Active Directory Module 1.2 for CMS 7.2-8.0

Administrator's Guide

How to install, configure, and use the AD module

Table of Contents

Chapter 1

Introduction

Sitecore CMS 7 and Sitecore® Experience Platform™ 8 (Sitecore XP) provide a solid, self-contained security model, based on the ASP.NET 2.0 provider model. Enterprise solutions may require a more complex security infrastructure. Many organizations have already set up a domain in some directory services, for instance, Active Directory.

The Sitecore CMS Active Directory module integrates the Active Directory domain with your Sitecore CMS solution. You can integrate the AD domain users and groups into Sitecore CMS as Sitecore users and Sitecore roles immediately after you install and configure the module. Furthermore, you can use custom properties from Active Directory to extend the user profiles in Sitecore.

This document describes how to install, configure, and use of the Active Directory module.

We recommended that you read the *Low-level Sitecore Security and Custom Providers* article before reading this document.

This manual contains the following chapters:

- **Chapter 1 — Introduction**
  This introduction to the manual.
- **Chapter 2 — Installation and Configuration**
  This chapter describes how to install and configure the Active Directory module.
- **Chapter 3 — Advanced Profile Configuration**
  This chapter describes how to configure the most important feature in the Active Directory module.
- **Chapter 4 — Other Features**
  This chapter describes some of the other features in the module.

Chapter 2

Installation and Configuration

This chapter describes how to install and configure the Active Directory module. Unlike other Sitecore CMS modules, the Active Directory module requires you to manually modify the `web.config` file.

This chapter contains the following sections:

- Installing and Configuring the Active Directory Module
- Users and Roles

2.1 Installing and Configuring the Active Directory Module

The Sitecore Active Directory module is distributed as a Sitecore Package. You must also modify some configuration files.

2.1.1 Prerequisites

Active Directory requires that port 389 is open by default:

- 389 — for client communications
- 445 — for Active Directory Profile Provider

Note
For more information about Active Directory Profile Provider, see the *Advanced Profile Configuration* chapter.

2.1.2 Installing the Sitecore Package

The Active Directory module is distributed as a Sitecore package.

To install the package:

1. In the Sitecore desktop click **Sitecore**, **Development Tools**, **Installation Wizard**.
2. The Installation Wizard will guide you through the installation process.

2.1.3 Modifying the Config Files

After you install the package, you must modify some configuration files to complete the installation.

You must modify:

- The `/App_Config/connectionStrings.config` file.
- The `domains.config.xml` file.
- The `system.web` and `sitecore/switchingProviders` sections of the `web.config` file.

Adding a Connection String to the Active Directory Domain

If you have decided to use the Active Directory module, you must have an appropriate Active Directory domain to connect with. In most cases this is a company domain which stores the entire security infrastructure of your company.

In the main `/App_Config/connectionstrings.config` file, add a connection string to the `<connectionStrings>` section.

The `connectionstrings.config` file can look like this:

```
<connectionStrings>
   <add name="ManagersConnString"
        connectionString="LDAP://testsrv/OU=Managers,DC=testdomain,DC=sitecore,DC=net" />
</connectionStrings>
```

In this example, *Managers* is just a sample organization unit. Replace it with the name of a real OU.

You can define a number of connection strings in the `<connectionStrings>` element.

Each connection string is defined by the `<add>` tag. This element has a number of attributes, but we only use two:

Attribute	Description
`Name`	The name of the connection string. Any entity using this connection string will address it by this name.
`connectionString`	The connection string itself.

The Active Directory module supports the LDAP format of the connection strings:

- Each string starts with the prefix `LDAP://`
- The prefix is followed by the server name, which the component should connect to. The slash `/` must follow the domain name.
- The last part of the connection string must contain the full path to the Active Directory container that the users and groups should be extracted from.

Important
To avoid some exceptions and for greater stability, specify the Active Directory server name as a Fully Qualified Domain Name (FQDN) with the port number. It should look like this: `ADServer.domain.name:389`.

The connection string is a path that looks like this:

```
LDAP://ADServer.domain.name:389/OU=Managers,DC=ADDomain,DC=company,DC=com
```

In this example, we assume that you have an Organization Unit called *Managers* in your organization and you wish to make the members of this unit available in Sitecore CMS. We also assume that the Active Directory server name is *ADServer* and that it is located in the `ADDomain.company.com` domain.

In this case the connection string looks like this:

```
LDAP://ADServer/OU=Managers,DC=ADDomain,DC=company,DC=com
```

If you want to use a sub unit, for example, *Support Managers*, the connection string is:

```
LDAP://ADServer/OU=Support Managers,OU=Managers,DC=ADDomain,DC=company,DC=com
```

If you want to use the entire domain, specify the following connection string:

```
LDAP://ADServer/DC=ADDomain,DC=company,DC=com
```

For more information about the format of the LDAP connection string, see the article *LDAP ADsPath* on MSDN.

If you're using SQLite, you must make the same changes in the `/App_Config/connectionstringssqlite.config` file.

Note
The *LDAP* prefix is case sensitive. You should use only capital letters when writing the *LDAP* prefix. For example:

<add name="scofus" connectionString="**LDAP**://root:389/OU=Users,OU=Locations,DC=sitecore,DC=net"/> - this string will work.

<add name="scofus" connectionString="**ldap**://root:389/OU=Users,OU=Locations,DC=sitecore,DC=net"/> - this string WILL NOT work.

Configuring the ASP.NET Security Providers

The Active Directory module is based on the ASP.NET security model architecture.

The module therefore contains a set of basic providers that you can use to manage users, roles, and profile properties:

Service	Description
Membership provider	Provides a set of operations to get, create, update, and delete the users. You can also perform some other operations, like validating the user (username and password) and changing a user's password. For more information about the membership service, visit the MSDN library.
Role provider	Provides a set of operations to get, create, delete roles, add users to, and remove users from roles. For more information about the roles service, visit MSDN library.
Profile provider	Provides a set of operations to get/set the properties of the user profile, as well as different actions for the profile objects (delete/find profiles, and so on.) For more information about the profile service, visit MSDN library.

Configuring the Membership Provider

Open the `web.config` file, search for the `<membership>` element in the `<system.web>` section and paste the following code — the order is not important:

```
<add name="ad"
     type="LightLDAP.SitecoreADMembershipProvider"
     connectionStringName="ManagersConnString"
     applicationName="sitecore"
     minRequiredPasswordLength="1"
     minRequiredNonalphanumericCharacters="0"
     requiresQuestionAndAnswer="false"
     requiresUniqueEmail="false"
     connectionUsername="[put the username here]"
     connectionPassword="[put the password here]"
     connectionProtection="Secure"
     attributeMapUsername="sAMAccountName"
     enableSearchMethods="true"
    />
```

The following table explains every attribute of this provider definition:

Attribute	Description
`name`	The provider name. In general, this can be any string value that is unique within the set of membership providers. However, because some of the configuration is done automatically, this particular element requires the name `ad` in order to leave everything unchanged.
`type`	The full name of the provider class.
`connectionStringName`	The name of the connection string. In our example, it is `ManagersConnString`.
`applicationName`	A standard attribute of any provider that defines

Attribute	Description
	the area of visibility of the provider data. It should be `sitecore` in our example. See the MSDN documentation for details.
`minRequiredPasswordLength`	The minimum number of characters required in a user password. The default value is `1`.
`minRequiredNonalphanumericCharacters`	The minimum number of non-alphanumeric characters required in a user password. The default value is `0`.
`requiresQuestionAndAnswer`	Defines whether the provider requires question and answer to be set for the user passwords. The default value is `false`.
`requiresUniqueEmail`	Defines whether the provider requires each user to have a unique email address. The default value is `false`.
`connectionUsername`	The user name. To connect to the Active Directory domain, you should specify a user who has sufficient rights to perform the necessary operations. The provider uses these credentials when it connects to the AD domain.
`connectionPassword`	The user password.
`connectionProtection`	A system attribute of the provider. Must be set to `Secure` (default value).
`attributeMapUsername`	This attribute defines which Active Directory attribute is to be used as the user name. The user has to explicitly set the "attributeMapUsername" property to "userPrincipalName if you use "userPrincipalName" as the user names. The user has to explicitly set the "attributeMapUsername" property to "sAMAccountName" if you use "sAMAccountName" as the user names.
`enableSearchMethods`	Enables the search functionality of the provider when set to `true` (default value).

Important
If a user is not locked out when they keep entering incorrect credentials, verify that the Account Lockout policy is set correctly in the AD domain you use.

To enable the reset password functionality, you must specify the `enablePasswordReset` attribute and set its value to `true` in the configuration. Furthermore, the provider requires the mapping of several attributes to the Active Directory properties. For more information, take a look at the MSDN article on this topic. You should also verify that the new password meets the requirements of the domain security policy for passwords in your AD domain.

The `connectionProtection` attribute set to `Secure` requires that you add one more element to the `<system.web>` section. You can place it anywhere inside this section:

```
<!-- Machine key attributes -->
<machineKey
validationKey="BDDFE367CD36AAA81E195761BEFB073839549FF7B8E34E42C0DEA4600851B0065856B211719ADEFC76
F3F3A556BC61A5FC8C9F28F958CB1D3BD8EF9518143DB6"
decryptionKey="0DAC68D020B8193DF0FCEE1BAF7A07B4B0D40DCD3E5BA90D" validation="SHA1" />
```

This key is used to encrypt and decrypt the data that is transferred between the client and the Active Directory server. You can either paste this key into your `web.config` file or generate another unique key at:

- http://www.orcsweb.com/articles/aspnetmachinekey.aspx

  or

- http://www.developmentnow.com/articles/machinekey_generator.aspx

If you logged in to the Sitecore CMS shell before performing the `.config` modifications described earlier, you may receive an error such as *Padding is not valid*. This is a consequence of adding a system key and changing the connection protection level. To remove this error, delete all your browser cookies. If this doesn't help, restart the browser.

Configuring the Role Provider

Open the `web.config` file, search for the `<roleManager>` element in the `<system.web>` section and paste the following definition inside it (the order is not important):

```
        <add name="ad" type="LightLDAP.SitecoreADRoleProvider"
connectionStringName="ManagersConnString"
             applicationName="sitecore" username="[put the username here]"
             password="[put the password here]"
             attributeMapUsername="sAMAccountName" cacheSize="2MB" />
```

The following table explains specific attributes of this definition:

Attribute	Description
`cacheSize`	Defines the size of the role cache. The default value is 2 MB.

The other attributes of this element are described in the previous table and also apply to the role provider.

Note
The user has to explicitly set the "attributeMapUsername" property to "userPrincipalName if you use "userPrincipalName" as the user names.

The user has to explicitly set the "attributeMapUsername" property to "sAMAccountName" if you use "sAMAccountName" as the user names.

Configuring the Profile Provider (Optional)

The Active Directory module provides an option to store additional properties in the Active Directory domain. The configuration has extra steps apart from configuring the provider in `web.config`.

For more information, see the section *Advanced Profile Configuration*.

If you do not intend to extend the user profile with custom attributes from the Active Directory, you can skip the section *Advanced Profile Configuration*.

Note
The user has to explicitly set the "attributeMapUsername" property to "userPrincipalName if you use "userPrincipalName" as the user names.

The user has to explicitly set the "attributeMapUsername" property to "sAMAccountName" if you use "sAMAccountName" as the user names.

Activating Switching Providers

To make the system aware of an extra source of users and roles, the switching mechanism must be activated.

To activate the switching mechanism:

- In `web.config` file, in `<system.web>` section, browse for `<membership>` element and find the provider called *sitecore* and set its `realProviderName` attribute to switcher.
- In `web.config` file, in `<system.web>` section, browse for `<roleManager>` element find the provider inside called *sitecore* and set its `realProviderName` attribute to switcher.

The result is the same as modifying the `defaultProvider` attribute for the root element of the service (membership or roleManager). The CMS system requires the providers called *sitecore*. For more information, see the article *Low-Level Sitecore CMS Security and Custom Providers*.

Adding a New Domain

Open the `App_Config/Security/Domains.config.xml` file and add the following line to the root element:

```
<domain name="ad" ensureAnonymousUser="false"/>
```

Adding the Domain-Provider Mappings

Open `web.config` file, and in of `<sitecore>` section, browse to the `<switchingProviders>` element.

It contains three groups: `<membership>`, `<roleManager>` and `<profile>`:

- Add the following line to the `<membership>` group — the order is not important:

  ```
  <provider providerName="ad" storeFullNames="false" wildcard="*"
  domains="ad" />
  ```

- Add the following line to the `<roleManager>` group — the order is not important:

  ```
  <provider providerName="ad" storeFullNames="false" wildcard="*"
  domains="ad" />
  ```

- [Optional] Add the following line to the `<profile>` group. It must be the first definition in this group:

  ```
  <provider providerName="ad" storeFullNames="false" wildcard="*"
  domains="ad" />
  ```

  For more information about this feature, see the section Advanced Profile Configuration.

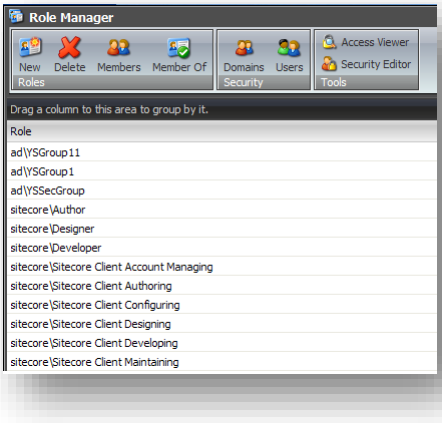
The only difference the order makes is to the sequence in which the users and roles are listed in the Sitecore CMS 6 security tools. For instance, if you put the *ad* membership mapping before the *sql* one, then you'll see the Active Directory users listed in the *User Manager* before the default Sitecore CMS ones.

2.2 Users and Roles

Warning
Any changes made in Sitecore CMS to the active directory users are done in LIVE mode. The changes are applied immediately to the real Active Directory objects. The only exception is user lock-out; in this case the users are locked out locally from Sitecore CMS and remain active in the Active Directory domain.

After you have configured the module, open Sitecore CMS, and log into the Sitecore Desktop as an administrator.

Click **Sitecore**, **Security Tools**, **Role Manager** to open the **Role Manager**. You can see the roles from Active Directory along with the Sitecore CMS roles.



The *ad*\ prefix means that the roles belong to the Sitecore domain called *ad*. Each Sitecore CMS domain can be served via a single provider. For more details on this, see the *Low-Level Sitecore CMS Security and Custom Providers* article.

Start the **User Manager** and you can see a list of the Active Directory users.

Note
You may experience a delay before the Role Manager or the User Manager opens compared to before the Active Directory module was installed. The delay can vary depending on the number of entities in the Active Directory domain. The greater the number of users and roles in the AD, the longer is the delay.

2.2.1 Managing Users and Roles

You can make any changes to the users that are fetched from the Active Directory domain, in the same way as you can change ordinary Sitecore CMS users. You use the standard Role Manager or User Manager for this.

For instance, if you want a user called *ad\john* to log in to the Sitecore CMS desktop, you should add this user to the *sitecore\Sitecore Client Users* role. After you do this, you can log in to the Sitecore desktop as *ad\john*.

Note
You must specify the full username (with the domain name) when you log in.

To allow any user who is a member of an Active Directory role to log in to Sitecore, you should add this role to the *sitecore\Sitecore Client Users* role. In this case, the *Roles-In-Roles* functionality will do the job.

There is a limitation however, to add a user to the Active Directory role, the user must already exist in the same Active Directory domain. As a result, a user from Sitecore CMS cannot be added to the role from Active Directory.

But you can use the Roles-In-Roles functionality to work around this limitation. Add the necessary Sitecore CMS users to the Sitecore CMS role, and then add this role to the Active Directory role. The Roles-In-Roles service works on top of the ASP.NET providers, thereby making this scenario possible.

Creating users/roles

To create a user/role in Active Directory, you should select the appropriate domain (*ad* in our case) in the **New User** or the **New Role** dialog box.

User Specific Operations

The Change Password, Enable, and Lock operations are applied to the Active Directory users in the same way as to they are applied to regular Sitecore CMS users.

2.3 Firewall Modification

The LDAP "Well-known" ports have been established as 389 for LDAP and 636 for LDAP SSL. The Active Directory providers attempt to connect to Active Directory using SSL. If SSL fails, a second attempt to connect to Active Directory using sign-and-seal will be made. If both attempts fail, the providers instance will throw a ProviderException exception.

Active Directory requires either port 389 or 636 to be opened.

Microsoft SMB/CIFS port 445 is required to be opened for Active Directory Profile Provider.

Chapter 3

Advanced Profile Configuration

The Active Directory module allows you to store the custom properties of a user profile in the attributes of the corresponding domain user object. For example, in Active Directory you have a telephone number for each user and you want this information to be visible for each user in Sitecore CMS.

This chapter contains the following sections:

- Configuring the Profile Provider
- Configuring the Custom Properties
- Extending the Sitecore Template

3.1 Configuring the Profile Provider

To change the profile provider, you must add a profile provider definition.

To change the profile provider:

1. Open the `web.config` file, in the `<system.web>` section, locate the `<profile>` element.
2. Paste the following provider definition inside the `profile/providers` element (the order is not important):

   ```
   <add name="ad" type="LightLDAP.SitecoreADProfileProvider"
   connectionStringName="ManagersConnString" applicationName="sitecore" username="[put the
   username here]" password="[put the password here]" sitecoreMapDomainName="[domain name]"
   />
   ```

   The following table explains attributes of this definition:

Attribute	Description
`sitecoreMapDomainName`	Specifies the Sitecore security domain handled by the current profile provider. By default, it is equal to the provider name.

3. Change the `defaultProvider` attribute of the `<profile>` element to `switcher`.
4. During the general configuration process, you must have created a mapping for this profile provider just as you would for any other service.
5. In your main `web.config` file, locate the `switchingProviders/profile` element and insert the following line:

   ```
   <provider providerName="ad" storeFullNames="false" wildcard="*"
   domains="ad" />
   ```

Technical Note
This element must be listed first in this section. The SQL server provider, which is the default for the profile service, has been designed to be very flexible. As such, if it can't find the appropriate profile property, it considers this property to be a custom property and stores it in the *Custom* table.

As a result, if the SQL Server is listed first in this section, it will always handle all the properties. The definition of the AD provider must therefore be listed first in this section to allow the AD provider to handle its properties and supply the information from the AD domain.

3.2 Configuring the Custom Properties

Any custom property, which you wish to extend the profile with, should be defined in the `<properties>` group of the `<profile>` element. In our example, we should place the `telephoneNumber` property there. So, browse to the `<profile>` element in the `<system.web>` section of `web.config` file, locate the `<properties>` group inside this element, and add the following line (the order is not important):

```
<add type="System.String" name="Telephone" customProviderData="ad|unicode string|telephoneNumber"
/>
```

The following table explains each attribute of this definition:

Attribute	Description
`Name`	A unique name of the property.
`Type`	The .NET type of the property. This is the type the property will have in the ASP.NET environment. This parameter must always be `System.String` for `ad` profile properties. Sitecore works with all properties in the same way as the string type.
`customProviderData`	Any data required for the provider serving this property. In our example, `ad` means that this property is handled by the *ad* provider. The `unicode string` defines the native type of the property in Active Directory and `telephoneNumber` specifies the name of the corresponding Active Directory attribute. In general, this attribute can be of any form. The *ad* provider expects a pipe separator `|` to be placed between the attribute parts.

The Active Directory module expects the `customProviderData` to use the following format: `ad|[native type name]|[ad native name]`.

The `native type name` is the type of the property in Active Directory.

The following basic types are supported:

- `unicode string`
- `boolean`
- `integer`
- `large integer`

If anything else is specified, the `unicode string` type is used as a fallback. The last optional `ad native name` part is the real name of the corresponding property in Active Directory. If it is omitted, the value falls back to the `name` attribute.

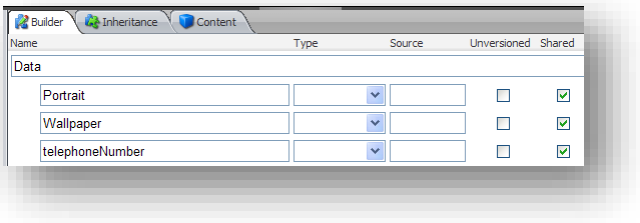
Note
The Active Directory module does not support reflecting the custom Active Directory attributes to the Sitecore CMS roles. Sitecore CMS security is based on the .NET security model, and this underlying engine does not have the facility to have custom attributes as profiles for the CMS roles. Such reflection is not supported in the Active Directory module, as well as in the Sitecore CMS.

3.3 Extending the Sitecore Template

After you have defined the custom properties in the `web.config` file, you should extend the Sitecore template to make those properties accessible from the Sitecore CMS security applications.

To extend the Sitecore template:

1. Start Sitecore CMS and log in with the administrator credentials and switch to the `core` database.
2. Open the **Content Editor**, and in the content tree navigate to the `/sitecore/templates/System/Security/User` template.
3. Add a new field to this template called `telephoneNumber` (exactly the same as the appropriate property name in `web.config`).



4. Select the required data type, for example the `Single-Line Text` type, and mark it as `Shared`.

   Note
   If the new custom property is mapped to a multi-valued attribute, the field must use the *Multi-Line Text* data type. Otherwise, the value in the field may be saved incorrectly as a multi-valued attribute of the AD user.

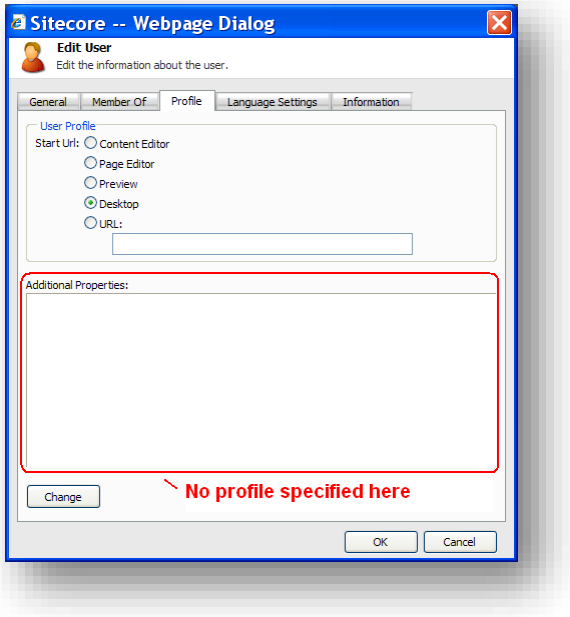
   We cannot guarantee that multiple values are stored or displayed in the correct order because of the way that Microsoft has implemented LDAP in Active Directory. For more information, see the description of the isSingleValued property on the following page:

   http://msdn.microsoft.com/en-us/library/windows/desktop/ms675578(v=vs.85).aspx

   Note that this description says: "Multi-valued attributes are unordered; there is no guarantee they will be stored or returned in any specific order."

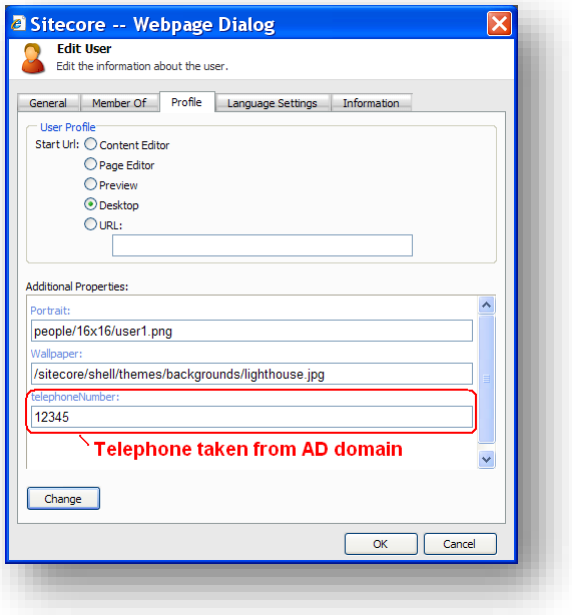
5. Save your changes and switch to the `master` database.
6. Open the *User Manager* and edit any user from the Active Directory domain, for instance, `ad\john`.

7. Click the **Profile** tab



8. Click **Change** and select a user template.
9. Close this window.

If you edit *ad\john* user again and switch to the **Profile** tab, you'll see that the profile is extended with the *telephoneNumber* property:



The property is filled in with the `telephoneNumber` attribute value from the corresponding Active Directory user.

For more information about this feature, see the *Low-Level Sitecore CMS Security and Custom Providers* article. This article also contains information about the conditions that the provider should satisfy to support the profile extension.

Assigning the non-default profile to AD users automatically

The previous description of how to extend a Sitecore profile template includes a step about assigning the profile item to the user. It is obvious that such a manual operation is not a way to go for hundreds of users. For this purpose, Sitecore allows to set a common profile item on the domain level.

Open the `/App_Config/Security/Domains.config.xml` file and add the following line to the domains element:

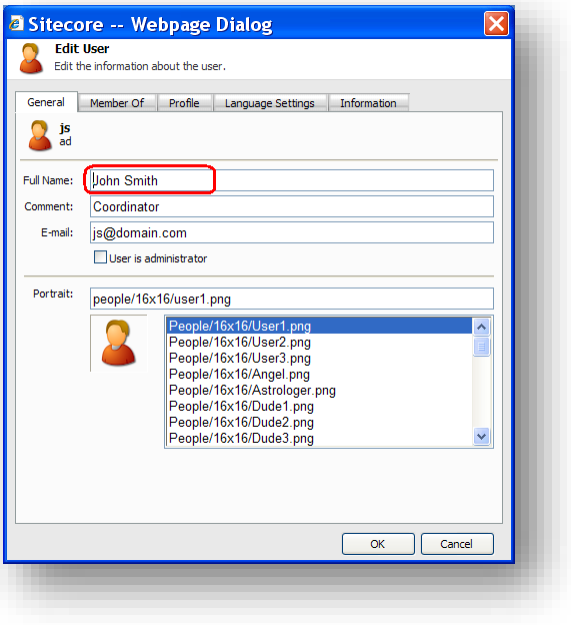
```
<domain name="ad" ensureAnonymousUser="false" defaultProfileItemID="{DDEDA46F-169B-4A70-8732-DBD3F407AF2E}"/>
```

The `defaultProfileItemID` attribute defines the profile item that is used for users from the domain if the profile is not set for the user explicitly.

3.3.1 Mapping the AD *Display Name* Attribute to the *Full Name* Property in Sitecore

The value from the *DisplayName* property of the AD user is automatically passed to the default **Full Name** field.

You do not have to extend the user's profile template with a new field for this property — the value is displayed in the **Edit User** dialog box, on the **General** tab, in the **Full Name** field:



If you want to map another AD property to the **Full Name** field, you can do it in the `App_config/Include/ldap.config` file. Set the appropriate value for the `LDAP.FullName` setting:

```
<!-- FULL NAME PROPERTY NAME
     Determines the full name property mapping.
-->
<setting name="LDAP.FullName" value="ad|unicode string|displayName" />
```

Note

Remember that the *Advanced* profile feature must be configured as described earlier to support this mapping.

Chapter 4

Other Features

The Active Directory module provides some extra features to make security resolution easier.

This chapter contains the following sections:

- Custom Filter
- Create new AD entity pipelines
- The Status Page
- Debug Mode
- Connecting to Multiple Domains
- Cache Settings
- Directory Notification
- Nested Groups (Indirect Membership)
- User Permissions
- File List
- Minimum properties of AD objects
- FAQs
- Developers Notes

4.1 Custom Filter

You are allowed to provide a custom filter that will be applied to all requests to the AD server. It can be set in the provider level.

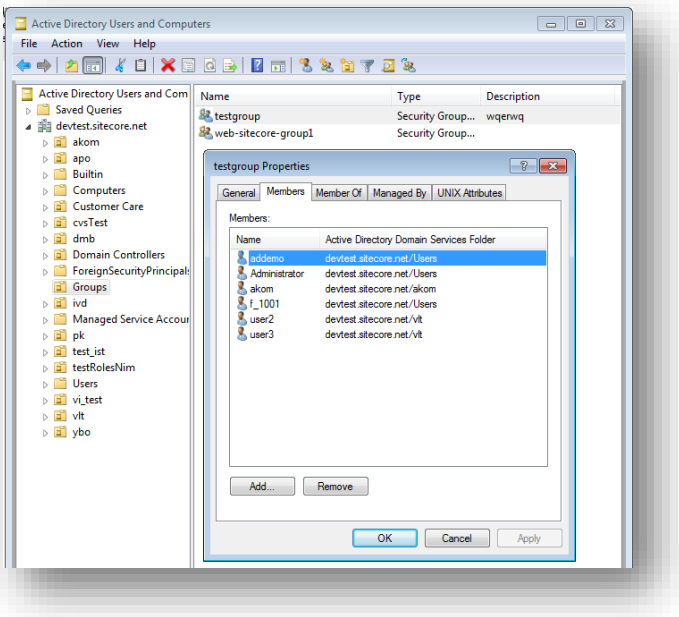
```
        <add name="ad"
        type="LightLDAP.SitecoreADMembershipProvider"
        connectionStringName="ManagersConnString"
        applicationName="sitecore"
        minRequiredPasswordLength="1"
        minRequiredNonalphanumericCharacters="0"
        requiresQuestionAndAnswer="false"
        requiresUniqueEmail="false"
        connectionUsername="user"
        connectionPassword="12345"
        attributeMapUsername="sAMAccountName"
        enableSearchMethods="true"
        customFilter="(memberOf=cn=test role 1,OU=CRM,DC=VM)"
        />
```

The filter syntax is described in the Creating a Query Filter article on MSDN.

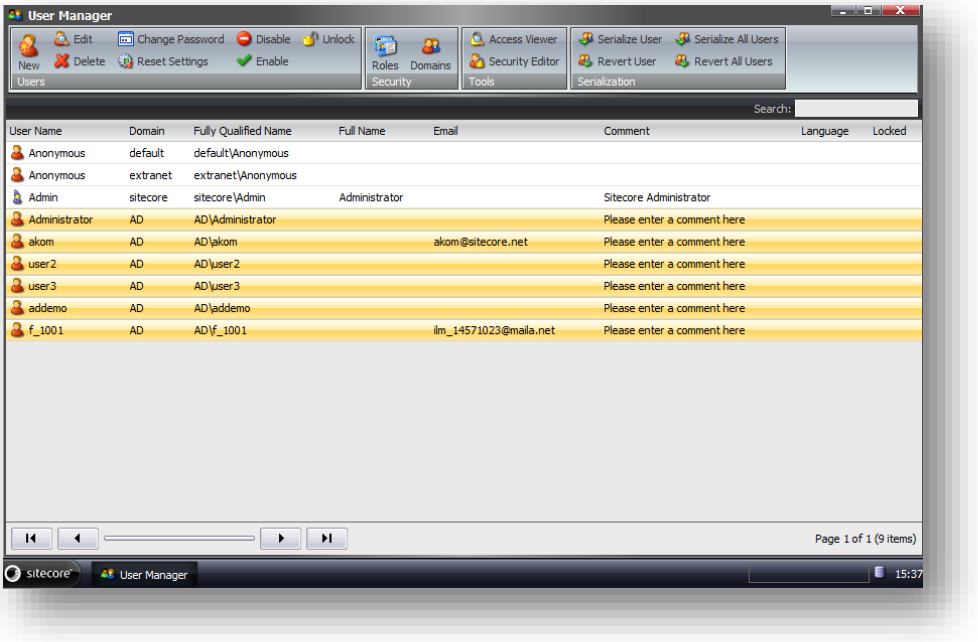
Using the custom filter, you can plug in all users and roles from several AD containers (OU) to a single domain. The users and roles must have a common attribute for it or be members of a common role. The following sample receives all the users and roles from the `sitecore` domain controller that are members of the `testgroup`. Test role is placed under the **Groups** organizational unit. The `customFilter` attribute is valid for membership, role and profile provider.

```
        customFilter="(memberOf=CN=testgroup,OU=Groups,DC=devtest,DC=sitecore,DC=net)"
```

In the screenshot below you can see the users of the *testgroup* role in Active Directory



In this screenshot you can see the users of the *testgroup* role in the Sitecore User Manager after applying the custom filter



Samples

The following samples show how to use the custom filter by placing the code in the config file.

- Select all objects from AD who aren't the part of the *testgroup* role. The *testgroup* role is placed under the Groups container:

```
(!(memberOf=CN=TestGroup,OU=Groups,DC=devtest,DC=sitecore,DC=net))
```

- Select all objects from AD whose name starts from "test":

```
(name=test*)
```

- Select all objects from AD whose attribute displayName starts from "One":

```
(displayName=One*)
```

Warning
The membership, roleManager and profile providers that use the common connection string must also use the equal custom filter.

Note
The custom filter allows filtering such objects as users and roles within a certain range of OU. This range is strictly determined by the principles of work of Active Directory (starting from the OU which you have been already connected to, and down along the tree).

4.2 Create new AD entity pipelines

The Custom filter affects both new and existing users and roles. If you use a custom filter and create new users and roles in Sitecore that do not match the parameters of this custom filter, they will not be visible in Sitecore.

You can fix this issue by using two pipelines in the `ldap.config` file: `initializeAdUserEntry`, `initializeAdRoleEntry`.

```
      <initializeAdUserEntry>
        <!--
        Use the processor if all new user should have a predefiled value in a property.
        The PropertyName parameter defines the name of the property.
        The DefaultValue parameter defines the default value of the property.
        -->
        <!--
        <processor type="LightLDAP.Pipelines.InitializeAdEntry.SetPropertyValue, LightLDAP">
          <PropertyName desc="AD property name ">type the property name here</PropertyName>
          <DefaultValue desc="AD property value ">type the default property value
here</DefaultValue>
        </processor>
        -->
        <!--
          Use the processor if all new roles should be a member of the predefined role.
          The RoleName parameter defines the name of the main role.
        -->
        <!--
        <processor type="LightLDAP.Pipelines.InitializeAdEntry.AddToRole, LightLDAP">
          <RoleName desc="AD group">type role name here</RoleName>
        </processor>
        -->
        <processor type="LightLDAP.Pipelines.InitializeAdEntry.CommitChanges, LightLDAP"/>
      </initializeAdUserEntry>
      <initializeAdRoleEntry>
        <!--
        Use the processor if all new user should have a predefiled value in a property.
        The PropertyName parameter defines the name of the property.
        The DefaultValue parameter defines the default value of the property.
        -->
        <!--
        <processor type="LightLDAP.Pipelines.InitializeAdEntry.SetPropertyValue, LightLDAP">
          <PropertyName desc="AD property name ">type the property name here</PropertyName>
          <DefaultValue desc="AD property name ">type the default property value
here</DefaultValue>
        </processor>
        -->
        <!--
          Use the processor if all new roles should be a member of the predefined role.
          The RoleName parameter defines the name of the main role.
        -->
        <!--
        <processor type="LightLDAP.Pipelines.InitializeAdEntry.AddToRole, LightLDAP">
          <RoleName desc="AD group">type role name here</RoleName>
        </processor>
        -->
        <processor type="LightLDAP.Pipelines.InitializeAdEntry.CommitChanges, LightLDAP"/>
      </initializeAdRoleEntry>
```

The `LightLDAP.Pipelines.InitializeAdEntry.SetPropertyValue` allows you to initialize the property of a user or role with a predefined value.

The `LightLDAP.Pipelines.InitializeAdEntry.AddToRole` allows you to add a user or role to the main role.

4.3 Single Sign-on

In most cases, if an organization has the domain controller set up, the workstations are usually included into this domain. Imagine that you have established a connection between the *Managers* organization unit and your Sitecore CMS installation. This means that the members of this organization unit are now able to work in Sitecore CMS according to their roles. Naturally, these users wish to be logged in to Sitecore CMS automatically. When users start Sitecore CMS, they are definitely logged in to their organization domain.

This feature is called Single Sign-on and the Active Directory module supports this it.

4.3.1 Prerequisites

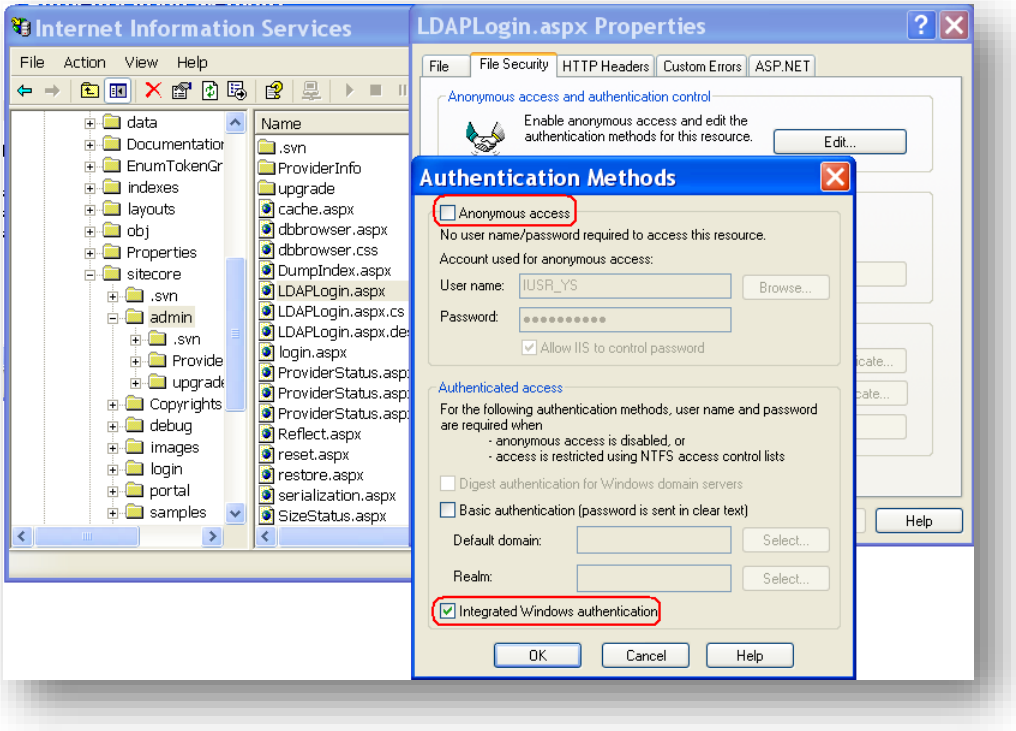
There are some prerequisites for using this functionality:

- The workstation must be a member of the appropriate domain.
- The anonymous access must be disabled to the `/sitecore/admin/ldaplogin.aspx` page and the Integrated Windows security mode must be turned on.

To disable anonymous access in IIS:

1. Open IIS.
2. Expand the target website.
3. Navigate to the `/sitecore/admin` folder and select the `LDAPLogin.aspx` page.
4. Right-click the `LDAPLogin.aspx` page and then click **Properties**.
5. In the **LDAPLogin.aspx Properties** dialog box, click the **File Security** tab and clear the **Anonymous access** checkbox.

The IIS configuration should look similar to this:



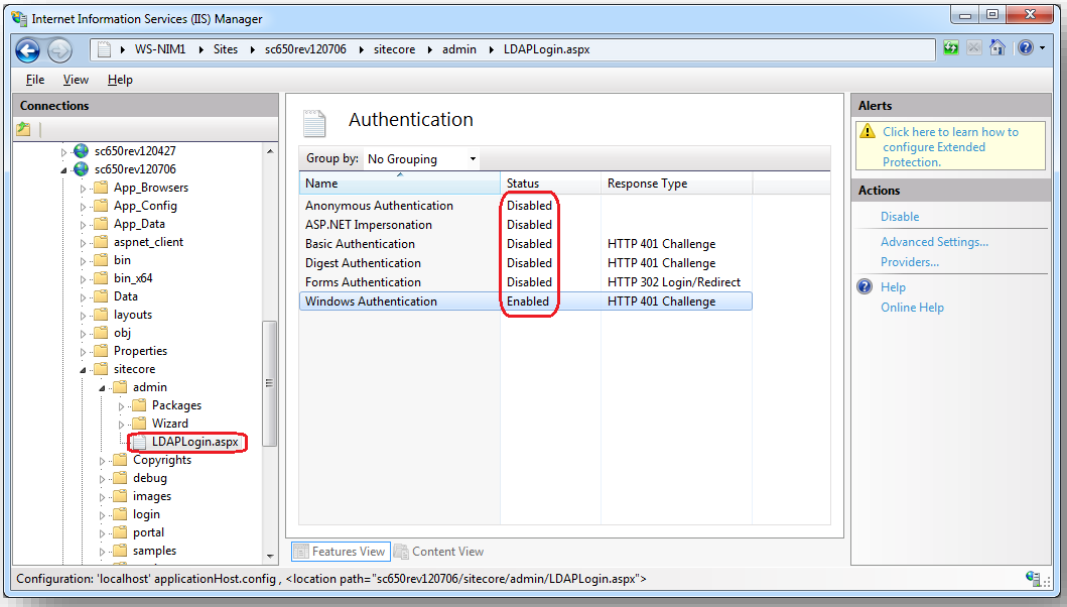
IIS 7 Configuration

Note
IIS 7 does not support mixed authentication mode. Hence you cannot have several authentication types enabled on the */sitecore/admin/ldaplogin.aspx* page. To use **Windows Authentication**, you must disable all other authentications and enable **Windows Authentication** for this page.

To configure IIS 7:

1. Open IIS.
2. Expand the target website.
3. Navigate to the */sitecore/admin* folder and in the context menu select **Switch to Content View**.
4. Select the *LDAPLogin.aspx* page and click on **Switch to Features View**.
5. In the right-hand pane click on the **Authentication**.
6. Disable all the authentications and enable the **Windows Authentication** one.

The configuration should look similar to the following screenshot:

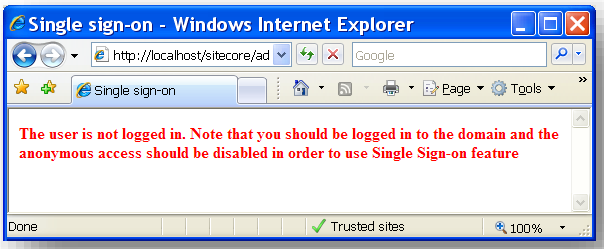


4.3.2 Feature Usage

When the prerequisites are satisfied, you can log in to Sitecore CMS with your system account without manually providing your user credentials. Enter the following URL in your browser:
`http://[yoursite]/sitecore/admin/LDAPLogin.aspx`

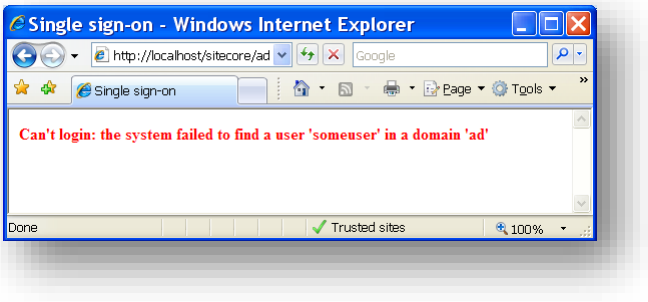
Note
You can still login in the usual way by opening the default Sitecore shell login page (`http://[yoursite]/sitecore`)

If you forget to verify the prerequisites and your machine appears not to be in a domain or anonymous access has not been removed from the login page, the system will not let you log in, displaying the reason of the refusal:

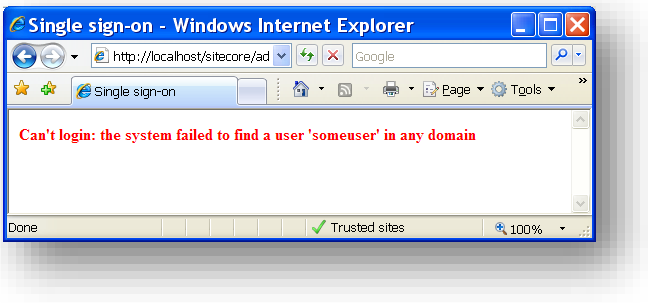


Some errors may occur when the system begins to analyze the user credentials. For instance, if the domain name is correct, and you are a member of the Active Directory domain, but you're not a member

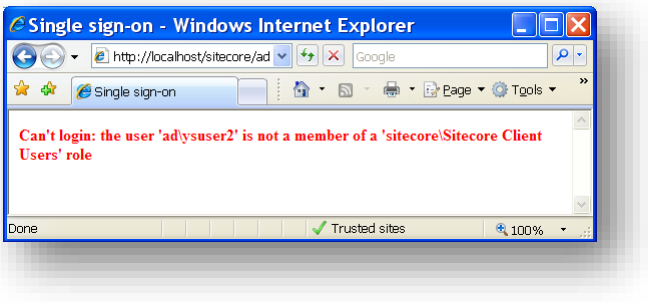
of the Managers organization unit which is plugged into Sitecore, you'll receive the following warning:



The real domain name might differ from the domain name entered in Sitecore CMS. For instance, you may be a user of the Active Directory domain called `Company.com`, but this domain is plugged into Sitecore CMS as "ad" (which is done by default). In this case, the system won't reject your attempt to log in, but it will iterate the existing Sitecore CMS domains in an attempt to find the appropriate user. If the user cannot be found, the following warning is displayed:



When the user is found in the Active Directory domain, but doesn't have enough permissions to log in to Sitecore CMS (the user is not included in the *sitecore\Sitecore Client Users* role), the system will reject the log in attempt and display the following message:



Finally, if everything is fine and the user is allowed to log in, you'll be logged in automatically and redirected to the Sitecore CMS desktop.

The page redirects you to a user's StartURL. Please check the following method:

```
        //LightLDAP.LDAPLogin
        private string GetStartUrl(User user)
        {
        string text = WebUtil.GetCookieValue("sitecore_starturl");
        if (user != null)
        {
        text = StringUtil.GetString(new string[]
```

```
        {
        user.Profile.StartUrl,
        text
        });
        }
        return StringUtil.GetString(new string[]
        {
        text,
        "/sitecore/shell/applications/clientusesoswindows.aspx"
        });
        }
```

By default, a user's StartUrl file (user.Profile.StartUrl) is empty (StartURL.png). Therefore, the clientusesoswindows.aspx file is used.

The page redirects you to the Content Editor if your user has access to it:

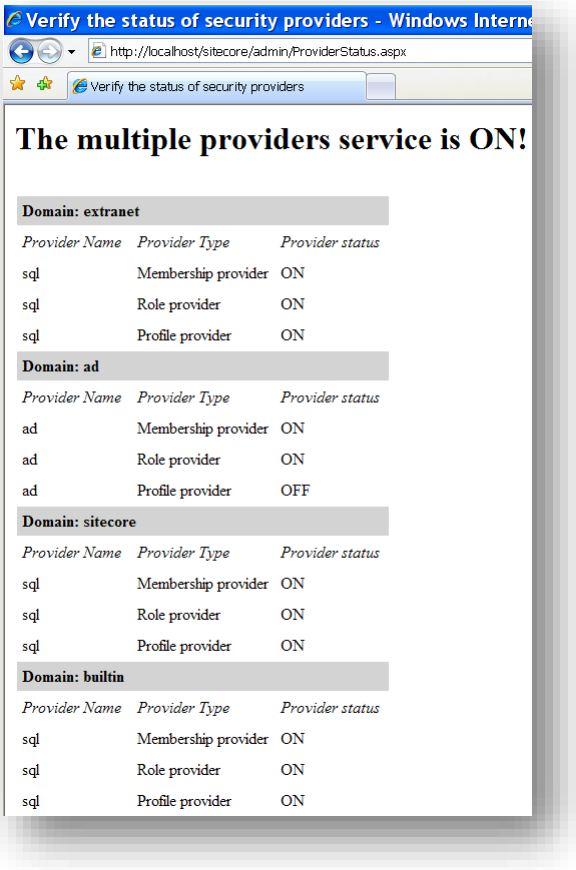
```
        <%@ Page language="c#" AutoEventWireup="false" %>
        <%@ Import namespace="Sitecore.Data.Items"%>
        <%
        Sitecore.Configuration.State.Client.UsesBrowserWindows = true;
        Sitecore.Configuration.State.Client.NoDesktop = true;

        Item item = Sitecore.Context.Database.Items["/sitecore/content/Applications/Content
Editor"];

        if (item != null && item.Access.CanRead()) {
        Response.Redirect("/sitecore/shell/Applications/Content editor.aspx");
        }

        Response.Redirect("/sitecore/login/default.aspx?sc_error=You do not have access to the
Content Editor.");
        %>
```

Note
If you are in debug mode, you'll first see the entire list of roles that you are a member of in Sitecore CMS. The **Login** button will be enabled allowing you to login then. You can read more about debug mode later in this article.

4.4 The Status Page

As an additional option for troubleshooting potential security problems, the module has a special status page. To open this status page, use this URL:
`http://[yoursite]/sitecore/admin/ProviderStatus.aspx`

The status page should look something like this:



This page contains basic information about the domains and security providers. You can see each domain listed here and the providers which handle these domains. It also contains some high level status information about each provider:

- If the status is *ON*, the provider is functioning and can serve the requests.
- If the status is *OFF*, the provider has refused a simple request and the system has marked it as broken.

Depending on the problem, it may not be possible to display the *Status* page. Sometimes the entire application will fail to start if there is an error in the provider configuration. But this is the first step that you should take when troubleshooting the Active Directory module configuration or the security provider's definition. When a provider is flagged as being *Off*, it is not functioning and its configuration should be adjusted. The next troubleshooting step is to examine the log file.

Developer's Note
The status page performs a simple call to any method of the appropriate provider, which doesn't require valid input parameters, to make a decision if the provider works. For instance, the `GetAllUsers()` method is called for the membership provider. If the method throws an exception, the status is automatically set to OFF.

4.5 Debug Mode

The Active Directory module contains a debug mode — in which more detailed information is written to the log file and is displayed on a login page. This method is also called verbose logging in some systems. Using debug mode, the administrator can examine the detailed log files to locate a potential problem. It is also quite helpful for the technical support — the detailed log files contain enough information to identify the type and place of the error.

To enable debug mode:

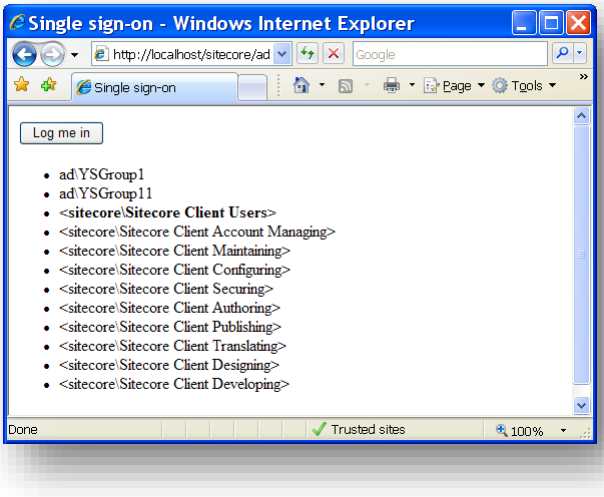
- Open the `/App_Config/Include/ldap.config` file, locate the `LDAP.Debug` setting and set it to `true`.

  This setting is set to `false` by default. When the changes to the pluggable `.config` file are applied immediately, the next request to the module will dump much more information to the Sitecore CMS log file.

The `LDAP.Debug` setting applies to the entire module.

However, there is also an option to enable this mode for the Single Sign-on feature only.

To enable this mode for the Single Sign-on feature only, append the following query string parameter to the login page URL: `?debug=true`. When you try to log in in debug mode, you are presented with membership information about your account. This information includes the direct and indirect roles the user is a member of. You can login by clicking **Log me in**:



4.6 Connecting to Multiple Domains

The Active Directory module and the Sitecore security model allow you connect to as many AD domains as you wish. You may want to the managers in your company headquarters and the developers in your regional office to access one Sitecore CMS installation.

To connect to multiple domains this, you must configure one or more sets of providers in the `web.config` file:

- Add a connection string to the new AD domain.
- Add a membership provider definition.
- Add a role provider definition.
- Add a profile provider definition (if you wish to share the user profile).
- Add a new Sitecore CMS domain and point it to the new set of providers.

For more information about configuring extra security providers, see the article *Low-Level Sitecore CMS Security and Custom Providers*.

4.7 Cache Settings

The module caches some information to improve the performance. The cache settings are defined in the `App_Config/include/ldap.config` file:

- `LDAP.Caching.UserCache` — determines the cache size for the user information.
- `LDAP.Caching.MemberOfCache` — determines the cache size for the *memberOf* information.
- `LDAP.Caching.MembersCache` — determines the cache size for the *members* information.

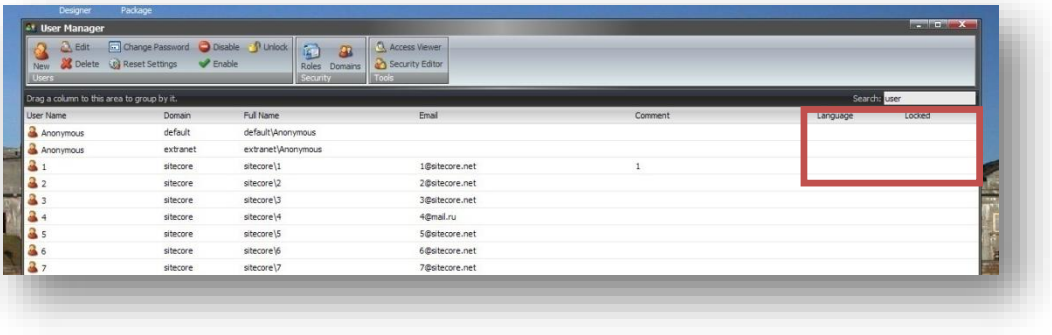
4.7.1 Additional LDAP.config Settings

The `/App_Config/Include/ldap.config` file contains some additional settings:

- `LDAP.EnableSorting` — whether or not sorting is enabled. The default value is false.

  Sorting doesn't work on Windows Server 2008.

- `LDAP.SortKey` — the name of the AD attribute that is used for sorting AD users. This setting influences the performance of AD. The more different values for this field that are in AD the more time it takes the AD server to sort requested users. The default value is the `codePage`.
- `LDAP.SizeLimit` — the maximum number of users that AD returns for the "*" search. The User Manager only shows the number of users specified in `LDAP.SizeLimit` for each AD domain. The setting has been added to improve the performance. The default value is 1000. Narrowing down the search criteria will help find a particular user.
- `LDAP.FindSizeLimit` — the maximum number of users that AD returns for other searches. The User Manager only shows the number of users specified in `LDAP.FindSizeLimit` for each AD domain. The setting has been added to improve performance. The default value is 100.



- `LDAP.SettingsPropertyValueFactory` — by default the AD module allows you to work with boolean, unicode string, and integer types of AD properties from the user profile. You can implement a class that allows you to use more AD property types.
- `LDAP.ReconnectPeriod` — the period when the module restores a broken notification connection. To keep the actual cache data the module is notified by the AD server about any changes.
- `LDAP.NotificationTimeOut` — the timeout of notification connections.
- `LDAP.DeleteScope` — whether or not the module is allowed to delete AD objects from the whole AD tree or just under in the connection string Organization Unit (one level).

4.8 Directory Notification

The module uses the DirectoryNotificationControl (`http://msdn.microsoft.com/en-us/library/system.directoryservices.protocols.directorynotificationcontrol.aspx`) to be notified when changes are made to an object in the Active Directory Domain Services.

If all the changes in the AD are made by the Sitecore application, you can disable this feature.

Set the `useNotification="false"` attribute for all the AD Role, Profile and Membership providers listed in the `web.config` file:

```
                <add name="ad"
                     type="LightLDAP.SitecoreADMembershipProvider"
                     connectionStringName="ManagersConnString"
                     applicationName="sitecore"
                     minRequiredPasswordLength="1"
                     minRequiredNonalphanumericCharacters="0"
                     requiresQuestionAndAnswer="false"
                     requiresUniqueEmail="false"
                     connectionUsername="[put the username here]"
                     connectionPassword="[put the password here]"
                     connectionProtection="Secure"
                     attributeMapUsername="sAMAccountName"
                     enableSearchMethods="true"
                    useNotification=”false”
                    />


<add name="ad" type="LightLDAP.SitecoreADRoleProvider" connectionStringName="ManagersConnString"
     applicationName="sitecore" username="[put the username here]"
     password="[put the password here]" useNotification=”false”/>

        <add name="ad" type="LightLDAP.SitecoreADProfileProvider"
connectionStringName="ManagersConnString" applicationName="sitecore" username="[put the username
here]" password="[put the password here]" sitecoreMapDomainName="[domain name]"
useNotification=”false” />
```

4.9 Nested Groups (Indirect Membership)

The Sitecore Active Directory module supports indirect membership.

For example, if you have a user called *John*, who is a member of the *Professional Service* group, which in turn is a member of a parent group called *Solution Department*.

To allow *John* to log in to Sitecore, you must enable the indirect membership feature and add the *Solution Department* group to the *sitecore\Sitecore Client Users* default role.

To enable indirect membership:

1. Open the `/App_Config/Include/ldap.config` file.
2. Set the value of the `LDAP.IncludeIndirectMembership` to `True`.

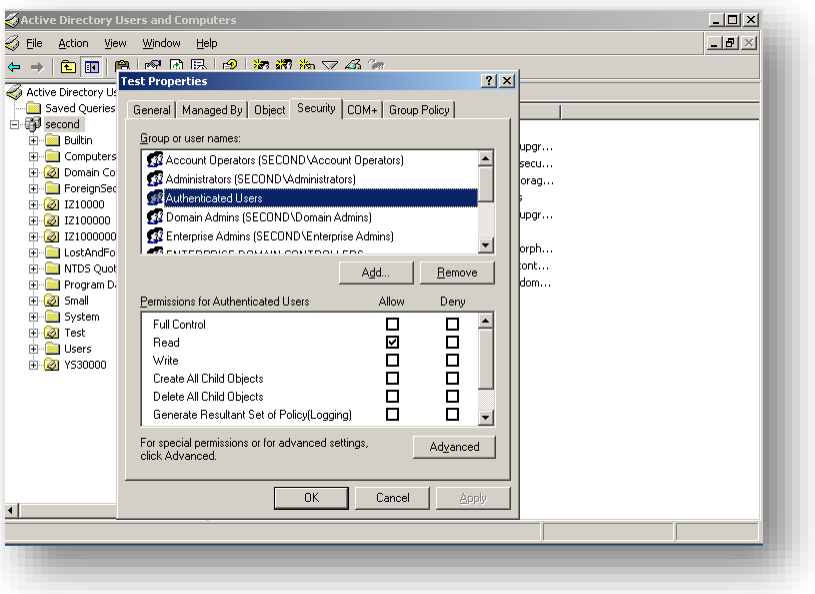
Note
It is appropriate to use the indirect membership feature to allow users to log in to Sitecore. However, if you enable indirect membership, the Sitecore security tools will not display a complete list of the roles that a user is a member of or a complete list of the users that are members of a role. The Sitecore security tools will only display a list of the roles that a user is a direct member of and the users that are direct members of a role.

The `IsUserInRole` method is used to verify role membership. It returns *True* if the user is a member of the role in question and if indirect membership is turned on, it also takes the parent roles into consideration.

4.10 User Permissions

The module uses credentials to communicate with the AD server. It does not always require *Write* access.

To set the necessary permissions, open the **Properties** dialog box for the organization unit in question.



You can use one of the following access levels to improve your configuration:

- Read-only.
- Limited Read-write.
- Change Password.
- Full Read-write.

4.10.1 Read-only

This access level allows you to:

- Read users/roles.
- Reading profile properties.

Minimum required permissions:

- List contents.
- Read all properties.

4.10.2 Limited Read-write

This access level includes read-only operations and allows you to:

- Modify profile properties.
- Perform password operations.

- Modifying email.
- Add an AD user to an AD role.

Minimum required permissions:

- List contents.
- Read all properties.
- Write all properties.

4.10.3 Change Password

This access level requires these additional permissions:

- Read all properties.
- Change password.
- Reset password.

You must also add some permission to an AD container (the *Authenticated Users* role):

- List contents.
- Read all properties.

Important

If any of those permissions is not set, you will see the following exception:
Exception Details: System.DirectoryServices.DirectoryServicesCOMException: There is no such object on the server.

4.10.4 Full Read-write

This access level includes read-write access and allows you to:

- Change passwords.
- Create users/roles.
- Delete users/roles.

The minimum required permissions are:

- List contents.
- Read all properties.
- Write all property.
- Create group objects.
- Delete group objects.
- Create user objects.
- Delete user objects.
- All extended rights.

4.11 File List

The module package only contains files. No items or security entities are included.

The entire list is:

- `/bin/LightLDAP.dll` (the main assembly of the module).
- `/bin/LightLDAPClient.dll` (the client assembly of the module).
- `/App_Config/Include/ldap.config` (the pluggable configuration file of the module).
- `/sitecore/admin/LDAPLogin.aspx` (the login page for the Single Sign-On feature).
- `/sitecore/admin/ProviderStatus.aspx` (the statistics page of the provider status).

4.12 Minimum properties of AD objects

The AD user must have the following obligatory properties:

- securityIdentifier
- userPrincipalName
- sAMAccountName
- comment
- whenCreated
- mail
- pwdLastSet
- UserAccountControl,
- msDSUserAccountControlComputed
- cn,
- DN,
- objectCategory,
- objectClass
- isdeleted,
- lastknownparent,
- lockoutTime,
- primaryGroupID,
- pwdLastSet,
- tokenGroups,
- usnchanged,
- usncreated

The AD group must have the following obligatory properties:

- sAMAccountName
- cn,
- primaryGroupToken,
- whenCreated,
- usncreated,
- usnchanged

4.13 FAQs

This section contains the frequently asked questions about the AD module.

Q

I'm trying to create a user in the Active Directory domain from Sitecore CMS, but the CMS says that it can't create the user. I'm filling in all the required fields, but still no luck. What is the reason?

A

One possible reason is that the user already exists. If the User Manager does not contain this user, you are probably only plugging a single organization unit into Sitecore CMS and not the entire Active Directory domain. In this case, the user probably exists in another organization unit.

When you create a user, the module verifies the uniqueness of the user name in the entire domain.

Q

When I create a user in the Active Directory domain from Sitecore CMS, I don't specify a value for the **Comment** field. But when the user is created, I can see the *please enter a comment here* message in the field. Why is it filled in automatically? Can I prevent this?

A

You may also occur when you update the user and leave the **Comment** field blank. The Sitecore CMS Active Directory provider is based on the default Microsoft Active Directory provider, which requires the comment not to be blank when updating the user information. That is why it is populated with the default value if you leave it blank. You can enter a comment at any time.

Q

I have created a user and I can see that their title and department values are set to *undefined*. Why?

A

That is because you have set the `requiresQuestionAndAnswer` attribute of the provider definition element to `True`. You should also have placed the additional attributes for mapping the password question and answer to the AD attributes there. For example, if you have mapped them to the `title` and `department` attributes. When you create the user and the configuration requires a password question and answer, these fields can't be left empty or null. They are therefore set to *undefined* in this case. To avoid this behavior, change the attribute `requiresQuestionAndAnswer` to `False`.

Q

I have renamed a user and it lost its security assignments immediately. But when I rename it back, it obtains its old assignments back. Where the trick is?

A

This behavior reflects the way .NET security treats the username. The fully-qualified username plays the role of a unique identifier. Due to the fact that this name is used to store security settings for items in Sitecore CMS, renaming a user is similar to creating a new user. The general recommendation is not to rename users which have got any security assignments.

Q

How can I access the profile properties which come from the Active Directory?

A

As other profile properties, these properties can be addressed in the following way:
`user.Profile["property_name"] = "property_value";`
This line assumes that the user is an object of class `Sitecore.Security.Accounts.User`.

Q

I would like to reflect some group attributes of Active Directory to the role in Sitecore. Is this possible?

A

The Active Directory module does not support this scenario. Neither does in Sitecore CMS, because Sitecore security is based on the .NET security model, and this underlying engine doesn't provide an option to have profiles for the roles.

4.14 Developers Notes

4.14.1 The timeout alert can occur in an AD containing 1,000,000 users

You can receive the timeout alert when you search for a user in an Active Directory that contains more than 1,000,000 users.

Workaround:

Add the `serverPageTimeLimit` attribute to the membership definition:

```
<add name="ad"
     type="LightLDAP.SitecoreADMembershipProvider"
     connectionStringName="ManagersConnString"
     applicationName="sitecore"
     minRequiredPasswordLength="1"
     minRequiredNonalphanumericCharacters="0"
     requiresQuestionAndAnswer="false"
     requiresUniqueEmail="false"
     connectionUsername="[put the username here]"
     connectionPassword="[put the password here]"
     connectionProtection="Secure"
     attributeMapUsername="sAMAccountName"
     enableSearchMethods="true"
    serverPageTimeLimit =”5”
    />
```

This attribute defines the time limit to use when the server searches for an individual page of results.

The default value is -1 second and this means that Active Directory will search for the user until all the users have been analyzed.

4.14.2 Sorting in Windows 2008

If the `LDAP.EnableSorting` attribute is set to `True`, the User Manager shows the following error:

The server does not support the requested critical extension.

Sorting must be disabled on Windows 2008.