Active Roles 8.2.1

Feature Guide

Legend

WARNING: A WARNING icon highlights a potential risk of bodily injury or property damage, for which industry-standard safety precautions are advised. This icon is often associated with electrical hazards related to hardware.

CAUTION: A CAUTION icon indicates potential damage to hardware or loss of data if instructions are not followed.

Active Roles Feature Guide
Updated - 28 November 2024, 10:29

For the most recent documents and product information, see Online product documentation.

Contents

1

Introduction

This document provides a detailed description of Active Roles and its major features available in Active Roles 8.2.1.

The document describes each feature in a separate section containing the following information:

- **Feature name**: The name of the feature, indicated in the section title.
- **Description**: A detailed description of the feature.
- **Getting started**: Information on how to start using the feature. In most cases, this includes references to other Active Roles documents, such as the Administration Guide or the available User Guides, depending on the target users of the feature.

  NOTE: Consider the following regarding the **Getting started** information:

  - Unless indicated otherwise, using the described Active Roles features requires an Active Roles Admin account. By default, an Active Roles Admin is any member of the Administrators local group on the computer running the Active Roles Administration Service.
  - When attempting to use features of the Active Roles Console, make sure that the Console interface is set to "Advanced view mode". To do so, in the **View** menu, click **Mode** > **Advanced Mode**.

About Active Roles

Active Roles delivers a reliable, policy-based administration and provisioning solution, allowing enterprises to fully benefit from Active Directory and Microsoft Exchange deployment.

One of the most valuable features of the product is the ability to automate provisioning tasks on directory objects in compliance with corporate administrative policies in corporate Active Directory and Exchange environments.

Active Roles provides consistent enforcement of corporate policies, a role-based administrative model, and flexible, rule-based administrative views, creating a reliable and secure environment for distributed administration and account provisioning.

NOTE: For information on the Active Roles features, see the latest *Active Roles Release Notes*.

Main Active Roles features

Active Roles provides out-of-the-box user and group account management, strictly enforced administrator-based role security, day-to-day identity administration and built-in auditing and reporting for Active Directory and Azure Active Directory (AD) environments. The following features and capabilities make Active Roles a practical solution for secure management of objects in AD and AD-joined systems:

- Secure access: Acts as a virtual firewall around AD, enabling you to control access through delegation using a least privilege model. Based on defined administrative policies and associated permissions generates and strictly enforces access rules, eliminating the errors and inconsistencies common with native approaches to AD management. Plus, robust and personalized approval procedures establish an IT process and oversight consistent with business requirements, with responsibility chains that complement the automated management of directory data.
- Automate object creation: Automates a wide variety of tasks, including:
  - Creating user, groups, and contacts in AD and Azure AD.
  - Creating mailboxes on Exchange Server and assigning licenses in Microsoft 365.
  - Managing on-premises Exchange and Exchange Online properties.

  Active Roles also automates the process of reassigning and removing user access rights in AD and AD-joined systems (including user and group deprovisioning) to ensure an efficient and secure administrative process over the user and group lifetimes. When user accesses must be changed or removed, updates are made automatically in Active Directory, Azure AD, Exchange, Exchange Online, SharePoint, Skype for Business, and Windows, as well as any AD-joined systems such as Unix, Linux, and Mac OS X.

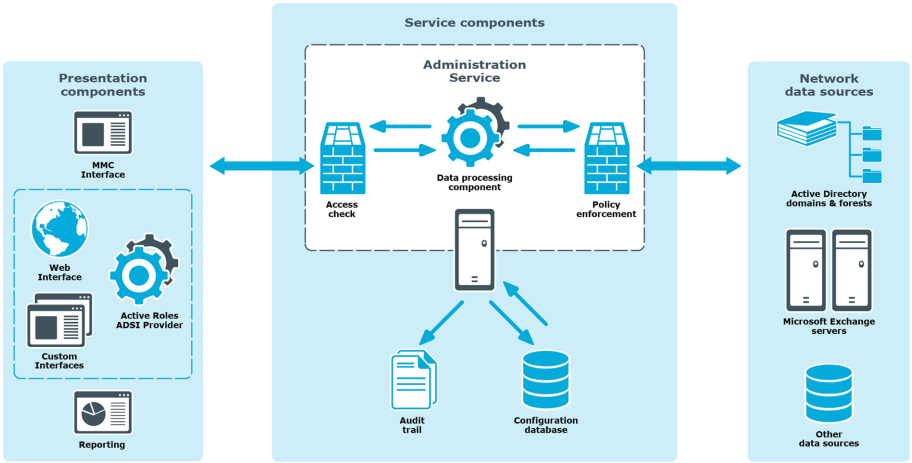
- Day-to-day directory management: Simplifies management of:
  - Exchange recipients, including mailbox assignment, creation, movement, deletion, permissions, and distribution list management.
  - Groups
  - IT resources, including computers, shared folders, printers, local users and groups.
  - AD, Azure AD, Exchange Online and AD LDS resources.

  Active Roles also includes intuitive interfaces for improving day-to-day administration and help desk operations via an MMC snap-in known as the Active Roles Console and a Web Interface.
- Manage users, groups, and contacts in a hosted environment: Provides Synchronization Service to operate in hosted environments where accounts from client AD domains are synchronized with host domains. Active Roles enables user, group, and contact management from the client domain to the hosted domain, while also synchronizing attributes and passwords.
- Consolidate management points through integration: Complements your existing technology and identity and access management strategy. Simplifies and consolidates management points by ensuring easy integration with many One Identity products and Quest products, including One Identity Manager, Safeguard Authentication Services, Defender, ChangeAuditor, and GPO Admin. Active Roles also automates and extends the capabilities of PowerShell, ADSI Provider, SPML Provider and the customizable Web Interface.

Technical overview of Active Roles

Active Roles divides the workload of directory administration and provisioning into three functional layers—presentation components, service components, and network data sources.

Figure 1: Active Roles Components



The presentation components include client interfaces for the Windows platform and the web, which allow regular users to perform a precisely defined set of administrative activities. The reporting solution facilitates automated generation of reports on management activities.

The service components constitute a secure layer between administrators and managed data sources. This layer ensures consistent policy enforcement, provides advanced automation capabilities, and enables the integration of business processes for administration of Active Directory, Microsoft Exchange, and other corporate data sources.

The Administration Database stores information about all permission and policy settings, and other data related to the Active Roles configuration.

On a very high level, the Active Roles components work together as follows to manipulate directory data:

1. An administrator uses the Active Roles Console or Web Interface to access Active Roles.
2. The administrator submits an operation request, such as a query or data change to the Administration Service.
3. On receipt of the operation request, the Administration Service checks whether the administrator has sufficient permissions to perform the requested operation (access check).
4. The Administration Service ensures that the requested operation does not violate the corporate policies (policy enforcement).
5. The Administration Service performs all actions required by the corporate policies, before committing the request (policy enforcement).
6. The Administration Service issues operating system function calls to perform the requested operation on network data sources.
7. The Administration Service performs all related actions required by the corporate policies, after the request is processed by the operating system (policy enforcement).
8. The Administration Service generates an audit trail that includes records about all operations performed or attempted with Active Roles. Directory-change tracking reports are based on the audit trail.

About presentation components

The presentation components include user interfaces to serve a variety of needs. The user interfaces accept commands, display communication, and give results in a clear, concise fashion.

About Active Roles Console (MMC Interface)

The Active Roles Console, also referred to as MMC Interface, is a comprehensive administrative tool that you can use to:

- Manage Active Directory and Microsoft Exchange resources.
- Configure organization-level access and administration policies.
- Set up or work with automation or approval workflows for your administrators or helpdesk personnel.

The Active Roles Console window is divided into two panes. The left pane contains the Console tree, showing the items that are available in the Console. The right pane, known as the details pane, displays information about items you select in the Console tree. You can perform most management tasks from this pane using commands on the **Action** menu.

Additional information is displayed in the lower sub-pane of the details pane when you check the **Advanced Details Pane** command on the **View** menu. You can perform management tasks from the lower sub-pane using commands on the **Action** menu.

About the Console tree

The left pane of the Active Roles Console contains the Console tree.

The Console tree root is labeled **Active Roles**. The name of the Administration Service is shown in square brackets. If you have Advanced view mode selected for Active Roles console display (**View** > **Mode**), the following folders are shown under the Console tree root:

- **Configuration**: Contains all Active Roles proprietary objects held in containers with appropriate names.
- **Active Directory**: Contains a list of domains registered with Active Roles. In this folder, you can browse domains for directory objects (users, group, computers), and perform management tasks on those objects.
- **AD LDS (ADAM)**: Contains a list of AD LDS directory partitions registered with Active Roles. In this folder, you can browse partitions for directory objects (users, group, containers), and perform management tasks on those objects.
- **Applications**: Contains a list of applications integrated with Active Roles, such as Reporting, and allows for quick access to those applications.

The Console display mode determines which folders are displayed in the Console tree. For more information, see About the view mode.

About the details pane

When you select an item in the **Console tree**, the details pane changes accordingly. To perform administrative tasks, click items in the details pane and use commands on the **Action** menu. The **Action** menu commands also appear on the shortcut menu that you can access by right-clicking items in the console tree or details pane.

By default, the objects listed in the details pane are sorted in ascending order by object name. You can change the sorting order by clicking a column heading. You can add and remove columns in the details pane using the **Choose Columns** command on the **View** menu.

In the Active Roles Console, you can apply filters to the details pane in order to search for directory objects. To configure a filter, select a domain, then click **Filter Options** on the **View** menu. It is also possible to find an object in the details pane by typing a few characters. This will select the first item in the sorted column that matches what you typed.

About the advanced details pane

The advanced pane appears at the bottom of the details pane if you check **Advanced Details Pane** on the **View** menu. You can use the advanced pane to administer an object selected in the **Console tree** or the **Details pane**: right-click an existing entry in the list to administer it, or right-click a blank area of the advanced pane to add a new entry.

The **Advanced details pane** is composed of a number of tabbed pages. The selected object determines which tabs are displayed. The possible tabs in the **Advanced details pane** and their descriptions are as follows:

- **Active Roles Security**: Lists the Active Roles Access Templates applied to the selected object. Under this tab, you can perform the following operations:
    - Apply additional Access Templates to the selected object.
    - Display Access Templates that affect the selected object owing to inheritance.
    - Synchronize from Active Roles security to Active Directory security.
- **Links**: Lists the objects to which the selected Access Template is applied.
- **Active Roles Policy**: Lists the Active Roles Policy Objects applied to the selected object. Under this tab, you can perform the following actions:
    - Apply additional Policy Objects to the selected object.
    - Display Policy Objects that affect the selected object owing to inheritance.
- **Native Security**: Lists the Active Directory permission entries specified for the selected object. Under this tab, you can perform the following actions:
    - Display permission entries that are inherited from parent objects.
    - Display default permission entries specified by the AD schema.
- **Member Of**: Lists the groups to which the selected object belongs. Under this tab, you can perform the following operations:
    - Add the selected object to groups.
    - Set the group as the primary group for the selected object.
- **Members**: Lists the members of the selected group. Under this tab, you can perform the following operations:
    - Add the selected object to groups.
    - Set the group as the primary group for the selected object.

NOTE: The Console displays the **Active Roles Security**, **Active Roles Policy**, and **Native Security** tabs for a selected object only if your user account has the **Read Control** right to the selected object.

Depending on the tab you have selected in the **Advanced details pane**, the toolbar displays the following buttons to help you work with the entries on the tab.

About the view mode

In the Active Roles console you can choose view mode—Basic, Advanced, or Raw. Changing view mode makes it possible to filter out advanced objects and containers from the display.

Basic mode displays Active Directory objects and Managed Units, and filters out objects and containers related to the Active Roles configuration. Basic mode should normally be used by delegated administrators and help-desk operators.

Advanced mode displays all objects and containers except those reserved for Active Roles internal use. Advanced mode is designed for administrators who are responsible for configuring the system and managing Active Roles proprietary objects.

Raw mode displays all objects and containers defined in the Active Roles namespace. This mode is primarily designed for troubleshooting.

With Raw mode, the console displays all data it receives from the Administration Service. With Basic or Advanced mode, some data is filtered out. For example, the **Configuration** folder is not shown in the console tree with Basic mode. Another example is the **Configuration Container** folder used to display the Active Directory configuration naming context, which is displayed with Raw mode only. In addition, there are some commands and property pages that are only displayed when the console is in Raw mode.

In short, when you choose Raw mode, the snap-in displays everything it is able to display. Otherwise, some items are hidden. Note that changing view mode does not modify any items. Rather, this only shows or hides particular items from the display.

To change view mode, click **Mode** on the **View** menu. In the **View Mode** dialog box, click **Basic Mode**, **Advanced Mode**, or **Raw Mode**.

About controlled objects

The Active Roles Console provides for visual indication of the objects to which Access Templates or Policy Objects are linked. The console marks those objects by adding an arrow icon at the lower-left corner of the icon that represents the object in the console tree or details pane. As a result, the icon looks similar to the following image:  .

To enable this feature, click **Mark Controlled Objects** on the **View** menu, and select check boxes to specify the category of object to be marked.

Trustees, permissions, roles and Managed Units

Active Roles offers three main security and administration elements:

- **Trustees**: Users or groups that have permissions to administer users, groups, computers, or other directory objects.
- **Permissions and Roles**: Permissions are grouped in Access Templates (roles) to define how a Trustee can manage directory objects.

- **Managed Units**: Collections of directory objects delegated to Trustees for administration.

The directory administrator defines which users or groups are designated as Trustees, which roles and permissions are assigned to Trustees, and what objects are included in Managed Units.

Managed Units are used to determine the directory objects that a Trustee can administer. As a Trustee, you can administer Managed Units for which you have assigned permissions. Managed Units containing objects you are authorized to administer are displayed under **Managed Units** in the **Console tree**.

When you select a Managed Unit in the **Console tree**, the details pane displays a list of objects included in that Managed Unit. To administer objects, select them from the list and use the commands on the **Action** menu.

If a Managed Unit includes a container, such as an Organizational Unit, the container is displayed under the Managed Unit in the **Console tree**. When you select a container in the **Console tree**, the details pane lists all child objects and sub-containers held in that container.

Finding directory objects

In the Active Roles Console you can search for objects of different types using the **Find** window. To access the **Find** window, right-click a container and click **Find**.

From the **In** list, you can select the container or Managed Unit you want to search. The list includes the container that you selected before activating the **Find** window. To add containers to the list, click **Browse**. From the **Find** list, you can select the type of the objects you want to find.

When you select an object type, the **Find** window changes accordingly. For example, **Users, Contacts, and Groups** searches for users, contacts, or groups using criteria such as user name, a note describing a contact, or the name of a group. In the **Find** list, Active Roles splits the **Users, Contacts, and Groups** category into three, providing the option for a more streamlined search.

By selecting **Custom Search** from the **Find** list, you can build custom search queries using advanced search options.

Using the **Find** window, you can search for any directory objects, such as users, groups, computers, Organizational Units, printers or shared folders. It is also possible to search for Active Roles configuration objects such as Access Templates, Managed Units, and Policy Objects. When you search for Access Templates, Policy Objects or Managed Units and select an appropriate object type from the **Find** list, the relevant container appears in the **In** list.

Once the search has completed, the objects matching the search criteria (search results) are listed at the bottom of the **Find** window. You can quickly find an object in the search results list by typing a few characters. This will select the first name that matches what you typed.

Once you have found the object, you can manage it by right-clicking the entry in the search results list, and clicking the applicable commands on the shortcut menu.

About Active Roles Web Interface

The Active RolesWeb Interface is a customizable web application for data administration and provisioning in Active Directory. With the Web Interface, an intranet user (such as a helpdesk agent or a delegated administrator) can connect to Active Roles using a web browser and perform day-to-day administrative tasks, including user management tasks, such as modifying personal data, or adding users to groups.

Web Interface users can perform administrative tasks and view or modify directory data. However, their scope of authority is limited by the rights delegated in Active Roles. A Web Interface user sees only the commands, directory objects, and object properties to which they have administrative access.

Administrators can customize the pages of the Web Interface, and administrators with the proper privileges can also add or remove commands or fields displaying property values.

Key features and benefits

The key features of the Active Roles Web Interface include the following:

- **Single-page lists**: All search results are listed on a single page, making it easier to sort, filter, locate and select the objects you want to manage.
- **Enhanced search tools**: To further facilitate searches, the Web Interface features a unified toolbar for configuring search conditions and filter conditions. This includes a flexible condition builder, allowing you to choose predefined conditions, configure a wide variety of property-based conditions, or specify complex conditions using LDAP syntax.
- **Pop-up property pages**: The pages for creating, viewing or changing objects appear on the top of the object list, allowing you to keep the list visible while selecting and managing individual objects.
- **Views**: The Web Interface allows you to create, save and reuse personal views for the various AD and Azure AD containers. Each view is essentially a search query for objects contained in a particular container, and returns the list of objects matching the specified search conditions, with the specified set of list columns and list sorting order.
- **Role-based web pages**: Active RolesWeb Interface supports multiple websites on the same intranet, each of them providing a separate, customizable set of menus, commands, and forms. By default, the Web Interface ships with three default pages: the Administrator Site, the Helpdesk Site, and the Self-Service Site.
- **Dynamic role-based configuration**: You can dynamically adapt the contents of any Web Interface site to align them to the roles of their Web Interface users. As such, you can make sure that a user can only see the commands, directory objects and object properties to which they have administrative access.
- **Point-and-click customization**: Administrators can customize the menus, commands, and pages of a site without writing a single line of code. As such, administrators can easily adapt the sites to any role, such as day-to-day

administrators, business data owners, helpdesk operators, or even regular end-users.

- **Active Directory and Azure AD support**: Users can administer a wide range of Active Directory and hybrid or cloud-only Azure AD resources, including users, groups, or computers.
- **Managing computer resources**: Users can manage the computer resources of your organization, such as printers, shares, services, devices, local users and groups.
- **User Profile Editor**: With the proper permissions configured, end-users can manage their personal or emergency data through an easy-to-use profile editor.
- **Enforcing organizational rules**: The Web Interface efficiently supplements and restricts user input based on the organizational rules defined with Active Roles. As such, the Web Interface sites display only property values generated according to the rules in effect, and prohibits users to enter values that violate the rules.
- **Single sign-on with integrated Windows authentication**: Active RolesWeb Interface supports single sign-on, without requiring users to enter their passwords again once they are logged in and authenticated by the operating system.
- **Localization support**: Besides English, the Active RolesWeb Interface supports the following languages:
    - Chinese (Simplified and Traditional)
    - French
    - German
    - Portuguese (Brazilian and European)
    - Spanish

In addition, the Web Interface also provides the following benefits:

- Individually customizable Web Interface sites, shipping with separate Administrator, Helpdesk, and Self-Service sites by default.
- User permission-based views for each page.
- Self-administration support.
- Attractive design with superior flexibility.
- Easy navigation with a simple layout and large UI elements, with most UI elements supporting resizing, collapsing or expanding. This allows you to adapt your UI workspace to your needs.

Getting started

To open the Active RolesWeb Interface, you must know:

- The name of the web server running the Web Interface component.
- The name of the Web Interface site you want to access.

When configuring the Web Interface, the Administration Service creates the following Web Interface sites by default:

- **ARWebAdmin**: The Administration Site, supporting a broad range of administrative tasks.
- **ARWebHelpDesk**: The Helpdesk Site, supporting the most common administrative tasks, typically performed by helpdesk personnel in an organization.
- **ARWebSelfService**: The Self-Service Site, allowing users to manage their own personal accounts.

To connect to a Web Interface site

1. In the web browser, enter the URL of the Web Interface site.

   For example, to connect to the default Administration Site, specify the following URL:

   `http://<server>/ARWebAdmin`

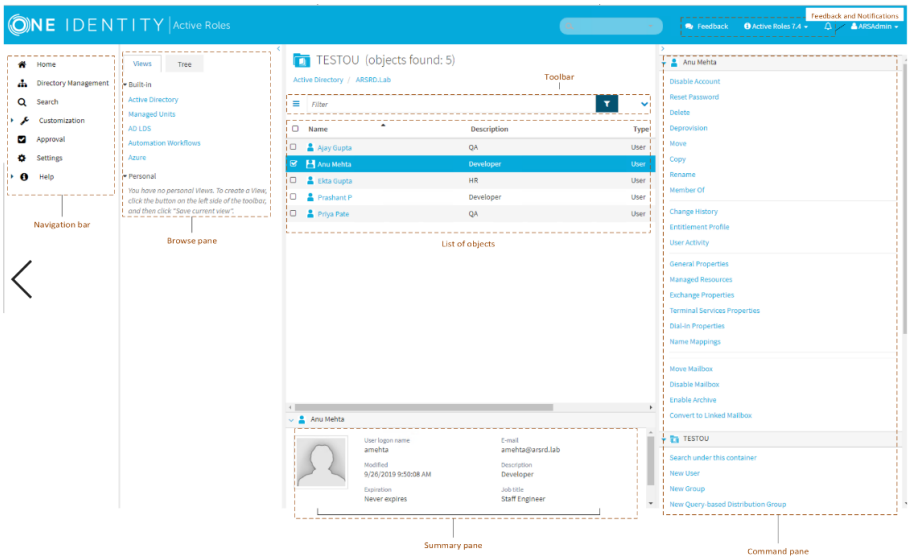
   In the above example, `<server>` is the name of the web server running the Web Interface.
2. To connect, press **Enter**.

About the Web Interface components

The Web Interface UI consists of the following main parts:

- The **Header** area at the top of the page, containing the **Quick Search bar**, the **Feedback** button, the **About** button, and the **Logout** option for your currently logged-in user.
- The **Navigation bar** and the **Browse pane** on the left side. The **Navigation bar** lets you navigate between the main Web Interface components, while the **Browse pane** shows the available **View** and **Tree** settings.
- The **Object list** next to the **Browse pane**.
- The **Toolbar** above the **Object list**, allowing you to manage the loaded **Object list**.
- The **Command pane** on the right side, containing the available administration actions for the selected object(s).
- The **Summary pane** under the list of objects, containing a short summary of data about the selected object(s).

Figure 2: UI Elements



The following sub-sections describe each UI component in more detail.

About the Web Interface Navigation bar

Located on the left side of the Web Interface UI, the **Navigation bar** provides the first level of navigation for most of the tasks you can perform with the Web Interface. The **Navigation bar** lists all major Web Interface areas, and provides access to the following pages and features:

- **Home**: Opens the Web Interface home page.
- **Directory Management**: Allows you to browse and administer the AD and Azure AD objects in your organization.
- **Search**: Allows you to search and administer the AD and Azure AD objects in your organization.
- **Customization**: Allows you to customize your Web Interface pages.

  NOTE: This option is available for Active Roles Admin users only.
- **Approval**: Allows you to perform tasks related to the approval of administrative operations.
- **Settings**: Contains your personal settings related to displaying the Web Interface.

About the Web Interface Browse pane

Located next to the **Navigation bar**, the **Browse pane** lists the built-in **Views** and **Personal views**, and also lets you access the **Tree** view:

- Built-in **Views** are default entry points for browsing objects in your AD and Azure AD environment. **Personal views**, on the other hand, are filter or search queries that

you can build and save to use later.

- The **Tree** view helps you browse AD and Azure AD objects via the hierarchical directory tree structure of the containers.

About the Web Interface Object list

When you select a container or view in the **Browse pane**, a list of objects appears. If you select a container (such as an Organizational Unit, or OU), the list includes the objects held in that container. If you select a view, the list includes the objects that match the view settings. The object list appears on a single page, allowing you to search the entire contents of the container or view easily.

In addition to browsing the object list, you can also:

- Use various built-in conditions or create custom conditions to filter the object list. You can also customize the list by sorting and filtering, and by adding or removing list columns.
- Select objects from the list and apply administration commands to the selected object(s). When you click the name of a container object, such as a domain or an OU, the list changes to display the objects held in that container, allowing you to browse through containers in the directory.

About the Web Interface Toolbar

Located above the list of objects, the **Toolbar** contains a number of controls allowing you to manage the current **Object list**:

- To save the current object list as a **Personal view**, add or remove list columns, or export the list to a text file, click **Menu** on the left side of the **Toolbar**.
- To filter the results of the **Object list**, enter your filtering conditions in the **Filter** field, then click the button next to the field.
- To configure filtering criteria based on object properties, click **Expand/Collapse** on the right side of the **Toolbar**. To have the list include only the objects that match your filtering criteria, click the button next to the **Filter** field.

About the Web Interface Command pane

Located to the right of the **Object list**, the **Command pane** provides administrative commands you can apply to the objects you select from the list, as well as commands you can apply to the current container.

- If no objects are selected in the **Object list**, the **Command pane** includes only the commands that apply to the current container. These commands are grouped under a heading that shows the name of the current container.
- If a single object is selected in the list, the **Command pane** also contains the set of commands that are applicable to the selected object, under a heading that shows the name of the object.

- If multiple objects are selected from the list, the **Command pane** contains a set of batch commands that apply to all of the objects, under a heading that shows the number of the selected objects.

About the Web Interface Summary pane

The **Summary pane** provides information about the selected object under the **Object list**. The information shown on the pane includes the commonly used properties of the object, and depends on the object type.

For example, user properties provide more detailed information about a user account, such as the login name, email address, description, job title, department, expiration date, and the date and time when the account was last changed.

TIP: If you do not see the **Summary pane**, click the area below the **Object list**.

About the Web Interface Personal views

Personal views provide a filter-based object list, with the objects either belonging to the same container, or to the same search query. When searching a container (such as an OU), you can filter the search using either via search conditions or filter conditions as you need, then save the resulting search or filter query as your **Personal view**.

The **Personal view** shows the list of objects that match your specified conditions, with the specified list sorting order and set of list columns. **Personal views** are stored on a per-user basis, so each user can have their own views.

Using Personal views

To locate directory objects, the Active Roles Web Interface lets you use search or filter queries. When creating a query, you specify a set of rules that determine the contents of the resulting **Object list**. You can, for instance, specify to list only user accounts from a specific OU. In addition, you can adjust the set of columns and the sort order in the list of search or filtering results.

Locating objects quickly and easily is a critical Web Interface feature, as you need to focus your attention only on the objects you actually need to manage. However, creating a search or filter query that displays the objects you are interested in for a particular task can be time-consuming.

Personal views provide a way for you to save that work. Once you created a query that shows just the objects you need, you can set a name for the query and save it for use later. That saved query is a **Personal view**. Each view saves the following settings that you specify:

- The container to search or filter.
- The search or filtering criteria.
- The set of columns and the sort order in the list of search or filtering results.

Creating a Personal view

Personal views are like search or filter queries that you named and saved. After creating a **Personal view**, you can reuse it without re-creating its underlying search or filter query. To reuse a personal view, click the name of the view on the **Views** tab in the **Browse pane**. The Web Interface then applies the search or filter query saved in the view, and displays the results in the list with the same set of columns and sort order with which you created the view.

To create a personal view

1. Configure and perform a search, or create a filtered list of objects.
2. On the left side of the **Toolbar**, click **Menu**, then click **Save current view**.
3. In the dialog that appears, specify a name for the **Personal view**, then click **Save**.

Locating directory objects in the Web Interface

The Active Roles Web Interface provides search and filtering tools to help you locate directory objects quickly and easily. By creating and applying a proper search or filter query, you can build shorter object lists, which makes it easier to select the objects needed to accomplish your administrative tasks.

You can also save search and filter queries as your **Personal views**, and use them again at a later time. Each view saves the following settings that you specify:

- The container to search or filter.
- The search or filtering criteria.
- The set of columns.
- The sort order in the list of search or filtering results.

Searching for directory objects in the Web Interface

To search for directory objects, use the **Search** page that allows you to select the container to search and specify criteria for the objects you want to find. The Web Interface runs searches both in the selected containers and their subcontainers.

The Web Interface opens the **Search** page when you do any of the following:

- Type in the **Search** field located in the upper right corner of the **Web Interface** window, then press **Enter** or click the magnifying glass icon in the **Search** field. In this case, the Web Interface will search all managed AD or Azure AD domains for objects whose naming properties match what you specified, and the **Search** page will list the search results accordingly. The naming properties include name, first name, last name, display name, and login name.
- Click **Search** on the **Navigation bar**. This will open the **Search** page, allowing you to configure and start a search.

To configure and start a search

1. On the **Toolbar** of the Web Interface, click **Search in**, then select the container that you want to search. You can select more than one container.
2. Specify the criteria for the objects that you want to find:
    - To search by naming properties, enter them in the **Search** field on the **Toolbar**. The Web Interface will then search for objects whose naming properties match your criteria. The naming properties include name, first name, last name, display name, and logon name.
    - To search by other properties, expand the **Toolbar** by clicking the button on the right side of the **Toolbar**, click **Add criteria**, choose the properties by which you want to search, and click **Add**. Then, configure the criteria as appropriate. The Web Interface will search for objects that match the criteria you configured.
3. To start the search, press **Enter**.

The search results then appear on the **Search** page.

TIP: You can customize the list by adding or removing list columns and sorting the list by column data. To add or remove list columns, click **Menu** on the left side of the **Toolbar**, then click **Choose columns**. To sort the list by column data, click the column headings.

Example: Searching by object type

This example procedure shows how to use the **Search** option to list all groups that exist in the AD domains managed by Active Roles.

To list all groups in all AD environments managed by Active Roles

1. On the **Navigation bar** of the Web Interface, click **Search**.
2. Expand the **Toolbar** by clicking the button on the right side of the **Toolbar**, click **Add criteria**, select **Object type is User/InetOrgPerson/Computer/Group/Organizational Unit**, then click **Add**.
3. On the **Toolbar**, click **Group** in the list next to **The object type is**, then press **Enter**.

Filtering the contents of a container in the Web Interface

If a container, such as an OU in an AD, holds many objects, you can narrow down the displayed list of objects by filtering them.

To filter the objects held in a container

1. In the Active Roles Web Interface, navigate to the container whose contents you want to filter.

   To navigate to a container:

   - Search the container object, then click its name in the list of search results on the **Search** page.
   - Alternatively, browse the container objects with the **Browse pane** and the **Object list**.

   NOTE: The scope of filtering is always set to the current container, and does not include any subcontainers of that container. Filtering is essentially a search for objects stored in a given container only. If you want to search the current container and all of its subcontainers, click **Search under this container** in the **Command pane**, and configure a search instead.

2. Specify how you want to filter the objects of the container.

   - To filter objects by naming properties, specify your criteria in the **Filter** field on the **Toolbar**, then press **Enter**. Alternatively, click the button next to the **Filter** field. The list of objects will include only the objects whose naming properties match what you typed. The naming properties include name, first name, last name, display name, and login name.
   - To filter objects by other properties, expand the **Toolbar** by clicking the button on the right side of the **Toolbar**, click **Add criteria**, choose the properties by which you want to filter, and click **Add**. Then, configure the criteria as you need. The list of objects will include only the objects that match the criteria you configured.

3. To apply the filter, press **Enter** or click the button next to the **Filter** field on the **Toolbar**.

When the Active Roles Web Interface applies the configured filter, it lists a subset of all objects held in that container.

TIP: To view all objects again, remove the filter.

- If you did not specify any criteria, clear the **Filter** field on the **Toolbar**, and press **Enter**.
- If you specified any criteria, expand the **Toolbar**, click **Clear all**, and press **Enter**.

Example: Filtering by object type

This example procedure shows how to configure a filter that lists only user accounts in a specific OU, removing all other objects from the list.

To filter to user accounts in a specific OU

1. In the Active Roles Web Interface, navigate to the OU.
2. To expand the **Toolbar**, click the button on the right side of the Toolbar. Then, click **Add criteria**, select **Object type is User/InetOrgPerson/Computer/Group/Organizational Unit**, and click **Add**.
3. On the **Toolbar**, confirm that the field next to **The object type is** reads **User**, then either click the button next to the **Filter** field, or press **Enter**.

About Active Roles Management Shell

Part of the Active Roles Management Tools, the Management Shell provides Windows PowerShell-based command-line tools (cmdlets), allowing you to run and automate administrative tasks in Active Roles.

These Management Shell cmdlets are shipped in two modules.

ActiveRolesManagementShell

The **ActiveRolesManagementShell** module provides cmdlets for the following administration operations:

- Managing users, groups, computers and other Active Directory (AD) objects via Active Roles.
- Managing digital certificates.
- Administering certain Active Roles objects.

The names of the cmdlets provided by this module start with the `QAD` or `QARS` prefixes, such as `New-QADUser`, `Add-QADCertificate`, or `New-QARSAccessTemplateLink`.

ActiveRolesConfiguration

The **ActiveRolesConfiguration** module (also known as the "Configuration Shell") provides cmdlets for configuring Active Roles Administration Service instances and Web Interface sites. The names of the cmdlets provided by this module start with the `AR` prefix, such as `New-ARDatabase`, `New-ARService`, or `New-ARWebSite`.

NOTE: Consider the following when planning to use the **ActiveRolesConfiguration** module:

- This module is available on 64-bit operating systems only.
- You can only install this module on computers where the Administration Service or Web Interface modules are also installed. Otherwise, the module will not provide all cmdlets.

For more information, see About Active Roles Configuration Shell.

Getting started

You can start using the Management Shell component from the Windows Start menu or the Apps page, depending on the version of the operating system.

To start the Active Roles Management Shell

1. Log in to the computer where the Administration Service or the Management Shell is installed.
2. To start the Active Roles Management Shell, in the Windows Start menu or the Apps page, click **Active Roles 8.2.1 Management Shell**.
3. To view the reference manual providing detailed information about the available cmdlets, in the Management Shell command-line interface, enter `QuickRef`, then press **Enter**.
4. To load the available modules and access their cmdlets, enter the `Import-Module` command, then press **Enter**.

About custom interfaces

In addition to the Active Roles Console and the Web Interface, Active Roles enables the development of custom interfaces that use the Active Roles ADSI Provider to access the features of Active Roles. Administrators familiar with scripting and programming can create custom interfaces to meet the specific needs of their network administration.

About Active Roles ADSI Provider

The Active Roles ADSI Provider operates as part of the presentation components to enable custom user interfaces and applications to access Active Directory services through Active Roles. The Active Roles ADSI Provider translates client requests into DCOM calls and interacts with the Active Roles Administration Service.

The Active Roles ADSI Provider allows custom scripts and applications, such as web-based applications, to communicate with Active Directory, while taking full advantage of the security, workflow integration and reporting benefits of Active Roles. For example, using the Active Roles ADSI Provider, you can create web-based helpdesk operator pages for property modifications, restricted by the corporate rules enforced with Active Roles.

About Active Roles Reporting

Active Roles offers comprehensive reporting to monitor administrative actions, corporate policy compliance, and the state of directory objects. The Active Roles reporting solution includes Data Collector and Report Pack.

Report Pack provides report definitions for creating reports based on the data gathered by Data Collector. Active Roles comes with an extensive suite of report definitions that cover all administrative actions available in this product.

Report Pack is deployed on Microsoft SQL Server Reporting Services (SSRS). You can use the tools included with SSRS to view, save, print, publish, and schedule Active Roles reports.

Data Collector is used to gather data required for reporting. The Data Collector Wizard allows you to configure and schedule data collection jobs.

Once configured, Data Collector retrieves data from various sources, accessing them via the Active Roles Administration Service, and stores the data in a SQL Server database. Data Collector also provides a means for managing the gathered data, including the ability to export or delete obsolete data.

Overview of service components

At the core of Active Roles lies the Administration Service. It features advanced delegation capabilities and ensures the reliable enforcement of administrative policies that keep data current and accurate. The Administration Service acts as a bridge between the presentation components and network data sources. In large networks, multiple Administration Service instances can be deployed to improve performance and ensure fault tolerance.

About the data processing component

The data processing component accepts administrative requests and validates them by checking permissions and rules stored in the Administration Database. This component manages the network data sources, retrieving or changing the appropriate network object data based on administrative requests and policy definitions.

The data processing component operates as a secure service. It logs on with domain user accounts having sufficient privileges to access the domains registered with Active Roles (managed domains). The access to the managed domains is limited by the access rights of those user accounts.

About the configuration database

The Administration Service uses the configuration database to store configuration data. The configuration data includes definitions of objects specific to Active Roles, assignments of administrative roles and policies, and procedures used to enforce policies. The configuration database is only used to store Active Roles configuration data. It does not store copies of the objects that reside in the managed data sources, nor is it used as an object data cache.

Active Roles uses Microsoft SQL Server to host the configuration database. The replication capabilities of SQL Server facilitate implementation of multiple equivalent configuration databases used by different Administration Service.

Active Roles now supports database configuration on on-premises databases and Azure SQL databases. You can configure Azure SQL database variants, such as Azure SQL database, Azure SQL Managed instance and Azure SQL Elastic Pool in Active Roles.

NOTE: Active Roles supports database configuration over encrypted SQL Server configurations. For more information, see Knowledge Base Article Is SQL Server encryption supported? on the *One Identity Support Portal*.

About the audit trail

The data processing component provides a complete audit trail by creating records in the event log on the computer running the Administration Service. The log shows all actions performed and by whom, including actions that were not permitted. The log entries display the success or failure of each action, as well as which attributes were changed.

About network data sources

Through the Administration Service, Active Roles accesses and controls the object data stored in the following data sources:

- Active Directory domains and forests: Provides the directory object information in Active Directory domains.
- Microsoft Exchange Server: Provides information about mailboxes maintained by Microsoft Exchange.
- Azure AD: Provides information about users in Azure Active Directory.
- Microsoft 365: Provides information about users in Microsoft 365.
- Exchange Online: Provides information about users in Exchange Online.
- Other data sources: Provides information about objects that exist outside of Active Directory. This includes information from corporate databases, such as human resources databases, and information about computer resources, such as services, printers, and network file shares.

Active Roles is designed to help with the use and management of these data sources. Directory administrators can define and enforce business rules and policies to ensure that the data in the managed data sources remains current and accurate.

With Active Roles, you can utilize the information stores from a wide variety of data sources in your network, such as human resource data or inventories. You can use scripting to integrate these important data sources. This reduces the duplication of work, reduces data pollution, and allows for the validation of information that is often stored in more than one database.

Active Roles makes it possible for a custom script to receive control upon a request to perform an administrative operation, such as object creation, modification, or deletion. Custom scripts can be invoked through Policy Objects, which Active Roles uses to enforce corporate rules. For example, you could implement a Policy Object containing a custom script that will receive control whenever Active Roles is requested to create a user object in a certain OU.

The Policy Object could be configured so that Active Roles continues with the user creation only after a certain piece of the script (the pre-create event handler) has successfully executed. In this way, the script prohibits the creation of user objects whose properties violate corporate rules. It prevents the population of object properties with values taken from external data sources, and generates default property values in accordance with the corporate rules.

The Policy Object may also be configured to pass control to another piece of the script (the post-create event handler) immediately after a user object is successfully created. This enables the script to trigger additional actions, required by corporate rules, after the object has been created. For example, it can update external data stores, provision the user with access to resources, and notify that the user object has been created.

About security and administration elements

Active Roles offers three key security and administration elements, which are stored as objects in the Administration Database:

- Access Templates
- Policy Objects
- Managed Units

These elements enable any user or group in Active Directory to be given limited and effectively controlled administrative privileges.

Users and groups that are given administrative permissions in Active Roles are referred to as **Trustees**. Trustees can be assigned to Managed Units or directory objects and containers.

Trustees do not need special administrative rights within Active Directory. To give Trustees access to Active Directory, Active Roles implements proxy mechanisms that use Access Templates to specify the level of access. When Trustees exercise their access permissions, these mechanisms use Policy Objects to trigger additional actions, such as running integration scripts and validating input data.

When designating a user or group as a Trustee, you must specify the Access Templates that control what the Trustee can do. Permissions granted to a group are extended to all members of that group. To reduce administration time, administrative control should be delegated to groups, rather than to individual users.

To implement policy constraints and automation, you must configure and apply Policy Objects that invoke built-in or custom procedures upon administrative requests. Policy procedures may include running custom scripts to synchronize Active Directory data with

other data sources, performing a data validity checkup, and initiating additional administrative operations.

Access Templates for role-based administration

An **Access Template** is a collection of permissions that define what actions can be performed by an administrative role. Active Roles applies Access Templates to directory objects, containers, and administrative views (Managed Units) in relation to groups and users designated as Trustees.

Active Roles offers an extensive suite of preconfigured Access Templates that represent typical administrative roles, enabling the correct level of administrative authority to be delegated quickly and consistently. Access Templates significantly simplify the delegation and administration of management rights, speed up the deployment of the delegation model, and reduce management costs. For more information on the built-in Access Templates available in Active Roles, see the *Active Roles Built-in Access Templates Reference Guide* document.

Access Templates enable centralized administrators to define administrative roles with various levels of authority, speeding up the deployment of access control and streamlining change tracking of permission settings across the enterprise.

It is also possible to create custom Access Templates based on business requirements. Custom Access Templates can be modified at any time. When an Access Template is modified, the permission settings on all objects where that Access Template is applied change accordingly.

Policy Objects to enforce corporate rules

A Policy Object is a collection of administrative policy definitions that specify corporate rules to be enforced. Access Templates define who can make changes to a piece of data, and Policy Objects control what changes can be made to the data. Active Roles enforces corporate rules by linking Policy Objects to:

- Administrative views (Managed Units)
- Active Directory containers
- Individual (leaf) directory objects

Policy Objects define the behavior of the system when directory objects are created, modified, moved, or deleted. Policies are enforced regardless of the Trustee permissions.

A Policy Object includes stored policy procedures and specifications of events that activate each procedure. Based on policy requirements, a policy procedure could:

- Validate specific property values.
- Allow or deny entire operations.
- Trigger additional actions.

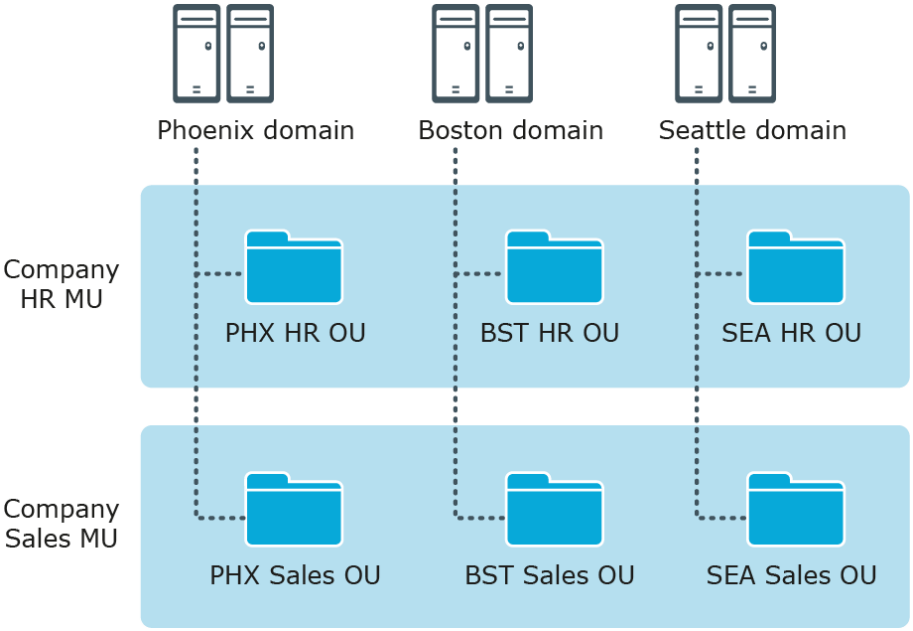
A Policy Object associates specific events with its policy procedures, which can be built-in procedures or custom scripts. This provides an easy way to implement sophisticated validation criteria, synchronize different data sources, and combine a number of administrative tasks into a single batch.

Managed Units to provide administrative views

A Managed Unit is a collection of objects collectively managed with Active Roles, created for the distribution of administrative responsibilities, enforcement of business rules and corporate standards, and management of complex network environments. Using Managed Units, the management framework can be separated from the Active Directory design. Directory objects can easily be grouped into administrative views, regardless of their location in Active Directory.

For example, the Active Directory design might be based on geographic location, with domains named after cities or regions and Organizational Units named after corporate departments or groups. However, Managed Units could be designed to manage specific departments or groups that are divided across multiple geographic locations.

Figure 3: Managed Units



In this example, each AD domain has a Human Resources (HR) OU and a Sales OU. The Active Roles design has an HR MU and a Sales MU. The HR MU enables administrators to configure the policies and security restrictions needed for all HR users regardless of their location, while the Sales MU enables the same for all Sales users.

Managed Units are defined with the use of membership rules—criteria used by Active Roles to evaluate whether or not an object belongs to a given Managed Unit. This enables Managed Units to dynamically change as the network environment changes. For example, you can define a Managed Unit by specifying rules that include all objects whose properties

match specific conditions. The specified rules will force the new or modified objects to be members of the correct Managed Unit.

Managed Units extend the functionality of Organizational Units (OUs), providing convenient scope to delegate administration and enforce corporate rules. A Managed Unit has the following characteristics:

- Represents a collection of objects (one object can belong to more than one Managed Unit).
- Supports rule-based specifications for its members (a Managed Unit only holds objects that satisfy the membership rules specified for the Managed Unit).
- Can hold directory objects that reside in different Organizational Units, domains, forests, and other Managed Units.

Active Roles ensures that permission and policy settings specified for a Managed Unit are inherited by all objects that belong to that Managed Unit. When a directory container belongs to a Managed Unit, all child objects in that container inherit the permission and policy settings defined at the Managed Unit level. This inheritance continues down the directory tree within all container objects that are members of the Managed Unit.

About Active Directory security management

The Active Roles Console makes it easy to examine and manage permission entries in Active Directory, by showing the access available to each user, along with the scope of their access. A centralized view of all permission entries for any given object helps with the analysis and administration of permissions in Active Directory. For each permission entry, the view displays a number of entry properties, including the permission description, origin, and security principal. From the main window, additional properties can be displayed and the native security editor can be accessed.

The centralized display of native security allows the administrator to quickly view permissions assigned to objects in Active Directory, and to determine whether the permission is inherited. The list of permission entries can be sorted by security principal name to determine who has access to the selected object. If a permission entry is inherited, Active Roles identifies the object from which the permission originates, so that the administrator can easily find and edit the permission entry for that object.

The Active Roles Console provides the capability to view the permissions for an object by simply clicking the object to display the permission entries in a centralized view. This makes it easier for the administrator to verify the permissions on security-sensitive objects, and to identify possible security problems.

Management of native security

Active Roles Access Templates can be used to specify permissions in Active Directory. Designed to support the role-based grouping of permissions, Access Templates provide an

efficient mechanism for setting and maintaining access control, simplifying and enhancing the management of permissions in Active Directory.

To provide this capability, Active Roles gives the administrator the option to keep Active Directory native security updated with selected permissions specified using Access Templates. This option, referred to as Permissions Propagation, is intended to provision users and applications with native permissions to Active Directory. The normal operation of Active Roles does not rely on this option.

For Active Roles permission entries with the Permissions Propagation option set, Active Roles generates Active Directory native permission entries in accordance with the Active Roles permissions. Once set, the option ensures that every time Active Roles permission assignments or templates change, the associated native permission entries change accordingly.

Customization using ADSI Provider and script policies

Active Roles offers the facility to customize its off-the-shelf functionality using scripts and applications that interact with the Administration Service. It allows a high degree of customer modification to meet specific business and organizational needs. This gives customers greater flexibility when using the product, and enables them to build solutions that can easily be integrated with existing systems and data.

The following list shows some of the ways in which the product can be customized:

- Using the Active Roles ADSI Provider, the existing proprietary applications or custom web-based interfaces could communicate with Active Roles to perform administration and provisioning tasks on user accounts and groups.
- Using policy scripts, custom corporate rules could be enforced to regulate data format and administrative workflows.
- Using policy scripts, the data stored in an HR database or ERP system could be incorporated into the administration and provision of users.

Active Roles makes it possible for user-developed scripts and applications to manipulate directory objects through the Administration Service (known as **persistent objects**), and to take control of objects that are in the process of being created, modified, or deleted with Active Roles (**in-process objects**).

Having programmatic access to persistent and in-process objects makes it easy for developers to customize Active Roles in these two areas:

- Creating custom applications and user interfaces.
- Enforcing corporate administrative policies by running custom scripts (known as **script policies**).

About custom applications and user interfaces

A custom application or user interface can be created to manipulate directory objects in Active Roles. Active Roles offers the ADSI Provider to communicate with the Administration Service using standard COM interfaces that conform to the Microsoft ADSI 2.5 specification.

Custom applications are executables that provide, retrieve and process data to or from the Administration Service. For example, an organization with a separate human resources database could develop and deploy a custom application that extracts personal information from the database, then passes it to the Administration Service to facilitate user account provisioning.

Custom user interfaces are usually web-based interfaces that distribute certain tasks to users. Custom user interfaces can also be used to streamline the workflow of network administrators and help-desk operators. For example, web-based pages could be created so that helpdesk operators only see the fields related to user properties that they can view and modify, according to their corporate standards.

Both custom applications and user interfaces rely on the Active Roles ADSI Provider to access the functionality of Active Roles.

About custom script policies

Active Roles provides the ability to implement administrative policies by running user-developed scripts. This makes it possible to:

- Facilitate the provisioning of user accounts: Populate user properties through external database integration and automate multi-step provisioning tasks.
- Maintain the integrity of directory content: Prevent inconsistency of Active Directory data by enforcing update-sequence and data-format policies across the enterprise.
- Enforce business rules: Maintain security design and capture administration expertise by integrating business rules into the administrative workflow.

Once configured, the custom script-based policies are enforced without user interaction. Active Roles automatically handles the execution of policy scripts that supplement particular administrative operations and trigger additional administrative actions. For example, policy scripts can be used to:

- Perform a sophisticated validity check on input data.
- Synchronously change information in multiple data sources, such as the Active Directory store, Microsoft Exchange Server, and HR or ERP-system database.
- Ensure that delegated administrators follow a prescribed administrative workflow.
- Link multiple administrative tasks into one operator transaction.

About dynamic groups

Active Roles helps streamline group maintenance by defining group membership dynamically, with rule-based membership criteria. Dynamic group membership eliminates the need to manually update membership lists for security and distribution groups.

To automate the maintenance of group membership lists, Active Roles provides:

- Rule-based mechanism that automatically adds and removes objects to groups whenever object attributes change in Active Directory.
- Flexible membership criteria that enable both query-based and static population of groups.

The membership criteria fall into these categories:

- **Include Explicitly**: Ensures that specified objects are included in the membership list, regardless of any changes made to the objects.
- **Include by Query**: Populates the membership list with objects that have certain properties. When an object is created, or when its properties are changed, Active Roles adds or removes it from the membership list (depending on whether the object's properties match the search criteria).
- **Include Group Members**: Populates the membership list with members of specified selected groups. When an object is added or removed from the selected groups, Active Roles adds or removes that object from the membership list.
- **Exclude Explicitly**: Ensures that specified objects are not in the membership list, regardless of any changes made to the objects.
- **Exclude by Query**: Ensures that objects with certain properties are not in the membership list. Active Roles automatically removes objects from the membership list (depending on whether the objects' properties match the search criteria).
- **Exclude Group Members**: Ensures that members of specified groups are not in the membership list. When an object is added to any one of the selected groups, Active Roles automatically removes that object from the membership list.

These membership criteria are also applicable to Managed Units.

About workflows

Active Roles provides a rich workflow system for directory data management automation and integration. Based on the Microsoft Windows Workflow Foundation technology, this workflow system enables IT to define, automate and enforce management rules quickly and easily. Workflows extend the capabilities of Active Roles by delivering a framework that enables combining versatile management rules such as provisioning and deprovisioning of identity information in the directory, enforcement of policy rules on changes to identity data, routing data changes for approval, e-mail notifications of particular events and conditions, as well as the ability to implement custom actions using script technologies such as Microsoft Windows PowerShell or VBScript.

Suppose you need to provision user accounts based on data from external systems. The data is retrieved and then conveyed to the directory by using feed services that work in conjunction with Active Roles. A workflow can be created to coordinate the operations in account provisioning. For example, different rules can be applied for creating or updating accounts held in different containers.

Workflows may also include approval rules that require certain changes to be authorized by designated persons (approvers). When designing an approval workflow, the administrator specifies which kind of operation causes the workflow to start, and adds approval rules to the workflow. The approval rules determine who is authorized to approve the operation, the required sequence of approvals, and who needs to be notified of approval tasks or decisions.

By delivering email notifications, workflows extend the reach of management process automation throughout the enterprise. Notification activities in a workflow let people be notified via email about events, conditions or tasks awaiting their attention. For example, approval rules can notify of change requests pending approval, or separate notification rules can be applied to inform about data changes in the directory. Notification messages include all necessary supporting information, and provide hyperlinks enabling message recipients to take actions using a standard web browser.

The logic of an automated management process can be implemented by using administrative policies in Active Roles. Yet creating and maintaining complex, multi-step processes in that way can be challenging. Workflows provide a different approach, enabling IT administrators to define a management process graphically. This can be faster than building the process by applying individual policies, and it also makes the process easier to understand, explain and change.

About workflow features and activities

Active Roles supports the following major workflow features and activities:

- Saving object properties
- Modifying requested changes
- Using initialization scripts
- Searching for expiring users
- Sending plain-text notification messages

Getting started

To get started with workflows, see the following resources:

- For more information on the listed workflow features and activities, see the linked sections.
- For more information on workflows in general, see *Workflows* in the *Active Roles Administration Guide*.

Workflows to save object properties

Workflows configured in the Active Roles Console support saving object properties when running the workflow with the **Saving Object Properties** activity. The properties are saved in the workflow data context and can be retrieved by other workflow activities either before or after the object changed.

Saving object properties is useful for situations that require knowing not only the current state or properties of the changed object, but also its previous states or property values. Such earlier states or property values may be required for informational, archival or decision making purposes.

For example, to notify users and administrators of object deletions, you can create a workflow that:

1. Starts when requesting the deletion of the object.
2. Saves the name of the object to be deleted.
3. After the object is deleted, it sends a notification message with the saved name of the deleted object.

Workflow configuration options

The **Saving Object Properties** activity has the following configuration options:

- **Activity target**: Specifies the object whose properties will be saved. The available settings are the following:
    - **Workflow target object**: Specifies the target object of the request in a change workflow that started the workflow.

      For example, in case of a change workflow starting with the delete request of an object, selecting this setting will result in the activity saving the properties of the object to be deleted.
    - **Fixed object in directory**: Specifies a particular object that you select in Active Directory.
    - **Object identified by workflow parameter**: Specifies the object via the value of a certain parameter in the workflow. You can select the parameter from the workflow definition.
    - **Object from workflow data context**: When selected, the activity will select the object based on the workflow environment data collected while running the workflow. You can select the object for the activity when the workflow is initiated.
    - **Object identified by DN-value rule expression**: Specifies the object via its Distinguished Name (DN) by the string value of a certain rule expression. By using a rule expression, you can compose a string value based on the properties of various objects found in the workflow environment when running

the workflow. You can create the desired rule expression when you configure the activity.

- **Target properties**: Specifies the object properties you want the activity to save. The Workflow Designer contains a default list of properties; however, you can change the list as you need.

  By default, the activity saves all single-value non-constructed attributes found in the directory schema of the target object, including custom virtual attributes added to the directory schema by Active Roles.

- **Notification**: Configures notifications for the runs of the activity, and subscribes recipients to the following notification events:

  - **Activity completed successfully**: Sends a notification email if no significant errors occurred during the run of the activity.
  - **Activity encountered an error**: Sends a notification email if significant errors occurred during the run of the activity.

  The notification settings specify the notification events and recipients. When run by the workflow, the activity prepares a notification message according to the specified event. Active Roles retains the message prepared by the activity, and sends the message to the specified recipients when the event occurs.

- **Error handling**: Specifies the action to take when detecting any errors. Selecting **Continue workflow even if this activity encounters an error** will suppress any errors detected by Active Roles during the workflow run. Leaving this setting clear will result in Active Roles stopping the workflow if the activity detects any errors. By default, this setting is not selected.

Retrieving saved properties

If you use any workflows that include the **Save Object Properties** activity, you can configure additional activities to retrieve the object property information saved by the **Save Object Properties** activity. You can do this by three means:

- Using a **Script** activity with the following expression:

  ```
  $workflow.SavedObjectProperties("activityName").get("attributeName")
  ```

  In this expression, `activityName` is the name of the **Save Object Properties** activity, while `attributeName` is the LDAP display name of the attribute representing the property you want the script to retrieve.

  NOTE: You must specify an attribute listed in the **Target properties** setting of the **Save Object Properties** activity. Otherwise, the expression will return no property value during runtime.

- Adding the **Workflow - Saved Object Properties** token to the notification message template. To do so:

1. In the **Insert Token** dialog, in the list of tokens, click **Workflow - Saved Object Properties**, then click **OK**.
2. In the dialog that appears, select the name of the **Save Object Properties** activity and the saved property you want the token to retrieve.

NOTE: You must specify an attribute listed in the **Target properties** setting of the **Save Object Properties** activity. Otherwise, the token you configured will return no property value during runtime.

- If you use an **If-Else** branch condition, a **Search** filter, or a **Create**, **Update** or **Add Report Section** activity, by selecting the **Property of object from workflow data context** configuration option. To do so:
  1. In the **Object Property** dialog, click the link in the **Target object** field, then click **More choices**.
  2. In the dialog that appears, click **Saved Object Properties**. Then, in the **Activity** list, select the name of the **Save Object Properties** activity and click **OK**.
  3. In the **Object Property** dialog, click the link in the **Target property** field, then select the property you want.

  NOTE: You must specify an attribute listed in the **Target properties** setting of the **Save Object Properties** activity. Otherwise, the entry you configured will return no property value during runtime.

Getting started

For more information on how to configure object property saving in a workflow, see *Configuring a Save Object Properties activity* in the Active Roles Administration Guide.

Workflows to modify requested changes

Change workflows configured in the Active Roles Console support updating change requests that started a workflow with the **Modify Requested Changes** activity. This activity lets you add or remove changes to the properties of the workflow target object while the workflow is running.

For example:

- In a workflow that starts when requesting the creation of an object, you can use the **Modify Requested Changes** activity to either modify the properties that will be assigned to the new object, or change the container in which the object will be created.
- In a workflow that starts when requesting the change an object, you can use the **Modify Requested Changes** activity to modify the requested property changes of the object.

NOTE: The **Modify Requested Changes** activity is not available in automation workflows.

Workflow configuration options

The **Modify Requested Changes** activity has the following configuration options:

- **Target changes**: Specifies the property changes to add or remove from the change request. Use this setting to select:
  - The **Property** (or properties) you want the activity to change.
  - The **Action** to perform for each property (for example, adding, setting or removing the value of the property, or removing the property itself from the request).
  - The **Value** to add, remove or modify.

  You can add, remove and modify values both for single-value and multi-value properties, with the following options.

  NOTE: The various properties may only support some of the following settings.

  - **Fixed object in directory**: Specifies a particular object that you select in Active Directory.
  - **Text string**: Lets you specify the value of the property manually via a string.
  - **Workflow target object**: Specifies the target object of the request in a change workflow that started the workflow.
  - **Property of workflow target object**: Uses the value of a specific property of the target object in the request that started the workflow. When selecting this option, you can select the property from a list of object properties.
  - **Workflow initiator object**: Uses the object that initiated the workflow. When selecting this option, you can select the object from a list.
  - **Property of workflow initiator**: Uses the value of a specific property of the user who initiated the workflow. When selecting this option, you can select the property from a list of object properties.
  - **Object identified by workflow parameter**: Specifies the object via the value of a certain parameter in the workflow. You can select the parameter from the workflow definition.
  - **Object from workflow data context**: When selected, the activity will select the object based on the workflow environment data collected while running the workflow. You can select the object for the activity when the workflow is initiated.
  - **Object identified by DN-value rule expression**: Specifies the object via its Distinguished Name (DN) by the string value of a certain rule expression. By using a rule expression, you can compose a string value based on the properties of various objects found in the workflow environment when running the workflow. You can create the desired rule expression when you configure the activity.

- **Changed value of workflow target object property**: Uses the value that the workflow requests to be assigned to a certain property of the workflow target object. When selecting this option, you can select the property from a list of object properties.
- **Workflow parameter value**: Uses the value of a certain parameter of the workflow. When selecting this option, you can select the property from a list of workflow parameters.
- **Property of object from workflow data context**: Uses the value of a certain object property selected by the activity on the basis of the data found in the workflow run-time environment. You can choose the desired property and specify which object you want the activity to select when the workflow runs.
- **Value generated by rule expression**: Uses the string value of a certain rule expression. By using a rule expression you can compose a string value based on properties of various objects found in the workflow runtime environment. You can create the desired rule expression when you configure the activity.

- **Notification**: Configures notifications for the runs of the activity, and subscribes recipients to the following notification events:
  - **Activity completed successfully**: Sends a notification email if no significant errors occurred during the run of the activity.
  - **Activity encountered an error**: Sends a notification email if significant errors occurred during the run of the activity.

  The notification settings specify the notification events and recipients. When run by the workflow, the activity prepares a notification message according to the specified event. Active Roles retains the message prepared by the activity, and sends the message to the specified recipients when the event occurs.
- **Error handling**: Specifies the action to take when detecting any errors. Selecting **Continue workflow even if this activity encounters an error** will suppress any errors detected by Active Roles during the workflow run. Leaving this setting clear will result in Active Roles stopping the workflow if the activity detects any errors. By default, this setting is not selected.
- **Additional settings**: The **Modify Requested Changes** activity also contains the following settings:
  - **Modify object creation requests so as to create objects in this container**: Allows you to change the container where Active Roles creates the new objects, while ensuring that the policies and workflows will be applied from the container where the object will be created (rather than from the container that was originally specified in the object creation request).
  - **Include or exclude these controls from the change request**: Allows you to add or remove Active Roles controls from the request. "Controls" are pieces of data that provide additional information for Active Roles on how to process the request.

    If you do not specify any controls in the request, Active Roles will process the request based on the type of the request only. You can either configure the activity to add certain controls to the request (include controls) or to ensure

that certain controls never occur in the request (exclude controls). For more information about adding Active Roles controls to a request, see the *Active Roles SDK documentation*.

Getting started

For more information on how to configure object property saving in a workflow, see *Configuring a Modify Requested Changes activity* in the Active Roles Administration Guide.

Workflows for initialization scripts

When running a workflow instance, Active Roles uses a single PowerShell operating environment (called "runspace") for all script activities held in that workflow. The workflow runtime engine creates a runspace once the workflow instance started, and maintains the runspace during the run of the workflow instance.

When you configure a workflow, you can specify PowerShell commands you want the workflow runtime engine to initialize immediately after creating the runspace. These commands are part of an **initialization script** that the workflow engine runs prior to performing the script activities.

With an initialization script, you can define runspace configuration data separately from the logic of other script activities, and you can use it to initialize the environment for initializing script activities. Specifically, you can:

- Load PowerShell modules and snap-ins. All activity scripts can use the modules and snap-ins loaded in the initialization script without having to load the prerequisite modules or snap-ins on a per-activity basis.

  The modules and snap-ins loaded in the initialization script are available to all script activities at workflow runtime. For example, the `Import-Module 'SmbShare'` command added to the initialization script makes the Server Message Block (SMB) Share-specific cmdlets available to all script activities within the workflow.

- Initialize environment-specific variables, referred to as "global variables". All activity scripts can retrieve and update global variables, which makes it possible to exchange data between different activity scripts.

  The global variables are visible to all script activities at workflow runtime. For example, the `$rGuid = [Guid]::NewGuid()` command added to the initialization script makes the `$rGuid` variable available to all script activities within the workflow. To reference a variable defined in the initialization script, the activity script must use the `$global:` qualifier, such as `$global:rGuid`.

  > TIP: If the run of the workflow instance is suspended (for example, because it is waiting for approval), then resumed (for example, after receiving approval), the runspace is reinitialized, so the global variables may change.
  >
  > In such cases, if you need to preserve the value of a global variable, add the `[Persist()]` attribute to the variable name in the initialization script, such as `[Persist()]$rGuid = [Guid]::NewGuid()`. Global variables defined this way are saved to a persistent storage when the workflow instane is suspended, then restored from the storage when the workflow instance is resumed.

To save a variable, Active Roles creates and stores an XML-based representation of the object represented by the variable, similarly to the `Export-Clixml` command in Windows PowerShell. When restoring the variable, Active Roles retrieves the XML data that represents the object, and creates the object based on that data, similarly to the `Import-Clixml` command.

Getting started

You can create new initialization scripts in the Workflow Designer of the Active Roles Console.

To start creating a new initialization script

1. In the Active Roles Console, navigate to **Configuration** > **Policies** > **Workflow**.
2. To open the Workflow Designer, select the workflow you want to configure.
3. In the details pane, click **Workflow options and start conditions** > **Configure**.
4. To open the initialization script editor, click **Initialization script**.

The **Initialization script** tab then displays the currently used script (if it exists). To add a new script or modify the existing one, use the editor.

Workflows to search for expiring users

You can use the **Search** activity in an Active Roles workflow to search directory objects (such as users or groups), that match the criteria you specify with your search terms. Active Roles can then pass the search results to other workflow activities to perform additional actions.

The **Search** activity also supports searching for user accounts that will expire within the specified amount of time.

Getting started

To search for user accounts that expire within a certain amount of days, use the **Search** activity of the Workflow Designer in the Active Roles Console.

To search for expiring user accounts with a workflow

1. In the Active Roles Console, navigate to **Configuration** > **Policies** > **Workflow**.
2. To open the Workflow Designer, select the workflow you want to configure.
3. Add a **Search** activity to the workflow, or right-click an existing one, and select **Properties**.
4. To filter the search to user accounts that will expire, select **Retrieve only expiring user accounts**.
5. In the dialog that opens, specify the number of days to check. The **Search** activity will list user accounts that expire within the specified number of days.

Workflows to send plain-text notification messages

When configuring an Active Roles workflow, you can set email notification messages for the workflow based on a message template. The template specifies the format and contents of the notification message, including its subject and body.

The notification messages are created (and by default, sent) in HTML format. However, when configuring a **Notification** or **Approval** activity, you can also send them in plain-text format. Sending notification messages in plain-text format is useful for integration solutions that use mail flow for data exchange between Active Roles and other solution components in your organization.

Getting started

To configure a plain-text notification message for a **Notification** or **Approval** activity, use the Workflow Designer in the Active Roles Console.

To configure plain-text notification messages for a workflow

1. In the Active Roles Console, navigate to **Configuration** > **Policies** > **Workflow**.
2. To open the Workflow Designer, select the workflow you want to configure.
3. Right-click the **Notification** or **Approval** activity you want the notification for, or add them to the workflow from the Workflow Designer options.
4. In the **Notification Message** page, select **Format notification message as plain text**.

Operation in multi-forest environments

Active Directory organizes network elements into a hierarchical structure based on the concept of containers, with the top-level container being referred to as a forest. Today, many real-world Active Directory implementations consist of several forests. Common reasons for multi-forest deployments are the isolation of the administrative authority, organizational structure issues (for example, autonomous business units and decentralized IT departments), business policy, or legal and regulatory requirements.

This section provides information on the features and benefits of Active Roles as applied to environments where multiple Active Directory forests have been deployed.

With Active Roles, you can create a scalable, secure, and manageable infrastructure that simplifies user and resource management in a multi-forest environment. Benefits of deploying Active Roles in such environments include:

- Centralized management of directory data in domains that belong to different forests.
- Administrative views spanning forest boundaries.
- The ability to delegate administrative control of directory data where appropriate, without regard to forest boundaries.

- Policy-based control and automation of directory data management across forest boundaries.

By registering Active Directory domains with Active Roles, you form a collection of managed domains that represents an Active Roles security and administrative boundary in Active Directory. The collection need not be restricted to domains from a single forest. Rather, you can register domains from any forest in your environment, configuring the Active Roles Administration Service to use the appropriate administrative credentials on a per-domain basis.

To centralize management of directory data across the managed domains, Active Roles retrieves and consolidates the Active Directory schema definitions from all forests to which those domains belong. The consolidated schema description is stored in the Active Roles configuration database, and contains information about the object classes and the attributes of the object classes that can be stored in the managed domains. By using the consolidated schema, Active Roles extends the scope of its administrative operations to cover the entire collection of managed domains regardless of forest boundaries.

Active Roles allows administrators to organize directory objects (such as users, groups, computers, and so on) into a relational structure made up of rule-based administrative views (referred to as Managed Units), each of which includes only the objects that meet certain membership criteria defined by the administrator. This structure can be designed independently from the logical model of Active Directory, which is based on the concept of containers and thus implies rigid boundaries between containers, be it forests, domains or Organizational Units. Administrators can configure Managed Units so that each Unit represents the appropriate collection of directory objects that reside in the same Active Directory container or in different containers, with different forests not being the exception.

To facilitate the management of directory data, Active Roles provides for administrative delegation at the Managed Unit level as well as at the level of individual containers in Active Directory. Through delegation, authority over directory objects held in a given Unit or container can be transferred to certain users or groups. Delegation of control over Managed Units provides the ability to distribute administration of directory data among individuals trusted to perform management of specific groups and types of objects, without taking into account the location of the objects in the Active Directory structure. Thus, Active Roles makes it easy to delegate control of directory data from one forest to users or groups located in the same forest or in a different forest.

Active Roles also allows policy-based control and automation of directory data management to be implemented at the Managed Unit level. By applying policy and automation rules to Managed Units, administrators can ensure consistent control of the well-defined collections of directory objects located in different Organizational Units, domains, or forests. In addition, policy and automation rules can be consistently applied to different containers, whether in the same forest or in different forests, which provides the platform for complex automation scenarios that involve cross-forest operations. An example could be provisioning users from one forest with resources in another forest.

When adding objects to a group, Active Roles allows you to select objects from different managed domains, including those that belong to different forests. This operation requires a trust relationship between the domain that holds the group and the domain that holds the object you want to add to the group. Otherwise, Active Directory denies the operation and, therefore, Active Roles does not allow you to select the object. Note that Active Directory automatically establishes trust relationships between domains within one forest. As for

domains in different forests, administrators must explicitly establish trust relationships as needed.

The rule-based mechanisms that Active Roles provides for auto-populating groups can also be freely used in multi-forest environments. You can configure rules to have Active Roles populate groups with objects that reside in different domains, whether in the same forest or in different forests. However, the capabilities of Active Roles to automatically manage group membership lists are also restricted by the Active Directory constraints that only allow a group to include objects from the domain that holds the group or from the domains trusted by that domain. In other words, unless a trust relationship is established between the domain that holds the group and the domain that holds a given object, the object cannot be added to the group, neither manually nor automatically by Active Roles.

Examples of use

Active Roles can be configured to provide a wide range of directory management solutions, allowing organizations to create more secure, productive, and manageable Active Directory and Microsoft Exchange environments. This section highlights how Active Roles helps to address the challenges faced by enterprises today.

Distributing administration

Suppose a large company wants to introduce distributed administration, but wants to avoid the large costs involved in training their helpdesk and business units to correctly use complex administrative tools. In this situation, there is the need for an easy-to-use tool, to control what actions the helpdesk and business units can perform, and to enforce company policies and procedures.

Solution

Active Roles allows organizations to create Managed Units and to designate Trustees over those Managed Units. Trustees only see the objects to which they have access. They are given only the rights they need for the objects within these Managed Units, down to individual properties. Unlike native Active Directory Organizational Units, Managed Units provide virtual boundaries that span across domains and forests, offering more flexible delegation capabilities.

Delegating limited control over Managed Units efficiently eliminates the need for high-level administrative user ID's, allowing organizations to securely distribute administrative authority to local management. To improve network security and make distributed administration safe, Active Roles defines and enforces customizable administrative polices.

Active Roles allows organizations to safely implement administration for business units. If a company has a number of different business units, each of equal importance and each located in a separate office, a single network administrator could support all of the sites.

Active Roles allows the company to create a single Managed Unit, giving an administrator control over users and resources that span multiple domains.

Integrating with other systems

Suppose a company wants to integrate its HR system, administration, and physical security to provide a workflow that reduces repetitive data. Normally, the HR team creates a user profile, the IT team also creates a user profile in Windows and Exchange, and the security team activates an access card for the new employee. The three teams do not synchronize with each another and instead duplicate their work. This results in increased administration costs and introduces security issues. For example, some individuals may no longer work for the company but may still have valid user IDs and access cards. In this scenario, there is a need to integrate the HR system and other systems of the company, and to automate performing user provisioning tasks.

Solution

With Active Roles, a suitable property set can be established to include data from network data sources other than Active Directory. For instance, a property set might be configured to retrieve the personal information of a user from an HR database. When the user account is created, this data could then be passed to Active Directory and Microsoft Exchange. If these property values change, an update could be made to both Active Directory and to the HR system.

Active Roles also provides the ability to set up administrative policies that reduce the amount of input required to carry out a task. For example, when a user moves to a different location, Active Roles could automatically update the user profile in the HR system, based only on the change to the site code or department of the user in Active Directory. Additionally, when a user joins or leaves the company, their access card could automatically be enabled or disabled.

Managing multi-forest Active Directory design

Suppose a host company has client customers who need to place domain controllers on their premises. In Active Directory, every domain controller holds a writable copy of the schema and configuration of the entire forest. Anyone with administrative or backup/restore rights on any domain controller, or physical access to any domain controller, could potentially disrupt the entire forest. For instance, they could attempt to circumvent Windows security, or they could edit the Active Directory database, and the changes would be propagated to all domains in the forest. To avoid such an incident, the company needs to create a separate forest for each client who requires domain controllers on their premises. Otherwise, the actions of one malicious user could affect directory service delivery for other clients in the same forest.

Having multiple forests increases the complexity of the Active Directory structure. This in turn leads to increased administration, as each forest needs separate directory service administration. In this case, there is a need for an administrative system that enables the cross-forest management of Active Directory.

Solution

Active Roles provides a unified management structure that can extend across multiple Active Directory forests. The Active Roles user interface provides a single interface for the management of Active Directory domains that belong to different forests. It offers administrative views (Managed Units) that can hold objects from multiple forests, thereby enabling the unified application of corporate rules and roles across forest boundaries.

With its ability to safely delegate administration in multi-forest environments, Active Roles provides the necessary level of control for the host company's customers, while enabling the company to implement role-based security, and restrict the customers' administrative actions based on corporate policies.

For security reasons, it may be unacceptable to have an administrative tool with the same level of rights as a domain administrator. This is because administrative access to an entire domain in a forest may be used to gain administrative access to the whole forest, via the elevation of privileges attack. Active Roles can operate in a multi-forest environment within a precisely defined scope of access to domains, with no special requirement to have administrative access to entire domains or security-sensitive containers. This addresses the need for a product that provides advanced administrative capabilities, while effectively preventing the elevation of privileges.

Simplifying Active Directory structure

Suppose a company wants to design an Active Directory structure based on physical location. As a rule, the administration/IT department, business units, and Exchange team would each prefer to have a different structure. As a result, they agree to a compromise that does not fully satisfy their requirements. Clearly, there is a need to simplify the Active Directory structural requirements.

Solution

In Active Roles, Managed Units allow organizations to achieve acceptable security boundaries without setting up extra domains or Organizational Units. This significantly simplifies the Active Directory structure and reduces security risks.

By using Managed Units for delegation purposes, Active Roles creates a rule-based overlay of Active Directory for administration. This simplifies the process of choosing an Active Directory structure. Different administrative tasks often require different OU structures. For instance, an OU structure designed purely for the delegation of administration differs from an OU structure shaped purely for Group Policy. It becomes much easier to design an Active Directory structure by using Managed Units to handle delegation issues.

Handling organizational changes

Consider a company in the process of reorganization. Multiple departments are changing names, merging, or separating from one another. Such reorganization involves an increase in administrative, security, and business liabilities, as well as the high cost of manually updating data. This situation demands a means to automatically update and move data.

Solution

Active Roles provides the ability to define administrative policies that make organizational changes easier to handle. By using Managed Units, rule-based overlays of the actual data in Active Directory can be set up for both the current and planned organizational structures. Administrative policies can be specified so that when data moves from one Managed Unit to another, policy definitions will automatically be applied, based on the change. This will update user properties, such as the manager, department, group memberships and OU memberships.

As another example, consider a user who changes departments. Depending on the department to which the user moves, Active Roles could automatically move user data, change user group memberships, and specify the manager of the user.

User account management

Suppose a company provides services based on Active Directory and Microsoft Exchange. The company relies on the Active Directory infrastructure as a basis for their service offerings.

Configuration of Active Directory involves setting security and partitioning the directory, so that any user has proper access to directory resources. It is paramount to have a framework that facilitates the creation of new user accounts and the assignment of appropriate access rights. There is a need for a robust system that maintains user creation and management with minimal administrative effort.

Solution

Active Roles offers a reliable solution to simplify and safely distribute user account management. It addresses the need to create and manage a large number of user accounts, and to ensure that each user can only access their own resources. By implementing an administrative model based on business rules, Active Roles allows domain-level administrators to easily establish and maintain very tight security, while facilitating the provisioning of new users with the appropriate access to IT resources.

Active Roles has the ability to safely delegate routine user-management tasks to designated persons. By incorporating policy enforcement and role-based security, Active Roles allows the organization to restrict the administrative actions according to the corporate policies defined by the high-level administrators. In addition, it allows the

administrators to change the policies, ensuring that new policy settings are automatically propagated and enforced without additional development.

Active Roles makes it simpler for the organization to delegate authority to administrative and support groups, while enhancing the overall security. The Web Interface can serve as an administrative tool that allows the assistant administrators to manage users, groups, and mailboxes. Active Roles ensures that all actions performed by a Web Interface user are in compliance with the corporate security policies.

Administrative rules and roles

The following sections provide an overview of the Active Roles features related to:

- Workflow capabilities.
- Policies (that is, administrative rules).
- Delegation model (that is, administrative roles).

About Managed Units

Enterprises usually design their Organizational Unit-based (OU-based) network structure on geographical or departmental boundaries, restricting the ability to delegate administration outside these boundaries. However, they may face situations that require directory objects to be grouped together by logic that does not align with the OU structure.

Both Active Directory (AD) and Azure Active Directory (Azure AD) offer a comprehensive delegation model. However, since the scope of delegation is defined using OUs, distributed administration is constrained by the OU structure. For example, in AD, without changing the directory structure, you cannot regroup objects so that new groups support inheritance for their members when delegating control or enforcing policy.

To solve this problem and provide more flexibility in managing AD and Azure AD resources, Active Roles provides special administrative views that can meet any directory management needs. These securable, flexible, rule-based administrative views, known as Managed Units or MUs, allow configuring distributed administration independent of the OU hierarchy. As such, MUs are dynamic virtual collections of AD or Azure AD directory objects, and may include them regardless of their location in the organization network.

While Managed Units allow organizations to implement OU structures on a geographical basis, it distributes administration on a functional basis. This means that, for example, all users or Azure users of a particular department could be grouped into a single MU to delegate access control and enforce administrative policies regardless of their location in different OUs. However, grouping said users or Azure users into an MU still keeps the geographically-defined OUs of the users intact, leaving the OU-based structure unaffected.

As such, MUs make it possible to organize an enterprise by any custom logic without changing the underlying domain and OU structure, resulting in a secure and easy-to-manage administration environment.

MUs can include:

- AD objects from different domains, trees, or forests, provided that they are configured in Active Roles.
- Azure users, Azure guest users, Azure contacts, Microsoft 365 (M365) groups, Azure distribution groups and Azure security groups from any Azure tenant configured in Active Roles.

  NOTE: MUs do not support any Azure mailbox types and dynamic distribution groups.

MU membership is not exclusive: You can include AD or Azure AD objects in an MU even if they are already members of another MU.

How Managed Units work

Managed Units (MUs) use membership rules to determine whether an object is a member of a specific MU. For example, you can specify an MU membership rule which states that all users or Azure users who are geographically located in the United States can belong to a certain MU. The configured membership rule then works as a query in Active Roles, searching for users or Azure users located in the United States, and populating the MU accordingly. Active Roles stores the MU membership rules as part of the MU properties, ensuring that:

- Whenever a new directory object that meets the membership requirements is created, it is added to the MU.
- Whenever an existing member object changes in a way that it does not meet the membership requirements, it is removed from the MU.

Active Roles allows configuring MU-level permissions and policy settings, with their inheritance also working seamlessly across the Active Directory (AD) and Azure Active Directory (Azure AD) environment. Similarly to the MU membership of directory objects, these object permissions and policy settings can also change as well, so that the MU can dynamically adapt to the changing organization, simplifying administration maintenance.

Once configured, MUs are ideal for delegated administration for several reasons:

- When using MUs, delegated administrators no longer have to browse OUs to search for managed objects.
- You can delegate the administrative control of MUs to specific users or groups, similarly to OUs.
- With MUs, you can locate all objects managed by the same user or group in one place.

Deployment considerations for Managed Units

Managed Units (MUs) are virtual Organizational Units (OUs), allowing you to group directory objects and configure delegation and policy settings for them by a logic that may not correspond to your Active Directory (AD) or Azure Active Directory (Azure AD) structure. However, technically, they are LDAP queries stored in the Active Roles Configuration Database. As such, MUs can only be configured and accessed via Active Roles interfaces.

Managed Unit membership rules

It is membership rules that determine the list of objects to be included in a Managed Unit. Although several types of membership rule are available, there are two that are most commonly used. These are query-based inclusion or exclusion rules and explicit inclusion or exclusion rules.

Typically you would use query-based rules to include objects that span multiple Organizational Units or Organizational Unit structures. An example is a Managed Unit that includes all disabled user accounts or a Managed Unit that includes all user accounts without mailboxes. Query-based rules are also used to build logical structures from a flat Organizational Unit structure.

You have to be careful with query-based rules because in essence these are conditions imposed on object attributes. If the value of an object's attribute does not meet the specified conditions, the object is not included in the Managed Unit. The opposite is also true. If you, for example, configure a Managed Unit to include all users whose name begins with letter A, the Managed Unit would include the Administrator account. If the Helpdesk had delegated control over that Managed Unit, Helpdesk operators could gain control over the Administrator account. This brings in the use for explicit rules.

Explicit rules allow you to include or exclude objects based upon their identifier (GUID), so no matter how the object is changed or renamed, as long as that object exists in the directory, the rule will be in effect. Explicit rules normally complement query-based rules to include an object the query does not cover or to exclude an object that may meet the conditions of the query. Other uses are to statically include an object so no matter what that object is named it will always be included. Most typically this is Organizational Units. You can build a logical structure of Organizational Units from any part of the directory tree by explicitly adding them to a Managed Unit. This makes delegation and policy application much easier, since either can be done at the Managed Unit level instead of each individual Organizational Unit.

The following table lists some useful examples of membership rules. These examples demonstrate how to control membership rules by using LDAP filters. You can apply an LDAP filter under the **Custom Search** option in either of the query-based rule types.

Table 1: Managed Unit membership rules

Managed Unit Contents	LDAP Filter
Groups hidden from the Exchange Address List	`(&(objectCategory=group)(mailnickName=*) (msExchHideFromAddressLists=True))`
Mail-enabled groups	`(&(objectCategory=group)(mailNickName=*))`
Mail-enabled users with forwarders set	`(&(sAMAccountType=805306368)(mailNickName=*) (altRecipient=*))`
Users who do not have an Exchange mailbox	`(&(sAMAccountType=805306368)(!(homeMDB=*)))`
Distribution groups	`(&(objectCategory=group)(! (groupType:1.2.840.113556.1.4.803:=2147483648)))`
Security groups	`(&(objectCategory=group) (groupType:1.2.840.113556.1.4.803:=2147483648))`
Disabled user accounts	`(&(objectCategory=user) (userAccountControl:1.2.840.113556.1.4.803:=2))`
Users from the Sales department	`(&(objectCategory=user)(department=Sales))`

Delegation of Managed Units

You can delegate Managed Units exactly like Organizational Units or an entire domain, by applying Access Templates in Active Roles. This can drastically expedite deployment, ease administrative burden for Active Roles, and simplify the training and job processes for the administrators using this tool.

For example, by grouping all disabled and locked out accounts within a single Managed Unit, you can delegate control to a Helpdesk group so that they can quickly and easily perform a large part of their job function by only having to enumerate and look through a single structure. Also, when delegating control of a Managed Unit, you do not have to give your delegated administrators access to any Organizational Unit: all objects that meet the membership rules are in the Managed Unit regardless of what Organizational Units hold those objects.

One more example would be where Active Directory has a very flat structure of Organizational Units, however different administrators are responsible for different locations. As long as the location code is stored in an attribute of the objects to be managed, you can create Managed Units based on that attribute, and delegate to each set of administrators a Managed Unit containing their respective objects that meet a particular location code. Since Managed Units are merely groupings of objects based on certain attributes, the objects will move in and out of Managed Units regardless of how their attributes change, either through Active Roles interfaces or natively.

An important feature of Managed Units is the fact that a single Managed Unit can include objects from any domains managed by Active Roles (managed domains). As long as a given domain is registered with Active Roles, regardless of the domain's forest, any object from that domain can be added to a Managed Unit. When doing delegation of a Managed Unit that holds objects from different domains, it is recommended to use domain local groups from the domain where the Active Roles installation exists, or universal groups. This is because Active Roles allows you to do delegation to any security group within the managed domains; however, if the Active Roles installation exists in domain A and a delegation was done to a domain local group in domain B, an administrator who authenticates against Active Roles in domain A will never have the local group from domain B added to his security token, therefore he will not have his delegated rights. Also global groups can be used as long as all administrative users reside in the domain where Active Roles is installed.

One precaution you must consider with delegating control of Managed Units is the ability to synchronize the delegated permissions to Active Directory. When you apply an Access Template to a Managed Unit and do not sync the permissions with Active Directory, the permission settings are only stored within Active Roles' configuration database. Active Roles maintains the parent-child relationship for the objects held in the Managed Unit, thus allowing permission inheritance to work. If you choose to sync the permissions with Active Directory, there is no way to maintain that parent-child relationship in Active Directory since a Managed Unit is not truly an object within Active Directory so to Active Directory that parent does not exist. As a result, every permission entry found in the Access Template will be included into the native Access Control List of every object held in the Managed Unit. Potentially this could cause performance issues.

About Access Templates

Active Roles provides safe, distributed administration through advanced delegation of rights with high granularity to individual users or groups. This relieves highly skilled administrators from routine day-to-day tasks, saving time and increasing productivity. For example, an administrator can allow helpdesk to perform specific tasks, such as resetting passwords or managing group memberships, without granting full administrative privileges.

As you develop your administration and security design, you define delegated administrators (Trustees) and administrative roles (Access Templates). Then, you define Managed Units and apply Access Templates, designating Trustees for each Managed Unit. You can also apply Access Templates to objects and folders in Active Directory, assigning the permissions to the necessary Trustees. This three-way relationship between Trustees, Access Templates, and managed objects is central to the implementation of your role-based administration model.

The Active Directory Users and Computers tool provides the facility to delegate administrative responsibilities. However, every time you want to delegate rights, you need to define a set of permissions. This makes the delegation procedure time-consuming and prone to errors. Active Roles overcomes this problem by consolidating permissions into customizable administrative roles, known as Access Templates. The logical grouping of permissions simplifies the management of delegation settings.

Access Templates are collections of permissions representing administrative roles. Permissions are used to allow or deny certain administrative operations to a user or group. You can create an Access Template that incorporates all permissions required to perform a particular administrative role.

To assign the role to a user or group, link the the Access Template to a Managed Unit, Organizational Unit, domain, or individual object, depending on the scope of the role, and then select a user or group to designate as a Trustee. As a result, the individual user, or each member of the group, acquires the rights specified by the role to administer objects that reside in the collection or folder to which the Access Template has been linked.

How Access Templates work

Active Roles implements delegated administration by linking Access Templates to collections of objects (Managed Units), directory folders (containers), or individual (leaf) objects.

When applied to a directory object, an Access Template specifies permission settings for that object and its child objects. Applying Access Templates to Managed Units is a convenient way to manage permissions on collections of directory objects.

Each Access Template is applied in relation to some users and/or groups (Trustees), and the permissions specified in the Access Template determine their access to managed objects. When an Access Template is modified or no longer applied, permissions set for the directory objects are modified accordingly.

When permissions on a Managed Unit change, Active Roles recalculates the permission settings on all the Managed Unit members. Likewise, the permission information is modified whenever the list of objects in a Managed Unit changes. When objects join or leave a Managed Unit (due to object property changes, for example), all permission settings on those objects are recalculated.

Every object inherits its permission settings from the Managed Units in which it resides. For example, if a Trustee has permissions to access multiple Managed Units that hold a given object, the permissions of the Trustee to access that object are simply defined as a union of all permissions specified at the Managed Unit level.

Applying Access Templates to a container object (directory folder) establishes the access of the Trustee to both the container and its child objects. The Trustee, having permissions specified over a container, possesses inherited permissions for the child objects residing in the container.

Permission management with Access Templates

You can assign permissions to Active Directory (AD) objects with Access Templates (ATs) in the Active RolesConsole.

Delegating permissions with ATs is an effective method to grant specific types of access for specific users or groups to specific organizational resources. For example, directory administrators of a domain can receive full control for managing that domain, while helpdesk operators can quickly receive permission to reset passwords for domain users.

Active Roles supports specifying ATs to all AD object types: administrative views (Managed Units), directory folders (containers), or individual (leaf) objects as well. When applying an AT to an AD object, you:

1. Designate a **trustee** (also known as **security principal**) who will receive the permissions granted by the AT. Trustees are typically users or groups.
2. Assign permissions to that trustee for the AD object in the scope of the AT. Such AD objects are called **securable objects**.

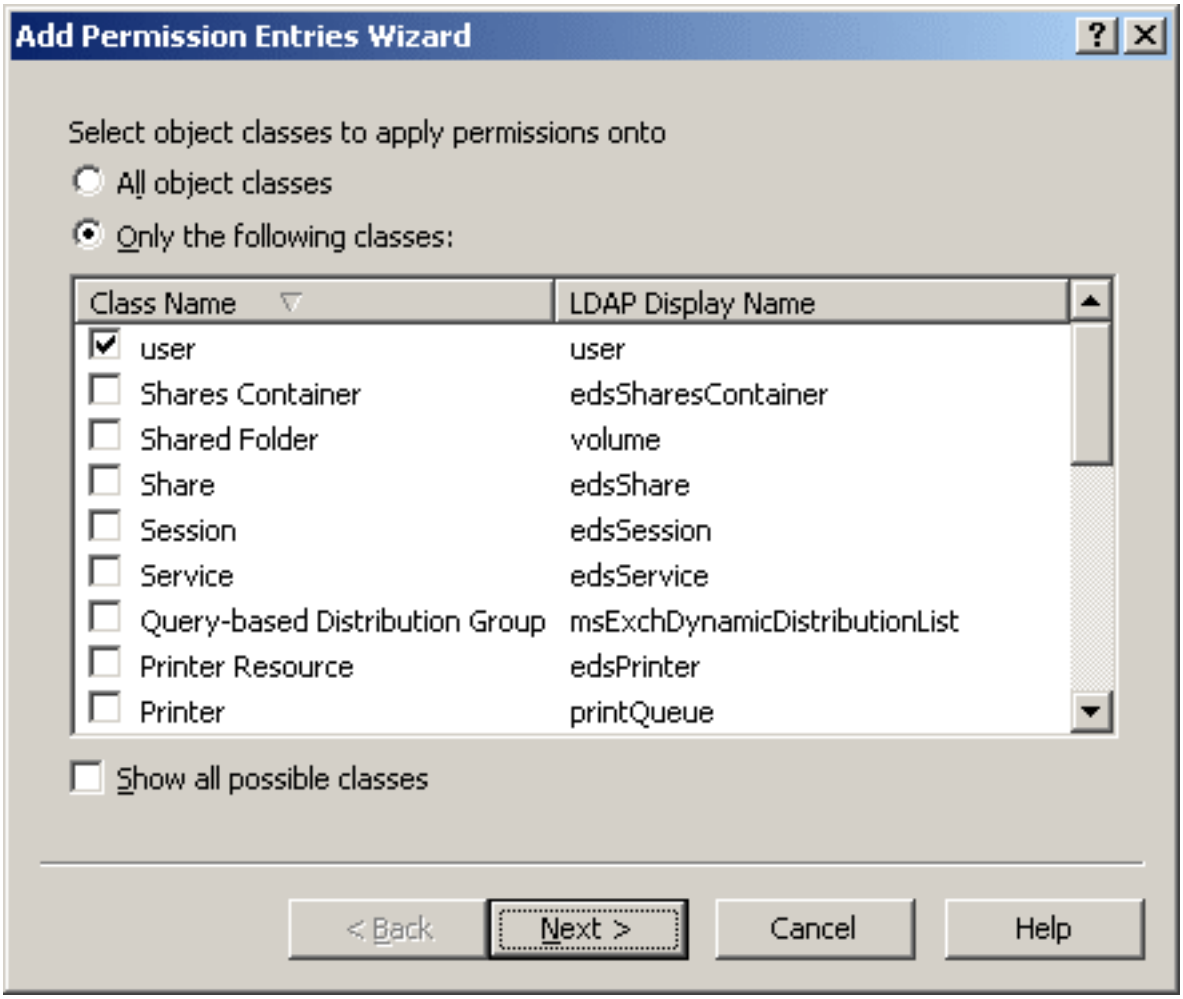
   As a result, the trustee receives access to the securable object according to the permissions defined in the AT.

For the steps of applying Access Templates to directory objects, see *Applying Access Templates* in the *Active Roles Administration Guide*.

About the Add Permission Entries wizard

The **Add Permission Entries Wizard** lets you specify the permission to be added into the Access Template. The first page of the wizard looks as shown in the following figure.

Figure 4: Add Permission Entries



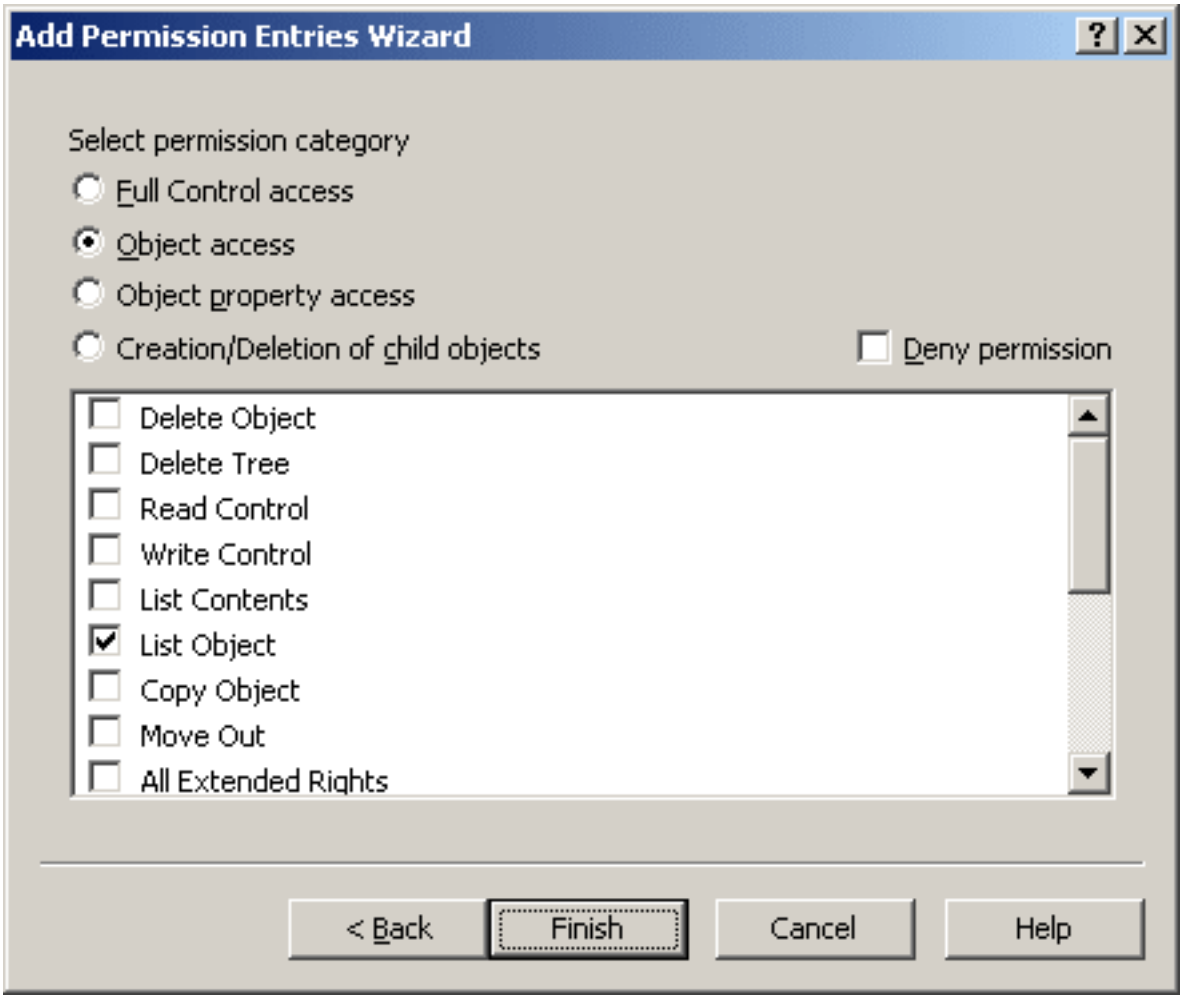
On this page, you select the types of objects to which you want the permission to allow (or deny) access. You can select one of these options:

- **All object classes**: With this option, the permission controls access to objects of any type.
- **Only the following classes**: With this option, the permission controls access to objects of the type you choose by selecting the appropriate check boxes in the list.

NOTE: By default, all object classes are not displayed in the list. To display all object classes, select the **Show all possible classes** check box.

After you have selected the object classes you want, click **Next**. The next page of the wizard looks as shown in the following figure.

Figure 5: Permission category



On this page, you select a permission category, and specify whether you want the permission to allow or deny certain administrative actions.

You can select one of the following permission categories:

- **Full Control access**: Allows or denies all administrative actions on an object.
- **Object access**: Controls how an object is accessed and controlled.
- **Object property access**: Controls access to an object's attributes.
- **Creation/Deletion of child objects**: Allows or denies creation or deletion of objects in a container.

If you want the permission to deny certain administrative actions, select the **Deny permission** check box.

The following sections elaborate on the permission categories you can select in the **Add Permission Entries Wizard**.

Full Control access

Permissions in this category grant access to all object (and object property) administrative operations for the classes selected in the previous step of the **Add Permission Entries Wizard**.

After you select **Full Control access** and click **Finish**, the permission is added to the newly-created Access Template.

Object access

Permissions in this category grant access to object (but not object property) administrative operations for the classes selected in the previous step of the **Add Permission Entries Wizard**.

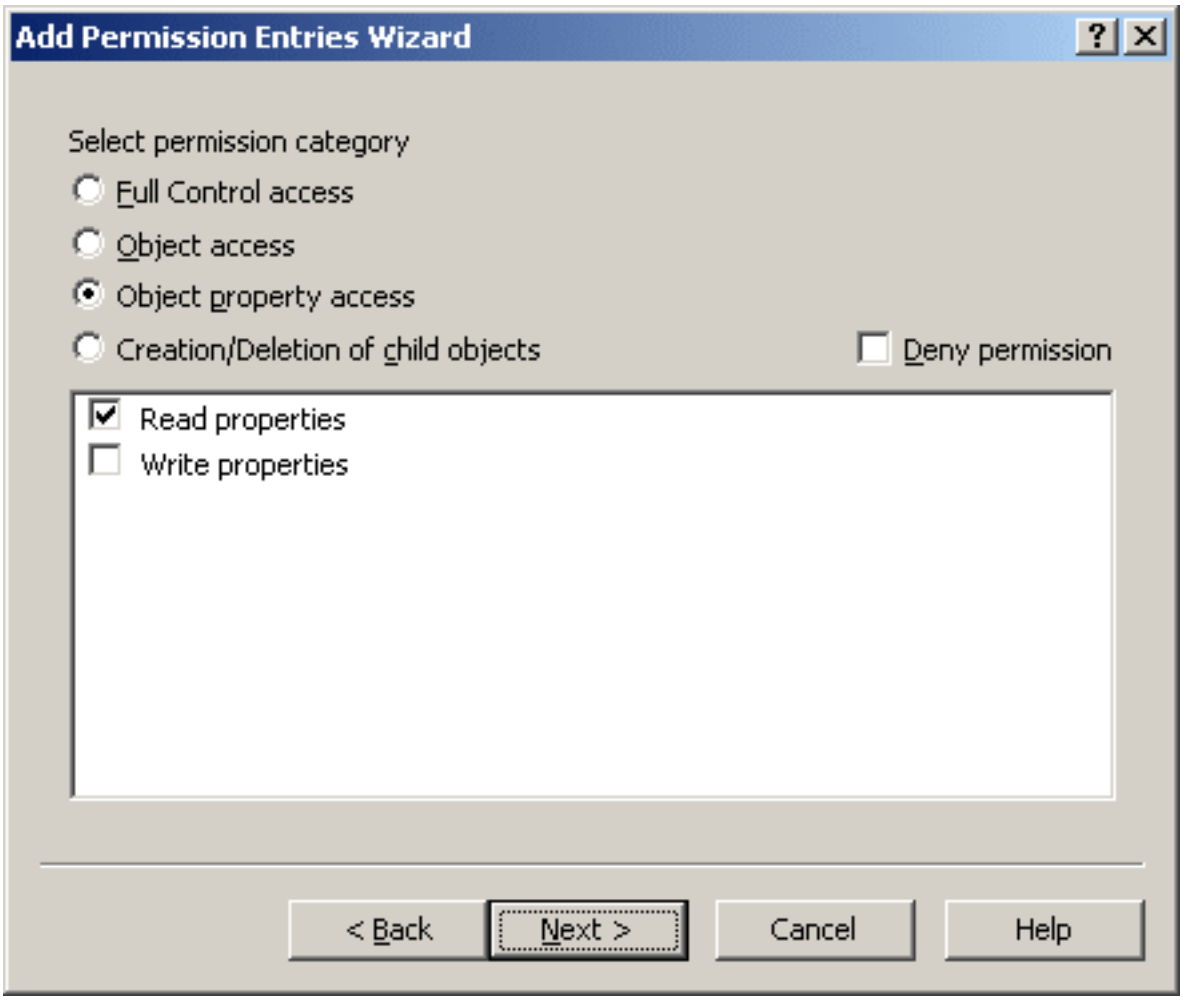
Administrative operations are selected from the list, which is displayed when you select **Object access**. You select the necessary operations by selecting the appropriate check boxes. For example, you might select **List Object** to allow viewing objects of certain types.

After you have selected the operations, click **Finish** to complete the **Add Permission Entries Wizard**. The permission is added to the newly-created Access Template.

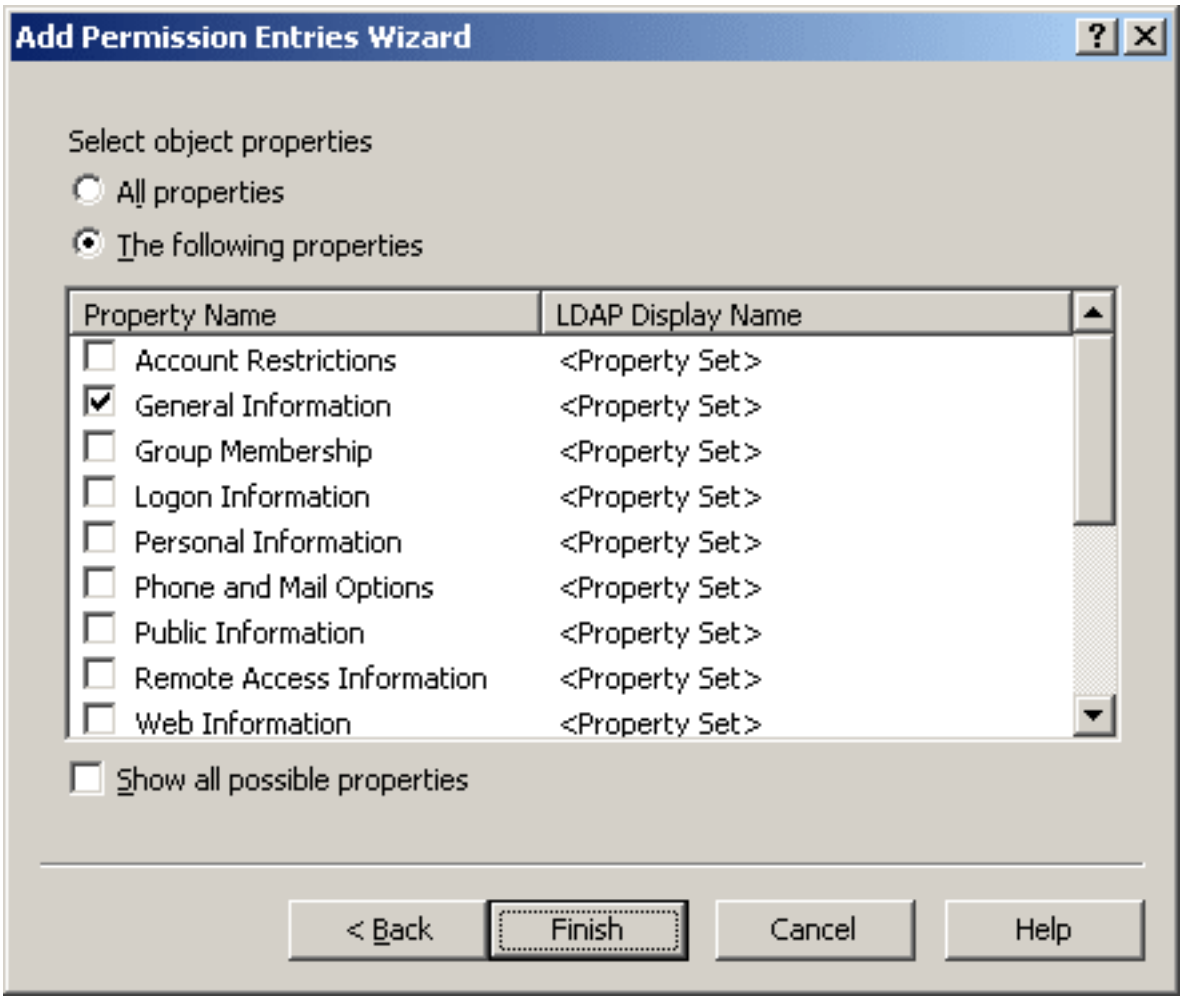
Object property access

Permissions in this category grant access to object property administrative operations for the classes selected in the **Add Permission Entries Wizard**.

When you select **Object property access**, you specify access to object properties. You can select **Read properties** and **Write properties**, as shown in the following figure.



After you click **Next**, the wizard displays a page where you can select the properties to which you want the permission to allow (or deny) access. The page is similar to the following figure.



On that page, you can select one of the following options:

- **All properties**: With this option, the permission controls access to all properties.
- **The following properties**: With this option, the permission controls access to the properties you select from the list by selecting the appropriate check boxes.

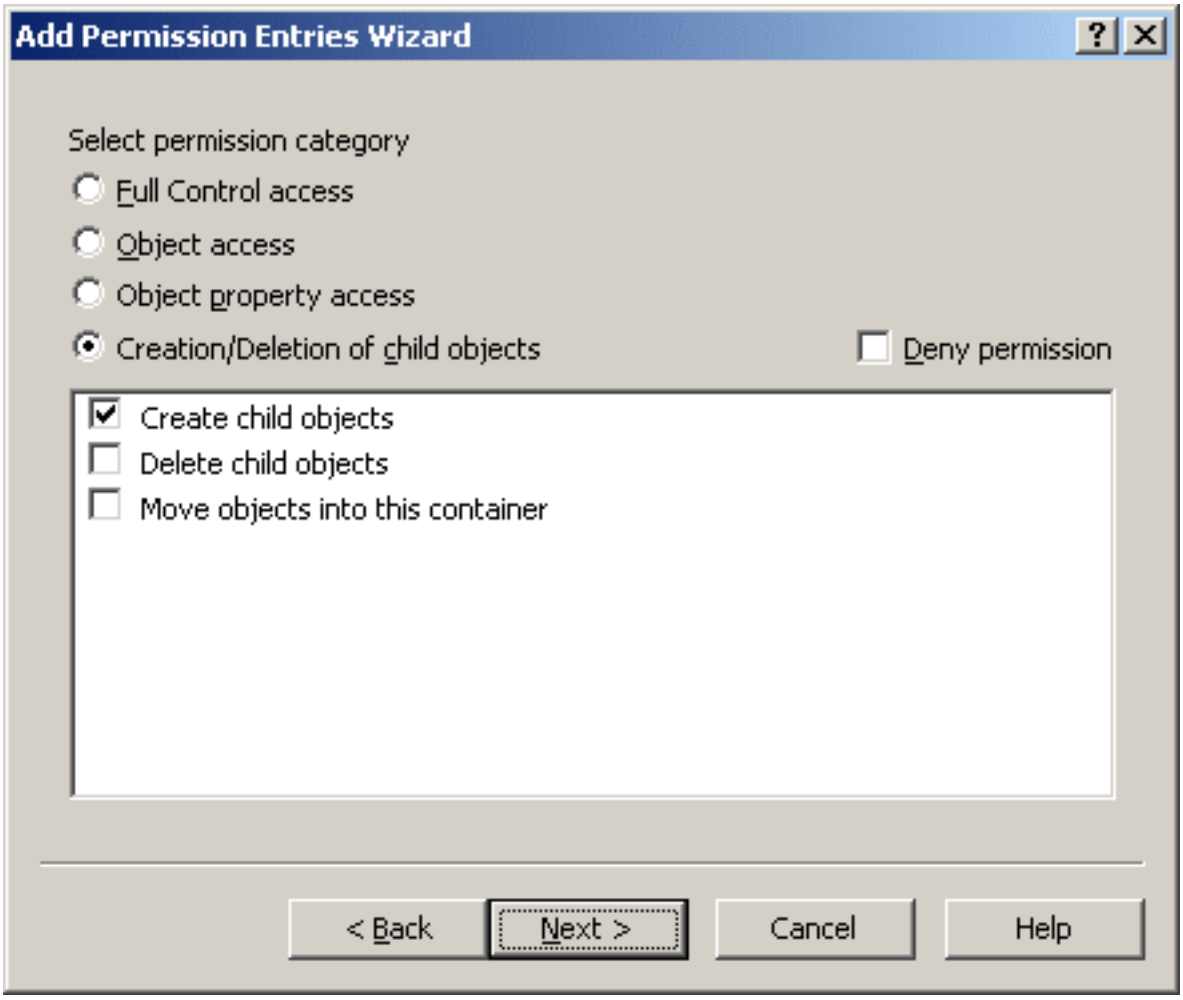
NOTE: By default, not all object properties appear in the list. To display all object properties, select the **Show all possible properties** check box.

After you have selected the properties you want, click **Finish** to complete the **Add Permission Entries Wizard**. The permission is added to the Access Template.

Creating or deleting child object permissions

Permissions in this category provide for creation and deletion of child objects in container objects of the classes you selected in the previous step of the **Add Permission Entries Wizard**.

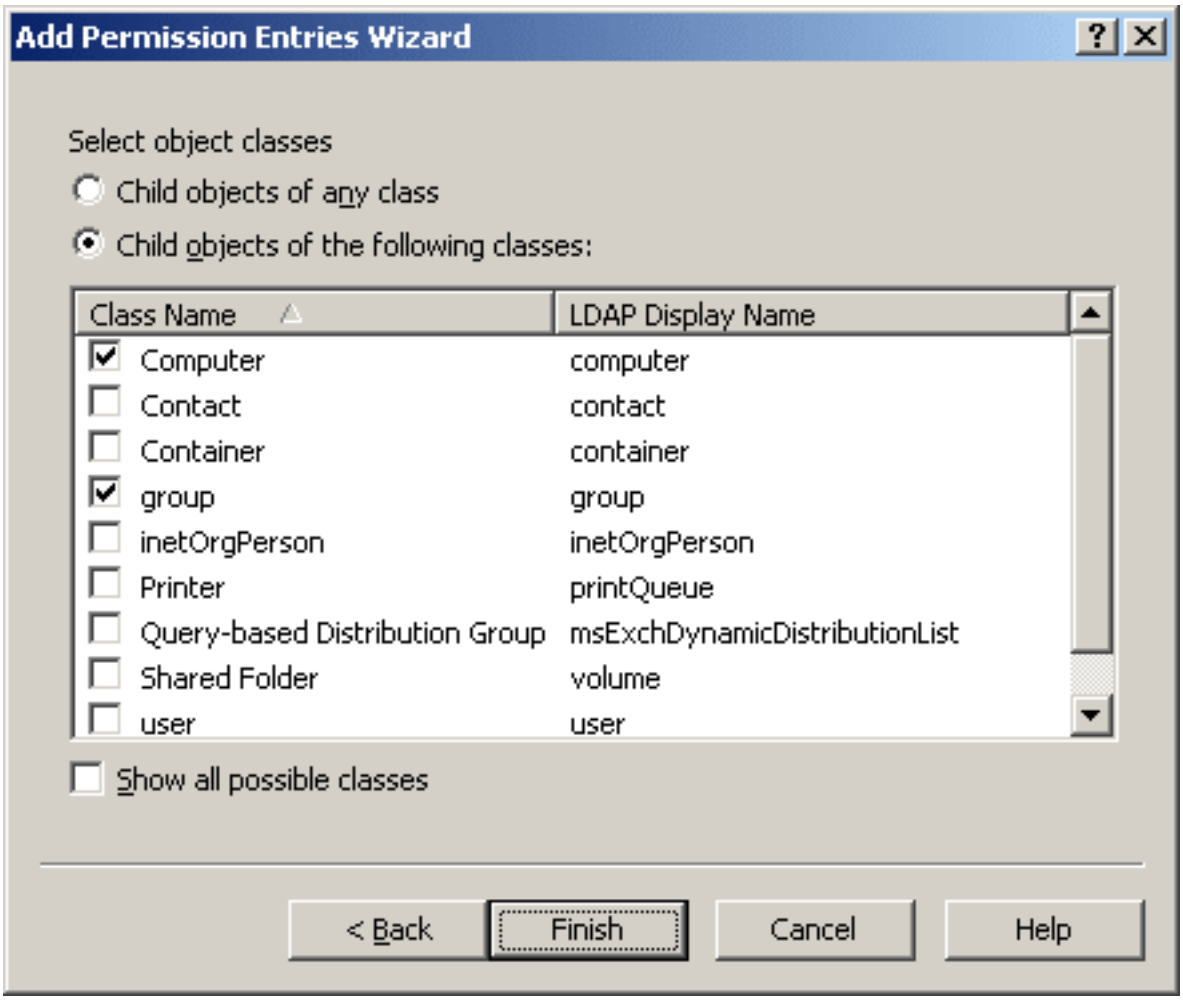
When you select **Creation/Deletion of child objects**, you specify the creation, deletion, and move operations you want the permission to allow (or deny). The list of operations looks as shown in the following figure.



You can select the following operations:

- **Create child objects**: Controls the creation of child objects of the classes you select in the next step.
- **Delete child objects**: Controls the deletion of child objects of the classes you select in the next step.
- **Move objects into this container**: Controls the relocation of object of the classes you select in the next step. This operation assumes moving objects from one container to another without permission to delete existing objects or create new objects.

After you click **Next**, the wizard displays the page where you can select the types of objects on which you want the permission to allow (or deny) the operations you selected in the previous step. The page is similar to the following figure.



On that page, you select the types of objects for which you want the permission to allow (or deny) the creation, deletion, or move operation. You can select one of these options:

- **Child objects of any class**: With this option, the permission controls the operations on objects of any type.
- **Child objects of the following classes**: With this option, the permission controls the operations on objects of the type you select from the list by selecting the appropriate check boxes.

NOTE: By default, all object classes are not displayed in the list. To display all object classes, select the **Show all possible classes** check box.

After you selected the object classes, click **Finish** to complete the **Add Permission Entries Wizard**. The permission is added to the Access Template.

Management of Access Template links

When applying an Access Template, Active Roles creates an Access Template link. Thus, administrative rights are specified by linking Access Templates to securable objects, such as Managed Units, directory folders (containers), or individual (leaf) objects.

Each Access Template link includes the identifier (SID) of the security principal—user or group—to which the specified administrative rights are assigned. When an Access Template link is created, the user or group becomes a Trustee over the collection of objects or the folder to which the Access Template is linked, with permissions specified by that Access Template.

When an Access Template is modified or no longer applied, the permission information on objects affected by the Access Template changes accordingly.

You can display a list of Access Template links starting from one of the following points:

- **Access Template**: Right-click an Access Template and click **Links**.

  This displays the links in which the Access Template occurs.
- **Security principal (Trustee)**: Right-click a group or user, and click **Delegated Rights**.

  This displays the links in which the group or user occurs as a Trustee either directly or due to group memberships.
- **Securable object**: Right-click a container object or Managed Unit and click **Delegate Control**. For a leaf object, open the **Properties** dialog, go to the **Administration** tab, and click **Security**.

  This displays the links in which the selected object occurs as a securable object (referred to as **Directory Object**).

Another way to see a list of Access Template links is to use the **Advanced Details Pane**. Select the setting in in the **View** menu, then select one of the following:

- Access Template

  The **Links** tab lists the links in which the selected Access Template occurs.
- Other object (Managed Unit, container, or leaf object).

  The **Active Roles Security** tab lists the links in which the selected object occurs as a securable object (referred to as **Directory Object**).

The Active Roles Console displays a list of Access Template links in a separate window. Thus, the **Active Roles Security** window is displayed when you start from a securable object (for example, by clicking a Managed Unit or Organizational Unit and then clicking **Delegate Control**).

Each entry in the list of the Access Template links includes the following information:

- **Trustee**: The link defines administrative rights of this security principal (group or user).
- **Access Template**: The Access Template that determines the rights of the Trustee.

- **Directory Object**: The link defines the rights of the Trustee to this securable object.
- **Sync to Native Security**: Indicates whether the permissions are synced to Active Directory.
- **Disabled**: Indicates whether the link is disabled. If a link is disabled, the permissions defined by that link have no effect.
- **Access Rule**: Indicates whether an Access Rule is applied to this link. For more information, see Management of Windows claims.

The **Active Roles Security** window (as well as the **Active Roles Security** tab in the advanced details pane) lists the links of these categories:

- **Direct links**: The Access Template is applied (linked) directly to the securable object you have selected.
- **Inherited links**: The Access Template is applied (linked) to a container in the hierarchy of containers above the securable object you have selected, or to a Managed Unit to which the securable object belongs.

The links inherited from parent objects can be filtered out of the list:

- When using the **Active Roles Security** window, clear the **Show inherited** check box.
- When using the **Active Roles Security** tab, right-click the list and then click **Show Inherited** to deselect the menu item.

A window or tab that displays Access Template links allows you to manage links. In a window, you can use buttons beneath the list. In a tab, you can right-click a list entry or a blank area, and then use commands on the shortcut menu. For example, the following buttons appear in the **Active Roles Security** window:

- **Add**: Starts the **Delegation of Control Wizard** to create apply Access Templates.
- **Remove**: Deletes the selected entries from the list of links. Available for direct links only.
- **View/Edit**: Displays the dialog to view or modify link properties such as permissions inheritance and propagation options.
- **Sync to AD**: Toggles the permissions propagation option of the links selected in the list.
- **Disable**: Disables or enables the link. If a link is disabled, the permissions specified by the link takes no effect.

TIP: In the **Active Roles Security** dialog box, the **Remove** button is available on direct links only. When you need to delete links, it is advisable to manage them using the **Links** command on the Access Template.

Synchronization of Access Template permissions to Active Directory

Permissions defined in an Access Template can be propagated to Active Directory, with all changes made to them in Active Roles being automatically synchronized to Active Directory.

By enabling synchronization from Active Roles security to Active Directory native security, Active Roles provides the facility to specify Active Directory security settings with Access Templates. Access Templates simplify and enhance the management of permissions in Active Directory, enable the logical grouping of permissions, and providing an efficient mechanism for setting and maintaining access control.

For each permission entry defined in Active Roles and configured with the **Permissions Propagation** option set, Active Roles generates native Active Directory permission entries based on the Active Roles permission entry.

The **Permissions Propagation** option (also referred to as **Sync to Native Security** or **Sync to AD** in the user interface) ensures that every time Active Roles permissions change, the associated native permission entries change accordingly.

Disabling the **Permissions Propagation** option on existing Active Roles permissions, or deleting Active Roles permissions with this option set, deletes all native permission entries specified through those Active Roles permissions.

If a propagated permission entry is deleted or modified in Active Directory, whether intentionally or by mistake, Active Roles restores that entry based on Access Template information, thus ensuring the correct permission settings in Active Directory. The **Sync of Permissions to Active Directory** scheduled task is used in Active Roles to create or update permission entries in Active Directory based on the Access Template links that have the **Permissions Propagation** option enabled.

Management of Active Directory permission entries

The **Native Security** tab in the advanced details pane lists the native Active Directory permission entries for the securable object (for example, an Organizational Unit) selected in the Console tree.

By analyzing information in the **Type** and **Source** columns on the **Native Security** tab, you can determine whether a given entry is synchronized from Active Roles.

In the **Type** column, the synchronized entries are marked with the  icon. This icon changes to  if synchronization of the entry is invalid or unfinished. For example, if you delete a synchronized entry from Active Directory, Active Roles detects the deletion and re-creates the entry. Until the entry is re-created, the **Type** column marks the entry with the  icon.

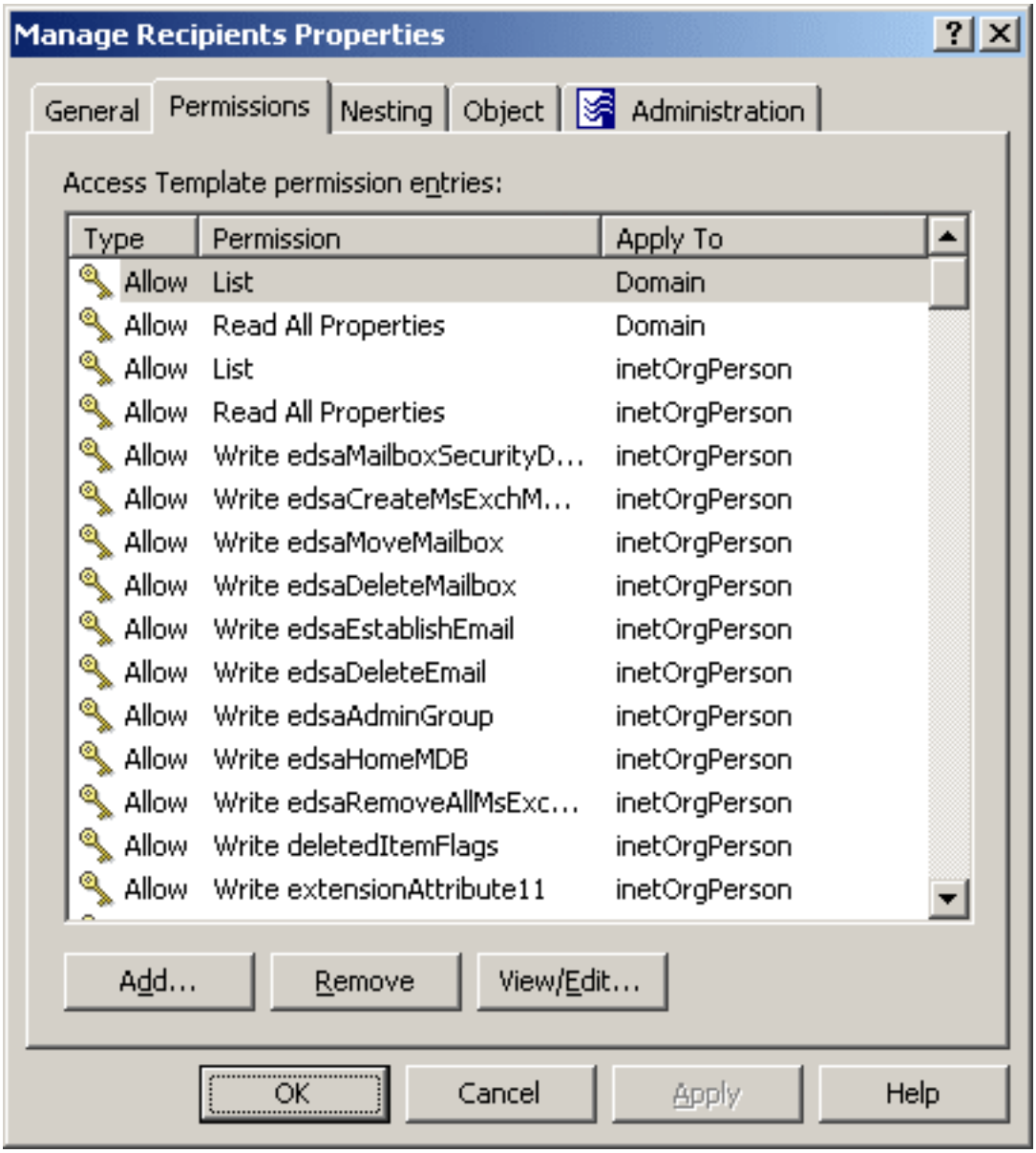
For each synchronized entry, the **Source** column displays the name of the Access Template that defines the permissions synchronized to that entry.

From the **Native Security** tab, you can manage permission entries: right-click an entry, and click **Edit Native Security**. This displays the **Permissions** dialog where you can add, remove and modify Active Directory permission entries for the securable object you selected.

Adding, modifying, or removing permissions to Access Templates

When you add, remove, or modify permissions in an Access Template, permission settings automatically change on all objects to which the Access Template is applied (linked), including those that are affected by the Access Template because of inheritance.

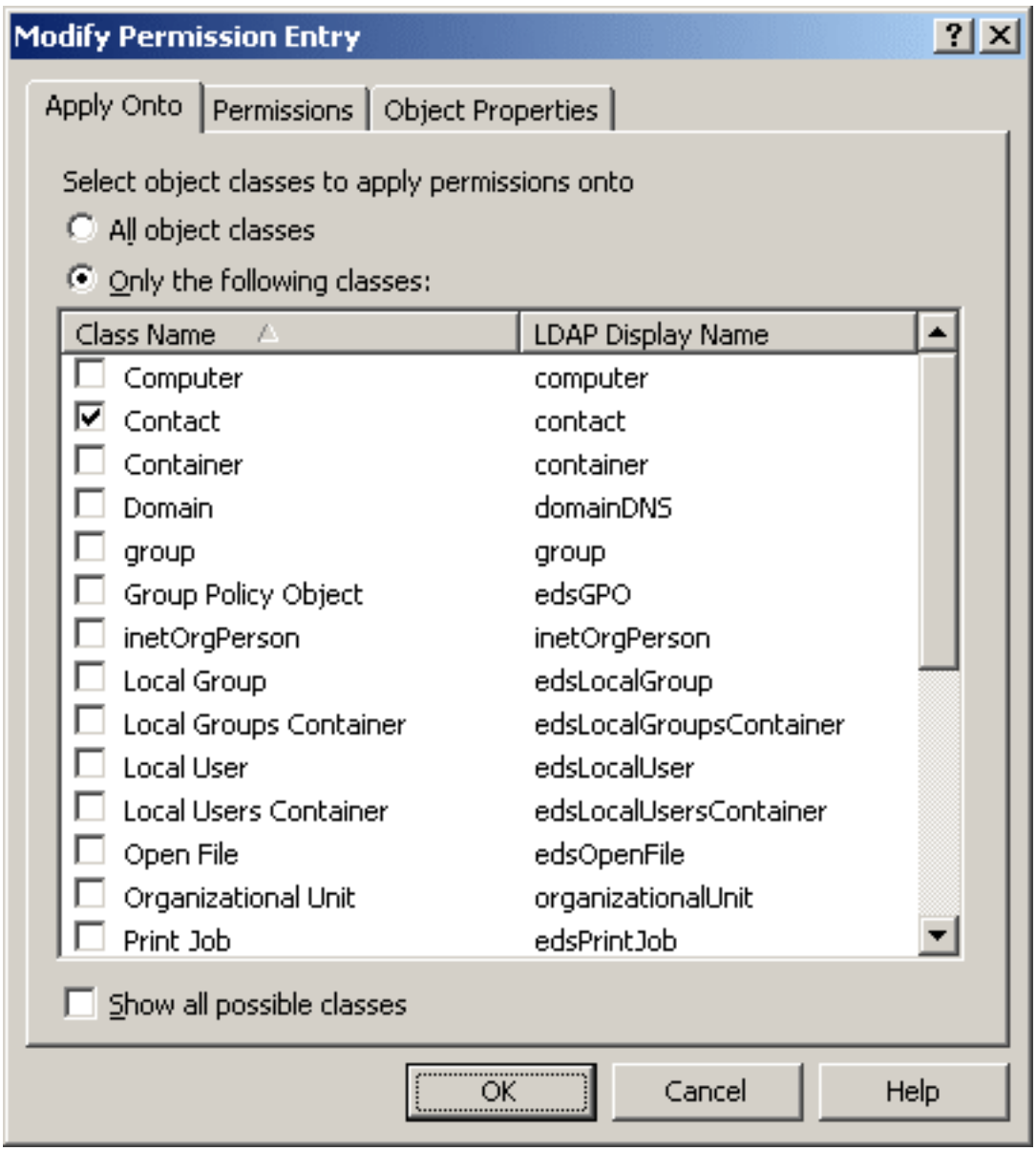
Figure 6: Access Template - Manage permissions



The **Permissions** tab in the **Properties** dialog lists permission entries defined in the Access Template. Each entry in the list includes the following information:

- **Type**: Specifies whether the permission allows or denies access.
- **Permission**: Name of the permission.
- **Apply To**: Type of objects that are subject to the permission.

Figure 7: Access Template - Modify permissions



You can use the tabs in that dialog to modify the permission as needed. The tabs are similar to the pages in the **Add Permission Entries Wizard**, discussed in About the Add Permission Entries wizard.

For the steps of how to add, remove or delete permissions from an Access Template, see *Adding, modifying, or removing Access Template permissions* in the *Active Roles Administration Guide*.

Nesting Access Templates

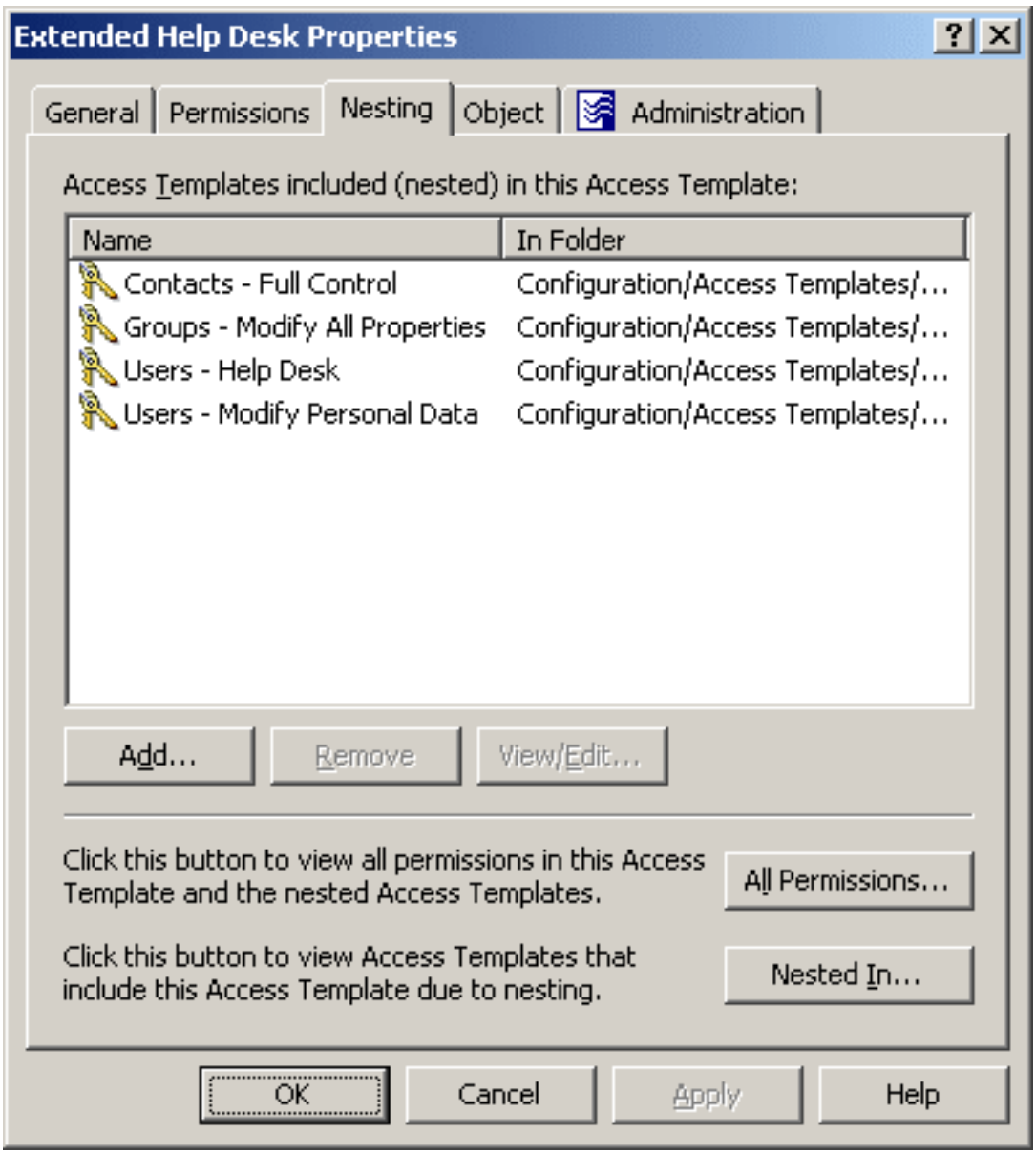
You can define permissions in an Access Template (AT) by including (nesting) other ATs. This reduces the work required if you need to create a new AT that is similar to an existing one. Instead of modifying an existing Template to add new permissions, you can nest it into a new AT.

This feature simplifies Access Template management by re-using the existing preconfigured or custom Access Templates. For example, if you need to add permissions to the predefined Help Desk Access Template, you can create a new Access Template, nest the Help Desk Access Template into the new Access Template, and add permissions to the new Access Template as needed.

To nest Access Templates to a given Access Template, use the **Nesting** tab in the **Properties** dialog for that Access Template.

The **Nesting** tab lists all Access Templates that are included (nested) in the selected Access Template, similar to the following figure:

Figure 8: Nesting Access Templates



Each entry in the list provides the following information:

- **Name**: The name of the nested Access Template.
- **In Folder**: Path to the container that holds the nested Access Template.

You can manage the list on the **Nesting** tab by using the button beneath the list:

- **Add**: Click this button to select Access Templates you want to nest into the Access Template being administered.
- **Remove**: Select Access Templates from the list and click this button to remove them from the Access Template being administered.

- **View/Edit**: Select an Access Template from the list and click this button to view or modify the selected Access Template.

From the **Nesting** tab, you can also access the following information:

- **All Permissions**: Displays all permissions in the Access Template, including those that come from the nested Access Templates.
- **Nested In**: Displays a list of Access Templates in which the Access Template is included due to nesting.

Examples of using Access Templates

This section discusses scenarios to help you understand and use the role-based administration features available in Active Roles. The following scenarios are covered:

- Example 1: Implementing a helpdesk
- Example 2: Implementing Self-Administration

Example 1: Implementing a helpdesk

This scenario shows how to use an Access Template that allows a helpdesk service to perform day-to-day operations on user accounts, such as resetting passwords, viewing user properties, locking and unlocking user accounts.

The scenario also involves a group to hold helpdesk operators. The Access Template is applied so that the group is designated as a Trustee, thus giving the administrative rights to the helpdesk operators. When both the Access Template and group are prepared, you can implement a helpdesk administration in your organization.

For example, if you need to authorize the helpdesk to manage user accounts in the **Sales** Organizational Unit, you must perform the following steps:

1. Prepare a **Helpdesk** Access Template that defines the help desk operator permissions on user accounts.
2. Create and populate a **Helpdesk** group to hold the helpdesk operators.
3. Apply the **Helpdesk** Access Template to the **Sales** Organizational Unit, selecting the **Helpdesk** group as a Trustee.

As a result of these steps, each member of the **Helpdesk** group is authorized to perform management tasks on user accounts in the **Sales** Organizational Unit. The **Helpdesk** Access Template determines the scope of the tasks.

The following sections elaborate on each of these steps.

Preparing a helpdesk Access Template

For the purposes of this scenario, you can use the predefined Access Template **Users – Help Desk**, located in the **Configuration** > **Access Templates** > **Active Directory** container. The **Users – Help Desk** Access Template specifies the necessary permissions to reset user passwords, unlock user accounts, and view properties of user accounts.

If you want to add or remove permissions from the **Users – Help Desk** Access Template, you need to first create a copy of that Access Template, then modify and apply the copy.

This scenario assumes that you apply the predefined Access Template **Users – Help Desk**.

Creating a helpdesk group

To create a group, right-click an Organizational Unit (OU) in the **Console tree**, select **New** > **Group**, and follow the instructions of the **New Object - Group** wizard. The wizard includes the page where you can add members (in this case, helpdesk operators) to the group you are creating.

For step-by-step instructions on how to create groups, see *Creating a Group* in the *Active Roles User Guide*.

Applying the Help Desk Access Template

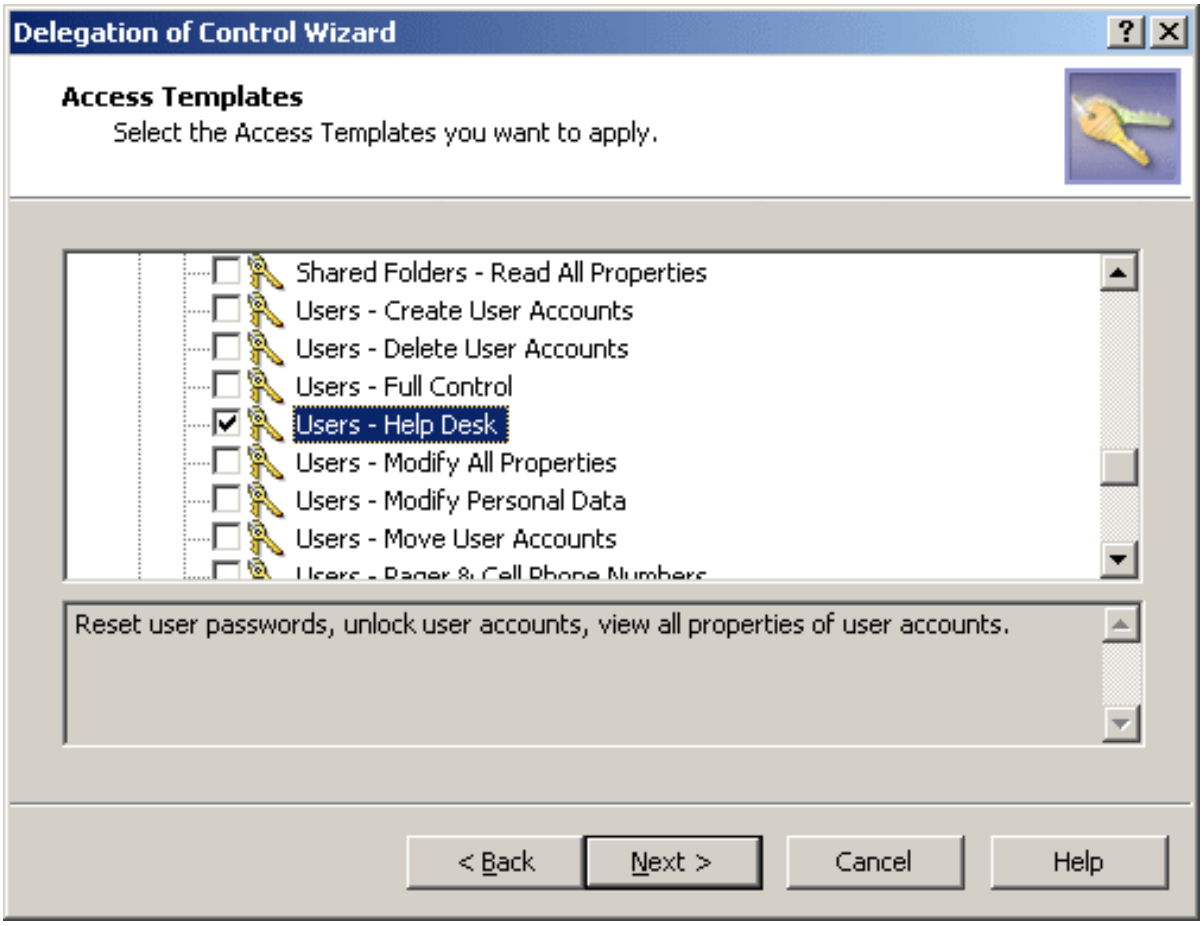
You can apply the Access Template using the **Delegation of Control Wizard**.

First, you start the wizard on the **Sales** Organizational Unit: right-click the Organizational Unit, click **Delegate Control**, and then, in the **Active Roles Security** window, click **Add**.

Next, on the **Users or Groups** page of the wizard, add the **Help Desk** group to the list.

Next, on the **Access Templates** page of the wizard, expand **Access Templates** > **Active Directory** and select the check box next to **Users - Help Desk**, as shown in the following figure.

Figure 9: Access Template – Delegation of control



Click **Next** and accept the default settings in the wizard. On the completion page, click **Finish**. Finally, click **OK** to close the **Active Roles Security** window.

For more information about the **Delegation of Control Wizard**, see *Applying Access Templates* in the *Active Roles Administration Guide*.

Example 2: Implementing Self-Administration

This scenario shows how to use an Access Template that allows users to modify certain portions of their personal information in Active Directory.

The Active Roles Web Interface provides the Site for Self-Administration to manage user accounts. The site displays users their personal information, such as the first and last names, address information, phone numbers, and other data. By default, Web Interface users are only authorized to view their personal information. To enable the users to also modify their personal information, you must give them additional permissions.

Suppose you need to authorize the users in the **Sales** Organizational Unit to perform self-administration. To implement this scenario, you should perform the following steps:

1. Prepare a **Self-Administration** Access Template that defines the appropriate permissions on user accounts.
2. Apply the **Self-Administration** Access Template to the **Sales** Organizational Unit, selecting the **Self** object as a Trustee.

As a result of these steps, users from the **Sales** Organizational Unit are authorized to perform self-management tasks on their personal accounts. The **Self-Administration** Access Template determines what data the users are permitted to modify. Users can manage their personal information via the Site for Self-Administration. For information about the Site for Self-Administration, refer to the *Active Roles Web Interface User Guide*.

The following sections elaborate on the steps involved in this scenario.

Preparing a self-administration Access Template

For the purposes of this scenario, you can use the predefined **Self - Account Management** Access Template, located in the **Configuration** > **Access Templates** > **User Self-management** container. This Access Template specifies the necessary permissions to view a basic set of user properties and modify telephone numbers.

If you want to add or remove permissions from the **Self - Account Management** Access Template, you need to first create a copy of that Access Template, then modify and apply the copy.

This scenario assumes that you apply the predefined **Self - Account Management** Access Template.

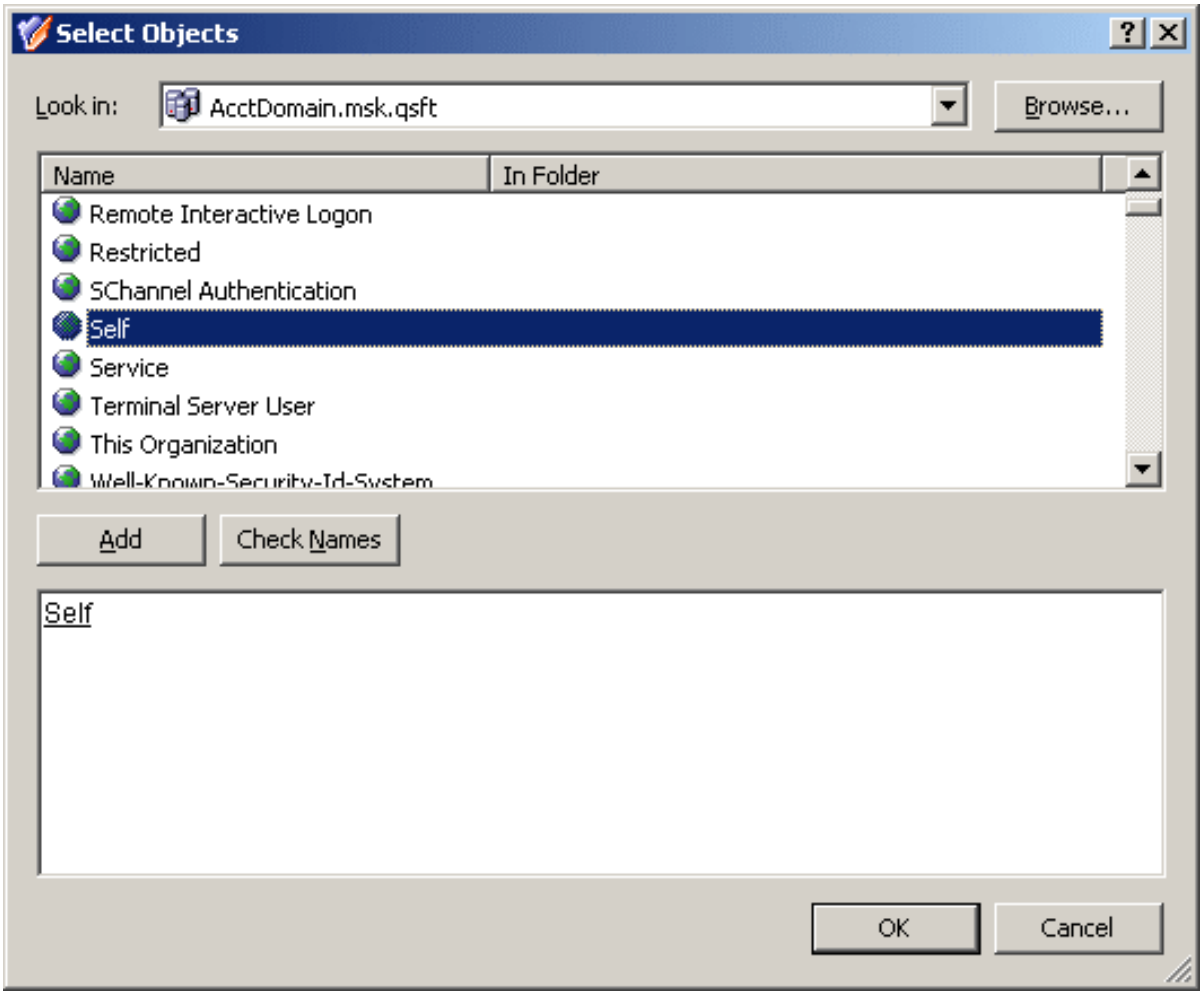
Applying the self-administration Access Template

You can apply the Access Template using the **Delegation of Control Wizard**.

First, you start the wizard on the **Sales** Organizational Unit (OU): right-click the OU, click **Delegate Control**, and then, in the **Active Roles Security** window, click **Add**.

Next, on the **Users or Groups** page of the wizard, click **Add**. In the **Select Objects** window, select the **Self** object, as shown in the following figure, click **Add**, and then click **OK**.

Figure 10: Access Template – Self administration



Next, on the **Access Templates** page of the wizard, expand **Access Templates** > **User Self-management** and select the check box next to **Self - Account Management**.

Click **Next** and accept the default settings in the wizard. On the completion page, click **Finish**. Finally, click **OK** to close the **Active Roles Security** window.

For more information about the **Delegation of Control Wizard**, see *Applying Access Templates* in the *Active Roles Release Notes*.

Deployment considerations for Access Templates

Active Roles utilizes role-based delegation for assigning administrative permissions. The advantage of this model is that a role can be created once and delegated to multiple groups of users that fit that role. If a change is needed, an update to the role will take effect for everyone. These roles are known in Active Roles as **Access Templates**.

When doing delegation with Active Roles, consider the following:

- Active Roles administrators (Active Roles Admins) have full control throughout the system and cannot be denied access anywhere within Active Roles. Everyone else starts with nothing and permissions are added from the ground up.
- Permissions are cumulative. You must adhere to the following permission precedence. The permission on top has the highest precedence:
    1. Explicit Deny
    2. Explicit Allow
    3. Inherited Deny
    4. Inherited Allow
- Keep your permission model as simple as possible. Sometimes this means giving users all read/write permissions and denying the ability to write a few fields.
- Do not use the default (built-in) Access Templates as they cannot be modified. Instead, copy those Access Templates and move them to a new container. This way all of the Access Templates you are using are stored within a particular structure.

There are three basic types of permissions that can be added to an Access Template:

- **Object access**: With this permission type, you can set permissions that affect an object as a whole. For example, `Move`, `List` and `Deprovision` are object permissions.
- **Object property access**: These are used to control access to individual attributes of an object, such as the object description, `samAccountName`, or `homeFolder`. With this permission type, you can delegate granular rights over an object. However just because the rights that can be delegated can be granular does not mean that they should. For instance, if a helpdesk operator needs to be able to manage a large set of user properties, it makes more sense to delegate read/write for all properties as this is one permission entry instead of delegating read/write for every individual attribute since each attribute would need to have its own permission entry.
- **Child object creation/deletion**: With this permission type, you can set permissions for creation or deletion of objects. For instance, to set up an Access Template that allows creation of users, you should add a permission entry that applies to the Organizational Unit and Container object classes, and contains a `Create child objects` permission for the User object class.

The following sections give a sample set of the permissions necessary for certain delegation scenarios:

- Delegation of Organizational Unit administration duties
- Delegation of group administration

Delegation of Organizational Unit administration duties

The following table lists a sample set of permission entries for a scenario of delegating administration duties for Organizational Units:

Table 2: Permission entries for delegating administration of Organizational Units

Object Class	Permission Type	Attribute or Permission
Domain	Object Access	Allow List
Domain	Object Property Access	Allow Read All Properties
Domain	Object Property Access	Allow Write LDAP Server (permission to change Operational Domain Controller)
Organizational Unit	Object Access	Allow List
Organizational Unit	Object Property Access	Allow Read All Properties
Organizational Unit	Child Object Creation/Deletion	Allow Create/Delete Users
User	Object Access	Allow List
User	Object Property Access	Allow Read/Write All Properties
User	Object Property Access	Deny Write Employee ID

This set of permission entries has several important characteristics:

- It allows access to the Domain and the Organizational Unit object classes. This is because without access to the domain and the Organizational Units a delegated administrator cannot see the users beneath. This access must include the `List` and `Read All Properties` permissions.
- It gives a delegated administrator the ability to create and delete user objects. This permission applies to the Organizational Unit object class.
- It gives a delegated administrator the ability to see (List) users and modify any property except `Employee ID`.

Delegation of group administration

The following table lists a sample set of permission entries for a scenario of delegating administration of groups:

Table 3: Permission entries for delegating administration of groups

Object Class	Permission Type	Attribute or Permission
Domain	Object Access	Allow List
Domain	Object Property Access	Allow Read All Properties
Domain	Object Property Access	Allow Write LDAP Server (permission to change Operational Domain Controller)
Organizational Unit	Object Access	Allow List
Organizational Unit	Object Property Access	Allow Read All Properties
Organizational Unit	Child Object Creation/Deletion	Allow Create/Delete Groups
Group	Object Access	Allow List
Group	Object Property Access	Allow Read All Properties
Group	Object Property Access	Allow Write Members
User	Object Access	Allow List
User	Object Property Access	Allow Read All Properties

This set of permission entries has several important characteristics:

- It allows access to the Domain and the Organizational Unit object classes. This is because without access to the domain and the Organizational Units a delegated administrator cannot see the groups and users beneath. This access should always include the `List` and `Read All Properties` permissions.
- It gives a delegated administrator the ability to create and delete group objects. This permission applies to the Organizational Unit object class.
- It gives a delegated administrator the ability to see (`List`) groups, view any property of a group (`Read All Properties`), and add or remove members from a group (`Write Members`).
- It gives a delegated administrator the ability to see (`List`) users and view any

property of a user (`Read All Properties`). This is necessary for a delegated administrator to add users to a group.

Delegation in functional and hosted environments

For your delegation model to work correctly, you need to determine whether you have a functional or hosted environment.

Delegation in a functional environment

In a functional environment there is a separate group of administrators for each function. So there may be a group for managing users, a helpdesk, domain administrators, and Exchange administrators. In case of a functional environment, you need to decide on a certain role for each function. These roles usually cross Organizational Unit boundaries so delegation is typically done at the root of the domain or domains. Typically a delegation model for this scenario would look something like the following:

Table 4: Delegation model in a functional environment

Location / Template	Permission	Delegate (Trustee)
Domain / Read All Objects	• All Objects - List • All Objects - Read All Properties • Domain - Write LDAP Server Property (permission to change Operational Domain Controller)	Authenticated Users
Domain / User Admin	• User Objects - Full Control • Organizational Unit - Create/Delete User Objects	User Admin group
Domain / Group Admin	• Group Objects - Full Control • Organizational Unit - Create/Delete Group Objects	Group Admin group

Delegation in a hosted environment

In a hosted environment there is an admin group or set of admin groups responsible for each top-level Organizational Unit (OU). In this case administrators may not want others to see what is going on in their OU structure. Active Roles can accommodate this easily. Since except for the Active Roles administrators no one has any default rights, a delegation model may look something like the following:

Table 5: Delegation model in a hosted environment

Location / Template	Permission	Delegate (Trustee)
Domain / Read Domain	• Domain - List • Domain - Read All Properties • Domain - Write LDAP Server Property	Authenticated Users
Top-level OU / OU Admin	• All Objects - List • All Objects - Read all Properties • Organizational Unit - Create/Delete User/Group Objects • User Objects - Full Control • Group Objects - Full Control	OU Admin

With this delegation model, everyone can see the domain and change the domain controller they are using for management. However, below that only the OU admin can see their associated OU. This keeps administrators from seeing or managing anything outside of their control.

More than likely a delegation model would incorporate features of both. For instance, you may have a hosted environment where each business unit is responsible for their own Active Directory management, with a central helpdesk to perform basic user and group management tasks.

Lastly is the issue of syncing permission to Active Directory. Although Active Roles enables you to accomplish this task, it is a better idea to keep all of the permissions within Active Roles for the following reasons:

- This protects your Active Directory. Directory-enabled applications can be modified to use the Active Roles ADSI Provider allowing for granular access to only the data and areas that are needed. Doing so helps prevent malicious software from destroying data in Active Directory.
- This ensures directory integrity. By forcing all administrators to use Active Roles, you ensure that all policies, such as naming standards, are correctly enforced.
- This gives a complete auditing picture. By having all applications and administrators use Active Roles interfaces, you ensure that the Active Roles Data Collector can gather every action that happens in the directory, down to the attribute level.

About Access Rules

Active Roles introduces claims-based authorization rules (access rules) to allow or deny access to Active Directory objects depending on the attributes of the identity attempting to access those objects. Built on the concept of Dynamic Access Control (DAC), this feature

enables Active Roles to recognize and evaluate the attribute-based claims of the identity that requests access to data held in Active Directory.

Access rules improve access control management for Active Directory administration. With access rules, Active Roles adds more flexibility and precision in delegating control of Active Directory objects, such as users, computers or groups, through the use of claims (the Active Directory user and computer properties) in the Active Roles authorization model.

By using access rules, you can control access to Active Directory objects based on the characteristics of both the objects and the delegated administrators requesting access to the objects. This feature enables you to define and enforce very specific requirements for granting administrative access to Active Directory data. For example, you can easily restrict access of delegated administrators to user accounts whose properties (such as **Department** or **Country**) match the properties of the delegated administrator's account in Active Directory.

Access rules help you create more complete access controls on Active Directory objects by comparing object properties with user and device claims. A domain controller issues claims to an identity that consist of assertions based on the properties of that identity retrieved from Active Directory. When an identity requests access to a particular object, Active Roles evaluates the claims of that identity and the properties of that object against the access rules, and then, depending upon the evaluation results, applies the appropriate Access Templates to make an authorization decision.

Understanding Access Rules

Access Rules enable administrators to apply access-control permissions and restrictions based on well-defined conditions that can include the properties of the target objects, the properties of the user who requests access to target objects, and the properties of the device from which the user requests access to target objects. For example, when the role or the job of a user changes (resulting in changes to the attributes of the user account in Active Directory), Access Rules can cause the user permissions to change dynamically without additional intervention from the administrator.

An Access Rule is an expression of authorization rules that can include conditions that involve user groups, user claims, device groups, device claims, and target object properties. When you apply an Access Template, you can use an Access Rule to determine the conditions that must be satisfied for the permissions resulting from the Access Template to take effect.

Conditional Access Template links

Active Roles enhances its authorization model by introducing conditional Access Template links, and takes advantage of conditional links by inserting user claims, device claims, and target object properties, into conditional expressions specified in Access Rules. An Access Rule can be applied to an Access Template link, causing the link to have an effect only if the condition of the access rule evaluates to `True`. During permission check, Active Roles inserts the claims and properties into conditional expressions found in the Access Rule,

evaluates these expressions, and enables or disables the Access Template link based on results of the evaluation. In this way, the Access Rule determines the results of the permission check.

Access Rules, along with conditional Access Template links, enable Active Roles to leverage claims for authorization to securable objects. This authorization mechanism (known as claims-based access control) supplements Access Template based access control to provide an additional layer of authorization that is flexible to the varying needs of the enterprise environment.

Management of Windows claims

Claims are statements about an authenticated user or device, issued by an Active Directory domain controller running Windows Server 2016 or later. Claims can contain information about the user or device retrieved from Active Directory.

Dynamic Access Control (DAC), a feature of Windows Server 2012, employs claims-based authorization to create versatile and flexible access controls on sensitive resources by using access rules that evaluate information about the user who accesses those resources and about the device from which the user accesses those resources. By leveraging claims in the authentication token of the user, DAC makes it possible to allow or deny access to resources based on the Active Directory attributes of the user or device.

Active Roles uses claims-based access rules to improve authorization management for Active Directory administration. With claims-based access rules, Active Roles adds more flexibility and precision in delegating control of Active Directory objects, such as users, computers or groups, by extending the Active Roles authorization model to recognize and evaluate the claims specific to the user who requests access to those objects or device used to request access.

For the steps of managing Windows claims in Active Roles, see *Managing Windows claims* in the *Active Roles Administration Guide*.

Overview of claim type management

After you enable the **KDC support for claims, compound authentication and Kerberos armoring** Group Policy setting, your Windows Server 2012 (or later) domain controllers are ready to issue claims in response to authentication requests. However, you need to configure claim types before the domain controller can issue claims.

You can use Active Roles to create attribute-based claim types that source their information from user and computer attributes. The claim types you create are stored in the configuration partition of the Active Directory forest. All domains within that forest share the claim types and domain controllers from those respective domains issue claim information during user authentication.

It is important that the Active Directory attributes intended to source claim types contain accurate information. Incorrect attribute information can lead to unexpected access to data using claims-based authorization. You can ensure the accuracy of information held in claim

source attributes by leveraging property generation and validation policies provided by Active Roles.

You can use the Active Roles Console to create, modify and delete user and computer claim types. The claim type objects are stored in the configuration partition of the Active Directory forest, and appear under the **Active Directory** > **Claim Types** node in the Active Roles Console. If you have domains from multiple forests registered with Active Roles, then the Console tree provides a separate **Claim Types** node for each forest. The forest to which a given **Claim Types** node applies is identified by the name (or a part of the name) of the forest root domain shown in brackets next to the name of the node.

The Active Roles Console provides the following pages for creating and modifying claim types:

- **Source Attribute**: On this page you can select the Active Directory attribute from which the claim value is obtained, specify the display name and description for the claim type, and choose whether the claim type applies to a user, computer, or both.
- **Suggested Values**: This page allows you to configure predetermined selectable values from which you can choose when using the claim type in a conditional expression for an access rule.

On these pages you can view or change the following configuration settings.

About the Source Attribute setting

On the **Source Attribute** page you can select, view or change the source attribute for the claim type. The source attribute is the Active Directory attribute from which the value is obtained for claims of this claim type.

The page provides a list allowing you to select the desired attribute. The list includes the attributes for the `User`, `Computer`, `InetOrgPerson`, `ManagedServiceAccount`, `GroupManagedServiceAccount` and `Auxiliary` classes of object, with the exception of:

- Attributes marked as defunct in the Active Directory schema.
- Password attributes such as `dBCSPwd`, `lmPwdHistory`, and `unicodePwd`.
- Attributes that are not replicated among domain controllers.
- Attributes that are not available on read-only domain controllers.
- Attributes with an Active Directory syntax type other than:
  - String: `DN String`, `Unicode`, `NT Security Descriptor`, or `Object ID`.
  - `Integer` or `Large Integer`.
  - `Boolean`.

For an existing claim type, the page displays the current source attribute of the claim type, and allows you to select a different attribute of the same syntax type. However, changing the source attribute does not change the ID of the claim type.

About the claim type identifier setting

The claim type identifier (ID) determines the Common Name (cn) of the claim type object in Active Directory. Normally, Active Roles automatically generates an ID when creating a claim type. The automatically generated ID has the following format:

`ad://ext/attributeName:uniqueHexidecimalNumber`

In this format, `attributeName` stands for the LDAP display name of the source attribute of the claim type, while `uniqueHexidecimalNumber` is a random-generated string of hexadecimal characters that ensures the uniqueness of the claim type ID.

To enable authorization scenarios where claims are used across a forest trust, you need to create claim types in both the trusted forest and trusting forest with the same claim type ID. Domain controllers in a trusting forest receiving claims from a trusted forest cannot understand these claims unless:

- Each claim has a claim type object created in both forests.
- The claim type ID in the trusting forest is identical to the claim type ID in the trusted forest.
- A Claim Transformation Policy Object is applied to allow incoming claims across the forest trust.

Therefore, when you create a claim type object, you may need to specify the appropriate claim type ID by hand. The option **Set ID to a semantically identical claim type in a trusted forest** serves this purpose, allowing you to type in an ID instead of having it created automatically. If you choose to enter an ID by hand, ensure that your ID string specifies a unique ID and conforms to the following format:

- Starts with the `ad://ext/` prefix.
- The prefix is followed by 1 to 32 characters.
- Does not contain space characters or these characters: `\ * ? " < > |`.
- If a slash mark (`/`) occurs after the `ad://ext/` prefix, then the slash mark must be surrounded by a character on each side. The surrounding character must not be a colon (`:`) or slash mark.

A valid example of an ID string is `ad://ext/BusinessImpact`.

The option **Set ID to a semantically identical claim type in a trusted forest** is available only when you create a claim type object. The ID should not be changed on existing claim type objects. When you create a claim type object, it is advisable to let an ID be generated automatically unless a business need justifies otherwise, such as the use of claim transformation policies in a multi-forest environment. This ensures that the newly created claim type has a valid, unique ID.

About the display name setting

The display name of the claim type object is used to represent the claim type as a choice throughout the user interface. Thus, when you configure a conditional expression for an

access rule, the condition builder allows you to select a claim type from a list where each list item is the display name of a certain claim type object. For this reason, each claim type object must be given a unique display name. The display name accepts alphanumeric characters as valid data.

About the description setting

You can use the description of the claim type object to specify a short comment about the claim type. Comments typically include purpose, department usage, or business justification.

About the user or computer claim issuance setting

You have the option to choose whether claims of the given claim type can be issued for user or computer object class, or both. With the option to issue claims for the user object class, the claim type causes domain controllers (DCs) to issue user claims based on the attribute of the authenticating user. With the option to issue claims for the computer object class, the claim type causes DCs to issue device claims based the attribute of the computer of the authenticating user. You can configure a claim type to issue both user and device claims. When you create a conditional expression for an access rule, and choose the claim type to evaluate, the condition builder allows you to distinguish between user and device claims of the same claim type.

Protection from accidental deletion

By default, claim type objects are protected from accidental deletion. This option prohibits all users, including domain and enterprise administrators, from deleting the claim type object. Protection is achieved by adding an explicit permission entry to the claim type object that denies everyone the right to delete the object. When you create a claim type object, the option to protect the object from accidental deletion is selected by default. As a best practice, it is advisable to leave this option selected.

About the suggested values setting

The suggested values setting allows you to configure predefined values from which you can choose when using the claim type in a conditional expression. If you create a claim type without suggested values, you will have to type rather than select values in the condition builder. Another option is to create one or more suggested values for the claim type. These values will appear in a list provided by the condition builder.

You can add, edit or remove suggested values for a given claim type when creating or modifying the respective claim type object. When you add or edit a suggested value, you are prompted to complete the following fields:

- **Value**: This value data will be used when evaluating conditional expressions that include the suggested value you are configuring.
- **Display name**: This is the name of the suggested value that appears in the list when you configure a conditional expression.

Population of claim source attributes

Creating a claim type object makes the Active Directory forest aware of the claim type. However, claim type objects do not provide information held in the actual claims. When issuing claims, domain controllers (DCs) retrieve that information from user and computer objects. Hence, in addition to claim type objects, user and computer objects must contain the information necessary for DCs to issue claims.

Attribute-based claim types define the attributes from which to source the claims. These are attributes of user and computer objects. Each claim type object specifies a certain attribute that the DC retrieves when creating and issuing claims of that type. During authentication of a user, the claim-aware DC reads all enabled claim types from the user's Active Directory forest, and maps them to the attributes of the authenticating user or computer. Then, the DC retrieves information from the mapped attributes, and issues claims containing that information.

As DCs do not issue blank claims, you may encounter a situation where you have created a valid claim type but the DC does not issue the claim during authentication. This is because a claim type object merely maps claims to a certain attribute, directing the DC to issue claims based on the information present in that attribute. If the attribute of the authenticating user or computer does not contain information, the DC does not issue the claim.

Therefore, it is important that claim source attributes contain information. Additionally, as authorization decisions depend upon information found in claims, claim source attributes must contain valid information. Incorrect attribute information can lead to unexpected access to data using claims-based authorization.

To ensure that claim source attributes contain valid information, you could periodically inspect and, if needed, set or correct the properties of users and computers by using the Active Roles Console or Web Interface. However, it would be more practical to leverage property generation and validation policies provided by Active Roles. You can use policies to:

- Auto-generate the appropriate values for user and computer properties upon creation of user and computer objects.
- Prevent invalid values from being assigned to user and computer properties, by applying validation rules or creating immutable lists of suggested values.

Property generation and validation policies allow you to specify, and enforce, conditions that the property values must meet, and determine default property values. For further information, see *Property Generation and Validation* in the *Active Roles Administration Guide*.

About the conditional expression editor

The Access Rule management pages provide a built-in editor for configuring conditional expressions. Each Access Rule holds a certain conditional expression that evaluates during permission check. A conditional expression is composed of conditions combined using AND/OR logic. Each condition is a certain statement that specifies criteria allowing permission check to determine whether to apply a given Access Template.

When you configure a conditional expression, you need to add at least one condition, but you are not limited in the number of conditions that you can add. You can add, delete and group conditions using various operators. It is possible to nest condition groups within other condition groups to achieve the results that you want.

A condition group contains one or more conditions connected by the same logical operator. By grouping conditions, you specify that those conditions should be evaluated as a single unit. The effect is the same as if you put parentheses around an expression in a mathematical equation or logic statement.

By default, a single condition group is created when you add a condition. You can create additional condition groups to group a set of conditions and nest grouped conditions within other condition groups.

In a condition group, conditions are connected using the **AND** or **OR** logical operator:

- An **AND** group evaluates to **TRUE** if all conditions in the group are **TRUE**.
- An **OR** group evaluates to **TRUE** if any condition in the group is **TRUE**.

By default, **AND** is the logical operator between the conditions in a condition group. You can change the logical operator by converting the condition group to a different group type.

When you add a condition, the conditional expression editor first prompts you to specify what you want the condition to evaluate. The following options are available:

- **Device claim**: Evaluate a computer claim, or groups the computer account is a member of. You can choose one of the existing computer claim types or, to evaluate groups, you can select the **Group** item in the claim type list provided by the condition builder.
- **Target object property**: Evaluate a certain property of the object to which the authorizing user requests access. You can select the desired property from a list provided by the condition builder.
- **User claim**: Evaluate a user claim, or groups the user account is a member of. You can select one of the existing user claim types or, to evaluate groups, you can select the **Group** item in the claim type list provided by the condition builder.

Once you have specified what you want the condition to evaluate, you can choose a comparison operator and specify a comparison value. The comparison operator determines the operation of comparing the claim, group membership, or property with the comparison value you specified, and causes the condition to evaluate to **TRUE** or **FALSE** depending on the outcome of that operation.

The following comparison operators are available:

- **equals**: The condition evaluates to **True** if the comparison value evaluates to the exact value of the claim or property; otherwise, the condition evaluates to **False**.
- **does not equal**: The condition evaluates to **False** if the comparison value evaluates to the exact value of the claim or property; otherwise, the condition evaluates to **True**.
- **member of any**: The condition evaluates to **True** if the comparison value lists any of the groups the user (or computer) is a member of. If the user (or computer) is not a member of any of the groups listed in the comparison value, the condition evaluates to **False**.
- **member of each**: The condition evaluates to **True** if the comparison value lists only the groups the user (or computer) is a member of. If the user (or computer) is not a member of each group listed in the comparison value, the condition evaluates to **False**.
- **not member of any**: The condition evaluates to **False** if the comparison value lists any of the groups the user (or computer) is a member of. If the user (or computer) is not a member of any of the groups listed in the comparison value, the condition evaluates to **True**.
- **not member of each**: The condition evaluates to **False** if the comparison value lists only the groups the user (or computer) is a member of. If the user (or computer) is not a member of each group listed in the comparison value, the condition evaluates to **True**.

You can choose from the following options to specify a comparison value:

- **Device claim**: The comparison value is the value of a certain computer claim. You can select one of the existing computer claim types from the claim type list provided by the condition builder.
- **Target object property**: The comparison value is the value of a certain property of the object to which the authorizing user requests access. You can select the desired property from a list provided by the condition builder.
- **User claim**: The comparison value is the value of a certain user claim. You can select one of the existing user claim types from the claim type list provided by the condition builder.
- **Value**: Depending on what the condition is intended to evaluate, this option allows you to specify a particular text string, integer, Boolean value (`True` or `False`), or a list of groups. In case of a claim type that provides a list of suggested values, the condition builder prompts you to select a value from the list.

About rule-based autoprovisioning and deprovisioning

Active Directory (AD) supports delegating control with fine granularity. However, simply restricting control, access and permissions may not always be a sufficient or effective way of managing the resources of an organization.

Many directory administration processes (such as creating or disabling user accounts, enforcing user name conventions, resetting passwords, and so on) are based on predefined workflows that often share the same procedures. In practice, this means that administrators have to repeatedly perform configuration tasks with similar steps.

To make the management of such administrative tasks easier, Active Roles provides a policy-based administration solution to automate and speed up repeat procedures when administering on-premises, hybrid and Azure cloud-only objects. This approach is represented with **Policy Objects**, available in the **Configuration** > **Policies** > **Administration** node of the Active Roles Console.

For more information on configuring autoprovisioning and deprovisioning rules, see *Configuring rule-based autoprovisioning and deprovisioning* in the *Active Roles Administration Guide*.

Summary of Policy Objects

Each configured Policy Object contains one or more policies, defining either the behavior of the Active Roles system, or the actions that Active Roles performs when certain directory objects are created, modified, or deleted. This way, Active Roles can automate the administrative workflow within the organization.

Policy Objects specify what AD objects to change, how, when, whenever they are created, modified, or deleted. You can also configure policies to have Active Roles accept certain data changes only if they conform to the formatting requirements specified by the policy. This helps maintain control over the data stored in AD, and also keeps network objects in a consistent state with each defined policy.

To offer additional flexibility for configuring policies, Active RolesPolicy Objects can also run customizable scripts before or after running a task. These scripts are handled via the **Script Execution** Policy Objects.

Example: Use case for setting up a policy

A typical use case for an Active Roles policy is to automate the administration of a new employee. When creating a user account for a new employee, you can create a policy that makes Active Roles automatically perform all of the following steps:

1. Retrieve information from the HR database of the organization.
2. Use the retrieved information as the default data for filling user account properties, such as name, contact information, and so on.
3. Create a home folder and home share for the new user account.
4. Add the user account to all relevant groups within the organization.
5. Create an Exchange mailbox for the user account, and add the mailbox to the relevant distribution lists.

With one or more properly configured Policy Objects, this entire procedure can be performed either automatically, or with minimal manual administrator work. Without policies, it would require time-consuming manual administrative actions each time a new user is administered.

NOTE: Active Roles does not automatically check for changes in directory objects, containers or groups specified for provisioning in the configured Policy Objects. This means that if any changes are made in any directory resources in use in a policy, you must update the impacted policies manually. For example, if a directory group used by a **Group Membership AutoProvisioning** Policy Group is deleted, the Policy Group must be updated manually to reflect the changes.

Advantages of using Policy Objects

Configuring Policy Objects has the following advantages:

- They reduce the workload and time needed to perform common administration duties by automating tasks, combining multiple tasks into a single workflow, or even eliminating certain tasks altogether.
- They offer automated (or largely simplified) workflows for provisioning, reprovisioning and deprovisioning directory objects in the organization.
- They improve network security.
- They ensure the consistency of the managed AD objects across the organization.
- They minimize administration errors.

Types of Policy Objects

To help you configure, organize and apply Policy Objects, they are grouped into two main categories in the Active Roles Console:

- **Provisioning Policy Objects** are used to specify provisioning rules, such as:
    - Populating and validating directory data.
    - Creating account resources (such as home folders and mailboxes).
    - Administering access to resources within the organization.
- **Deprovisioning Policy Objects** are used to deprovision a selected user or group. Deprovisioning rules can include:
    - Removing user accounts or email addresses.
    - Revoking group and distribution list memberships.
    - Disabling security permissions and application access rights.

Both categories can contain multiple Policy Objects. For more information on configuring these Policy Objects, see the relevant subsections of *Policy configuration tasks* in the *Active RolesAdministration Guide*.

Built-in Policy Objects

To help you get started with configuring policy-based administration in your organization, Active Roles includes a set of built-in Policy Objects that offer provisioning and deprovisioning rules to the most typical administrative use cases. To find the built-in Policy Objects, navigate to the following node of the Active Roles Console:

Configuration > **Policies** > **Administration** > **Builtin**

To help you configure **Script Execution** policies, Active Roles also ships with several built-in **Script Modules** that you can use to set up your own **Script Execution** policies. Find these built-in **Script Modules** in the following node of the Active Roles Console:

Configuration > **Script Modules** > **Builtin**

About Provisioning and Deprovisioning Policy Objects

A Policy Object is a collection of administrative policies that specifies the business rules to enforce. A Policy Object includes stored policy procedures and specifications of events that activate each procedure.

A Policy Object associates specific events with its policy procedures, that are either built-in procedures or custom scripts. This provides an easy way to define policy constraints, implement sophisticated validation criteria, synchronize different data sources, and perform a number of administrative tasks as a single batch.

Active Roles enforces business rules by linking Policy Objects to:

- Active Roles administrative views (known as Managed Units).
- Active Directory containers, such as Organizational Units.
- Individual (leaf) directory objects, like user or group objects.

By choosing where to link a Policy Object, you determine the policy scope. For example, if you link a Policy Object to a container, all objects in the container and its sub-containers are normally subject to the Policy Object.

You can link different Policy Objects to different containers to establish container-specific policies. This is required, for example, if each Organizational Unit uses a dedicated Exchange Server to store mailboxes or file server to store home folders.

You can also link a Policy Object to a leaf object, such as a user object. As an example, consider a policy that prohibits changes to group memberships when copying a certain user object.

Policy Objects define the behavior of the system when directory objects are created, modified, moved, or deleted within the policy scope. Policies are enforced regardless of administrative rights of a user performing a management task.

IMPORTANT: Administrative policies configured via Policy Objects affect every user in their scope, even Active Roles Administrators.

For more information on configuring autoprovisioning and deprovisioning rules, see *Configuring rule-based autoprovisioning and deprovisioning* in the *Active Roles Administration Guide*.

Policy Object deployment considerations and event handlers

Active Roles enforces policies by applying Policy Objects to promote data integrity throughout the directory. This is done by generating and validating the data entered into the directory. Each Policy Object is basically a container that holds one or more policy entries (also referred to as policies).

There are several types of policy entries that can be configured within a Policy Object. The two major ones are Property Generation and Validation, and Script Execution. Property Generation and Validation policy entries provide a point-and-click interface for creating basic rules for attribute population. Script Execution policy entries enable the use of scripting for a broad range of custom actions that could supplement, extend, or replace the policy types included with Active Roles out of the box.

Just as with Group Policy Objects in Active Directory, the location that Active Roles' Policy Objects are linked to is critical:

- Any policies that are intended to affect the entire domain should be included into a Policy Object linked at the domain level. If needed, filtering can be used to exclude specific objects or containers (Organizational Units) from being processed by these policies.
- If more than one object or containers needs to be excluded from the effect of a domain-wide policy, it is best to include those objects or containers explicitly into a Managed Unit and then apply policy filtering to that Managed Unit by using the **Block Inheritance** option.

From here, the best way to apply policies is at the top level of the directory tree they will affect. Usually, however policies are only needed to affect certain Organizational Units within the tree. In this case, a Managed Unit is the most effective way to apply the policies. Include the desired Organizational Units explicitly into a Managed Unit, and then link the Policy Object to that Managed Unit.

A policy consists of three major components. These are:

- A policy entry that defines the policy
- A Policy Object containing that policy entry
- A Policy Object link that determines where the policy is applied in the directory

Typically, a single Policy Object includes all the entries for a specific set of policies. It is not efficient to create one entry per Policy Object since this defeats the purpose of having separation between the Policy Object and policy entries.

A policy cannot be filtered for specific sets of administrators. Once applied to a given object or container, a policy will be in effect for every administrator under every condition. This is unless a Script Execution policy is included as a policy entry that utilizes the

IEDSEffectivePolicyRequest interface to override the policies determined by other policy entries. This interface is documented in Active Roles SDK.

Script Execution polices are policy entries that utilize scripts written in a scripting language such as Microsoft Windows PowerShell or VBScript. Policy scripts use event handles that are initiated before or after every action that can happen in the directory. See the following table for a list of these handlers.

Table 6: Event handlers

Name	Description
onPreCreate	In a script policy applied to a container; receives control upon a request to create an object in that container. This enables a script to perform custom actions prior to creating an object.
onPostCreate	In a script policy applied to a container; receives control after a request to create an object in that container is completed. This enables a script to perform custom actions further to creating an object.
onPreDelete	Receives control upon a request to delete an object. Enables a script to perform custom actions prior to deleting an object.
onPostDelete	Receives control after a request to delete an object is completed. Enables a script to perform custom actions further to deleting an object.
onPreModify	Receives control upon a request to start changing object properties. Enables a script to perform custom actions prior to applying changes to an object.
onPostModify	Receives control after a request to change object properties is completed. Enables a script to perform custom action further to changing an object's property values.
onPreMove	In a script policy applied to a container, this function receives control upon a request to start moving an object from that container. This enables a script to perform custom actions prior to moving an object.
onPostMove	In a script policy applied to a container, this function receives control after a request to move an object to that container is completed. This enables a script to perform custom actions further to moving an object.
onPreRename	Receives control upon a request to start renaming an object. Enables a script to perform custom actions prior to renaming an object.
onPostRename	Receives control after a request to rename an object is completed. Enables a script to perform custom actions further to renaming an object.

Name	Description
onPreGet	Receives control upon a request to retrieve object properties. Enables a script to perform custom actions prior to starting the retrieval of an object's property values.
onPostGet	Receives control after a request to retrieve object properties is completed. Enables a script to perform custom actions following the retrieval of an object's property values.
onPreSearch	Receives control upon a request to start a search. Enables a script to perform custom actions prior to starting a search.
onPreDeprovision	Receives control upon a request to run the Deprovision operation. Enables a script to perform custom actions prior to starting the operation.
onDeprovision	Receives control in the course of processing a request to run the Deprovision operation. Enables the use of a script for customizing the behavior of the operation.
onPostDeprovision	Receives control after a request to run the Deprovision operation is completed. Enables a script to perform custom actions following the operation.
onPreUnDeprovision	Receives control upon a request to run the Undo Deprovisioning operation. Enables a script to perform custom actions prior to starting the operation.
onUnDeprovision	Receives control in the course of processing a request to run the Undo Deprovisioning operation. Enables the use of a script for customizing the behavior of the operation.
onPostUnDeprovision	Receives control after a request to run the Undo Deprovisioning operation is completed. Enables a script to perform custom actions following the operation.
onPreUnDelete	Receives control upon a request to run the Undelete operation. Enables a script to perform custom actions prior to starting the operation.
onPostUnDelete	Receives control after a request to run the Undelete operation is completed. Enables a script to perform custom actions following the operation.
onCheckPropertyValues	Receives control upon a request to verify and validate the changes that are going to be made to an object. Enables a script to perform custom actions further to normal validity checks on an object.
onGetEffectivePolicy	Receives control upon a request to retrieve the policy settings that are in effect on a particular object (such as policy constraints on property values). Enables a script to perform

Name	Description
	custom actions further to retrieval of policy settings.
onInit	Receives control when the Administration Service retrieves the definition of the script parameters, enabling the script to manifest the name and other characteristics of each parameter.
onFilter	Boolean-valued function that is evaluated during execution of the onPreSearch event handler, allowing search results to be filtered based on properties of objects returned by the search.

Basically, when an action happens, Active Roles looks to see if there are any Policy Objects applied that hold Script Execution policies. If so, the policy script is checked to see if it has an event handler for the specific action being performed. The object being acted upon is passed into the event handler for further actions. These event handlers are normally run in the security context of the service account, so even if a user does not have rights to perform the actions outlined in the policy script, it will still run correctly. If any errors occur during the execution of a policy script, the errors can be found in the Active Roles event log for post-action handlers and are displayed to the client for pre-action handlers.

Policy scripts are typically written in a scripting language such as Windows PowerShell or VBScript.

It is also important to note that policy scripts can pick up and take action upon directory changes made natively, as well. To turn on this behavior, you should choose the option that directs in the policy script to handle directory changes reported by the directory synchronization function (select the **Handle changes from DirSync control** check box on the **Script Module** tab in the **Properties** dialog for the policy entry), and use the **IEDSRequestParameters** interface in a post-action event handler.

Application of provisioning or deprovisioning Policy Objects

Implementing a policy to enforce business rules is a two-phase process where configuring the policy within a Policy Object is only the first step. When you create a new policy, you select a policy type from the available options, then define the options that make up the policy. The second step is to use Active RolesConsole to enforce the policy on the selected areas of the directory.

Active Roles allows enforcing of policies on any type of directory object, whether it is an administrative view (Managed Unit), a directory folder (container) or an individual (leaf) object. Policies are enforced by applying (linking) a Policy Object that holds the policies.

When you apply a Policy Object to a Managed Unit or directory folder, the policies control both the objects as well as the Managed Unit or folder that contain the objects. When you apply a Policy Object to a leaf object, such as a user or group, the policies only control that object. For example, applying a Policy Object to a group does not affect the members of the group.

Objects that are assigned to a Policy Object, that is, the objects controlled by the policies that are defined in that Policy Object, are collectively referred to as the policy scope. For example, if you apply a Policy Object to a Managed Unit, the policy scope is composed of the objects within the Managed Unit.

The policy scope normally includes all objects within the container or Managed Unit to which the Policy Object is applied. However, you might need to exclude individual objects or sub-containers from the policy scope, to prevent certain objects from being affected by policies.

TIP: You can selectively exclude objects or entire containers from the policy scope. You can also block policy inheritance on individual objects or containers, refining the policy scope.

For more information on applying policy objects, see *Applying Policy Objects* in the *Active Roles Administration Guide*.

Management of the Policy Object scope

When applying a Policy Object to a directory object, Active Roles creates a Policy Object link.

Link Policy Objects to enforce business rules on different types of directory objects in the Active Roles Console. These objects are the following:

- Administrative views (Active Roles Managed Units).
- Active Directory containers (Organizational Units).
- Individual (leaf) directory objects, such as user or group objects.

Each Policy Object link includes the following information:

- The Policy Object that defines the policy.
- The directory object that is the target of the link.
- An **Include** or **Exclude** flag that specifies whether the directory object is included or excluded from the policy scope.

When you display a list of Policy Object links for a directory object, the list appears in a separate window. Each entry in the list includes the following information:

- **Policy Object**: Name of the Policy Object.
- **Directory Object**: Canonical name of the object to which the Policy Object is linked, that is, the target object of the link.
- **Include/Exclude**: Flag that determines the behavior of the link:
    - **Include Explicitly** means the link puts the target object within the policy scope, that is, the policies defined in the Policy Object control the target object.
    - **Exclude Explicitly** means the link puts the target object out of the policy scope, that is, the policies defined in the Policy Object do not control the target object.

The **Exclude** flag takes precedence over the **Include** flag. If there are two links with the same Policy Object, one of which is flagged **Include** while another one is flagged **Exclude**, the object is effectively excluded from the policy scope of the Policy Object.

The list of Policy Object links displays the links of these categories:

- **Direct links**: The Policy Object is linked directly to the object that you have selected.
- **Inherited links**: The Policy Object is linked to a parent container or Managed Unit of the object that you selected.

From the list, you can filter out the links that are inherited from parent objects. To do this, clear **Show inherited**.

TIP: The **Remove** button is only available on direct links. If you delete links, manage them by using the **Policy Scope** command on the Policy Object.

To simplify the management of policy effect on directory objects, the Active Roles Console allows you to manage policy scope without directly managing links to Policy Objects. For a directory object, you can view and modify its policy list, that is a list of Policy Objects that control (affect) the directory object, instead of having to sort through direct and inherited links.

Each entry in the policy list includes the following information:

- **Policy Object**: The name of the Policy Object. The Policy Object controls this directory object due to a direct link or inherited links.
- **Block Inheritance**: Indicates whether the effect of the policy is blocked on this directory object.

You can also access the policy list from the advanced details pane. When you select a directory object, the list is displayed on the **Active Roles Policy** tab.

On the **Active Roles Policy** tab, you can perform the same management tasks as in the **Active Roles Policy** window. Right-click a list entry or a blank area and use commands on the shortcut menu. The commands act in the same way as the buttons in the **Active Roles Policy** window.

Given a Policy Object, you can also manage its policy scope by using a list of directory objects to which the Policy Object is applied (linked). The list can be displayed in a separate window or on a tab in the **Advanced Details Pane**:

- To display the list in a window, right-click the Policy Object and click **Policy Scope**.
- To display the list on a tab, ensure that **Advanced Details Pane** is selected on the **View** menu and select the Policy Object.

For more information on configuring Policy Object scopes, see .

Policy compliance checks

Checking for policy compliance provides information on directory data that does not comply with the policies—such as user or group naming conventions—defined with Active Roles. If

you define some policies when data has already been entered, you can check the data and modify it accordingly to ensure that the data meets the policy requirements.

Although business rules and policies normally cannot be bypassed once they have been configured, there are situations where the actual directory data may violate some of the prescribed policies or business rules. For example, when applying a new policy, Active Roles does not automatically verify the existing directory data in order to determine whether that data conforms to the new policy. Another example is a process that automatically creates new objects, such as user or group objects, by directly accessing Active Directory without the use of Active Roles.

The Active Roles Report Pack includes a number of reports that help detect policy violations in directory data by collecting and analyzing information on the state of directory objects as against the prescribed policies. However, as retrieving such information may take much time and effort, the reports on policy compliance sometimes do not allow policy-related issues to be resolved in a timely fashion.

In order to address this problem, Active Roles makes it possible to quickly build and examine policy check results on individual objects or entire containers. The policy check results provide a list of directory objects violating policies, and describe the detected violations. From the policy check results, you can make appropriate changes to objects or policies:

- Modify object properties in conformity with policies.
- Prevent individual objects from being affected by particular policies.
- Modify Policy Objects as needed.
- Perform an administrative task—for example, disable or move user objects that violate policies.

In addition, you can save policy check results to a file, print them out, or send them to an email recipient.

For more information on how to check the policy compliance of individual objects, or check if policy compliance works in general, see *Checking for policy compliance* in the *Active Roles Administration Guide*.

Overview of Provisioning Policy Objects

To configure provisioning policies for user name and email generation, group memberships, property generation or script running, use the policies available via the **Provisioning Policy Objects** option.

NOTE: Policy Object settings that are specific to Azure cloud-only objects (such as cloud-only Azure users, guest users, or contacts) are available only if your Active Roles deployment is licensed for managing cloud-only Azure objects. Contact One Identity support for more information.

Also, Policy Objects that are specific to Azure cloud-only objects will work correctly only if an Azure tenant is already configured in the AD of the organization, and Active Roles is already set as a consented Azure application for that Azure tenant. For more information

on these settings, see Configuring a new Azure tenant and consenting Active Roles as an Azure application*Configuring a new Azure tenant and consenting Active Roles as an Azure application* in the *Active Roles Administration Guide*.

For more information on configuring deprovisioning policies, see *Configuring Policy Objects* in the *Active Roles Administration Guide*.

About User Logon Name Generation

The **User Logon Name Generation** provisioning Policy Object type is used to automate the assignment of pre-Windows 2000 user login names when creating or modifying a user account, with flexible options to ensure the uniqueness of the policy-generated name.

Generating unique names

Generating a unique name is essential. If Active Roles attempts to assign a policy-generated name while an existing user account with the same pre-Windows 2000 user login name already exists, a naming conflict will occur, because Active Directory does not support multiple accounts with the same pre-Windows 2000 user login name. In such cases, to prevent naming conflicts with existing accounts, configure a policy to generate a series of names.

When configuring a **User Logon Name Generation** policy type, you can define multiple rules so that the policy applies them successively, attempting to generate a unique name in the event of a naming conflict. You can also configure a rule to include an incremental numeric value to ensure uniqueness of the policy-generated name. You also have the option to allow policy-generated names to be modified by operators who create or update user accounts.

How the User Logon Name Generation policy works

When creating a user account, Active Roles relies on the **User Logon Name Generation** policy type to assign a certain pre-Windows 2000 user login name to the user account. The policy generates the name based on the properties of the user account being created. A policy can include one or more rules that construct the name value as a concatenation of entries that are similar to those you encounter when you are using a **Property Generation and Validation** policy.

The **Uniqueness number** is a special entry used to make the policy-generated name unique. A uniqueness number entry represents a numeric value the policy will increment in the event of a naming conflict. For example, a policy can provide the option to change the new name from **ExampleUser** to **ExampleUser1** if there is an existing user account with the pre-Windows 2000 user login name set to **ExampleUser**. If the name **ExampleUser1** is also in use, the new name can be changed to **ExampleUser2**, and so on.

The policy configuration provides the option to allow or deny manual edits of policy-generated names. You can restrict granting permission to modify a policy-generated name to the case where the name is already in use by another account.

Consider the following specific behavior of the policy:

- If you have a single policy rule that does not use a uniqueness number, Active Roles simply attempts to assign the generated name to the user account. This operation can fail if the generated name is not unique, that is, the same pre-Windows 2000 user login name is already assigned to a different user account. If the policy allows the manual editing of policy-generated names, you can correct the name, if you have created the user account.
- If you have multiple policy rules or a rule that uses a uniqueness number, Active Roles adds a button at the client side, next to the **User logon name (pre-Windows 2000)** field on the user creation and modification forms.
- To generate a name, click the button next to the **User logon name (pre-Windows 2000)** field. Clicking the **Generate** button applies a subsequent rule or increases the uniqueness number by one, and this ensures that the generated name is unique.
- The following characters are not supported in pre-Windows 2000 user logon names: `" / \ [ ] : ; | = , + * ? < >`
- The policy causes Active Roles to deny processing operation requests that assign an empty value to a pre-Windows 2000 user login name.
- When checking user accounts for policy compliance, Active Roles detects and reports the pre-Windows 2000 user login names that are not compliant with the **User Logon Name Generation** policy.

For more information on configuring this Policy Object type, see *Configuring a User Logon Name Generation policy* in the *Active Roles Administration Guide*.

About E-Mail Alias Generation

E-mail Alias Generation policies automate the assignment of the email alias when designating a user as mailbox-enabled on Microsoft Exchange Server. By default, Microsoft Exchange Server provides the following recipient email address format: `<email-alias>@<domain-name>`.

You can use predefined rules to generate email aliases, or configure custom rules. For example, you can configure a policy to compose the email alias of the first initial followed by the last name of the user. Custom rules provide for the addition of an incremental numeric value to ensure uniqueness of the alias. You can also specify whether the alias can be modified by the operator who creates or updates the user account.

When making a user mailbox-enabled, Active Roles relies on this policy to assign a certain email alias to the user account. The policy generates the alias based on user properties, such as the pre-Windows 2000 user logon name, first name, initials, and last name. A custom rule can be configured to use other properties.

A custom rule can also be configured to add so-called uniqueness number. A uniqueness number is a numeric value the policy includes into the alias, incrementing that value in the event of an alias naming conflict. For example, the policy can automatically change the generated alias from **John.Smith** to **John1.Smith** if a mailbox with the alias **John.Smith** already exists. If the alias **John1.Smith** is also in use, the new alias will be changed to **John2.Smith**, and so on.

The policy configuration provides the option to allow or disallow manual edits of policy-generated aliases. Permission to modify a policy-generated alias can be restricted to the case where the alias is in use by another mailbox.

Some specific features of the policy behavior are as follows:

- With a rule that does not use a uniqueness number, Active Roles simply attempts to assign the generated alias to the user account. The operation may fail if the generated alias is not unique, that is, the alias is already assigned to a different user account. If the policy allows manual edits of policy-generated aliases, the alias can be corrected by the operator who creates the user account.
- With a custom rule that uses a uniqueness number, Active Roles adds a button at the client side, next to the **Alias** field on the user creation and modification forms.

  To generate an alias, the client user (operator) must click that button, which also applies if the generated alias is in use. Clicking **Generate** increases the uniqueness number by one, thereby allowing the alias to be made unique.
- With a custom rule configured to include user properties that are normally not displayed on the user creation forms, an extra page is added to the **New Object - User Wizard** in the Active Roles Console, making it possible to specify the user properties required to generate the alias.
- The policy defines a list of characters that are unacceptable in e-mail aliases. Space characters and the following characters are not accepted:

  `@ * + | = \ ; : ? [ ] , < > /`
- The policy denies processing of operation requests that assign the empty value to the e-mail alias.
- When checking user accounts for Active Roles policy compliance, Active Roles detects, and reports on, the aliases that are not set up as prescribed by the alias generation policy.

For more information on configuring this Policy Object, see *Configuring an E-Mail Alias Generation policy* in the *Active Roles Administration Guide*.

About Exchange Mailbox AutoProvisioning

Exchange Mailbox AutoProvisioning policies automate the selection of a mailbox store or database when designating a user as mailbox-enabled, or creating a mailbox on Microsoft Exchange Server.

When configuring the policy, you can:

- Specify Exchange Servers and mailbox stores or databases where mailbox creation is allowed.
- Specify rules to distribute mailboxes among multiple stores.

For example, you can configure a policy to automatically choose a store that holds the least number of mailboxes.

When making a user mailbox-enabled or creating a mailbox, Active Roles relies on this policy to select the mailbox store or database. The policy defines a single store, or a set of stores, in which creation of mailboxes is allowed. Some specific features of the policy behavior are as follows:

- If the policy specifies a single store, mailboxes are created in that store. A different store cannot be selected by the operator who creates or updates the user account.
- If the policy specifies multiple stores, the store is selected either automatically (by Active Roles) or manually (by the operator who creates or updates the user account), depending on policy options.

In case of multiple stores, the policy provides these options to govern the selection of a store:

- **Manually**: Allows the operator to select a store from the list defined by the policy.
- **By using the round-robin method**: Redirects mailbox creation requests sequentially across the stores, selecting the first store for the first request, the second store for the second request and so on. After the last store is reached, the next request is passed to the first store in the sequence.
- **Containing the least number of mailboxes**: Forwards mailbox creation requests to the store that holds the least amount of mailboxes.

For more information on configuring this Policy Object, see *Configuring an Exchange Mailbox AutoProvisioning policy* in the *Active Roles Administration Guide*.

About Group Membership AutoProvisioning

Group Membership AutoProvisioning policies help you to automate adding or removing the specified objects (such as user objects) to or from the specified groups.

In case of cloud-only Azure objects, you can use the Group Membership Autoprovisioning policy to automatically assign (or unassign) Azure users and Azure guest users to (or from) the specified O365 group(s) in the same Azure tenant.

To set up a policy, select the type of objects you want to provision, select the affected group(s), and then configure the policy rules. Once set up, the policy adds (or removes) directory objects to (or from) the selected groups depending on whether the provisioned objects meet the specified rules.

To help you get started with configuring policy-based administration in your organization, Active Roles includes a set of built-in Policy Objects that offer provisioning and deprovisioning rules to the most typical administrative use cases. To find the built-in Policy Objects, navigate to the following node of the Active Roles Console:

Configuration > **Policies** > **Administration** > **Builtin**

NOTE: Active Roles does not automatically check for changes in directory objects, containers or groups specified for provisioning in the configured Policy Objects. This means that if any changes are made in any directory resources in use in a policy, you must update the impacted policies manually. For example, if a directory group used by a **Group Membership AutoProvisioning** Policy Group is deleted, the Policy Group must be

updated manually to reflect the changes.

A Group Membership AutoProvisioning policy performs provisioning tasks such as adding or removing users from groups. A policy can be configured to define a list of groups and conditions so that a user account is automatically added to, or removed from, those groups depending on whether the properties of the user account meet the policy conditions.

Active Roles automatically checks users against conditions, and adds or removes users from specified groups based on the check results. Although the capabilities of this policy are similar to those provided by Dynamic Groups, a Group Membership AutoProvisioning policy gives the administrator extra flexibility and control over group memberships.

Whereas the Dynamic Groups feature delivers a rules-based mechanism for managing a group membership list as a whole, a Group Membership AutoProvisioning policy allows the administrator to define membership rules on a per-user basis. This policy automates the process of adding particular users to particular groups without affecting the other members of those groups.

For more information on configuring this Policy Object, see *Configuring a Group Membership AutoProvisioning policy* in the *Active Roles Administration Guide*.

About Home Folder AutoProvisioning

Home Folder AutoProvisioning policies automate the creation or renaming of user home folders and home shares when creating or renaming user accounts via Active Roles.

With this policy, you can:

- Specify the server on which to create home folders and home shares.
- Define how to set permissions for new home folders and shares.
- Specify naming conventions for new home folders and home shares.
- Limit the number of concurrent connections to home shares.

For example, using the Home Folder AutoProvisioning policy, you can define a corporate rule so that every time a user account is created with Active Roles, the system also creates a folder on a network file share and assigns it as the user's home folder.

For more information on configuring this Policy Object, see *Configuring a Home Folder AutoProvisioning policy* in the *Active Roles Administration Guide*.

When running a Home Folder AutoProvisioning policy, Active Roles performs various actions depending on whether a user is created, copied, or renamed.

Creating home folders and shares for new user accounts

When Active Roles creates a user account (whether from scratch or by copying an existing account), the policy can cause Active Roles to create a home folder and, optionally, a home share for the account using the path specified in the policy. The name of the home share is composed of the user name, and the prefix and suffix specified in the policy.

The policy provides the option to enable creation of home folders with paths and names that differ from the path and name prescribed by the policy. For example, a Property

Generation and Validation policy can be configured to generate the **Home Drive** and **Home Directory** properties on user accounts. When making changes to those properties, Active Roles verifies that the specified home folder exists, and creates the home folder if necessary.

A special policy is implemented in Active Roles that restricts the folders on the network file shares in which home folders can be created. The Policy Object containing that policy is located in the **Configuration/Policies/Administration/Builtin** container. The name of the Policy Object is **Built-in Policy - Home Folder Location Restriction**. You can access it by using the Active RolesConsole. The policy settings include a list of the folders on the network file shares in which creation of home folders is allowed. For instructions on how to view or modify that list, see *Configuring the Home Folder Location Restriction policy* in the *Active Roles Administration Guide*.

Renaming home folders for renamed user accounts

When Active Roles modifies the user logon name (pre-Windows 2000) of a user account, the policy can rename the home folder and, optionally, re-create the home share for that user account. The name of the new home share is set up in accordance with the naming convention specified in the policy.

The policy renames the existing home folder based on the new user logon name (pre-Windows 2000). However, if the home folder is in use, Active Roles cannot rename the folder. In this case, Active Roles creates a new home folder with the new name and does not affect the existing home folder.

Preventing disk operations on the file server

By default, Active Roles attempts to create or rename a (non-local) home folder on the file server when the Home Directory property is set or modified on a user account in Active Directory. If the creation or renaming of the home folder fails (for example, because the file server is inaccessible), then the creation or modification of the user account fails as well. To prevent such an error, a Home Folder AutoProvisioning policy can be configured so that Active Roles applies the changes to the **Home Drive** and **Home Directory** properties in Active Directory without attempting an operation on the file server. This policy option enables the use of a tool other than Active Roles for creating home folders on the file server.

Active Roles comes with a preconfigured Policy Object that allows the creation or renaming of home folders when setting home folder properties on user accounts in Active Directory. The Policy Object is located in the **Configuration/Policies/Administration/Builtin** container in Active Roles Console tree. The name of the Policy Object is **Built-in Policy - Default Rules to Provision Home Folders**. If you want to prevent Active Roles from attempting to create or rename home folders, you can modify the policy in the built-in Policy Object or configure and apply another Home Folder AutoProvisioning policy with the respective option turned off.

About Home Folder Restriction

When creating home folders, Active Roles operates in the security context of the service account under which the Administration Service is running. This means that the service account must have sufficient rights to create home folders. Normally, the service account has administrative rights on an entire file server, which enables Active Roles to create home folders in any folder on any network file share that exists on that server. To restrict Active Roles to create home folders only on a specific list of network file shares and folders, use the Home Folder Location Restriction policy.

The Home Folder Location Restriction policy determines the folders on the network file shares in which Active Roles is allowed to create home folders, and prevents Active Roles from creating home folders in other locations. The restrictions imposed by this policy do not apply if the home folder creation operation is performed by an Active Roles Admin role holder (normally, these are the users that have membership in the Administrators local group on the computer running the Active Roles Administration Service). Thus, when an Active Roles Admin role holder creates a user account, and a certain policy is in effect to facilitate home folder provisioning, the home folder is created regardless of the Home Folder Location Restriction policy settings.

By default, no network file shares and folders are listed in the policy. This means that Active Roles cannot create a home folder unless the user management operation that involves creation of the home folder is performed by the Active Roles Admin role holder. In order to allow delegated administrators to create home folders, you have to configure the policy so that it lists the folders on the network file shares in which creation of home folders is allowed.

For more information on configuring this Policy Object, see *Configuring the Home Folder Location Restriction policy* in the *Active Roles Administration Guide*.

About Property Generation and Validation

Property Generation and Validation policies automate the configuration of directory object properties. You can configure these policies to:

- Populate new directory objects with default property values (for example, when creating new user accounts or groups).
- Validate the compliance of directory property values against corporate rules.

For example, you can configure a policy to enforce a certain type of telephone number formatting in the contact information properties for your directory.

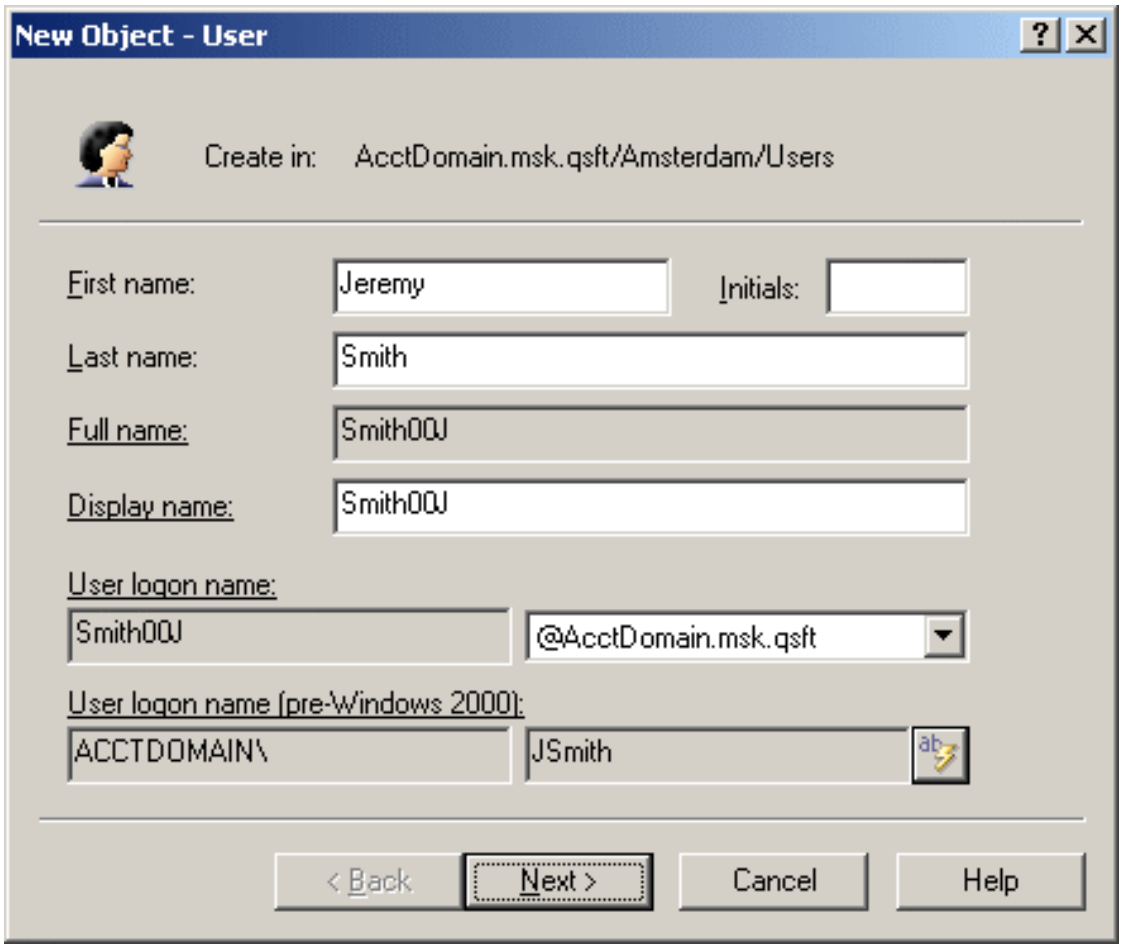
Policy-controlled properties and default values

In the **Object Creation Wizard** and **Properties** dialogs, the properties that are controlled by the policy are displayed as hyperlinks.

Example: Default values configured for a property

If you have a policy configured to populate a property with a certain value (generate the default value), the property field is unavailable for editing, as you can see in the following example.

Figure 11: Object creation



To display the policy details, click the hyperlink.

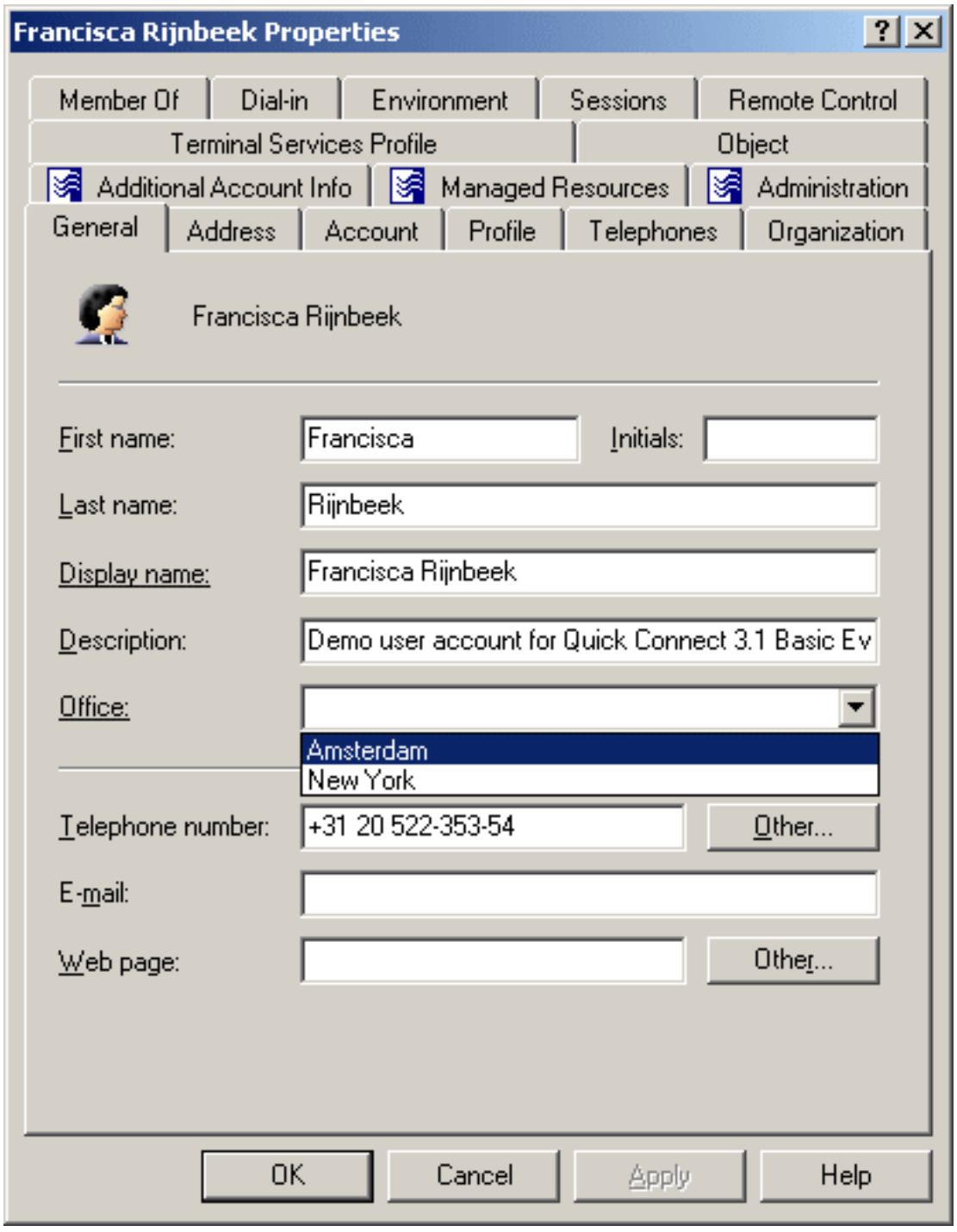
Policy-controlled acceptable values

With a policy that is configured to display only a set of acceptable values for a given property, the Active Roles Console displays a drop-down list of values when you modify that property. This saves the effort of having to type the value, or the issue of typing it incorrectly.

Example: Selecting acceptable values that are defined by a policy

In the following example, the **Office** drop-down displays a list of acceptable values that are prescribed by policy.

Figure 12: Acceptable values for a policy



Property validation

When creating or modifying an object, Active Roles checks whether the property values satisfy the criteria that were defined in the policy. If they do not, Active Roles prevents you from creating or modifying the object.

For more information on configuring this Policy Object type, see *Property Generation and Validation* in the *Active Roles Administration Guide*.

About Script Execution

Script execution policies help you to run supplementary PowerShell, or other script modules in Active Roles during or after performing certain administrative operations. When linking a custom script to an administrative operation via a **Script Execution** policy, the script will receive control in Active Roles either when the operation is requested or when it is completed.

Use **Script Execution** policies to set up custom scripts (residing in **Script Modules** in the Active Roles Console) to:

- Trigger additional actions when performing directory object provisioning.
- Regulate object data format and requirements (such as for generating user passwords).
- Further automate administrative tasks.

Example use case for a Script Execution policy

Consider a scenario where employees of an organization are frequently transferred among its office branches temporarily due to various projects.

To administer such temporary assignments quickly and efficiently, write and apply a custom script that automatically reassigns the employee's user account from the OU of their original office to the OU of their new office, whenever their **City** or **Office Location** attributes are updated in Active Roles.

TIP: Consider the following when planning to use custom scripts for your provisioning policies:

- To help you configure **Script Execution** policies, Active Roles also ships with several built-in **Script Modules** that you can use to set up your own **Script Execution** policies. Find these built-in **Script Modules** in the following node of the Active Roles Console:

  Configuration > **Script Modules** > **Builtin**
- If the directory of your organization contains any cloud-only Azure users, then use the built-in **Generate User Password - Azure only** script module to set up a

password generation policy for cloud-only Azure users that meets the password strength criteria of both your organization and Microsoft Azure Active Directory (Azure AD).

For more information on configuring this Policy Object, see *Configuring a Script Execution policy* in the *Active Roles Administration Guide*.

About O365 and Azure Tenant Selection

O365 and Azure Tenant Selection policies help you:

- Manage Azure tenant selection for Azure users, guest users, groups and contacts.
- Microsoft 365 (M365) license and role selection.
- OneDrive provisioning for hybrid Azure users in the Azure tenant.

This is a unified policy for Azure AD-specific and Microsoft 365 management, controlled or restricted by creating a new provisioning policy and applying the policy to the Organizational Unit.

For more information on configuring this Policy Object, see *Configuring an O365 and Azure Tenant Selection policy* in the *Active Roles Administration Guide*.

About AutoProvisioning in SaaS products

AutoProvisioning in SaaS products policies automate the provisioning of users and groups in the selected SaaS products using Starling Connectors.

Active Roles uses this policy when creating users to provision access to the connected SaaS systems based on the Starling Connectors selected during the configuration of the policy.

For more information on configuring this Policy Object, see *Creating a provisioning policy for Starling Connect* in the *Active Roles Administration Guide*.

Overview of Deprovisioning Policy Objects

The Active Roles user interfaces, both Active Roles Console and Web Interface, provide the **Deprovision** command on user and group objects. This command initiates a request to deprovision the selected objects. When processing the request, Active Roles performs all operations that are set via the configured deprovisioning Policy Objects.

The **Deprovision** command is available in both the Active Roles Console and Web Interface. By using the **Deprovision** command, you start the deprovisioning operation on the objects you have selected.

The operation progress and results are displayed in the **Deprovisioning Results** window. When the operation is completed, the window displays the operation summary, and allows you to examine operation results in detail.

The left pane of the **Deprovisioning Results** window lists the objects that have been deprovisioned. The right pane displays the operation status and error messages, if any.

To view operation results, select an object in the left pane. The right pane shows a report on all actions taken during the deprovisioning of the selected object.

For more information on configuring deprovisioning policies, see *Configuring Policy Objects* in the *Active Roles Administration Guide*.

List of default built-in deprovisioning policy options

The following tables summarize the default deprovisioning policy options. If you do not add, remove, or change deprovisioning policies, Active Roles operates in accordance with these options when carrying out the **Deprovision** command on a user or group.

The following table summarizes the default deprovisioning policy options for users, defined by the **Built-in Policy - User Default Deprovisioning** Policy Object.

Table 7: Policy options for users: Built-in Policy - User Default Deprovisioning

Policy	Options
User Account Deprovisioning	• Disable the user account. • Set the user password to a random value. • Change the user name to include the suffix `deprovisioned` followed by the date when the user was deprovisioned. • Fill in the user description to state that this user account is deprovisioned. • Clear certain properties of the user account, such as city, company, and postal address.
Group Membership Removal	• Remove the user account from all security groups. • Remove the user account from all distribution groups.
Exchange Mailbox Deprovisioning	• Hide the user mailbox from Exchange address lists, thus preventing access to the mailbox.
Home Folder Deprovisioning	• Revoke access to the user home folder from the user account. • Give the user's manager read access to the user home folder. • Designate Administrators as the home folder owner.
User Account Relocation	• Do not move the user account from the Organizational Unit in which the account was located at the time of deprovisioning.
User Account Permanent Deletion	• Do not delete the user account.

The following table summarizes the default deprovisioning policy options for groups, defined by the **Built-in Policy - Group Default Deprovisioning** Policy Object.

Table 8: Policy options for groups: Built-in Policy - Group Default Deprovisioning

Policy	Options
Group Object Deprovisioning	• Change the group type from Security to Distribution. • Hide the group from the Global Address List (GAL). • Change the group name to include the suffix "deprovisioned" followed by the date when the group was deprovisioned. • Remove all members from the group. • Fill in the group description to state that this group is deprovisioned.
Group Object Relocation	• Do not move the group from the Organizational Unit in which the group was located at the time of deprovisioning.
Group Object Permanent Deletion	• Do not delete the group.

For more information on configuring deprovisioning Policy Objects, see *Configuring Policy Objects* in the *Active Roles Administration Guide*.

About User Account Deprovisioning

User Account Deprovisioning policies automate the following deprovisioning tasks on user accounts:

- Disabling the user account.
- Setting the user password to a random value.
- Setting the user login names to random values.
- Renaming the user account.
- Modifying other properties of the user account (for example, user membership in Managed Units).

After the deprovisioning of the user account is completed, the user will be unable to log in to the network with their credentials.

When processing a request to deprovision a user, Active Roles uses this policy to modify the user's account so that once the user has been deprovisioned, they cannot log on to the network.

A policy can also be configured to update user accounts. Depending on the policy configuration, each policy-based update results in the following:

- Certain portions of account information are removed from the directory by resetting specified properties to empty values.
- Certain properties of user accounts are set to new, non-empty values.

A policy can be configured so that new property values include:

- Properties of the user account being deprovisioned, retrieved from the directory prior to starting the process of the user deprovisioning.
- Properties of the user who originated the deprovisioning request.
- Date and time when the user was deprovisioned.

Thus, when deprovisioning a user, Active Roles modifies the user's account in Active Directory as determined by the User Account Deprovisioning policy that is in effect.

For more information on configuring this Policy Object, see *Configuring a User Account Deprovisioning policy* in the *Active Roles Administration Guide*.

About Group Membership Removal

Group Membership Removal policies automate the removal of deprovisioned user accounts from groups. You can configure such policies to remove user accounts from all groups, with optional exceptions.

When configuring a Group Membership Removal policy, you can configure policy rules separately for security groups and mail-enabled groups in the **New Deprovisioning Policy Object Wizard**.

When processing a request to deprovision a user, Active Roles uses this policy to determine what changes are to be made to group memberships of the user account. By removing the account from security groups, the policy revokes user access to resources. By removing the account from mail-enabled groups, the policy prevents erroneous situations where email is sent to the deprovisioned mailbox.

IMPORTANT: The deprovisioned users are automatically removed from all Dynamic Groups, regardless of the Group Membership Removal policy settings.

A Group Membership Removal policy includes separate rules for security groups and for mail-enabled groups. For each category of groups, a rule can instruct Active Roles to perform one of the actions that are summarized in the following table.

Table 9: Group Membership Removal policy includes separate rules

Category	Action	Result
Security groups	**Do not remove from groups.**	The deprovisioned user remains in all security groups it was a member of as of the time of deprovisioning, except for the Dynamic Groups.
	Remove from all groups.	The deprovisioned user is removed from all security groups.

Category	Action	Result
	Remove from all groups except for the specified ones.	The deprovisioned user is not removed from the specified security groups, with the exception of Dynamic Groups. The user is removed from all the other security groups.
Mail-enabled groups	**Do not remove from groups.**	The deprovisioned user is not removed from distribution groups or mail-enabled security groups, except for the Dynamic Groups.
	Remove from all groups.	The deprovisioned user is removed from all distribution groups and from all mail-enabled security groups.
	Remove from all groups except for the specified ones.	The deprovisioned user is not removed from the specified distribution or mail-enabled security groups, with the exception of Dynamic Groups. The user is removed from all the other distribution and mail-enabled security groups.

In the event of a conflict in policy implementation, the remove action takes precedence. For example, with a rule configured to remove the user account from all security groups, the user account is removed from all security groups even if there is another rule according to which Active Roles does not remove the user account from mail-enabled security groups.

Another conflict may occur in the situation where a policy of this category attempts to remove a deprovisioned user from a group that is configured as an Active Roles Dynamic Group. For more information, see *Dynamic groups* in the *Active Roles Administration Guide*.

The Dynamic Group policy detects the removal, and might add the deprovisioned user back to the Dynamic Group. To avoid this, Active Roles does not allow Dynamic Groups to hold deprovisioned users. Once a user is deprovisioned, the user account is removed from all Dynamic Groups.

For more information on configuring this Policy Object, see *Configuring a Group Membership Removal policy* in the *Active Roles Administration Guide*.

About User Account Relocation

User Account Relocation policies automate the movement of deprovisioned user accounts to specified Organizational Units. This removes deprovisioned user accounts from the control of administrators who are responsible for managing the Organizational Units in which those user accounts were originally located. However, you can also configure this policy not to move deprovisioned user accounts.

When processing a request to deprovision a user, Active Roles uses this policy to determine whether to move the deprovisioned user account to a different Organizational Unit.

A policy configured to move user accounts also specifies the destination Organizational Unit to which Active Roles moves deprovisioned user accounts.

A policy can be configured not to move user accounts. When applied at a certain level of the directory hierarchy, such a policy overrides any other policy of this category applied at a higher level of the directory hierarchy.

Let us consider an example to clarify this behavior. Suppose you configure a policy to move accounts and apply that policy to a certain parent container. In general, the policy is passed down from parent to child containers, that is, the policy applies to all child containers beneath the parent container, causing Active Roles to move deprovisioned user accounts from each container. However, if you configure a different policy not to move accounts and apply that new policy to a child container, the child container policy overrides the policy inherited from the parent container. Active Roles does not move deprovisioned user accounts from that child container or any container beneath that child container.

For more information on configuring this Policy Object, see *Configuring a User Account Relocation policy* in the *Active Roles Administration Guide*.

About Exchange Mailbox Deprovisioning

Exchange Mailbox Deprovisioning policies automate the deprovisioning of Microsoft Exchange resources for deprovisioned users. When activated, the policy:

- Hides deprovisioned users from address lists.
- Prevents sending non-delivery reports.
- Grants the designated persons full access to deprovisioned mailboxes.
- Redirects email messages sent to deprovisioned users.
- Forces the mailbox of the deprovisioned user to send automatic replies.

As such, the policy is meant to reduce the amount of email messages sent to the deprovisioned mailbox, and to authorize designated persons to monitor the mailbox of the deprovisioned user.

When processing a request to deprovision a user, Active Roles uses this policy to determine the Exchange mailbox deprovisioning options, and then updates the user account and mailbox accordingly.

The available mailbox-deprovisioning options are summarized in the following table. For each option, the table outlines the policy effect on a user mailbox.

Table 10: Policy effect on a user's mailbox

Option	Policy effect
Hide the mailbox from the Global Address List (GAL), to prevent access to the mailbox	Prevents the deprovisioned user from appearing in your Exchange organization's address lists. If you select this option, the deprovisioned user is hidden from all address lists. This option renders the mailbox inaccessible. You cannot log on to Exchange Server as the mailbox user or otherwise access the hidden mailbox.

Option	Policy effect
Prevent non-delivery reports (NDR) from being sent	Prevents non-delivery reports from being generated when emails are sent to the deprovisioned mailbox. (Non-delivery report is a notice that a message was not delivered to the recipient.)
Grant the user's manager full access to the mailbox	Provides the person designated as the deprovisioned user's manager with full access to the mailbox of that user. The manager is determined based on the **Manager** attribute of the deprovisioned user account in Active Directory.
Grant the selected users or groups full access to the mailbox	Provides the specified users or groups with full access to the deprovisioned user mailbox.
Disallow forwarding messages to alternate recipients	Email addressed to the deprovisioned user is not forwarded to an alternate recipient.
Forward all incoming messages to the user's manager	E-mail addressed to the deprovisioned user is forwarded to the user's manager. The manager is determined based on the **Manager** attribute of the deprovisioned user account in Active Directory.
Leave copies in the mailbox	Email addressed to the deprovisioned user is delivered to both the mailbox of the user's manager and the mailbox of the deprovisioned user. If you do not select this option, such email is only delivered to the manager's mailbox.
Don't change the mailbox autoreply settings	Active Roles makes no changes to the Automatic Replies configuration of the mailbox. Thus, if the mailbox is configured to send automatic replies, deprovisioning the mailbox user does not cause the mailbox to stop sending automatic replies.
Auto-reply with the following messages (once for each sender)	Active Roles configures the mailbox to send the Automatic Replies messages specified by the policy. This option provides for the following policy settings: • The Automatic Replies message that is sent to senders within the organization. • Whether to send an Automatic Replies message to senders outside of the organization (external senders). • Whether to send an Automatic Replies message to all external senders or only to the user's contacts. • The Automatic Replies message that is sent to external senders.

For more information on configuring this Policy Object, see *Exchange Mailbox Deprovisioning* in the *Active Roles Administration Guide*.

About Home Folder Deprovisioning

Home Folder Deprovisioning policies automate the following steps when deprovisioning users:

- Revoke access to home folders from deprovisioned user accounts.
- Grant the designated persons read access to deprovisioned home folders.
- Change the ownership on deprovisioned home folders.
- Delete the deprovisioned home folders.

When configuring Home Folder Deprovisioning policies, you can specify:

- How Active Roles must modify security on the deprovisioned user's home folder.
- Whether you want Active Roles to delete home folders when the user account is deleted.

When processing a request to deprovision a user, Active Roles uses this policy to determine the home folder deprovisioning options, and then updates the configuration of the user's home folder accordingly.

The available home folder deprovisioning options are summarized in the following table. For each option, the table outlines the policy effect on the user's home folder.

Table 11: Policy effect on the user's home folder

Option	Policy effect
Remove the user's permissions on the home folder	Modifies the home folder security so that the deprovisioned user cannot access their home folder.
Grant the user's manager read access to the home folder	Makes it possible for the person designated as the deprovisioned user's manager to view and retrieve data from the home folder of that user. The manager is determined based on the **Manager** attribute of the deprovisioned user account in Active Directory.
Grant selected users or groups read access to the home folder	Makes it possible for the specified users or groups to view and retrieve data from the deprovisioned user's home folder.
Make the selected user or group the owner of the	Designates the specified user or group as the owner of the deprovisioned user's home folder. The owner is authorized to control how permissions are set on the folder, and can grant permissions to others.

Option	Policy effect
home folder	
Delete the home folder when the user account is deleted	Upon the deletion of a user account, analyzes whether the user's home folder is empty, and then deletes or retains the home folder, depending on the policy configuration. A policy can be configured to only delete empty folders. Another option is to delete both empty and non-empty folders.

For more information on configuring this Policy Object, see *Configuring a Home Folder Deprovisioning policy* in the *Active Roles Administration Guide*.

About User Account Permanent Deletion

User Account Permanent Deletion policies automate the deletion of deprovisioned user accounts. Deprovisioned user accounts are retained for a specified amount of time before being permanently deleted. However, you can also configure the policy to only deprovision the user account, instead of deleting it after deprovisioning.

When processing a request to deprovision a user, Active Roles uses this policy to determine whether to schedule the deprovisioned user account for deletion. When scheduled for deletion, a user account is permanently deleted after a certain time period, referred to as a retention period.

A policy configured to delete user accounts specifies the number of days to retain deprovisioned user accounts. With such a policy, Active Roles permanently deletes a user account after the specified number of days has passed since the user was deprovisioned.

A policy can be configured not to delete user accounts. When applied at a certain level of the directory hierarchy, such a policy overrides any other policy of this category applied at a higher level of the directory hierarchy.

Let us consider an example to clarify this behavior. Suppose you configure a policy to delete accounts and apply that policy to a certain container. In general, the policy is passed down from parent to child containers, that is, the policy applies to all child containers beneath the parent container, causing Active Roles to delete deprovisioned user accounts in each container. However, if you configure a different policy not to delete accounts and apply that new policy to a child container, the child container policy overrides the policy inherited from the parent container. Active Roles does not delete deprovisioned user accounts in that child container or any container beneath that child container.

One more option of this policy is intended for domains where Active Directory Recycle Bin is enabled. The policy can be configured so that once a user account is deprovisioned, the account is moved to Recycle Bin (which effectively means that the account will be deleted immediately, without any retention period). Moving deprovisioned user accounts to the Recycle Bin may be required for security reasons, as an extra security precaution. The Active Directory Recycle Bin ensures that the account can be restored, if necessary, without any loss of data. Active Roles provides the ability to un-delete and then un-deprovision user accounts that were deprovisioned to the Recycle Bin.

For more information on configuring this Policy Object, see *Configuring a User Account Permanent Deletion policy* in the *Active Roles Administration Guide*.

About Office 365 Licenses Retention

Office 365 Licenses Retention policies automate the retention of all (or the selected) Microsoft 365 licenses assigned to an Azure AD user after successfully deprovisioning the Azure AD user.

When processing a request to deprovision an Azure AD user, Active Roles uses this policy to determine if the licenses assigned to the Azure AD user must be retained.

When an Azure AD User is deprovisioned, this policy ensures that the administrator-assigned Microsoft 365 licenses are retained based on the policy configuration.

You can configure the **Office 365 Licenses Retention** policy to specify how you want Active Roles to modify the Azure AD user's licenses in Azure AD upon a request to deprovision the Azure AD user.

When an Azure user is deprovisioned from the Active Roles Console, Web Interface, or Management Shell, the Microsoft 365 licenses that were assigned to the user during user provisioning are retained based on the **Office 365 Licenses Retention** policy configuration. As per the policy set, all the licenses or only selected licenses are retained upon the user deprovision.

The changes that take effect after deprovisioning the user are reflected in the Azure portal and the **Azure Properties** > **Licenses** tab of the Azure AD user in the Web Interface.

Active Roles Console enables you to create a new Deprovisioning Policy Object or add to the existing **Built-in Policy – User Default Deprovisioning** policy. The **Office 365 Licenses Retention** policy from the **User Deprovisioning Policies** must be selected to enable retention of the required Microsoft 365 licenses upon Azure AD user deprovisioning.

NOTE: The **Office 365 Licenses Retention** policy is enabled only if Azure AD is configured.

For more information on configuring this Policy Object, see *Configuring a Microsoft 365 license retention policy* in the *Active Roles Administration Guide*.

About Group Object Deprovisioning

Group Object Deprovisioning policies specify the changes within an organization that make group objects in Active Directory deprovisioned, preventing their use. When initiated, this policy deprovisions the affected group(s) by:

- Hiding the group(s) in the Global Address List (GAL) to prevent access to the group from Exchange Server client applications, such as Microsoft Outlook.
- Changing the type of the group(s) from **Security** to **Distribution** to revoke access rights from the group.
- Renaming the group(s) to clearly differentiate them from non-deprovisioned groups.

- Removing members from the group(s) to revoke user access to resources controlled by the group(s). However, you can also specify members who will not be removed from the group(s).

In addition, you can also configure the policy to change or clear any properties of a group, such as its pre-Windows 2000 name, email address(es), or description.

When processing a request to deprovision a group, Active Roles uses this policy to modify the group object in Active Directory, so that once the group has been deprovisioned, it cannot be used.

A policy can also be configured to update individual properties of groups. Depending on the policy configuration, each policy-based update results in the following:

- Certain portions of group information, such as information about group members, are removed from the directory.
- Certain properties of groups are changed or cleared.

A policy can be configured so that new property values include:

- Properties of the group being deprovisioned, retrieved from the directory prior to starting the process of the group deprovisioning.
- Properties of the user who originated the deprovisioning request.
- Date and time when the group was deprovisioned.

Thus, when deprovisioning a group, Active Roles modifies the group object in Active Directory as determined by the Group Object Deprovisioning policy that is in effect.

For more information on configuring this Policy Object, see *Configuring a Group Object Deprovisioning policy* in the *Active Roles Administration Guide*.

About Group Object Relocation

Group Object Relocation policies automate the movement of deprovisioned group objects to specified Organizational Units. This removes deprovisioned groups from the control of administrators who are responsible for managing the Organizational Units in which those groups were originally located. However, you can also configure this policy not to move deprovisioned groups.

When processing a request to deprovision a group, Active Roles uses this policy to determine whether to move the deprovisioned group object to a different Organizational Unit.

A policy configured to move group objects also specifies the destination Organizational Unit to which Active Roles moves deprovisioned group objects.

A policy can be configured not to move group objects. When applied at a certain level of the directory hierarchy, such a policy overrides any other policy of this category applied at a higher level of the directory hierarchy.

For more information on configuring this Policy Object, see *Configuring a Group Object Relocation policy* in the *Active Roles Administration Guide*.

About Group Object Permanent Deletion

Group Object Permanent Deletion policies automate the deletion of deprovisioned groups, with deprovisioned group objects retained for a specified amount of time before they are permanently deleted by Active Roles. Alternatively, you can also configure this policy not to delete, but to deprovision group objects instead.

When processing a request to deprovision a group, Active Roles uses this policy to determine whether to schedule the deprovisioned group object for deletion. When scheduled for deletion, a group object is permanently deleted after a certain time period, referred to as a retention period.

A policy configured to delete groups specifies the number of days to retain deprovisioned group objects. With such a policy, Active Roles permanently deletes a group after the specified number of days has passed since the group was deprovisioned.

A policy can be configured not to delete groups. When applied at a certain level of the directory hierarchy, such a policy overrides any other policy of this category applied at a higher level of the directory hierarchy.

One more option of this policy is intended for domains where Active Directory Recycle Bin is enabled. The policy can be configured so that once a group is deprovisioned, the group object is moved to the Recycle Bin (which effectively means that the group will be deleted immediately, without any retention period). Moving deprovisioned group objects to the Recycle Bin may be required for security reasons, as an extra security precaution. The Active Directory Recycle Bin ensures that the group object can be restored, if necessary, without any loss of data. Active Roles provides the ability to un-delete and then un-deprovision groups that were deprovisioned to the Recycle Bin.

For more information on configuring this Policy Object, see *Configuring a Group Object Permanent Deletion policy* in the *Active Roles Administration Guide*.

About Notification Distribution

Notification Distribution policies automatically generate and send email notifications about deprovisioning requests. As such, these policies are meant to notify designated persons about object deprovisioning requests, so that they can take additional deprovisioning-related actions on that object, if necessary.

When configuring a Notification Distribution policy, you can:

- Specify the list of notification recipients.
- Customize the subject and body of the notification message.

When processing a deprovisioning request, Active Roles uses this policy to determine whether anyone must be notified of the deprovisioning operation that is requested. Then, it generates a notification message and sends it to the recipients, if any specified in the policy configuration.

When a deprovisioning operation is requested, Active Roles issues a notification message regardless of operation results. Hence, a notification message cannot be considered as an indication of success or failure of the operation. Rather, it only indicates that deprovisioning

has been requested. If you need to inform anybody of deprovisioning results, use a Report Distribution policy.

Notification performs on a per-object basis: Each notification message contains information about a request to deprovision one object. When deprovisioning multiple objects, Active Roles sends multiple notification messages, one message per object.

Active Roles sends notification messages via an SMTP server. The policy configuration specifies the outbound SMTP server by using Active Roles email settings that include the name of the SMTP server and information required to connect to the SMTP server.

For more information on configuring this Policy Object, see *Configuring a Report Distribution policy* in the *Active Roles Administration Guide*.

About Report Distribution

Report Distribution policies automatically send a report on deprovisioning results after completing a deprovisioning operation. As such, this policy is meant to inform designated persons about any problems that might occur when performing deprovisioning requests.

Reports are delivered via email. When configuring a Report Distribution policy, you can set up a list of report recipients, customize the subject of report messages, and specify whether to send a report if no errors occurred.

Upon completion of a deprovisioning operation, Active Roles uses this policy to determine if the report on deprovisioning results must be sent. Then, Active Roles generates the report message and sends it to the recipients specified in the policy configuration. The report includes a list of actions taken during the deprovisioning operation. For each action, the report informs of whether the action is completed successfully, and provides information about the action results.

With the Report Distribution policy configured not to send reports if no errors occurred, Active Roles examines the deprovisioning results for errors. If there are no errors, the report is not sent.

Active Roles generates deprovisioning reports on a per-object basis: Each report message contains information on the deprovisioning of one object. When deprovisioning multiple objects, Active Roles sends multiple report messages, one message per deprovisioned object.

Active Roles sends report messages via an SMTP server. The policy configuration specifies the outbound SMTP server by using Active Roles email settings that include the name of the SMTP server and information required to connect to the SMTP server.

For more information on configuring this Policy Object, see *Configuring a Report Distribution policy* in the *Active Roles Administration Guide*.

About the Undo Deprovisioning operation

Active Roles provides the ability to restore deprovisioned objects, such as deprovisioned users or groups. The purpose of this operation, referred to as the Undo Deprovisioning

operation, is to roll back the changes that were made to an object by the Deprovision operation. When a deprovisioned object needs to be restored (for example, if a user account has been deprovisioned by mistake), the Undo Deprovisioning operation allows the object to be quickly returned to the state it was in before the changes were made.

The Undo Deprovisioning operation rolls back the changes that were made to the object in accord with the standard Deprovisioning policies. For example, assume a User Account Deprovisioning policy is configured so that a deprovisioned user account:

- Is disabled.
- Is renamed.
- Has the Description changed.
- Has a number of properties cleared out.
- Has the password set to a random value.

In this case, the Undo Deprovisioning operation:

- Enables the user account.
- Sets the Description, Name, and other properties to the original values on the user account.
- Can provide the option to reset the password so as to enable the user to log on.

Similar behavior is in effect for the other policies of the Deprovisioning category:

- If the Deprovision operation revokes user access to resources such as the home folder or Exchange mailbox, then the Undo Deprovisioning operation attempts to restore user access to the resources.
- If the Deprovision operation removes a user account from certain groups, the Undo Deprovisioning operation can add the user account to those groups, restoring the original group memberships of the user account.

To offer another example, suppose the deprovisioning policy is configured so that Deprovision operation on a group:

- Removes all members from the group
- Renames the group
- Moves the group to a certain container

In this case, the Undo Deprovisioning operation:

- Restores the original membership list of the group, as it was at the time of deprovisioning
- Renames the group, restoring the original name of the group
- Moves the group to the container that held the group at the time of deprovisioning

Similar behavior is in effect for the other group deprovisioning policy options:

- If the Deprovision operation hides the group from the Global Address List (GAL), Undo Deprovisioning restores the visibility of the group in the GAL.

- If the Deprovision operation changes the group type from Security to Distribution, Undo Deprovisioning sets the group type back to Security.
- If the Deprovision operation changes any other properties of the group, Undo Deprovisioning restores the original property values.

Both the Active RolesConsole and Web Interface provide the Undo Deprovisioning command on deprovisioned users or groups. When selected on a deprovisioned object, this command originates a request to restore the object. Upon receipt of the request, Active Roles performs all necessary actions to undo the results of deprovisioning on the object, and provides a detailed report of the actions that were taken along with information about success or failure of each action.

For more information on the procedure of restoring deprovisioned objects, see *Restoring deprovisioned users or groups* in the *Active Roles Administration Guide*.

About Container Deletion Prevention

If you select and delete a container object that has child objects (for example, an Organizational Unit), you perform a bulk deletion.

While bulk deletions are rare, they can be still disruptive operations. To prevent accidental bulk deletion of your directory objects, Active Roles has a Container Deletion Prevention policy that you can configure in your organization.

By default, Active Roles performs bulk deletion on a container in the following way:

1. Active Roles builds a list of all the child objects found in the container.
2. Active Roles starts deleting the listed objects one by one.
3. For every object in the list, Active Roles performs an access check to determine if the user or process that requested the deletion has the required permissions to delete the object. If the access check allows the deletion, then the object is deleted; otherwise, Active Roles does not delete the object, and continues the process with the next object in the list.
4. Finally, once all child objects are deleted, Active Roles deletes the container itself. If any child objects are not deleted, Active Roles will not delete the container either.

While the permission check can prevent accidental deletion in many scenarios, administrators with full control over an Organizational Unit in Active Roles can still delete the entire OU with a single delete operation. To prevent this, Active Roles provides the Container Deletion Prevention policy, denying the deletion of non-empty containers.

NOTE: The Container Deletion Prevention policy defines a configurable list of object types as specified by the Active Directory schema (for example, the Organizational Unit object type). If an Active Roles client requests the deletion of a particular container, the Active Roles Administration Service evaluates the request to determine if the type of the container is in the list of the configured policy. If the container type is in the list and the container holds any objects, the Active Roles Administration Service denies the request, preventing the deletion of the container. Because of this, even if you otherwise have permissions to delete containers, first you must delete all objects in the container before

deleting the container itself.

For more information on configuring a Container deletion prevention policy, see *Configuring a Container Deletion Prevention policy* in the *Active Roles Administration Guide*.

About the Protect container from accidental deletion option

You can also protect Organizational Units against accidental deletion by using the **Protect container from accidental deletion** option of the Active Roles Console or Web Interface when creating or modifying an OU in your managed Active Directory domains or Active DirectoryLightweight Directory Services (AD LDS) partitions.

This **Protect container from accidental deletion** option removes the **Delete** and **Delete Subtree** permissions on the OU, along with the **Delete All Child Objects** permission on the parent container of the OU. An OU created with this option cannot be deleted, regardless of whether you use Active Roles or other system-provided Active Directory administration tools, as the option removes the deletion-related permissions by applying the appropriate Access Templates in Active Roles, then replicating the resulting permission entries to Active Directory.

The option to protect existing OUs or other objects from deletion is available on the **Properties** > **Object** tab for an object in the Active Roles Console or Web Interface. If you select the **Protect object from accidental deletion** check box on that tab, Active Roles configures the permission entries on the object in the same way as with the **Protect container from accidental deletion** option for an Organizational Unit. After that, attempting to delete a protected object will result in the operation returning an error, indicating that the object is protected or access is denied.

Protecting objects from deletion this way adds the following Access Template links:

- On the object to protect, Active Roles adds a link to the **Objects - Deny Deletion** Access Template for the **Everyone** group.
- On the parent container of the object, Active Roles adds a link to the **Objects - Deny Deletion of Child Objects** Access Template for the **Everyone** group. (Active Roles does not add this link if it detects that a link of the same configuration already exists.)

The links are configured to apply the Access Template permission entries not only in Active Roles, but also in Active Directory. This adds the following access control entries (ACEs) in Active Directory:

- On the object to protect, Active Roles adds explicit Deny ACEs for the **Delete** and **Delete Subtree** permissions for the **Everyone** group.
- On the parent container of the object, adds an explicit Deny ACE for the **Delete All Child Objects** permission for the **Everyone** group. However, Active Roles does not add this ACE if it detects that an ACE of the same configuration already exists.

If you clear the **Protect object from accidental deletion** check box for a given object, Active Roles updates the object to remove the link to the **Objects - Deny Deletion** Access

Template in Active Roles, along with the explicit Deny ACEs for the **Delete** and **Delete Subtree** permissions for the **Everyone** group in Active Directory. As a result, the object is no longer guarded against deletion.

NOTE: Clearing the **Protect object from accidental deletion** check box for a specific object removes the Access Template links and ACEs only from that object, leaving the Access Template links and ACEs on the parent container intact. This is because the parent container can hold other objects that are protected from deletion. If the container does not hold any protected objects, you can remove the link to the **Objects - Deny Deletion of Child Objects** Access Template by using the **Delegate Control** command on that container in the Active RolesConsole, which will also delete the corresponding ACE in Active Directory.

TIP: You can configure Active Roles so that the **Protect container from accidental deletion** check box will be selected by default on the pages for creating Organizational Units in the Active RolesConsole or Web Interface.

To enable this behavior within a domain or container, apply the **Built-in Policy - Set Option to Protect OU from Deletion** Policy Object to that domain or container. This Policy Object ensures that Organizational Units created by Active Roles are protected from deletion regardless of the method used to create them. As such, Organizational Units created using Active Roles script interfaces will also be protected by default.

About picture management rules

You can use the Active Roles Console or Web Interface to add a picture for a user, group, or contact object.

Using pictures (such as photographs or logos) is generally recommended, as they make it easier to recognize the user, group, or contact in email clients and web applications that can retrieve the picture from Active Directory. When you supply a picture for a user, group or contact via Active Roles, the picture is saved in the `thumbnailPhoto` attribute of that user, contact, or group in Active Directory.

Active Roles provides a policy to enforce the picture size limits, including maximum and minimum dimensions and the option to resize the picture automatically. When you add a picture to the user, group, or contact, Active Roles checks the dimensions of the picture, and does not apply the picture if it does not meet the policy settings. If automatic picture resizing is enabled, Active Roles reduces the dimensions of the picture as needed by downsampling the original picture.

For more information on viewing or configuring picture management rules, see *Configuring picture management rules* in the *Active Roles Administration Guide*.

About policy extension using custom Policy Types

By default, you can configure predefined Policy Types that are installed with Active Roles. These Policy Types are available via the Active Roles Console, and include Policy Types such as **Home Folder AutoProvisioning** or **User Account Deprovisioning**. However, you can extend the default list by adding new, custom Policy Types.

Each Policy Type determines:

- A certain policy action (for example, creating a home folder for a user account).
- A collection of policy parameters to configure the policy action (for example, parameters that specify the network location where to create home folders).

Active Roles supports creating custom policies based on the **Script Execution** built-in Policy Type. However, creating and configuring a script policy from scratch can be time-consuming. Custom Policy Types provide a way to mitigate this overhead. Once a custom Policy Type is deployed that points to a particular script, administrators can easily configure and apply policies of that type, having those policies perform the actions determined by the script. The policy script also defines the policy parameters specific to the Policy Type.

Custom Policy Types provide an extensible mechanism for deploying custom policies. This feature is implemented by using the Policy Type object class. You can create Policy Types via the Active Roles Console, with each object representing a specific custom Policy Type.

For more information on changing, exporting or deleting Policy Types, see *Creating and managing custom Policy Types* in the *Active Roles Administration Guide*.

Main steps of policy extension

Implementing custom policy extensions has two main steps:

- Deploying Policy Types.
- Using Policy Types.

Deploying Policy Types

Deploying a custom Policy Type includes:

1. Developing a script that implements the policy action and declares the policy parameters.
2. Creating a **Script Module** containing the script.
3. Creating the Policy Type object referring to that **Script Module**.

Alternatively, to deploy a Policy Type to a different environment, you can:

1. Export the Policy Type to an export file in the source environment.
2. Import the file in the destination environment.

TIP: Exporting custom Policy Types makes it easy to distribute them throughout your organization.

For more information, see the following resources:

- For details on how to script Policy Type objects, refer to the Active Roles SDK.
- For the steps of exporting and importing custom Policy Types, see *Exporting Policy Type* and *Importing Policy Type* in the *Active Roles Administration Guide*.

TIP: One Identity recommends developing custom Policy Types in a separate environment, then exporting the final Policy Type for use.

Using Policy Types

Using the custom Policy Types means that you configure a new Policy Object that will use the custom Policy Types, or add the custom policies to an existing Policy Object.

For example, the **New Provisioning Policy Object Wizard** and **New Deprovisoning Policy Object Wizard** both have a **Policy to Configure** page for selecting a policy. By default, this page lists the built-in Policy Types shipped with Active Roles, but once you have custom Policy Types created, they will appear in this list, too.

If you select a custom Policy Type, the wizard provides a page for configuring the policy parameters specific to that Policy Type. After you complete the wizard, the Policy Object contains a fully functional policy of the selected custom Policy Type.

Active Roles provides a graphical user interface, complete with a programming interface, for creating and managing custom Policy Types. Using those interfaces, you can extend Active Roles policies to meet the needs of a particular environment. Active Roles also has a deployment mechanism that you can use to roll out new Policy Types.

For the steps of configuring Policy Objects with custom Policy Types, see *Creating a Policy Type object* in the *Active Roles Administration Guide*.

Active Roles interfaces to manage custom Policy Types

When using custom Policy Types, the various Active Roles components have the following roles in storing, maintaining and exposing the custom Policy Types:

- The Administration Service maintains Policy Type definitions, exposing Policy Types to its clients such as the Active RolesConsole or ADSI Provider.
- The Active Roles Console supports:
    - Creating a new custom Policy Type, either from scratch or by importing a Policy Type that was exported from another environment.
    - Modifying existing custom Policy Types.
    - Adding a policy of a particular custom type to a Policy Object, making the

necessary changes to the policy parameters provided for by the Policy Type definition.

Main attributes of policy extension

Policy extension is based on custom Policy Types, each of which represents a single type of policy.

When deploying a new custom policy, you must create a new Policy Type object. Then, when adding the custom policy to a Policy Object, Active Roles retrieves the definition of the custom policy from the respective custom Policy Type.

Policy types have the following attributes to specify the properties of custom policies:

- **Display name**: Identifies the Policy Type. This name appears in the **New Provisioning Policy Object Wizard** and **New Deprovisoning Policy Object Wizard** when you select the policy to configure, or adding a policy to an existing Policy Object.
- **Description**: Describes the Policy Type. This text appears in the **New Provisioning Policy Object Wizard** and **New Deprovisoning Policy Object Wizard** when you select the policy to configure, or adding a policy to an existing Policy Object.
- **Reference to Script Module**: Identifies the script to run when initiating the Policy Type. When adding a policy of a custom Policy Type, you effectively create a policy that runs the script from the **Script Module** specified by the respective Policy Type.
- **Policy Type category**: Identifies the Policy Object category to which you can add the Policy Type. A Policy Type can be either **Provisioning** or **Deprovisioning**, allowing policies of that type to be added either to provisioning or deprovisioning Policy Objects, respectively.
- **Function to declare parameters**: Identifies the name of the script function that declares the configurable parameters of the administration policy that is based on the Policy Type. This script function must exist in the **Script Module** selected for the Policy Type. By default, Active Roles expects that the parameters are declared by the `onInit` function.
- **Policy Type icon**: The image that appears next to the display name of the Policy Type on the wizard page where you select a policy to configure, to help identify and visually distinguish this Policy Type from the other Policy Types.

To create a custom policy, you must:

1. Create a **Script Module** that will hold the policy script.
2. Create the Policy Type referring to that **Script Module**.
3. Add the custom Policy Type to a Policy Object.

If you import a Policy Type, Active Roles automatically creates both the **Script Module** and the Policy Type.

For the steps of configuring Policy Objects with custom Policy Types, see *Creating a Policy Type object* in the *Active Roles Administration Guide*.

3

Configuring and administering Active Roles

This section summarizes the major configuration, deployment, and maintenance features of Active Roles.

About the Active Roles Setup wizard

The Active Roles Setup wizard facilitates the evaluation, deployment, upgrade and configuration of Active Roles. The key highlights of the wizard include the following:

- **Unified setup process**: Active Roles is shipped with a single wizard for installing all core product components, including the Administration Service, the Web Interface, and the Console (also known as the MMC Interface).
- **Configuration Center**: After installation, Active Roles launches the Configuration Center, an application that you can use to perform the core configuration tasks after installation, or to finish upgrading Active Roles. As such, the Configuration Center lets you configure Administration Service instances and deploy Web Interface sites. For more information on the Configuration Center, see About Active Roles Configuration Center.
- **Side-by-side deployment**: The Active Roles Setup allows you to deploy new Active Roles versions side-by-side on the same computers with Active Roles 6.9. This allows you to use the same hardware and infrastructure to run newer versions of Active Roles while also keeping Active Roles 6.9 deployed for your business needs.

CAUTION: Upgrading from Active Roles 6.9 to a newer version is only meant to be a temporary solution, as the side-by-side installation of two different Active Roles versions can have a negative impact on the environment.

Different versions of Active Roles are not supported in the same Active Directory (AD) domain. Different versions of Active Roles servers in the same AD domain will cause issues with dynamic groups, policies, workflows, custom scripts, and conflicts in product functionality.

When upgrading Active Roles to a later version, One Identity recommends to upgrade all servers running Active Roles components to the same version, otherwise the configuration is not supported.

For more information, see Knowledge Base Article 4307177.

NOTE: To avoid potential conflicts with Active Roles 6.9, newer versions of the product use a different name for the Windows service of the Administration Service and for the default Web Interface sites.

- **Separate component installation files**: Although the Active Roles Setup allows you to install every major product component at once, the installation `*.iso` delivers each component (such as the Administration Service, the Web Interface, the Add-on Manager, the SPML Provider, or the Management Shell) in separate `*.msi` files. This allows you to install the various Active Roles components individually without the need of running the Active Roles Setup.

About Active Roles Configuration Center

The Active Roles Configuration Center is a configuration application that provides a unified configuration platform for the Active Roles Administration Service and the Web Interface component. This allows administrators to perform the core Active Roles configuration tasks from a single application, including the following:

- Performing the initial configuration of Active Roles, such as setting up the Administration Service instances and the default Web Interface sites.
- Importing the configuration database and the management history database from earlier Active Roles versions.
- Managing the core Administration Service resources, such as the Active Roles Admin account, service account, and database connections.
- Creating new Web Interface sites either based on the site configuration objects of the current Active Roles version, or by importing site configuration objects from earlier Active Roles versions.
- Managing core Web Interface site settings, such as site addresses on the web server, or the configuration object in the Administration Service.
- Configuring secure communication for the Active Roles Web Interface through forced SSL redirection.

- Integrating Active Roles with One Identity Starling. For more information, see *One Identity Starling Join and configuration through Active Roles* in the *Active Roles Administration Guide*.
- Managing user login settings for the Active Roles Console (also known as the MMC Interface).
- Configuring federated authentication, allowing you to access an application or website by authenticating against a certain set of rules, known as "claims".
- Configuring log management and Solution Intelligence.

For more information on these features, see the following subsections.

Getting Started

Active Roles Configuration Center is automatically installed and started by default if you select to install either the Administration Service or the Web Interface components to a computer. Later, you can start Configuration Center again either from the Windows Start menu, or from the Apps page of the operating system.

About Configuration Center components

The Configuration Center provides a unified, single, simple, wizard-based user interface for all core Active Roles configuration tasks, making it a single point of access to all management wizards for all configuration tasks.

The Configuration Center consists of the following elements.

Initial configuration wizards

After installing Active Roles, the Configuration Center allows administrators to run the initial configuration wizards and create the new Active Roles instance, including the Administration Service and the Web Interface.

Hub pages and management wizards

Once the initial configuration is completed, the Configuration Center provides a consolidated view of the core Active Roles configuration settings, and offers tools for changing those settings.

The hub pages of the Configuration Center show the current settings specific to the Administration Service and the Web Interface, including the commands to start the management wizards for changing those settings. The available hub pages are the following:

- **Administration Service**: This page allows administrators to:
  - View or change the Active Roles Admin account, service account, and databases.
  - Import the configuration data and management history data either from an earlier Active Roles version or from the current Active Roles database.
  - View status information, such as whether the Administration Service is started and ready for use, stopped, or being restarted (along with the options to start, stop and restart the service).
- **Web Interface**: This page allows administrators to:
  - View, create, modify or delete Web Interface sites. The configurable site settings include the site address, and the configuration object that stores the site configuration data in the Administration Service.

    When creating or modifying a Web Interface site, administrators can either reuse an existing configuration object, or create a new one based on a template or by importing data from another configuration object or from an export file.
  - Export the configuration of any existing Web Interface site to a file.
  - Open each site in a web browser.

Configuration Shell

The `ActiveRolesConfiguration` module (also known as the Configuration Shell) of the Active Roles Management Shell allows administrators to access all Configuration Center features and functions from a Windows PowerShell command-line interface or with scripts, facilitating the unattended configuration of Active Roles components. The `ActiveRolesConfiguration` module provides cmdlets for key configuration tasks, such as:

- Creating the Active Roles database.
- Creating or modifying the Administration Service instances and the Web Interface sites.
- Performing data exchange between Active Roles databases and between site configuration objects.
- Querying the current state of the Administration Service.
- Starting, stopping or restarting the Administration Service.

Configuration of a local or remote Active Roles instance

Configuration Center is installed as part of the Management Tools component if you install Active Roles on a 64-bit system. You can use the Management Tools package to perform configuration tasks on the local or remote computer that has the current version of the Administration Service or Web Interface installed.

Once installed, the Configuration Center looks for these components on the local computer, and if it does not find any of these components, it prompts you to connect to a remote computer. However, you can also connect to a remote computer by clicking the drop-down menu in the Configuration Center header.

NOTE: Consider the following when planning to use the Configuration Center on a remote computer:

- When connecting to a remote computer, Configuration Center prompts you for a user name and password. The account you use to log in must match the domain user account belonging to the Administrators group on the remote computer. In addition, whether you are going to perform configuration tasks on the local computer or on a remote computer, your login account must be a member of the Administrators group on the computer running Configuration Center.
- To perform configuration tasks on a remote computer, Configuration Center requires Windows PowerShell remoting to be enabled on that computer. PowerShell remoting is enabled by default on Microsoft Windows Server 2016 or newer operating systems; however, if it is turned off for any reason on the remote computer, you can enable it by running the **`Enable-PSRemoting`** command in Windows PowerShell. For more information, see Enable-PSRemoting in the *Microsoft PowerShell documentation*.

About running the Configuration Center

The Configuration Center is installed and, by default, automatically started after installing the Active Roles Administration Service or Web Interface component on a computer, allowing you to perform the initial configuration tasks for these components. If you close the Configuration Center, you can start it again later from the Windows Start menu or the Apps page of the operating system.

As the Configuration Center can manage Active Roles not only on the local computer but also on remote computers, you can run it both on client and server operating systems. However, you can only install the Configuration Center on a 64-bit operating system. Once the component is installed on a client operating system, you must start and connect it to the remote server where the Administration Service or Web Interface instances you want to configure are installed. Similarly to a server operating system, you can launch the Active Roles Configuration Center either from the Windows Start menu or from the Apps page.

NOTE: To run the Configuration Center on a client computer, you must be logged in with Administrator privileges.

If neither the Administration Service nor the Web Interface is installed on the local computer, the Configuration Center will prompt you to select a remote computer. In the **Select Server** dialog that appears, supply the fully qualified domain name of a server on which the Administration Service or the Web Interface instance is installed, then enter the name and password of a domain user account that has administrator rights on that server. You can connect to a remote server at any time by clicking the **Connect to another server** option in the header of the Active Roles Configuration Center.

Supported Configuration Center tasks

The Configuration Center lets administrators perform:

- Initial configuration tasks, such as creating the Administration Service instance and the default Web Interface sites. For more information, see Initial configuration tasks.
- Configuration management tasks, that is managing existing Administration Service and Web Interface instances. For more information, see Configuration management tasks.
- Managing user access to the Active Roles Console. For more information, see Delegation of user access to the Active Roles Console.
- Managing the logging settings of Active Roles. For more information, see Configuration of Active Roles logging settings.
- Configuring Solution Intelligence. For more information, see Configuration of Solution Intelligence.

Initial configuration tasks

Once the Active Roles Setup wizard installs Active Roles, the Configuration Center starts automatically so that administrators can create an Administration Service instance and deploy the default Web Interface sites. The following sections describe these tasks in detail.

Configuration of the Administration Service

The **Configure Administration Service** wizard creates the Administration Service instance, preparing it for use. The wizard needs the following data for configuration:

- The login name and password of the account in which the configured Administration Service instance will be running (service account). In case of a Group Managed Service account, you must specify the service account details.
- The name of the group or user account that will have full access to all Active Roles features and functions through the configured Administration Service instance. This group or account is known as the Active Roles Admin.
- The database in which the configured Administration Service instance will store the configuration data and management history data. When specifying the database, you can either create a new database, or use an existing database compatible with the current Active Roles version. You can use the same database for multiple Administration Service instances.
- The authentication mode that the configured Administration Service instance will use when connecting to the database:
    - When using **Windows authentication**, the Administration Service will use the credentials of the service account.

When using **SQL Server authentication**, the Administration Service will use the SQL login name and password you specify in the wizard.

To start the wizard, in the **Administration Service** tab, click **Configure**.

Configuration of the Web Interface

The **Configure Web Interface** wizard creates the default Web Interface sites, getting the Web Interface component ready for use. The wizard prompts you to choose which Administration Service instance will be used by the Web Interface instance you are configuring. The Web Interface can:

- Use the Administration Service instance running on the same computer as the Web Interface.
- Use an Administration Service instance running on a different computer. In this case, you must supply the fully qualified domain name of the computer running the preferred instance of the Administration Service.
- Let the Web Interface choose any Administration Service instance that has the same configuration as the specified one. In this case, you must supply the fully qualified domain name of the computer running the Administration Service instance of the desired configuration.

  NOTE: If your environment uses Active Roles replication, you must specify the computer running the Administration Service instance whose database server acts as the Publisher of the Active Roles configuration database.

You can access the **Configure Web Interface** wizard from the **Configure** > **Web Interface** menu of the Configuration Center **Dashboard**.

After configuring the Web Interface, you can perform the following additional Web Interface configuration steps in the Configuration Center:

- **Forcing SSL redirection**: By default, Active Roles users can connect to the configured Web Interface sites via HTTP protocol that does not encrypt data during communication. To enable secure communication for the Web Interface on local and remote servers, One Identity recommends enabling the HTTPS protocol with the **Force SSL Redirection** option.
- **Federated authentication**: You can authenticate the Web Interface sites against a certain set of rules (known as "claims"), by using the federated authentication. The implementation in Active Roles uses Security Assertion Markup Language (SAML), through which you can sign in to an application via single sign-on, then authenticate to access the configured Web Interface sites. For more information, see *Working with federated authentication* in the *Active Roles Administration Guide*.

About Starling Join

Active Roles supports integration with One Identity Starling via the Starling Join feature. Joining Active Roles to Starling enables access to the various Starling services, including Identity Analytics and Risk Intelligence, and Connect. For more information, see *One*

Identity Starling Join and configuration through Active Roles in the *Active Roles Administration Guide*.

Configuration management tasks

Once you completed the initial configuration of Active Roles in the Configuration Center as described in Initial configuration tasks, you can check the state of the Administration Service and Web Interface components anytime, and can also perform various management tasks on them. The following sections describe these tasks in detail.

Administration Service management tasks

After installing Active Roles, first you must create the Administration Service instance as described in Configuration of the Administration Service. Then, you can use the Configuration Center to:

- View or change the core Administration Service settings, such as the Active Roles Admin account, Active Roles service account, and the Active Roles databases. For more information, see Core Administration Service settings.
- Import configuration data from another (current version or earlier version) Active Roles database to the current database of the Administration Service. For more information, see Configuration data import.
- Import Management History data from another (current version or earlier version) Active Roles database to the current database of the Administration Service. For more information, see Management History data import.
- Check the state of the Administration Service. For more information, see Administration Service states.
- Start, stop or restart the Administration Service by clicking **Start**, **Stop** or **Restart** at the top of the **Administration Service** page in the Configuration Center main window.

Core Administration Service settings

On the **Administration Service** page of the Configuration Center, you can check:

- The logon name of the service account.
- The name of the group or user account that has the Active Roles Admin rights.
- The SQL Server instance that hosts the Active Roles Configuration database.
- The name of the Active Roles Configuration database.
- The Configuration database connection authentication mode (Windows authentication or SQL Server login).
- The SQL Server instance that hosts the Active RolesManagement History database.
- The name of the Active RolesManagement History database.

- The Management History database connection authentication mode (Windows authentication or SQL Server login).

From the **Administration Service** page in the Configuration Center main window, you can change:

- The service account.

  Click **Change** in the **Service account** area. In the wizard that appears, supply the logon name and password of the domain user account in which you want the Administration Service to run.

- The Active Roles Admin account.

  Click **Change** in the **Active Roles Admin** area. In the wizard that appears, specify the group or user account you want to have the Active Roles Admin rights.

- The Active Roles database.

  Click **Change** in the **Active Roles database** area. In the wizard that appears, specify the database type and the database server instance and the database you want the Administration Service to use, and choose the database connection authentication mode (Windows authentication or SQL Server login). You have the option to specify a separate database for storing management history data.

  NOTE: Azure Databases can be connected only using SQL Server authentication.

Configuration data import

When deploying the Administration Service, you might need to import configuration data from an existing database to ensure that the new Administration Service instance has the same configuration as the existing one. Importing configuration data to a newly created database instead of attaching the Administration Service to an existing database is necessary if the version of the Administration Service you are deploying is greater than the version of the database you want to use. Some examples of such a situation are the following:

- Upgrading the Administration Service while preserving its configuration.
- Restoring configuration data from a backup copy of the database whose version does not match the version of the Administration Service.

When upgrading the Administration Service, you must import configuration data from the earlier version of Active Roles to the new version of the product. To do so, in the Configuration Center, click **Administration Service** > **Import Configuration**, then follow the steps in the wizard that appears. For more information, see *Importing configuration data* in the *Active Roles Installation Guide* or *Active Roles Upgrade Guide*.

During the import operation, the wizard retrieves and upgrades the data from the source database, and replaces the data in the destination database with the upgraded data from the source database.

You can script the configuration operations available in the Configuration Center using the Windows PowerShell command-line tools of the Active Roles Management Shell. For more information, see About Active Roles Management Shell.

Management History data import

After configuring the Administration Service, the Management History data storage will be empty with the option to create a new database. During the import of configuration data, the Configuration Center transfers only the administrative right assignments, policy definitions, administrative view settings, workflow definitions and other parameters that determine the Active Roles work environment. Management History data is excluded from the import operation to reduce the time it takes to upgrade the configuration of the Administration Service.

The Management History data describes changes that were made to directory data via Active Roles. This includes information about directory data management tasks, such as:

- The changes a user performed.
- The users performing the changes.
- The time the change was performed.

The Management History data is used for change history and user activity reports. In addition, the Management History data storage holds information about various tasks related to approval workflows and temporal group memberships.

After configuring the Administration Service and importing configuration data from an existing database, you must take additional steps to transfer the Management History data. You can do this using the **Import Management History** wizard in the Active Roles Configuration Center.

Importing Management History data is similar to the task of importing configuration data. However, there are some important differences:

- Due to a much larger volume of Management History data compared to configuration data, importing Management History data might take longer.
- As Management History data has dependencies on configuration data, you must import the configuration data first. You can import the Management History data after that, if needed.

Because of these considerations, Configuration Center provides a different wizard for importing Management History. The main features of the **Import Management History** wizard are the following:

- The wizard does not replace the existing data in the destination database. It only retrieves and upgrades Management History records from the source database, then adds the upgraded records to the destination database.
- The wizard allows you to specify the date range for the Management History records you want to import, so you can import only records that occurred within a particular timeframe instead of importing all records at a time.
- Canceling the wizard while the import operation is in progress does not cause you to lose the import results, so you can stop the import operation at any time. The records imported by the time that you cancel the wizard are retained in the destination database. If you start the wizard again, the wizard only imports records that were not imported earlier.

- Optionally, you can also synchronize SQL Server users and login data, but only if the destination database is also hosted on an SQL Server. Synchronizing users between SQL Server and Azure SQL is not supported.

For more information, see *Importing Management History data* in the *Active Roles Installation Guide* or *Active Roles Upgrade Guide*.

Administration Service states

The **Administration Service** page of the Active Roles Configuration Center indicates the states of the service with the following status labels:

- **Ready for use**: Administration Service is running and is ready to process requests.
- **Getting ready**: Administration Service just started and is preparing to process client requests.
- **Stopping**: Administration Service is preparing to stop.
- **Stopped**: Administration Service is not running.
- **Unknown**: Configuration Center cannot check the state of Administration Service.

Web Interface management tasks

After installing Active Roles, you can perform the initial configuration of the Web Interface in the Configuration Center, preparing the component for use. Then, you can use the Configuration Center to:

- Identify the Web Interface sites currently deployed on the web server running the Web Interface. For more information, see Identify the Web Interface sites.
- Create, modify or delete Web Interface sites. For more information, see:
  - Create a Web Interface site
  - Modify a Web Interface site
  - Delete a Web Interface site
- Export the configuration object of a Web Interface site to a file. For more information, see Export the configuration of a Web Interface site to a file

Identify the Web Interface sites

You can use the **Web Interface** page of the Configuration Center to identify the Web Interface sites deployed on the web server running the Web Interface. For each Web Interface site, the list provides the following information:

- **IIS Web site**: The name of the website holding the web application that runs the Web Interface site.
- **Web app alias**: The alias of the web application that runs the Web Interface site. Tha alias defines the virtual path of that application on the web server.

- **Configuration**: The object which holds the site configuration and customization data of the Web Interface site in the Active Roles Administration Service.

TIP: You can also open the configured Web Interface sites from the **Web Interface** page of the Configuration Center. To open any of the configured sites, click the site in the list, then click **Open in Browser**.

Create a Web Interface site

You can create a new Web Interface site with the **Web Interface** > **Create** option of the Configuration Center. This opens the **Create Web Interface Site** wizard, allowing you to:

- Choose the IIS website configuration that will contain the web application which implements the Web Interface site.
- Specify the alias of the web application, defining the virtual path for the URL of the Web Interface site.

The wizard then lets you specify the object that will hold the configuration and customization data of the new Web Interface site in the Active Roles Administration Service. You can choose from the following options:

- **Create from a template**: If you select this option, Active Roles will create the new site from the configuration and customization settings of the template you select.
- **Use an existing configuration**: If you select this option, the new site will have the same configuration and customization as any existing Web Interface site that also uses the configuration object you select.

  TIP: Use this option if you want to create an additional instance of an existing Web Interface site on a different web server.
- **Import from an existing configuration**: If you select this option, the new Web Interface site will have the same configuration and customization as the site you select as a baseline. In this case, Active Roles imports the configuration data from the previous version of the configuration to the new Administration Service instance, then creates the new Web Interface configuration objects based on that earlier version.

  TIP: Use this option during upgrades if you want to create the new Web Interface sites based on the sites of the Active Roles version you upgraded from.
- **Import from a file**: If you select this option, the new Web Interface site will use the configuration and customization stored in the browsed export file.

  TIP: Use this option during upgrades if you want to create the new Web Interface sites from a previously exported configuration.

You can create any number of Active Roles Web Interface sites, either with each site having its own configuration, or sharing the configuration with other sites.

These site configuration entities contain all customizable settings of the user interface elements, such as the website menus, commands, and web page forms that appear on the Web Interface. Each configuration is identified by name, stored as an entity, and applied on a per-site basis. In addition, each Web Interface site configuration is stored and replicated by the Administration Service, with the same configuration files reusable for additional Web Interface sites. This allows you to:

- Reuse the configuration of existing Web Interface sites.
- Share a common configuration among multiple Web Interface sites.

NOTE: If multiple Web Interface sites share a common configuration, any customization made to one site will be automatically applied to the other sites using the same configuration. For example, if you add a command or modify a form on one site, the new command or modified form appears on all the other sites using the same configuration.

Modify a Web Interface site

You can modify existing Web Interface sites with the **Web Interface** > **Modify** option of the Configuration Center. This opens the **Modify Web Interface Site** wizard, allowing you to:

- Modify the IIS website configuration that will contain the web application which implements the Web Interface site.
- Modify the alias of the web application, defining the virtual path for the URL of the Web Interface site.

The wizard then lets you specify the object that will hold the configuration and customization data of the modified Web Interface site in the Active Roles Administration Service. You can choose from the following options:

- **Keep the current configuration**: If you select this option, the wizard will keep the current configuration of the modified website.

  TIP: Use this option if you plan no further changes in the website configuration apart from changing the IIS website configuration or the site alias.
- **Create from a template**: If you select this option, Active Roles will change the configuration and customization settings of the modified Web Interface site to those of the template you select.
- **Use an existing configuration**: If you select this option, Active Roles will change the configuration and customization settings of the modified website to those of the site you select.

  TIP: Use this option if you want to align the configuration and customization of an existing Web Interface site with that of another existing website.
- **Import from an existing configuration**: If you select this option, Active Roles will change the configuration and customization of the modified Web Interface site to that of the configuration you select as baseline. In this case, Active Roles imports the configuration data from the previous version of the configuration to the new Administration Service instance, then creates the new Web Interface configuration objects based on that earlier version.

  TIP: Use this option during upgrades if you want to modify an existing Web Interface site based on another site of the Active Roles version you upgraded from.
- **Import from a file**: If you select this option, Active Roles will change the configuration and customization of the existing Web Interface site to that of the exported configuration you select.

TIP: Use this option during upgrades if you want to align the configuration and customization of an existing Web Interface site with that of a previously exported configuration.

Delete a Web Interface site

You can delete existing Web Interface sites with the **Web Interface** > **Delete** option of the Configuration Center. This opens the **Delete Web Interface Site** wizard, deleting the selected Web Interface site from the web server.

NOTE: The wizard does not delete the site configuration object from the Administration Service. This allows you to set up new Web Interface sites later even with the configuration of the deleted site.

Export the configuration of a Web Interface site to a file

You can export the configuration object of an existing Web Interface site with the **Web Interface** > **Export Configuration** option of the Configuration Center. This opens the **Export Web Interface Site Configuration** wizard, allowing you to save the configuration of the site from the Administration Service into an `*.xml` file with the specified name to the specified location.

TIP: Use this option to back up the configuration of existing Web Interface sites, then use them as a baseline for creating additional Web Interface sites in your organization.

Delegation of user access to the Active Roles Console

By default, after installing Active Roles, every user can log in to the Active Roles Console (also known as the MMC Interface). To restrict user access to the Console, in the Configuration Center, use the **MMC Interface Access** > **Modify** menu, then select the **Restrict Console (MMC Interface) access for all users** option.

Doing so restricts all non-Active Roles Admin users from using the Active Roles Console.

TIP: You can give Active Roles Console access later to selected users with the **User Interface Management - MMC Full control** Access Template (AT) of the Active Roles Console. This AT gives access permission to the **Server Configuration** > **User Interfaces** > **MMC Interface** object.

For more information on how to use ATs, see *Applying Access Templates* in the *Active Roles Administration Guide*.

Configuration of Active Roles logging settings

The Active Roles Configuration Center also allows you to manage the logging settings of the various Active Roles components. As part of this, you can:

- Enable or disable logging for each Active Roles component.
- Open the location of the various component log files.
- Open the component logs directly in the Active Roles Log Viewer utility.

To view, configure and manage Active Roles logs, in the Configuration Center, navigate to the **Logging** page. Once opened, the page lists the following information:

- **Component**: The name of the Active Roles component producing the log, such as the Administration Service or the Active Roles Console.
- **Logging**: Indicates whether logging is enabled or disabled for the component, and shows the logging level (**Basic** or **Verbose**). While **Basic** logging includes only errors, warnings and informational messages in the log files, **Verbose** logging also adds debugging and tracing messages.
- **Log location**: Indicates the full path of the log file.

The toolbar of the **Logging** page allows you to perform the following log management tasks:

- To enable or disable logging for a component, or change the logging level, select the component in the list, then click **Modify**.
- To open the folder that contains the log file(s) of a component, select the component in the list, then click **Browse with Explorer**.
- To open the Administration Service log in the Active Roles Log Viewer utility, select Administration Service in the list of components, then click **Open in Log Viewer**. For more information, see About Active Roles Log Viewer.

Configuration of Solution Intelligence

You can enable or disable **Solution Intelligence** in the Active Roles Configuration Center for your Web Interface sites. Solution Intelligence is an optional Active Roles feature used by One Identity to gather standard telemetry data about your Active Roles deployment, containing load, performance and usage metrics, exception reports, and other diagnostic information used to improve Active Roles.

Solution Intelligence is disabled by default.

About Active Roles Configuration Shell

The **ActiveRolesConfiguration** module (also known as the "Configuration Shell") provides cmdlets for configuring Active Roles Administration Service instances and Web Interface sites. The names of the cmdlets provided by this module start with the `AR` prefix, such as `New-ARDatabase`, `New-ARService`, or `New-ARWebSite`.

NOTE: Consider the following when planning to use the **ActiveRolesConfiguration** module:

- This module is available on 64-bit operating systems only.
- You can only install this module on computers where the Administration Service or Web Interface modules are also installed. Otherwise, the module will not provide all cmdlets.

The following table lists the cmdlets of the Configuration Shell.

Table 12: Configuration Shell Cmdlets

Command	Description
Get-ARComponentStatus	Returns the installation and configuration status of the Active Roles components.
New-ARDatabase	Creates a new Active Roles database.
Import-ARDatabase	Transfers Active Roles configuration data or management history data from one database to another.
Backup-AREncryptionKey	Backs up the current encryption key of the configuration database in the local Administration Service instance into a file.
Restore-AREncryptionKey	Restores the configuration database encryption key from a backup file to the local Administration Service instance.
Reset-AREncryptionKey	Creates a new encryption key for the configuration database in the local Administration Service instance.
New-ARService	Creates the Active Roles Administration Service instance on the local computer.
Get-ARService	Gets the status of the Active Roles Administration Service instance from the local computer.
Set-ARService	Modifies the Active Roles Administration Service instance on the local computer.
Start-ARService	Starts the Active Roles Administration Service instance on the local computer.
Stop-ARService	Stops the Active Roles Administration Service instance on the local computer.
Restart-ARService	Stops and starts the Active Roles Administration Service instance on the local computer.
Remove-ARService	Deletes the Active Roles Administration Service instance from the local computer.
Test-ARServiceDatabaseSettings	Verifies whether the specified Active Roles database settings would cause Management History issues due to setting separate Configuration and Management History databases.
Get-ARServiceStatus	Gets the Active Roles Administration Service status information from the local computer.
Get-ARVersion	Gets the version of the local Active Roles installation.
New-ARWebSite	Creates a new Active Roles Web Interface site.

Command	Description
Get-ARWebSite	Gets the Active Roles Web Interface sites from the web server.
Set-ARWebSite	Modifies the specified Active Roles Web Interface site on the web server.
Remove-ARWebSite	Deletes the specified Active Roles Web Interface site from the web server.
Get-ARWebSiteConfig	Gets Web Interface site configuration objects from the Active Roles Administration Service.
Export-ARWebSiteConfig	Exports the specified Web Interface site configuration to a file.

About System Checker

You can start the System Checker by running the **Active Roles System Checker** application from the **Start** menu or **Apps** page, depending upon your version of the Windows operating system.

From the **System Checker** main window, you can perform the following tasks:

- To check your computer, click **System Readiness Checks**, then select the appropriate Active Roles version for which to perform the checks.
- To check a particular SQL Server instance, click **SQL Server Checks** and specify the SQL Server instance to check. You can also specify the authentication method and connection credentials for access to the SQL Server instance.
- To check a particular Active Directory domain or a particular Domain Controller (DC), click **Active Directory Checks** and specify the name of the domain or the name of the DC. You can also specify connection credentials for access to the domain or DC.

System Checker then creates a report of the selected action, and displays it in its report viewer. Reports are divided into sections, each of which represents the results of a single check. If a report section includes any errors or warning messages, you can view the messages by expanding the section in the report viewer.

The report viewer also allows you to:

- Print the report.
- Export the report to an HTML file, so that you can open the report in a web browser later.
- Save the report to a report file, so that you can open the saved report in the report viewer later.
- Open a saved report by clicking **Open** in the main menu of System Checker, and selecting the report file.

- Rebuild the report, and optionally also changing the report options.

  To rebuild the report, click **Recheck** on the toolbar of the report viewer.

About Active Roles Log Viewer

The Active Roles Log Viewer tool allows you to browse and analyze:

- Diagnostic log files created by the Active Roles Administration Service.
- Event log files created by saving the Active Roles event logs in the Windows Event Viewer on the computer running the Administration Service.

The Log Viewer tool can help you to:

- Check the sequence or hierarchy of requests processed by the Administration Service.
- Identify error conditions that the Administration Service encountered during request processing.
- Find Knowledge Base (KB) Articles for specific log messages and errors.

You can open Active Roles diagnostic log files (`ds.log`) or saved event log files (`*.evtx`) with the Log Viewer tool, allowing you to check:

- The errors encountered by the Administration Service and recorded in the log file.
- Requests processed by the Administration Service and traced in the log file.
- All trace records found in the diagnostic log file.
- All events found in the event log file.

When you select an error from the list, you can also look for applicable One Identity KB Articles to learn more about the log entry or troubleshoot selected errors.

In addition, the Active Roles Log Viewer tool also allows you to:

- Search in the loaded log file for a particular text string, such as an error message.
- Filter the list by various conditions to narrow the listed items to those you are actually interested in.
- View detailed information about each list item, such as error details, request details or stack trace.

Getting started

To start using Active Roles Log Viewer, see the following resources:

- For more information on how to install Active Roles Log Viewer, see *Installing the Diagnostic Tools* in the *Active Roles Installation Guide*.

- For more information on using Active Roles Log Viewer, see *Using the Log Viewer tool* in the *Active Roles Administration Guide*.

About federated authentication

Federated authentication (also known as claim-based authentication) allows users to access applications or websites by authenticating them against a certain set of rules, known as claims. When federated authentication is configured, users are validated across multiple applications, websites or IT systems via authentication tickets or their token.

During federated authentication, authorization is performed by acquiring the identity-related information of users both for on-premises and cloud-based products. Based on the predefined claims to identify the users trying to access the applications or websites, a single token is created for each user. This security token is used to identify the user type after the user is successfully identified.

Active Roles supports federated authentication using the WS-Federation protocol as well as SAML 2.0 authentication, allowing users to access websites or sign in to an application once with the single sign-on (SSO) option.

CAUTION: Due to RSTS connection limitations, federated authentication must be enabled for only one Active Roles instance. If you try to configure federated authentication for multiple Active Roles instances, the connection to the Active Roles database will break in the previously configured Active Roles instance.

NOTE: To use SAML 2.0 authentication, you must have a valid SSL/TLS certificate configured for Active Roles.

NOTE: After an Active Roles upgrade, to ensure that Active Roles automatically refreshes expired certificates:

- in case of using WS-Federation, in the Active RolesConfiguration Center, in **Web Interface** > **Authentication**, reconfigure federated authentication.
- in case of using SAML 2.0 authentication, in the Active RolesConfiguration Center, in **Web Interface** > **Authentication**, make sure to load the federation metadata from URL (instead of loading it from file) when configuring the federated authentication.

NOTE: Federated authentication is not supported and does not work on a standalone Active Roles Web Interface instance.

For more information on configuring federated authentication for various identity providers, see *Federated authentication settings and identity providers* in the *Active Roles Administration Guide*.

SAML 2.0 authentication in Active Roles

Starting from version 8.2, Active Roles provides SAML 2.0 authentication support to the Web Interface by integration with a Redistributable Secure Token Server (RSTS). RSTS acts as both a WS-Federation provider for Active Roles and a SAML 2.0 consumer for your identity provider. As a broker between your IdP and Active Roles, you must prepare for a few design considerations when planning to use it.

For more information on configuring SAML 2.0 authentication for Active Roles, see *Configuring SAML 2.0 authentication* in the *Active Roles Administration Guide*.

Associated Active Directory

When a user authenticates via SAML 2.0 to Active Roles, they actually authenticate to the RSTS application. RSTS then attempts to validate that the user exists in an associated Active Directory. When a user authenticates to Active Roles, regardless of the authentication type, they may only authenticate with an account that exists in the Active Directory domain that the Active Roles Server is joined to, or any other trusted domains. However, when authenticating with SAML 2.0, the account must exist in the specific domain that is set as the **Associated Active Directory**. It will not authenticate users in additional trusted domains.

The **Default Active Directory** domain will always be the same domain that Active Roles is joined to. If you need to authenticate users whose AD accounts exist in other domains that have a trust relationship with the domain the Active Roles Server is joined to, you can add additional Active Directory providers and external federation providers associated with those Active Directory providers.

Claims Mapping

After a user successfully authenticates to the SAML 2.0 provider, RSTS will attempt to locate an account for that user in the associated Active Directory domain. When doing so, it will only check the first of the following claims:

1. `Email`
2. `Name`
3. `NameID` or `NameIdentifier`

NOTE: One Identity recommends only using the required **`NameID`** claim to send the appropriate value. RSTS will search the Associated Active Directory for an account that has that value in the following 3 attributes:

1. `objectGUID`
2. `userPrincipalName`
3. `sAMAccountName`

After a successful match, RSTS will send a WS-Federation response to Active Roles which contains the following claims generated from the Active Directory account.

Claim	Schema (Claim value)	AD attribute
NameIdentifier	http://schemas.xmlsoap.org/ws/2005/05/identity/claims/nameidentifier	ObjectGUID
UPN	http://schemas.xmlsoap.org/ws/2005/05/identity/claims/upn	Domain\SamAccountName
EmailAddress	http://schemas.xmlsoap.org/ws/2005/05/identity/claims/emailaddress	Mail
Name	http://schemas.xmlsoap.org/ws/2005/05/identity/claims/name	DisplayName

NOTE: When configuring the **Claims Mapping** in Active Roles, One Identity recommends always using **GUID** as the **Claim Type** and `http://schemas.xmlsoap.org/ws/2005/05/identity/claims/nameidentifier` as the **Claim value**.

Voluntary threshold for managed object count

By default, Active Roles does not limit the number of managed objects you can manage. However, as the license fee is based on the managed object count, you may need to verify that the object count stays under a certain threshold. To do so, you must specify a threshold value for the number of managed objects.

Once you configure this voluntary threshold, the scheduled task that counts the managed objects will raise an alert whenever it detects that the current number of managed objects exceeds the configured threshold value. Active Roles will indicate this alert in the **Product Usage Statistics** page of the Active Roles Console, and can also send a notification over email.

Getting started

For more information on how to configure the threshold, see *Voluntary thresholds for the managed object count* in the *Active Roles Administration Guide*.

About the installation label

To identify your Active Roles installation in the Managed Object Statistics report, you can set a label for your deployment in the Active Roles Console.

This is useful, for example, if you have several Active Roles deployments installed in your organization (for example, separate pilot, non-production and production environments) and you want to easily distinguish them visually.

Once configured, the installation label appears in the title of the Managed Object Statistics report.

Getting started

For more information on how to configure an installation label for Active Roles, see *Installation label* in the *Active Roles Administration Guide*.

About safe mode

Active Roles provides a troubleshooting mode called "Safe mode" that starts Administration Service in a limited state.

When you enable Safe mode, Administration Service:

- Disregards all custom policies, workflows, scripts, scheduled tasks and other custom-made assets that may prevent Active Roles from starting and operating normally.
- Rejects connections from any users that do not have Active Roles Admin privileges.

While Safe mode is active, only Active Roles Admins can connect to Administration Service, so that they can troubleshoot problems by changing the existing Active Roles configuration or removing any customizations that could cause issues. Once troubleshooting is finished, Active Roles Admins can also turn off Safe mode and resume normal Active Roles operation.

Getting started

You can enable Safe mode from the Active Roles Management Shell.

To enable or disable Safe mode

1. On the computer running the Active Roles Administration Service, log in with a user account that has administrator rights on the computer.

   NOTE: You can enable or disable Safe mode only with a user account that has local administrator rights on the computer running Administration Service.

2. Start the Active Roles Management Shell from the Windows Start menu or the Apps page of the operating system.
3. To enable safe mode, enter the following commands in the Management Shell command-line interface:

   ```
   Set-ARService -SafeModeEnabled $true
   ```

   ```
   Restart-ARService
   ```

4. To enable safe mode, enter the following commands in the Management Shell command-line interface:

   ```
   Set-ARService -SafeModeEnabled $false

   Restart-ARService
   ```

About Exchange Resource Forest Management

The Exchange Resource Forest Management (ERFM) feature of Active Roles allows you to automate mailbox provisioning for on-premises users in environments where the mailboxes and the user accounts are managed in different Active Directory (AD) forests. Such multi-forest environments are based on the resource forest model, and mailboxes provisioned in such environments are called linked mailboxes.

Multi-forest AD deployments have higher administrative and support costs. However, they offer the highest level of security isolation between AD objects and the Exchange service. As such, One Identity recommends configuring the resource forest model for use with Active Roles in organizations that:

- Aim for an extra layer of data security.
- Frequently experience organizational changes (for example, buying companies, or consolidating and breaking off branch companies, departments and other business units).
- Abide by certain legal or regulatory requirements.

AD deployments following the resource forest model use two types of AD forests:

- **Account forests**: These AD forests store the user objects. Organizations can use one or more account forests in the resource forest model.
- **Resource forest**: This AD forest contains the Exchange server and stores the mailboxes of the user objects.

With ERFM, you can automate the **provisioning**, **synchronization** and **deprovisioning** of linked mailboxes in the resource forest for user accounts in the account forest(s).

- During **provisioning**, Active Roles can automatically create linked mailboxes for new users (if you select to create a mailbox for the user), or create linked mailboxes for existing users without a mailbox.

  In both cases, Active Roles creates a disabled **shadow user** account in the resource forest for the user, then links it to the user account of the user in the account forest (also known as the **master account**).

  NOTE: By default, the shadow user account has the same name as the master user account in the account forest. However, if a shadow account with the same name already exists (for example, because Active Roles has already created a linked

mailbox for a user in a different account forest), Active Roles uses a different shadow account name to maintain uniqueness.

- Once a linked mailbox is created, Active Roles automatically **synchronizes** the properties of the master user accounts with their shadow accounts, whenever you modify them.
- Finally, if the master user account is **deprovisioned**, Active Roles automatically deprovisions its shadow account as well, provided that you applied mailbox deprovisioning policies to the container that holds the shadow accounts in the resource forest.

  NOTE: Like other AD objects, you can un-deprovision master user accounts as well. However, their shadow accounts are un-deprovisioned automatically only if the container of the deprovisioned master accounts has the **ERFM - Mailbox Management** built-in policy applied on them.

Getting started

For more information on the prerequisites and configuration of ERFM and linked mailboxes, see *Configuring linked mailboxes with Exchange Resource Forest Management* in the *Active Roles Administration Guide*.

About Skype for Business Server User Management

To provision Skype for Business Server user accounts in single-forest and multi-forest Active Directory (AD) environments, Active Roles offers the Skype for Business User Management feature.

The Skype for Business Server User Management feature provides built-in Active Roles policies that synchronize user account information between Active Roles and Skype for Business Server, allowing you to perform Skype for Business Server user management tasks via the Active Roles Web Interface.

Skype for Business Server User Management lets you use Active Roles to:

- Add and enable new Skype for Business users.
- View or change Skype for Business Server user properties and policy assignments.
- Move Skype for Business Server users from one Skype for Business Server pool to another.
- Disable or re-enable user accounts for Skype for Business Server.
- Remove users from Skype for Business Server.

To perform these administration tasks, the feature adds the following elements to Active Roles:

- Built-in Policy Objects that enable Active Roles to perform user management tasks on Skype for Business Server, either in a single-forest or a multi-forest AD environment.
- Additional commands and pages in the Active Roles Web Interface for managing Skype for Business Server users.
- Access Templates (ATs) to delegate Skype for Business Server user management tasks.

The Skype for Business Server User Management policy allows you to control the following factors of creating and managing Skype for Business Server users:

- **SIP user name generation rules**. When adding and enabling a new Skype for Business Server user, Active Roles can generate a SIP user name based on other properties of the user account.
- **SIP domain selection rules**. When configuring the SIP address for a Skype for Business Server user, Active Roles can restrict the list of selectable SIP domains and suggest which SIP domain to select by default.
- **Telephony selection rules**. When configuring telephony for a Skype for Business Server user, Active Roles can restrict the list of selectable telephony options and can suggest default options to select.
- **Pool selection rules**. When adding and enabling a new Skype for Business Server user, Active Roles can restrict the list of selectable registrar pools and suggest which pool to select by default. This rule also applies to selecting the destination pool when moving a Skype for Business Server user from one pool to another.

Skype for Business Server User Management provides a number of ATs allowing you to delegate the following tasks in Active Roles:

- Add and enable new Skype for Business Server users.
- View existing Skype for Business Server users.
- View or change the SIP address for Skype for Business Server users.
- View or change the telephony option and related settings for Skype for Business Server users.
- View or change Skype for Business Server user policy assignments.
- Disable or re-enable user accounts for Skype for Business Server.
- Move users from one Skype for Business Server pool to another.
- Remove users from Skype for Business Server.

Getting started

For more information on the prerequisites and configuration of Skype for Business Server User Management, see *Skype for Business Server Solution* in the *Active Roles Administration Guide*.

Active Directory topologies supported by Skype for Business Server User Management

Skype for Business Server User Management supports the following Active Directory Domain Services (AD DS) topologies.

Single forest with single tree or multiple trees

In a single forest topology, the login-enabled user accounts managed by Active Roles are stored in the same Active Directory forest in which Skype for Business Server is deployed.

Skype for Business Server user management tasks have two main steps in a single-forest configuration:

1. First, Active Roles makes changes to the attributes of the configured user account.
2. Then, based on the attribute changes, the Skype for Business Server User Management policy requests the Skype for Business Server remote shell to update the user account accordingly.

For example, when creating a new Skype for Business Server user, Active Roles sets a virtual attribute on that user account directing the policy to invoke the remote shell command for enabling the new user for Skype for Business Server. When making changes to an existing Skype for Business Server user, Active Roles populates the attributes of the user account with the desired changes, causing the policy to apply those changes via the remote shell.

Multiple forests in a resource forest topology

In a resource forest topology, the servers running Skype for Business Server are hosted in a separate Skype for Business Server forest that does not host any login-enabled user accounts. Instead, the user accounts are stored in a user forest (or forests) where no Skype for Business Server instances are hosted.

1. When creating a Skype for Business Server account for a user from an external forest, Active Roles:
2. Creates an inactive user account (known as the "shadow account") in the Skype for Business Server forest.
3. Links the associated user account in the user forest ("master account") with the inactive shadow account.
4. Activates the shadow account for Skype for Business Server.

The policies of the Skype for Business Server User Management feature then work as follows:

The Master Account Management policy ensures that the attributes of the shadow account are synchronized with the attributes of the master account, so that you can administer Skype for Business Server user properties on the master account via Active Roles.

The User Management policy detects the attribute changes replicated from the master account to the shadow account in the Skype for Business Server forest, and translates them to remote shell commands on Skype for Business Server, similarly to how synchronization is performed in a single-forest configuration.

Multiple forests in a central forest topology

In a central forest topology, the servers running Skype for Business Server are hosted in a separate Skype for Business Server forest. However, unlike in a resource forest topology, this forest can also host login-enabled accounts. Outside the Skype for Business Server forest, user forests host login-enabled user accounts, but no servers running Skype for Business Server.

In this forest configuration, the Skype for Business Server User Management policy is applied to login-enabled user accounts in the Skype for Business Server forest. As a result, Active Roles can enable and administer those user accounts for Skype for Business Server in the same way as in case of using a single-forest configuration.

When creating a Skype for Business Server account for a user from an external forest, Active Roles performs the following actions:

1. Creates a contact in the Skype for Business Server forest.
2. Links the user account in the user forest (that is, the "master account") and the contact in the Skype for Business Server forest (that is, the "shadow account").
3. Activates the contact for Skype for Business Server.
4. The Master Account Management policy then ensures that the attributes of the contact are synchronized with the attributes of the user account, so that Skype for Business Server user properties can be administered on the user account via Active Roles.
5. In the Skype for Business Server forest, the User Management policy detects the attribute changes replicated from the user account to the contact, and translates them to remote shell commands on Skype for Business Server, similarly to how synchronization is performed in a single-forest configuration.

Overview of Active Roles Synchronization Service

Identity information can be stored in various data systems, such as directories, databases, or even formatted text files. However, managing and synchronizing such identity information among several different data systems have several challenges:

- The synchronization process can require considerable time and effort.
- Performing data synchronization tasks manually is error-prone and can lead to duplicate information or incompatible data formats.

Active Roles Synchronization Service helps you avoid these problems by automating the process of identity data synchronization among various data systems used in your enterprise environment.

Synchronization Service increases the efficiency of identity data management by allowing you to automate the creation, deprovisioning, and update operations between the data systems you use. For example, when an employee joins or leaves the organization, the identity information managed by Synchronization Service is automatically updated in the managed data systems, reducing administrative workload and getting the new users up and running faster.

Synchronization Service also supports scripting capabilities, providing a flexible way to automate administrative tasks and integrate the administration of managed data systems with other business processes. By automating conventional tasks, Synchronization Service helps your organization to concentrate on strategic issues, such as planning the directory, increasing enterprise security, and supporting business-critical applications.

For more information on the main features of Synchronization Service, see the following sections.

Getting started

For more information on how to install, configure and use Synchronization Service, see the *Active Roles Synchronization Service Administration Guide*.

About bidirectional synchronization

Bidirectional synchronization allows you to synchronize all changes to identity information between your data systems. Using this feature, you can prevent potential identity information conflicts between different data sources.

NOTE: This feature is only supported by certain data systems. For more information, see the relevant data connector documentation in the *Active Roles Synchronization Service Administration Guide*.

About delta processing

Delta processing allows you to synchronize identities faster by processing only data that has changed in the source and target connected systems since the last synchronization run.

By offering both full synchronization or quick delta processing methods between two data systems, Synchronization Service provides you the flexibility of choosing the appropriate method for your synchronization tasks.

NOTE: This feature is only supported by certain data systems. For more information, see the relevant data connector documentation in the *Active Roles Synchronization Service Administration Guide*.

About group membership synchronization

Synchronization Service ensures that group membership information is synchronized across all connected data systems. For example, when creating a group object from an Active Directory (AD) domain to an AD LDS (ADAM) instance, you can configure rules to synchronize the **Member** attribute from the AD domain to the AD LDS (ADAM) instance.

About Windows PowerShell scripting

Synchronization Service supports Windows PowerShell-based scripting for data synchronization. The shell is implemented as a Windows PowerShell module, allowing you to automate synchronization tasks via PowerShell scripts.

For more information and examples, see the following sections of the *Active Roles Synchronization Service Administration Guide*:

- *Developing PowerShell scripts for attribute synchronization rules*
- *Using PowerShell script to transform passwords*

About attribute synchronization rules

Synchronization Service allows you to create and configure synchronization rules to generate values for target object attributes. These rules support three synchronization types:

- **Direct synchronization**: Assigns the value of a source object attribute to the target object attribute you specify.
- **Script-based synchronization**: Uses your custom Windows PowerShell script to generate the target object attribute value.
- **Rule-based synchronization**: Uses your custom synchronization rules to generate the target object attribute value you want.

About rule-based generation of Distinguished Names

Synchronization Service provides flexible rules for generating the Distinguished Names (DNs) for the created objects. These DN generation rules allow you to ensure that the created objects are named in full compliance with the naming conventions existing in your organization.

About synchronization scheduling

To meet your organizational policies and save both time and effort, you can schedule and automate the configured data synchronization tasks with Synchronization Service.

About extensive data system support

To access external data systems, Synchronization Service uses so-called "connectors", enabling Synchronization Service to read and synchronize identity data from the specific data systems.

Active Roles Synchronization Service can connect to the following data systems:

- Data sources accessible via an OLE DB provider.
- Delimited text files.
- IBM AS/400, IBM Db2, and IBM RACF systems.
- LDAP directory service.
- Micro Focus NetIQ Directory systems.
- The following Microsoft services and resources:
    - Active Directory Domain Services (AD DS) with the domain or forest functional level of Windows Server 2016 or higher.

        - Active Directory Lightweight Directory Services (AD LDS) running on any Windows Server operating system supported by Microsoft.
        - Azure Active Directory (Azure AD) using Microsoft Graph API version 1.0.
        - Exchange Online services.
        - Exchange Server with the following versions:
            - Microsoft Exchange Server 2019
            - Microsoft Exchange Server 2016
        - Lync Server version 2013 with limited support.
        - SharePoint 2019, 2016, or 2013.
        - SharePoint Online service.
        - Skype for Business 2019, 2016 or 2015.
        - Skype for Business Online service.
        - SQL Server, any version supported by Microsoft.
    - One Identity Active Roles version 7.4.3, 7.4.1, 7.3, 7.2, 7.1, 7.0, and 6.9.
    - One Identity Manager version 8.0 and 7.0 (D1IM 7.0).
    - OpenLDAP directory service.
    - Oracle Database, Oracle Database User Accounts, and Oracle Unified Directory data systems.
    - MySQL databases.
    - Salesforce systems.
    - SCIM-based data systems.
    - ServiceNow systems.

For more information on using these connectors, see *External data systems supported with built-in connectors* in the *Active Roles Synchronization Service Administration Guide*.

Support for AWS Managed Microsoft AD

NOTE: This feature is officially supported starting from Active Roles 8.1.3 SP1 (build 8.1.3.10). It is not supported on Active Roles 8.1.3 (build 8.1.3.2) and earlier versions.

Active Roles supports deployment and configuration in the Amazon cloud to manage AWS Managed Microsoft AD instances hosted via AWS Directory Service.

This allows you to:

- Perform Active Directory management tasks in your AWS Managed Microsoft AD environment.
- Synchronize directory data from an on-premises AD environment to AWS Managed Microsoft AD.
- Synchronize passwords from an on-premises Active Directory to AWS Managed Microsoft AD (with certain limitations).

For more information on configuring Active Roles to manage AWS Managed Microsoft AD environments, see *Configuring Active Roles for AWS Managed Microsoft AD* in the *Active RolesInstallation Guide* or *Active RolesAdministration Guide* documents.

For more information on configuring Active RolesSynchronization Service to synchronize resources to or from AWS Managed Microsoft AD, see *Installing and configuring Synchronization Service for use with AWS Managed Microsoft AD* in the *Active RolesSynchronization ServiceAdministration Guide*.

Supported AWS Managed Microsoft AD deployment configuration

To manage AWS Managed Microsoft AD environments, you must deploy Active Roles in Amazon Web Services (AWS) in the following configuration:

- Active Roles must be deployed on an Amazon Elastic Compute Cloud (EC2) instance or instances. For more information, see the *Amazon Elastic Compute Cloud documentation*.
- The SQL Server required by Active Roles Administration Service must run on a separate Amazon Relational Database Service for Microsoft SQL Server (RDS for SQL Server) instance. For more information, see the *Amazon RDS documentation*.
- The Active Directory environment must be hosted in AWS via AWS Directory Service. For more information, see the *AWS Directory Service documentation*.

NOTE: Support for AWS Managed Microsoft AD by Active Roles was tested only in this configuration. Active Roles does not officially support managing AWS Managed Microsoft AD environments in a hybrid deployment, that is, using an on-premises Active Roles and/or SQL Server installation and hosting AD via AWS Directory Service.

Supported Active Roles features with AWS Managed Microsoft AD

If configured to manage AWS Managed Microsoft AD, Active Roles offers a feature set similar to managing an on-premises AD service. This includes:

- Performing the day-to-day administration tasks of AD objects (users, contacts, computers, distribution and security groups, Organizational Units, shared folders) in the Active Roles Console or the Web Interface.
- Rule-based and role-based administrative views and permissions for AD objects (Managed Units and Access Templates).
- Automation and approval workflows for AD objects.
- Importing the Management History database and/or Configuration database from an on-premises Active Roles installation of the same version. This is useful if you want to migrate the configuration of an existing on-premises Active Roles installation to your Active Roles installation running in an EC2 instance to manage AWS Managed Microsoft AD.
- Synchronization Service connections and sync workflows based on the following Active Roles Synchronization Service connectors:
    - Active Directory Connector
    - Active Roles Connector
    - Delimited Text File Connector
- Synchronizing passwords with Active Roles Synchronization Service from on-premises AD to AWS Managed Microsoft AD.

NOTE: For the limitations of password synchronization from on-premises AD to AWS Managed Microsoft AD, see Active Roles feature limitations when using AWS Managed Microsoft AD.

Active Roles feature limitations when using AWS Managed Microsoft AD

When using Active Roles to manage AWS Managed Microsoft AD resources, consider the following limitations.

Amazon Web Services limitations

For Active Roles installations deployed in Amazon Elastic Compute Cloud (EC2) instances and SQL Servers hosted on Amazon Relational Database Service for SQL Server (RDS for SQL Server) instances, the known EC2 and RDS limitations apply.

- For more information about the known EC2 limitations, see Launch template restrictions, Hibernation limitations and (if applicable) Constraints on the size and configuration of an EBS volume in the *Amazon EC2 documentation*.
- For more information about the known Amazon RDS limitations, see Quotas and constraints in the *Amazon RDS documentation*.

AD LDS, Azure AD, Exchange and Exchange Online support

Active Roles components (such as the Active Roles Console or Web Interface) that also support directory services other than AD (AD LDS, Azure AD, Exchange, or Exchange Online) were only tested to support AD-related configuration and administration tasks.

Likewise, Active Roles features (such as Managed Units or Access Templates) that also support managing objects from directory services other than AD (AD LDS, Azure AD, Exchange, or Exchange Online) were only tested to support AD object and permission management.

Domain Admin account management

As AWS has exclusive control over Domain Admin accounts, managing such accounts with Active Roles is not possible in AWS Managed Microsoft AD.

For more information, see Admin account in the *AWS Directory Service documentation*.

Federated authentication support

Federated authentication with WS-Fed was not tested to work with AWS Managed Microsoft AD.

Non-AD specific Active Roles features

Active Roles features used to manage non-AD directory services (such as Exchange Resource Forest Management) were not tested to work with AWS Managed Microsoft AD.

Service Connection Point discovery

Active Roles connected services (such as the Active Roles Console) rely on AD Discovery to create Service Connection Points (SCPs) and find other Active Roles services.

As AWS Directory Service does not support AD Discovery, Active Roles services installed on an EC2 instance to manage AWS Managed Microsoft AD may not be able to automatically discover the Active Roles Administration Service, impacting the user experience.

Synchronization Service limitations

- When synchronizing directory data or passwords from on-premises Active Directory to AWS Managed Microsoft AD, Active Roles Synchronization Service has the following limitations:
    - Active Roles Synchronization Service was only tested to work with connections and sync workflows based on the following connectors:
        - Active Directory Connector
        - Active Roles Connector
        - Delimited Text File Connector

      Sync workflows and connections based on other connectors are not officially supported.
    - When synchronizing passwords from an on-premises Active Directory to AWS Managed Microsoft AD, synchronizing the `pwdHash` attribute and synchronizing then populating the `SIDHistory` attribute to AWS Managed Microsoft AD is not supported. This is because the Synchronization Service Capture Agent cannot be installed in an AWS Managed Microsoft AD environment.
- Synchronizing passwords from AWS Managed Microsoft AD to on-premises AD with Active Roles Synchronization Service is not supported. This is because the Synchronization Service Capture Agent cannot be installed in an AWS Managed Microsoft AD environment.

SQL Server replication support

As Active Roles uses RDS for SQL Server when managing AWS Managed Microsoft AD, the SQL server replication feature of Active Roles is not supported.

Usable Organizational Unit in the AD domain

After you connect the Active Roles Console to your AWS Managed Microsoft AD environment, the AD domain and its containers will appear in the Active Roles Console (and if configured, in the Web Interface as well). By default, the AWS Managed Microsoft AD environment contains three types of containers:

- AWS-specific containers.
- The default AD-specific containers (such as `Builtin`, `Computers`, `Domain Controllers`, `ForeignSecurityPrincipals`, and so on).
- An Organizational Unit container matching the NetBIOS (or shortname) of the AWS Managed Microsoft AD deployment. For example, if the shortname of your AD domain is **ARDEMO**, the name of this container will also be **ARDEMO**.

Consider that out of these three container types, you can manage AD resources only in the Organizational Unit with the name matching the shortname of your AWS Managed Microsoft AD environment. All other containers will be read-only.

4

FIPS compliance

Active Roles 8.2.1 supports cryptography libraries and algorithms compliant with Federal Information Processing Standards (FIPS) 140-2. For more information on FIPS-compliant libraries and algorithms, see FIPS 140-2: Security Requirements for Cryptographic Modules.

NOTE: Consider the following when planning to use FIPS-compliant cryptography libraries or algorithms:

- Although Active Roles continues to support non-FIPS compliant cryptography libraries and algorithms, it will not work properly if it is configured to use non-FIPS compliant solutions in a FIPS-compliant environment.
- If you already use FIPS-compliant security algorithms in your environment (such as the TripleDES security algorithm, or the SHA256 hash algorithm), you must export your existing configuration, and import it in a new Active Roles installation.

LSA protection support

The Active Roles Synchronization Service Capture Agent supports Local Security Authority (LSA). For more information, see Configuring Additional LSA Protection in the *Microsoft Windows Server documentation*.

STIG compliance

Active Roles 8.2.1 has been checked against the following Security Technical Implementation Guidelines (STIGs) of the Defense Information Systems Agency (DISA).

- Application Security and Development
- MS SQL Server 2016 Database
- MS SQL Server 2016 Instance

The checks performed during STIG validation are compliant with the following National Institute of Standards and Technology (NIST) Special Publications (SP):

- NIST SP 800-53
- NIST SP 800-53A
- NIST SP 800-53 Revision 4

About us

One Identity solutions eliminate the complexities and time-consuming processes often required to govern identities, manage privileged accounts and control access. Our solutions enhance business agility while addressing your IAM challenges with on-premises, cloud and hybrid environments.

Contacting us

For sales and other inquiries, such as licensing, support, and renewals, visit https://www.oneidentity.com/company/contact-us.aspx.

Technical support resources

Technical support is available to One Identity customers with a valid maintenance contract and customers who have trial versions. You can access the Support Portal at https://support.oneidentity.com/.

The Support Portal provides self-help tools you can use to solve problems quickly and independently, 24 hours a day, 365 days a year. The Support Portal enables you to:

- Submit and manage a Service Request
- View Knowledge Base articles
- Sign up for product notifications
- Download software and technical documentation
- View how-to videos at www.YouTube.com/OneIdentity
- Engage in community discussions
- Chat with support engineers online
- View services to assist you with your product