



Digital Safe Connector

Software Version: 11.6

Administration Guide

Document Release Date: February 2018

Software Release Date: February 2018

Legal notices

Warranty

The only warranties for Seattle SpinCo, Inc. and its subsidiaries ("Seattle") products and services are set forth in the express warranty statements accompanying such products and services. Nothing herein should be construed as constituting an additional warranty. Seattle shall not be liable for technical or editorial errors or omissions contained herein. The information contained herein is subject to change without notice.

Restricted rights legend

Confidential computer software. Except as specifically indicated, valid license from Seattle required for possession, use or copying. Consistent with FAR 12.211 and 12.212, Commercial Computer Software, Computer Software Documentation, and Technical Data for Commercial Items are licensed to the U.S. Government under vendor's standard commercial license.

Copyright notice

© Copyright 2018 EntIT Software LLC, a Micro Focus company

Trademark notices

Adobe™ is a trademark of Adobe Systems Incorporated.

Microsoft® and Windows® are U.S. registered trademarks of Microsoft Corporation.

UNIX® is a registered trademark of The Open Group.

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 verify you are using the most recent edition of a document, go to

<https://softwaresupport.softwaregrp.com/group/softwaresupport/search-result?doctype=online help>.

This site requires you to sign in with a Software Passport. You can register for a Passport through a link on the site.

You will also receive updated or new editions if you subscribe to the appropriate product support service. Contact your Micro Focus sales representative for details.

Support

Visit the Micro Focus Software Support Online website at <https://softwaresupport.softwaregrp.com>.

This website provides contact information and details about the products, services, and support that Micro Focus offers.

Micro Focus online support provides customer self-solve capabilities. It provides 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 support website to:

- Search for knowledge documents of interest
- Submit and track support cases and enhancement requests
- Access the Software Licenses and Downloads portal
- Download software patches
- Access product documentation
- Manage support contracts

- Look up Micro Focus support contacts
- Review information about available services
- Enter into discussions with other software customers
- Research and register for software training

Most of the support areas require you to register as a Passport user and sign in. Many also require a support contract.

You can register for a Software Passport through a link on the Software Support Online site.

To find more information about access levels, go to

<https://softwaresupport.softwaregrp.com/web/softwaresupport/access-levels>.

About this PDF version of online Help

This document is a PDF version of the online Help.

This PDF file is provided so you can easily print multiple topics or read the online Help.

Because this content was originally created to be viewed as online help in a web browser, some topics may not be formatted properly. Some interactive topics may not be present in this PDF version. Those topics can be successfully printed from within the online Help.

Contents

Chapter 1: Introduction	7
Digital Safe Connector	7
Supported Actions	7
OEM Certification	8
Chapter 2: Install Digital Safe Connector	9
System Requirements	9
Install Digital Safe Connector on Windows	9
Install Digital Safe Connector on Linux	10
Configure the License Server Host and Port	11
Chapter 3: Configure Digital Safe Connector	12
Digital Safe Connector Configuration File	12
Modify Configuration Parameter Values	14
Include an External Configuration File	15
Include the Whole External Configuration File	15
Include Sections of an External Configuration File	16
Include a Parameter from an External Configuration File	16
Merge a Section from an External Configuration File	17
Encrypt Passwords	17
Create a Key File	17
Encrypt a Password	18
Decrypt a Password	19
Configure Client Authorization	20
Register with a Distributed Connector	21
Set Up Secure Communication	22
Configure Outgoing SSL Connections	22
Configure Incoming SSL Connections	23
Backup and Restore the Connector's State	23
Backup a Connector's State	24
Restore a Connector's State	24
Validate the Configuration File	24
Chapter 4: Start and Stop the Connector	26

Start the Connector	26
Verify that Digital Safe Connector is Running	27
GetStatus	27
GetLicenseInfo	27
Stop the Connector	27
 Chapter 5: Send Actions to Digital Safe Connector	 29
Send Actions to Digital Safe Connector	29
Asynchronous Actions	29
Check the Status of an Asynchronous Action	30
Cancel an Asynchronous Action that is Queued	30
Stop an Asynchronous Action that is Running	30
Store Action Queues in an External Database	31
Prerequisites	31
Configure Digital Safe Connector	32
Store Action Queues in Memory	33
Use XSL Templates to Transform Action Responses	34
Example XSL Templates	35
 Chapter 6: Use the Connector	 36
Insert Files into Digital Safe	36
 Chapter 7: Monitor the Connector	 38
IDOL Admin	38
Prerequisites	38
Supported Browsers	38
Install IDOL Admin	38
Access IDOL Admin	39
Use the Connector Logs	40
Customize Logging	40
Monitor Asynchronous Actions using Event Handlers	41
Configure an Event Handler	42
Write a Lua Script to Handle Events	43
Set Up Performance Monitoring	43
Configure the Connector to Pause	44
Determine if an Action is Paused	45
 Glossary	 46
 Send documentation feedback	 49

Chapter 1: Introduction

This section provides an overview of the Micro Focus Digital Safe Connector.

- [Digital Safe Connector](#) 7
- [Supported Actions](#) 7
- [OEM Certification](#) 8

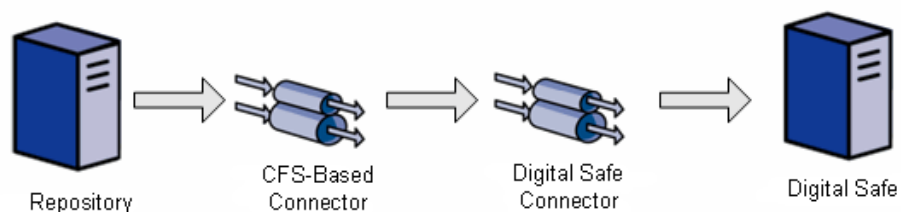
Digital Safe Connector

The Digital Safe Connector is an IDOL Connector that inserts information into Digital Safe.

NOTE:

The Digital Safe Connector is unlike other IDOL Connectors because it does not support the `synchronize` fetch action, to retrieve information from Digital Safe. You can use the `collect` or `view` fetch actions to retrieve information from Digital Safe.











You can use any IDOL Connector to retrieve files and then send the files to Digital Safe Connector, which places the files into Digital Safe.



Supported Actions

The Digital Safe Connector supports the following actions:

Action	Supported	Further Information
Synchronize	X	The Digital Safe Connector does not retrieve information from Digital Safe.
Synchronize (identifiers)	X	

Synchronize Groups		
Collect		
Identifiers		
Insert		Insert Files into Digital Safe, on page 36
Delete/Remove		
Hold/ReleaseHold		
Update		
Stub		
GetURI		
View		

OEM Certification

Digital Safe Connector works in OEM licensed environments.

Chapter 2: Install Digital Safe Connector

This section describes how to install the Digital Safe Connector.

• System Requirements	9
• Install Digital Safe Connector on Windows	9
• Install Digital Safe Connector on Linux	10
• Configure the License Server Host and Port	11

System Requirements

For information about the minimum system requirements required to run IDOL components, including Digital Safe Connector, refer to the *IDOL Getting Started Guide*.

Install Digital Safe Connector on Windows

To install the Digital Safe Connector on Windows, use the following procedure.

To install the Digital Safe Connector

1. Run the Digital Safe Connector installation program.
The installation wizard opens.
2. Read the installation instructions and click **Next**.
The License Agreement dialog box opens.
3. Read the license agreement. If you agree to its terms, click **I accept the agreement** and click **Next**.
The Installation Directory dialog box opens.
4. Choose an installation folder for Digital Safe Connector and click **Next**.
The Service Name dialog box opens.
5. In the **Service name** box, type a name to use for the connector's Windows service and click **Next**.
The Service Port and ACI Port dialog box opens.
6. Type the following information, and click **Next**.

Service port	The port used by the connector to listen for service actions.
ACI port	The port used by the connector to listen for actions.

The License Server Configuration dialog box opens.

7. Type the following information, and click **Next**.

License server host The host name or IP address of your License server.

License server port The ACI port of your License server.

The IDOL database dialog box opens.

8. In the **IDOL Database** box, type the name of the IDOL database that you want to index documents into, and click **Next**.

The Proxy Server dialog box opens.

9. If a proxy server is required to access the Digital Safe server, type the following information and click **Next**. Otherwise, just click **Next**.

Proxy host The host name or IP address of the proxy server to use to access the repository.

Proxy port The port of the proxy server to use to access the repository.

Proxy username The user name to use to authenticate with the proxy server.

Proxy password The password to use to authenticate with the proxy server.

The Digital Safe Connector Task Configuration dialog box opens.

10. Type the following information, and then click **Next**.

Server The Digital Safe server URL, including port if necessary.

Domain Name The name of the domain hosting the Digital Safe.

Mail From The e-mail address from which Digital Safe e-mails appear to have been sent.

Mail To The e-mail address to which Digital Safe e-mails appear to have been sent.

The Pre-Installation Summary dialog box opens.

11. Review the installation settings. If necessary, click **Back** to go back and change any settings. If you are satisfied with the settings, click **Next**.

The connector is installed.

12. Click **Finish**.

Install Digital Safe Connector on Linux

To install the Digital Safe Connector, use the following procedure.

To install Digital Safe Connector on Linux

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

```
./ConnectorName_VersionNumber_Platform.exe --mode text
```
2. Follow the on-screen instructions. For information about the options that are specified during

installation, see [Install Digital Safe Connector on Windows](#). For more information about installing IDOL components, refer to the *IDOL Getting Started Guide*.

Configure the License Server Host and Port

Digital Safe Connector is licensed through License Server. In the Digital Safe Connector configuration file, specify the information required to connect to the License Server.

To specify the license server host and port

1. Open your configuration file in a text editor.
2. In the [License] section, modify the following parameters to point to your License Server.

LicenseServerHost The host name or IP address of your License Server.

LicenseServerACIPort The ACI port of your License Server.

For example:

```
[License]
LicenseServerHost=licenses
LicenseServerACIPort=20000
```

3. Save and close the configuration file.

Chapter 3: Configure Digital Safe Connector

This section describes how to configure the Digital Safe Connector.

• Digital Safe Connector Configuration File	12
• Modify Configuration Parameter Values	14
• Include an External Configuration File	15
• Encrypt Passwords	17
• Configure Client Authorization	20
• Register with a Distributed Connector	21
• Set Up Secure Communication	22
• Backup and Restore the Connector's State	23
• Validate the Configuration File	24

Digital Safe Connector Configuration File

You can configure the Digital Safe Connector by editing the configuration file. The configuration file is located in the connector's installation folder. You can modify the file with a text editor.

The parameters in the configuration file are divided into sections that represent connector functionality.

Some parameters can be set in more than one section of the configuration file. If a parameter is set in more than one section, the value of the parameter located in the most specific section overrides the value of the parameter defined in the other sections. For example, if a parameter can be set in "*TaskName* or *FetchTasks* or *Default*", the value in the *TaskName* section overrides the value in the *FetchTasks* section, which in turn overrides the value in the *Default* section. This means that you can set a default value for a parameter, and then override that value for specific tasks.

For information about the parameters that you can use to configure the Digital Safe Connector, refer to the *Digital Safe Connector Reference*.

Server Section

The `[Server]` section specifies the ACI port of the connector. It can also contain parameters that control the way the connector handles ACI requests.

Service Section

The `[Service]` section specifies the service port of the connector.

Actions Section

The `[Actions]` section contains configuration parameters that specify how the connector processes actions that are sent to the ACI port. For example, you can configure event handlers that run when an action starts, finishes, or encounters an error.

Logging Section

The `[Logging]` section contains configuration parameters that determine how messages are logged. You can use *log streams* to send different types of message to separate log files. The configuration file also contains a section to configure each of the log streams.

Connector Section

The `[Connector]` section contains parameters that control general connector behavior. For example, you can specify a schedule for the fetch tasks that you configure.

FetchTasks Section

The `[FetchTasks]` section lists the fetch tasks that you want to run. A *fetch task* is a task that retrieves data from a repository. Fetch tasks are usually run automatically by the connector, but you can also run a fetch task by sending an action to the connector's ACI port.

In this section, enter the total number of fetch tasks in the `Number` parameter and then list the tasks in consecutive order starting from 0 (zero). For example:

```
[FetchTasks]
Number=2
0=MyTask0
1=MyTask1
```

`[TaskName]` Section

The `[TaskName]` section contains configuration parameters that apply to a specific task. There must be a `[TaskName]` section for every task listed in the `[FetchTasks]` section.

DistributedConnector Section

The `[DistributedConnector]` section configures the connector to operate with the Distributed Connector. The Distributed Connector is an ACI server that distributes actions (*synchronize*, *collect* and so on) between multiple connectors.

For more information about the Distributed Connector, refer to the *Distributed Connector Administration Guide*.

ViewServer Section

The [ViewServer] section contains parameters that allow the connector's *view* action to use a View Server. If necessary, the View Server converts files to HTML so that they can be viewed in a web browser.

License Section

The [License] section contains details about the License server (the server on which your license file is located).

Related Topics

- [Modify Configuration Parameter Values, below](#)
- [Customize Logging, on page 40](#)

Modify Configuration Parameter Values

You modify Digital Safe Connector 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 Digital Safe Connector 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 Digital Safe Connector 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 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 to and name of the external configuration file, in quotation marks (""). You can use relative paths and network locations. After the configuration file name, 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 a Parameter from an External Configuration File

This method allows you to import a parameter from an external configuration file at a specified point in your configuration file. You can include a section or a single parameter in this way, but the value in the external file must exactly match what you want to use in your file.

To include a parameter 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 parameter from 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. After the configuration file name, add the name of the configuration section name 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
```

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 Digital Safe Connector 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, Digital Safe Connector uses the values in the [SharedSSLOptions] section of the external configuration file as the values in the [SSLOptions1] section of the Digital Safe Connector 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, Digital Safe Connector 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.

Encrypt Passwords

Micro Focus recommends that you encrypt all passwords that you enter into a configuration file.

Create a Key File

A key file is required to use AES encryption.

To create a new key file

1. Open a command-line window and change directory to the Digital Safe Connector installation folder.
2. At the command line, type:

```
autopassword -x -tAES -oKeyFile=./MyKeyFile.ky
```

A new key file is created with the name `MyKeyFile.ky`

CAUTION:

To keep your passwords secure, you must protect the key file. Set the permissions on the key file so that only authorized users and processes can read it. Digital Safe Connector must be able to read the key file to decrypt passwords, so do not move or rename it.

Encrypt a Password

The following procedure describes how to encrypt a password.

To encrypt a password

1. Open a command-line window and change directory to the Digital Safe Connector installation folder.
2. At the command line, type:

```
autopassword -e -tEncryptionType [-oKeyFile] [-cFILE -sSECTION -pPARAMETER]  
PasswordString
```

where:

Option	Description
<code>-t</code> <i>EncryptionType</i>	The type of encryption to use: <ul style="list-style-type: none">• Basic• AES For example: <code>-tAES</code> <div>NOTE: AES is more secure than basic encryption.</div>
<code>-oKeyFile</code>	AES encryption requires a key file. This option specifies the path and file name of a key file. The key file must contain 64 hexadecimal characters. For example: <code>-oKeyFile=./key.ky</code>
<code>-cFILE -sSECTION -pPARAMETER</code>	(Optional) You can use these options to write the password directly into a configuration file. You must specify all three options. <ul style="list-style-type: none">• <code>-c</code>. The configuration file in which to write the encrypted password.• <code>-s</code>. The name of the section in the configuration file in which to write the password.• <code>-p</code>. The name of the parameter in which to write the encrypted

Option	Description
	password. For example: -c./Config.cfg -sMyTask -pPassword
<i>PasswordString</i>	The password to encrypt.

For example:

```
autopassword -e -tBASIC MyPassword
```

```
autopassword -e -tAES -oKeyFile=./key.ky MyPassword
```

```
autopassword -e -tAES -oKeyFile=./key.ky -c./Config.cfg -sDefault -pPassword  
MyPassword
```

The password is returned, or written to the configuration file.

Decrypt a Password

The following procedure describes how to decrypt a password.

To decrypt a password

1. Open a command-line window and change directory to the Digital Safe Connector installation folder.
2. At the command line, type:

```
autopassword -d -tEncryptionType [-oKeyFile] PasswordString
```

where:

Option	Description
-t <i>EncryptionType</i>	The type of encryption: <ul style="list-style-type: none">• Basic• AES For example: -tAES
-oKeyFile	AES encryption and decryption requires a key file. This option specifies the path and file name of the key file used to decrypt the password. For example: -oKeyFile=./key.ky
<i>PasswordString</i>	The password to decrypt.

For example:

```
autopassword -d -tBASIC 9t3M3t7awt/J8A
```

```
autopassword -d -tAES -oKeyFile=./key.ky 9t3M3t7awt/J8A
```

The password is returned in plain text.

Configure Client Authorization

You can configure Digital Safe Connector 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 *Digital Safe Connector Reference*.

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 Digital Safe Connector for your changes to take effect.

Register with a Distributed Connector

To receive actions from a Distributed Connector, a connector must register with the Distributed Connector and join a *connector group*. A connector group is a group of similar connectors. The connectors in a group must be of the same type (for example, all HTTP Connectors), and must be able to access the same repository.

To configure a connector to register with a Distributed Connector, follow these steps. For more information about the Distributed Connector, refer to the *Distributed Connector Administration Guide*.

To register with a Distributed Connector

1. Stop the connector.
2. Open the connector's configuration file in a text editor.
3. In the [DistributedConnector] section, set the following parameters:

RegisterConnector	(Required) To register with a Distributed Connector, set this parameter to true .
HostN	(Required) The host name or IP address of the Distributed Connector.
PortN	(Required) The ACI port of the Distributed Connector.
DataPortN	(Optional) The data port of the Distributed Connector.
ConnectorGroup	(Required) The name of the connector group to join. The value of this parameter is passed to the Distributed Connector.
ConnectorPriority	(Optional) The Distributed Connector can distribute actions to connectors based on a priority value. The lower the value assigned to <i>ConnectorPriority</i> , the higher the probability that an action is assigned to this connector, rather than other connectors in the same connector group.
SharedPath	(Optional) The location of a shared folder that is accessible to all of the connectors in the <i>ConnectorGroup</i> . This folder is used to store the connectors' datastore files, so that whichever connector in the group receives an action, it can access the information required to complete it. If you set the <i>DataPortN</i> parameter, the datastore file is streamed directly to the Distributed Connector, and this parameter is ignored.

4. Save and close the configuration file.
5. Start the connector.

The connector registers with the Distributed Connector. When actions are sent to the Distributed Connector for the connector group that you configured, they are forwarded to this connector or to another connector in the group.

Set Up Secure Communication

You can configure Secure Socket Layer (SSL) connections between the connector and other ACI servers.

Configure Outgoing SSL Connections

To configure the connector to send data to other components (for example Connector Framework Server) over SSL, follow these steps.

To configure outgoing SSL connections

1. Open the Digital Safe Connector configuration file in a text editor.
2. Specify the name of a section in the configuration file where the SSL settings are provided:
 - To send data to an ingestion server over SSL, set the `IngestSSLConfig` parameter in the `[Ingestion]` section. To send data from a single fetch task to an ingestion server over SSL, set `IngestSSLConfig` in a `[TaskName]` section.
 - To send data to a Distributed Connector over SSL, set the `SSLConfig` parameter in the `[DistributedConnector]` section.
 - To send data to a View Server over SSL, set the `SSLConfig` parameter in the `[ViewServer]` section.

You can use the same settings for each connection. For example:

```
[Ingestion]
IngestSSLConfig=SSLOptions
```

```
[DistributedConnector]
SSLConfig=SSLOptions
```

3. Create a new section in the configuration file. The name of the section must match the name you specified in the `IngestSSLConfig` or `SSLConfig` parameter. Then specify the SSL settings to use.

<code>SSLMethod</code>	The SSL protocol to use.
<code>SSLCertificate</code>	(Optional) The SSL certificate to use (in PEM format).
<code>SSLPrivateKey</code>	(Optional) The private key for the SSL certificate (in PEM format).

For example:

```
[SSLOptions]
SSLMethod=TLSV1.2
SSLCertificate=host1.crt
SSLPrivateKey=host1.key
```

4. Save and close the configuration file.
5. Restart the connector.

Related Topics

- [Start and Stop the Connector, on page 26](#)

Configure Incoming SSL Connections

To configure a connector to accept data sent to its ACI port over SSL, follow these steps.

To configure an incoming SSL Connection

1. Stop the connector.
2. Open the configuration file in a text editor.
3. In the [Server] section set the `SSLConfig` parameter to specify the name of a section in the configuration file for the SSL settings. For example:

```
[Server]
SSLConfig=SSLOptions
```

4. Create a new section in the configuration file (the name must match the name you used in the `SSLConfig` parameter). Then, use the SSL configuration parameters to specify the details for the connection. You must set the following parameters:

`SSLMethod` The SSL protocol to use.

`SSLCertificate` The SSL certificate to use (in PEM format).

`SSLPrivateKey` The private key for the SSL certificate (in PEM format).

For example:

```
[SSLOptions]
SSLMethod=TLSV1.2
SSLCertificate=host1.crt
SSLPrivateKey=host1.key
```

5. Save and close the configuration file.
6. Restart the connector.

Related Topics

- [Start and Stop the Connector, on page 26](#)

Backup and Restore the Connector's State

After configuring a connector, and while the connector is running, you can create a backup of the connector's state. In the event of a failure, you can restore the connector's state from the backup.

To create a backup, use the `backupServer` action. The `backupServer` action saves a ZIP file to a path that you specify. The backup includes:

- a copy of the `actions` folder, which stores information about actions that have been queued, are running, and have finished.
- a copy of the configuration file.
- a copy of the connector's datastore files, which contain information about the files, records, or other data that the connector has retrieved from a repository.

Backup a Connector's State

To create a backup of the connectors state

- In the address bar of your Web browser, type the following action and press **ENTER**:

`http://host:port/action=backupServer&path=path`

where,

host The host name or IP address of the machine where the connector is running.

port The connector's ACI port.

path The folder where you want to save the backup.

For example:

`http://localhost:1234/action=backupServer&path=./backups`

Restore a Connector's State

To restore a connector's state

- In the address bar of your Web browser, type the following action and press **ENTER**:

`http://host:port/action=restoreServer&filename=filename`

where,

host The host name or IP address of the machine where the connector is running.

port The connector's ACI port.

filename The path of the backup that you created.

For example:

`http://localhost:1234/action=restoreServer&filename=./backups/filename.zip`

Validate the Configuration File

You can use the `ValidateConfig` service action to check for errors in the configuration file.

NOTE:

For the `ValidateConfig` action to validate a configuration section, Digital Safe Connector must have previously read that configuration. In some cases, the configuration might be read when a task is run, rather than when the component starts up. In these cases, `ValidateConfig` reports any unread sections of the configuration file as unused.

To validate the configuration file

- Send the following action to Digital Safe Connector:

`http://Host:ServicePort/action=ValidateConfig`

where:

Host is the host name or IP address of the machine where Digital Safe Connector is installed.

ServicePort is the service port, as specified in the `[Service]` section of the configuration file.

Chapter 4: Start and Stop the Connector

This section describes how to start and stop the Digital Safe Connector.

- [Start the Connector](#) 26
- [Verify that Digital Safe Connector is Running](#) 27
- [Stop the Connector](#) 27

NOTE:

You must start and stop the Connector Framework Server separately from the Digital Safe Connector.

Start the Connector

After you have installed and configured a connector, you are ready to run it. Start the connector using one of the following methods.

Start the Connector on Windows

To start the connector using Windows Services

1. Open the Windows Services dialog box.
2. Select the connector service, and click **Start**.
3. Close the Windows Services dialog box.

To start the connector by running the executable

- In the connector installation directory, double-click the connector executable file.

Start the Connector on UNIX

To start the connector on a UNIX operating system, follow these steps.

To start the connector using the UNIX start script

1. Change to the installation directory.
2. Enter the following command:

```
./startconnector.sh
```
3. If you want to check the Digital Safe Connector service is running, enter the following command:

```
ps -aef | grep ConnectorInstallName
```

This command returns the Digital Safe Connector service process ID number if the service is running.

Verify that Digital Safe Connector is Running

After starting Digital Safe Connector, you can run the following actions to verify that Digital Safe Connector is running.

- [GetStatus](#)
- [GetLicenseInfo](#)

GetStatus

You can use the `GetStatus` service action to verify the Digital Safe Connector is running. For example:

`http://Host:ServicePort/action=GetStatus`

NOTE:

You can send the `GetStatus` action to the ACI port instead of the service port. The `GetStatus` ACI action returns information about the Digital Safe Connector setup.

GetLicenseInfo

You can send a `GetLicenseInfo` action to Digital Safe Connector to return information about your license. This action checks whether your license is valid and returns the operations that your license includes.

Send the `GetLicenseInfo` action to the Digital Safe Connector ACI port. For example:

`http://Host:ACIport/action=GetLicenseInfo`

The following result indicates that your license is valid.

```
<autn:license>
  <autn:validlicense>true</autn:validlicense>
</autn:license>
```

As an alternative to submitting the `GetLicenseInfo` action, you can view information about your license, and about licensed and unlicensed actions, on the **License** tab in the Status section of IDOL Admin.

Stop the Connector

You must stop the connector before making any changes to the configuration file.

To stop the connector using Windows Services

1. Open the Windows Services dialog box.
2. Select the *ConnectorInstallName* service, and click **Stop**.

3. Close the Windows Services dialog box.

To stop the connector by sending an action to the service port

- Type the following command in the address bar of your Web browser, and press ENTER:

`http://host:ServicePort/action=stop`

host The IP address or host name of the machine where the connector is running.

ServicePort The connector's service port (specified in the [Service] section of the connector's configuration file).

Chapter 5: Send Actions to Digital Safe Connector

This section describes how to send actions to Digital Safe Connector.

• Send Actions to Digital Safe Connector	29
• Asynchronous Actions	29
• Store Action Queues in an External Database	31
• Store Action Queues in Memory	33
• Use XSL Templates to Transform Action Responses	34

Send Actions to Digital Safe Connector

Digital Safe Connector actions are HTTP requests, which you can send, for example, from your web browser. The general syntax of these actions is:

```
http://host:port/action=action&parameters
```

where:

- host* is the IP address or name of the machine where Digital Safe Connector is installed.
- port* is the Digital Safe Connector ACI port. The ACI port is specified by the `Port` parameter in the [Server] section of the Digital Safe Connector configuration file. For more information about the `Port` parameter, see the *Digital Safe Connector Reference*.
- action* is the name of the action you want to run.
- parameters* are the required and optional parameters for the action.

NOTE:

Separate individual parameters with an ampersand (&). Separate parameter names from values with an equals sign (=). You must percent-encode all parameter values.

For more information about actions, see the *Digital Safe Connector Reference*.

Asynchronous Actions

When you send an asynchronous action to Digital Safe Connector, the connector adds the task to a queue and returns a token. Digital Safe Connector performs the task when a thread becomes available. You can use the token with the `QueueInfo` action to check the status of the action and retrieve the results of the action.

Most of the features provided by the connector are available through `action=fetch`, so when you use the `QueueInfo` action, query the `fetch` action queue, for example:

```
/action=QueueInfo&QueueName=Fetch&QueueAction=GetStatus
```

Check the Status of an Asynchronous Action

To check the status of an asynchronous action, use the token that was returned by Digital Safe Connector with the `QueueInfo` action. For more information about the `QueueInfo` action, refer to the *Digital Safe Connector Reference*.

To check the status of an asynchronous action

- Send the `QueueInfo` action to Digital Safe Connector with the following parameters.

QueueName	The name of the action queue that you want to check.
QueueAction	The action to perform. Set this parameter to <code>GetStatus</code> .
Token	(Optional) The token that the asynchronous action returned. If you do not specify a token, Digital Safe Connector returns the status of every action in the queue.

For example:

```
/action=QueueInfo&QueueName=fetch&QueueAction=getstatus&Token=...
```

Cancel an Asynchronous Action that is Queued

To cancel an asynchronous action that is waiting in a queue, use the following procedure.

To cancel an asynchronous action that is queued

- Send the `QueueInfo` action to Digital Safe Connector with the following parameters.

QueueName	The name of the action queue that contains the action to cancel.
QueueAction	The action to perform . Set this parameter to <code>Cancel</code> .
Token	The token that the asynchronous action returned.

For example:

```
/action=QueueInfo&QueueName=fetch&QueueAction=Cancel&Token=...
```

Stop an Asynchronous Action that is Running

You can stop an asynchronous action at any point.

To stop an asynchronous action that is running

- Send the `QueueInfo` action to Digital Safe Connector with the following parameters.

QueueName	The name of the action queue that contains the action to stop.
QueueAction	The action to perform. Set this parameter to Stop .
Token	The token that the asynchronous action returned.

For example:

```
/action=QueueInfo&QueueName=fetch&QueueAction=Stop&Token=...
```

Store Action Queues in an External Database

Digital Safe Connector provides asynchronous actions. Each asynchronous action has a queue to store requests until threads become available to process them. You can configure Digital Safe Connector to store these queues either in an internal database file, or in an external database hosted on a database server.

The default configuration stores queues in an internal database. Using this type of database does not require any additional configuration.

You might want to store the action queues in an external database so that several servers can share the same queues. In this configuration, sending a request to any of the servers adds the request to the shared queue. Whenever a server is ready to start processing a new request, it takes the next request from the shared queue, runs the action, and adds the results of the action back to the shared database so that they can be retrieved by any of the servers. You can therefore distribute requests between components without configuring a Distributed Action Handler (DAH).

NOTE:

You cannot use multiple servers to process a single request. Each request is processed by one server.

NOTE:

Although you can configure several connectors to share the same action queues, the connectors do not share fetch task data. If you share action queues between several connectors and distribute synchronize actions, the connector that processes a synchronize action cannot determine which items the other connectors have retrieved. This might result in some documents being ingested several times.

Prerequisites

- Supported databases:
 - PostgreSQL 9.0 or later.
 - MySQL 5.0 or later.
- If you use PostgreSQL, you must set the PostgreSQL ODBC driver setting `MaxVarChar` to **0** (zero).

If you use a DSN, you can configure this parameter when you create the DSN. Otherwise, you can set the `MaxVarcharSize` parameter in the connection string.

Configure Digital Safe Connector

To configure Digital Safe Connector to use a shared action queue, follow these steps.

To store action queues in an external database

1. Stop Digital Safe Connector, if it is running.
2. Open the Digital Safe Connector configuration file.
3. Find the relevant section in the configuration file:
 - To store queues for all asynchronous actions in the external database, find the `[Actions]` section.
 - To store the queue for a single asynchronous action in the external database, find the section that configures that action.
4. Set the following configuration parameters.

`AsyncStoreLibraryDirectory` The path of the directory that contains the library to use to connect to the database. Specify either an absolute path, or a path relative to the server executable file.

`AsyncStoreLibraryName` The name of the library to use to connect to the database. You can omit the file extension. The following libraries are available:

- `postgresAsyncStoreLibrary` - for connecting to a PostgreSQL database.
- `mysqlAsyncStoreLibrary` - for connecting to a MySQL database.

`ConnectionString` The connection string to use to connect to the database. The user that you specify must have permission to create tables in the database. For example:

`ConnectionString=DSN=my_ASYNC_QUEUE`

or

`ConnectionString=Driver={PostgreSQL};
Server=10.0.0.1; Port=9876;
Database=SharedActions; Uid=user; Pwd=password;
MaxVarcharSize=0;`

For example:

```
[Actions]
AsyncStoreLibraryDirectory=acidlls
AsyncStoreLibraryName=postgresAsyncStoreLibrary
ConnectionString=DSN=ActionStore
```

5. If you are using the same database to store action queues for more than one type of component,

set the following parameter in the [Actions] section of the configuration file.

DatastoreSharingGroupName The group of components to share actions with. You can set this parameter to any string, but the value must be the same for each server in the group. For example, to configure several Digital Safe Connectors to share their action queues, set this parameter to the same value in every Digital Safe Connector configuration. Micro Focus recommends setting this parameter to the name of the component.

CAUTION:

Do not configure different components (for example, two different types of connector) to share the same action queues. This will result in unexpected behavior.

For example:

```
[Actions]
...
DatastoreSharingGroupName=ComponentType
```

6. Save and close the configuration file.

When you start Digital Safe Connector it connects to the shared database.

Store Action Queues in Memory

Digital Safe Connector provides asynchronous actions. Each asynchronous action has a queue to store requests until threads become available to process them. These queues are usually stored in a datastore file or in a database hosted on a database server, but in some cases you can increase performance by storing these queues in memory.

NOTE:

Storing action queues in memory improves performance only when the server receives large numbers of actions that complete quickly. Before storing queues in memory, you should also consider the following:

- The queues (including queued actions and the results of finished actions) are lost if Digital Safe Connector stops unexpectedly, for example due to a power failure or the component being forcibly stopped. This could result in some requests being lost, and if the queues are restored to a previous state some actions could run more than once.
- Storing action queues in memory prevents multiple instances of a component being able to share the same queues.
- Storing action queues in memory increases memory use, so please ensure that the server has sufficient memory to complete actions and store the action queues.

If you stop Digital Safe Connector cleanly, Digital Safe Connector writes the action queues from memory to disk so that it can resume processing when it is next started.

To configure Digital Safe Connector to store asynchronous action queues in memory, follow these steps.

To store action queues in memory

1. Stop Digital Safe Connector, if it is running.
2. Open the Digital Safe Connector configuration file and find the [Actions] section.
3. If you have set any of the following parameters, remove them:
 - AsyncStoreLibraryDirectory
 - AsyncStoreLibraryName
 - ConnectionString
 - UseStringentDatastore
4. Set the following configuration parameters.

UseInMemoryDatastore

A Boolean value that specifies whether to keep the queues for asynchronous actions in memory. Set this parameter to TRUE.

InMemoryDatastoreBackupIntervalMins

(Optional) The time interval (in minutes) at which the action queues are written to disk. Writing the queues to disk can reduce the number of queued actions that would be lost if Digital Safe Connector stops unexpectedly, but configuring a frequent backup will increase the load on the datastore and might reduce performance.

For example:

```
[Actions]
UseInMemoryDatastore=TRUE
InMemoryDatastoreBackupIntervalMins=30
```

5. Save and close the configuration file.

When you start Digital Safe Connector, it stores action queues in memory.

Use XSL Templates to Transform Action Responses

You can transform the action responses returned by Digital Safe Connector using XSL templates. You must write your own XSL templates and save them with either an .xsl or .tmpl file extension.

After creating the templates, you must configure Digital Safe Connector to use them, and then apply them to the relevant actions.

To enable XSL transformations

1. Ensure that the autnxslt library is located in the same directory as your configuration file. If the library is not included in your installation, you can obtain it from Micro Focus Support.
2. Open the Digital Safe Connector configuration file in a text editor.
3. In the [Server] section, ensure that the XSLTemplates parameter is set to true.

CAUTION:

If `XSLTemplates` is set to `true` and the `autnxs1t` library is not present in the same directory as the configuration file, the server will not start.

4. (Optional) In the `[Paths]` section, set the `TemplateDirectory` parameter to the path to the directory that contains your XSL templates. The default directory is `acitemplates`.
5. Save and close the configuration file.
6. Restart Digital Safe Connector for your changes to take effect.

To apply a template to action output

- Add the following parameters to the action:

<code>Template</code>	The name of the template to use to transform the action output. Exclude the folder path and file extension.
<code>ForceTemplateRefresh</code>	(Optional) If you modified the template after the server started, set this parameter to <code>true</code> to force the ACI server to reload the template from disk rather than from the cache.

For example:

```
action=QueueInfo&QueueName=Fetch
      &QueueAction=GetStatus
      &Token=...
      &Template=myTemplate
```

In this example, Digital Safe Connector applies the XSL template `myTemplate` to the response from a `QueueInfo` action.

NOTE:

If the action returns an error response, Digital Safe Connector does not apply the XSL template.

Example XSL Templates

Digital Safe Connector includes the following sample XSL templates, in the `acitemplates` folder:

XSL Template	Description
<code>LuaDebug</code>	Transforms the output from the <code>LuaDebug</code> action, to assist with debugging Lua scripts.

Chapter 6: Use the Connector

This section describes how to use the connector.

- [Insert Files into Digital Safe](#)36

Insert Files into Digital Safe

The connector's `insert` fetch action can insert files (usually, e-mail messages) into Digital Safe. To use the `insert` action, you must construct some XML that specifies the file to insert.

Use the `file` element to specify the content to insert. There are several ways that you can do this, for example specifying the path to a file or providing the content base-64 encoded. For more information about how to specify the source file, refer to the documentation for the `insert` action in the *Digital Safe Connector Reference*.

The following example would insert the file located at `C:\files\file.eml` into Digital Safe.

```
<insertXML>
  <insert>
    <property name="DOMAINNAME" value="testdomain1"/>
    <property name="SERVICEURL"
      value="http://server/nexch/services/ZANTAZ_StoreAndRetrieveService"/>
    <metadata name="MAILFROM" value="..." />
    <metadata name="RCPTTO" value="..." />
    <file>
      <type>tempfile</type>
      <displayname>c:\files\file.eml</displayname>
      <content>c:\files\file.eml</content>
    </file>
  </insert>
</insertXML>
```

The properties that you can set are described in the following table:

DOMAINNAME	(Optional) Set this property if you want to override the value of the <code>DomainName</code> parameter that is set in the connector's configuration file.
SERVICEURL	(Optional) Set this property if you want to override the value of the <code>StoreAndRetrieveServiceUrl</code> parameter that is set in the connector's configuration file.

The `MAILFROM` and `RCPTTO` metadata fields are optional but can be used to override the `MailFrom` and `RcptTo` parameters in the connector's configuration file.

Add this XML to the `insert` fetch action as the value of the `insertXML` action parameter. The XML must be URL encoded before being used in the action command. For example:

```
http://host:port/action=Fetch&FetchAction=Insert
      &ConfigSection=MyTask
      &InsertXML=[URL encoded XML]
```

In the response to the `insert` action, the connector returns the `DOMAINNAME`, `SERVICEURL`, and `ZDK` (a Digital Safe document identifier) for the inserted document(s).

For more information about using the `insert` fetch action, refer to the *Digital Safe Connector Reference*.

Chapter 7: Monitor the Connector

This section describes how to monitor the connector.

• IDOL Admin	38
• Use the Connector Logs	40
• Monitor Asynchronous Actions using Event Handlers	41
• Set Up Performance Monitoring	43

IDOL Admin

IDOL Admin is an administration interface for performing ACI server administration tasks, such as gathering status information, monitoring performance, and controlling the service. IDOL Admin provides an alternative to constructing actions and sending them from your web browser.

Prerequisites

By default, the latest version of Digital Safe Connector should include the `admin.dat` file that is required to run IDOL Admin. If you do not have this file, you must download it separately.

Supported Browsers

IDOL Admin supports the following browsers:

- Internet Explorer 11 and later
- Edge
- Chrome (latest version)
- Firefox (latest version)

Install IDOL Admin

You must install IDOL Admin on the same host that the ACI server or component is installed on. To set up a component to use IDOL Admin, you must configure the location of the `admin.dat` file and enable Cross Origin Resource Sharing.

To install IDOL Admin

1. Stop the ACI server.
2. Save the `admin.dat` file to any directory on the host.
3. Using a text editor, open the ACI server or component configuration file. For the location of the configuration file, see the ACI server documentation.

4. In the [Paths] section of the configuration file, set the `AdminFile` parameter to the location of the `admin.dat` file. If you do not set this parameter, the ACI server attempts to find the `admin.dat` file in its working directory when you call the IDOL Admin interface.
5. Enable Cross Origin Resource Sharing.
6. In the [Service] section, add the `Access-Control-Allow-Origin` parameter and set its value to the URLs that you want to use to access the interface.

Each URL must include:

- the `http://` or `https://` prefix

NOTE:

URLs can contain the `https://` prefix if the ACI server or component has SSL enabled.

- The host that IDOL Admin is installed on
- The ACI port of the component that you are using IDOL Admin for

Separate multiple URLs with spaces.

For example, you could specify different URLs for the local host and remote hosts:

```
Access-Control-Allow-Origin=http://localhost:9010  
http://Computer1.Company.com:9010
```

Alternatively, you can set `Access-Control-Allow-Origin=*`, which allows you to access IDOL Admin using any valid URL (for example, `localhost`, direct IP address, or the host name). The wildcard character (*) is supported only if no other entries are specified.

If you do not set the `Access-Control-Allow-Origin` parameter, IDOL Admin can communicate only with the server's ACI port, and not the index or service ports.

7. Start the ACI server.

You can now access IDOL Admin (see [Access IDOL Admin, below](#)).

Access IDOL Admin

You access IDOL Admin from a web browser. You can access the interface only through URLs that are set in the `Access-Control-Allow-Origin` parameter in the ACI server or component configuration file. For more information about configuring URL access, see [Install IDOL Admin, on the previous page](#).

To access IDOL Admin from the host that it is installed on

- Type the following URL into the address bar of your web browser:

```
http://localhost:port/action=admin
```

where *port* is the ACI server or component ACI port.

To access IDOL Admin from a different host

- Type the following URL into the address bar of your web browser:

```
http://host:port/action=admin
```

where:

host is the name or IP address of the host that IDOL Admin is installed on.

port is the ACI server or component ACI port of the IDOL Admin host.

Use the Connector Logs

As the Digital Safe Connector runs, it outputs messages to its logs. Most log messages occur due to normal operation, for example when the connector starts, receives actions, or sends documents for ingestion. If the connector encounters an error, the logs are the first place to look for information to help troubleshoot the problem.

The connector separates messages into the following message types, each of which relates to specific features:

Log Message Type	Description
Action	Logs actions that are received by the connector, and related messages.
Application	Logs application-related occurrences, such as when the connector starts.
Collect	Messages related to the <code>Collect</code> fetch action.
Delete	Messages related to the <code>Delete</code> fetch action.
Insert	Messages related to the <code>Insert</code> fetch action.
Synchronize	Messages related to the <code>Synchronize</code> fetch action.
View	Messages related to the <code>View</code> action.

Customize Logging

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

To set up log streams

1. Open the Digital Safe Connector 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 *Digital Safe Connector 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.

Monitor Asynchronous Actions using Event Handlers

The fetch actions sent to a connector are asynchronous. Asynchronous actions do not run immediately, but are added to a queue. This means that the person or application that sends the action does not receive an immediate response. However, you can configure the connector to call an event handler when an asynchronous action starts, finishes, or encounters an error.

You can use an event handler to:

- return data about an event back to the application that sent the action.
- write event data to a text file, to log any errors that occur.

You can also use event handlers to monitor the size of asynchronous action queues. If a queue becomes full this might indicate a problem, or that applications are making requests to Digital Safe Connector faster than they can be processed.

Digital Safe Connector can call an event handler for the following events.

OnStart	The OnStart event handler is called when Digital Safe Connector starts processing an asynchronous action.
OnFinish	The OnFinish event handler is called when Digital Safe Connector successfully

finishes processing an asynchronous action.

- OnError** The `OnError` event handler is called when an asynchronous action fails and cannot continue.
- OnQueueEvent** The `OnQueueEvent` handler is called when an asynchronous action queue becomes full, becomes empty, or the queue size passes certain thresholds.
- A `QueueFull` event occurs when the action queue becomes full.
 - A `QueueFilling` event occurs when the queue size exceeds a configurable threshold (`QueueFillingThreshold`) and the last event was a `QueueEmpty` or `QueueEmptying` event.
 - A `QueueEmptying` event occurs when the queue size falls below a configurable threshold (`QueueEmptyingThreshold`) and the last event was a `QueueFull` or `QueueFilling` event.
 - A `QueueEmpty` event occurs when the action queue becomes empty.

Digital Safe Connector supports the following types of event handler:

- The `TextFileHandler` writes event data to a text file.
- The `HttpHandler` sends event data to a URL.
- The `LuaHandler` runs a Lua script. The event data is passed into the script.

Configure an Event Handler

To configure an event handler, follow these steps.

To configure an event handler

1. Stop the connector.
2. Open the connector's configuration file in a text editor.
3. Set the `OnStart`, `OnFinish`, `OnError`, or `OnQueueEvent` parameter to specify the name of a section in the configuration file that contains the event handler settings.
 - To run an event handler for all asynchronous actions, set these parameters in the `[Actions]` section. For example:

```
[Actions]
OnStart=NormalEvents
OnFinish=NormalEvents
OnError=ErrorEvents
```

- To run an event handler for specific actions, use the action name as a section in the configuration file. The following example calls an event handler when the *Fetch* action starts and finishes successfully:

```
[Fetch]
OnStart=NormalEvents
OnFinish=NormalEvents
```

4. Create a new section in the configuration file to contain the settings for your event handler. You must name the section using the name you specified with the `OnStart`, `OnFinish`, `OnError`, or `OnQueueEvent` parameter.
5. In the new section, set the `LibraryName` parameter.

`LibraryName` The type of event handler to use to handle the event:

- To write event data to a text file, set this parameter to `TextFileHandler`, and then set the `FilePath` parameter to specify the path of the file.
- To send event data to a URL, set this parameter to `HttpHandler`, and then use the HTTP event handler parameters to specify the URL, proxy server settings, credentials and so on.
- To run a Lua script, set this parameter to `LuaHandler`, and then set the `LuaScript` parameter to specify the script to run. For information about writing the script, see [Write a Lua Script to Handle Events, below](#).

For example:

```
[NormalEvents]
LibraryName=TextFileHandler
FilePath=./events.txt
```

```
[ErrorEvents]
LibraryName=LuaHandler
LuaScript=./error.lua
```

6. Save and close the configuration file. You must restart Digital Safe Connector for your changes to take effect.

Write a Lua Script to Handle Events

The Lua event handler runs a Lua script to handle events. The Lua script must contain a function named `handler` with the arguments `request` and `xml`, as shown below:

```
function handler(request, xml)
    ...
end
```

- `request` is a table holding the request parameters. For example, if the request was `action=Example&MyParam=Value`, the table will contain a key `MyParam` with the value `Value`. Some events, for example queue size events, are not related to a specific action and so the table might be empty.
- `xml` is a string of XML that contains information about the event.

Set Up Performance Monitoring

You can configure a connector to pause tasks temporarily if performance indicators on the local machine or a remote machine breach certain limits. For example, if there is a high load on the CPU or

memory of the repository from which you are retrieving information, you might want the connector to pause until the machine recovers.

NOTE:

Performance monitoring is available on Windows platforms only. To monitor a remote machine, both the connector machine and remote machine must be running Windows.

Configure the Connector to Pause

To configure the connector to pause

1. Open the configuration file in a text editor.
2. Find the `[FetchTasks]` section, or a `[TaskName]` section.
 - To pause all tasks, use the `[FetchTasks]` section.
 - To specify settings for a single task, find the `[TaskName]` section for the task.
3. Set the following configuration parameters:

<code>PerfMonCounterNameN</code>	The names of the performance counters that you want the connector to monitor. You can use any counter that is available in the Windows <code>perfmon</code> utility.
<code>PerfMonCounterMinN</code>	The minimum value permitted for the specified performance counter. If the counter falls below this value, the connector pauses until the counter meets the limits again.
<code>PerfMonCounterMaxN</code>	The maximum value permitted for the specified performance counter. If the counter exceeds this value, the connector pauses until the counter meets the limits again.
<code>PerfMonAvgOverReadings</code>	(Optional) The number of readings that the connector averages before checking a performance counter against the specified limits. For example, if you set this parameter to 5, the connector averages the last five readings and pauses only if the average breaches the limits. Increasing this value makes the connector less likely to pause if the limits are breached for a short time. Decreasing this value allows the connector to continue working faster following a pause.
<code>PerfMonQueryFrequency</code>	(Optional) The amount of time, in seconds, that the connector waits between taking readings from a performance counter.

For example:

```
[FetchTasks]
PerfMonCounterName0=\\machine-hostname\\Memory\\Available MBytes
PerfMonCounterMin0=1024
PerfMonCounterMax0=1024000
PerfMonCounterName1=\\machine-hostname\\Processor(_Total)\\% Processor Time
PerfMonCounterMin1=0
```

```
PerfMonCounterMax1=70  
PerfMonAvgOverReadings=5  
PerfMonQueryFrequency=10
```

NOTE:

You must set both a minimum and maximum value for each performance counter. You can not set only a minimum or only a maximum.

4. Save and close the configuration file.

Determine if an Action is Paused

To determine whether an action has been paused for performance reasons, use the QueueInfo action:

```
/action=queueInfo&queueAction=getStatus&queueName=fetch
```

You can also include the optional token parameter to return information about a single action:

```
/action=queueInfo&queueAction=getStatus&queueName=fetch&token=...
```

The connector returns the status, for example:

```
<autnresponse>  
  <action>QUEUEINFO</action>  
  <response>SUCCESS</response>  
  <responsedata>  
    <actions>  
      <action owner="2266112570">  
        <status>Processing</status>  
        <queued_time>2016-Jul-27 14:49:40</queued_time>  
        <time_in_queue>1</time_in_queue>  
        <process_start_time>2016-Jul-27 14:49:41</process_start_time>  
        <time_processing>219</time_processing>  
        <documentcounts>  
          <documentcount errors="0" task="MYTASK"/>  
        </documentcounts>  
        <fetchaction>SYNCHRONIZE</fetchaction>  
        <pausedforperformance>true</pausedforperformance>  
        <token>...</token>  
      </action>  
    </actions>  
  </responsedata>  
</autnresponse>
```

When the element `pausedforperformance` has a value of `true`, the connector has paused the task for performance reasons. If the `pausedforperformance` element is not present in the response, the connector has not paused the task.

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.

C

Category component

The IDOL Server component that manages categorization and clustering.

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.

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.

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.

Import

Importing is the process where CFS, using KeyView, extracts metadata, content, and sub-files from items retrieved by a connector. CFS adds the information to documents so that it is indexed into IDOL Server. Importing allows IDOL server to use the information in a repository, without needing to process the information in its native format.

Ingest

Ingestion converts information that exists in a repository into documents that can be indexed into IDOL Server. Ingestion starts when a connector finds new documents in a repository, or documents that have been updated or deleted, and sends this information to CFS. Ingestion includes the import process, and processing tasks that can modify and enrich the information in a document.

Intellectual Asset Protection System (IAS)

An integrated security solution to protect your data. At the front end, authentication checks

that users are allowed to access the system that contains the result data. At the back end, entitlement checking and authentication combine to ensure that query results contain only documents that the user is allowed to see, from repositories that the user has permission to access. For more information, refer to the IDOL Document Security Administration Guide.

K

KeyView

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

L

LDAP

Lightweight Directory Access Protocol. Applications can use LDAP to retrieve information from a server. LDAP is used for directory services (such as corporate email and telephone directories) and user authentication. See also: active directory, primary domain controller.

License Server

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

O

OmniGroupServer (OGS)

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

P

primary domain controller

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

V

View

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

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 Digital Safe Connector 11.6)

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!