URL open database property**

You should also delete the webadmin.ntf and the webadmin.nsf databases. This removes the risk of unauthorized Web administration access. An article written by Carl Kriger titled,
"Building Secure Domino Web Applications: How to Avoid 8 Development Pitfalls That Leave Your Application Wide Open," which was published in the July/August 2000 edition of The View, describes in detail database development best practices for a secure Domino Web application.

Secure Socket Layer (SSL)

We have secured access to the data, but what happens when a user is given access to the data, and a hacker can access the data packets as they are being sent across the Internet? To address this possibility, it is important to force all communications over Secure Socket Layer (SSL). SSL is a protocol that encrypts network communication to and from an end user and a Web server. This allows for secure encrypted Web browsing, ensuring that all data sent between the user and the server is encrypted, preventing hackers from accessing the information while it is in transit.

With Lotus Domino, you can create your own SSL certificates, or you can buy a third-party certificate. Most third-party certificates come with guarantees that the key will not be broken and offers to pay penalties if it is. SSL can have some performance impacts because every piece of data sent to and from the server must be decrypted. For high-volume Web sites, an SSL accelerator card or appliance should be used.

Other data security issues

Servers should be monitored for hacking attempts. Configure event monitoring and enable warnings generated when database ACLs are changed. This will notify you when changes are made, and you should verify that the change was authorized and planned. Additionally, enable Domino Web server logs. This keeps track of every GET and POST request to the server. Logging records incoming IP, the requested URL, and the referring URL. The servers should also be monitored for excessively high CPU and memory usage. This may be indicative of a Denial of Service type of attack or of a Web server that is overly utilized.

Denial of Service attacks are both a security threat and a stability threat. They have a direct impact on performance of the Web server because too much traffic can cause the server to crash. There are third-party products that are available to prevent Denial of Service type of attacks on a Domino Web server. These third-party products record every IP that accesses the server. If a particular IP generates failed login attempts repeatedly, the server will stop processing HTTP requests for that IP. Most of these products allow IP access to the server after a specified period of time. The blocking of processing HTTP requests for particular IPs prevents brute force password attacks. If an IP tries to access an account and repeatedly fails logins, that user IP would be blocked at the server. This is an effective way to protect password security.

Vulnerability scanning helps to identify any potential security holes in either the operating system or in the configuration of Lotus Domino. If the server CPU and memory seem to be running excessively high, the Domino Web logs should be checked for a Denial of Service type of attack.

Conclusion

In this article, we provided an overview of the Domino security model and some specific settings to make the Domino Web server more secure in a single-server environment. These tips will help
ensure that your Domino-hosted Web sites are as secure as possible, while still providing high performance, reliability, and ease-of-use to your users.
Related topics

- The developerWorks: Lotus article, "Overview of Notes/Domino security," presents a detailed summary of the security features available in Lotus Notes and Domino.
- Refer to the Lotus Domino documentation for detailed information about all the security features available in Lotus Domino.
- To learn more about Lotus Notes and Domino, see the Lotus Notes/Domino product page.

© Copyright IBM Corporation 2005
Trademarks
(www.ibm.com/developerworks/ibm/trademarks/)