Distributed Action Handler

Software Version 12.4

Administration Guide

MICRO FOCUS

Document Release Date: October 2019
Software Release Date: October 2019

Legal notices

Copyright notice

The only warranties for products and services of Micro Focus and its affiliates and licensors (“Micro Focus”) are set forth in the express warranty statements accompanying such products and services. Nothing herein should be construed as constituting an additional warranty. Micro Focus shall not be liable for technical or editorial errors or omissions contained herein. The information contained herein is subject to change without notice.

Documentation updates

The title page of this document contains the following identifying information:

- Software Version number, which indicates the software version.
- Document Release Date, which changes each time the document is updated.
- Software Release Date, which indicates the release date of this version of the software.

To check for updated documentation, visit https://www.microfocus.com/support-and-services/documentation/.

Support

Visit the MySupport portal to access contact information and details about the products, services, and support that Micro Focus offers.

This portal also provides customer self-solve capabilities. It gives you a fast and efficient way to access interactive technical support tools needed to manage your business. As a valued support customer, you can benefit by using the MySupport portal to:

- Search for knowledge documents of interest
- Access product documentation
- View software vulnerability alerts
- Enter into discussions with other software customers
- Download software patches
- Manage software licenses, downloads, and support contracts
- Submit and track service requests
- Contact customer support
- View information about all services that Support offers

Many areas of the portal require you to sign in. If you need an account, you can create one when prompted to sign in. To learn about the different access levels the portal uses, see the Access Levels descriptions.

Contents

Part I: Use the Distributed Action Handler

This section describes how to install, configure, and operate Micro Focus Distributed Action Handler.

Chapter 1: Introduction

This chapter describes the basic functions of the Micro Focus Distributed Action Handler (DAH).

About the Distributed Action Handler

Micro Focus Distributed Action Handler (DAH) is a server that distributes ACI actions to IDOL Servers. Distribution enables you to scale your system in a linear manner, increasing the speed at which it runs actions and saving processing time. Having multiple IDOL Servers also ensures uninterrupted service if any IDOL Server fails.

The setup of your IDOL Servers is independent of the DAH, because they are architecturally unaware of it.

You can run the DAH in the following modes:

- **Mirror mode**. The *child servers* (the servers that the DAH distributes actions to) are identical—that is, each one is configured the same way and contains the same data as the others.
- **Non-mirror mode**. The child servers are all different—that is, each one is configured differently and contains different data from the others. If you run the DAH in non-mirror mode, you must set up *virtual databases* (VDBs) of the following types:
    - **Combinator VDB**. The virtual database forwards an action to all the databases that it represents. The VDB collates and sorts the results before it returns them.
    - **Distributor VDB**. The virtual database forwards an action to one of the databases that it represents. These databases must be identical (that is, all the databases are exact copies of each other and contain the same data). The distribution method determines which database the distributor VDB forwards actions to.

> **NOTE:** Virtual databases can map to IDOL Server databases or to other virtual databases that you set up for the DAH.
>
> However, in general, Micro Focus recommends that combinator virtual databases map only to other virtual databases (combinator VDBs combine results from distributor VDBs). Distributor databases normally map directly to the IDOL Server databases. Micro Focus recommends that you do not use them to distribute queries to combinator VDBs.

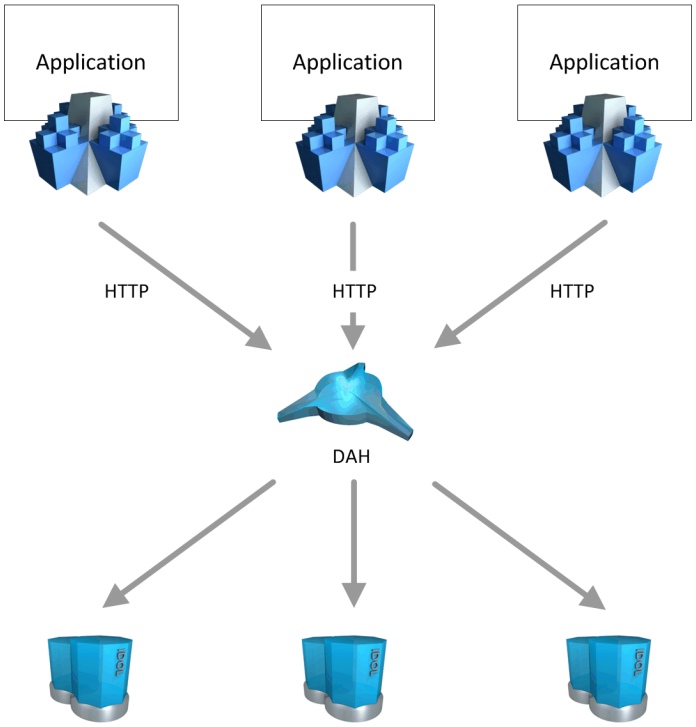
OEM Certification

Distributed Action Handler works in OEM licensed environments.

Distribution in Mirror Mode

The diagram below represents a DAH run in mirror mode. In this setup, the DAH distributes actions to several identical IDOL Servers.

DAH system architecture (mirror mode)



The DAH distributes the actions according to the value of the `DistributionMethod` parameter in the DAH configuration file. You can use the following methods:

- **Load balancing**. The DAH assigns each incoming action to just one of the connected IDOL Servers. It uses a cumulative predictive algorithm to spread the action load efficiently. When this IDOL Server responds with a result, the DAH returns it to the client software.

  If an IDOL Server stops responding for any reason (for example, because of a hardware failure or network outage), the DAH marks it as down. DAH assigns incoming actions only to the available IDOL Servers, which saves the time that it takes to attempt to communicate with the failed IDOL Server.

- **Failover**. The DAH forwards incoming actions to the first IDOL Server that you list in the DAH configuration file `[DistributedEngine`*N*`]` section. This server is the primary server. If this IDOL Server stops responding for any reason (for example, because of a hardware or network failure), the DAH marks it as down. The DAH switches to the next IDOL Server (the second one listed in the `[DistributedEngine`*N*`]` section), which becomes the new primary server, and so on, and seamlessly continues to service client actions.

The DAH periodically checks IDOL servers that are down. If an IDOL Server has come online again (for example, because the hardware has been rebooted or the network connection repaired), the DAH adds it back into the list of active IDOL Servers.

- For load balancing, this IDOL Server becomes a valid choice for actions again.
- For failover, DAH continues sending actions to the new primary server unless that server stops responding, or unless you manually change the primary server.

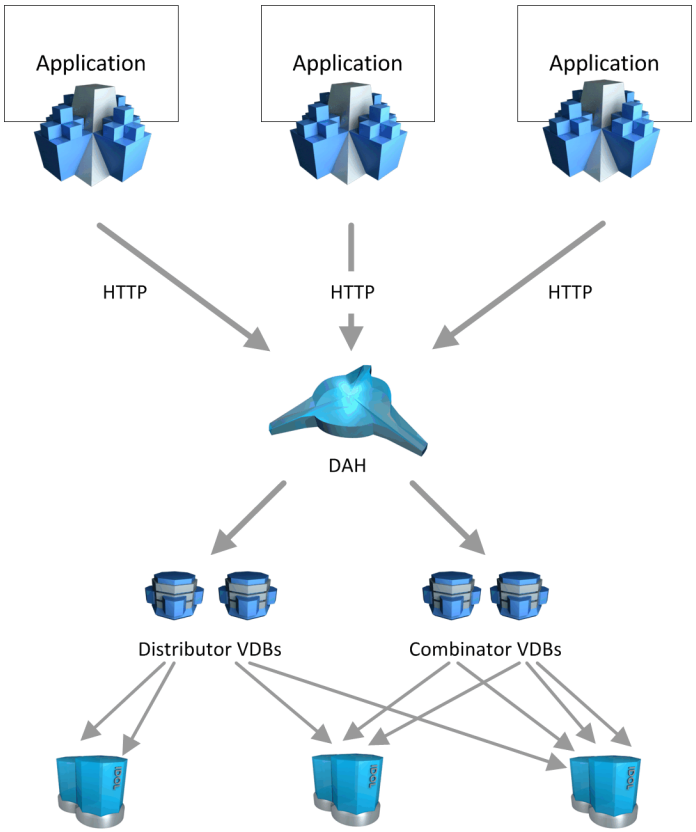
NOTE: The DAH always sends state-changing actions (actions that cause changes in a child server) to all child servers, to ensure that they remain consistent. The DAH queues state-changing actions for any child servers that are not running, and sends these actions before it distributes new actions to the child server.

The DAH cannot always queue `CategoryMove` and `CategorySetDetails` actions. It returns an error if child servers stop running.

Distribution in Non-Mirror Mode

The diagram below represents a DAH run in non-mirror mode. The DAH distributes actions to several different IDOL Servers.

DAH system architecture (non-mirror mode)



The DAH uses either combinator or distributor virtual databases (VDBs) to distribute the actions to one or more specific IDOL Server databases. The VDBs can map to one or more IDOL Server databases, or to other virtual databases that you set up for the DAH. The way a VDB forwards actions depends on its type.

Combinator VDBs

A combinator virtual database forwards actions to all the databases that it represents. The VDB collates and sorts the results before it returns them.

Distributor VDBs

Distributor virtual databases map to a set of identical databases that contain identical data. You determine how the VDB distributes actions by using the `DistributionMethod` configuration parameter in the DAH configuration file. You can use one of the following methods:

- **Load balancing**. The distributor VDB assigns each action to just one of the databases that it maps to. It uses a cumulative predictive algorithm to spread the action load efficiently. When this database responds with a result, the DAH forwards it to the client.

  If an IDOL Server stops responding for any reason (for example, because of a hardware failure or network outage), the DAH marks it as down. DAH assigns incoming actions only to the available IDOL Servers, which saves the time that it takes to attempt to communicate with the failed IDOL Server.

- **Failover**. The distributor VDB forwards incoming actions to the first database that you list in the virtual database `MapsTo` parameter. If this database stops responding for any reason (for example, because of a hardware failure or network outage), the DAH marks it as down. The DAH switches to the next database (the second one listed for the `MapsTo` parameter) and so on, and seamlessly continues to service client actions.

The DAH periodically checks IDOL Servers that are down. If it finds that an IDOL Server has come online again (for example, because the hardware has been rebooted or the network connection repaired), it adds it back into the list of active IDOL servers.

- For load balancing, this IDOL Server becomes a valid choice for actions again.
- For failover, the first IDOL Server takes over as the sole destination for actions (because it has precedence over the IDOL Server that is currently serving results).

Supported Actions

If you run the DAH in mirror mode, it can distribute all IDOL Server actions. Refer to the *IDOL Server Reference* for a complete listing of actions.

If you run the DAH in non-mirror mode, it can distribute the actions described in the *Distributed Action Handler Reference*.

Related Topics

- Display Online Help, on page 28

Stand-Alone and Integrated Distributed Action Handler

You can install and operate the DAH either as a stand-alone component or integrated with IDOL. If your system integrates the DAH with IDOL:

- You must use a unified configuration to avoid duplicating configuration settings.
- You must configure the child servers to which the DAH distributes actions differently.

Related Topics

- Unified Configuration, on page 23
- Add Child Servers to the Distributed Action Handler, on page 36

Chapter 2: Install the Distributed Action Handler

This chapter describes how to install the DAH by using the IDOL Server installer.

NOTE: After you install the DAH, you must configure at least two child IDOL Servers to distribute to before you can use it. You configure child servers differently for stand-alone or integrated DAH operation.

Requirements

This section describes the software and hardware requirements for IDOL and the DAH.

Supported Platforms

IDOL runs on a variety of Windows and UNIX platforms. For details of supported platforms, refer to the *IDOL Server Release Notes*.

NOTE: To run the DAH on Microsoft Windows operating systems, you must install the `msvcrt.dll` library.

IDOL Servers Required

You must have at least two IDOL Servers installed for the DAH to distribute to.

NOTE: A single DAH can distribute to up to 1,024 IDOL Servers. However, Micro Focus recommends that you use the DAH to distribute to no more than 16 IDOL Servers, because distributing to a larger number can have a negative impact on processing time.

State-Changing Actions File

In mirror mode, DAH uses the `sca.dat` file to determine the state-changing actions that it must distribute to all child servers.

When you are running the DAH in mirror mode, you must ensure that the `sca.dat` file is present. By default, this file must be in the same directory as the DAH, but you can also set the `SCAFileDirectory` configuration parameter to the location of this file.

If you upgrade DAH, you might need to download a new `sca.dat` file from the Micro Focus Big Data customer support site. Refer to the *Distributed Action Handler Release Notes*.

Install Distributed Action Handler on Windows

Use the following procedure to install Distributed Action Handler on Microsoft Windows operating systems, by using the IDOL Server installer.

The IDOL Server installer provides the major IDOL components. It also includes License Server, which Distributed Action Handler requires to run.

To install Distributed Action Handler

1. Double-click the appropriate installer package:

   `IDOLServer_`*`VersionNumber_Platform`*`.exe`

   where:

`VersionNumber`	is the product version.
`Platform`	is your software platform.

   The Setup dialog box opens.
2. Click **Next**.

   The License Agreement dialog box opens.
3. Read the license agreement. Select **I accept the agreement**, and then click **Next**.

   The Installation Directory dialog box opens.
4. Specify the directory to install Distributed Action Handler (and optionally other components such as License Server) in. By default, the system installs on `C:\MicroFocus\IDOLServer-`*`VersionNumber`*. Click  to choose another location. Click **Next**.

   The Installation Mode dialog box opens.
5. Select **Custom**, and then click **Next**.

   The OEM installation dialog box opens.
6. Choose whether to install IDOL for OEM usage.
   - To install IDOL for OEM usage
     a. Select the **OEM installation** check box.
     b. Select how to provide your license key file by choosing one of the following options:
        - **Copy from file**, then click  to find the licensekey.dat file to use.
        - **Copy the licensekey.dat manually after the installation**.

c. Click **Next**.

The Component Selection dialog box opens. Skip to Step 8.

- To install IDOL for standard usage, click **Next**.

The License Server dialog box opens. Proceed to Step 7.

7. Choose whether you have an existing License Server.
   - To use an existing License Server
     a. On the License Server dialog box, click **Yes**, and then click **Next**. The Existing License Server dialog box opens.
     b. Specify the host and ACI port of your License Server, and then click **Next**.
   - To install a new instance of License Server
     a. On the License Server dialog box, click **No**, and then click **Next**. The Service Name dialog box opens.
     b. In the Service name box, type the name of the Windows service to use for the License Server, and then click **Next**. The License Server dialog box opens.
     c. Specify the ports that you want License Server to listen on, and then type the path to your IDOL license key file (`licensekey.dat`), which you obtained when you purchased Distributed Action Handler, or click  and navigate to the location. Click **Next**.

   The Component Selection dialog box opens.
8. Click **Next**.
9. Select the check boxes for the components that you want to install, and specify the port information for each component, or leave the fields blank to accept the default port settings.

   For the DAH, you can specify the following ports:

ACI Port	The port that client machines use to send ACI actions to the DAH. Default: 9060
Service Port	The port that client machines use to send service requests to the DAH. Default: 9062

   If you do not specify a value, the installer uses the specified default ports.

   Click **Next** or **Back** to move between components.
10. After you have specified your settings, the Summary dialog box opens. Verify the settings you made and click **Next**.

    The Ready to Install dialog box opens.
11. Click **Next**.

    The Installing dialog box opens, indicating the progress of the installation. If you want to end the installation process, click **Cancel**.

12. After installation is complete, click **Finish** to close the installation wizard.

Install an IDOL Component as a Service on Windows

On Microsoft Windows operating systems, you can install any IDOL component as a Windows service. Installing a component as a Windows service makes it easy to start and stop the component, and you can configure a component to start automatically when you start Windows.

Use the following procedure to install Distributed Action Handler as a Windows service from a command line.

To install a component as a Windows service

1. Open a command prompt with administrative privileges (right-click the icon and select **Run as administrator**).
2. Navigate to the directory that contains the component that you want to install as a service.
3. Send the following command:

   Component.exe -install

   where *Component*.exe is the executable file of the component that you want to install as a service.

   The -install command has the following optional arguments:

-start {[auto] | [manual] | [disable]}	The startup mode for the component. **Auto** means that Windows services automatically starts the component. **Manual** means that you must start the service manually. **Disable** means that you cannot start the service. The default option is **Auto**.
-username *UserName*	The user name that the service runs under. By default, it uses a local system account.
-password *Password*	The password for the service user.
-servicename *ServiceName*	The name to use for the service. If your service name contains spaces, use quotation marks (") around the name. By default, it uses the executable name.
-displayname *DisplayName*	The name to display for the service in the Windows services manager. If your display name contains spaces, use quotation marks (") around the name. By default, it uses the service name.
-depend *Dependency1* [,*Dependency2 ...*]	A comma-separated list of the names of Windows services that Windows must start before the new service. For example, you might want to add the License Server as a dependency.

For example:

```
Component.exe -install -servicename ServiceName -displayname "Component Display
Name" -depend LicenseServer
```

After you have installed the service, you can start and stop the service from the Windows Services manager.

When you no longer require a service, you can uninstall it again.

To uninstall an IDOL Windows Service

1. Open a command prompt.
2. Navigate to the directory that contains the component service that you want to uninstall.
3. Send the following command:

   Component.exe -uninstall

   where *Component*.exe is the executable file of the component service that you want to uninstall.

   If you did not use the default service name when you installed the component, you must also add the -servicename argument. For example:

   Component.exe -uninstall -servicename *ServiceName*

Install Distributed Action Handler on UNIX

Use the following procedure to install Distributed Action Handler in text mode on UNIX platforms.

To install Distributed Action Handler on UNIX

1. Open a terminal in the directory in which you have placed the installer, and enter the following command:

   ./IDOLServer_*VersionNumber_Platform*.exe --mode text

   where:

VersionNumber	is the product version
Platform	is the name of your UNIX platform

   > **NOTE:** Ensure that you have execute permission for the installer file.

   The console installer starts and displays the Welcome screen.
2. Read the information and then press the Enter key.

   The license information is displayed.
3. Read the license information, pressing Enter to continue through the text. After you finish

reading the text, type **Y** to accept the license terms.

4. Type the path to the location where you want to install the servers, or press `Enter` to accept the default path.

   The Installation Mode screen is displayed.

5. Press `2` to select the Custom installation mode.

6. Choose whether to install IDOL for OEM usage.

   - To install IDOL for OEM usage

     a. Press `2` to select the OEM installation mode.

     b. Select how to provide your license key file by choosing one of the following options:

        - **Copy from file**. Press `1`, and then type the location of your `licensekey.dat` file.
        - **Copy the licensekey.dat manually after the installation**. Press `2`.

     The Component Selection dialog box opens. Go to Step 9.

   - To install IDOL for standard usage, click **Next**.

     The License Server screen is displayed. Proceed to Step 7.

7. Choose whether you have an existing License Server.

   - To use an existing License Server, type **Y**. Specify the host and port details for your License Server (or press `Enter` to accept the defaults), and then press `Enter`. Go to Step 9.
   - To install a new instance of License Server, type **N**.

8. If you want to install a new License Server, provide information for the ports that the License Server uses.

   a. Type the value for the ACI Port and press `Enter` (or press `Enter` to accept the default value).

ACI Port	The port that client machines use to send ACI actions to the License Server.

   b. Type the value for the Service Port and press `Enter` (or press `Enter` to accept the default value).

Service Port	The port by which you send service actions to the License Server. This port must not be used by any other service.

   c. Type the location of your IDOL license key file (`licensekey.dat`), which you obtained when you purchased Distributed Action Handler. Press `Enter`.

9. The Component Selection screen is displayed. Press `Enter`. When prompted, type **Y** for the components that you want to install. Specify the port information for each component, and then press `Enter`. Alternatively, leave the fields blank and press `Enter` to accept the default port settings.

   For the DAH, you can specify the following ports:

ACI Port	The port that client machines use to send ACI actions to the DAH. Default: 9060
Service Port	The port that client machines use to send service requests to the DAH. Default: 9062

If you do not specify a value, the installer uses the specified default ports.

> **NOTE:** These ports must not be used by any other service.

The Init Scripts screen is displayed.

10. Type the user that the server should run as, and then press `Enter`.

> **NOTE:** The installer does not create this user. It must exist already.

11. Type the group that the server should run under, and then press `Enter`.

> **NOTE:** If you do not want to generate init scripts for installed components, you can simply press `Enter` to move to the next stage of the installation process without specifying a user or group.

The Summary screen is displayed.

12. Verify the settings that you made, then press `Enter` to begin installation.

The Installing screen is displayed.

This screen indicates the progress of the installation process.

The Installation Complete screen is displayed.

13. Press `Enter` to finish the installation.

Install an IDOL Component as a Service on Linux

On Linux operating systems, you can configure a component as a service to allow you to easily start and stop it. You can also configure the service to run when the machine boots. The following procedures describe how to install Distributed Action Handler as a service on Linux.

> **NOTE:** To use these procedures, you must have `root` permissions.

> **NOTE:** When you install Distributed Action Handler on Linux, the installer prompts you to supply a user name to use to run the server. The installer populates the init scripts, but it does not create the user in your system (the user must already exist).

The procedure that you must use depends on the operating system and boot system type.

- For Linux operating system versions that use systemd (including CentOS 7, and Ubuntu version 15.04 and later), see Install a Component as a Service for a systemd Boot System, on the next

page.

- For Linux operating system versions that use System V, see Install a Component as a Service for a System V Boot System, on the next page.

Install a Component as a Service for a systemd Boot System

NOTE: If your setup has an externally mounted drive that Distributed Action Handler uses, you might need to modify the init script. The installed init script contains examples for an NFS mount requirement.

To install an IDOL component as a service

1. Run the appropriate command for your Linux operating system environment to copy the init scripts to your `init.d` directory.
   - Red Hat Enterprise Linux (and CentOS)

     ```
     cp IDOLInstallDir/scripts/init/systemd/componentname
     /etc/systemd/system/componentname.service
     ```

   - Debian (including Ubuntu):

     ```
     cp IDOLInstallDir/scripts/init/systemd/componentname
     /lib/systemd/system/componentname.service
     ```

   where *componentname* is the name of the init script that you want to use, which is the name of the component executable (without the file extension).

   For other Linux environments, refer to the operating system documentation.
2. Run the following commands to set the appropriate access, owner, and group permissions for the component:
   - Red Hat Enterprise Linux (and CentOS)

     ```
     chmod 755 /etc/systemd/system/componentname
     chown root /etc/systemd/system/componentname
     chgrp root /etc/systemd/system/componentname
     ```

   - Debian (including Ubuntu):

     ```
     chmod 755 /lib/systemd/system/componentname
     chown root /lib/systemd/system/componentname
     chgrp root /lib/systemd/system/componentname
     ```

   where *componentname* is the name of the component executable that you want to run (without the file extension).

   For other Linux environments, refer to the operating system documentation.
3. (Optional) If you want to start the component when the machine boots, run the following command:

   ```
   systemctl enable componentname
   ```

Install a Component as a Service for a System V Boot System

To install an IDOL component as a service

1. Run the following command to copy the init scripts to your `init.d` directory.

   `cp` *`IDOLInstallDir`*`/scripts/init/systemv/`*`componentname`* `/etc/init.d/`

   where *componentname* is the name of the init script that you want to use, which is the name of the component executable (without the file extension).

2. Run the following commands to set the appropriate access, owner, and group permissions for the component:

   ```
   chmod 755 /etc/init.d/componentname
   chown root /etc/init.d/componentname
   chgrp root /etc/init.d/componentname
   ```

3. (Optional) If you want to start the component when the machine boots, run the appropriate command for your Linux operating system environment:

   - Red Hat Enterprise Linux (and CentOS):

     ```
     chkconfig --add componentname
     chkconfig componentname on
     ```

   - Debian (including Ubuntu):

     ```
     update-rc.d componentname defaults
     ```

   For other Linux environments, refer to the operating system documentation.

Chapter 3: Configure the Distributed Action Handler

The DAH configuration file contains the settings that determine how the DAH operates. You can modify these settings to customize DAH according to your requirements.

Edit the Configuration File

You configure the DAH by modifying the DAH configuration file. The configuration file, `dah.cfg`, is installed in the DAH installation subdirectory:

installDir`\dah\dah.cfg`

where *installDir* is the directory in which the DAH is installed.

Unified Configuration

DAH is generally installed and operated as a stand-alone component, where you use a separate `dah.cfg` file to configure the DAH. However, in simple testing and training setups you can configure the DAH as part of a unified IDOL Server, by using the `idolserver.cfg` file.

For more details about unified and component setups, refer to the *IDOL Getting Started Guide*.

If you use a unified configuration, use the `[DistributionSettings]` section of the `idolserver.cfg` file to enter the configuration options that normally appear in the `[Server]` section of the `dah.cfg` file for a stand-alone configuration. All other section names are the same in both configuration files.

There are three ways to configure the DAH child servers, `DistributedEngines`, `[DAHEngines]` and `[DistributionIDOLServers]`.

In a stand-alone DAH, you can use any of these options. However, you must use the same section format for all your child servers (that is, you must not specify some child servers in one type of sections, and some in another). The most common method is to use the `DistributedEngines` parameter with the `[DistributedEngineN]` child server configuration sections.

In a unified IDOL Server configuration, the section that you use depends on your system setup:

- If you use identical child servers for the DIH and the DAH, use `[DistributionIDOLServers]` with `[IDOLServerN]` child server configuration sections. For example, you might have DIH and DAH both sending actions directly to the Content components.
- If you use different child server configurations for the DIH and the DAH, use `[DAHEngines]` with `[DAHEngineN]` child server configuration sections. For example, you might be using a tiered set up, so that the DAH child servers are child DAH components, while the DIH child servers are child DIH components.

  In this case, you configure the DIH by using the `[DIHEngines]` with `[DIHEngineN]` sections. For more information, refer to the *Distributed Index Handler Administration Guide*.

The configuration examples in this guide generally consider DAH as a stand-alone component, with its own configuration file.

Related Topics

- Add Child Servers to the Distributed Action Handler, on page 36

Modify Configuration Parameter Values

You modify Distributed Action Handler configuration parameters by directly editing the parameters in the configuration file. When you set configuration parameter values, you must use UTF-8.

CAUTION: You must stop and restart Distributed Action Handler for new configuration settings to take effect.

This section describes how to enter parameter values in the configuration file.

Enter Boolean Values

The following settings for Boolean parameters are interchangeable:

```
TRUE = true = ON = on = Y = y = 1
```

```
FALSE = false = OFF = off = N = n = 0
```

Enter String Values

To enter a comma-separated list of strings when one of the strings contains a comma, you can indicate the start and the end of the string with quotation marks, for example:

ParameterName=**cat,dog,bird,"wing,beak",turtle**

Alternatively, you can escape the comma with a backslash:

ParameterName=**cat,dog,bird,wing\,beak,turtle**

If any string in a comma-separated list contains quotation marks, you must put this string into quotation marks and escape each quotation mark in the string by inserting a backslash before it. For example:

ParameterName=**"<font face=\"arial\" size=\"+1\"><b>","<p>"**

Here, quotation marks indicate the beginning and end of the string. All quotation marks that are contained in the string are escaped.

Include an External Configuration File

You can share configuration sections or parameters between ACI server configuration files. The following sections describe different ways to include content from an external configuration file.

You can include a configuration file in its entirety, specified configuration sections, or a single parameter.

When you include content from an external configuration file, the `GetConfig` and `ValidateConfig` actions operate on the combined configuration, after any external content is merged in.

In the procedures in the following sections, you can specify external configuration file locations by using absolute paths, relative paths, and network locations. For example:

```
../sharedconfig.cfg
K:\sharedconfig\sharedsettings.cfg
\\example.com\shared\idol.cfg
file://example.com/shared/idol.cfg
```

Relative paths are relative to the primary configuration file.

> **NOTE:** You can use nested inclusions, for example, you can refer to a shared configuration file that references a third file. However, the external configuration files must not refer back to your original configuration file. These circular references result in an error, and Distributed Action Handler does not start.
>
> Similarly, you cannot use any of these methods to refer to a different section in your primary configuration file.

Include the Whole External Configuration File

This method allows you to import the whole external configuration file at a specified point in your configuration file.

To include the whole external configuration file

1. Open your configuration file in a text editor.
2. Find the place in the configuration file where you want to add the external configuration file.
3. On a new line, type a left angle bracket (<), followed by the path to and name of the external configuration file, in quotation marks (""). You can use relative paths and network locations. For example:

```
< "K:\sharedconfig\sharedsettings.cfg"
```

4. Save and close the configuration file.

Include Sections of an External Configuration File

This method allows you to import one or more configuration sections (including the section headings) from an external configuration file at a specified point in your configuration file. You can include a whole configuration section in this way, but the configuration section name in the external file must exactly match what you want to use in your file. If you want to use a configuration section from the external file with a different name, see Merge a Section from an External Configuration File, on the next page.

To include sections of an external configuration file

1. Open your configuration file in a text editor.
2. Find the place in the configuration file where you want to add the external configuration file section.
3. On a new line, type a left angle bracket (<), followed by the path of the external configuration file, in quotation marks (""). You can use relative paths and network locations. After the configuration file path, add the configuration section name that you want to include. For example:

   ```
   < "K:\sharedconfig\extrasettings.cfg" [License]
   ```

   > **NOTE:** You cannot include a section that already exists in your configuration file.

4. Save and close the configuration file.

Include Parameters from an External Configuration File

This method allows you to import one or more parameters from an external configuration file at a specified point in your configuration file. You can import a single parameter or use wildcards to specify multiple parameters. The parameter values in the external file must match what you want to use in your file. This method does not import the section heading, such as `[License]` in the following examples.

To include parameters from an external configuration file

1. Open your configuration file in a text editor.
2. Find the place in the configuration file where you want to add the parameters from the external configuration file.
3. On a new line, type a left angle bracket (<), followed by the path of the external configuration file, in quotation marks (""). You can use relative paths and network locations. After the configuration file path, add the name of the section that contains the parameter, followed by the parameter name. For example:

   ```
   < "license.cfg" [License] LicenseServerHost
   ```

To specify a default value for the parameter, in case it does not exist in the external configuration file, specify the configuration section, parameter name, and then an equals sign (=) followed by the default value. For example:

```
< "license.cfg" [License] LicenseServerHost=localhost
```

You can use wildcards to import multiple parameters, but this method does not support default values. The * wildcard matches zero or more characters. The ? wildcard matches any single character. Use the pipe character | as a separator between wildcard strings. For example:

```
< "license.cfg" [License] LicenseServer*
```

4. Save and close the configuration file.

Merge a Section from an External Configuration File

This method allows you to include a configuration section from an external configuration file as part of your Distributed Action Handler configuration file. For example, you might want to specify a standard SSL configuration section in an external file and share it between several servers. You can use this method if the configuration section that you want to import has a different name to the one you want to use.

To merge a configuration section from an external configuration file

1. Open your configuration file in a text editor.
2. Find or create the configuration section that you want to include from an external file. For example:

   ```
   [SSLOptions1]
   ```

3. After the configuration section name, type a left angle bracket (<), followed by the path to and name of the external configuration file, in quotation marks (""). You can use relative paths and network locations. For example:

   ```
   [SSLOptions1] < "../sharedconfig/ssloptions.cfg"
   ```

   If the configuration section name in the external configuration file does not match the name that you want to use in your configuration file, specify the section to import after the configuration file name. For example:

   ```
   [SSLOptions1] < "../sharedconfig/ssloptions.cfg" [SharedSSLOptions]
   ```

   In this example, Distributed Action Handler uses the values in the `[SharedSSLOptions]` section of the external configuration file as the values in the `[SSLOptions1]` section of the Distributed Action Handler configuration file.

   > **NOTE:** You can include additional configuration parameters in the section in your file. If these parameters also exist in the imported external configuration file, Distributed Action Handler uses the values in the local configuration file. For example:
   >
   > ```
   > [SSLOptions1] < "ssloptions.cfg" [SharedSSLOptions]
   > ```

```
SSLCACertificatesPath=C:\IDOL\HTTPConnector\CACERTS\
```

4. Save and close the configuration file.

The Distributed Action Handler Configuration File

The DAH configuration file contains settings that determine the basic details that the DAH needs to run. It includes location and port details for the DAH and its child servers (IDOL Servers that it connects to), licensing details, logging details, and several settings that control behavior, such as which mode the DAH runs in and how it handles actions.

The settings are in the configuration file `dah.cfg`, located in your DAH installation directory. You modify these settings to customize the DAH according to your requirements. For details on all available configuration parameters, refer to the *Distributed Action Handler Reference*.

Related Topics

- Display Online Help, below

Display Online Help

You can display the Distributed Action Handler Reference by sending an action from your web browser. The Distributed Action Handler Reference describes the actions and configuration parameters that you can use with Distributed Action Handler.

For Distributed Action Handler to display help, the help data file (`help.dat`) must be available in the installation folder.

To display help for Distributed Action Handler

1. Start Distributed Action Handler.
2. Send the following action from your web browser:

   `http://`*`host`*`:`*`port`*`/action=Help`

   where:

   `host` is the IP address or name of the machine on which Distributed Action Handler is installed.

   `port` is the ACI port by which you send actions to Distributed Action Handler (set by the `Port` parameter in the `[Server]` section of the configuration file).

   For example:

   `http://12.3.4.56:9000/action=help`

Configuration File Sections

The DAH configuration file contains several sections that represent different areas that you can configure. For details on all available configuration parameters, refer to the *Distributed Action Handler Reference*.

Related Topics

- Display Online Help, on the previous page

[ACIEncryption] Section

You can encrypt communications between the DAH and other ACI servers (for example, IDOL Servers and License Server) by configuring an `[ACIEncryption]` section in each of the application configuration files. For example:

```
[ACIEncryption]
CommsEncryptionType=GSS
ServiceName=Kerberos
```

[AuthorizationRoles] Section

The `[AuthorizationRoles]` defines roles that enable a particular set of actions for particular clients, SSL identities, and GSS principals. You must create a subsection for each authorization role that you define in the `[AuthorizationRoles]` configuration section.

For example:

```
[AuthorizationRoles]
0=AdminRole
1=UserRole

[AdminRole]
StandardRoles=Admin,Index,ServiceControl
Clients=localhost
SSLIdentities=admin.example.com

[UserRole]
StandardRoles=Query,ServiceStatus
SSLIdentities=admin.example.com,userserver.example.com
```

[DistributedCommand*N*] Section

The `[DistributedCommand`*N*`]` sections contain settings for actions that DAH must send to all child servers. Each section specifies the action, templates, and parameters for one action. You must number the distributed command sections in consecutive order, starting from 0 (zero). To use distributed commands, you must also set `DistributedCommands` in the `[Server]` section to the number of `[DistributedCommands`*N*`]` configuration sections that you create.

NOTE: If you want to use distributed commands in the DAH, you must configure and use XSL templates. Set `XSLTemplates` to **`True`** in the `[Server]` section. You can then set the `Templates` parameter for each `DistributedCommand`*N* section to use an XSL template. The IDOL Server installation for DAH includes a `templates` directory, which contains some example templates.

```
[DistributedCommand0]
Action=Query
Template=query_storedstatedetail.xsl
RequiredParams=2
ParamName0=storestate
ParamValue0=True
ParamName1=storedstatedetail
ParamValue1=True

[DistributedCommand1]
Action=queueinfo
Template=queueinfo.xsl
```

[DistributedEngine*N*] Section

The `[DistributedEngine`*N*`]` section contains settings that determine the details of individual child servers to which the DAH distributes actions. For example:

```
[DistributedEngine0]
Host=localhost
Port=9100
Type=IDOL

[DistributedEngine1]
Host=Galileo
Port=9100
Type=IDOL
DefaultRelevance=75
QueryFormat=PQF
RequiresAuthentication=True
ResponseFormat=XML
SecurityKeys=12,34,6567,999
SecurityType=Federated
```

NOTE: You must set the `DistributedEngines` parameter in the `[Server]` section to the number of child servers that you require.

[Federated] Section

The `[Federated]` section contains configuration settings for federated search, including the number of defined federated modules, their types, and the paths to their libraries. For example:

```
[Federated]
Number=1
```

```
DefineTypeCSVs0=Z39.50,K2
LibraryPath0=federated.dll
```

[License] Section

The `[License]` section contains licensing details which you must not change. For example:

```
LicenseServerHost=127.0.0.1
LicenseServerACIPort=20000
LicenseServerTimeout=600000
LicenseServerRetries=1
```

[Logging] Section

The `[Logging]` section lists the log streams that you set up to create separate log files for different log message types (action and application). It also contains a subsection for each of the listed log streams, in which you configure the settings that determine how to log each stream. For example:

```
[Logging]
LogDirectory=./logs
0=ACTION_LOG_STREAM
1=APP_LOG_STREAM

[ACTION_LOG_STREAM]
LogFile=action.log
LogTime=True
LogEcho=True
LogMaxSizeKBs=1024
LogTypeCSVs=action
LogLevel=full
LogExpireAction=datestamp

[APP_LOG_STREAM]
LogFile=application.log
LogTime=True
LogEcho=False
LogMaxSizeKBs=1024
LogTypeCSVs=application
LogLevel=full
LogExpireAction=previous
```

[MultiStageQuery] Section

The `[MultiStageQuery]` section contains settings for stage information for multistage querying. This section applies only to the DAH running in non-mirror mode. You configure the number of stages and specify the VDBs for each stage of the query. For example:

```
[MultiStageQuery]
Stages=3
```

```
0=db0
1=db1
2=db2,db3
```

[QueryTemplate] Section

The `[QueryTemplate]` section contains settings to specify the XSL templates to use when merging responses from child servers.

```
[QueryTemplate]
QueueInfo=queueinfo.xsl
ProcessInfo=processinfo.xsl
```

[Server] Section

The `[Server]` section contains general settings. For example:

```
[Server]
MirrorMode=True
Port=9060
TimeBetweenRetries=60000
DistributionMethod=0
XSLTemplates=False
DistributedEngines=2
VirtualDatabases=2
```

[Service] Section

The `[Service]` section contains settings that determine which machines can use and control the IDOL Server service. For example:

```
[Service]
ServicePort=9061
```

[StateChangingActions] Section

If you run the DAH in mirror mode, the `[StateChangingActions]` section contains settings that determine how the DAH handles actions that change the state of its children. For example:

```
[StateChangingActions]
RestrictSCAs=False
MapsCategoryIDs=True
MaxQueueSize=1000000
CustomLibraryCSVs=convertusers
MaxQueuedSCARetries=3
```

[SSLOption*N*] Section

The [SSLOption*N*] section contains settings that determine incoming or outgoing SSL connections between the DAH and other servers. For example:

```
[SSLOption0]
SSLMethod=TLSV1.3
SSLCertificate=host1.crt
SSLPrivateKey=host1.key

[SSLOption1]
SSLMethod=TLSV1.3
SSLCertificate=host2.crt
SSLPrivateKey=host2.key
SSLPrivateKeyPassword=sample1XQ
SSLCheckCommonName=True
```

> **NOTE:** You must create an SSLOption section for each unique value set by the SSLConfig parameter in the [Server], [DistributedEngine*N*], [IDOLServer*N*], or [DAHEngine*N*] section.

[VDB*N*] Section

If you run the DAH in non-mirror mode, you must create a [VDB*N*] section for each of the virtual databases that you want to create. Use this section to specify the database details. For example:

```
[VDB0]
DbName=AllNews
Type=Combinator
MapsTo=0:Today,2:News,3:CurrentAffairs

[VDB1]
DbName=Archive
Type=Distributor
MapsTo=0:Archive,1:Library
```

> **NOTE:** You must set the VirtualDatabases parameter in the [Server] section to the number of virtual databases that you require.

Example Configuration Files

This section shows examples of configuration files for two stand-alone DAH instances in different configurations.

Configuration for a Distributed Action Handler Running in Mirror Mode

```
[Service]
ServicePort=9061
```

```
[Server]
MirrorMode=True
Port=9060
TimeBetweenRetries=60000
DistributionMethod=0
DistributedEngines=2

[DistributedEngine0]
Host=testserver1
Port=9100
Type=IDOL

[DistributedEngine1]
Host=testserver2
Port=9100
Type=IDOL

[Logging]
LogDirectory=./logs
0=APP_LOG_STREAM

[APP_LOG_STREAM]
LogFile=application.log
LogTime=True
LogEcho=False
LogMaxSizeKBs=1024
LogTypeCSVs=application
LogLevel=full
LogExpireAction=previous
```

Configuration for a Distributed Action Handler Running in Non-Mirror Mode

```
[Service]
ServicePort=9061

[Server]
MirrorMode=False
Port=9060
TimeBetweenRetries=60000
VirtualDatabases=3
ShowVDBInfo=True
DistributedEngines=3

[VDB0]
DbName=News
Type=Distributor
```

```
MapsTo=0:News,1:News,2:News

[VDB1]
DbName=Accounts
Type=Combinator
MapsTo=0:Personnel,1:FinancialDocs

[VDB2]
DbName=ProjectManagers
Type=Combinator
MapsTo=0:Personnel,2:ProjectPlans

[DistributedEngine0]
Host=testserver1
Port=9100
Type=IDOL

[DistributedEngine1]
Host=testserver2
Port=9100
Type=IDOL

[DistributedEngine2]
Host=testserver3
Port=9100
Type=IDOL

[Logging]
0=ACTION_LOG_STREAM
1=APPLICATION_LOG_STREAM

[ACTION_LOG_STREAM]
LogFile=action.log
LogTime=True
LogEcho=True
LogMaxSizeKBs=1024
LogTypeCSVs=action
LogLevel=normal
LogExpireAction=datestamp

[APPLICATION_LOG_STREAM]
LogFile=application.log
LogTime=True
LogEcho=False
LogMaxSizeKBs=2048
LogTypeCSVs=application
LogLevel=normal
LogExpireAction=datestamp
```

Add Child Servers to the Distributed Action Handler

You can add child servers to the DAH either by altering the configuration file, or by using an action.

Add Child Servers to the Configuration File

You configure distribution child servers differently for a stand-alone DAH setup and a unified DAH setup where DAH is integrated with IDOL Server.

To add a distribution child server for stand-alone DAH operation

1. Open the DAH configuration file in a text editor.
2. Find the `[Server]` section.
3. In the `[Server]` section, set the `DistributedEngines` parameter to the number of distribution child servers that you require.
4. Add a new `[DistributedEngine`*N*`]` section for each distribution child server that you require, in consecutive order starting at 0 (zero). For example, for two child servers:

   ```
   [DistributedEngine0]
   Host=testserver1
   Port=9100
   Type=IDOL

   [DistributedEngine1]
   Host=testserver2
   Port=9100
   Type=IDOL
   ```

   The above example shows two child IDOL servers. To add a new IDOL server, increase the `DistributedEngines` parameter in the `[Server]` section by one and add a new `[DistributedEngine`*N*`]` section for the new child server:

   ```
   [DistributedEngine2]
   Host=testserver3
   Port=9100
   Type=IDOL
   ```
5. Save and close the configuration file.
6. Restart the DAH for your changes to take effect.

To add a distribution child server for integrated DAH operation in which DAH and DIH are configured together

1. Open the IDOL configuration file in a text editor.
2. In the `[DistributionIDOLServers]` section, set the `Number` parameter to the number of distribution child servers that you require.
3. In the `[DistributionIDOLServers]` section, add a new `[IDOLServer`*N*`]` section for each

distribution child server that you require, in consecutive order starting at 0 (zero). For example, for two child servers:

```
[DistributionIDOLServers]
Number=2

[IDOLServer0]
Host=localhost
Port=9100

[IDOLServer1]
Host=localhost
Port=9200
```

The above example shows two child servers. To add a new child server, increase the `Number` parameter in the `[DistributionIDOLServers]` section by one and add a new `[IDOLServer`*N*`]` section for the new child server:

```
[IDOLServer2]
Host=localhost
Port=9300
```

4. Save and close the configuration file.
5. Restart the IDOL Server for your changes to take effect.

To add a distribution child server for an integrated DAH operation in which DAH and DIH are configured separately

1. Open the IDOL Server configuration file in a text editor.
2. In the `[DAHEngines]` section, set the `Number` parameter to the number of distribution child servers that you require.
3. In the `[DAHEngines]` section, add a new `[DAHEngine`*N*`]` section for each distribution child server that you require, in consecutive order starting at 0 (zero). For example, for two child servers:

```
[DAHEngines]
Number=2

[DAHEngine0]
Host=localhost
Port=9100

[DAHEngine1]
Host=localhost
Port=9200
```

The above example shows two child servers. To add a new child server, increase the `Number` parameter in the `[DAHEngines]` section by one and add a new `[DAHEngine`*N*`]` section for the new child server:

```
[DAHEngine2]
Host=localhost
Port=9300
```

4. Save and close the configuration file.
5. Restart the IDOL Server for your changes to take effect.

Add Child Server Groups

In non-mirror mode, you can configure DAH with child server groups. Each child server group contains a set of mirrored child servers. DAH contacts only one of the child servers in the mirror group for each action, according to your configured `DistributionMethod`.

> **NOTE:** If you use child server groups, you cannot also use virtual databases.

To add a child server mirror to an existing child server

1. Open the DAH configuration file in a text editor.
2. Find the section where you configure your child servers (`[DAHEngine`*N*`]`, `[DistributedEngine`*N*`]`, or `[IDOLServer`*N*`]`).
3. In the `Host` parameter, add the host names or IP addresses of the mirrored child servers. Separate multiple names with commas.

   ```
   Host=host1,host2
   ```

4. In the `Port` parameter, add the ports of the mirrored child servers, separated with commas. The first port must correspond to the first value in the `Host` parameter, the second port must correspond to the second value in the `Host` parameter, and so on.

   ```
   Port=9100,9100
   ```

5. If you use the `HostPortAlias`, `Krb5Service`, or `Krb5Realm` configuration parameters, update these parameters to specify a value for each mirror.

   ```
   HostPortAlias=12.3.4.56:9100,23.45.67.12:9100
   Krb5Realm=COMPANY.COM,COMPANY.COM
   Krb5Service=IDOL,IDOL
   ```

6. (Optional) If you use the `Weight`, `Priority`, or `SSLConfig` parameters, you can specify a value for each child server. If you use a single value, all child servers in the group use the same value.

   ```
   Weight=75,25
   Priority=4
   SSLConfig=SSLOption3
   ```

7. If you have set `UseEngineAlias` to **`True`** in the `[Server]` section, add the `MirrorName` configuration parameter to the child server configuration. Set this parameter to a comma-separated list of aliases. You must configure an alias for each child server.

   ```
   MirrorName=childserver1,childserver2
   ```

8. Save and close the configuration file.
9. Restart the DAH for your changes to take effect.

Add Child Servers with an Action

You can add child servers by using the `EngineManagement` action, with the `EngineAction` parameter set to **`EngineAdd`**. If you use this method, you must also specify the `EngineHost` and `EnginePort` parameters.

For example:

```
action=EngineManagement&EngineAction=EngineAdd&EngineHost=server2&EnginePort=9100
```

This action adds the new server to the configuration file in either the `[DistributedEngine`*`N`*`]`, `[IDOLServer`*`N`*`]`, or `[DAHEngine`*`N`*`]` section, depending on how the system is set up.

Remove a Child Server from the Distributed Action Handler

You can remove a distribution child server from the DAH when the DAH is running either as a stand-alone DAH, or a DAH integrated with IDOL. The procedure is different for each case.

To remove a distribution child server from a stand-alone DAH

1. Open the DAH configuration file in a text editor.
2. Find the `[DistributedEngines`*`N`*`]` section for the distribution child server that you no longer require, and delete it.
3. In the `[Server]` section, decrease the `DistributedEngines` parameter setting by one.
4. Make sure that the remaining `[DistributedEngines`*`N`*`]` sections (which list the remaining child servers) are listed consecutively, starting from 0 (zero).
5. Save and close the configuration file.
6. Restart the DAH for your changes to take effect.

To remove a distribution child server from an integrated DAH in which DAH and DIH are configured together

1. Open the DAH configuration file in a text editor.
2. Find the `[IDOLServer`*`N`*`]` section for the distribution child server that you no longer require, and delete it.
3. In the `[DistributionIDOLServers]` section, decrease the `Number` parameter by one.
4. Make sure that the remaining `[IDOLServer`*`N`*`]` sections (which list the remaining child servers) are listed consecutively, starting from 0 (zero).
5. Save and close the configuration file.
6. Restart the DAH for your changes to take effect.

To remove a distribution child server from an integrated DAH in which DAH and DIH are configured separately

1. Open the DAH configuration file in a text editor.
2. Find the `[DAHEngineN]` section for the distribution child server that you no longer require, and delete it.
3. In the `[DAHEngines]` section, decrease the `Number` parameter by one.
4. Make sure the remaining `[DAHEngineN]` sections (which list the remaining child servers) are listed consecutively, starting from 0 (zero).
5. Save and close the configuration file.
6. Restart the DAH for your changes to take effect.

Run the Distributed Action Handler in Mirror Mode

In mirror mode, the DAH distributes ACI actions to identical IDOL Servers (that is, all the IDOL Servers have the same configuration and contain the same data).

DAH can distribute all actions in mirror mode. For details of the available actions in IDOL Server, refer to the *IDOL Server Reference*.

Related Topics

- Display Online Help, on page 28

To run the DAH in mirror mode

1. Open the DAH configuration file in a text editor.
2. Find the `MirrorMode` setting in the `[Server]` section, and set it to **`True`**.
3. In the `[Server]` section, set the `DistributionMethod` setting to one of the following options. This setting determines how the DAH distributes actions to the connected IDOL Servers:

   - `DistributionMethod=0`

     Failover distribution method. The DAH forwards incoming actions to the first IDOL server that you list in the DAH configuration file `[DistributedEngineN]` section. This server is the primary server.

     If this IDOL Server stops responding for any reason (for example, because of a hardware failure or network outage), the DAH marks it as down. It switches to the next IDOL Server (the second one listed in the `[DistributedEngineN]` section), which becomes the primary server, and so on, and seamlessly continues to service client actions. You can also manually set the primary server by using the `EngineManagement` action.

     If the IDOL Server comes online again (for example, because the hardware has been rebooted or the network connection repaired), the DAH adds it to the list of active IDOL Servers. However, it continues to use the new primary server unless that server stops responding, or unless you manually change the primary server.

   - `DistributionMethod=1`

Load balancing distribution method. The DAH assigns each incoming action to just one of the connected IDOL Servers (using a cumulative predictive algorithm that spreads the action load efficiently). When this IDOL Server responds with a result, the DAH forwards it to the client software.

If an IDOL Server stops responding for any reason (for example, because of a hardware failure or network outage), the DAH marks it as down. It assigns incoming actions only to the IDOL Servers that are running, saving the time that it takes to attempt to communicate with the failed IDOL Server.

If the IDOL Server comes online again (for example, because the hardware has been rebooted or the network connection repaired), the DAH adds it to the list of active IDOL Servers. The IDOL Server again becomes a valid choice for actions.

- `DistributionMethod=2`

  View documents by reference distribution method. The DAH uses the distribute by reference distribution method. You can use this method only to distribute `View`, `GetLink`, and `ViewGetDocInfo` actions in mirror mode. The DAH distributes `View` and `GetLink` actions between View servers based on the reference of the document to view. Actions that request the same document are then always sent to the same View server.

  When View server first receives a request to view a document, it caches any embedded images in the result document. When you distribute between View servers, using distribute by reference ensures that each document is cached in only one View server. The cached image is then used in subsequent requests for the same document.

4. Save and close the configuration file.
5. Restart the DAH for your changes to take effect.

Fast Mirror Mode

When you use the DAH only for requesting information from child servers, you can run it in fast mirror mode to improve performance.

In fast mirror mode, DAH does not perform any extra processing on actions. It simply distributes actions between child servers, either with load balancing or failover. In this case, you can use DAH only for requesting information from child servers.

When running the DAH in fast mirror mode:

- You must not send state-changing actions to the DAH. For example, you must not use the action parameters `Delete`, `State`, `StoreState` and so on.
- You must not use actions that require DAH to perform extra processing on actions. For example, you must not use the action parameters `Template`, `EncryptResponse`, `Output` and so on.
- You must not use action parameters that use document IDs (such as the `ID` parameter).
- You must not use distributed actions (actions sent to all child servers) or asynchronous actions.

To run the DAH in fast mirror mode

1. Open the DAH configuration file in a text editor.

2. In the `[Server]` section, set the `MirrorMode` parameter to **`True`**.
3. In the `[Server]` section, set the `FastMirrorMode` parameter to **`True`**.
4. In the `[Server]` section, set the `DistributionMethod` setting to determine how the DAH distributes the actions that it receives to the connected IDOL Servers.

   For details of the options available for the `DistributionMethod` parameter, see Run the Distributed Action Handler in Mirror Mode, on page 40.
5. Save and close the configuration file.
6. Restart the DAH for your changes to take effect.

Set the Primary Server

In mirror mode, when you are using the failover distribution method, the current child server that DAH sends actions to is the primary server. Similarly, in non-mirror mode, when you have child server groups set up to use the failover distribution method, the child server in each group that DAH sends actions to is the primary server.

By default, DAH selects the first child server listed in the configuration file (either the first `[DistributedEngine`*`N`*`]` section, or the first child server listed in the `Host` parameter for a server group). If this server stops responding, DAH switches to the next configured server, which becomes the primary server.

You can find the current primary server by sending an `EngineManagement` action. This action returns a `<primary_engine>` tag, which shows the ID of the current primary server. To find the primary server at any time, you can send the `EngineManagement` action with the `EngineAction` parameter set to **`ShowStatus`**.

When the first server comes online again, DAH does not automatically switch back. If the server was offline for some time, it might not contain the latest data, so you might want to wait for the server to catch up with indexing jobs. At this point, you can manually change the primary server, by using the `EngineManagement` action, with the `EngineAction` parameter set to **`SetPrimaryEngine`**.

- In mirror mode, you can specify the child server to use by adding the `EngineID` or `EngineName` parameter.

  For example:

  ```
  action=EngineManagement&EngineAction=SetPrimaryEngine&EngineID=0
  ```

  This action sets the child server with ID **`0`** as the primary server. The ID is the number of the child server as described in the `[DistributedEngine`*`N`*`]`, `[IDOLServer`*`N`*`]`, or `[DAHEngine`*`N`*`]` section.
- In non-mirror mode, you can specify the child server to use by adding the `PrimaryEngines` parameter. Specify each primary servers in the following format:

  `LogicalServerNumber``:`*`ServerID`*

  where:

  - *`LogicalServerNumber`* is the ID of the server group. This value is the number of the configuration section that contains the details for this server group.
  - *`ServerID`* is the engine ID of the child server that you want to make the primary server of this

group.

You can find the current primary server, and the IDs of the servers that belong to each server group by sending an `EngineManagement` action. The `<logical_engines>` tag lists the child server IDs that correspond to each configured server group (logical engine).

To set the primary server for multiple server groups, separate each *`LogicalServerNumber`*`:`*`ServerID`* pair with a comma.

For example:

```
action=EngineManagement&EngineAction=SetPrimaryEngine&PrimaryEngines=0:2,1:5
```

This action sets the child server with ID **`2`** to be the primary server for the server group with configured ID **`0`**, and sets the child server with ID **`5`** to be the primary server for the server group with ID **`1`**.

Run the Distributed Action Handler in Non-Mirror Mode

In non-mirror mode, the IDOL Servers that the DAH distributes ACI actions to are different (that is, each IDOL Server is configured differently and contains different data). If you run the DAH in non-mirror mode, you must set up virtual databases, which can be of the following types:

- **Combinator**. The virtual database forwards an action to all the databases that it represents. It collates and sorts the results before it returns them.
- **Distributor**. The virtual database forwards an action according to the selected distribution method to one of the databases it represents:
  - **Load balancing**. Each distributor VDB assigns each incoming action to just one of the databases that it maps to. When this database responds with a result, the DAH forwards it to the client software.

    If an IDOL Server stops responding for any reason (for example, because of a hardware failure or network outage), the DAH marks it as down. It assigns incoming actions only to the databases on IDOL Server installations that are running, saving the time that it takes to attempt to communicate with the failed database.
  - **Failover**. Each distributor VDB forwards incoming actions to the first database that is listed for the virtual database `MapsTo` parameter. If this database stops responding for any reason (for example, because of a hardware failure or network outage), the DAH marks it as down and switches to the next database (the second one listed for the `MapsTo` parameter) and so on, and seamlessly continues to service client actions.

Virtual databases can map to IDOL Server databases or to other VDBs that you set up for the DAH.

> **NOTE:** When you implement a non-mirror mode architecture in which VDBs map to other VDBs, set up combinator VDBs to map only to other VDBs. Distributor virtual databases normally map directly to the databases on the servers. (That is, the combinator VDB combines results from distributor VDBs; a distributor VDB must not distribute queries to combinator VDBs.)

For details of the actions that the DAH can distribute in non-mirror mode, refer to the *Distributed Action Handler Reference*.

Related Topics

- Display Online Help, on page 28

To run the DAH in non-mirror mode

1. Open the DAH configuration file in a text editor.
2. Find the `MirrorMode` setting in the `[Server]` section and set it to **`False`**.
3. In the `[Server]` section, set the `VirtualDatabases` setting to the total number of virtual databases that you want to create. You can create the following virtual database types:
   - **Combinator**. The virtual database forwards an action to all the IDOL Server databases that it represents. It collates and sorts the results before it returns them.
   - **Distributor**. The virtual database forwards an action to one of the IDOL Server databases that it represents. These databases must be identical (that is, all the databases are exact copies of each other and contain the same data). You determine the way that it forwards the action by using the distribution method.
4. To create distributor VDBs, set the `DistributionMethod` setting in the `[Server]` section to one of the following to determine how the distributor VDBs forward the actions that they receive:
   - `DistributionMethod=0`

     Failover distribution method. Each distributor VDB forwards incoming actions to the first IDOL Server database that you list in the virtual database `MapsTo` parameter. If this IDOL Server database stops responding for any reason (for example, because of a hardware failure or network outage), the DAH marks it as down. It switches to the next IDOL Server database (the second one listed in the `MapsTo` parameter) and so on, and seamlessly continues to service client actions.
   - `DistributionMethod=1`

     Load balancing distribution method. Each distributor VDB assigns each incoming action to just one of the IDOL Server databases that it maps to. When this IDOL Server database responds with a result, the DAH forwards it to the client software.

     If an IDOL Server stops responding for any reason (for example, because of a hardware failure or network outage), the DAH marks it as down. It assigns incoming actions only to the IDOL Server databases that are running, saving the time that it takes to attempt to communicate with the failed IDOL Server database.
5. Create a `[VDB`*`N`*`]` section for each of the virtual databases that you want to create. For example:

   ```
   [VDB0]
   [VDB1]
   [VDB2]
   ```

   > **NOTE:** The virtual databases are numbered in consecutive order starting from `0` (zero).
6. Specify the settings to apply to each VDB in the appropriate virtual database section. You can specify the name, type, and the IDOL Server databases that it maps to. For example:

```
[VDB0]
DbName=News
Type=Distributor
MapsTo=0:News,1:News,2:News

[VDB1]
DbName=Accounts
Type=Combinator
MapsTo=0:Personnel,1:FinancialDocs

[VDB2]
DbName=ProjectManagers
Type=Combinator
MapsTo=1:Personnel,Accounts
```

In this example, the VDB configured in `[VDB2]` maps to the `Accounts` VDB configured in `[VDB1]`, as well as to an IDOL Server database specified by `1:Personnel`.

If you add a virtual database that maps to another virtual database, you must ensure that the section number of the virtual database that you add is higher than the section number of the virtual database that you are mapping to. That is, the virtual database in `[VDB2]` can map to the virtual database in `[VDB1]`, but the virtual database in `[VDB1]` must not map to the virtual database in `[VDB2]`. This restriction ensures that virtual database mappings are not circular.

7. Save and close the configuration file.
8. Restart the DAH for your changes to take effect.

Multistage Queries

In non-mirror mode, the DAH supports multistage querying. You can configure the stage information for multistage querying in the DAH configuration file. A multistage query is processed in stages across specified VDBs. The specified VDBs serve the query in stages, until they find the required number of results.

You enable multistage querying by using the `MultiStage` or `MultiStageMinResults` action parameters with `Query` or `SuggestOnText` actions. For details of multistage action parameters, refer to the *IDOL Server Reference*.

Related Topics

- Display Online Help, on page 28

To configure multistage query stage information

1. Open the DAH configuration file in a text editor.
2. Find the `[MultiStageQuery]` section.
3. Add the `Stages` option to specify the total number of stages to use for a multistage query. For example:

```
[MultiStageQuery]
Stages=3
```

4. For each stage, use the *N* option to specify a comma-separated list of VDBs to use for that stage, starting with stage `0` (zero). The DAH processes the query stage by stage, in order. For example:

   ```
   [MultiStageQuery]
   Stages=3
   0=db0
   1=db1
   2=db2,db3
   ```

5. Save and close the configuration file.
6. Restart the DAH for your changes to take effect.

NOTE: If you specify the total number of stages and VDBs to use at each stage by using parameters in the `Query` or `SuggestOnText` action, these settings override the DAH configuration settings for multistage querying.

Set Up SSL Connections

You can configure Secure Socket Layer (SSL) connections between the DAH and other servers. You can configure SSL connections in a combination of different configuration sections:

- `[Server]`. Configure SSL in this section for connections for incoming ACI calls. You can configure this section in either the IDOL Server configuration file or DAH configuration file, depending on whether the system uses a unified or stand-alone setup.
- `[DistributedEngine`*N*`]`. Configure SSL in this section for connections for outgoing ACI calls in a stand-alone setup.
- `[IDOLServer`*N*`]`. Configure SSL in this section for connections for outgoing ACI calls in a unified setup in which you configure DAH and DIH together.
- `[DAHEngine`*N*`]`. Configure SSL in this section for connections for outgoing ACI calls in a unified setup in which you configure DAH and DIH separately.

The configuration section that you modify depends on how you want to manage your connections:

- To establish SSL connections on incoming calls only, use the `[Server]` section.
- To establish SSL connections on outgoing calls only in a stand-alone system, use the `[DistributedEngine`*N*`]` section.
- To establish SSL connections for both incoming and outgoing calls in a unified system, use the `[Server]` section, and either the `[IDOLServer`*N*`]` section or the `[DAHEngine`*N*`]` section.

Configure SSL in a Stand-Alone System

Use the following procedure to configure SSL in a setup where you configure the DAH in the stand-alone DAH configuration file.

To configure an SSL connection in a stand-alone system

1. Open the DAH configuration file in a text editor.
2. Find either the `[Server]` section, the `[DistributedEngine`*N*`]` section, or both.
3. Add the `SSLConfig` setting to specify the section in which you have set the SSL details for the connection, usually `SSLOption`*N*. For example:

   ```
   [Server]
   (other server settings...)
   SSLConfig=SSLOption0

   [DistributedEngine0]
   SSLConfig=SSLOption0

   [DistributedEngine1]
   SSLConfig=SSLOption1
   ```

   In this example, incoming ACI calls and outgoing calls to `DistributedEngine0` share the same SSL configuration, and outgoing calls to `DistributedEngine1` use a different configuration.
4. Create an `[SSLOption`*N*`]` section for each unique `SSLConfig` setting. Each `SSLOption` entry must contain the `SSLMethod`, `SSLCertificate`, and `SSLPrivateKey` parameters. For example:

   ```
   [SSLOption0]
   SSLMethod=TLSV1.3
   SSLCertificate=host1.crt
   SSLPrivateKey=host1.key

   [SSLOption1]
   SSLMethod=TLSV1.3
   SSLCertificate=host2.crt
   SSLPrivateKey=host2.key
   ```
5. Save and close the configuration file.
6. Restart the DAH for your changes to take effect.

Configure SSL in a Unified System

Use the following procedure to configure SSL in a setup where you configure the DAH in the unified IDOL Server configuration file.

To configure an SSL connection in a unified system

1. Open the IDOL Server configuration file in a text editor.
2. Find the `[Server]` section, the `[IDOLServer`*N*`]` or `[DAHEngine`*N*`]` section, or a combination of the `[Server]` section and one of the others.
3. Add the `SSLConfig` setting to specify the section in which you have set the SSL details for the connection, usually `SSLOption`*N*. For example:

   ```
   [Server]
   (other server settings...)
   SSLConfig=SSLOption0

   [IDOLServer0]
   SSLConfig=SSLOption0

   [IDOLServer1]
   SSLConfig=SSLOption1
   ```

   In this example, incoming ACI calls and outgoing calls to `IDOLServer0` share the same SSL configuration, and outgoing calls to `IDOLServer1` use a different configuration.
4. Create an `[SSLOption`*N*`]` section for each unique `SSLConfig` setting. Each `SSLOption` entry must contain the `SSLMethod`, `SSLCertificate`, and `SSLPrivateKey` parameters. For example:

   ```
   [SSLOption0]
   SSLMethod=TLSV1.3
   SSLCertificate=host1.crt
   SSLPrivateKey=host1.key

   [SSLOption1]
   SSLMethod=TLSV1.3
   SSLCertificate=host2.crt
   SSLPrivateKey=host2.key
   ```
5. Save and close the configuration file. Restart the IDOL Server for your changes to take effect.

Set Language Encoding for Results

You can specify the encoding that the DAH uses for the results that it returns.

To set language encoding

1. Open the DAH configuration file in a text editor.
2. Find the `[Server]` section.
3. Add the `DefaultEncoding` setting to specify the encoding that you want the DAH to use for results that it returns, in cases where the query does not specify a language encoding to use. For example:

```
[Server]
(other server settings...)
DefaultEncoding=ASCII
```

4. Add the `LanguageDirectory` parameter to specify the location of the directory that contains the files that the DAH needs to encode results. By default, the files are in the `langfiles` directory located in the main DAH installation directory. For example:

   ```
   [Server]
   (other server settings...)
   DefaultEncoding=ASCII
   LanguageDirectory=./langfiles
   ```

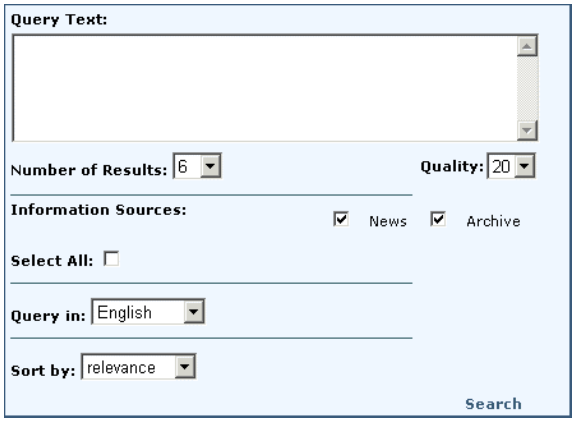
   > **NOTE:** If the DAH cannot find the language directory that you specify, it returns results in UTF-8.

5. Save and close the configuration file.
6. Restart the DAH for your changes to take effect.

XSLT Templates

You can use XSLT templates to format the results that the DAH returns. You can write your own XSLT templates, or you can use the templates that the DAH installation includes in the `acitemplates` directory:

- `QueryForm.tmpl`. You can apply this template to any action (for example, `GetVersion`) to produce a simple HTML query interface. You can use this interface to create queries to send to IDOL Server:



- `GetStatusForm.tmpl`. You can apply this template to the `GetStatus` action to generate HTML-

formatted status information:

Autonomy IDOL Server Version 5.0.4 Build 65249

ACI Port:	9000
Index Port:	9001
Service Port:	9002
Directory:	D:\Autonomy\IDOLserver\IDOL
ACI Threads:	4
Licensed Languages:	English

Documents:	7185
Document Sections:	7185
Committed Sections:	7196
Terms:	77134
Total Terms:	94421
Total Hashes:	1048576
Record Size:	53
Max Occurrences:	273
Terms Per Doc:	50
Suggest Terms:	40

Caches:

Term Cache	
Terms:	10292
Used:	6814 Kb
Max:	102400 Kb
Requests:	17713
Hit Rate:	38

Index Cache	
Terms:	0
Used:	0 Kb
Max:	102400 Kb
Blocks:	1

Databases: 2

No.	Name	Documents	Document Sections	Readonly	Expiry Hours	Expiry Action
1	News	7185	7185	false	-	-
2	Archive	0	0	false	-	-

Enable XSLT Templates

Use the following procedure to configure DAH to use XSLT templates.

To enable XSL templates

1. Open the DAH configuration file in a text editor.
2. Find the `[Server]` section.
3. Set the `XSLTemplates` parameter to **`True`** (if the `[Server]` section does not contain this setting, add it).
4. Save and close the configuration file.
5. Restart the DAH for your changes to take effect.

Apply XSLT Templates to Actions

To apply a template to action output, add the following parameters to the action:

- `Template`. Set this parameter to the name of the template to use to format action output.
- `OutputEncoding`. Set this parameter to **`UTF8`**.

Examples

```
http://localhost:9000/action=GetVersion&Template=QueryForm&OutputEncoding=UTF8
```

This action applies the `QueryForm.tmpl` XSLT template to a `GetVersion` action, to provide an interface for queries.

```
http://localhost:9000/action=GetStatus&Template=GetStatusForm&OutputEncoding=UTF8
```

This action applies the `GetStatusForm.tmpl` XSLT template to a `GetStatus` action.

Customize Logging

You can customize logging by setting up your own *log streams*. Each log stream creates a separate log file in which specific log message types (for example, action, index, application, or import) are logged.

To set up log streams

1. Open the Distributed Action Handler configuration file in a text editor.
2. Find the `[Logging]` section. If the configuration file does not contain a `[Logging]` section, add one.
3. In the `[Logging]` section, create a list of the log streams that you want to set up, in the format *`N=LogStreamName`*. List the log streams in consecutive order, starting from 0 (zero). For example:

   ```
   [Logging]
   LogLevel=FULL
   LogDirectory=logs
   0=ApplicationLogStream
   1=ActionLogStream
   ```

   You can also use the `[Logging]` section to configure any default values for logging configuration parameters, such as `LogLevel`. For more information, see the *Distributed Action Handler Reference*.
4. Create a new section for each of the log streams. Each section must have the same name as the log stream. For example:

   ```
   [ApplicationLogStream]
   [ActionLogStream]
   ```
5. Specify the settings for each log stream in the appropriate section. You can specify the type of logging to perform (for example, full logging), whether to display log messages on the console, the maximum size of log files, and so on. For example:

   ```
   [ApplicationLogStream]
   LogTypeCSVs=application
   LogFile=application.log
   LogHistorySize=50
   LogTime=True
   ```

```
LogEcho=False
LogMaxSizeKBs=1024

[ActionLogStream]
LogTypeCSVs=action
LogFile=logs/action.log
LogHistorySize=50
LogTime=True
LogEcho=False
LogMaxSizeKBs=1024
```

6. Save and close the configuration file. Restart the service for your changes to take effect.

Configure Asynchronous Actions

You can use the DAH to distribute *asynchronous* actions to child servers in mirror mode. When you send an asynchronous action, the child server immediately returns a token and queues the action. To check the status of the action, you send the `QueueInfo` action with the token returned by the child server.

For details of asynchronous actions and the `QueueInfo` action, refer to the documentation for the server you are using.

To use DAH to distribute asynchronous actions

- Add the `AsynchronousCommands` configuration parameter to the DAH configuration file.

  For example:

  ```
  [Server]
  AysnchronousCommands=ingest,tiff
  ```

  In this example, DAH distributes the `ingest` and `tiff` asynchronous actions to its child servers.

Distribute the QueueInfo Action

After you configure asynchronous actions, you can use the `QueueInfo` action to check the status of the action queue or a specific action, and manage the action queue.

To check a specific action, send the `QueueInfo` action with the asynchronous action token returned by the child server. DAH uses the host and port information in the token to send the action to the correct child server.

By default, DAH retrieves this information from the child servers at startup to match the value in the token to its child server configuration.

If DAH cannot retrieve this information, it uses the `HostPortAlias` configuration parameter to apply the child server configuration options to the `QueueInfo` action. DAH matches the `HostPortAlias` information to the child server configuration. For example, using the `HostPortAlias` configuration allows DAH to apply the appropriate SSL settings for the child server.

Find the Alias for a Child Server

If DAH cannot retrieve host and port information from its child servers for any reason, it uses the configured value of `HostPortAlias` to match tokens to its child servers.

The host and port that you specify for a child server in the `HostPortAlias` parameter must exactly match the host and port information in the asynchronous action tokens. This value does not necessarily match the `Host` and `Port` configuration parameters. For example, you might set `Host` to the computer name, and the token might return an IP address.

To find the host and port for a child server

1. Open the DAH configuration file in a text editor.
2. In the `[Logging]` section, set `LogLevel` to **`Full`**.
3. Save and close the configuration file.
4. Restart the DAH for your changes to take effect.
5. After the DAH has started, send a `QueueInfo` action to the DAH. Set the `Token` parameter to an action token for the child server for which you want to find the alias. For example:

   ```
   action=QueueInfo&Token=FOO-123&QueueAction=getstatus
   ```

6. Open the action log. In a default unified configuration, this file is located in the following directory:

   `installdir``/IDOL/logs`

7. Find the log message of the following type:

   `Checking alias for engine host(`*`Host`*`) and port(`*`Port`*`).`

   This *`Host`* and *`Port`* value is the value that you must use in the `HostPortAlias` configuration parameter.

Use HostPortAlias

You can apply a host and port alias:

- individually for each child server by using the `HostPortAlias` configuration parameter.
- for all child servers by using the `UseDefaultHostPortAlias` configuration parameter.

  In this case, if DAH cannot retrieve the child server host and port from the child servers, it matches the host and port information in the ansynchronous action tokens to the `Host` and `Port` parameters for each child server. You can use this option if the configured values exactly match the values in the asynchronous action token.

To set HostPortAlias for individual child servers

1. Open the DAH configuration file in a text editor.
2. Find the `[IDOLServer`*`N`*`]` or `[DAHEngine`*`N`*`]` configuration sections.
3. Set the `HostPortAlias` parameter to the host and port for the child server; separate the

values with a colon. The host must exactly match the host and port in the asynchronous action token.

4. Save and close the configuration file.
5. Restart the DAH for your changes to take effect.

To set HostPortAlias for all child servers

1. Open the DAH configuration file in a text editor.
2. In the `[Server]` section, set the `UseDefaultHostPortAlias` configuration parameter to **`True`**.
3. Save and close the configuration file.
4. Restart the DAH for your changes to take effect.

Related Topics

- Find the Alias for a Child Server, on the previous page

Configure Actions that Return a Binary Response

In mirror mode, you can use the DAH to distribute actions to child servers for actions that return a binary response. For these actions, you must configure the DAH with a list of the actions that return binary data, so that DAH processes and returns the response correctly.

For details of actions that return binary data, refer to the documentation for the server you are using.

To use DAH to distribute actions that return a binary response

- Add the `BinaryCommands` configuration parameter to the DAH configuration file. Set this parameter to a comma-separated list of the actions that return binary data.

  For example:

  ```
  [Server]
  BinaryCommands=CreateImage,EditImage
  ```

  In this example, DAH distributes the `CreateImage` and `EditImage` actions, and returns the binary response from its child servers.

Configure Client Authorization

You can configure Distributed Action Handler to authorize different operations for different connections.

Authorization roles define a set of operations for a set of users. You define the operations by using the `StandardRoles` configuration parameter, or by explicitly defining a list of allowed actions in the `Actions` and `ServiceActions` parameters. You define the authorized users by using a client IP address, SSL identities, and GSS principals, depending on your security and system configuration.

For more information about the available parameters, see the *Distributed Action Handler Reference*.

IMPORTANT: To ensure that Distributed Action Handler allows only the options that you configure in `[AuthorizationRoles]`, make sure that you delete any deprecated *`Role`*`Clients` parameters from your configuration (where *`Role`* corresponds to a standard role name, for example `AdminClients`).

To configure authorization roles

1. Open your configuration file in a text editor.
2. Find the `[AuthorizationRoles]` section, or create one if it does not exist.
3. In the `[AuthorizationRoles]` section, list the user authorization roles that you want to create. For example:

   ```
   [AuthorizationRoles]
   0=AdminRole
   1=UserRole
   ```

4. Create a section for each authorization role that you listed. The section name must match the name that you set in the `[AuthorizationRoles]` list. For example:

   ```
   [AdminRole]
   ```

5. In the section for each role, define the operations that you want the role to be able to perform. You can set `StandardRoles` to a list of appropriate values, or specify an explicit list of allowed actions by using `Actions`, and `ServiceActions`. For example:

   ```
   [AdminRole]
   StandardRoles=Admin,ServiceControl,ServiceStatus

   [UserRole]
   Actions=GetVersion
   ServiceActions=GetStatus
   ```

   NOTE: The standard roles do not overlap. If you want a particular role to be able to perform all actions, you must include all the standard roles, or ensure that the clients, SSL identities, and so on, are assigned to all relevant roles.

6. In the section for each role, define the access permissions for the role, by setting `Clients`, `SSLIdentities`, and `GSSPrincipals`, as appropriate. If an incoming connection matches one of the allowed clients, principals, or SSL identities, the user has permission to perform the operations allowed by the role. For example:

   ```
   [AdminRole]
   StandardRoles=Admin,ServiceControl,ServiceStatus
   Clients=localhost
   SSLIdentities=admin.example.com
   ```

7. Save and close the configuration file.
8. Restart Distributed Action Handler for your changes to take effect.

IMPORTANT: If you do not provide any authorization roles for a standard role, Distributed Action

Handler uses the default client authorization for the role (localhost for `Admin` and `ServiceControl`, all clients for `Query` and `ServiceStatus`). If you define authorization only by actions, Micro Focus recommends that you configure an authorization role that disallows all users for all roles by default. For example:

```
[ForbidAllRoles]
StandardRoles=*
Clients=""
```

This configuration ensures that Distributed Action Handler uses only your action-based authorizations.

Chapter 4: Operate the Distributed Action Handler

You administer and operate the DAH by changing settings in the DAH configuration file, and by sending actions and service actions to it.

Start and Stop the Distributed Action Handler

You can use several different methods to start or stop the DAH.

Start the Distributed Action Handler

The following sections describe the different ways that you can start the DAH.

Before you can start the DAH, you must start the License Server.

Start the DAH on Microsoft Windows

- Double-click the `DAH.exe` file in your component installation directory.
- Start the DAH service from a system dialog box. DAH must be installed as a Windows Service. See Install an IDOL Component as a Service on Windows, on page 16.
  1. Display the Windows **Services** dialog box.
  2. Select the **DAH** service for the component, and click **Start** to start the component.
  3. Click **Close** to close the **Services** dialog box.

  > **TIP:** You can also configure the Windows Service to run automatically when you start the machine.
- Start a component from the command line. For more information, refer to the *IDOL Getting Started Guide*.

Start the DAH on UNIX

- Start the IDOL component service from the command line. The component must be installed as a service. See Install an IDOL Component as a Service on Linux, on page 19 You can use one of the following commands to start the service:

- On systemd Linux platforms:

  ```
  systemctl start DAH
  ```

- On System V Linux platforms:

  ```
  service DAH start
  ```

- On Solaris platforms (using System V):

  ```
  /etc/init.d/DAH start
  ```

> **TIP:** You can also configure the service to run automatically when you start the machine.

- Start the DAH from the command line. For more information, refer to the *IDOL Getting Started Guide*.
- Use the start script (`start-dah.sh`).

> **NOTE:** In most cases, Micro Focus recommends that you use the provided init scripts instead.

Stop the Distributed Action Handler

You can stop the DAH from running in several different ways.

- (All Platforms) Send the `Stop` service action to the component service port:

  `http://`*`host`*`:`*`servicePort`*`/action=stop`

  where *`host`* is the name or IP address of the host on which the DAH is running, and *`servicePort`* is the component service port (which is specified in the `[Service]`section of the Distributed Action Handler configuration file).

- On Windows platforms, when the component is installed as a service, you can use the system dialog box to stop the service:

  1. Display the Windows **Services** dialog box.
  2. Select the **DAH** service, and click **Stop** to stop Distributed Action Handler.
  3. Click **Close** to close the **Services** dialog box.

- On UNIX platforms, when the component is installed as a service, you can run one of the following commands to stop the service:

  - On systemd platforms:

    ```
    systemctl stop DAH
    ```

  - On system V platforms:

    ```
    service DAH stop
    ```

  - On Solaris platforms (using System V):

```
/etc/init.d/DAH stop
```

- On UNIX platforms, you can also use the stop script, `stop-dah.sh`.

Distribute IDOL Actions

The main function of the DAH is to distribute actions to its child servers to query them. The DAH can distribute different actions depending on whether you run it in mirror or non-mirror mode:

- If you run the DAH in mirror mode, it can distribute all ACI server actions. Refer to your product Reference (for example, the *IDOL Server Reference*) for a complete list of actions.
- If you run the DAH in non-mirror mode, it can distribute the following IDOL Server actions:

`DetectLanguage`	`Query`
`DocumentStats`	`Suggest`
`GetContent`	`SuggestOnText`
`GetQueryTagValues`	`Summarize`
`GetTagNames`	`TermExpand`
`GetTagValues`	`TermGetBest`
`Highlight`	`TermGetInfo`
`LanguageSettings`	

In addition, you can use some actions to query the DAH for information about its operation (for example, to return a list of requests that were sent to the DAH, or to return license information). For details of supported actions and action parameters, refer to the *Distributed Action Handler Reference*.

NOTE: For the `LanguageSettings` action, when the child servers have compatible language configurations, DAH returns the child server response, except for the `LanguageDirectory` configuration. The child server language configuration must be the same for all child servers, except for `LanguageDirectory`. If the child server configuration is not compatible, DAH returns the message `Engines returned inconsistent information`.

Return Child Server Information

For the following actions, you can use the `PrintEngine` action parameter to return additional information for each result document that shows the child server that the result comes from.

`GetContent`
`Query`
`Suggest`
`SuggestOnText`

When you set `PrintEngine` to **`True`** for these actions, the response includes an `<autn:engines>` tag for each result, which includes the child server information in the format *`X`*`[.`*`Y`*`.`*`Z`*`...]`, where:

- the number of elements represents the number of levels in your DAH hierarchy
- each element corresponds to the ID of the child server that returned the response. The ID is the number of the child server in the parent configuration file.

For example, if you have a DAH, with a layer of sub-DAHs, and then a layer of child IDOL Servers, a value of `3.2` for `<autn:engines>` represents a document coming from child server 2 of sub-DAH 3 of the top level DAH.

Send Distributed Action Handler Actions

You can send certain actions directly to the DAH, to get information or to manage its queue.

Action	Description
`DAHFlushQueue`	Deletes all queued actions for one or more child IDOL Servers.
`DAHReadQueue`	Displays child server status and action queue information.
`EngineManagement`	Administers child servers dynamically.
`VDBManagement`	Administers virtual databases dynamically.
`TokenManagement`	Administers state tokens for the DAH and its child servers.

There are also several standard actions that the DAH supports. For more details about the supported DAH actions, and the parameters that each action takes, refer to the *Distributed Action Handler Reference*.

Part II: Appendixes

This section includes the following appendixes:

- Query Non-IDOL Servers
- Error Codes

Appendix A: Query Non-IDOL Servers

IDOL component servers use the standard ACI (Autonomy Content Infrastructure) protocol. The DAH also supports querying Z39.50 servers, K2 servers, and Ultraseek servers by using federated search.

Enable Federated Search

Use the following procedure to enable the DAH to use federated search with Z39.50, K2, or Ultraseek query servers.

To enable the DAH for federated search

1. Open the DAH configuration file in a text editor.
2. In the `[Federated]` section of the configuration file, configure the following options for a federated search:
   - Set the `Number` parameter to the number of federated modules to use.
   - Set the `DefineTypeCSVs`*N* parameter to a comma-separated list of the types of query server for each federated module. Do not add spaces after the commas. Number the parameters consecutively, starting at 0 (zero). You can use the following server types:
     - Z39.50 servers must have the type `Z39.50`.
     - K2 servers must have the type `K2`.
     - Ultraseek servers must have the type `Ultraseek`.
   - Set the `LibraryPath`*N* parameter to the library file for each federated module that you configure. Number the parameters consecutively, starting at 0 (zero). The library file provides the federated search functionality and corresponds to the query server types defined for the federated module.

   For example:

   ```
   [Federated]
   Number=1
   DefineTypeCSVs0=Z39.50,K2
   LibraryPath0=federated.dll
   ```
3. Configure each of the query server types defined for the federated search module in either the `[DistributedEngine`*N*`]` section or the `[IDOLServer`*N*`]` section.

   NOTE: Each type of query server has type-specific configuration parameters. Details of these parameters are given in the *Distributed Action Handler Reference*. See Display Online Help, on page 28.

4. Save and close the configuration file.
5. Restart the DAH for your changes to take effect.

An earlier type of configuration for DAH federated search for Z39.50 query servers is still supported. Instead of using the `[Federated]` section and configuring the query server types by using the `[DistributedEngine`*N*`]` section or the `[IDOLServer`*N*`]` section, you can use the following procedure:

1. In the `[Server]` section, set the following parameters:
   - `FederatedLibrary`. Enter the name of the library file that provides the federated search functionality.
   - `FederatedResponseFormat`. Enter the format to use for query responses. It must be a format that the query servers that you want to communicate with accept. The default value is `XML`.
   - `FederatedDefaultRelevance`. Enter the relevance value to display for results that the query servers produce (this is necessary because the servers do not return relevance values for results). The default value is `90`.
   - `FederatedQueryFormat`. Enter **`PQF`** or **`CQL`** to indicate the format to use to query the query servers that you want to communicate with. The default value is `PQF`.

     NOTE: *PQF* (Prefix Query Format) and *CQL* (Common Query Language) are non-IDOL syntax standards. For more information, refer to the external documentation.
   - `FederatedSecurityType`. Enter the name of the federated security type.
   - `FederatedSecurityKeys`. Enter the security string that contains federated information (user name, group, password).
2. In the `[Engines]` section, increase the `Number` setting by one for each of the query servers that you want to query. Use the `Engine`*N* setting to list the servers in consecutive order, and to specify their IP addresses and port numbers. Use the `Type`*N* setting to specify the type of the servers (Z39.50 servers are type 3). For example:

   ```
   [Engines]
   Number=2
   Engine0=123.45.67.89:9949
   Type0=3
   Engine1=123.45.01.23:9787
   Type1=3
   ```

   If a Z39.50 server requires authentication, you must also set the `RequiresAuthentication`*N* setting to **`True`** for this server in the `[Engines]` section. For example:

   ```
   [Engines]
   Number=2
   Engine0=123.45.67.89:9949
   Type0=3
   Engine1=123.45.01.23:9787
   Type1=3
   RequiresAuthentication1=True
   ```

Query Z39.50 Query Servers

After you enable federated search of a Z39.50 query server through the standard ACI (Autonomy Content Infrastructure), you can send `Query` actions in PQF or CQL format to the DAH. The DAH then passes the queries to the query servers it connects to.

You can use the following `Query` action parameters:

- `Text`
- `MaxResults`
- `DatabaseMatch`
- `TotalResults`
- `Start`

For details on these settings, refer to the *IDOL Server Reference*.

Example Queries

PQF:

`http://host:port/action=Query&Text=cat`

`http://host:port/action=Query&Text="Abyssinian Bobtail"`

`http://host:port/action=Query&Text=@and fat cat`

`http://host:port/action=Query&Text=@attr 1=4 @attr4=1 "hot tin roof"`

CQL:

`http://host:port/action=Query&Text=cat`

`http://host:port/action=Query&Text="Abyssinian Bobtail"`

`http://host:port/action=Query&Text=fat and cat`

`http://host:port/action=Query&Text=title="hot tin roof"`

where:

- *host* is the IP address (or host name) of the machine on which the DAH is installed.
- *port* is the port number that clients use to communicate with the DAH (you specify this option in the `Port` setting in the `[Server]` section of the DAH configuration file).

NOTE: PQF (Prefix Query Format) and CQL (Common Query Language) are non-IDOL syntax standards. For more information, refer to the external documentation.

Appendix B: Error Codes

This appendix describes the standard and DAH error codes that the DAH returns.

Standard HTTP Error Codes

The DAH returns the following standard HTTP error codes:

503	Service Unavailable None of the IDOL servers that the DAH is connected to is running.
504	Request Timeout The DAH did not receive a reply from any of the IDOL servers that it sent the request to.

Distributed Action Handler Error Codes

In addition to the standard HTTP error codes, the DAH returns the following DAH error codes. For more details of these errors, look at your DAH log files.

2147437725	SecurityInfo String Has Expired
2147437726	Datastore Error
2147437727	Token Not Found
2147437728	Configuration Error
2147437741	Bad Query Parameter
2147437742	Conflicting Parameters
2147437743	Unsupported Parameter
2147437744	Missing Parameter
2147437759	ACI Connection Error
2147437760	ACI Error
2147437774	An Engine Is Down While State Changing Action Restriction
2147437775	At Least One Engine Returned Error

2147437776	Abridged Query Error
2147437791	Category Move or Rename Disabled
2147437792	State Changing Action Queue Full
2147437806	No Results
2147437807	All Engines Returned Error
2147437808	No Engines Available
2147437823	All VDBs Are Offline
2147437824	VDB Error
2147441868	Missing Required Parameter
2147483362	XML Parsing Error
2147483364	File Not Found
2147483371	Library Interface Error
2147483372	Library Loading Error
2147483374	Invalid Output
2147483377	Operation Failed
2147483381	Internal Error
2147483385	Assertion Failure
2147483386	Bad Parameter Value
2147483389	Not Implemented
2147483390	Memory Error
2147483391	Bad Parameter

Glossary

A

ACI (Autonomy Content Infrastructure)
A technology layer that automates operations on unstructured information for cross-enterprise applications. ACI enables an automated and compatible business-to-business, peer-to-peer infrastructure. The ACI allows enterprise applications to understand and process content that exists in unstructured formats, such as email, Web pages, Microsoft Office documents, and IBM Notes.

ACI Server
A server component that runs on the Autonomy Content Infrastructure (ACI).

ACL (access control list)
An ACL is metadata associated with a document that defines which users and groups are permitted to access the document.

action
A request sent to an ACI server.

active directory
A domain controller for the Microsoft Windows operating system, which uses LDAP to authenticate users and computers on a network.

agent
A process that searches for information about a specific topic. An administrator can create agents for users or allow users to create their own agents.

asynchronous actions
An ACI action that is queued and performed asynchronously (compare with synchronous actions). In an asynchronous action, the server returns a token immediately, but adds the action to a queue. You can use the token to track the progress of the response, and retrieve the results when the action is complete.

authentication
The process of checking user credentials (user names, passwords, and PIN codes) against an IDOL server or external security repository. The authentication process identifies a user, and allows IDOL Server to confirm their access permissions for different documents.

C

category
A set of criteria that define a particular topic, which you can use to categorize documents that contain content relevant to the topic.

Category component
The IDOL Server component that manages categorization and clustering.

child server
An ACI server that the DAH distributes to. The child server is often an IDOL Content component, in which case the DAH can distribute queries between multiple servers. In mirror mode, the DAH can also distribute to multiple identical copies of any ACI server.

combinator database
A DAH virtual database that combines results from several non-identical IDOL Server databases. See Also: virtual database, distributor database.

Community component
The IDOL Server component that manages users and communities.

connector
An IDOL component (for example File System Connector) that retrieves information from a local or remote repository (for example, a file system, database, or Web site).

Connector Framework Server (CFS)
Connector Framework Server processes the information that is retrieved by connectors. Connector Framework Server uses KeyView to extract document content and metadata from over 1,000 different file types. When the information has been processed, it is sent to an IDOL Server or Distributed Index Handler (DIH).

Content component
The IDOL Server component that manages the data index and performs most of the search and retrieval operations from the index.

CQL
Common Query Language, or Contextual Query Language. This format is an external formal language used to communicate with Z39.50 servers.

D

DAH (Distributed Action Handler)
DAH distributes actions to multiple copies of IDOL Server or a component. It allows you to use failover, load balancing, or distributed content.

database
An IDOL Server data pool that stores indexed information. The administrator can set up one or more databases, and specify how to feed data to the databases. By default, IDOL Server contains the databases Profile, Agent, Activated, Deactivated, News, and Archive.

DIH (Distributed Index Handler)
DIH allows you to efficiently split and index extremely large quantities of data into multiple copies of IDOL Server or the Content component. DIH allows you to create a scalable solution that delivers high performance and high availability. It provides a flexible way to batch, route, and categorize the indexing of internal and external content into IDOL Server.

distributor database
A DAH virtual database that retrieves results from several identical IDOL Server databases. For each query, it retrieves results from only one of the identical copies. See Also: virtual database, combinator database

F

fetch
The process of downloading documents from the repository in which they are stored (such as a local folder, Web site, database, or Lotus Domino server), importing them to IDX format, and indexing them into IDOL Server.

fetch task
A group of settings that instruct a connector how to retrieve data from a repository. Connectors can run fetch tasks automatically, or in response to an action.

H

hyperlink
In IDOL, the ability to connect related documents to results, by using suggestions. See Also: suggest

I

IAS (Intellectual Asset Protection System)
An integrated security solution to protect your data. At the front end, authentication checks that users have permission to access the system that contains the result data. At the back end, entitlement checking and authentication combine to ensure that query results contain only documents that the user has permission to see, from repositories that the user has permission to access. For more information, refer to the IDOL Document Security Administration Guide.

IDOL
The Intelligent Data Operating Layer (IDOL) Server, which integrates unstructured, semi-structured and structured information from multiple repositories through an understanding of the content. It delivers a real-time environment in which operations across applications and content are automated.

IDOL Proxy component
An IDOL Server component that accepts incoming actions and distributes them to the appropriate subcomponent. IDOL Proxy also performs some maintenance operations to make sure that the subcomponents are running, and to start and stop them when necessary.

IDX
The standard data format for indexing into IDOL Server. You can use a connector to import files into this format or you can manually create IDX files.

importing
After a document has been downloaded from the repository in which it is stored, it is imported to an IDX or XML file format. This process is called importing.

index
The IDOL Server data index contains document content and field information for analysis and retrieval

index action
An IDOL Server command to index data, or to maintain or manipulate the data index.

indexing
The process of storing data in IDOL Server. IDOL Server stores data in different field types (such as index, numeric, and ordinary fields). It is important to store data in appropriate field types to ensure optimized performance.

Intellectual Asset Protection System (IAS)
An integrated security solution to protect your data. At the front end, authentication checks that users are allowed to access the system that contains the result data. At the back end, entitlement checking and authentication combine to ensure that query results contain only documents that the user is allowed to see, from repositories that the user has permission to access. For more information, refer to the IDOL Document Security Administration Guide.

K

KeyView
The IDOL component that extracts data, including text, metadata, and subfiles from over 1,000 different file types. KeyView can also convert documents to HTML format for viewing in a Web browser.

L

LDAP
Lightweight Directory Access Protocol. Applications can use LDAP to retrieve information from a server. LDAP is used for

directory services (such as corporate email and telephone directories) and user authentication. See also: active directory, primary domain controller.

License Server
License Server enables you to license and run multiple IDOL solutions. You must have a License Server on a machine with a known, static IP address.

O

OmniGroupServer (OGS)
A server that manages access permissions for your users. It communicates with your repositories and IDOL Server to apply access permissions to documents.

P

PIN code
Personal Identification Number security feature used in addition to a user ID and password.

PQF
Prefix Query Format. An external format used to communicate with Z39.50 servers.

primary domain controller
A server computer in a Microsoft Windows domain that controls various computer resources. See also: active directory, LDAP.

privilege
Role-based capabilities that determine, for example, whether a user is allowed to access specific data.

profile
Information about a user that is based on the concepts in documents that the user reads. Every time a user opens a document, IDOL Server updates their profile. This process allows the administrator to alert users to new content that matches the interests in their profiles.

promotions
Targeted content that you want to display to users but is not included in the search results, such as advertisements.

Q

QMS rules
A document stored in the Promotion Agentstore that defines how QMS manages a query. Rules can return promotion documents, modify the original query, or modify the results of a query. See Also: Query Manipulation Server (QMS)

query
A string that you submit to IDOL Server, which analyzes the concept of the query and returns documents that are conceptually similar to it. You can submit queries to IDOL Server to perform several kinds of search, such as natural language, Boolean, bracketed Boolean, and keyword.

query cooker
A JavaScript application that manipulates queries and query results.

Query Manipulation Server (QMS)
An ACI server that manipulates queries and results according to user-defined rules.

R

reference
A string that identifies a document. This might be a title or a URL, and allows IDOL to identify documents for retrieval, indexing, and deduplication.

relevance
The similarity that a particular query result has to the initial query. IDOL Server assigns results a percentage relevance score according to how closely it matches the query criteria.

role
A set of privileges that the administrator allocates to an IDOL Server user.

S

security
Security includes anything that makes sure that only authorized users can access or perform actions on data. It includes making sure that only permitted users can view and retrieve documents, user authentication, and security communications.

suggest
A type of query that returns documents that contain similar concepts to a particular document, rather than matching a particular query string. See Also: query

synchronous action
An ACI action that is performed synchronously. The action is processed immediately and does not return a response until it is complete. See Also: asynchronous action.

V

View
An IDOL component that converts files in a repository to HTML formats for viewing in a Web browser.

virtual database
In the DAH, a virtual database controls the mapping between the DAH and specific databases in the child servers. See Also: combinator database, distributor database

W

Wildcard
A character that stands in for any character or group of characters in a query.

X

XML
Extensible Markup Language. XML is a language that defines the different attributes of document content in a format that can be read by humans and machines. In IDOL Server, you can index documents in XML format. IDOL Server also returns action responses in XML format.

Z

Z39.50
A client-server protocol for searching and retrieving information from remote computer databases, often used in library environments. DAH can query a Z39.50 server using the federated search.

Send documentation feedback

If you have comments about this document, you can contact the documentation team by email. If an email client is configured on this system, click the link above and an email window opens with the following information in the subject line:

Feedback on Administration Guide (Micro Focus Distributed Action Handler 12.4)

Add your feedback to the email and click **Send**.

If no email client is available, copy the information above to a new message in a web mail client, and send your feedback to swpdl.idoldocsfeedback@microfocus.com.

We appreciate your feedback!