

## [Configuring Multiple Authentication Providers for SharePoint 2007](#)

Hi, my name is Steve Peschka, and I'm a Lead Architect within Microsoft Enterprise Services specializing in SharePoint-based solutions. I'm part of a team known internally as the "SharePoint Rangers," and we're usually "where the buck stops" when it comes to helping customers implement SharePoint. Since the topic of configuring multiple authentication providers for SharePoint 2007 (our nickname for the combination of Windows SharePoint Services V3 and Office SharePoint Server 2007) has become an FAQ via this blog as well as in public and private newsgroups, I'm posting our answer here for all to see.

Windows SharePoint Services (WSS) V3 contains several new features around authentication and authorization that make it easier to develop and deploy solutions in Internet facing environments, especially extranets. In the previous version of WSS, all security principals needed to resolve at some point to a Windows identity – either a user account or group. WSS V3 is built upon the ASP.NET 2.0 Framework, which allows the use of forms-based authentication (FBA) to authenticate users into the system. By riding on top of ASP.NET 2.0's pluggable authentication provider model, you can now support users stored in Active Directory as well as SQL Server, an LDAP directory, or any other directory that has an ASP.NET 2.0 Membership provider. Although WSS V3 will not ship with any Membership providers, Microsoft Office SharePoint Server (MOSS) 2007 will include an LDAP V3 Membership provider, and ASP.NET 2.0 includes a SQL Server provider. But if you want to use a directory and can't find a Membership provider for it, you can write your own! This is a key technology enabler for heterogeneous environments.

In a typical extranet environment, content will have two points of access: one on the intranet for employee use and the other on the extranet, where trusted partners can access specific sites, lists and libraries or individual items. Listed below are the WSS V3 features that support this scenario -- some are new while others are just terminology changes:

- **Web Application:** A web application is what was called a virtual server in the previous version of SharePoint. A single web application only supports a single authentication provider, such as Windows, Forms, etc.
- **Zones:** A zone is a way to map multiple web applications to a single set of content databases. It is also can be a division of authentication providers. For example, you can create a new web application, create a content database and configure it to use Windows authentication. You can then create a second web application and map it to the first. When you do that you need to assign a zone with which the second web application is associated, such as Intranet, Internet, Custom, or Extranet. The second web application can also use a completely different authentication mechanism, such as forms.
- **Policies:** A policy is useful in a number of different scenarios, including configuring a web application for forms authentication. It allows you to create policies to grant full access, read only access, deny write access or deny all access to a user or group on a web application. This policy grant applies to all sites in the web application, and it overrides any permissions established within individual sites, lists or items.
- **Alternate Access Mappings:** In the previous version of SharePoint, it wasn't as important in an extranet scenario to create an alternate access mapping (AAM) because SharePoint would look to IIS to get some of that information. In WSS V3, it's imperative to use AAM or things just flat out won't work. AAM is a way to define the different URL namespaces that are associated with a set of content databases. It effectively manages the zones relationship described above.
- **Authentication Providers:** So far I've described how WSS V3 uses the ASP.NET 2.0 pluggable authentication provider model using the Membership provider interface. As well, SharePoint also supports the Role provider interface, which enables you to surface attributes, such as group membership, about your users as well.

At a high level, creating an extranet solution in WSS V3 requires you to do the following steps. I'll walk through them briefly and then dive into more detail below. Since MOSS 2007 is built on top of WSS V3, all of the information below applies to MOSS as well. For this scenario, assume that you want to have an intranet style site used internally by your corporate users. They are all joined to your corporate Active Directory. In addition, you have a number of trusted partners to which you wish to give access via the Internet. Note that in this scenario I will not be touching on any aspects of securing your site with firewalls, proxy servers, segmented networks, DMZ Active Directory designs, security best practices around farm configuration, etc. You can read all about that in Joel's recent blog entry here: <http://blogs.msdn.com/sharepoint/archive/2006/08/08/691540.aspx>.

The process you would go through to build out such a site would be as follows.

1. After installing WSS V3 (or MOSS 2007) and having configured all of the services and servers in the farm, create a new web application. By default this will be configured to use Windows authentication and will be the entry point through which your intranet users will access the site. We'll refer to this site as **http://intranet**. Next, create a second web application. When you create the web application, select the option to *Extend an existing Web Application*. When you create your second web application, map it to the Extranet zone. Give it a Host Header name that you will configure in DNS for your extranet users to resolve against. We'll refer to this site as <http://extranet.contoso.com>.
2. If you haven't created and populated your directory of FBA users who will be accessing the site via the extranet, then you should do so at this time. For this scenario we'll assume that you are using FBA with the SQL Server Membership and Role providers that are included with ASP.NET 2.0.
3. Manually modify the web.config for the extranet site and add in the information about your Membership and Role provider (the Role provider is technically optional, but most implementations will use it). Add this same information into the web.config for the Central Administration site. Save both config files and do an IISRESET.
4. In the Central Admin site, go to the Application Management page and select the *Policy for Web Application* link. Add a user from your SQL Server directory to the Extranet zone for your web application. You should be able to type in the user name and resolve it, or use the People Picker dialog to search and find the user name. If everything is configured correctly then SharePoint will be able to resolve the user name you add. Give the user account Full access to the web application.
5. Navigate to the site using either entry point -- Windows or Forms-based authentication. If you use FBA, then you will need to sign in with the credentials of the user that was granted full access rights via policy. After you navigate to the site, go into Site Settings, People and Groups. From there you can add both Windows and forms users and groups to SharePoint Site Groups. Your users should now be able to access the site.

Now let's look at some of the above steps in more detail. Creating the web applications should be fairly straightforward using Central Administration, so I won't spend any time on that. The key takeaway here is that when you create the second web application, you need to make sure that you select the option to *Extend an existing Web Application* and map it to the Extranet zone. Also remember to give it a Host Header name that is in your external DNS -- this is the URL that external users will use to access the site via the Internet.

Next, you need to create the aspnetdb database used for storing membership and role information if you don't have one already set up. To create the database, do the following:

1. Open a command prompt and change to the .NET Framework directory (by default, it's C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727).
2. Run the following command: `aspnet_regsql -A all -E`
3. This will create the aspnetdb database on the local SQL Server. If you wish to install it on a different server, then run `aspnet_regsql /?` to determine the appropriate switch to use.

If you are creating your SQL Server provider database for the first time you will also need to create one or more users and optionally, one or more roles. These will be the security principals that you add to the Policy for the extranet web application as well as the SharePoint Site Groups. There are multiple ways to do this and a quick

search on the web will highlight some of those tools and methods. That's a bit out of scope for this already lengthy blog, so I'll continue on and assume that you've already created the users and roles for your SharePoint site.

Now we have our web applications as well as users and roles created in SQL Server, so we need to configure the web.config for the extranet and Central Administration web applications. The first step is to look for a connectionStrings element; if it doesn't exist then you can add it below the </SharePoint> and above the <system.web> elements. The new element should look like the following:

```
<add name="AspNetSqlProvider" connectionString="server=yourSqlServerName; database=aspnetdb;
Trusted_Connection=True" />
```

You'll want to take note of the *name* attribute above, because you will use that attribute name when configuring the Membership and Role providers. Add that information as follows:

1. Open the web.config file for your extranet web application in a text editor such as Notepad.
2. Add your connectionString element described above as the last item in the connectionStrings section in the web.config file.
3. Add the Membership and Role configuration information to the web.config file. It must be added below the <system.web> element and should look like the following:

```
<membership defaultProvider="AspNetSqlMembershipProvider">
  <providers>
    <remove name="AspNetSqlMembershipProvider" />
    <add connectionStringName="AspNetSqlProvider" passwordAttemptWindow="10"
enablePasswordRetrieval="false" enablePasswordReset="true" requiresQuestionAndAnswer="true"
applicationName="/" requiresUniqueEmail="false" passwordFormat="Hashed" description="Stores and
retrieves membership data from the Microsoft SQL Server database"
name="AspNetSqlMembershipProvider" type="System.Web.Security.SqlMembershipProvider,
System.Web, Version=2.0.3600.0, Culture=neutral, PublicKeyToken=b03f5f7f11d50a3a" />
  </providers>
</membership>

<roleManager enabled="true" defaultProvider="AspNetSqlRoleProvider">
  <providers>
    <remove name="AspNetSqlRoleProvider" />
    <add connectionStringName="AspNetSqlProvider" applicationName="/" description="Stores and
retrieves roles data from the local Microsoft SQL Server database" name="AspNetSqlRoleProvider"
type="System.Web.Security.SqlRoleProvider, System.Web, Version=2.0.3600.0, Culture=neutral,
PublicKeyToken=b03f5f7f11d50a3a" />
  </providers>
</roleManager>
```

4. Save and close the web.config file.

The name attributes of the Membership and Role providers are highlighted above. You need to note what these names are because you will enter them in Central Administration when you configure FBA for the site.

You also need to make the same exact changes to the web.config for the Central Administration site, with one minor exception. The roleManager element for the extranet web application looks like the following:

```
<roleManager enabled="true" defaultProvider="AspNetSqlRoleProvider">
```

You need to change this line to read as follows:

```
<roleManager enabled="true" defaultProvider="AspNetWindowsTokenRoleProvider">
```

This change is necessary because the Central Administration site still uses Windows authentication for the role provider -- that's why the `AspNetWindowsTokenRoleProvider` is set as the default provider.

Now you need to configure the Authentication provider for the extranet web application to use FBA. Open your browser and navigate to your farm's Central Administration site, click on *Application Management* and then on *Authentication Providers*. Make sure that you are working on the web application for which you wish to enable FBA. (If the correct application is not already pre-selected, click the *Change* button in the upper right hand corner of the page to select the application.)

You should see a list of two zones that are mapped for this web application; both should say Windows. Click on the link that says *Windows* for the web application in the Extranet zone and do the following:

1. In the *Authentication Type* section, click on the *Forms* radio button. The page will post back and expose two new edit boxes.
2. In the *Membership provider name* edit box, type in the name of your web application's Membership provider for the current zone. That is the value that was highlighted in the *defaultProvider* attribute of the *Membership* element above.
3. In the *Role manager name* edit box, type in the name of your web application's Role provider. That is the value that was highlighted in the *defaultProvider* attribute of the *roleManager* element above.
4. Click the *Save* button.

Your extranet web application is now configured to use FBA. However, until users, who will be accessing the site via FBA, are given permissions for the site, it will be inaccessible to them. To do this, you could go directly to the default zone (i.e. `http://intranet`) of the site, login with your Windows credentials, and add the FBA users. However, I'll describe an alternative approach because it's the one that you are most likely to use if you ever configure an application that only has one web application, which uses FBA.

To get started, open your browser and navigate to your farm's Central Administration site. Click on *Application Management* and then click on *Policy for Web Application*. Make sure that you are working on the extranet web application. Do the following steps:

1. Click on *Add Users*.
2. In the *Zones* drop down, select the appropriate Extranet zone. **IMPORTANT:** If you select the incorrect zone, you may not be able to resolve user names. Hence, the zone you select must match the zone of the web application that is configured to use FBA.
3. Click the *Next* button.
4. In the *Users* edit box, type the name of the FBA user whom you wish to have full control for the site.
5. Click the *Resolve* link next to the *Users* edit box. If the web application's FBA information has been configured correctly, the name will resolve and become underlined.
6. Check the *Full Control* checkbox.
7. Click the *Finish* button.

That's it -- that's all of the configuration needed! You can now navigate to either web application: `http://intranet` or `http://extranet.contoso.com`. Irrespective of which entry point you use, you can add, search and resolve both Windows and FBA users and groups and add them to SharePoint Site Groups. The People Picker is smart enough to know about all of the web applications that are mapped to the site and will try all of the authentication providers that those applications use.

Lastly, there are two other things for you to remember:

1. Resolving group names: The People Picker can only do wildcard searches for Windows group names. If you have a SQL Role provider group called "Readers" and enter "Read" in the People Picker search dialog, it will not find your group; if you enter "Readers" it will. This is not a bug -- the Role provider just doesn't provide a good way to do wildcard group searching.
2. Use Policies sparingly: The concept described above for adding a user or group via the web application Policy should only be used to provide a way for an FBA administrator to access the site. Policies are very

coarsely grained compared to the fine grain permissions that can be configured and granted within individual sites, lists and items. Once you've added your site administrator via Policy, all other users and groups should be added from within the site itself.

Admittedly, there are many steps involved in configuring multiple authentication providers for SharePoint, but I hope that by having read this blog entry, you now understand the reasoning behind each of the steps involved and are in a better position to implement or troubleshoot this particular SharePoint configuration.

Steve Peschka