

Distributed Index Handler

Software Version 12.7

Administration Guide



Document Release Date: October 2020
Software Release Date: October 2020

Legal notices

Copyright notice

© Copyright 2020 Micro Focus or one of its affiliates.

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 Index Handler	7
Chapter 1: Introduction	8
About DIH	8
OEM Certification	8
System Architecture	9
Mirror Mode	9
Non-Mirror Mode	10
Use Chained Distributed Index Handler Servers	11
Chapter 2: Install Distributed Index Handler	13
System Requirements	13
Supported Platforms	13
Install Distributed Index Handler on Windows	13
Install an IDOL Component as a Service on Windows	15
Install Distributed Index Handler on UNIX	17
Install an IDOL Component as a Service on Linux	19
systemd	19
System V	21
Chapter 3: Configure the Distributed Index Handler	22
Edit the Configuration File	22
Unified Configuration	22
Modify Configuration Parameter Values	23
Include an External Configuration File	24
Include the Whole External Configuration File	24
Include Sections of an External Configuration File	24
Include Parameters from an External Configuration File	25
Merge a Section from an External Configuration File	26
The DIH Configuration File	26
Display Online Help	27
Configuration File Sections	27
[ACIEncryption]	27
[AuthorizationRoles]	27
[DistributionIDOLServers] Section	28
[IndexNotify] Section	29
[IndexQueue] Section	29
[License] Section	29
[Logging] Section	29
[Paths] Section	30
[RoundRobinMode] Section	31
[Server] Section	31
[Service] Section	31

[SSLOptionN] Section	31
Example Configuration File	32
Set the Distribution Mode	33
Run the Distributed Index Handler in Mirror Mode	33
Run the Distributed Index Handler in Non-Mirror Mode	34
Manage Child Servers	35
Add an IDOL Server to the Distributed Index Handler	35
Remove an IDOL Server from the Distributed Index Handler	36
Add, Update, and Remove Child Servers Dynamically	37
Add a Child Server Dynamically	37
Remove a Child Server Dynamically	37
Update a Child Server Dynamically	38
Determine the State of Child Servers	38
Distribute Data Dynamically across Child Servers	38
Designate a Child Server as an Archive Server	39
Manage Client Connections	40
Manage the Indexing Process	40
Round Robin Indexing	41
Reference-Based Indexing	43
Field-Based Indexing	44
Field Value-Based Indexing	45
Date-Based Indexing	46
Send Minimal Documents	47
Use Consistent Hashing	48
Configure Consistent Hashing	48
Configure the DIH for Consistent Hashing Mode	49
Configure the Child Servers for Consistent Hashing Mode	49
Configure the DAH	50
Index Data in Consistent Hashing Mode	51
Add, Change, and Remove Child Servers in Consistent Hashing Mode	51
Add a New Child Server	51
Change the Weight of a Child Server	51
Remove a Child Server	52
Restore Child Servers in Consistent Hashing Mode	52
Change the Host and Port of a Child Server	52
Rebuild a Child Server	53
Manage the Index Queue	53
Archive Information	54
Archive Actions	54
Archive Failed Documents	55
Set Up SSL Connections	55
Configure Client Authorization	57
Customize Logging	58
Chapter 4: Operate the Distributed Index Handler	60
Start and Stop the Distributed Index Handler	60
Start the Distributed Index Handler	60

Start the DIH on Microsoft Windows	60
Start the DIH on UNIX	60
Stop the Distributed Index Handler	61
Index Data with the DIH	62
Use DREADD	62
Use DREADDDATA	62
Check Indexing Status	63
Example	63
IndexerGetStatus Status Codes	64
Administer IDOL Servers	66
Implement Configuration Changes	66
Compact the IDOL Servers	67
Back up the IDOL Servers	67
Initialize the IDOL Servers	68
Disable the IDOL Servers	68
Disable Index Actions	68
Manage Databases	69
Create a New Database in the IDOL Servers	69
Delete a Database and All the Documents that it Contains	69
Manage Documents	69
Delete Documents by Reference	70
Delete and Restore Documents by Document ID	70
Delete All Documents from a Database	71
Expire Documents	71
Change Document Field Values	71
Change Document Metadata	72
 Part II: Appendixes	 73
Appendix A: Troubleshooting	74
 Glossary	 75
 Send documentation feedback	 80

Part I: Use the Distributed Index Handler

This section describes how to set up and use the Micro Focus Distributed Index Handler (DIH).

- [Introduction](#)
- [Install Distributed Index Handler](#)
- [Configure the Distributed Index Handler](#)
- [Operate the Distributed Index Handler](#)

Chapter 1: Introduction

This chapter gives an overview of the Micro Focus Distributed Action Handler (DIH).

• About DIH	8
• System Architecture	9

About DIH

The Micro Focus Distributed Index Handler (DIH) is a distribution server that distributes index actions to IDOL Servers. Distribution enables you to scale your system in a linear manner, increasing the speed of index actions and saving processing time. Distributing the index between multiple copies of IDOL Server can also ensure uninterrupted service if an IDOL Server fails.

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

The DIH can index unstructured, semi-structured, or structured data into the connected IDOL Servers. You can aggregate this data from any type of repository by using IDOL *Connectors*. Connectors import the data into IDX file format (unless the data is in XML format, which IDOL Server also accepts) and passes it on to the DIH.

In addition to indexing data into the connected IDOL Servers, the DIH can also forward administrative index actions to its child IDOL Servers, for example to:

- activate IDOL Server configuration changes.
- delete content from IDOL Servers.
- create new IDOL Server databases.
- expire documents.
- compact IDOL Servers.
- change field values in IDOL Server documents.
- back up IDOL Servers.
- initialize IDOL Servers.
- validate the subindexes in an IDOL Server.

For details of all the index actions that the DIH accepts, refer to the *Distributed Index Handler Reference*.

OEM Certification

Distributed Index Handler works in OEM licensed environments.

System Architecture

The DIH receives index actions (data indexing requests or administrative actions) and distributes them to the connected IDOL servers.

You can run the DIH in either of two modes: *mirror mode* and *non-mirror mode*.

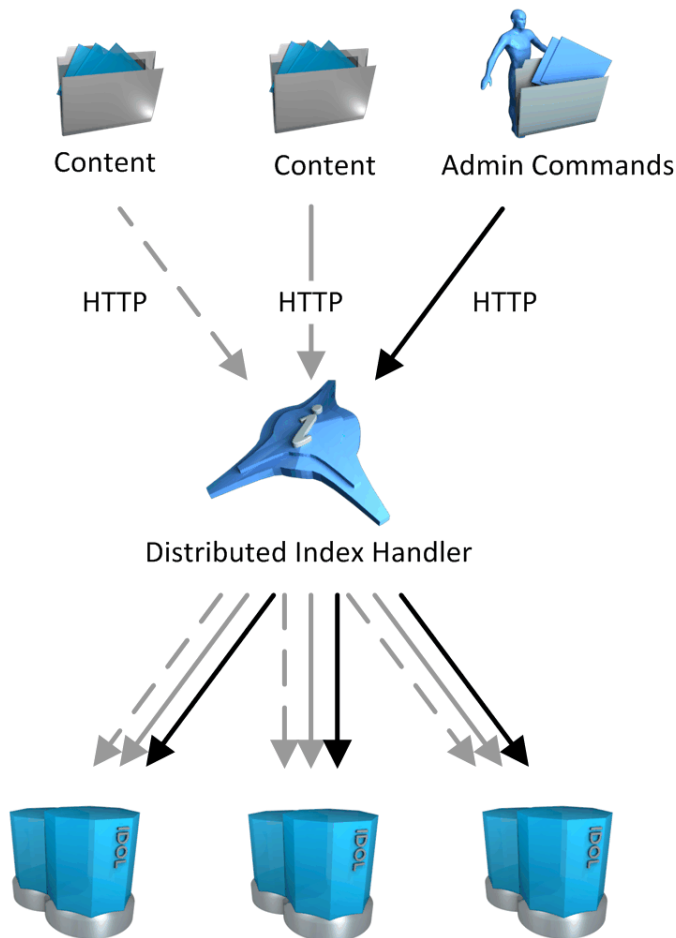
You determine the way that the DIH distributes index data by using the `MirrorMode` parameter in the DIH configuration file. See [Set the Distribution Mode, on page 33](#).

Mirror Mode

In mirror mode, the DIH indexes all data to all the connected IDOL servers. Each IDOL server is identical.

The following diagram shows how the DIH in mirror mode integrates into an IDOL Server installation.

DIH system architecture (mirror mode)



The DIH sends all the index data that it receives (represented by gray arrows in the diagram) to all the connected IDOL Servers. The IDOL Servers are exact copies of each other, and must all have the same configuration.

You can run the DIH in mirror mode to ensure uninterrupted service if one of the IDOL Servers fails. While one IDOL Server is inoperable, its identical copies continue to index data, and are still available to return data for queries.

The DIH periodically checks whether all connected IDOL Servers are operating. If an IDOL Server is unavailable, the DIH queues the data that this IDOL Server normally receives. When the IDOL Server is available again, the DIH indexes the queued data into it.

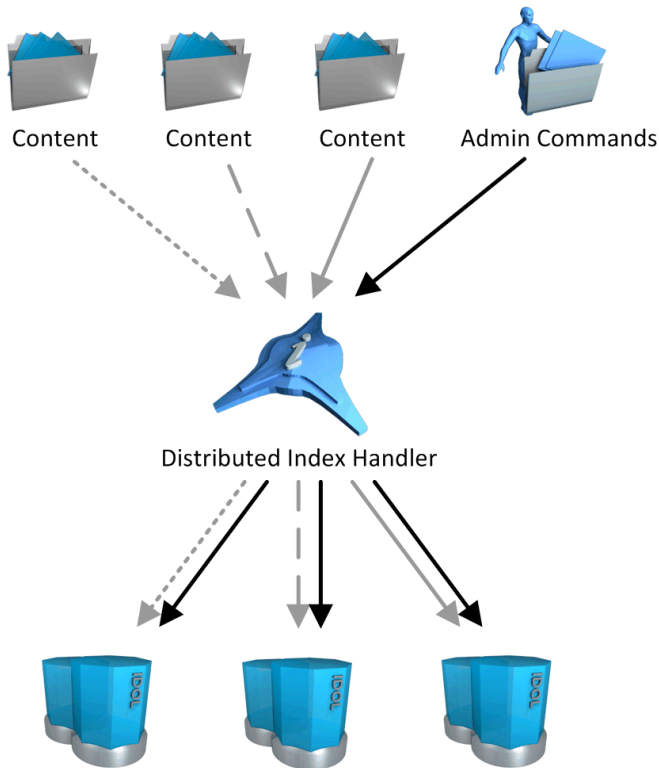
The DIH sends administrative index actions (represented by black arrows in the diagram) to all connected IDOL Servers.

Non-Mirror Mode

In non-mirror mode, the DIH divides the index data among the connected IDOL servers. Each IDOL server receives the same amount of data.

The diagram below shows how the DIH in non-mirror mode integrates into an IDOL server installation.

DIH system architecture (non-mirror mode)



The DIH distributes the index data that it receives (represented by gray arrows in the diagram above) evenly across the connected IDOL Servers. For example, if the DIH connects to four IDOL servers, it

indexes approximately one quarter of the data into each IDOL server. It does not split up sections of individual documents.

Run the DIH in non-mirror mode if the amount of data that you want to index is too large for a single IDOL Server. If the IDOL Servers that the DIH indexes into are on different machines, the indexing process requires less time.

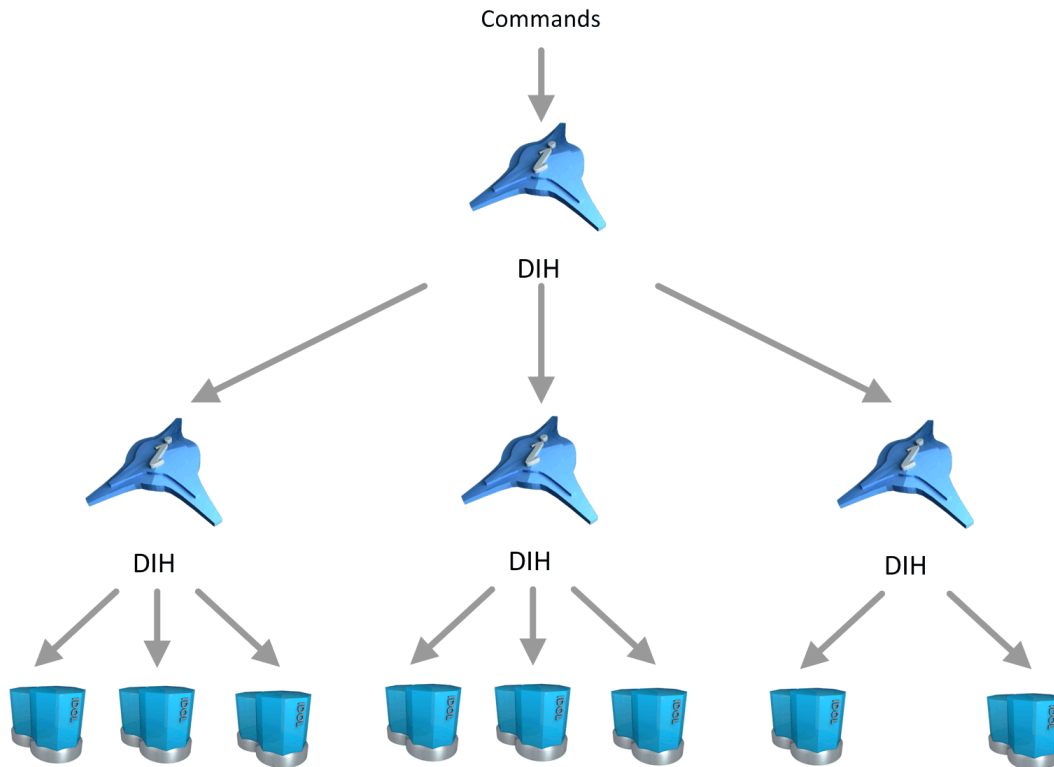
DIH periodically checks whether all the connected IDOL Servers are available. If an IDOL Server is unavailable, the DIH queues the data that this IDOL Server normally receives. When the IDOL Server is available again, the DIH indexes the queued data into it.

NOTE: Mirror mode and non-mirror mode refer only to how DIH distributes index data. DIH always distributes administrative commands to all connected IDOL Servers. This behavior is shown by the black arrows in the diagram.

Use Chained Distributed Index Handler Servers

You can set up multiple DIH instances in a chained configuration. In this configuration, a parent DIH distributes to child DIH servers, which in turn distribute to child IDOL servers.

DIH system architecture (chained configuration)



In this configuration, the parent DIH forwards all IDOL Server index actions to the child DIH servers in the same way as to child IDOL Servers.

NOTE: Some index actions have different effects when sent to a child DIH than when sent to an IDOL Server, because the DIH forwards the action to multiple IDOL Servers.

Chaining provides an extra level of redundancy both at the DIH level and at the IDOL Server level. It also distributes network traffic and system load over a larger number of computers. Use a chained DIH server configuration to create a pool of IDOL Servers that are both fault-tolerant for maximum availability, and distributed for the best performance.

Chapter 2: Install Distributed Index Handler

This section describes how to install the DIH by using the IDOL Server installer.

NOTE: After you install the DIH, you must configure at least two child IDOL servers to distribute to before you can use it.

- [System Requirements](#) 13
- [Install Distributed Index Handler on Windows](#) 13
- [Install an IDOL Component as a Service on Windows](#) 15
- [Install Distributed Index Handler on UNIX](#) 17
- [Install an IDOL Component as a Service on Linux](#) 19

System Requirements

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

Supported Platforms

IDOL runs on a variety of Windows and UNIX platforms. For details of supported platforms, refer to the *IDOL Server 12.7 Release Notes*.

Install Distributed Index Handler on Windows

Use the following procedure to install Distributed Index 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 Index Handler requires to run.

To install Distributed Index 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 Index 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 Index 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 DIH, you can specify the following ports:

ACI Port	The port that client machines use to send ACI actions to the DIH. Default: 9070
Index Port	The port that client machines use to send index actions to the DIH. Default: 9071
Service Port	The port that client machines use to send service requests to the DIH. Default: 9072

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 Index 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:

<code>-start {[auto] [manual] [disable]}</code>	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 .
<code>-username <i>UserName</i></code>	The user name that the service runs under. By default, it uses a local system account.
<code>-password <i>Password</i></code>	The password for the service user.
<code>-servicename <i>ServiceName</i></code>	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.
<code>-displayname <i>DisplayName</i></code>	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.
<code>-depend <i>Dependency1</i> [,<i>Dependency2</i> ...]</code>	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 Index Handler on UNIX

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

To install Distributed Index 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](#).
- 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**.
 - If you want to install a new License Server, provide information for the ports that the License Server uses.
 - 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.
 - 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.
 - Type the location of your IDOL license key file (`licensekey.dat`), which you obtained when you purchased Distributed Index Handler. Press `Enter`.
 - 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 DIH, you can specify the following ports:

ACI Port	The port that client machines use to send ACI actions to the DIH. Default: 9070
Index Port	The port that client machines use to send index actions to the DIH. Default: 9071
Service Port	The port that client machines use to send service requests to the DIH. Default: 9072

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.

- 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 install 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 Index Handler as a service on Linux.

IMPORTANT: These procedures assume that you install Distributed Index Handler by using the installer. The installer automatically populates some placeholder values in the init scripts. If you install components from a ZIP package, you must update these values manually before you attempt to install the service.

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

NOTE: When you install Distributed Index 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 init system.

- For Linux operating system versions that use `systemd` (including CentOS 7, and Ubuntu version 15.04 and later), see [systemd](#), below.
- For Linux operating system versions that use System V, see [System V](#), on page 21.

systemd

NOTE: If your setup has an externally mounted drive that Distributed Index 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 to copy the init scripts to the appropriate directory.

- Red Hat Enterprise Linux (and CentOS)

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

- Debian (including Ubuntu):

```
cp IDOLInstalLDir/scripts/init/systemd/componentname.service
/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.service
chown root /etc/systemd/system/componentname.service
chgrp root /etc/systemd/system/componentname.service
```

- Debian (including Ubuntu):

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

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
```

TIP: On systemd systems, services do not inherit file handle limits from the system limits or user settings. The default limits for services are configured separately in `*/systemd/system.conf` and `*/systemd/user.conf`.

In some cases this behavior might mean that a component fails to operate because it runs out of file handles. In this case, you can modify the `LimitNOFILE` parameter in the `componentname.service` file to increase the file handle limit before you install the service. Alternatively, you can create an `override.conf` file for the service.

System V

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 Index Handler

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

- [Edit the Configuration File](#) 22
- [The DIH Configuration File](#) 26
- [Set the Distribution Mode](#) 33
- [Manage Child Servers](#) 35
- [Manage Client Connections](#) 40
- [Manage the Indexing Process](#) 40
- [Use Consistent Hashing](#) 48
- [Manage the Index Queue](#) 53
- [Archive Information](#) 54
- [Set Up SSL Connections](#) 55
- [Configure Client Authorization](#) 57
- [Customize Logging](#) 58

Edit the Configuration File

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

```
installdir\dih\dih.cfg
```

where *installdir* is the directory in which you have installed IDOL components.

Unified Configuration

DIH is generally installed and operated as a stand-alone component, where you use a separate `dih.cfg` file to configure the DIH. However, in simple testing and training setups you can configure the DIH 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*.

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

There are two options for configuring DIH child servers, `[DIHEngines]` and `[DistributionIDOLServers]`.

In a stand-alone DIH with its own configuration file, you can use either option. 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 section, and some in the other).

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, in this case you might have DIH and DAH both sending actions directly to the Content component.
- If you use different child server configurations for the DIH and the DAH, use `[DIHEngines]` with `[DIHEngineN]` child server configuration sections. For example, in this case you might be using a tiered DIH set up, so that the DIH child servers are child DIH components, while the DAH child servers are child DAH components.

In this case, you configure the DAH by using the `[DAHEngines]` and `[DAHEngineN]` sections. For more information, refer to the *Distributed Action Handler Administration Guide*.

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

Modify Configuration Parameter Values

You modify Distributed Index 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 Index 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 Index 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 Index 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 Index Handler uses the values in the [SharedSSLOptions] section of the external configuration file as the values in the [SSLOptions1] section of the Distributed Index 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 Index 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 DIH Configuration File

By default, the DIH configuration file is named `dih.cfg`, and it is stored in your DIH installation directory:

```
InstallDir\dih\dih.cfg
```

where *InstallDir* is the directory in which you have installed IDOL components.

Display Online Help

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

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

To display help for Distributed Index Handler

1. Start Distributed Index 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 Index Handler is installed.

port is the ACI port by which you send actions to Distributed Index 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 DIH configuration file contains several sections that represent different areas that you can configure. For details on all available configuration parameters, refer to the *Distributed Index Handler Reference* (see [Display Online Help, above](#)).

[ACIEncryption]

You can use the [ACIEncryption] section to encrypt communications between ACI servers and any applications that use the ACI API. For example:

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

[AuthorizationRoles]

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
```

[DistributionIDOLServers] Section

The [DistributionIDOLServers] section contains settings that determine the location of the IDOL Servers that the DIH communicates with, and the ports by which this communication takes place.

There is a separate [IDOLServerN] subsection for each child server. The child servers can be IDOL Servers, Content components, or child DIH servers.

If you use date-based indexing, you must also configure [DateRangeN] subsections to specify the date ranges that each child server indexes.

For example:

```
[DistributionIDOLServers]
```

```
Number=2
```

```
[IDOLServer0]
```

```
Host=emerson
```

```
Port=9100
```

```
[IDOLServer1]
```

```
Host=thoreau
```

```
Port=9100
```

For non-mirrored systems, you can also create child server groups by specifying more than one child server in a comma-separated list. Each child in the group receives the same data from the parent DIH.

For example:

```
[DistributionIDOLServers]
```

```
Number=2
```

```
[IDOLServer0]
```

```
Host=DIH1.company.com,DIH2.company.com
```

```
Port=9100,9100
```

```
[IDOLServer1]
```

```
Host=DIH3.company.com
```

```
Port=9100
```

In this example, the DIH distributes data between the three DIH child servers. DIH1 and DIH2 receive the same data (they are mirrored copies of each other), and DIH3 receives a different set of data. The child servers can then distribute data between their respective child servers.

NOTE: If you configure DAH and DIH together in the IDOL Server configuration file, DAH can use the child server groups. However, you cannot also configure virtual databases in the DAH. Refer to the *Distributed Action Handler Administration Guide*.

[IndexNotify] Section

The [IndexNotify] section contains parameters that control and enable the automatic generation of index job information for a specified host. For example:

```
[IndexNotify]
Host=10.1.1.10
ACIPort=9100
BatchSize=1
BatchTimeout=10000
ConnectRetries=1
ConnectTimeout=5000
```

[IndexQueue] Section

The [IndexQueue] section contains parameters that control the index queue. For example:

```
[IndexQueue]
IndexQueueInitialSize=30000
IndexQueueMaxHistory=4000
IndexQueueMaxPendingItems=100
```

[License] Section

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

```
[License]
Holder=My Company
Key=01234567890
Operations=803|87sdhsdf9n94nmsf7oasda987w4yriasunfaasd
```

[Logging] Section

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

```
[Logging]
0=INDEX_LOG_STREAM
1=QUERY_LOG_STREAM
2=APP_LOG_STREAM
```

```
[INDEX_LOG_STREAM]
LogFile=index.log
LogDirectory=./logs/DIHIndexLogs
LogHistorySize=55
LogTime=True
LogEcho=True
LogMaxSizeKbs=1024
LogTypeCSVs=index
LogLevel=full
LogExpireAction=datestamp
```

```
[QUERY_LOG_STREAM]
LogFile=query.log
LogDirectory=./logs/DIHQueryLogs
LogHistorySize=50
LogTime=True
LogEcho=True
LogMaxSizeKbs=1024
LogTypeCSVs=query
LogLevel=full
LogExpireAction=consecutive
```

```
[APP_LOG_STREAM]
LogFile=application.log
LogDirectory=./logs/DIHAppLogs
LogHistorySize=50
LogTime=True
LogEcho=True
LogMaxSizeKbs=1024
LogTypeCSVs=application
LogLevel=full
LogExpireAction=datestamp
```

NOTE: The query logs truncate all queries to 4,000 characters.

[Paths] Section

The [Paths] section contains settings that determine where to store index data and other files. For example:

```
[Paths]
Main=./main
Incoming=./incoming
Archive=./archive
Failed=./failedTasks
```

[RoundRobinMode] Section

The [RoundRobinMode] section contains settings that control data indexing when you use round robin mode. Use settings in [DAHServerN] sections to configure the child servers for powerup and shutdown when round robin data indexing rollover occurs. For example:

```
[RoundRobinMode]
ServerImmediateStart=2
NextServerStartTime=00:00
NextServerStartDate=2006/10/20
PeriodInSec=86400
RoundRobinMode=True
```

```
[DAHServer0]
Host=host1
Port=9060
ShutDownEnginePeriods=0, -1
```

```
[DAHServer1]
Host=host2
Port=9060
StartUpEnginePeriods=-2
```

[Server] Section

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

```
[Server]
DIHPort=9071
Port=9070
MirrorMode=True
ArchiveMode=True
ArchiveFailedTasks=True
```

[Service] Section

The [Service] section contains settings that determine which machines have permission to use and control the DIH service. For example:

```
[Service]
ServicePort=9072
```

[SSLOptionN] Section

The [SSLOptionN] section contains settings that determine incoming or outgoing SSL connections between the DIH 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 [IDOLServerN] section, or the [DIHEngineN] section.

Example Configuration File

```
[Service]
ServicePort=9072
```

```
[Server]
Port=9070
DIHPort=9071
MirrorMode=True
```

```
[DistributionIDOLServers]
Number=2
```

```
[IDOLServer0]
Host=localhost
Port=5502
```

```
[IDOLServer1]
Host=localhost
Port=5602
```

```
[Logging]
0=INDEX_LOG_STREAM
1=QUERY_LOG_STREAM
2=APP_LOG_STREAM
```

```
[INDEX_LOG_STREAM]
LogFile=index.log
LogTypeCSVs=index
LogHistorySize=50
LogTime=True
LogEcho=True
LogMaxSizeKbs=1024
```



```
LogExpireAction=datestamp
```

```
[QUERY_LOG_STREAM]
```

```
LogFile=query.log
```

```
LogTypeCSVs=query
```

```
LogEcho=True
```

```
[APP_LOG_STREAM]
```

```
LogFile=application.log
```

```
LogTypeCSVs=application
```

```
LogEcho=True
```

```
[Paths]
```

```
IncomingPath=./incoming
```

```
LogPath=./logs
```

Set the Distribution Mode

You can configure the DIH to run in one of two alternate modes:

- Mirror mode
- Non-mirror mode

NOTE: The `MirrorMode` parameter is set to `True` in the default configuration file. However, if you do not set the `MirrorMode` configuration parameter, DIH runs in non-mirror mode.

Run the Distributed Index Handler in Mirror Mode

In mirror mode the DIH distributes all the index data it receives to all the connected IDOL Servers. The IDOL Servers are exact copies of each other and must all have the same configuration.

Run the DIH in mirror mode to ensure uninterrupted service if one of the IDOL servers fails. If one IDOL Server is inoperable, its identical copies continue to index data and return data for queries.

The DIH periodically checks if all the connected IDOL Servers are operating. If an IDOL server fails, the DIH queues the data that this IDOL Server normally receives. When the IDOL Server starts operating again, it indexes the queued data into it.

To run the DIH in mirror mode

1. Open the DIH configuration file in a text editor.
2. Find the `MirrorMode` setting in the `[Server]` section and set it to `True`.
3. Save and close the configuration file.
4. Restart the DIH for your changes to take effect.

NOTE: When you change the `MirrorMode` configuration option to enable or disable mirror mode,

you must also delete the `Main/` subdirectory in the DIH installation directory.

This additional action prevents accidentally switching between mirror and non-mirror mode, which can cause a loss of data. If you do not delete the `Main/` directory when you change the `MirrorMode` option, DIH does not start.

Run the Distributed Index Handler in Non-Mirror Mode

In non-mirror mode the DIH distributes the index data that it receives evenly across the connected IDOL Servers.

Run the DIH in non-mirror mode if the amount of data to index is too large for a single IDOL Server. This option can reduce the index process time, particularly if the IDOL Servers that the DIH indexes into are on different machines.

The DIH periodically checks if all the connected IDOL Servers are operating. The behavior when a child server fails depends on your configuration:

- In simple non-mirror distribution modes, the DIH treats any IDOL Server that fails as `UpdateOnly`. While the child server is down, DIH queues any updates to existing documents, but distributes new content between the remaining child servers. This behavior ensures that all content is indexed into the available servers, but that document updates and deduplication occurs in the offline child server when it comes back online.
- In advanced distribution modes (see [Manage the Indexing Process, on page 40](#)), the DIH queues the data that this IDOL Server normally receives, and when the IDOL Server starts operating again, it indexes the queued data into it.

To run the DIH in non-mirror mode

1. Open the DIH configuration file in a text editor.
2. In the `[Server]` section, find the `MirrorMode` setting and set it to `False`.
3. Save and close the configuration file.
4. Restart the DIH for your changes to take effect.

NOTE: When you change the `MirrorMode` configuration option to enable or disable mirror mode, you must also delete the `Main/` subdirectory in the DIH installation directory.

This additional action prevents accidentally switching between mirror and non-mirror mode, which can cause a loss of data. If you do not delete the `Main/` directory when you change the `MirrorMode` option, DIH does not start.

In non-mirror mode, you can choose various data distribution configurations.

- By default, the DIH distributes data as soon as it receives the data. To send documents in batches, set the `DistributeOnBatch` configuration parameter (in the `[Server]` section of the DIH configuration file) to `True`. This option can improve indexing speed, although the distribution of data among child servers can be more uneven.
- By default, the DIH sends all index data to all child servers. To send the full document only to the

child server that must index the content, set the `DistributeSendMinimal` configuration parameter. For more information, see [Send Minimal Documents, on page 47](#).

Manage Child Servers

This section describes how to add and remove child IDOL servers and distribute data.

Add an IDOL Server to the Distributed Index Handler

After you install the DIH, you can add more IDOL servers that the DIH can distribute actions to.

To add an IDOL server to the DIH

1. Open the DIH configuration file in a text editor.
2. Find the `[DistributionIDOLServers]` section.
3. Increase the value of the `Number` parameter by one, and add an `[IDOLServerN]` subsection for the new IDOL Server. The value of *N* must be one fewer than the value of the `Number` setting.

For example, if the following configuration is the original `[DistributionIDOLServers]` section:

```
[DistributionIDOLServers]
Number=2
```

```
[IDOLServer0]
Host=emerson
Port=9100
```

```
[IDOLServer1]
Host=thoreau
Port=9100
```

This section defines two IDOL servers. To add a third IDOL server, make the following changes:

- Increase `Number` to 3.
- Add an `[IDOLServer2]` for your new server.

For example:

```
[DistributionIDOLServers]
Number=3
```

```
[IDOLServer0]
Host=emerson
Port=9100
```

```
[IDOLServer1]
Host=thoreau
```

```
Port=9100
```

```
[IDOLServer2]
Host=jefferson
Port=9100
```

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

Remove an IDOL Server from the Distributed Index Handler

You can remove an IDOL server from the list of servers to which the DIH can distribute actions.

To remove an IDOL server from the DIH

1. Open the DIH configuration file in a text editor.
2. Find the [DistributionIDOLServers] section.
3. Decrease the value of the Number setting by one, and delete the subsection for the IDOL server to remove. Then, if necessary, renumber the individual subsections to restore a consecutive increasing sequence for *N*.

For example, if the following configuration is your current [DistributionIDOLServers] section:

```
[DistributionIDOLServers]
Number=3
```

```
[IDOLServer0]
Host=emerson
Port=9100
```

```
[IDOLServer1]
Host=thoreau
Port=9100
```

```
[IDOLServer2]
Host=jefferson
Port=9100
```

This section defines three IDOL servers. To remove the second IDOL server, make the following changes:

- Decrease Number to 2.
- Remove the [IDOLServer1] subsection.
- Rename the [IDOLServer2] subsection to [IDOLServer1] to preserve the numbering sequence.

For example:

```
[DistributionIDOLServers]
Number=2
```

```
[IDOLServer0]
Host=emerson
Port=9100
```

```
[IDOLServer1]
Host=jefferson
Port=9100
```

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

Add, Update, and Remove Child Servers Dynamically

You can add, edit, and remove child servers by using the `EngineManagement` action. This action updates the DIH configuration file, so that the changes are persistent.

For more details about the `EngineManagement` action, refer to the *DIH Reference*.

Add a Child Server Dynamically

You can add a new child server to the DIH by using the `EngineManagement` action.

NOTE: You can add a child server only in certain distribution modes.

To add a child server

- Send the `EngineManagement` action, with `EngineAction` set to **Add**. Set `Host` and `Port` to the host name and port of the child server.

You can optionally also set other parameters for the child server, such as `Disabled`, `Weight`, `UpdateOnly`, and `Polling`. For the full list of available parameters, see the *DIH Reference*.

TIP: By default, when you add a child server in this way, the DIH pings the specified host and port. The `Add` action fails if the child server is not running. Set `Disabled` to `True` to skip this step and add the child server in the offline state.

Remove a Child Server Dynamically

In *consistent hashing* mode, you can remove a child server or group. For more information about this mode, see [Use Consistent Hashing, on page 48](#).

To remove a child server in this mode, you must first set `Weight` to `0` (zero), and then use the `DREREDISTRIBUTE` index action to redistribute the virtual nodes. For more information, see [Remove a Child Server, on page 52](#).

Update a Child Server Dynamically

You can use the `EngineManagement` action to dynamically change the settings for a child server.

To update a child server

- Send the `EngineManagement` action, with `EngineAction` set to **Edit**. Set the `ID` parameter to the ID of the child server that you want to modify.

Set any of the optional parameters that you want to change for the child server, such as `Disabled`, `Weight`, `UpdateOnly`, and `Polling`.

You can also update the host and port of the server, for example replace a server that has become permanently inaccessible. In consistent hashing mode, you can use stored replicas to rebuild the lost server. For more information, see [Restore Child Servers in Consistent Hashing Mode, on page 52](#).

Determine the State of Child Servers

The DIH regularly pings all its child IDOL servers to determine their state; running (up) or not running (down). In mirror mode, if a server is down, the DIH queues the data for that server, and indexes the data into the server when it is available again. In non-mirror mode, the DIH redistributes the data among the available servers. In this case, it also queues any updates to documents that have already been indexed, and sends the updates when the server is available again.

You can configure how often the DIH checks the state of its child servers. Set the `PingInterval` parameter in the `[Server]` section of the DIH configuration file to the time interval between checks. The default value is 20 seconds.

Distribute Data Dynamically across Child Servers

In non-mirror mode, you can configure the DIH to distribute data dynamically across a bank of child servers, based on user-defined limits to the number of documents. This option also allows you to determine when all child servers are full and you require new machines.

NOTE: These options are available in simple non-mirror mode, and in `DistributeOnBatch` and `DistributeSendMinimal` modes. They are not available for `DistributeByReference`, `DistributeByDate`, or `DistributeByFields` modes.

To define a maximum for document indexing into child servers, set the `MaxDocumentCount` parameter for each child IDOL Server, in the `[Server]` section of the IDOL server configuration file. You can also use the `MaxDocumentCountUpper` and `MaxDocumentCountLower` parameters for more control over the document limits. Refer to the *IDOL Server Reference* for details on these configuration options.

When an IDOL Server reaches the maximum number of documents, it returns `<FULL>` in the `GetStatus` action response. IDOL servers also return a `<FULL_RATIO>` tag, to indicate how close the index is to being full.

Use the following configuration parameters to specify how the DIH deals with full child servers.

<code>CollectChildFullness</code>	Whether to send a <code>GetStatus</code> action to child servers to determine if they are full. The DIH then also returns its own fullness information in the response to a <code>GetStatus</code> action. If the DIH has no information from its child servers (for example, when you set <code>CollectChildFullness</code> to False), the DIH reports that it is not full.
<code>GetChildStatusMode</code>	How often to send <code>GetStatus</code> actions to the child servers. If you set <code>GetChildStatusMode</code> to Command , the DIH sends a <code>GetStatus</code> action with every index action. If you set <code>GetChildStatusMode</code> to ASync , the DIH sends a <code>GetStatus</code> action after every <code>PingInterval</code> .
<code>PingInterval</code>	How often to send <code>GetStatus</code> actions, if you set <code>GetChildStatusMode</code> to ASync .
<code>RespectChildFullness</code>	Whether to index into full child servers. If you set this parameter to True , the DIH routes actions to child servers that are not full. This implicitly sets <code>CollectChildFullness</code> to True .
<code>RespectChildFullness MaxIndexingGroups</code>	The maximum number of child server groups to use for indexing. When you set this parameter, DIH indexes into only the first <i>N</i> non-full child server groups in your configuration. When a child server group becomes full, indexing rolls over to the next non-full child server group.

If all child servers return <FULL>, you must either add more machines to your system, or create space on the existing machines.

For more information about these configuration parameters, refer to the *DIH Reference*.

Related Topics

- [Use Chained Distributed Index Handler Servers, on page 11](#)

Designate a Child Server as an Archive Server

In non-mirror mode, you can designate a child server as an archive server. An archive server receives document updates but does not receive new documents.

Set the `UpdateOnly` parameter to **True** in the `[DistributionIDOLServers]` section of the DIH configuration file to convert a child server into an archive server.

NOTE: You cannot use the `UpdateOnly` option with the `DistributeByFields`, `DistributeByReference`, or `RoundRobinMode` distribution modes. For more information about the compatible options for distribution modes, refer to *IDOL Expert*.

NOTE: If all children are marked as `UpdateOnly`, the DIH cannot process any `DREADD` or `DREADDATA` index actions, because there are no servers it can send new documents to. In this case, the DIH pauses index queue processing until you reconfigure one or more child servers to set `UpdateOnly` to **False**.

You can also change this setting dynamically (without restarting the DIH) by using the `EngineManagement` action. For example:

```
http://
DIHhost:ACIPort/action=EngineManagement&EngineAction=Edit&ID=1&UpdateOnly=True
```

Manage Client Connections

You can control several aspects relating to clients connecting to the DIH. Use the following configuration parameters, which you set in the `[Server]` section of the DIH configuration file.

- | | |
|---------------------------|---|
| <code>RecvTimeout</code> | How long the DIH attempts to read an client index action before it times out. The default value is 60 seconds. A small value can improve the indexing speed, but might drop more actions. |
| <code>RecvDuration</code> | How long a communication between the DIH and a client can last (in either direction). The default value is 600 seconds. A large value allows you to index more data with each action. |

You can also control the users that can perform index operations by using the `[AuthorizationRoles]` section of the configuration file. In particular, you might want to restrict the users that can use the Index standard role. For more information, refer to the *DIH Reference*.

Manage the Indexing Process

To tune indexing performance, you can adjust certain limits on the DIH and its child servers.

- Specify how many times the DIH attempts to send an index action to a child IDOL server before it assumes that the connection has failed.

Set the `MaximumRetries` configuration parameter (in the `[Server]` section of the DIH configuration file) to the maximum number of attempts. The default value is 10.

- Limit the number of indexing threads that DIH can employ.

Set the value in the `Threads` configuration parameter (in the `[Server]` section of the DIH configuration file). The default value is 10 threads. Micro Focus recommends that you use $(1 \times \text{Num. CPUs}) + 1$ spare thread.

- Limit the size of an indexing request string, which limits the amount of data that you can index in a single request.

Set the `MaxInputString` configuration parameter (in the `[Server]` section of the DIH configuration file) to the value that you want. The default value is 64000. A value of -1 means that there is no limit.

- Specify that DIH must have a certain amount of available disk space for indexing to proceed.

Set the `MinFreeSpaceMB` configuration parameter (in the `[Server]` section of the DIH configuration file) to the minimum amount of disk space that DIH must have. By default, DIH must have 20 MB of disk space.

- Specify that a certain number of child IDOL servers must be running for indexing to proceed.

Set the `MinChildrenAlive` configuration parameter (in the [Server] section of the DIH configuration file) to the value that you want. By default, there is no minimum requirement.

- Stop the DIH from turning DREADD actions into DREADDATA actions.

Set the `PreserveDREADD` configuration parameter (in the [Server] section of the DIH configuration file) to `True`. This option reduces network load by sending only the path to the IDX file to child servers, rather than streaming all the contents of the IDX file. You can use this parameter only in simple mirror or non-mirror mode. You must ensure that all child servers can access the file system containing the IDX file with the same file path.

- Use weighted indexing to alter the ratio in which documents distribute to different servers in standard distributed mode.

Set the weight for different servers by using the `EngineManagement` action with the `Weight` parameter. For example:

```
http://DIHhost:ACIPort/action=EngineManagement&EngineAction=Edit&ID=1&Weight=2
```

A server with weight `2` receives twice as many documents as a server with weight `1` and so on. You can set a weight of `0` to add no documents to a server.

- Use round-robin indexing to maximize indexing performance without compromising the IDOL Server pool availability. See [Round Robin Indexing, below](#).
- Use reference-based indexing to distribute the indexing load evenly between IDOL Servers. See [Reference-Based Indexing, on page 43](#).
- Use field-based indexing to distribute the indexing load evenly between IDOL Servers. See [Field-Based Indexing, on page 44](#).
- Use date-based indexing to distribute the indexing load between IDOL servers. See [Date-Based Indexing, on page 46](#).

Round Robin Indexing

You can use round robin indexing with the DIH to maximize indexing performance. This option rotates indexing over several child servers, so that only one DIH indexes at a time.

A round robin DIH forms part of a larger architecture that achieves extremely high performance and low search latency. IDOL Server provides the fastest queries when it is not indexing, and indexes fastest when it is not being queried. When you configure indexing for round robin mode, DIH suspends query handling for a specific child server. This server then has optimal indexing, and only one child server receives most incoming documents.

If you have the DAH installed, you can configure the DAH to divert search queries to other IDOL servers to allow the active server to devote all its resources to the indexing task.

Use the [DAHServer*N*] section options to configure child servers for powerup and shutdown when round robin indexing rollover occurs. This option optimizes both query handling and data indexing across a group of child servers.

For example, DAH disables queries to the first server, which optimizes indexing speed for that server. Query handling is optimal for the other servers, which are not indexing data. The first server indexes data during this period, before DAH queries it again.

NOTE: You can use round robin mode without installing the DAH to divert search queries from the active server. However, when the active server must perform search and indexing tasks simultaneously, it compromises performance for both.

To enable round robin mode, set `RoundRobinMode` to `True` in the `[Server]` section of the DIH configuration file. You must complete the following sections in the DIH configuration file (DAH portions apply only if you have installed it):

```
[Server]
Port=9070
DIHPort=9071
DateFormatCSVs=SHORTMONTH#SD+#SYYYY,DD/MM/YYYY,YYYY/MM/DD,YYYY-MM-DD
MirrorMode=False
RoundRobinMode=True
```

```
[DistributionIDOLServers]
Number=3
```

```
[IDOLServer0]
Host=localhost
Port=9100
```

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

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

```
[RoundRobinMode]
ServerImmediateStart=2
NextServerStartTime=00:00
NextServerStartDate=2006/10/20
PeriodInDay=1
```

```
[DAHServer0]
Host=dahhost0
Port=9060
ShutDownEnginePeriods=0,-1
```

```
[DAHServer1]
Host=dahhost1
Port=9160
StartUpEnginePeriods=-2
```

In this example, the DIH immediately starts to send index data to IDOL Server 2. Indexing switches to the next server, IDOL Server 0, on 20 October 2006 at 00:00. After this period, DIH dedicates one day for each child server before it rolls over to the next. To optimize data indexing when the rollover occurs, and to suspend query handling, it shuts down the DAH child servers that indexed data yesterday and today. At the same time, it powers up the DAH child server that indexed data the day before yesterday.

NOTE: You can use `RoundRobinMode` only when `MirrorMode` is set to `False`. DIH will not start if `RoundRobinMode` and `MirrorMode` are both set to `True`.

Reference-Based Indexing

You can use reference-based indexing to distribute the indexing load evenly between IDOL Servers and achieve efficient data distribution. Data indexing depends on the reference of the documents. In simple non-mirror mode, DIH sends all actions to all servers, and instructs the child servers to determine which documents to index. With reference-based indexing, DIH performs these calculations, which reduces network traffic and load on the child servers.

When enabled, reference-based indexing applies to the `DREADD`, `DREADDDATA`, and `DREREPLACE` index actions. You must configure the `DRREFERENCE` field by using standard field processing settings.

When you use reference-based indexing, you cannot alter the number of child servers.

In a chained DIH setup, the DIHs might distribute documents unevenly if more than one level of the chain uses reference-based indexing. To prevent uneven distribution, the number of child servers at each level must be coprime (that is, they have no common numerical factors).

For example, if you have a parent DIH with two DIH child servers, each of which has four IDOL Server children, documents are not distributed evenly in distribute-by-reference mode. The parent server splits the data into two using a checksum hash of the document reference. The first child server uses the same algorithm to distribute data to its four child servers. Because it has only the first half of the data, only two child servers receive data.

However, if you have a parent DIH with two child servers, each of which has three IDOL Server children, data is distributed evenly in distribute-by-reference mode. The parent server splits the data into two. The first child server then splits the data into three, so that all child servers receive data.

NOTE: Reference-based indexing might prevent deduplication of documents with different references. You can use reference-based indexing only with a `KillDuplicates=REFERENCE` or `KillDuplicates=NONE` setting in the `[Server]` section of the IDOL Server configuration file.

To enable reference-based indexing, set `DistributeByReference` to `True` in the `[Server]` section of the DIH configuration file.

For example:

```
[Server]
Port=9070
DIHPort=9071
MirrorMode=False
DistributeByReference=True
```

NOTE: You can use `DistributeByReference` only when `MirrorMode` is set to `False`. DIH will not

start if `DistributeByReference` and `MirrorMode` are both set to `True`.

You must also configure standard field processing options to specify the reference field to use to distribute documents. For more information, refer to the *IDOL Server Administration Guide*.

Field-Based Indexing

You can use field-based indexing to distribute the indexing load between IDOL servers. This mode is similar to reference-based indexing, except that you configure the document fields that determine which child server to send the document to. Data indexing uses the value of the specified field in the documents being indexed or in which you replace fields.

When you enable field-based indexing, it applies to the `DREADD`, `DREADDATA`, and `DREREPPLACE` index actions. DIH sends `DREREPPLACE` index actions to all child servers, because the `DREREPPLACE` action does not contain the information required to determine which child server contains the original document.

When you use field-based indexing, you cannot alter the number of child servers.

NOTE: Field-based indexing might prevent deduplication of documents with different field values. You can use field-based indexing only with a `KillDuplicates=NONE` setting in the `[Server]` section of the IDOL Server configuration file.

To enable field-based indexing

1. Open the DIH configuration file in a text editor.
2. In the `[Server]` section, set the `DistributeByFields` parameter to `True`.
3. In the `[Server]` section, Set `DistributeByFieldsCSVs` to a comma-separated list of fields that DIH uses to distribute index data between child servers. For example:

```
DistributeByFieldsCSVs=*/DeDupeHash,*/SecondDistributeField
```

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

For example:

```
[Server]
Port=9070
DIHPort=9071
MirrorMode=False
DistributeByFields=True
DistributeByFieldsCSVs=*/DeDupeHash,*/SecondDistributeField
```

NOTE: You can use `DistributeByFields` only when `MirrorMode` is set to `False`. DIH will not start if `DistributeByFields` and `MirrorMode` are both set to `True`.

You can also set the `BalanceDistributeByFields` configuration parameter to balance the distribution of documents that do not contain the specified distribution fields. In this option, DIH sends documents to a random child server if they lack the specified distribution fields.

Field Value-Based Indexing

By default, DIH internally determines how to distribute documents between child servers. This process ensures that DIH always sends duplicate documents to the same child server.

Instead, you can configure DIH to distribute documents to specific child servers when the field contains a specific value.

To enable field value-based indexing

1. Open the DIH configuration file in a text editor.
2. In the [Server] section, set the `DistributeByFields` parameter to **True**.
3. In the [Server] section, set the `DistributeByFieldsCSVs` parameter to a comma-separated list of fields that DIH uses to distribute index data between child servers.
4. In the [IDOLServerN] or [DIHEngineN] section for each group of child servers, set `DistributeByFieldsValues` to a comma-separated list of field values. If a document contains a field listed in the `DistributeByFieldsCSVs` parameter with this value, indexes it to this child server. For example:

```
[DIHEngine2]
DistributeByFieldsValues=backup,France
```

NOTE: You can configure each field value in the list for only one child server. If a field value occurs in the list for multiple child servers, indexes matching documents into the child server with the lowest ID.

5. In the [Server] section, set the `UnknownFieldValueAction` parameter to the action that DIH takes if the fields listed in the `DistributeByFieldsCSVs` parameter contain unknown values. The following actions are available:

Distribute	DIH uses a hash of the field values to distribute the document, as with conventional field-based indexing.
Ignore	DIH ignores the document and logs a warning.
Default	DIH sends the document to the server specified by the <code>UnknownFieldValueDefaultEngine</code> configuration parameter.

For example:

```
UnknownFieldValueAction=Distribute
```

6. In the [Server] section, set `UnknownFieldValueDefaultEngine` to the number of the child server that acts as the default server. Set this parameter only if you have set `UnknownFieldValueAction` to **Default**.
7. In the [Server] section, set `DistributeOnMultipleFieldValues` to **True** if you want to index documents into each server group that matches the particular field values. Set this parameter to **False** if you want the document to index only into the server with the lowest number.

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

For example:

```
[Server]
DistributeByFields=True
DistributeByFieldsCSVs=*/database,*/country
UnknownFieldValueAction=Default
UnknownFieldValueDefaultEngine=0
DistributeOnMultipleFieldValues=True
```

```
[DIHEngines]
Number=3
```

```
[DIHEngine0]
DistributeByFieldsValues=main
```

```
[DIHEngine1]
DistributeByFieldsValues=uk
```

```
[DIHEngine2]
DistributeByFieldsValues=backup,france
```

Date-Based Indexing

You can use date-based indexing to distribute the indexing load between IDOL servers. Date-based indexing uses the date of the documents being indexed or in which you replace fields.

When you enable date-based indexing, DIH indexes each document in a DREADD and DREADDATA action by its #DREDATE field, or another DateType field configured in the [FieldProcessing] section. It indexes each replace in a DREREPPLACE action based on its #DREDATE line, if it exists. Otherwise it sends the action to all child servers. It sends all other actions to all child servers.

When you use date-based indexing, you cannot alter the number of child servers.

To enable date-based indexing, set `DistributeByDate` to **True** in the [Server] section of the DIH configuration file.

Configure the date ranges for child servers by using a [DateRangeN] subsection in the [DistributionIDOLServers] section of the DIH configuration file.

For both DIH stand-alone and unified configuration, you must configure `DateFormatCSVs` in the [Server] section for date-based indexing to work. For example:

```
[Server]
Port=9070
DIHPort=9071
MirrorMode=False
DistributeByDate=True
DateFormatCSVs=DD/MM/YYYY,YYYY/MM/DD,YYYY-MM-DD
```

```
[DistributionIDOLServers]
```

```
Number=2
```

```
[IDOLServer0]
Host=localhost
Port=9100
```

```
[IDOLServer1]
Host=localhost
Port=9500
```

```
[DateRange0]
FromDate=1980/01/01
UpToDate=1990/01/01
Engines=0
```

```
[DateRange1]
FromRelative=-3
UpToRelative=5
Engines=1
```

In this example, server 0 indexes documents dated from 1 January 1980 to 31 December 1989. If it is Tuesday (relative 0), server 1 gets documents dated from the previous Saturday (relative -3) and from the following Saturday (relative 4). The upper limit is exclusive.

NOTE: You can use `DistributeByDate` only when `MirrorMode` is set to `False`. DIH will not start if `DistributeByDate` and `MirrorMode` are both set to `True`.

In `DistributeByDate` mode, you can also use the `UnknownFieldValueAction` configuration parameter to determine how to treat documents where the date field is missing, contains a value that DIH cannot parse as a date, or contains a date value that does not match the configured date ranges. For more information, refer to the *DIH Reference*.

Send Minimal Documents

In non-mirror mode, you can configure DIH to send each child server only the information it needs. In `DistributeSendMinimal` mode, the DIH determines which child server must index each document, and sends the complete document only to that child server.

DIH sends a minimal representation of the document to all other child servers. The content of this representation is defined by the `KillDuplicates` mode of the original index action.

This mode allows all child servers to perform correct deduplication, and reduces network traffic to child servers.

To enable minimal sending mode, set the `DistributeSendMinimal` configuration parameter to `True` in the `[Server]` section. For example:

```
[Server]
MirrorMode=False
DistributeSendMinimal=True
```

NOTE: When you want to deduplicate on a field other than `DRREFERENCE`, you must configure a field process in the DIH configuration file, with the fields that you want to use to deduplicate. DIH then includes these fields in the representation that it sends to its child servers. By default, it sends only the `DRREFERENCE`.

For more information about setting up a field process, refer to the *IDOL Server Administration Guide*.

Use Consistent Hashing

In the `DistributeByReference` and simple `DistributeByFields` modes, you can use consistent hashing mode to create a more flexible DIH architecture. In consistent hashing mode, you can add, remove, or change the weight of child servers in your DIH without having to reindex all your content.

NOTE: To use consistent hashing, you must have Content component version 10.1.1 or later in your child servers.

In consistent hashing mode, DIH creates a large, fixed number of *virtual nodes*. You configure the number of virtual nodes to be much larger than your intended number of child servers. DIH assigns the virtual nodes to one or more of your configured child servers.

When you index content, DIH distributes the data between the virtual nodes, according to your distribution mode (`DistributeByReference` or `DistributeByFields`). DIH creates a `DREVNODE` field in each document, which stores details of the virtual node, and then indexes the document to the assigned child servers for the virtual node.

NOTE: You cannot use consistent hashing mode in a distribution architecture with tiered DIH servers.

You also cannot use consistent hashing mode with `DistributeByFieldValues`. `DistributeByFieldValues` determines the child server to send a document to according to a specific value in a field, so it sends data to specific child servers, rather than to virtual nodes.

If you change the number or weight of child servers, DIH can redistribute the virtual nodes between the new set of servers. It modifies its internal mapping to assign the virtual nodes evenly among the new set of servers. It then sends index actions to export data from some servers and index it into others to redistribute the data.

Related Topics

- [Reference-Based Indexing, on page 43](#)
- [Field-Based Indexing, on page 44](#)

Configure Consistent Hashing

The following procedures describe how to configure consistent hashing mode in your DIH and child IDOL servers.

Configure the DIH for Consistent Hashing Mode

The following procedure describes the configuration changes to the DIH that enable consistent hashing mode. To use consistent hashing, you must configure DIH in `DistributeByReference` or `DistributeByField` mode.

NOTE: After you configure the number of `VirtualNodes` and `Replicas` for your system, you cannot change these values. If you want to change these numbers, you must use a clean DIH installation and reindex all your content.

For more information about these parameters, refer to the *DIH Reference*.

To configure consistent hashing in the DIH

1. Open the DIH configuration file in a text editor.
2. In the `[Server]` section, set the `UseConsistentHashing` parameter to `True`. In a unified IDOL configuration, set this parameter in the `[DistributionSettings]` configuration section. For example:

```
[Server]
UseConsistentHashing=True
```

3. Create a `[ConsistentHashing]` section.
4. Set the `VirtualNodes` parameter to the number of virtual nodes that you want to create. DIH rounds this value up to the nearest power of two (for example, if you set `VirtualNodes` to **4000**, it creates 4,096 virtual nodes).

Micro Focus recommends that you set the value of `VirtualNodes` to be an order of magnitude higher than the number of child servers that you expect to use.

The minimum value is the higher number of 33 (that is, 64 nodes), or the number of child servers in your system. If you create fewer virtual nodes than there are child servers, DIH does not start. In general, Micro Focus recommends that you do not reduce the default value of virtual nodes (4,096).

5. (Optional) Set the `Replicas` parameter to the number of identical copies of each document that you want to index. When you configure replicas, DIH copies the documents in a particular virtual node to two or more child servers. For more information, refer to the *Distributed Index Handler Reference*.
6. Save and close the DIH configuration file.

Configure the Child Servers for Consistent Hashing Mode

The following procedure describes the configuration changes to the child IDOL servers that are required for consistent hashing mode.

To configure consistent hashing for the child servers

1. Open the configuration file for one of your child IDOL or Content servers.
2. In the `[FieldProcessing]` section, add a new field process at the bottom of the list, and

increase the value of the Number parameter by one. For example:

```
[FieldProcessing]
Number=4
0=SetIndexFields
1=SetReferenceFields
2=SetSectionBreakFields
3=SetVNodeReferenceField
```

NOTE: You must create a new field process, rather than adding the field to an existing reference process.

3. Create a new configuration section for the field process. Set `PropertyFieldCSVs` to `*/DREVNODE`, and set `Property` to the name of a new property. For example:

```
[SetVNodeReferenceField]
Property=VNodeReferenceField
PropertyFieldCSVs=*/DREVNODE
```

4. Create a configuration section for the corresponding property. Set the `ReferenceType` and `TrimSpaces` properties to `True`. You might also want to use the `HiddenType` property for this field. For example:

```
[VNodeReferenceField]
ReferenceType=True
TrimSpaces=True
HiddenType=True
```

5. Save and close the configuration file. Restart the child server for your changes to take effect.
6. Repeat Step 1 to Step 5 for each of your child servers.

Configure the DAH

In consistent hashing mode when you have not configured replicas, you do not need to make any changes to the DAH configuration, except when you add or remove child servers.

When you configure replicas, the DIH distributes identical copies of the virtual nodes between its child servers, ensuring that it does not assign the copies to the same child server. Unlike full server mirroring, the DAH does not know where each of the copies are. In this case, you must:

- configure your DAH in simple combinator mode
- combine query results by reference

For more details about these settings, refer to the *DAH Administration Guide* and *IDOL Server Reference*.

In addition, when you send a `GetQueryTagValues` action to the DAH in a system that uses replicas, the document counts for tags include a value for each copy of the document that exists (that is, the action counts each replica as a separate document).

Index Data in Consistent Hashing Mode

In consistent hashing mode, Micro Focus recommends that you do not use the `Priority` index action parameter for any action that affects your data (for example, `DREADDDATA`, `DREREPPLACE`, `DREDELETEREF`). You can use index action priorities for purely administrative index actions as usual (for example `DRESYNC`, `DRECOMPACT`, `DREBACKUP`).

Add, Change, and Remove Child Servers in Consistent Hashing Mode

In consistent hashing mode, you can add or remove child servers, and change the weight of a child server, without reindexing all your content. This process uses the `EngineManagement` action and the `DREREDISTRIBUTE` index action to change your child server arrangement and redistribute the data respectively.

When the DIH receives a `DREREDISTRIBUTE` index action, it checks whether redistribution is required. If it is, the DIH remaps the virtual nodes, and automatically sends export and add index actions to its child servers to redistribute the associated content.

NOTE: DIH can process only one `DREREDISTRIBUTE` index action at the same time. If it starts to process a second `DREREDISTRIBUTE` index action in the queue before all child servers have finished redistributing the content, the second `DREREDISTRIBUTE` index action returns the `Unavailable` error code and does not run.

Add a New Child Server

Use the following procedure to add a new child server in consistent hashing mode.

To add a child server

1. Send the `EngineManagement` action to the DIH ACI port. Set the `EngineAction` parameter to **Add**, and set `Host` and `Port` to the host and port of the new child server. For example:

```
action=EngineManagement&EngineAction=Add&Host=Child3&Port=9100
```

You can also add a new child server to the last listed existing mirror group by setting the `Group` parameter in the `EngineManagement` action to the ID of the group.

2. Send the `DREREDISTRIBUTE` index action to the DIH index port. This index action checks whether redistribution is required, and runs the export and indexing process.

```
http://12.3.4.56:9071/DREREDISTRIBUTE
```

Change the Weight of a Child Server

Use the following procedure to change the weight of a child server in consistent hashing mode.

To change the weight of a child server

1. Send the EngineManagement action to the DIH ACI port. Set the EngineAction parameter to **Edit**, set ID to the ID number of the child server in the DIH configuration file, and set weight to the new weight for this child server. For example:

```
action=EngineManagement&EngineAction=Edit&ID=3&Weight=5
```

2. Send the DREREDISTRIBUTE index action to the DIH index port to redistribute the index data between child servers according to the new weighting.

```
http://12.3.4.56:9071/DREREDISTRIBUTE
```

Remove a Child Server

Use the following procedure to remove a child server group in consistent hashing mode. You cannot remove a group that has virtual nodes assigned, and you can remove only whole groups of servers.

To remove a child server

1. Use the EngineManagement action to set the weight for the child server to zero. Set the EngineAction parameter to **Edit**, set ID to the ID number of the child server in the DIH configuration file, and set weight to 0 (zero).

```
action=EngineManagement&EngineAction=Edit&ID=3&Weight=0
```

2. Send the DREREDISTRIBUTE index action to the DIH index port to redistribute the index data between child servers according to the new weighting.

```
http://12.3.4.56:9071/DREREDISTRIBUTE
```

This index action removes all the virtual nodes from the child server. DIH continues to send any mirrored index actions (such as DRESYNC and DRECOMPACT) to this child server until you remove it completely.

3. After the redistribution index action is complete, send the EngineManagement action again. Set the EngineAction parameter to **Remove**, and set Group to the ID of the group.

Restore Child Servers in Consistent Hashing Mode

In consistent hashing mode when you have configured replicas, you can restore a child server that becomes permanently unavailable (for example because of a hardware failure). DIH can rebuild the index of the lost child server from the replicas stored in other child servers in your system.

For information about how to configure replicas, see [Configure Consistent Hashing, on page 48](#).

To restore a child server that was lost, you must first update the DIH with information about the replacement, by using the EngineManagement action. Then you can use the DREREDISTRIBUTE index action to rebuild the child server.

Change the Host and Port of a Child Server

When a child server becomes permanently inaccessible, you can start a new child server in its place. You can modify the DIH to update the host and port for a particular child server ID, for example to replace a child server that is on an inaccessible host.

To replace an inaccessible child server with a new host and port

- Send an `EngineManagement` action to the DIH ACI port, with `EngineAction` set to `Edit`, and the `Host` and `Port` parameters set to the new host and port. For example:

```
action=EngineManagement&EngineAction=Edit&Host=mynewhost&Port=9000
```

TIP: By default, when you modify a child server in this way, the DIH pings the specified host and port. The `Edit` action fails if the child server is not running. Set `Disabled` to `True` to skip this step and add the child server in the offline state.

For more information about `EngineManagement`, see [Add, Change, and Remove Child Servers in Consistent Hashing Mode, on page 51](#) and [Add, Update, and Remove Child Servers Dynamically, on page 37](#).

Rebuild a Child Server

When you have a working child server with the existing server ID, you can use the `DREREDISTRIBUTE` index action to rebuild the index.

To rebuild a child server from replicas

- Send the `DREREDISTRIBUTE` index action to the DIH index port, with the `RebuildEngine` parameter set to the ID of the child server that you want to rebuild. This index action restores the index of the child server from replicas in other child servers.

```
http://12.3.4.56:9071/DREREDISTRIBUTE?RebuildEngine=3
```

NOTE: If you have more than one replicas, you can rebuild multiple child servers, by setting `RebuildEngine` to a comma-separated list of IDs. These child servers can be from different engine groups, up to the number of replicas (for example, if you have three replicas, you can rebuild child servers from up to three engine groups).

However, if you use mirrored engine groups, it might be easier to rebuild a child server from a mirror in the same engine group, rather than using `DREREDISTRIBUTE`.

Manage the Index Queue

The DIH maintains an internal queue of unfinished index actions that it receives. By default, the index queue size grows if the DIH needs to accept new index actions (that is, it does not discard jobs that have not been completed).

To tune indexing performance, you can control certain characteristics of the queue.

- You can pre-size the DIH index queue by setting the `IndexQueueInitialSize` parameter in the `[IndexQueue]` section of the configuration file. When you know that DIH is going to receive a large number of index jobs, you can use this option to reduce the number of times that DIH resizes the index queue.

- You can limit the size of the index queue by setting the `IndexQueueMaxPendingItems` parameter in the `[IndexQueue]` section of the configuration file. This parameter limits the number of incomplete index actions on the queue. When the queue size reaches this limit, DIH rejects any further index actions. You can override the queue limit for an individual index action by using the `IgnoreMaxPendingItems` index action parameter.
- You can specify the maximum number of finished (historical) index actions by setting the `IndexQueueMaxHistory` configuration parameter in the `[IndexQueue]` section of the configuration file. In this case, DIH keeps the most recent finished actions, and removes the oldest.
- By default, the DIH periodically polls each job in the index queue to see whether it is complete. When a job is complete,
 - Specify how often the polling occurs by modifying the `PollInterval` configuration parameter in the `[Server]` section of the DIH configuration file. The default value is 10 seconds.

Frequent polling might help to keep the queue size down if you exceed the number of jobs specified in the `IndexQueueMaxHistory` configuration parameter. However, polling too frequently might affect indexing performance.
 - You can turn off polling completely by setting the `Polling` configuration parameter to **False**. In this case, DIH marks the job as complete as soon as it sends it to the child servers. DIH keeps the number of finished jobs specified by `IndexQueueMaxHistory`.

Turn off polling for all child servers by setting `Polling` to **False** in the `[Server]` section. Alternatively, turn off polling for individual child servers by setting `Polling` to **False** in the `[IDOLServerN]` section.
- You can specify a directory for the DIH to use to store index queue status files. By default, it stores the files in the `./main` directory, relative to the DIH executable. Set the `Main` configuration parameter (in the `[Paths]` section of the DIH configuration file) to the full or relative path to the directory to use.

Archive Information

You can archive records of index actions that the DIH receives and documents that the DIH was unable to index.

Archive Actions

Use the following procedure to set up archiving for index actions. The archived data includes any data that you send in the action.

To enable archiving of index actions

1. Open the DIH configuration file in a text editor.
2. In the `[Server]` section, set `ArchiveMode` to **True**.
3. In the `[Paths]` section, set `Archive` to the path (either absolute or relative to the DIH executable) to a directory to store the list of actions in.

4. In the [Server] section, set `ArchiveFolderDateFormat` to the IDOL date format string to use to name the folders. You can also use forward slashes (/) as path separators to specify subfolders. For example:

```
ArchiveFolderDateFormat=YYYY_MM_DD/HH/NN
```

In this example, an archive folder is produced every day, with a subfolder every hour, each with a further subfolder every minute. The default value is `YYYY_MM_DD`.

5. In the [Server] section, set `CompressArchive` to **True** if you want to compress the action archive to save disk storage space.
6. In the [Server] section, set `ArchiveUseHashDir` to **False** if you do not want to use a hashed directory structure in your archive folder (64 folders each containing 64 folders).

A hashed directory structure can avoid slow file operations that can occur in some file systems when there are many files in a single folder. If you have set `ArchiveFolderDateFormat` to give a small time interval for each folder, hashed directories might not be necessary.

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

Archive Failed Documents

Use the following procedure to set up archiving for documents that the DIH was unable to index.

To enable archiving of failed documents

1. Open the DIH configuration file in a text editor.
2. In the [Server] section, set the `ArchiveFailedTasks` parameter to **True**.
3. In the [Paths] section, set the `Failed` configuration parameter to the path (either absolute or relative to the DIH executable) to a directory to store the unindexed documents in.
4. Save and close the configuration file.
5. Restart the DIH for your changes to take effect.

Set Up SSL Connections

You can configure Secure Socket Layer (SSL) connections between the DIH 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 configuration file or the DIH configuration file, depending on whether the system uses a unified or stand-alone setup.

NOTE: In a unified IDOL setup, you must set the `SSLConfig` configuration parameter in the [Server] section, rather than the [DistributionSettings] section.

- [IDOLServerN] or [DIHEngineN]. Configure SSL in this section for connections for outgoing ACI

calls. You can set this option in either the IDOL configuration file or DIH configuration file, depending on whether the system uses a unified or stand-alone setup.

You can also configure SSL connections between DIH and the service port of the child servers by using the `ServiceSSLConfig` parameter in this section.

- `[IndexServer]`. Configure SSL in this section for connections for the index port.

To configure an SSL connection

1. Open the configuration file in a text editor. If you use a unified setup, use the IDOL Server configuration file. If you use a stand-alone setup, use the DIH configuration file.
2. Find the `[Server]`, `[IDOLServerN]`, or `[DIHEngineN]` section, or create an `[IndexServer]` section.
3. Add the `SSLConfig` setting to specify the section in which you set the SSL details for the connection, usually `SSLOptionN`. 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. In the `[IDOLServerN]` section or the `[DIHEngineN]` section, add the `ServiceSSLConfig` setting to the name of the section in which you set the SSL details for connections to the child server service port.
5. Create an `[SSLOptionN]` section for each unique `SSLConfig` or `ServiceSSLConfig` 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
```

6. Save and close the configuration file. Restart IDOL Server (for a unified setup) or the DIH (for a stand-alone setup) for your changes to take effect.

Configure Client Authorization

You can configure Distributed Index 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 Index Handler Reference*.

IMPORTANT: To ensure that Distributed Index Handler allows only the options that you configure in `[AuthorizationRoles]`, make sure that you delete any deprecated `RoleClients` 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 Index Handler for your changes to take effect.

IMPORTANT: If you do not provide any authorization roles for a standard role, Distributed Index 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 Index Handler uses only your action-based authorizations.

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 Index 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 Index 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.

Chapter 4: Operate the Distributed Index Handler

This chapter describes the actions you can perform with the DIH.

• Start and Stop the Distributed Index Handler	60
• Index Data with the DIH	62
• Administer IDOL Servers	66
• Manage Databases	69
• Manage Documents	69

Start and Stop the Distributed Index Handler

You can use several different methods to start and stop the DIH.

Start the Distributed Index Handler

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

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

Start the DIH on Microsoft Windows

- Double-click the `DIH.exe` file in your component installation directory.
- Start the DIH service from a system dialog box. DIH must be installed as a Windows Service. See [Install an IDOL Component as a Service on Windows, on page 15](#).
 1. Display the Windows **Services** dialog box.
 2. Select the **DIH** 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 DIH 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 DIH`
- On System V Linux platforms:
`service DIH start`
- On Solaris platforms (using System V):
`/etc/init.d/DIH start`

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

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

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

Stop the Distributed Index Handler

You can stop the DIH 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 DIH is running, and *servicePort* is the component service port (which is specified in the [Service] section of the Distributed Index 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 **DIH** service, and click **Stop** to stop Distributed Index 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 DIH`
 - On system V platforms:
`service DIH stop`
 - On Solaris platforms (using System V):

```
/etc/init.d/DIH stop
```

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

Index Data with the DIH

You can send either of two IDOL Server indexing actions to the DIH. For complete information about these index actions and their parameters, refer to the *IDOL Server Administration Guide* and the *Distributed Index Handler Reference*.

Use DREADD

The DIH accepts the DREADD indexing action, which allows you to send data (that is accessible to the DIH through the file system) directly to the DIH for indexing. Use the following action syntax:

```
http://DIHhost:IndexPort/DREADD?parameters
```

When you send this action to the DIH instead of to an IDOL Server, note that:

- *DIHhost* and *IndexPort* are the host and index port of the DIH, not of any of its child IDOL Servers.
- In mirror mode, DIH sends the data to every child IDOL server for indexing. In simple non-mirror mode, the DIH decides which IDOL server receives index data, but all IDOL servers receive all actions. Use reference-based indexing or `DistributeSendMinimal` to further reduce the amount of traffic and IDOL Server load by allowing the DIH to distribute index data and index actions.
- When you use the `PreserveDREADD` configuration option, all the child servers must have access to the index file.

Use DREADDATA

The DIH accepts the DREADDATA indexing action, which allows you to send data over a socket to the DIH for indexing. This action requires a POST request method. Use the following action syntax:

```
http://
DIHhost:IndexPort/DREADDATA?optionalParamsData#DREENDDATAkillDuplicatesOption\n\n
```

When you send this action to the DIH instead of to an IDOL Server, note:

- *DIHhost* and *IndexPort* are the host and index port of the DIH, not of any of its child IDOL Servers.
- In mirror mode, DIH sends the data to every child IDOL server for indexing. In simple non-mirror mode, the DIH decides which IDOL server indexes the data, but all IDOL servers receive all actions. Use reference-based indexing or `DistributeSendMinimal` to further reduce the amount of traffic and IDOL server load by allowing the DIH to distribute index data and index actions.

Check Indexing Status

You can check whether the DIH has successfully indexed data into the connected IDOL servers by running the following action:

```
http://DIHhost:ACIport/action=IndexerGetStatus
```

where,

- *DIHhost* is the IP address or host name of the machine on which DIH is installed.
- *ACIport* is the port that you use to send ACI actions to the DIH (set in the Port parameter in the [Server] section of the DIH configuration file).

The IndexerGetStatus action displays the status of the IDOL Server index queue.

Example

The following example shows the action response from the DIH when you send an IndexerGetStatus action to the DIH after a DREADD index action:

```
<autnresponse>
<action>INDEXERGETSTATUS</action>
<response>SUCCESS</response>
<responsedata>
<item>
  <id>1</id>
  <origin_ip>127.0.0.1</origin_ip>
  <received_time>2008/10/22 11:30:13</received_time>
  <start_time>2008/10/22 11:30:14</start_time>
  <end_time>2008/10/22 11:32:14</end_time>
  <duration_secs>120</duration_secs>
  <percentage_processed>100</percentage_processed>
  <status>-1</status>
  <description>Finished</description>
  <index_command>
    /DREADD?myfile.idx&KILLDUPLICATES=REFERENCE&DREDBNAME=Archive
  </index_command>
</item>
</responsedata>
</autnresponse>
```

where,

Tag name	Description
<id>	The ID number of the index action.
<origin_ip>	The IP address of the machine that sent the index action to the DIH.
<received_	The time that the DIH received the action.

Tag name	Description
time>	
<start_time>	The time that the DIH started processing the index action.
<end_time>	The time that the DIH finished processing the index action.
<duration_secs>	The total amount of time in seconds that the DIH spent processing the index action.
<status>	The status code of the current status of the index action in the DIH index queue. See IndexerGetStatus Status Codes , below.
<description>	The description of the <status> number.
<index_command>	The index action that was used for the index job.

IndexerGetStatus Status Codes

NOTE: Codes in **bold** are status messages. All other codes indicate there is a problem with the indexing process.

Code	Message	Explanation
-1	Finished	The indexing process is complete. By default, the action reports a job as finished only when all the child servers report the job as finished. This behavior can result in large queue sizes. To mark each job as finished when one child server completes it, set the <code>Polling</code> parameter in the <code>[Server]</code> section of the DIH configuration file to False .
-2	Out of disk space	IDOL Server ran out of disk space before the indexing process was completed.
-3	File not found	The index file was not found.
-4	Database not found	The database that you tried to index into was not found.
-5	Bad parameter	The indexing action syntax is incorrect.
-6	Database exists	The database that you tried to create already exists.
-7	Queued	DIH queues the indexing action and runs it when all preceding indexing actions are complete.
-8	Unavailable	IDOL Server is about to shut down or indexing is paused.
-9	Out of Memory	IDOL Server ran out of memory before the indexing process was completed.

Code	Message	Explanation
-10	Interrupted	The indexing action was interrupted.
-11	XML is not well formed	Indexing failed because the XML is not well formed.
-12	Retrying interrupted command	IDOL Server is running an indexing action that was previously interrupted.
-13	Backup in progress	IDOL Server is performing a backup.
-14	Max index size reached	The indexing job exceeds the maximum indexing size (your license determines the maximum indexing size).
-15	Max number of sections reached	The indexing job exceeds the maximum number of sections that you can index (your license determines the number of sections you can index).
-16	Indexing Paused	The indexing process was paused.
-17	Indexing Resumed	The indexing process was restarted.
-18	Indexing Cancelled	The indexing process was cancelled.
-19	Out of file descriptors	IDOL Server ran out of file descriptors.
-20	LanguageType not found	The language type of the index data was not found.
-21	SecurityType not found	The security type of the index data was not found.
-22	Child engines returned differing messages	The child servers returned different messages to the DIH. This code is reported only by DIH.
-23	Badly formatted index command	The indexing action was rejected by a child server because the syntax is not valid.
-25	To be sent to DRE	DIH queues the index action to send to the IDOL Server. This code is reported only by DIH.
-26	DREADDATA: Data received did not include #DREENDDATA	The data in the DREADDATA action did not contain a #DREEDNDATA statement to indicate the end of the data.
-27	Command failed more times than the configured retry limit	The indexing action exceeded the maximum number of retries specified by the <code>MaximumRetries</code> parameter in the DIH configuration. This code is reported only by DIH.
-28	The index ID specified is invalid	The index ID returned by the child server is not valid. This code is reported only by DIH.
-29	Command was redistributed to sibling engines as this engine	The indexing action was sent to sibling servers because the child server was either unavailable or not accepting indexing jobs. This code is reported only by DIH.

Code	Message	Explanation
	was either unavailable or not accepting index jobs	
-30	Database name too long	The name of the database that you are indexing documents into is too long. The maximum length is 63 characters.
-31	Command ignored due to id match	The DREINITIAL action was not processed because it did not match the ID specified in the InitialID parameter.
-33		IDOL Server cannot create the database because it has reached the maximum number of databases. The maximum is 32,767.
-34	Pending commit	The indexing job is complete and the documents are available for searching after the next delayed synchronization cycle, which you specify in the DelayedSync parameter.
-35	Initializing	DIH is starting the indexing job. This code is reported only by DIH.
-36	Reading IDX	DIH is reading the IDX file from disk, prior to sending it to IDOL Server. This code is reported only by DIH.

NOTE: If the IndexerGetStatus action returns a positive number, this number indicates the percentage of the indexing queue that is complete.

Administer IDOL Servers

You can use the following IDOL Server index actions to administer the IDOL Servers managed by the DIH. Refer to the *IDOL Server Reference* for complete information on these actions and their parameters.

Implement Configuration Changes

When you make changes to the configuration files of one or more IDOL Servers, you can send the DRERESET index action to the DIH to ensure that all its child IDOL Servers implement the changes. Use the following action syntax:

```
http://DIHhost:IndexPort/DRERESET?
```

When you send this action to the DIH instead of to an IDOL server, note:

- *DIHhost* is the IP address (or name) of the machine on which the DIH is installed. *IndexPort* is the number of the port that you use to send index actions to the DIH.
- This action resets all child IDOL servers of the DIH.

Compact the IDOL Servers

The DRECOMPACT index action allows the DIH to reduce the space that documents take up in the IDOL Servers. The compact operation (similar to the defragmentation process) uses new documents to fill up the space that has been created through the deletion of other documents. Use the following action syntax:

```
http://DIHhost:IndexPort/DRECOMPACT?
```

When you send this action to the DIH instead of to an IDOL Server, note:

- *DIHhost* is the IP address (or name) of the machine on which the DIH is installed, and *IndexPort* is the number of the port that you use to send index actions to the DIH.
- This action compacts all child IDOL servers of the DIH.

Back up the IDOL Servers

The DREBACKUP index action allows the DIH to back up the data in its child IDOL Servers, to create a safe copies of the data. You can subsequently use a DREINITIAL index action to restore the backed up files to the IDOL Servers.

To back up IDOL Server

1. Send a DRECOMPACT index action to the DIH to compress the connected IDOL Servers (see [Compact the IDOL Servers, above](#)).
2. Send the following action from your Web browser to copy all the IDOL Server *.DB files to a new location:

```
http://DIHhost:IndexPort/DREBACKUP?Path=Path
```

When you send this action to the DIH instead of to an IDOL Server, note:

- *DIHhost* is the IP address (or name) of the machine on which the DIH is installed, *IndexPort* is the number of the port that you use to send index actions to the DIH, and *Path* is the path to the location where you want to create the IDOL Server backup.
- If the IDOL Servers are installed on different machines, you must ensure that the specified path is a valid directory path on each of the machines.
- If any two IDOL Servers are installed on one machine, you can specify a relative path. However, you must ensure that multiple IDOL Servers do not back up to the same location.

To restore backed-up data to IDOL Server

- Send the following action from your Web browser to restore the files to all databases on all child IDOL Servers:

```
http://DIHhost:IndexPort/DREINITIAL?Path=Path
```

When you send this action to the DIH instead of to an IDOL Server, note:

- *DIHhost* is the IP address (or name) of the machine on which the DIH is installed, *IndexPort* is the number of the port that you use to send index actions to the DIH, and *Path* is the path (previously specified in DRECOMPACT) to the location to store the IDOL Server backups.

Initialize the IDOL Servers

The DREINITIAL index action allows the DIH to delete the data that each child IDOL server contains and reset the server to the state it was in when first installed. Use the following action syntax:

```
http://DIHhost:IndexPort/DREINITIAL?
```

When you send this action to the DIH instead of to an IDOL server, note:

- *DIHhost* is the IP address (or name) of the machine on which the DIH is installed, and *IndexPort* is the number of the port that you use to send index actions to the DIH.
- All child servers of the DIH are reset.

Disable the IDOL Servers

You can use the EngineManagement action to dynamically take child servers offline (without restarting the DIH). Send the EngineManagement action with the EngineAction parameter set to **Edit** and the Disabled parameter set to **True** to temporarily disable a child server of the DIH:

```
http://DIHhost:ACIPort/action=EngineManagement&EngineAction=Edit&ID=1&Disabled=True
```

This action disables the server with ID 1. When a child server is disabled, the DIH continues to assign documents and queue index actions. It does not attempt to send them until you enable the server again, by sending another EngineManagement action. For example:

```
http://DIHhost:ACIPort/action=EngineManagement&EngineAction=Edit&ID=1&Disabled=False
```

This behavior is identical to when the DIH has lost contact with one of its child servers. It allows you to manually take a child server offline for maintenance.

Disable Index Actions

You can prevent the DIH from forwarding certain index actions to its child servers. For example, you might want to prevent a user from sending a DREINITIAL index action to all child servers of a DIH.

In the [IndexServer] section of the DIH configuration file, set `DisallowIndexCommands` to a comma-separated list of index actions that you do not want the DIH to forward. For example:

```
[IndexServer]
DisallowedIndexCommands=DREINITIAL ,DRECOMPACT
```

If a user sends these index actions to the DIH, it rejects them with a NOT AUTHORIZED error.

Manage Databases

You can use the following IDOL Server index actions to manage IDOL databases in the child IDOL Servers of your DIH. For more information on these actions and their parameters, refer to the *Distributed Index Handler Reference*.

Create a New Database in the IDOL Servers

The DRECREATEDBASE index action allows the DIH to create a new database in each of its IDOL Servers. For example, you can create a database to store documents that relate to one particular subject, or to store documents that are relevant to a particular user group. Use the following action syntax:

```
http://DIHhost:IndexPort/DRECREATEDBASE?DREDbName=DatabaseName
```

When you send this action to the DIH instead of to an IDOL Server, note:

- *DIHhost* is the IP address (or name) of the machine on which the DIH is installed, *IndexPort* is the number of the port that you use to send index actions to the DIH, and *DatabaseName* is the name of the database to create.
- This action creates a database with the specified name in each of the child IDOL Servers. Using this action might make the most sense in situations where all the child IDOL Servers are identical.

Delete a Database and All the Documents that it Contains

The DREREMOVEDBASE action allows the DIH to delete an IDOL database and all the documents it contains. Use the following action syntax:

```
http://DIHhost:IndexPort/DREREMOVEDBASE?DREDbName=DatabaseName
```

When you send this action to the DIH instead of to an IDOL Server, note:

- *DIHhost* is the IP address (or name) of the machine on which the DIH is installed, *IndexPort* is the number of the port that you use to send index actions to the DIH, and *DatabaseName* is the name of the database to remove.
- This action removes the database from every child IDOL Server that has a database of the specified name. If all IDOL Servers are identical, all are treated equally. If the IDOL Servers are not all identical, this action might have unintended consequences.

Manage Documents

You can use the following IDOL Server index actions to manage IDOL database documents in the child IDOL Servers of your DIH. For complete information on these actions and their parameters, refer to the *Distributed Index Handler Reference*.

Delete Documents by Reference

The DREDELETEREF index action instructs the DIH to delete one or more documents—specified by reference—from the child IDOL Servers. Use the following action syntax:

```
http://
DIHhost
:
IndexPort
/DREDELETEREF?Docs=DocumentReferences&Field=FieldNames&DREDbName=DatabaseName
```

When you send this action to the DIH instead of to an IDOL Server, note:

- *DIHhost* is the IP address (or name) of the machine on which the DIH is installed, and *IndexPort* is the number of the port that you use to send index actions to the DIH.
- This action deletes documents with the specified references that have the specified fields (if specified) and that are in the specified database (if specified) from all child IDOL Servers of the DIH.

Delete and Restore Documents by Document ID

The DREDELETEDOC index action instructs the DIH to delete one or more documents—specified by document ID—from the child IDOL Servers. You can also send the DREUNDELETEDOC index action to restore previously deleted documents.

Use the following action syntax for DREDELETEDOC:

```
http://DIHhost:IndexPort/DREDELETEDOC?Docs=DocumentIDs
```

When you send this action to the DIH instead of to an IDOL server, note:

- *DIHhost* is the IP address (or name) of the machine on which the DIH is installed, and *IndexPort* is the number of the port that you use to send index actions to the DIH.
- This action deletes all documents with the specified IDs (or with IDs in the specified range) from all child IDOL Servers of the DIH.

NOTE: Because dissimilar documents in different databases can have the same document ID, using document IDs with this action can have unintended consequences.

After you send a DREDELETEDOC index action to the DIH, you can restore some or all of the deleted documents to their IDOL Server databases by sending a DREUNDELETEDOC action. Use the following action syntax for DREUNDELETEDOC:

```
http://DIHhost:IndexPort/DREUNDELETEDOC?Docs=DocumentIDs
```

You can specify the same set of IDs that you used with DREDELETEDOC, or a subset. This index action cannot restore documents if you have run a compact operation (DRECOMPACT) after running DREDELETEDOC.

When you send this action to the DIH instead of to an IDOL Server, note:

- *DIHhost* is the IP address (or name) of the machine on which the DIH is installed, and *IndexPort* is the number of the port that you use to send index actions to the DIH.
- This action restores all documents with the specified IDs (or with IDs in the specified range) to their IDOL Servers (child servers of the DIH).

Delete All Documents from a Database

The DREDELDBASE action instructs the DIH to delete all documents from a database. Use the following action syntax:

```
http://DIHhost:IndexPort/DREDELDBASE?DREDbName=DatabaseName
```

When you send this action to the DIH instead of to an IDOL Server, note the following points:

- *DIHhost* is the IP address (or name) of the machine on which the DIH is installed, and *IndexPort* is the number of the port that you use to send index actions to the DIH.
- This action deletes all documents from all databases of the specified name in all child IDOL Servers of the DIH. Using this action might make the most sense in situations where all the child IDOL databases are identical.

Expire Documents

The DREEXPIRE index action instructs the DIH to delete or archive documents that have reached a specified age. You might archive to ensure that the documents in your IDOL servers are current. Use the following action syntax:

```
http://DIHhost:IndexPort/DREEXPIRE?
```

When you send this action to the DIH instead of to an IDOL server, note:

- *DIHhost* is the IP address (or name) of the machine on which the DIH is installed, and *IndexPort* is the number of the port that you use to send index actions to the DIH.
- This action expires all documents that have passed the expiration age on all child IDOL servers of the DIH.

Change Document Field Values

The DREREPPLACE action instructs the DIH to change the values of fields in its child IDOL Server indexed documents.

NOTE: This action requires a POST request method.

Use the following action syntax:

```
http://DIHhost:IndexPort/DREREPPLACE?Data#DREENDDATA
```

When you send this action to the DIH instead of to an IDOL Server, note:

- *DIHhost* is the IP address (or name) of the machine on which the DIH is installed, and *IndexPort* is the number of the port that you use to send index actions to the DIH.

- Specify documents only by reference, not by document ID. Because dissimilar documents in different databases can have the same document ID, using document IDs with this action can have unintended consequences.

Change Document Metadata

The DRECHANGEMETA index action instructs the DIH to change the values of the importance rating, database, index date, or expiration date for specific documents in its child IDOL Servers.

Use the following action syntax:

```
http://  
DIHhost  
:IndexPort/DRECHANGEMETA?Type=metadataField&Refs=docReferences&NewValue=value
```

When you send this action to the DIH instead of to an IDOL server, note:

- *DIHhost* is the IP address (or name) of the machine on which the DIH is installed, and *IndexPort* is the number of the port that you use to send index actions to the DIH.
- This action allows you to specify documents by either document reference or document ID. However, unless the child IDOL servers are all identical, you must specify documents by reference only, not by document ID. Because dissimilar documents in different databases can have the same document ID, using document IDs with this action can have unintended consequences.
- This action also allows you to change the database that you assign a document to. If the child IDOL servers do not all have identical databases, this action can have unintended consequences.

Part II: Appendixes

This section includes the following appendixes:

- [Troubleshooting](#)

Appendix A: Troubleshooting

This section describes some common issues and their solutions. If your DIH is not functioning correctly, try the remedies described in this appendix.

- **The DIH cannot obtain a valid service port**

If the DIH cannot obtain a valid service port, it displays the following warning:

Warning: Engine *n*. Unable to obtain a valid service port.

To resolve this issue, verify that:

- the DIH has been correctly added to an authorization role in the IDOL Server configuration files that can send actions (for example, the Query standard role).
- the IDOL Servers are all version 4.5.0. or later.

- **Actions fail**

If the ACI actions that you send to the DIH fail, verify that:

- the DIH client has been correctly added to an authorization role in the IDOL Server configuration files that can send actions (for example, the Query standard role).
- the DIH client has been correctly added to an authorization role in the IDOL Server configuration files that can send service status actions (for example, the ServiceStatus standard role).

- **Index actions fail**

If the index actions that you send to the DIH fail, verify that the DIH client has been correctly added to an authorization role in the IDOL Server configuration files that can send index actions (for example, the Index standard role).

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.

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 component

The IDOL Server component that manages categorization and clustering.

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.

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,

Lotus Domino server, and so on), importing them to IDX format, and indexing them into an 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.

field

Fields define different parts of content in IDOL documents, such as the title, content, and metadata information.

I

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

A structured file format that can be indexed 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. You can store data in different field types (index, numeric, and ordinary fields) or prevent IDOL from storing it. It is important to store data in appropriate field types to ensure optimized performance. IDOL Server can return any fields it stores for queries. However, you can query only for terms in Index fields.

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.

M

mirror mode

A distribution mode in which DIH distributes to several identical copies of the Content component, for failover or load-balancing.

N

non-mirror mode

A distribution mode in which DIH distributes the index between several Content components, which all contain a different segment of the index.

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.

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 is used to identify a document. This might be a title or a URL, and allows IDOL to identify documents for retrieval, indexing, and deduplication.

ReferenceType field

Fields used to identify documents. At index time IDOL Server can use ReferenceType fields to eliminate duplicate copies of documents. At query time IDOL Server can use ReferenceType to filter results.

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 secure 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.

T

term

The basic entity that IDOL Server indexes (for example, a word in a document after IDOL applies stemming to it).

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.

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 Index Handler 12.7)

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!