

# Re: Directory Services, LDAP or similar

---

*Source:*

<http://coding.derkeiler.com/Archive/Delphi/borland.public.delphi.non-technical/2007-02/msg02446.html>

---

- *From:* "Leonardo M. Ramé" <[martinrame@xxxxxxxxxx](mailto:martinrame@xxxxxxxxxx)>
  - *Date:* Fri, 16 Feb 2007 14:57:57 -0300
- 

Thanks Marco and Arnie, this surely will save a lot of work for us.

Arnie escribió:

""Leonardo M. Ramé"" <[martinrame@xxxxxxxxxx](mailto:martinrame@xxxxxxxxxx)> wrote in message  
[news:45d5d533@xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx](mailto:news:45d5d533@xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx)

I and a group of developers of my company are defining the user authentication method of an application we are building.

In other projects, we managed the user authentication by creating tables that define all users and its allowed capacities, then the application queries that data to verify if a user has access to some feature or not. But yesterday one of my coworkers said that we can leave all that stuff in the hands of MS Directory Services or LDAP, i know we can control the user permissions to the whole application using that tecnology, but can we interact with a service like that to give access to some parts of the program, like a particular dialog or screen?

Thanks in advance,  
Leonardo M. Ramé  
<http://leonardorame.blogspot.com>

Hello Leonardo,

We were in a similar situation. I don't know if this will help you out or not and it is rather long; sorry. I'll describe what we did. You may be able to work out your own solution from there.

We have an application that uses DB servers. A user logs into our application using an appID and appPswd. We have a 'login' service running on a server. The above ID and password are sent to the service at login time. The service looks up the DB name, ID and password in an encrypted file. (1) If the user is authenticated, the service returns a DB login ID and password among other things. Now, the application can actually login to the DB.

The DB has a USERS and USERGROUP tables. We look the user up in the USERS table to find their group. The group specifies what they have access to. It might be none, read-only or

## Re: Directory Services, LDAP or similar

read-write permissions. This allows for removing menu items entirely or making them accessible. Further, the group specifies what menu items the user might have read-only or read-write permissions on. A user can belong to only a single group. This whole architecture was put together almost 10 years ago.

So, our largest customer wanted to pull us out of the stone age. Yes, the tail is wagging the dog again. They wanted centralized control over users. Not surprising because they are a very large organization. They are using Novell eDirectory at the enterprise level; yes it's LDAP. Our goal was to implement a solution such that one build is used for all customers with 'decisions' made at runtime. We already do that for three different DB servers; SQL Server, DB2 and Oracle. It can get sort of nasty.

We added some info to the Registry for our app. Specifically the LDAP server info. See (1) above. If the Registry info isn't present, we go traditional. If it is, we go LDAP. In the latter case, our login service forwards the user's credentials to the LDAP server. If authenticated, we get back our group name (defined by the LDAP admin). Our login service then looks up the user name in the encrypted file, without further authentication, and returns the DB login info. We lookup the user in the USERS table, get their group and go from there.

Obviously, the user must have a group assigned in LDAP that matches a group defined in our USERGROUP table. So, some amount of coordination is required. However, the network admin can change user properties including what group they belong to (in our app). The admin of our app can add or change group properties as required. So, everything is essentially centrally controlled.

I've left a lot of details out of the above.

In your case, I suppose a user could be assigned to one or more of your groups (if you have any). Enumerating groups returned from the LDAP server would determine the user's overall permissions.

I think using group definitions for access is the way to go. Then, a number of users can be assigned to the same group to get the same permissions. Trying to assign permissions on a per user basis doesn't sound too good.

I hope some of this may have helped. If not, sorry for the long response.

– Arnie