Controlling the agents in your system

by Julie Kadashevich

[Editor's note: This article resides in "Iris Today", the technical Webzine located on the http://www.notes.net Web site produced by Iris Associates, the developers of Domino/Notes. This article focuses on technology specific to Notes 4.6. You can download Notes 4.6 from http://notes.net/gold]

Overview

Notes contains a powerful agent technology that gives you the ability to create automated tasks that range from simple to complex. To help you control the power of these agents, Notes allows you to specify restrictions on who can run an agent.

This article will introduce you to the different areas that control agent execution. The key thing to remember is that an agent can run only if the user has the appropriate rights -- rights in the database ACL where the agent will run, rights to run a particular type of agent, and rights to do what the agent is written to do. We'll also examine how the following factors come into play:

- How agents interact with database access control lists (ACLs).
- How agent restrictions vary according to the type of agent.
- How shared and personal agents differ.
- How embedded agents work when created by users with different restrictions.
- How restrictions change according to where the agent is run: on the desktop versus the server, in the foreground versus the background, or on the Notes client versus the Web client.

Then, we'll pull everything together by looking at some scenarios, and discussing the LotusScript properties you can use to help you control agents.

The basics of controlling agent execution

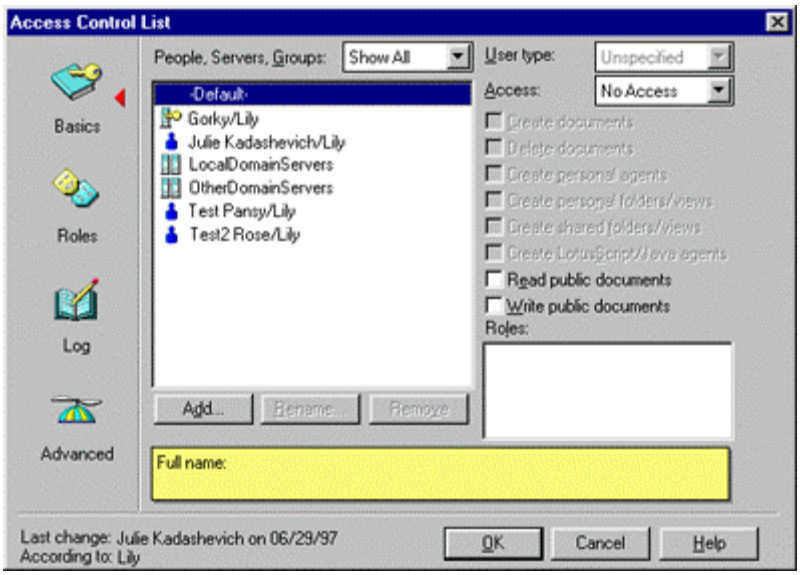
There are two basic areas that control whether a user can run an agent:

- The database ACL -- The user must have the appropriate rights in the database ACL where the agent will operate.
- Agent restrictions in the Server document -- The user must be able to run a particular type of an agent.

The tricky part is figuring out which user's rights are checked. An agent user may simply initiate the execution of the agent, or be the *invoker.* The user may also be the one that created or modified the agent, or the *creator.* The creator is sometimes referred to as the signer of the agent. The key point is that the invoker and the creator may or may not be the same user. Depending on the type of agent and where the agent is run, sometimes the rights of the invoker are checked, and sometimes it's the creator's rights that are checked. You'll learn more about these distinctions in the rest of this article.

Database ACL rights

Users with at least Reader access to a database can run agents, but the agents they run or create can only update documents to which they have at least Editor access. Once it is determined that the invoker of the agent has the rights to run the agent, the rights of the creator of the agent are used to determine what the agent can do.



Agent restrictions

After the user's rights in the database ACL have been checked, the next step is verifying that the user can run a particular type of agent. Different types of security controls exist for different types of agents, which we'll discuss according to what the agents runs, whether its personal or shared, and whether it's an embedded agent.

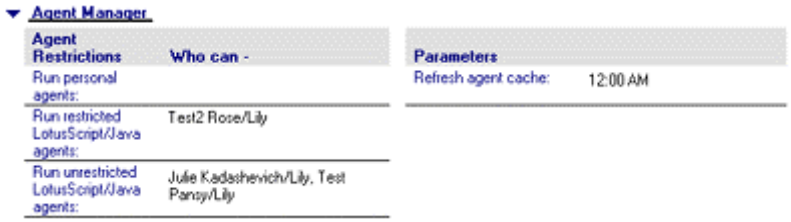
Agent types

When you create an agent, you must select what the agent will run: simple actions, formulas, LotusScript, or Java. Now, you'll learn about the security concerns for these different types of agents.

Simple agents and formula agents have no restrictions on who can run them. The documents they can access are dictated by the ACL rights of the database where they are created.

LotusScript and Java agents have two modes of operation: restricted and unrestricted. Restricted access allows a user to run agents with some features disabled (for example, file I/O); it is the more common access. Unrestricted access allows all features of LotusScript or Java to be used, which means that Notes security can be circumvented. Therefore, unrestricted access should only be given to trusted individuals.

You specify restrictions for LotusScript and Java agents in the Agent Manager section of the Server document, using the fields "Run restricted LotusScript/Java agents" and "Run unrestricted LotusScript/Java agents." The restrictions for LotusScript and Java agents are the same -- that is, you cannot specify restricted LotusScript rights and unrestricted Java rights for the same person at the same time. If either field is blank, then *no* users may run these agents with restricted/unrestricted access on this system. If either of the fields is filled in, then the user must be in the field to have restricted/unrestricted access.



Personal and shared agents

Any of the four types of agents can be personal or shared. A shared agent can be seen and run by anyone who has access to a database that contains the agents, including the agent creator. Personal, or private, agents can be seen and run only by the agent creator. If users have Designer access or higher to a database, they can create shared agents; otherwise, they can create only personal agents.

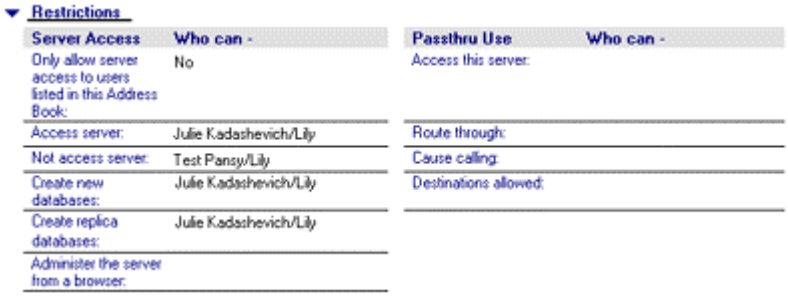
Whether a user can run a personal agent is determined by the "Run personal agents" field in the Agent Manager section under Agent Restrictions (shown in the previous screen). If the field is blank, then all users may run personal agents. If the field is filled in, the user must be explicitly listed in the field to be able to run personal agents. Personal agents cannot be accessed through the Web client.

Embedded agents

Agents can invoke other agents, and it is possible for the creators of these agents to be different. With embedded agents, the restrictions that apply to the top-level agent propagate to the embedded agents. (In contrast, on the Notes client, the database ACL rights of each agent creator control the agent execution. And, on the Web client, the rights depend on the setting of the "Run Agent as Web user" option, which you'll learn more about later in this article.)

Server access

An additional condition to consider is whether the user has the rights to do what the agent is written to do. For example, if the agent is written to update documents on another server, the user must have access to that server. For all agents, Notes uses the fields "Access server" and "Not access server" in the Restrictions section of the Server document to verify that users have the proper access rights to the server.



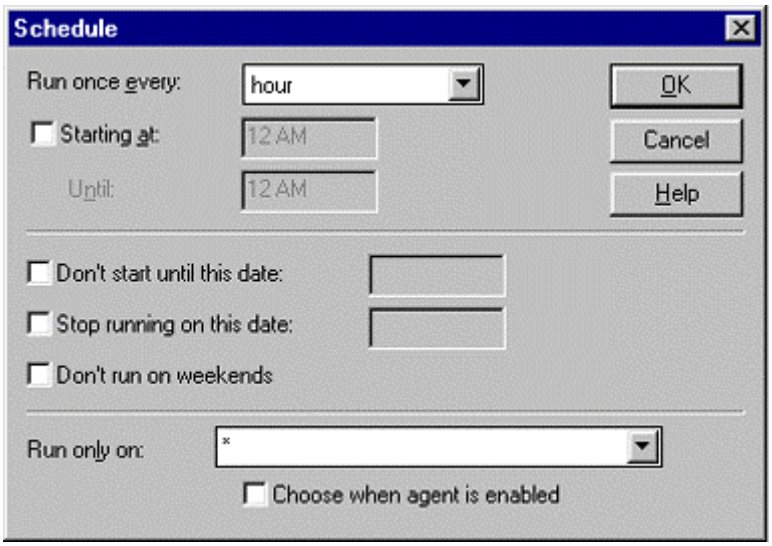
Restrictions

Server Access	Who can -	Passthru Use	Who can -
Only allow server access to users listed in this Address Book:	No	Access this server:	
Access server:	Julie Kadashevich/Lily	Route through:	
Not access server:	Test Pansy/Lily	Cause calling:	
Create new databases:	Julie Kadashevich/Lily	Destinations allowed:	
Create replica databases:	Julie Kadashevich/Lily		
Administer the server from a browser:			

If the user is listed explicitly in the "Not access server" field, the agent will not run. If the "Access server" field is blank, all users may access the server and run the agent. If the field is filled in, the user must be listed explicitly in the field to be able to access the server. Otherwise, the agent will not run. If the same name appears in both "Access server" and "Not access server" fields, the "Not access server" restriction takes precedence.

Changing the server

If you are given a database with an agent that is scheduled to run on a server that you don't have access to, you can change the server on which the agent is scheduled to run. To change the server; edit the agent, click the Schedule button, and select a different server name from the "Run only on" drop-down list. You can also specify an asterisk (*) to run the agent on the current server rather than on the server explicitly specified in the agent (that is, the server of the agent creator). If you specify an asterisk (*), the $MachineName field in Agent Properties is set to an asterisk (*).



Access to create new databases

For LotusScript/Java agents, a check is also made to verify that the user can create new databases. This information is found in the "Create new databases" field in the Restrictions section of the Server document. If the field is blank, then ALL users may create databases on this system. If the field is filled in, the user must be in the field to be able to create databases. If the user who does not have the rights to create a new database runs an agent that creates a new database, the agent will generate a run-time error.

Notes does not perform the "Server access" and "Create new database" checks if the agent is running on the desktop rather than on the server. Notes does not perform an ACL check if you are accessing a database on the Local server.

Running agents in various situations

The agent restrictions you've been learning about may change according to where you're running the agent: on the desktop or server, in the foreground or background, or on the Notes client or Web client.

Desktop or Server

An agent is running on a server if it is part of an agent on a server-based database, and the agent has one of the following triggers:

- When new mail arrives
- When documents have been created or modified
- On schedule hourly, daily, weekly, or monthly

When an agent runs on a server, all restriction and ACL checks are operational, except when the server is "Local." If the user has a local server, he/she can schedule agents to run on the Local server in two ways:

- Create an agent in the database on the server and refer to it as "Local" (and not by its true name).
- Select "Local" when creating an agent by clicking on the Schedule button and then selecting "Local" from the "Run only on" drop-down list. When the agent is run on a Local server, the restriction checks are by-passed.

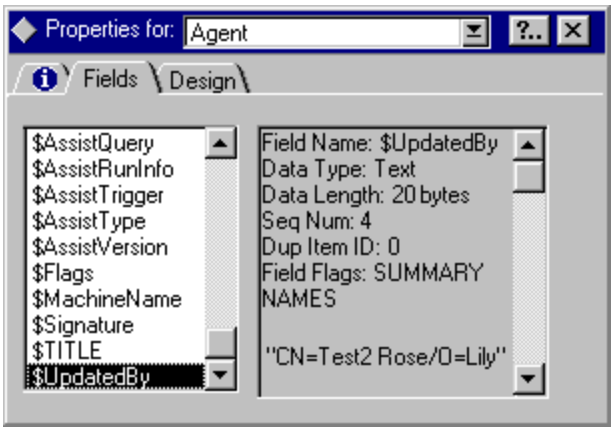
All other agents (invoked through the Actions menu, from the Agents list, or When documents have been pasted trigger) run on the workstation, regardless of whether the database itself is on a workstation or a server. When the agent is run locally on the workstation, the restriction checks are bypassed.

When an agent runs on the Web, Notes performs all the restriction and ACL checks, because Web agents always run on the server.

Foreground or Background
If the agent is triggered through the Notes user interface (manually through the Actions menu or from the Agent list), Notes is running the agent in the *foreground*. If the agent is invoked through a schedule or by an event (such as a document update or the arrival of new mail), Notes is running the agent in the *background*. The foreground/background concept does not apply to the agents invoked through the Web where agents always run on the Web server, which is a mix of the foreground/background environment.

On the Notes client
When the agent is invoked from the Notes client, the agent always runs with the rights of the agent creator. To see who the agent creator is:

1. Open the agent properties infobox.
2. Select the Fields tab, and look at the $UpdatedBy field.



To change the agent creator, you need to edit and save the agent while logged in as a person who will appear as the agent creator.

In order for a LotusScript/Java agent to run successfully, the agent creator needs to be listed in the appropriate restrictions field in the Agent Manager section of the Server document. If the creator is not listed in the Server document, Notes won't run the agent and will display the following error on the server console:

06/29/97 01:44:42 PM AMgr: Agent 'test' in 'test1.nsf' does not have proper execution access, cannot be run

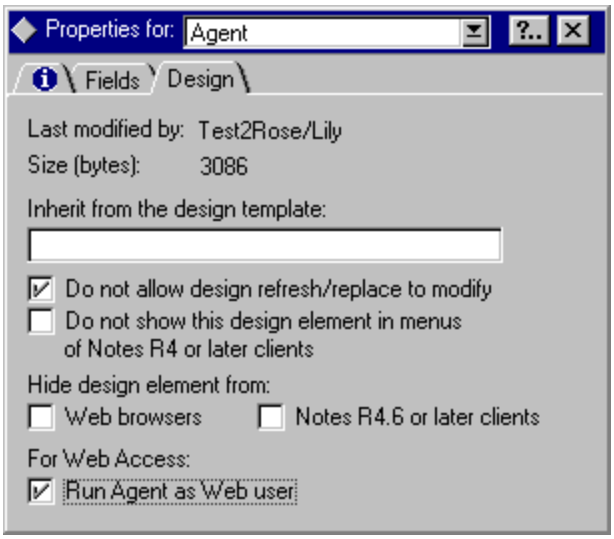
The person who is running the agent (the invoker) does not have to be listed in the Agent Manager restrictions, but should have at least Reader access to the database where the agent resides.

On the Web client
On the Web client, as with the Notes client, the agent restrictions are determined by the agent creator. However, Notes uses either the database rights of the agent invoker or the agent creator to verify database access.

By default, a Web user runs agents with the rights of the agent creator. However, you can specify that a Web user run an agent as him/herself by selecting the "Run Agent as Web user" option in the agent's properties:

1. Open the agent properties infobox.

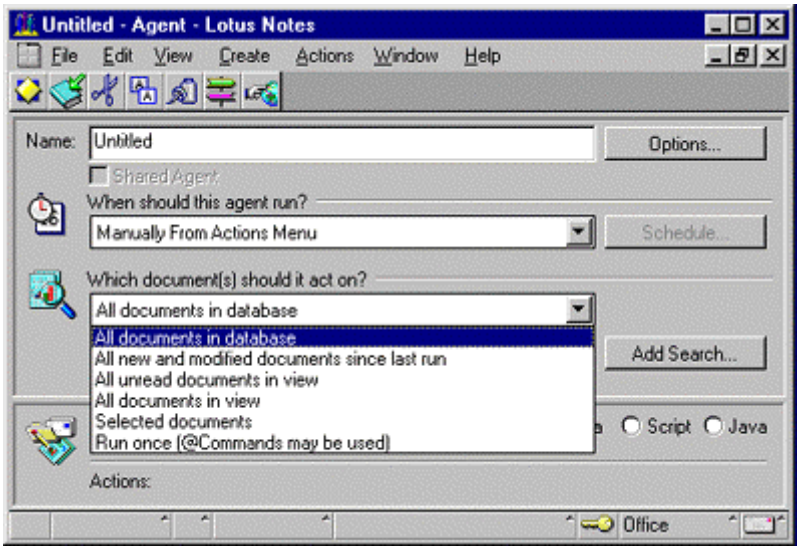
2. Select the Design tab.
3. Select the "Run Agent as Web user" box.



When "Run Agent as Web user" is selected, the Web user is prompted to log in with a name and password. Then, the user name is used to check for rights in the database ACL.

You can only create agents through a Notes client, not a Web client. Only shared agents can run on the Web client. The Web does not have a scheduling mechanism, so agents that run on the Web cannot be run at a pre-defined time. The "When should this agent run" option available during agent creation applies only to agents that are run from the Notes client. The option "Which documents should it act on" determines which documents the agent will operate on. It applies to both agents invoked from the Notes client as well as on the Web client. The following four options are not supported on the Web:

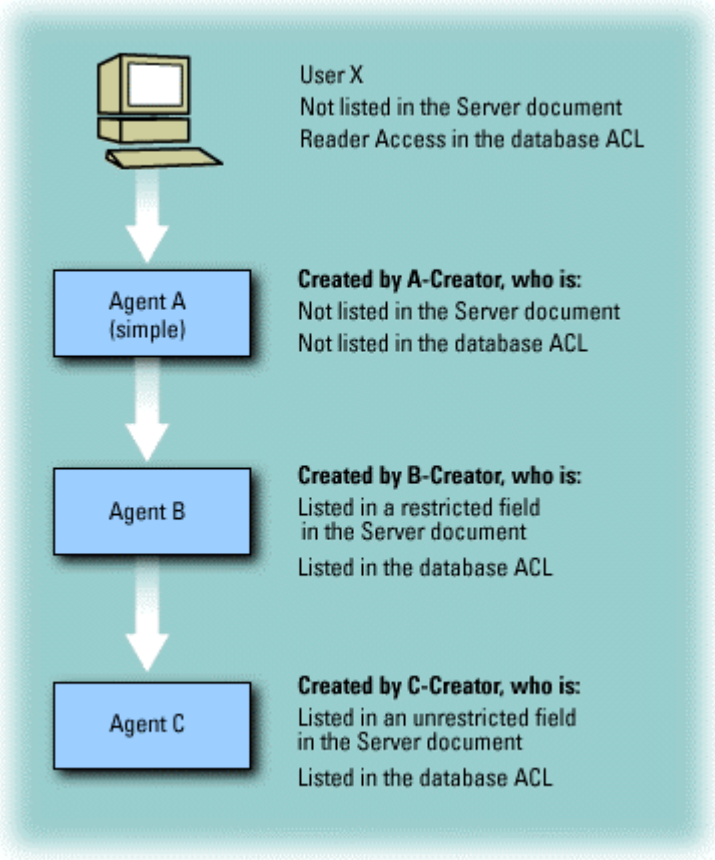
- All unread documents in view
- All documents in view
- Selected documents
- Pasted documents



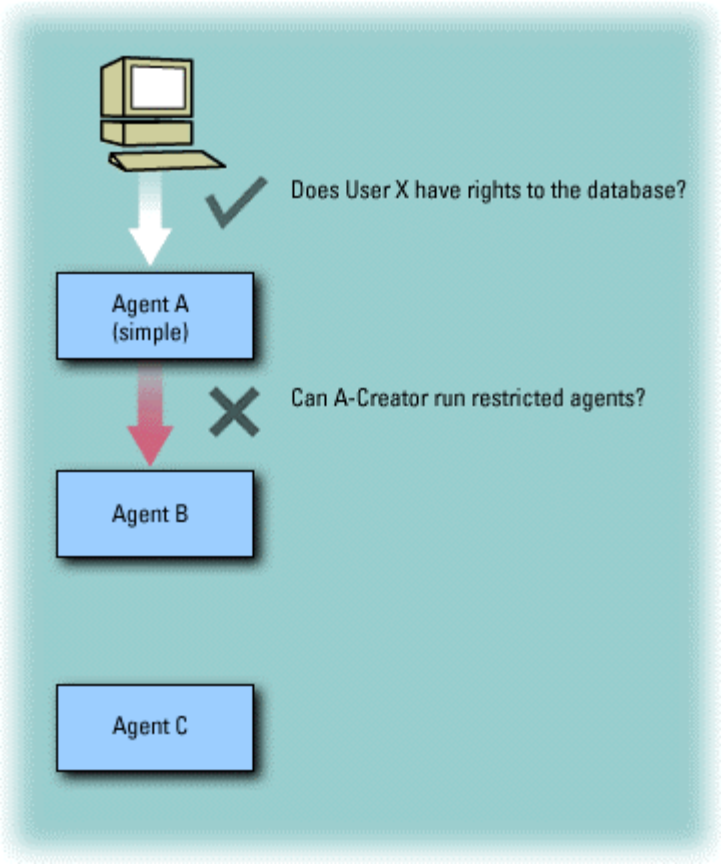
Scenarios

Now, we'll put the rules into action by examining some scenarios. In the following scenarios, let's assume that we have three shared agents: a simple agent A, which was created by A-Creator; agent B, which was created by B-Creator; and agent C, which was created by C-Creator. Agent A calls agent B, which in turn calls Agent C.

To initiate the first agent, we have User X, who is listed in the database ACL as a Reader. The database ACL also lists B-Creator and C-Creator with rights appropriate to the actions in their agents. (A-Creator does not need to be listed in the ACL, because agent A is a simple agent.) The Server document lists only B-Creator in a restricted field, and C-Creator in an unrestricted field.

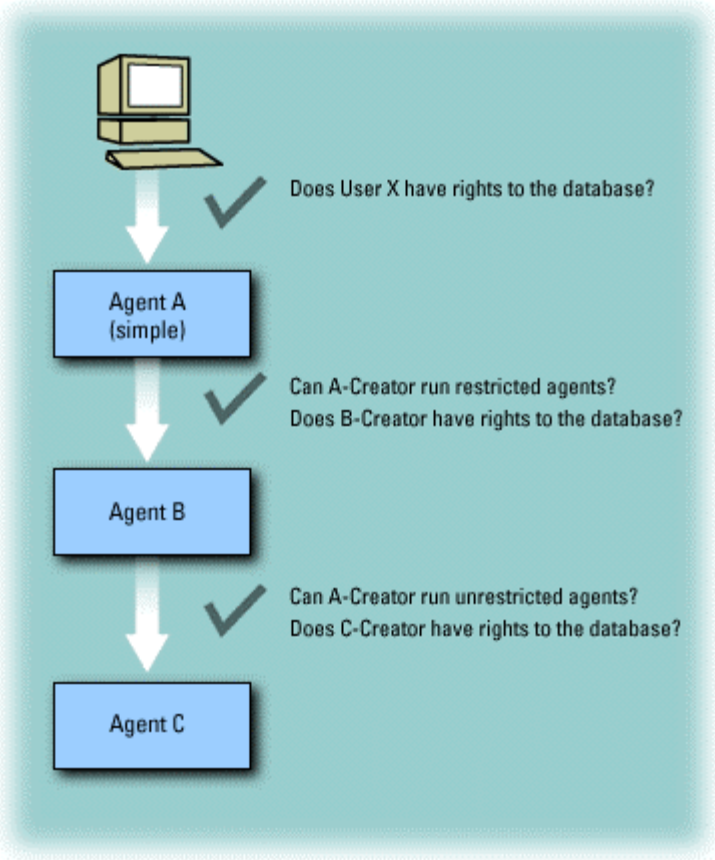


Scenario One:
User X, using the Notes client, enables Agent A to run. Agent A is scheduled to run every time a new document is created or an old document is updated. Since User X has rights to the database, s/he is allowed to run the agent. When Agent A starts, the rights of the A-Creator take over. Agent A prepares to invoke Agent B. The Agent Manager checks to see if A-Creator is allowed to run restricted agents. Because A-Creator is not listed in the Server document, the agent fails.



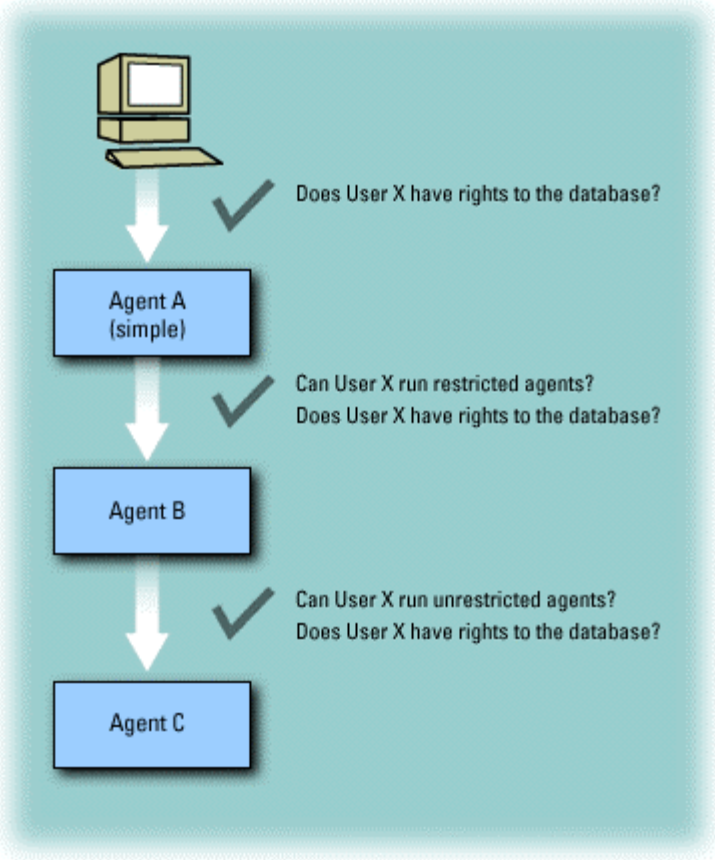
Scenario Two:
Let's add A-Creator to the unrestricted field in the Server document (under Agent Manager restrictions).

User X, using the Notes client, again enables Agent A to run. Since User X has rights to the database, s/he is allowed to run the agent. When Agent A starts, the rights of the A-Creator take over. Agent A prepares to invoke Agent B. The Agent Manager checks to see if A-Creator is allowed to run restricted agents. A-Creator has unrestricted rights, meaning that restricted rights are OK as well. Then, the Agent Manager checks to see if B-Creator has rights to the database to allow the agent to continue to execute. Agent-B prepares to invoke Agent C, and the Agent Manager checks to see if A-Creator has the rights to run an unrestricted agent. A-Creator does, so the Agent Manager checks if C-Creator has the rights to the database. Because C-Creator is listed in the database ACL, the agent successfully invokes all embedded agents.



Scenario Three:
Let's examine a similar scenario on the Web client, with the assumption that the "Run Agent as Web user" has been selected for all of the agents.

User X, using the Web client, clicks on a button that processes all new and modified documents. In order for this scenario to work, User X must be listed in the database ACL. While in Scenario Two, we checked if the different agent creators had rights to the database, we'll now check whether User X (the invoker) has the rights to the database. The rest of the scenario will work just as before.



If the "Run Agent as Web user" checkbox was not selected, this scenario would be identical to Scenario Two, where the agent creator's rights for the database were used in all database security checks.

LotusScript properties

To help you control agents, you can use properties of the LotusScript Session and NotesAgent classes. The following LotusScript properties should help you figure out who is the creator and invoker of agents, as well as other information relevant to controlling agent behavior.

LotusScript Session properties:

- *CurrentAgent* -- Agent that the program is running.
- *CurrentDatabase* -- Database in which the program is running.
- *EffectiveUserName* -- User ID of the creator of the current script. The user name that is in effect for the current script. For a script running on a workstation, this is the name of the current user. For a script running on a server, this is the name of the script's owner (the person who last saved the script). On the

Web this value depends on how the "Run Agent as Web user" option is set. If it is checked, this value will be the name of the Web user. If it is not set, it will be the name of the person who last modified and saved an agent (the agent's owner).

- *IsOnServer* -- True if the program is running on a server.
- *UserName* -- User ID of the current server or user. For a script running on a workstation, this is the name of the current user. For a script running on a server, this is the name of the server.

LotusScript NotesAgent properties:

- *IsEnabled* -- This property is intended for use with scheduled agents, which can be enabled and disabled. This property always returns True for hidden agents and agents that are run from a menu.
- *IsPublic* -- Indicates if an agent is public or personal. A public agent is accessible to all users of a database and is stored in the database. A personal agent is accessible only to its owner and is stored in the owner's desktop file.
- *Owner* -- The name of the person who last modified and saved an agent (the agent's owner).

Conclusion

To summarize, there are many factors that affect agent execution:

- How the agent interacts with the database ACL.
- What the agent contains -- simple actions, formulas, LotusScript, or Java. LotusScript and Java agents can be either restricted or unrestricted.
- Whether the agent is shared or personal.
- Whether the agent is embedded.
- Where you run the agent -- on the desktop or server, foreground or background, or from the Notes client or Web client.

If you have a problem running an agent, you want to learn exactly whose user rights are controlling the execution. You can first determine who the agent creator is by looking at the agent's $Updated by field. Then, you can determine the agent invoker at run time by using the LotusScript NotesAgent Owner property. Once you have the names of the agent creator and agent invoker, examine the Server document and the database ACL to make sure that the information matches up.

ABOUT THE AUTHOR

Julie Kadashevich came to Iris in March of 1997 from FTP Software, where she worked on Java and C++ mobile agent technology. For Notes Release 4.6 she has been focusing on the Agent Manager and Java.