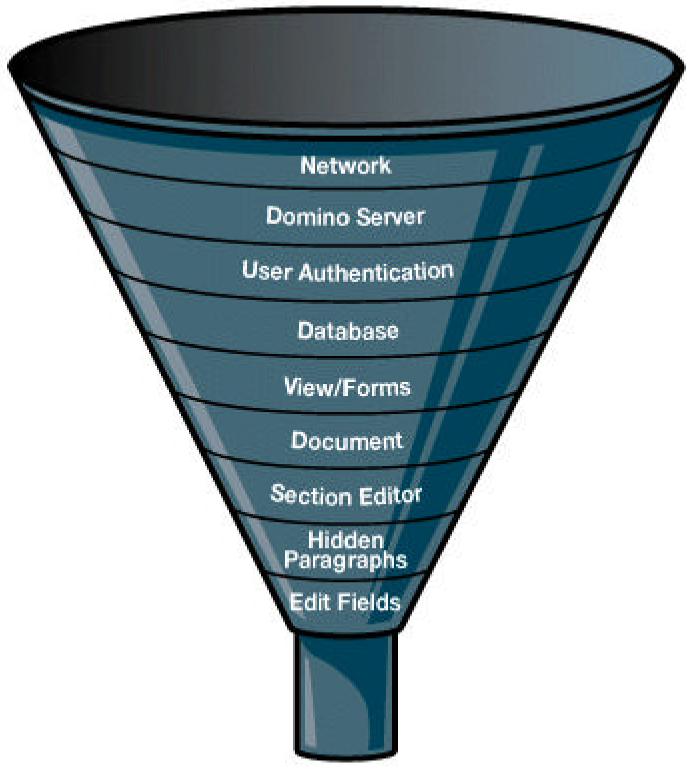
Designing a secure Domino app

by Howard Greenberg

Many companies are starting to use Domino applications to publish information to the Web. The Web is becoming a place to do business, not just an information appliance. As the Lotus ads say, “Work the Web!” Before you can work the Web, however, you want to make sure your data is safe and is only accessible by those you allow. This article examines the security issues in making your Domino databases available to Web users and assumes a basic understanding of Notes application development and security features.

You can think of the different levels of security as a funnel as shown in the graphic below. The further down the funnel you go, the more restrictive the access gets.



Access to the network

The first level of security is getting access to the network where the Domino server resides. Most companies have a firewall that prevents outside users on the Internet from accessing data on the internal network. If your Domino server is behind this firewall, you (or your administrators) have taken a big step towards making it is safe from outside sources. Typically, most firewalls prevent access to servers behind the firewall by blocking access to TCP/IP ports. The usual port used by the Domino Web server for Web browser access is 80. Domino uses port 1352 for Notes client communications. If the firewall blocks these ports, the outside world cannot access the Domino Web task using a Web browser or a Notes client.

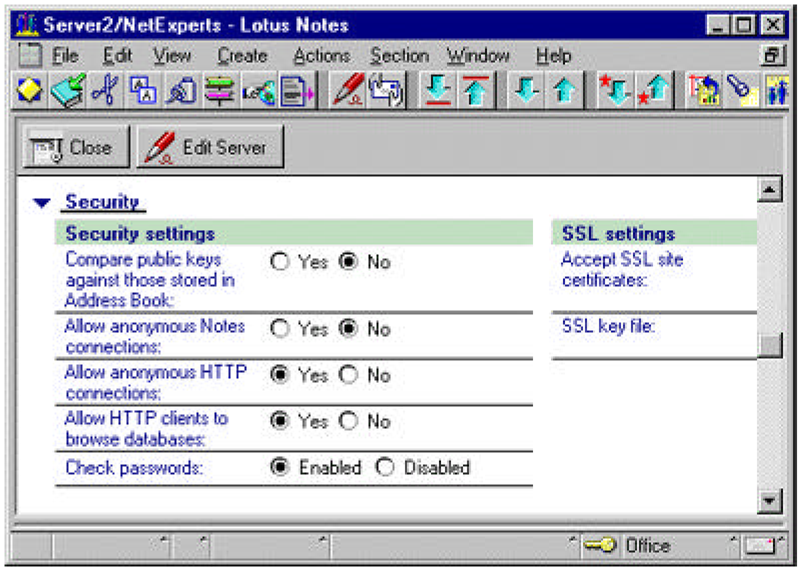
Of course, this scenario prevents users who have legitimate business from accessing the Domino server. If the Domino server has to be accessed by Web users, either the server must be placed outside the corporate firewall or TCP/IP traffic using port 80 must be allowed to pass through the firewall. Now the Domino server may be exposed to Internet hackers. In that case, all unneeded TCP/IP daemons like FTP and Telnet should be disabled to prevent access to the server through an unnecessary TCP/IP utility.

Access to the Domino server

In the Domino server, there are several settings that will affect access from the Web to the Domino server through a Web browser. These settings are in the Server document in the Public Address Book. The first setting allows you to choose whether anonymous HTTP connections are allowed. If "Allow Anonymous HTTP Connections" is checked No, users must log in when they try to access any database on the Domino server. If the setting is checked Yes, users do not need to authenticate when they access a database unless the database access control list (ACL) does not allow anonymous access.

Another security setting is to allow HTTP clients to browse databases. If "Allow anonymous Notes connections" is checked Yes, then users can use the http://ServerName/?OpenServer command. The Web users can then see a list of the databases on the server and try to access databases where security is set in the ACL to allow anonymous access.

The other setting that may affect security specifies a home URL in the HTTP Server section of the Server document. This is the default URL (database) that opens when a user accesses the Domino server without specifying a database to open. The application developer could specify a URL that will open a database using a particular view or document. While this does not affect security directly, it allows for easier access from the Web by opening up the database that will be the home page. This eliminates the need to present the list of databases and will allow the application developer to turn the "Allow HTTP clients to browse databases" setting to No.


Server2/NetExperts - Lotus Notes
File Edit View Create Actions Section Window Help
Close
Edit Server
Office

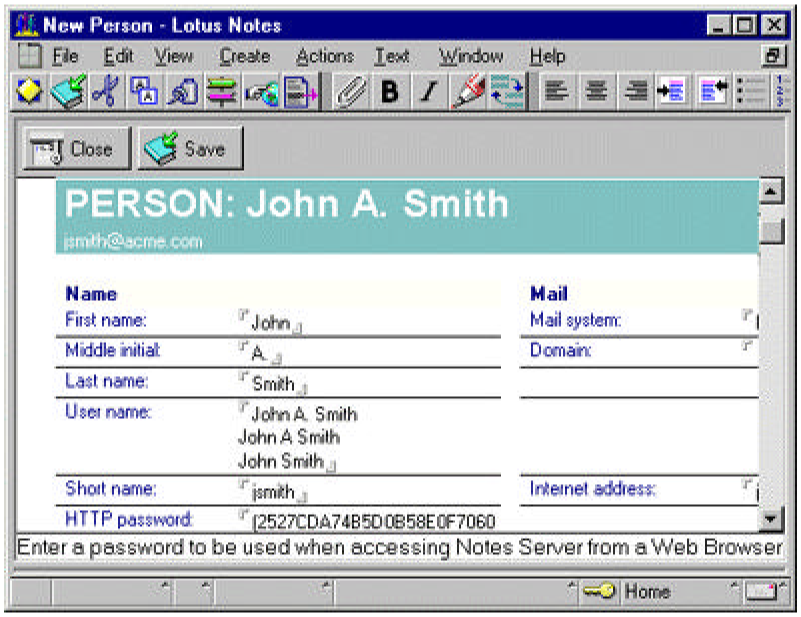
Security

Security settings		SSL settings	
Compare public keys against those stored in Address Book:	Yes [✓ No]	Accept SSL site certificates:	
Allow anonymous Notes connections:	Yes [✓ No]	SSL key file:	
Allow anonymous HTTP connections:	[✓ Yes] No		
Allow HTTP clients to browse databases:	[✓ Yes] No		
Check passwords:	[✓ Enabled] Disabled		

Access by user authentication

If the Domino server does not allow anonymous access or the Web user tries to access a database or other Domino object that does not allow anonymous access, the Web user is presented with an authentication dialog box. The Web user is prompted for the user name and password. The HTTP server will authenticate the user by comparing the password listed in the public address book with the password the user supplied. If the passwords agree, the user is considered authenticated. If not, the user is denied access. The ACL of the individual Domino database determines whether a person is considered authorized to access the database.

This process differs greatly from the way a Notes user is authenticated. The Notes client and the server each have an ID file. When the Notes user accesses a database on the server, both the client and server challenge each other to determine that they have a certificate in common. This challenge verifies that the certificate each computer claims actually exists and is not counterfeit. The Domino Web server uses a much less complex method of authentication. Domino searches through all the names listed in the public address book (using the names in the user name field) for the user name that was supplied. When it finds the first match, it checks the password that was supplied and compares it to the password that is listed for the name. The password is stored in hashed format to prevent readers of the public address book from accessing it. This Web password is in the field called "HTTP Password." It can only be seen while the Person document is in edit mode and continues to be stored in hashed format once the document is saved.



PERSON: John A. Smith

jsmith@acme.com

Name		**Mail**	
First name:	John	Mail system:	
Middle initial:	A.	Domain:	
Last name:	Smith		
User name:	John A. Smith John A Smith John Smith		
Short name:	jsmith	Internet address:	
HTTP password:	(2527CDA74B5D0B58E0F7060		

Enter a password to be used when accessing Notes Server from a Web Browser

Storing the Web users in the public address book raises several concerns. First of all, most companies would not want to store non-employee names in their public address book. When their Notes users click on the Address button of a memo, all of the Web users will be listed as well. A second concern is the size of the public address book. A large organization with 100,000 employees will not want to add names to an already large list. The solution is to have separate address books -- one for the employees and other Notes users and another (or several) for Web users. This can be done using the cascading address book feature of Domino. Separate address books are created using the template for the public address book.

On the Domino server, the notes.ini line "names=names" is modified to include all of the file names of the additional address books. For example, if you created a new name and address book called webuser.nsf, you would modify the notes.ini line to read "names=names, webuser". Make sure your domain's public address book continues to be called names.nsf and that it is the first listing in the "names=" line. When the Domino server searches the public address book to authenticate a Web user, it searches the names.nsf database first. If no match is found in that database, it searches for the name in the webuser.nsf database. Domino uses the first name it finds and checks the password. If you just wanted to have one copy of the webuser.nsf database on multiple servers, you can instruct Domino to find the webuser.nsf database across the network by including the canonical name of the server that is storing the Web user database. The following statement is an example of a notes.ini entry that instructs Domino to retrieve the other address books via the network:

```
NAMES=NAMES, CN=SERVER1/O=NETEXPERTS!!WEBUSERS
```

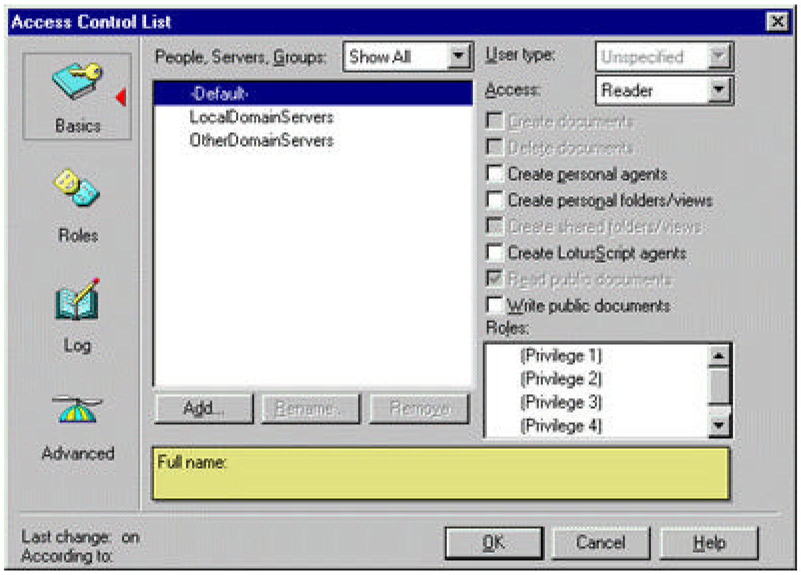
Many times a Web site requires users to register prior to accessing information. Lotus has a sample registration database that can be used by Web users to register themselves. The application then runs an agent that adds the new user to the public address book automatically. This application is available for download via the http://Notes.net web site.

Domino also supports server-side SSL (Secure Sockets Layer.) SSL encrypts data that is sent to and from the Web clients. It also detects if there was tampering with the transmitted message. The Web client is assured that the server is legitimate because the server has a certificate with a digital signature that is issued by a "trusted authority." (For details, refer to the **SSL: It's not just for commerce anymore** article in a previous issue of Iris

Today). Future versions of Domino will also support client-side SSL. This will allow the client to authenticate with the server without the Web users supplying a user name and password. The client will have a certificate that contains a digital signature, eliminating the need to store Web user names and passwords in the Public address book and the name will automatically be presented to the Domino server.

Access to the database

The next level of security is at the database level and is controlled by the access control list.



People, Servers, Groups: Show All

-Default-
LocalDomainServers
OtherDomainServers

User type: Unspecified
Access: Reader

- [ ] Create documents
- [ ] Delete documents
- [ ] Create personal agents
- [ ] Create personal folders/views
- [ ] Create shared folders/views
- [ ] Create LotusScript agents
- [x] Read public documents
- [ ] Write public documents

Roles:
[Privilege 1]
[Privilege 2]
[Privilege 3]
[Privilege 4]

Full name:

Last change: on
According to:

There are seven levels of access that can be assigned in the ACL:

1. No Access: Do nothing with the database
2. Depositor: Create new documents only.
3. Reader: Can read documents.
4. Author: Can read all documents and edit documents in which that person is listed in an authors field on the document
5. Editor: Can read and edit any document.
6. Designer: Can change the design of database plus read and edit any document (Note: A Web user cannot change the design of the database.
7. Manager: Can do the same as a designer plus change the access control list and delete the database. A Web user cannot change the access control list of a database.

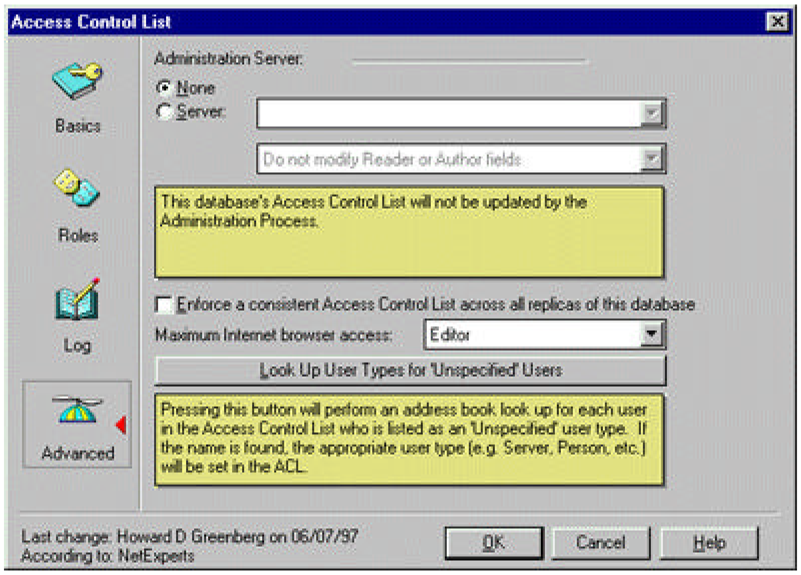
The author access level is normally used with a field that has the data type of Authors. Anyone who is an author in the ACL can create documents (as long as they are also given the right in the ACL). But once the document is created, only the users listed in the Authors fields of the document can edit the document. This field can be editable (one the user can change) or computed (one that uses a formula to determine the contents). If a document does not contain an Authors field, then anyone listed as an author in the ACL cannot edit the document. If you want to allow users to edit their own documents (such as discussion database documents), then use an Authors field that is computed when composed and contains the formula @UserName to allow the author to edit the document after it is saved. This method also assumes the user creating the document was authenticated. Otherwise, @UserName returns "anonymous" and any user who is not authenticated could edit the document. To force a user to authenticate, set the anonymous entry in the ACL to be "No access."

Entries in the ACL can be user names, group names, or roles. Web users can be listed in the ACL specifically by user name or in a group that contains their name. If the Web user is a member of multiple groups in an ACL, the user gets the highest level of access of all the groups of which he/she is a member. If the Web user is listed specifically in the ACL and is in one or more groups that are also listed in the ACL, then the user gets the level of access that is given to their specific user name and the access given to the groups of which the user is a member is ignored.

Each database ACL also has a Default entry, which controls the access level of all users who are not specifically listed in the ACL. If a user, including a Web user, is not listed specifically in the ACL or is a member of a group that is in the ACL, the user gets the access level of the Default entry. If the Web user has not been asked to supply a user name and password (was not authenticated), then the user gets the default access of the database unless an Anonymous entry was added to the ACL. All Web users who have not been authenticated are considered “anonymous.” An entry can be created in the ACL for anonymous that controls what level of access Web users have if they have not been authenticated yet. If there is no anonymous entry, then non-authenticated Web users get the default access.

If the Web user was authenticated, then the ACL is checked for the user name that was entered from the browser. If the Web user typed in “John A. Smith” and the Person document for John contains several ways to designate John's name including “John A. Smith,” John will be authenticated if the passwords match. But if the ACL entry in the database John tried to access had “John Smith” and not “John A. Smith,” then John does not have the level of access that was granted to “John Smith.” Instead, John gets the level of access granted to the default entry. This will change in Notes version 4.6. In 4.6, when a user authenticates from the Web, the name that is typed is searched for in the public address book. If found, the user is authenticated using the first name in the user names field in their Person document. This allows Web users some flexibility when typing in their name from the browser and gives administrators tighter control over the entries in the ACL.

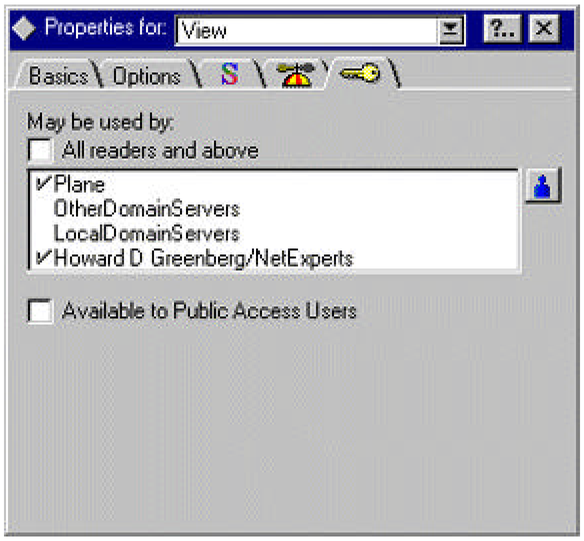
Another setting that affects Web users is the "Maximum access for Internet users" setting under the Advanced icon in the Access Control List dialog box. This setting is useful when you have users accessing the database from both the Web and Notes clients. Access for a Web user can be restricted to a certain level, yet when the database is accessed by the same user from a Notes client, the user gets whatever rights are listed in the ACL for the user.



Administration Server:
(● None) Server:
Do not modify Reader or Author fields
This database's Access Control List will not be updated by the Administration Process.
☐ Enforce a consistent Access Control List across all replicas of this database
Maximum Internet browser access: Editor
Look Up User Types for 'Unspecified' Users
Pressing this button will perform an address book look up for each user in the Access Control List who is listed as an 'Unspecified' user type. If the name is found, the appropriate user type (e.g. Server, Person, etc.) will be set in the ACL.
Last change: Howard D Greenberg on 06/07/97
According to: NetExperts

Access by views and forms

So far, we have discussed restricting access at the server, network, database, and user level. The next level of security is restricting the views a user can access. Each view has a view access list that lists the users who can access the view. The default is all users can access the view. The list is in the Properties box for the view when the view is in design mode. If you restrict access to certain views, make sure those views are not used in @DbColumn or @DbLookup formulas. The user needs access to the view to be able to look up information from them.


Properties for: View

Basics | Options

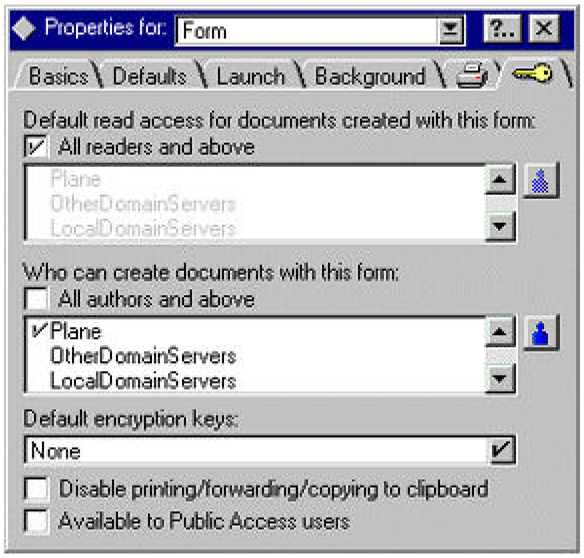
May be used by:

☐ All readers and above

✓ Plane
OtherDomainServers
LocalDomainServers
✓ Howard D Greenberg/NetExperts

☐ Available to Public Access Users

Designers can also restrict users from using forms to compose new documents. Like the view access list, form compose lists denote who can use the form to compose a new document. The default is everyone.


Properties for: Form

Basics | Defaults | Launch | Background

Default read access for documents created with this form:

☑ All readers and above

Plane
OtherDomainServers
LocalDomainServers

Who can create documents with this form:

☐ All authors and above

✓ Plane
OtherDomainServers
LocalDomainServers

Default encryption keys:

None

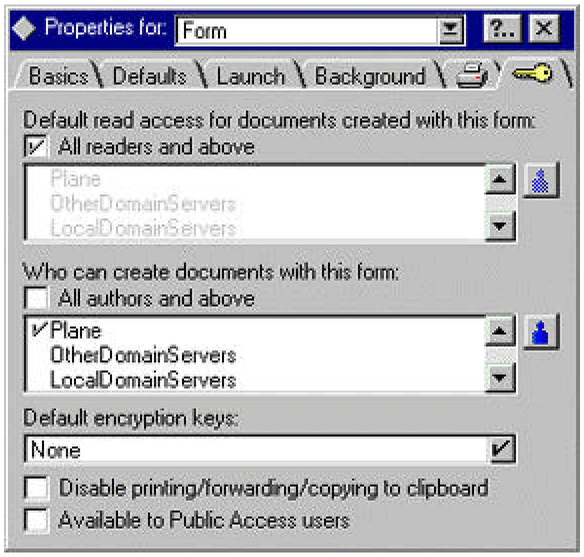
☐ Disable printing/forwarding/copying to clipboard

☐ Available to Public Access users

Access by document

Now the user has access to the database and we have determined what access privileges the user has in the database, what views the user can see, and what forms he/she can use to compose new documents. The next level of security is restricting who can see certain documents. This is done with a special type of field that has the data type of Readers. The Readers field defines who can read the document. This field can be editable (users can edit, and thus define who can read the document) or computed (compute the users who can be the readers of the document.) If there are multiple Readers fields, then all the Readers fields are added together to build a list of who can read the document. People listed in Authors fields are considered readers, too. The Readers field is powerful, since it applies to everyone, regardless of their access level in the ACL.

Below is a screen shot of a Readers field in use. This field, called
“Whocanread”, is a computed when composed field that has a formula to allow the groups “WebUsers” and “LocalDomainServers.” It is important to give your servers the ability to read the document if the document is to replicate to the different servers in your organization.



Default read access for documents created with this form:
☑ All readers and above
Plane
OtherDomainServers
LocalDomainServers

Who can create documents with this form:
☐ All authors and above
✓ Plane
OtherDomainServers
LocalDomainServers

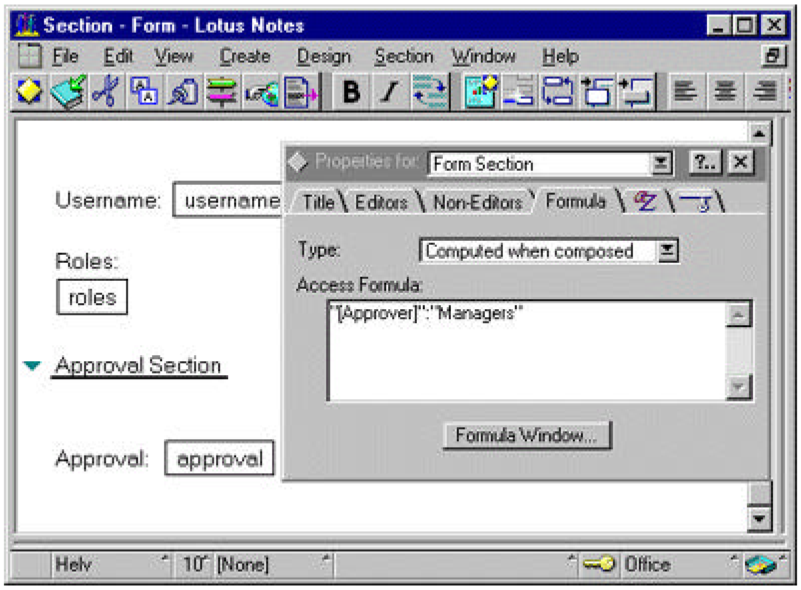
Default encryption keys:
None

☐ Disable printing/forwarding/copying to clipboard
☐ Available to Public Access users

Access by section editor

The next layer of security is controlled-access sections. A developer can define who can edit the fields in a section. This is useful for parts of the document that are used for approvals. A formula tab exists in the Properties box of the controlled-access section that defines the editors of the section. The formula can be editable, computed, computed when composed, or computed for display. The formula should return the name(s) of the users, groups, or roles who can edit that section.

To create a controlled-access section, highlight the static text and the fields that you want in the section. Then choose Create-Section-Controlled Access. Open the properties for the section, click the Formula tab, and enter the formula in the window. In the screen shot that follows, the formula allows users in the role of Approver and anyone in the group of Managers to edit the Approval field. Everyone can see this section, but only the section editors can change it.



Type: Computed when composed

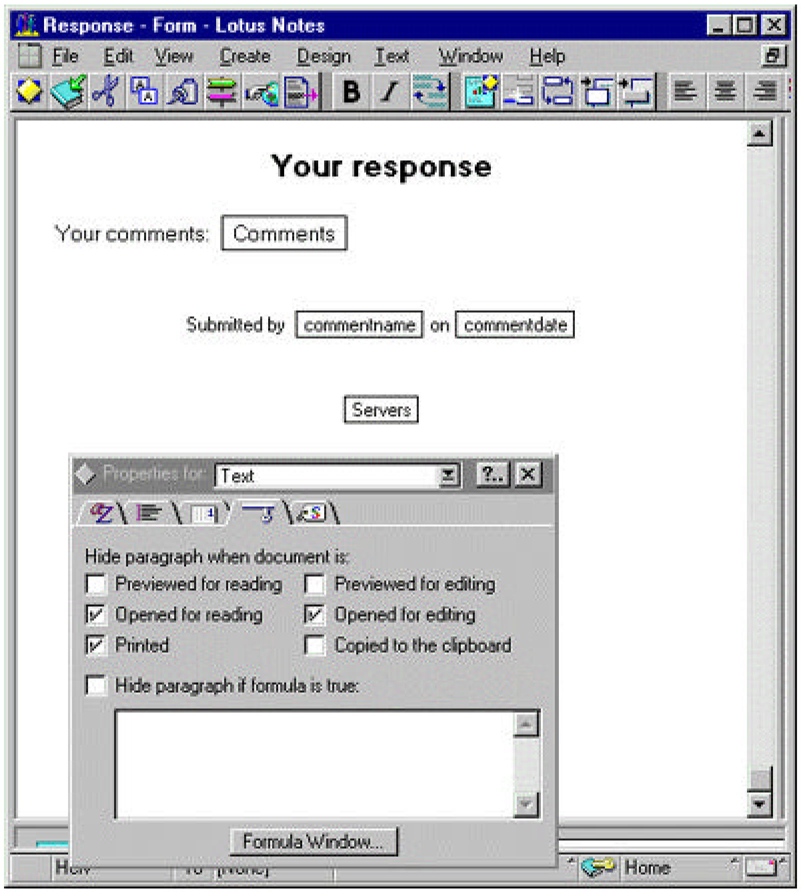
Access Formula:

```
"[Approver]":"Managers"
```

Access by hide/when

The next layer of security is controlling what information on a document is visible to the web users. This includes both static text and fields and is accomplished using the hide/when properties that control when information is presented to the user. Using a Notes client, the hide/when capability is not considered a security feature because a Notes client can always use the document properties dialog box to see the contents of all fields on a document, including hidden fields. But the Web browser does not have that capability. The Web browser user can only see what was transmitted to it from Domino and information hidden with a hide/when attribute is not sent to the Web browser.

There are several ways to use the hide/when properties. First, the hide/when properties apply to the paragraph. Therefore, the smallest level of information that can be hidden is one line, which if followed by a carriage return is considered a paragraph. Information can be hidden when the document is in read mode or edit mode. The application developer can also choose to hide the information from the Web user completely by choosing to hide the paragraph at all times. This is done by checking the selections to hide the paragraph when the document is in both read and edit mode. The hide/when properties can be set in the Properties dialog box by clicking on the “window shade” tab. Below is a screen shot of the hide/when settings that will control the viewing of the highlighted text and the field called Servers. The “Previewed for Reading,” “Previewed for Editing,” and "Printed" check boxes do not apply to a Web client.


Response - Form - Lotus Notes
File Edit View Create Design Text Window Help
Home
Formula Window...

Your response

Your comments: Comments

Submitted by commentname on commentdate

Servers

Properties for: Text

Hide paragraph when document is:

- [ ] Previewed for reading
- [ ] Previewed for editing
- [✓] Opened for reading
- [✓] Opened for editing
- [✓] Printed
- [ ] Copied to the clipboard
- [ ] Hide paragraph if formula is true:

Another way the document can be hidden is by writing a formula that controls when the paragraph appears. The formula goes into the "Hide Paragraph if formula is true" edit box and is saved when the syntax check box is selected. For example, if a paragraph should be visible only when "John Smith" is viewing the document, then the following formula could be used to hide the paragraph. (Note the exclamation point, which means "Not"):

@UserName="John Smith”

Another frequently used method is to hide a paragraph from all users accessing the document over the Web. To do this, the user role called $$WebClient can be used to determine if the user is accessing the database with a Web browser. The function @UserRoles returns a list of the user's roles. If the user is accessing the database from the Web, one of the user's roles will be the $$WebClient role. This is used in the following formula in the hide/when formula box:

```
!@IsMember("$$WebClient";@UserRoles)
```

In Notes 4.6, there will be an explicit checkbox called "Hide from Web Users" that will simplify hiding paragraphs from Web users.

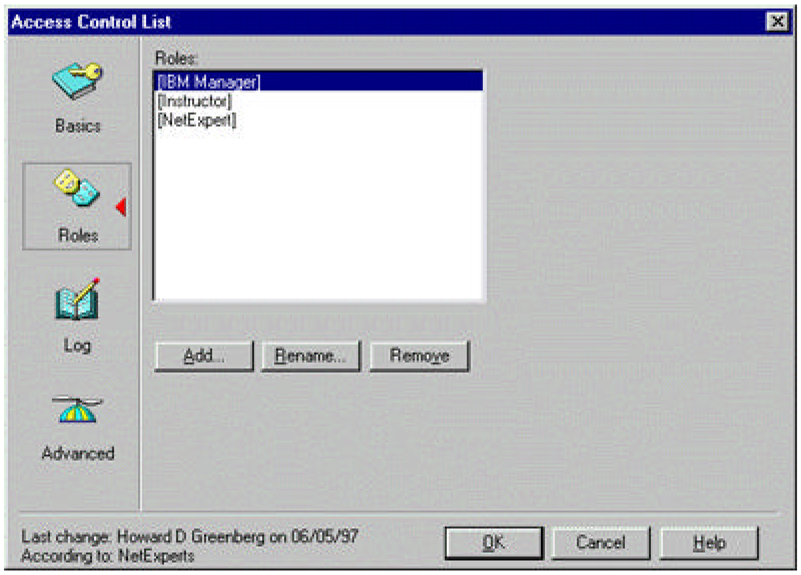
A way to ease implementing the various levels of security in Domino is the use of roles. In the example above, all Web users are assigned a role called "$$WebClient." The database manager can also create roles and places users or groups in a certain role. Let's say we have an application which has a document with an approval area. We want our approvers to be able to see the approval area but the other users of the database should not. For now, our approvers are Jane Jones, Mary Donahue, and Randy Holmes. We could write the following formula to hide the approval area:

```
applist:=”Jane Jones”: “Mary Donahue”:”Randy Holmes”;
!@IsMember(@UserName;applist)
```

Now we put our database into production. But a note comes from Human Resources explaining that Randy Holmes has moved to another job and is no longer an approver. The application designer must edit the formula. What we could have done instead is to create a role in the ACL called Approvers. Then we can add the three names to the ACL and assign them to the Approvers role. Now our formula could be:

```
!@IsMember(“[Approvers]”;@UserRoles)
```

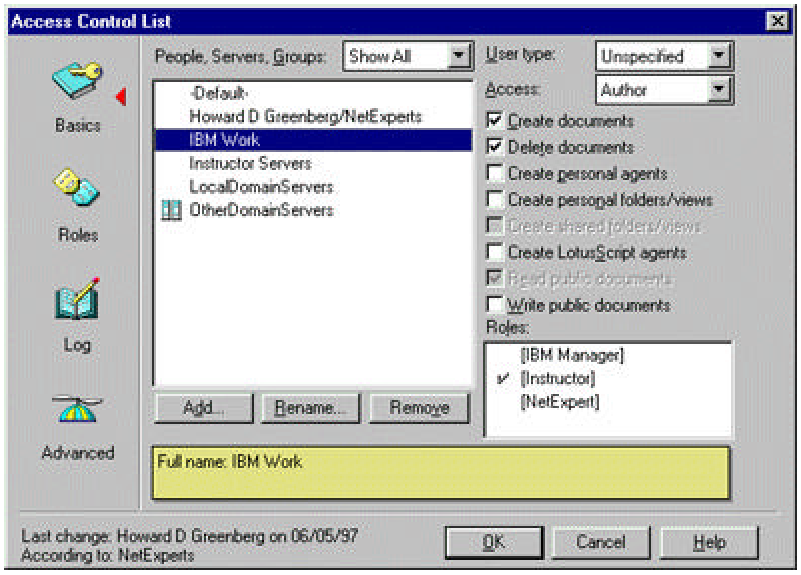
The role of Approvers gets the square brackets added when the role is created. @UserRoles returns a list of all the roles of the current user for the database. If a approver has to be added or removed from that role, the database manager can change the approver in the ACL. Roles are created using the Roles tab in the ACL. A screen shot follows.



Roles:
[IBM Manager]
[Instructor]
[NetExpert]

Last change: Howard D Greenberg on 06/05/97
According to: NetExperts

Then to place a person or group into a role, the person or group is highlighted in the ACL list and the appropriate roles are added or removed by selecting the roles in the Roles list box (lower right). In the screen shot below, the group IBM Work is in the Instructor role.



People, Servers, Groups: Show All

-Default-
Howard D Greenberg/NetExperts
IBM Work
Instructor Servers
LocalDomainServers
OtherDomainServers

User type: Unspecified
Access: Author

- [✓] Create documents
- [✓] Delete documents
- [ ] Create personal agents
- [ ] Create personal folders/views
- [ ] Create shared folders/views
- [ ] Create LotusScript agents
- [✓] Read public documents
- [ ] Write public documents

Roles:
[IBM Manager]
✓ [Instructor]
[NetExpert]

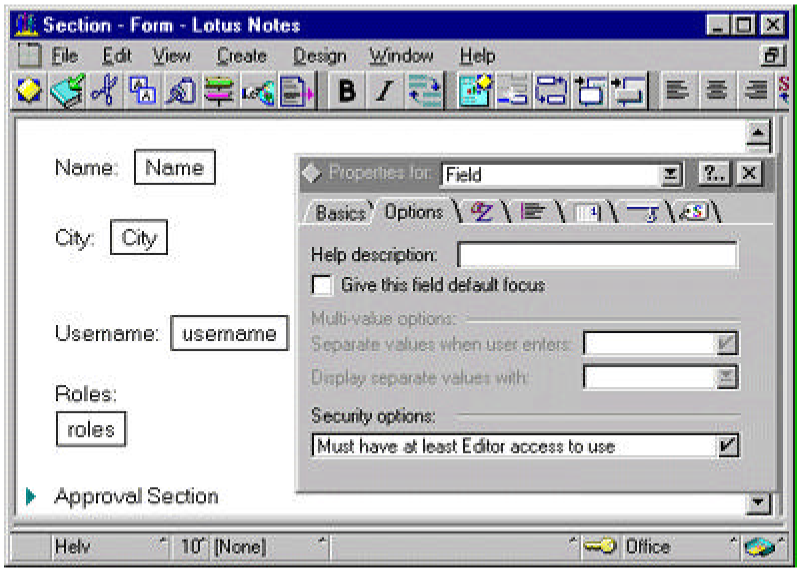
Full name: IBM Work

Last change: Howard D Greenberg on 06/05/97
According to: NetExperts

Roles can also be used in view access lists, form compose lists, readers and authors fields, and controlled-access sections and can greatly simplify the security administration of a database.

Access by field

The final level in the funnel is restricting access to edit certain fields. A field can be set to allow only users who have editor access or above to edit a particular field. This is set in the field properties under the Options tab by choosing "Must have at least editor access to use."



Help description:
- [ ] Give this field default focus

Multi-value options:
Separate values when user enters:
Display separate values with:

Security options:
Must have at least Editor access to use

Design elements to consider for security

Now that we have discussed the different levels of access to Domino data, you should also be aware of Domino design elements that may affect security when they are used.

Agents. Depending on the type of agent, the agent will run with the rights of the user who initiated the agent or the user who saved the agent. Domino Web applications can have agents that run before a document is opened and before a document is saved by using the $$QueryOpenAgent and $$QuerySaveAgent fields to specify the name of a LotusScript agent. The agents run with the access of the person who saved the agent, not the access level of the

Web user. This can be a double-edged sword. These agents can be used to build queries to either Notes data or to relational systems. As long as the person who last saved the agent can access Notes information, the information can be presented to the Web user using HTML and the print statement. The Web user does not need access to the information being accessed by the agent. In Domino 4.6, the designer will have the option to choose to run the agent with the access level of the server.id file or the Web user.

@DbColumn and @DbLookup formulas. Another area of concern will be the use of @DbColumn and @DbLookup formulas in fields on a form being accessed from the Web. Here the access level is determined by the Web user's access privileges. The Web user will need to be able to access the database, the view being used for the lookup, and must be able to see (if Readers fields were used) the looked-up documents.

Field encryption. If field encryption was used to encrypt any fields, those fields will not be visible by a Web user because they do not have the encryption keys necessary to decrypt the field.

Electronic signatures. Another Notes security feature that does not work from the Web is electronic signatures. A designer can choose to have a section signed when the document is saved but this feature will not work from the Web because there is no public key/private key for a Web user.

Server access list. The server access list in the server document that determines the users who can access a server does not work for Web users.

Domino has always had robust security features for an application designer or administrator to protect information. Many of these features also apply to Web clients accessing information via Domino. These security levels are unique to Domino and are easy to implement in applications. To implement similar security using a vanilla web server would require extensive CGI scripting. The extensive Domino security features make Domino an ideal Web development platform.

ABOUT THE AUTHOR

Howard Greenberg is President of NetExperts, Inc., a Notes consulting and training firm based in Boca Raton, FL (**http://www.netexpts.com**). He has over fifteen years of diversified experience in the computer industry specializing in client/server computing, personal computers and multimedia technologies with IBM. Howard has extensive training and experience on Lotus Notes, Novell NetWare, servers, workstations, and operating systems like DOS, Windows, NT, and OS/2. He is a level III Certified Lotus Notes Instructor and has taught Lotus Notes application development and system administration classes across the United States. He is also certified by Lotus as a Principal Application Developer and a System Administrator and by Novell, Inc. as a Master Certified NetWare Engineer. Howard holds degrees in Computer Science and Accounting and is a Certified Public Accountant.