IDOL

Software Version 12.13

Getting Started Guide

Document Release Date: October 2022
Software Release Date: October 2022

Legal notices

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:

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

Many areas of the portal require you to sign in. If you need an account, you can create one when prompted to sign in.

Contents

Part 1: IDOL Systems

This section describes Micro Focus IDOL server and describes the various set ups that you can use in your IDOL system.

Chapter 1: Introduction to IDOL

Micro Focus IDOL server integrates unstructured, semi-structured, and structured information from multiple repositories through an understanding of the content. It delivers a real time environment to automate operations across applications and content, removing all the manual processes involved in getting information to the right people at the right time.

Key IDOL Components

The following section describes the core components that IDOL systems use.

IDOL Server

The *Intelligent Data Operating Layer* (IDOL) gathers and processes unstructured, semi-structured, and structured information in any format from multiple repositories using IDOL connectors and a global relational index. It can automatically form a contextual understanding of the information in real time, linking disparate data sources together based on the concepts contained within them.

For example, IDOL can automatically link concepts contained in an e-mail message to a recorded phone conversation, which can be associated with a stock trade. This information is then imported into a format that is easily searchable, adding advanced retrieval, collaboration, and personalization to an application that integrates the technology.

OEM Certification

IDOL works in OEM licensed environments.

Connectors

Connectors enable automatic content aggregation from any type of local or remote repository (for example, a database, a Web site, a real-time telephone conversation and so on). Connectors form a unified solution across all information assets within the organization.

Security Overview

Micro Focus provides the software infrastructure that automates operations on unstructured information. This software infrastructure is based on IDOL Server.

Document security protects the information that you index into IDOL Server.

Your organization is likely to store information in many repositories. Many of these repositories have security features that apply permissions to files, so that they only authorized personnel can view them. Document security ensures that when you index information into IDOL Server, IDOL continues to enforce these permissions. In response to a query, IDOL returns only documents that a user is permitted to view.

IDOL includes the following features to protect your data:

- **User authentication**. At the front end, users must log on before they can query IDOL. IDOL can authenticate users against an existing directory service, such as Microsoft Active Directory.
- **Document security**. IDOL checks each document that is returned in response to a query. If the user who submitted the query does not have permission to view the document, the document is removed from the query results before the results are returned.
- **Secure communications**. You can encrypt the communications between ACI servers and any applications that use the Micro Focus IDOL ACI API.

Interfaces

- **IDOL Admin** allows you to administer an IDOL component. It provides a user interface for many common operations for various IDOL components. IDOL Admin is installed by the IDOL Server installer.
- **IDOL Site Admin** allows you to administer a wider IDOL installation, with multiple servers and components.
- **IDOL Data Admin** allows you to manage the content of data indexed in IDOL Servers. By creating and modifying different business projects, you can predefine which results are returned to users, and how the user views them. You can also view statistical information to help you refine your business projects and make them more effective.
- **Find** provides a basic end-user search application for IDOL. You can install Find by using the IDOL Server installer.
- **ACI API** uses HTTP to allow custom-built applications to communicate with IDOL ACI servers. Several IDOL SDKs are available to allow you to develop applications with the ACI API.

Related Topics

- Applications

Distributed Systems

IDOL distribution solutions facilitate linear scaling of systems through faster action execution and reduction of processing time.

- **DAH** (Distributed Action Handler) enables the distribution of ACI (Autonomy Content Infrastructure) actions to multiple IDOL Servers or components, providing failover and load balancing.
- **DIH** (Distributed Index Handler) enables distributed indexing of documents into multiple IDOL Servers or IDOL Content components, providing failover and load balancing.

Multimedia

IDOL Media Server allows you to incorporate information from video, image, and audio sources.

Media Server analyzes video files and streams, images, and audio to extract information about their content. Media Server can run analysis operations such as face recognition, number plate recognition, speech-to-text, and speaker identification.

IDOL Server Operations

IDOL Server can perform the following intelligent operations across structured, semistructured, and unstructured data.

Agents	Expertise
Alerts	Hyperlinks
Automatic Query Guidance	Mailing
Categorization	Profiles
Channels	Retrieval
Cluster Data	Spelling Correction
Collaboration	Summarization
Dynamic Clusters	Taxonomy Generation
Dynamic Thesaurus	Viewing
Eduction	

NOTE: Your license determines which of these operations your IDOL Server installation can perform.

Agents

Users can create agents in IDOL Server to find and monitor information that is relevant to their interests. The agents collect this information from a configurable list of Internet and intranet sites, news feeds, chat streams, and internal repositories.

Alerts

IDOL Server analyzes data when it receives new documents, and compares the concepts in documents with user agents. If new data matches a user agent, it immediately notifies the user by email or a third-party system (for example by SMS or a pager).

Automatic Query Guidance

IDOL Server finds the most salient terms and phrases in query results, and automatically clusters these terms and phrases. It uses the clustered phrases to provide a hierarchical set of queries that guide users to the result area that they are looking for.

Categorization

IDOL Server can automatically categorize data. IDOL categorization allows you to derive categories from the concepts found in unstructured text. This process ensures that IDOL Server accurately classifies all data in the correct context. IDOL categorization is a scalable solution capable of handling high volumes of information accurately and consistently.

Category Matching

IDOL Server accepts a category or piece of content and returns categories ranked by conceptual similarity. This ranking determines the most appropriate categories for the piece of content, so that IDOL Server can subsequently tag, route, or file the content accordingly.

Channels

IDOL Server can automatically provide users with a set of hierarchical channels with highly relevant information pertinent to the respective channel. Channels are similar to agents, aggregating information that is relevant to the channel concept. Usually, administrators set up channels that are available to all users.

Cluster Information

IDOL Server automatically clusters information. Clustering takes a large repository of unstructured data, agents, or profiles and automatically partitions the data to cluster similar information together.

Each cluster represents a concept area in the knowledge base, and contains a set of items with common properties.

Collaboration

IDOL Server automatically matches users with common explicit interest agents or similar implicit profiles. This information creates virtual expert knowledge groups.

Dynamic Clusters

When it processes queries, IDOL Server automatically clusters the query results, and then clusters the first set of clusters to produce subclusters. This process allows you to generate a hierarchy of clusters that allows users to navigate quickly to their area of interest.

Dynamic Thesaurus

When it processes queries, IDOL Server can automatically suggest alternative queries, allowing users to quickly produce a variety of relevant result sets.

Eduction

Eduction is a tool that you can use to extract an entity (a word, phrase, or block of information) from text, based on a pattern you define. The pattern can be a dictionary of names such as people or places. The pattern can also describe what the sequence of text looks like without listing it explicitly, for example, a telephone number. The entities are contained inside grammar files.

When you use Eduction with IDOL Server, Eduction extracts the entities while the document is indexed and adds them into fields for easy retrieval.

The Eduction capability of IDOL Server is described in the *Eduction User and Programming Guide*.

Expertise

IDOL Server accepts a natural language or Boolean search string and returns users who own matching agents or profiles. This process allows instant identification of experts in a subject, eliminating time-consuming searches for specialists, and unnecessary researching of subjects for which expert knowledge is already available.

Hyperlinks

You can automatically generate hyperlinks in real time. These link to contextually similar content, for example to recommend related articles, documents, affinity products or services, or media content that relates to textual content.

IDOL Server automatically inserts these links when it retrieves the document. This process means that new documents can reference older documents, and that archived documents can link to the latest news or material on the subject.

Email Users

IDOL Server matches the agents and profiles against its document content at regular intervals, and automatically notifies users of documents that match their agents or profiles by emailing them.

Profiles

IDOL Server automatically creates interest and expertise profiles for users, in real time.

You can create interest profiles by tracking the content that a user views and extracting a conceptual understanding of it. IDOL Server then uses this understanding to keep user interest profiles up to date. You can use interest profiles to:

- Target information to users.
- Recommend content to users.
- Alert users to the existence of content.
- Put users in touch with other users who have similar interests.

You can create expertise profiles by tracking the content that a user creates and extracting a conceptual understanding of it. IDOL Server uses this understanding to keep user expertise profiles up to date. You can use expertise profiles to trace users who are experts in particular subject areas.

Search and Retrieval

IDOL Server offers a range of retrieval methods, from simple legacy keyword search to sophisticated conceptual querying.

Examples of the kind of query that is possible with IDOL server include:

- Conceptual Matching
- Advanced Keyword Search
- Boolean/Bracketed Boolean Search
- Exact Phrase Search
- Field Restrictions
- Field Text Search
- Fuzzy Search
- Parametric Search
- Proper Names Search

- Proximity Search
- Soundex Keyword Search
- Synonym Search

Spell Check

IDOL Server can automatically spell check the query text it receives and suggest correct spelling for terms that its dictionary does not contain.

Summarization

IDOL Server accepts a piece of content and returns a summary of the information. IDOL Server can generate different types of summary.

- **Conceptual Summaries**. Conceptual summaries contain the most salient concepts of the content.
- **Contextual Summaries**. Contextual summaries relate to the context of the original query. They provide the most applicable dynamic summary in the results of a particular query.
- **Quick Summaries**. Quick summaries include a few sentences of the result documents.

Taxonomy Generation

Automatic taxonomy generation can automatically understand and create deep hierarchical contextual taxonomies of information. You can use clustering, or any other conceptual operation, as a seed for the process.

The resulting taxonomy can:

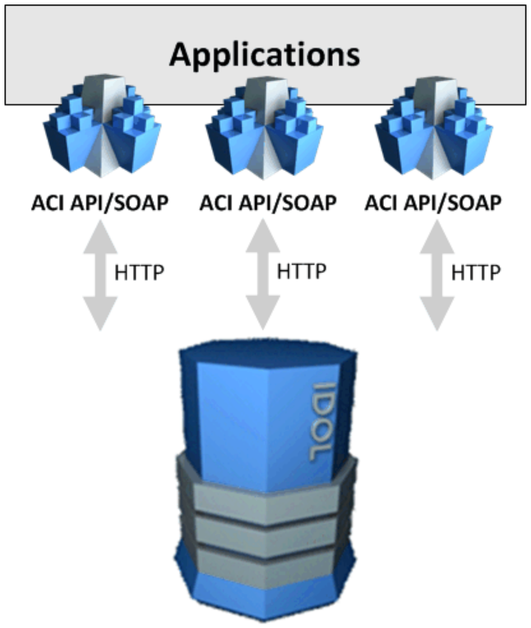
- Provide insight into specific areas of the information.
- Provide an overall information landscape.
- Act as training material for automatic categorization, which then places information into a formally dictated and controlled category hierarchy.

View Documents

IDOL Server uses IDOL KeyView filters to convert documents into HTML format for viewing in a Web browser.

IDOL System Architecture

IDOL server uses the ACI (Autonomy Content Infrastructure) Client API to communicate with custom-built applications that retrieve data using HTTP requests. It implements this communication over HTTP using XML and can adhere to SOAP.



Actions

When communicating with IDOL Server, there are two main types of action:

- ACI (Autonomy Content Infrastructure) actions
- Index actions

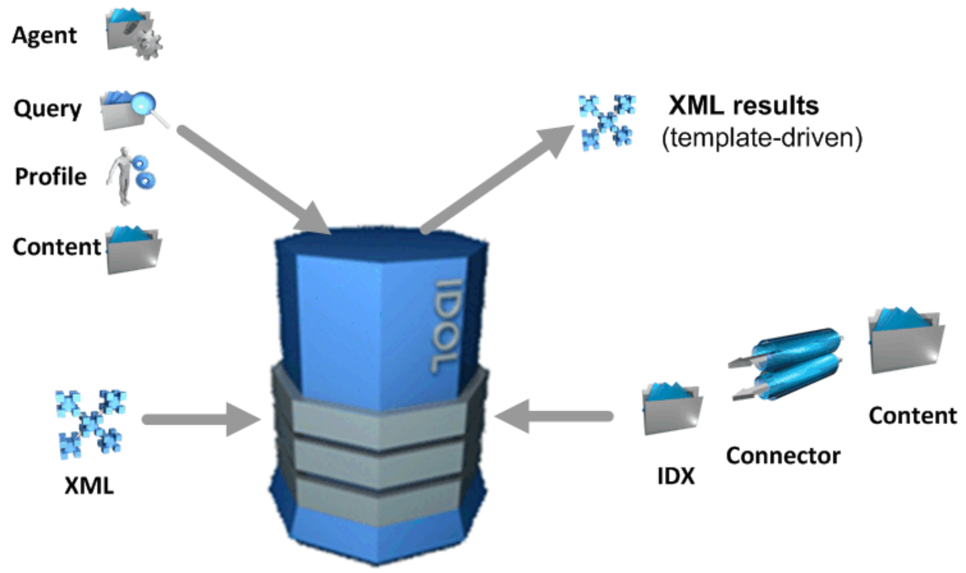
ACI actions request information or perform operations and return results. Different IDOL components accept different ACI actions. For example, the `Query` action requests results from the IDOL Content component, while the `UserRead` action requests user information from the IDOL Community component.

Index actions maintain the IDOL Server data index. You use them to index data, delete data, and perform operations on the data in IDOL Server.

Index and Query

You index documents into IDOL in IDOL IDX format or in XML format (directly or using a Connector). IDOL stores the concepts of the document. In response to queries, agents, profiles or content, it returns a link to the result document. IDOL also returns a percentage weighting, which indicates how relevant the result document is to the original query.

IDOL can return results as XML (even if the document was not in XML format when it was indexed) or other formats, such as plain text, using XSLT:



Security

It is often necessary to ensure that users can access or retrieve only data that they are authorized to view. IDOL enables you to set permissions for users or groups of users, to protect information, and ensure only the correct people can access it.

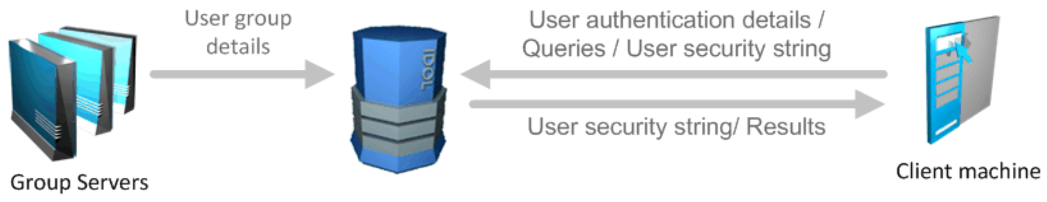
For more information on security, refer to the *IDOL Document Security Administration Guide*.

Text Queries

IDOL contains data that has been aggregated from one or more repositories. In this example each of the repositories has its own group server that stores the repository user names and the groups that these users belong to. IDOL aggregates this security information from the group servers.

When users log onto a client, their authentication details are sent to IDOL. IDOL returns the user security details to the client, which stores them until the client logs off or the session times out. Every time users send a text query from a client, the client attaches their security details to the query string that it sends to IDOL.

IDOL uses the security information in the query string to check the user permissions. It matches the security string against the document access control lists (ACLs). IDOL returns documents that match the query and that the user has permission to see.

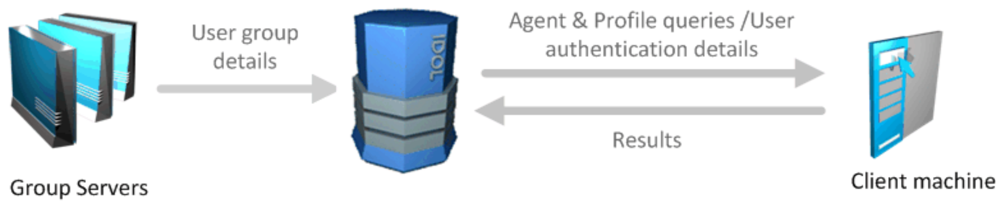

User group details
User authentication details / Queries / User security string
User security string/ Results
Group Servers
IDOL
Client machine

Agent, Profile and Category Queries

IDOL contains data aggregated from one or more repositories. In this example, each of the repositories has its own group server that stores the repository user names and the groups that these users belong to. IDOL aggregates this security information from the group servers.

The client sends an agent, profile, or category query to IDOL. IDOL (which stores all agents and profiles) matches this agent, profile or category against the documents it contains.

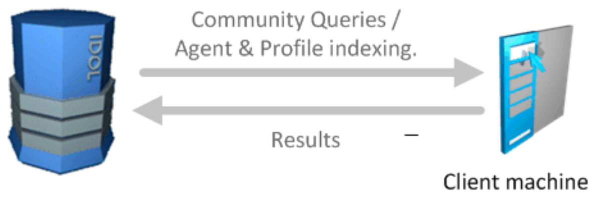
IDOL uses the information that it receives from the group servers to check the user permissions. It returns documents that match the query and that the user has permission to see.



Community Queries

IDOL stores user agents and profiles, to match them against community queries (that is, any type of query that returns agents or profiles).

When a client sends a community query to IDOL, IDOL matches it against the agents and profiles it stores. It returns matching agents, profiles, or both to the client.



Related Documentation

The following documents provide more details on IDOL.

Document	Description
IDOL Server Release Notes	Provides information about the new features and resolved issues in a release of IDOL Server.
IDOL Expert	Provides conceptual overviews and expert knowledge of IDOL and its features and functionality.
IDOL Server Administration Guide	Describes how to configure and use IDOL server.
IDOL Server Reference	Provides information about all the actions and configuration parameters that you can use in IDOL.
Content Component Reference *Category Component Reference* *Community Component Reference* *IDOL Proxy Component Reference* *View Server Reference*	Provides information about the actions and configuration parameters that you can use in each IDOL subcomponent.
IDOL Admin User Guide	Describes how to set up and use the IDOL Admin interface to administer the IDOL Content component.
IDOL Site Admin Installation Guide	Describes how to install set up IDOL Site Admin for monitoring and maintaining your IDOL installations.
IDOL Site Admin User Guide	Describes how to use IDOL Site Admin to monitor and maintain your IDOL installations.
Distributed Action Handler Administration Guide	Describes how to configure and use the Distributed Action Handler (DAH) with IDOL server.
Distributed Action Handler Reference	Provides information about all the actions and configuration parameters that you can use in the DAH.
Distributed Index Handler Administration Guide	Describes how to configure and use the Distributed Index Handler (DIH) with IDOL server.
Distributed Index	Provides information about all the actions and configuration parameters

Handler Reference	that you can use in the DIH.
Query Manipulation Server Administration Guide	Describes how to configure and use the Query Manipulation Server (QMS) to modify queries and results in IDOL server.
Query Manipulation Server Reference	Provides information about all the actions and configuration parameters that you can use in QMS.
License Server Administration Guide	Describes how to configure and use the License Server to manage IDOL services and licences.
License Server Reference	Provides information about all the actions and configuration parameters that you can use in the License Server.
Controller Reference	Provides information about all the actions and configuration parameters that you can use in the Controller component, which is used by IDOL Site Admin to monitor other IDOL components on a host.
Coordinator Reference	Provides information about all the actions and configuration parameters that you can use in the Coordinator component, which is used by IDOL Site Admin to monitor other hosts in an IDOL system.
IDOL Document Security Administration Guide	Describes how to set up and use security for IDOL server. This guide has been renamed from *Intellectual Asset Protection System (IAS) Administration Guide*.
OmniGroupServer Reference	Provides information about all the actions and configuration parameters that you can use in the OmniGroupServer.
Eduction User and Programming Guide	Describes how to set up and use Eduction to extract entities such as names, statistics, and locations from unstructured content.
Connector Framework Server Administration Guide	Describes how to configure and use Connector Framework Server (CFS) to convert data from connectors into index files, process content, and index data into IDOL server.
Connector Framework Server Reference	Provides information about all the actions and configuration parameters that you can use in the Connector Framework Server.
IDOL NiFi Ingest Help	Provides information about how to use NiFi Ingest to configure and run your document ingestion stream.
File System Connector Help *Web Connector Help*	Describes how to use IDOL connectors to aggregate information from different sources, so that it can be indexed into your IDOL index. The help also provides information about all the actions and configuration parameters that you can use with a connector.

HTTP Connector Help Other connector help, as needed.	
Media Server Administration Guide	Describes how to configure and use Media Server to process video, image, and audio data.
Media Server Reference	Provides information about all the actions and configuration parameters that you can use in Media Server.
Knowledge Graph Administration Guide	Describes how to configure and use the IDOL Knowledge Graph component to create and use a graph.
Knowledge Graph Reference	Provides information about all the actions and configuration parameters that you can use in the Knowledge Graph component.
Find Administration Guide	Provides information about how to install and set up the Find user interface.
IDOL Data Admin Administration Guide	Describes how to install and set up the IDOL Data Admin to manage the content of data indexed in IDOL servers.
IDOL Data Admin User Guide	Describes how to use IDOL Data Admin to manage the content of data indexed in IDOL servers.

Chapter 2: Types of IDOL Systems

You can set up IDOL systems in different ways to suit the requirements of your organization. This chapter discusses some of the possible IDOL setups, and some of the considerations involved when you choose a setup.

IDOL Core Components

This section describes the common IDOL components.

The following components form the core IDOL text indexing and processing functions. Together, they are often collectively known as *IDOL Server*. For more information, refer to the *IDOL Server Administration Guide*.

IDOL Text Processing

Service	Description
Content	Indexes, collects, manipulates and stores unstructured and semi-structured text data.
AgentStore	Indexes, collects, manipulates and stores agent and category information. Agentstore is a special configuration of the Content component.
Category	Stores predefined or customized categories in which data is organized.
Community	Stores information about users, roles, and permissions.
View	Converts documents to HTML format for viewing in a Web browser.

The following components form part of a wider IDOL infrastructure. They allow you to license, distribute, monitor, and control your IDOL components.

Infrastructure

Service	Description
License Server	Distributes and manages licenses for IDOL products and components. Refer to the *License Server Administration Guide* for more information.
IDOL Proxy	Routes actions to the correct components in a unified IDOL Server, and starts, restarts and stops IDOL components.
DAH	Distributed Action Handler. Distributes action requests across IDOL Servers or IDOL components. Refer to the *DAH Administration Guide* for more information.
DIH	Distributed Index Handler. Distributes indexing requests across IDOL Servers or IDOL Content components. Refer to the *DIH Administration Guide* for more information.
Controller	Monitors the IDOL services on a host machine, and communicates with Coordinator to report status information. This component is primarily used by IDOL Site Admin. Refer to the *IDOL Site Admin User Guide* for more information.
Coordinator	Manages status information from Controllers, and acts as a central point for viewing logs and monitoring the status of the IDOL system. This component is primarily used by IDOL Site Admin. Refer to the *IDOL Site Admin User Guide* for more information.

The following components allow you to automatically retrieve content and index into IDOL Server.

Connectors

Service	Description
CFS	Connector Framework Server. Aggregates data from IDOL connectors and indexes the data into IDOL Server. CFS can also manipulate the index content (for example, to additional fields relevant to the content), and index into other systems, such as Knowledge Graph and Vertica databases. Refer to the *Connector Framework Server Administration Guide* for more information.
NiFi Ingest	IDOL NiFi Ingest is an alternative to CFS. It uses Apache NiFi to allow you to easily configure and manipulate your data ingest process, from your connectors, to KeyView and other import processes (such as media analysis and Eduction), and your IDOL index.

Connectors, continued

Service	Description
	Refer to the *NiFi Ingest Help* for more information.
Connectors	Gather data from different sources for indexing into IDOL Server. The IDOL Server installer includes the File System Connector and the Web Connector. The File System Connector gathers data from system files, while the Web Connector gathers data from Web servers (internet or intranet). Refer to the Connector Administration guides for more details.

The following components perform additional operations and analysis on unstructured data. You can use many of these components with the IDOL Server text processing, or as part of an independent system.

Additional Services

Service	Description
QMS	Query Manipulation Server. Manages promotions, modifies queries to IDOL Server, and manipulates results from IDOL Server. Refer to the *Query Manipulation Server Administration Guide* for more information.
Statistics Server	Accumulates events from client applications or from a script that reads IDOL log files, then uses that data to report statistics. Refer to the *Query Manipulation Server Administration Guide* for more information.
Media Server	Analyzes video files and streams, images, and audio to extract information about their content. Media Server can run analysis operations such as face recognition, number plate recognition, speech-to-text, and speaker identification. Refer to the *Media Server Administration Guide* for more information.
Knowledge Graph	Uses connections between IDOL documents to create a set of relationships called a *graph*, and allows you to explore these connections in your data. Refer to the *Knowledge Graph Technical Note* for more information.
OGS	Omni Group Server. Collects security information from security repositories and uses it to ensure that users can access only documents that they have permission to access. Refer to the *IDOL Document Security Administration Guide* for more information.

IDOL System Setups

The flexibility of IDOL allows you to create systems that suit your environment and the unique needs of your organization. The following sections describe some of the ways that you can set up your IDOL system.

The installation procedures to support these setups are described in IDOL Installation and Setup, on page 42.

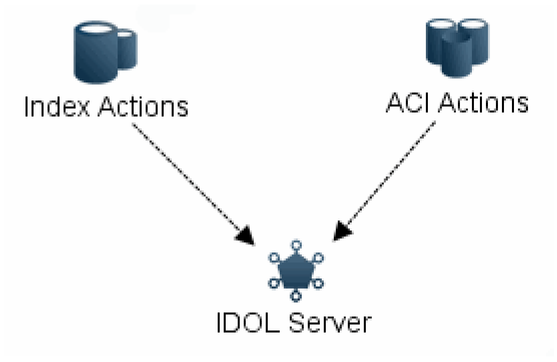
Unified Setup

A unified setup is the most basic setup, for training and test environments. It uses an integrated IDOL Server, rather than separate IDOL components. The integrated IDOL Server contains the core IDOL text indexing and processing functionality.

The unified IDOL Server includes the Content, Category, Community, AgentStore, View, and IDOL Proxy components. It does not include Query Manipulation Server (QMS), or the multimedia components, such as Media Server.

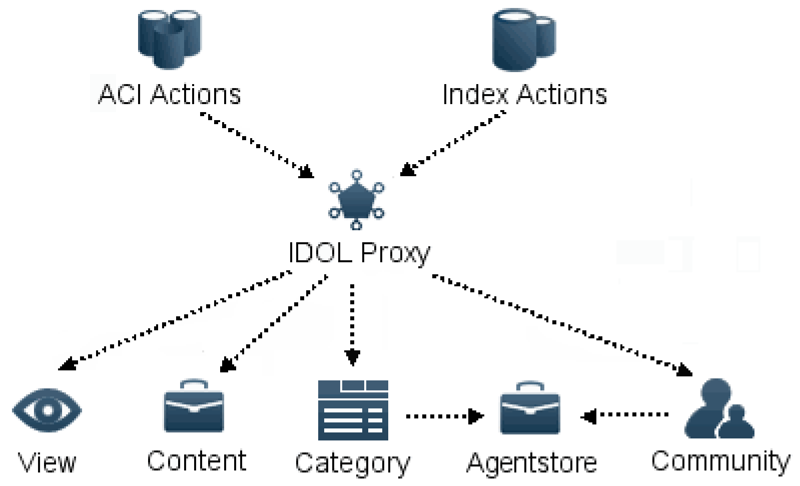
By default, it does not include the distribution components, but you can add these to a unified configuration if required.

In a unified setup, you configure all the operations for the IDOL Server components using a single configuration file (except for AgentStore, which has a separate configuration file). You send all actions and index actions to a single host and port, and IDOL Server stores and processes the data and actions centrally.



The unified setup uses the IDOL Proxy component. IDOL Proxy acts as a single point of contact for all your IDOL requests. It forwards all requests to the appropriate component. For example, it sends a `Query` action or indexing request to the Content component, and it sends a `UserRead` action to the Community component. IDOL Proxy also starts and stops the components in IDOL Server, and restarts components if they become inactive for any reason.

The following diagram shows the unified setup architecture in more detail:



This setup is useful for training and testing, because it allows you to easily modify the configuration for the server, and you do not have to understand the whole IDOL architecture. However, for most production environments, the unified IDOL setup is too small and restrictive. For example, it has limited scalability and no failover, it is difficult to add additional Content components to increase the size of the index, and you cannot include QMS.

When you want to set up a full IDOL system, you usually move to a component-based setup, where you configure and optimize components separately, and often on different hardware. For more information, see Component Setup, on the next page.

Related Topics

- Unified Installation, on page 43

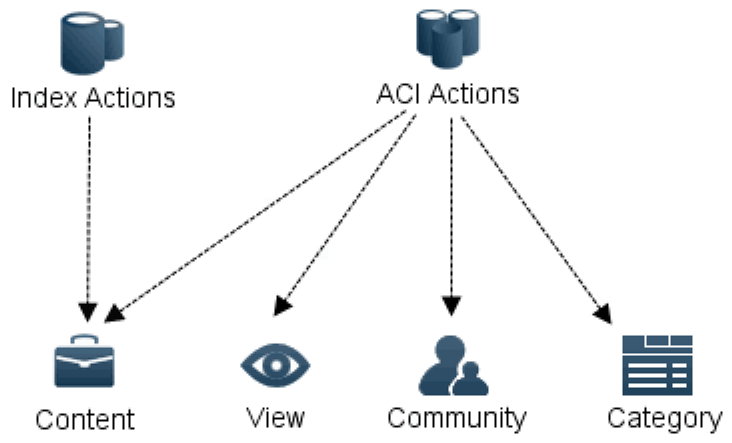
Component Setup

A component setup uses a combination of IDOL components, such as Content, Community, View, and so on, rather than the unified IDOL Server. This more advanced setup is for administrators who are already familiar with IDOL architecture.

Micro Focus recommends that you use a component-based setup in most production environments.

The component setup is more flexible than a unified IDOL Server. It allows you to:

- use only the IDOL components that you need.
- configure each component separately, which can be useful for optimization and tuning, as well as troubleshooting.
- set the components up on separate hardware or with dedicated resources to enhance the performance for different operations, and to allow you to scale resources for different components in a more flexible way.
- simplify component management and maintenance. In particular, it is easier to start, stop, and reinitialize individual components.
- design highly scalable, fault tolerant IDOL systems.



In a component setup, you configure each component with its own configuration file. For example, you configure the `content.exe` in the `content.cfg` file, you configure the `category.exe` in the `category.cfg` and so on.

Each component installation must also include any extra files or modules that you require. These additional files are included as part of the installation for the component if you use the IDOL Server installer, or you can use the zip package downloads, which also include the required files and modules.

> **NOTE:** There are also other dependencies between components, which you must configure in the component configuration files. For example, the Community and Category configuration files must contain the host and port details for the Agentstore component.

In general, Micro Focus does not recommend that you use the IDOL Proxy component in a component setup. However, if required, you can use a stand-alone IDOL Proxy to forward requests to the appropriate component. In this case, IDOL Proxy can only forward actions; it cannot perform any of the component maintenance (such as starting and stopping components).

If you use a stand-alone IDOL Proxy, the IDOL Proxy configuration file must contain the host IP address and ACI port for each of the IDOL components that you want it to forward requests to. It dynamically configures other ports (such as the index port) when the components start up.

Related Topics

- Distributed Setup, below
- Component Installation, on page 43
- Component ZIP Packages, on page 44

Distributed Setup

A distributed setup involves using a Distributed Index Handler (DIH) and Distributed Action Handler (DAH) to route actions to multiple instances of an IDOL component.

This kind of setup is effective for load-balancing among components, as well as for expanding indexes that no longer fit on one machine. DIH and DAH balance indexing and action requests among the appropriate components. You can either set up the distributed system in mirror mode or non-mirror mode.

- In mirror mode, each instance of the component that you distribute to is identical. You can use this option for load-balancing or failure tolerance.
- In non-mirror mode, each instance of the component is different. This option usually applies only to the IDOL Content component, to allow you to expand the size of your IDOL index.

The DIH distributes index actions (for example, to add or remove documents from your IDOL index). DIH can only distribute to components that have an index port.

The DAH distributes ACI actions. In mirror mode, it can distribute any action to any component. In non-mirror mode, it can distribute most IDOL Content component actions and combine the results of queries from multiple components.

The following sections provide more information about mirror mode and non-mirror mode for the DIH and DAH when distributing to multiple instances of the IDOL Content component.

For more information, refer to the *Distributed Index Handler Administration Guide*, and the *Distributed Action Handler Administration Guide*.

Mirror Mode

In a mirrored setup, you store the same set of data in each instance of the IDOL Content component. Each Content is identical to the others, and you must configure them in the same way.

Run the DIH in mirror mode to ensure uninterrupted service if one of theIDOL Content components becomes inoperable. While one Content is unavailable, the DIH continues to index data into its identical copies, which are also still available to return data for queries. DIH queues the actions for the inoperable Content, and sends them when the Content becomes available again, so that the servers do not become inconsistent.

DIH sends all index actions to all connected IDOL Content components.

In mirror mode, you can configure the DAH to distribute ACI actions in one of two ways:

- **Load Balancing**. The DAH assigns each incoming action to just one of the connected IDOL Content components (using a cumulative predictive algorithm that spreads the action load efficiently).
- **Failover**. The DAH forwards incoming actions to the first Content that you list in the DAH configuration file. If this IDOL Content component stops responding for any reason, the DAH marks it as down and switches to the next Content.

Non-Mirror Mode

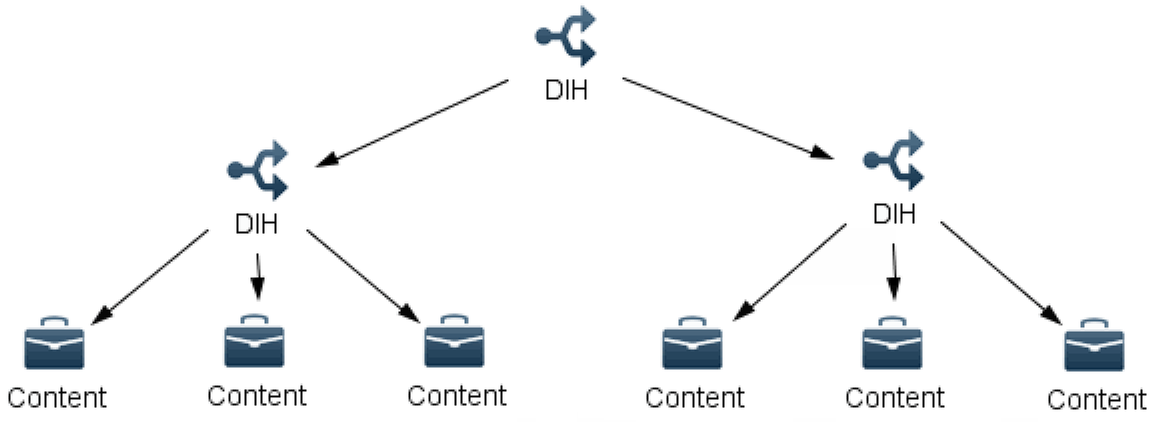
In a non-mirrored system, you distribute the data equally among the Content components.

Run the DIH in non-mirror mode if the amount of data to index is too large for a single Content. You can also separate the resources for each part of the index by setting the Content components up on different machines. This approach can improve the indexing time.

In non-mirror mode, the DAH sends ACI actions to all connected IDOL Servers. You can configure the DAH to combine the results in different ways when it returns them.

Chain Distribution Servers

You can set up multiple DIH and DAH instances in a chained configuration. For example, a parent DIH or DAH distributes actions to child DIH or DAH servers, which in turn distributes to child Content components.



In this configuration, the parent DIH and DAH distributes actions to child DIH and DAH servers in the same way as it distributes to child Content components. Each child DIH or DAH accepts all Content actions and forwards them.

Some actions may have a different effect when you send them to a child DIH or DAH server rather than a Content component, because the actions goes to multiple Content components.

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

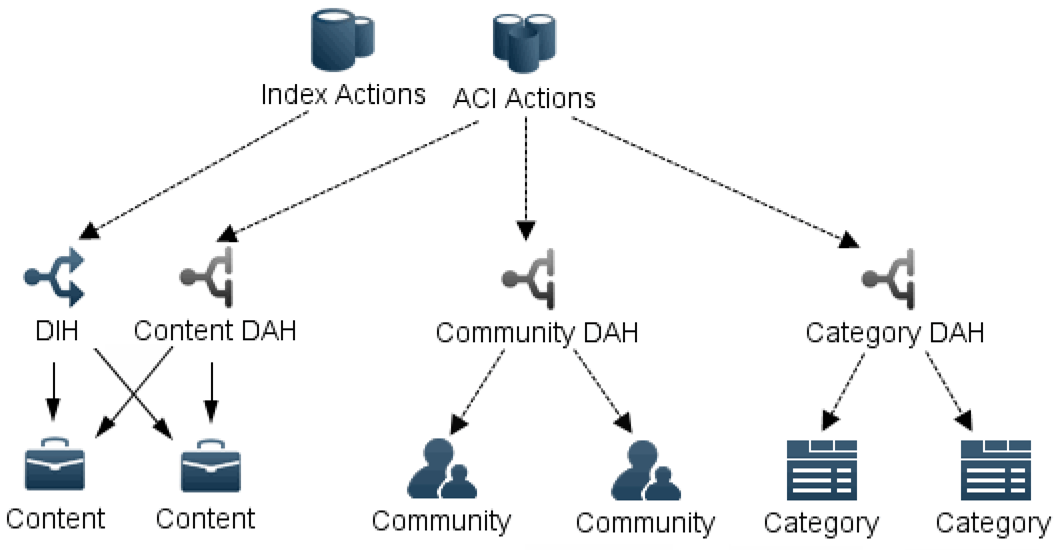
For more information about chaining distribution servers, and the architectural considerations, refer to *IDOL Expert*.

Distributed Component Setup

A distributed system with stand-alone components uses any combination of IDOL components with the DIH and DAH. You configure the IDOL components, DIH, and DAH separately with individual component configuration files.

This setup is highly flexible. You can distribute as many or as few of the IDOL components as you need to, and you can scale each component with additional instances as required by your usage. For example, if your system has a high load for categorization, you can increase the number of Category components without adding any other components.

The following diagram shows one example scenario. In this case, there are two Content components, two Community components, and two Category components.



In this example:

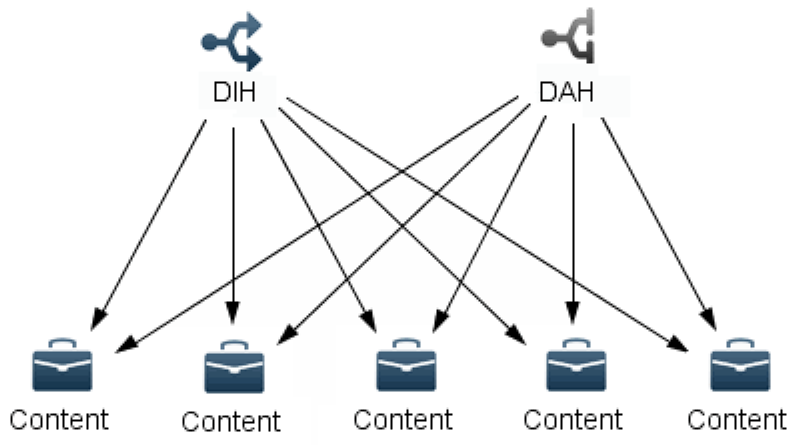
- You send index actions to the DIH, which distributes them between the two Content components.
- You route ACI actions between the three DAHs (for example, by using a front-end application, or an IDOL Proxy).
    - You send actions for Content (such as `Query`) to Content DAH, which distributes actions between the two Content components.
    - You send actions for Community (such as `UserRead`) to Community DAH, which distributes actions between the two Community components.

- You send actions for Category (such as `CategoryQuery`) to Category DAH, which distributes actions between the two Category components.

DAH cannot distribute all ACI actions in non-mirror mode, so the Community and Category components in this example must be mirrored.

Agentstore is not shown in this example, but you could use a single Agentstore for all the Category and Community components, or you can have Category and Community connect to an Agentstore DIH and Agentstore DAH. Agentstore indexes are not usually large, so you might not need to split the index into multiple Agentstore components, but you can create mirrored Agentstores for redundancy.

If you require only the basic indexing and retrieval functionality of IDOL Server, a more minimalist setup may be suitable. In this case, you can set up a number of Content servers, with DIH and DAH servers distributing index and ACI actions between them.:



Depending on the size or requirements of the system, you can either use mirror mode for fault tolerance, or non-mirror mode for performance. You can also use chained DIH and DAH servers to allow for both.

Related Topics

- Component Setup
- Component Installation, on page 43

Other IDOL Components

The previous sections have mainly considered the core IDOL Server text processing components, and distribution. IDOL also includes a large number of other components that allow you to analyze and process unstructured data, which you can use in combination with the text processing, or independently. This section describes the main components, and how you might want to include them in your IDOL system.

License Server

License Server is an essential part of any IDOL installation. It manages the licenses for your IDOL products and components.

When you purchase IDOL, Micro Focus Big Data support provide a license key. The license key is bound to a particular MAC address and port, which determines where you must install your License Server. The machine where you install License Server must be accessible over the network to the machines that contain the IDOL services that you want to license.

The only limit to the number of services that a single License Server can manage is your license capacity. In many cases you need only one License Server for your whole setup. However, in some cases you might want to use multiple license servers.

You might want to have multiple License Servers to license separate environments. For example, if you have a production and a test environment, these might have different License Servers. Alternatively, you might want to have a system where each machine that hosts IDOL services has its own License Server.

In these cases, each License Server must have its own license key, with a fixed MAC address and port.

NOTE: If you want to change or update your system, for example to include new components or features that change your license key, you must update every affected License Server. Similarly, if you need to change the host or port of a License Server, you must request a new license key from Micro Focus Big Data support.

License Server is available in the IDOL Server installer. For more information, see Install IDOL, on page 62, and the *License Server Administration Guide*.

Connectors and Connector Framework Server

IDOL Connectors and the Connector Framework Server (CFS) allow you to collect data from different repositories to import into IDOL.

- Connectors access a particular repository, and extract the content to send to CFS. Each connector is specific to a particular repository, rather than a particular file type. For example, you must use File System Connector to access a file system, and Web Connector to access files from the Internet, but both connectors can extract any available file content.
- Connector Framework Server processes the files from your connectors. CFS uses IDOL KeyView and the IDOL multimedia components to extract text from files, which it then sends to your IDOL index (the Content or DIH component). CFS can also perform data enrichment before you index to make it easier to retrieve information.

The number of connectors and CFS components that you need depends on the number of repositories you want to collect data from, and the amount of processing that you want to do.

CFS can process data from multiple connectors. For example, if you want to index content from a few small repositories, and the repositories do not change very often, you might set all the connectors up to send content to the same CFS, which indexes into IDOL Server.

In other cases, you might want to set up multiple CFS instances to process data from different connectors. If your repositories change regularly (generating a large number of new files and updates), or if they are very large, you can use one CFS for each connector.

The CFS is available in the IDOL Server installer. The IDOL Server installer also includes File System Connector and Web Connector. Other connectors are available as separate installers.

For more information, see Install IDOL, on page 62, and refer to the *Connector Framework Server Administration Guide*, as well as the administration guides for the individual connectors.

IDOL NiFi Ingest

IDOL NiFi Ingest provides an alternative to Connector Framework Server, with a graphical user interface to help you easily set up and configure your ingestion stream.

To use IDOL NiFi Ingest, you must install Apache NiFi and the IDOL NiFi Ingest package. You can then configure your ingest stream in the NiFi interface, and update your connector configurations to ingest into NiFi, rather than CFS.

For more information, refer to the *NiFi Ingest Help*.

Multimedia

IDOL Media Server allows you to process and analyze images, video, and audio files.

Media Server analyzes video files and streams, images, and audio to extract information about their content. Media Server can run analysis operations such as face recognition, number plate recognition, speech-to-text, and speaker identification.

You can use this component in combination with the IDOL text processing components. For example, you might want to use Media Server with CFS to transcribe audio files that you extract from a repository, and then index the text output into the Content component. You can also use Media Server independently of other IDOL components, for example for face recognition from video files.

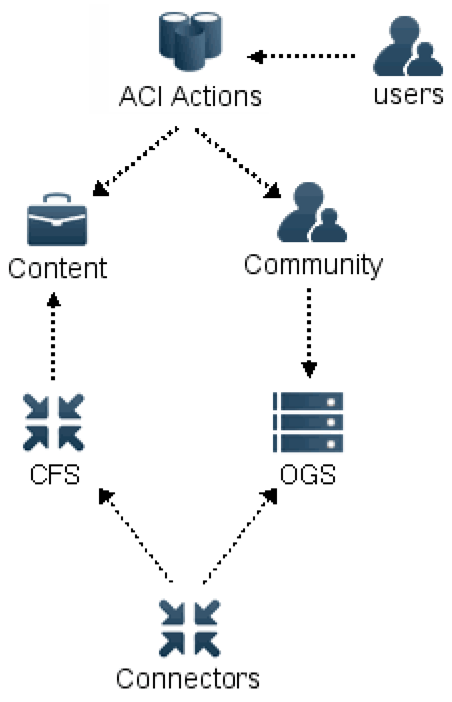
When you use Media Server with CFS, you must configure CFS with details of the component to use for processing. If necessary, you can use these components with a DAH for load-balancing and failover.

Media Server is available in the IDOL Server installer. For more information, see Install IDOL, on page 62. For more information about the components, refer to the *Media Server Administration Guide*, and the *Media Server Reference*.

Secured Search

To add secured search for IDOL Server, you might need to install Omni Group Server (OGS). OGS manages security for documents that you index in the Content component, and ensures that users can access only documents that they have permission to see. OGS can collect security information from several different security repositories and store this information, updating it at regular intervals.

When a user starts a session with the IDOL Community component, the client application requests security information. Community retrieves the user security details from OGS. The client application adds the security information to every subsequent query that it sends to the IDOL Content component. Content can then compare the user security details to the document access control lists (ACLs) to determine what type of access to grant.



Omni Group Server is available as a separate installer. For more information, see Security in IDOL, on page 49, and refer to the *IDOL Document Security Administration Guide*.

Setup for secured search requires configuration in the Content and Community components, and your connectors. For more information, refer to *IDOL Expert*.

Related Topics

- Mapped Security Example

Query Manipulation Server

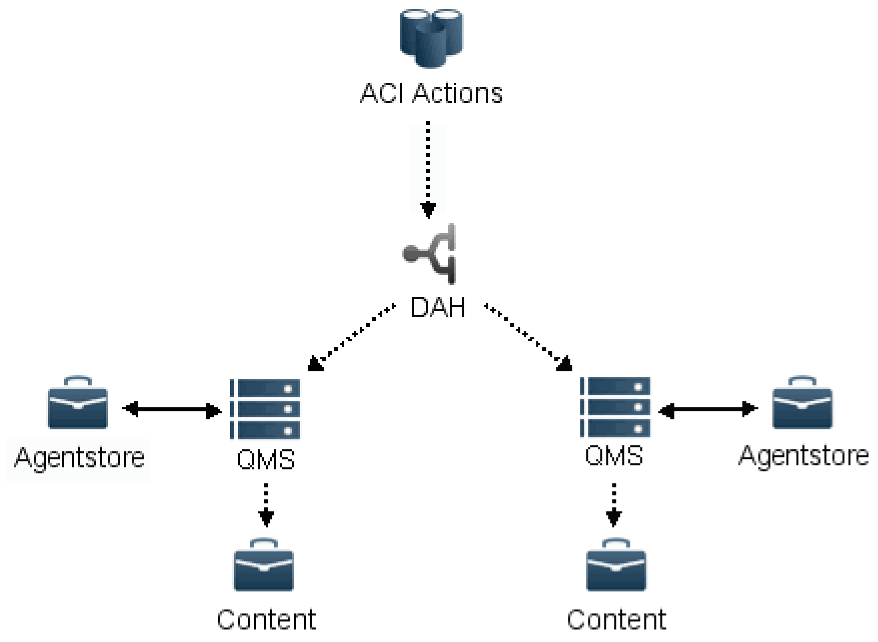
Query Manipulation Server (QMS) is an additional component that you can use with a Content index to modify user queries and results. For example, you can use QMS to remove certain terms from the query text, set up promotions to return when a user searches for a particular item, or to return a promoted item to a particular position in the search results.

QMS communicates with a Promotion Agentstore, which stores the rules that QMS uses to determine how to modify queries.

In an IDOL architecture, you send the Content actions that you want to modify to QMS, rather than directly to Content. QMS performs any additional processing, and then forwards the actions to Content.

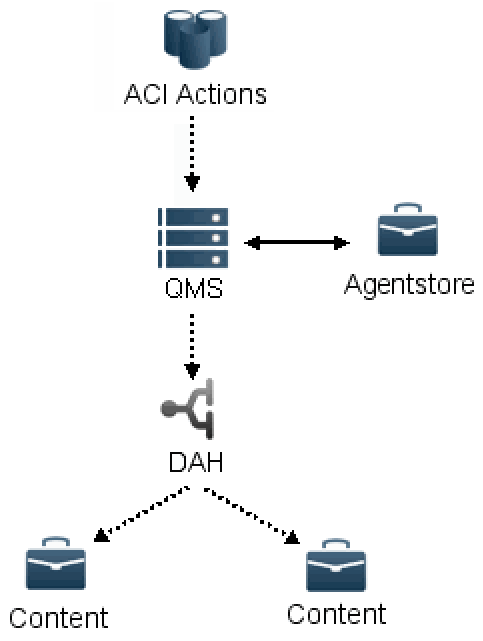
When you have configured distribution with your Content components, you must decide how to include QMS in the distributed architecture.

- You can use one QMS component for each Content, and direct the actions to a DAH that distributes to the QMS components.



In a mirror mode configuration with multiple identical Content components, this architecture ensures that QMS is part of the load-balancing or failover that you use for your system.

- You can use one QMS component, and then send the Content actions to the DAH after the QMS processing.



In a non-mirror mode configuration where your index is split over multiple Content components, this architecture reduces the number of additional components you need to configure. In a mirror mode configuration, you can use this architecture if you do not need to include load-balancing or failover for the QMS part of the system.

You can use QMS alone or as part of IDOL Data Admin. IDOL Data Admin provides a front-end application for manipulating and monitoring your queries. For more information, see IDOL Data Admin, on page 47.

QMS is available as part of the IDOL Server installer. For more information, see Install IDOL, on page 62, and refer to the *Query Manipulation Server Administration Guide*.

Statistics Server

Statistics Server monitors IDOL logs and collects information on different events.

You can use Statistics Server on its own, but typically you install it as part of an IDOL Data Admin installation. IDOL Data Admin uses Statistics Server to collect information about the queries and terms that you use in your IDOL installation. For more information about IDOL Data Admin, see IDOL Data Admin, on page 47.

Statistics Server is available as part of the IDOL Server installer. For more information, see Install IDOL, on page 62, and refer to the *Query Manipulation Server Administration Guide*.

Service Control

The Controller and Coordinator components form part of a wider monitoring architecture. Usually, you install these components as part of an IDOL Site Admin installation to monitor and control IDOL services. See IDOL Site Admin, on page 47.

- Controller monitors the IDOL services on a single host. In the typical case where you have your IDOL system distributed across multiple machines, you include a Controller server on each machine. Each Controller reports information to the central Coordinator server.
- Coordinator collects the monitoring information from your Controller servers. You have one Coordinator for the IDOL system that you want to monitor. You can then use the IDOL Site Admin application.

Controller and Coordinator are available as part of the IDOL Server installer. For more information, see Install IDOL, on page 62.

NOTE: The Controller component is selected by default in the installer, so that it is automatically deployed when you install components on a new host matchine.

For information on how to install IDOL Site Admin, refer to the *IDOL Site Admin Installation Guide*. For information about how to use Controller and Coordinator as part of IDOL Site Admin, refer to the *IDOL Site Admin User Guide*.

Knowledge Graph

Knowledge Graph can index content to create a graph of the relationships between concepts and entities in your data. You can use Knowledge Graph to provide additional tools for manipulating your unstructured data, and to model connections in a different way to the normal IDOL text processing methods.

You can configure CFS to index data into Knowledge Graph, so that you can create a graph from a set of data. Other than this, Knowledge Graph does not interact directly with other IDOL components in your architecture.

Front-End Applications

When you have multiple front-end applications, it can be beneficial to use separate IDOL installations for each application.

For example, you might have one application for collaboration in your organization, and another for data mining. You can set up each application to use a separate IDOL system, with any required components, distribution, and security.

Separating the applications in this way has several advantages:

- It is straightforward to use different data sets for each application, if required. Each different installation can have its own set of connectors, indexing data from the required sources. When the applications have different uses, this approach allows you to tailor the content for these different uses.
- It can be easier to manage security. For example you can separate an application for use on an intranet from another application that is accessible over the Internet.
- You can configure each IDOL system individually, to optimize performance and functionality for its associated front-end application. In particular, if the different applications have different usage and availability requirements, you can manage these requirements more easily in a separated system.
- You can optimize and scale resources more efficiently.
- It is straightforward to update your application or IDOL installation without causing interruptions to other applications.
- It is easier to test the effects of new configurations.

IDOL Installation and Setup

After you have decided on an IDOL architecture, you must decide on how to set up and distribute your system across multiple machines.

For the simple unified IDOL Server installation for testing, you can usually use a single machine for all components. However, in a more advanced system with a component-based setup, you usually separate some of the components on to separate disks or hardware.

In systems where you have large processing requirements for certain tasks, you can separate the IDOL components onto their own servers. This separation allows you to isolate and scale resources independently for different functions, according to their individual requirements. You can also place the components on their own hardware or virtual machine.

For more information about sizing and scaling your IDOL system, refer to *IDOL Expert*.

For full details about installing IDOL Server and the components, see Install IDOL, on page 62.

Unified Installation

The IDOL Server installer **Unified** installation option allows you to install a simple unified IDOL Server installation. This option has a very simple installation procedure, and installs:

- the main IDOL Server components (Content, Category, Community, Agentstore, and View)
- the IDOL Proxy component
- the unified IDOL Server configuration file, which contains standard configuration options for the main components
- the Agentstore configuration file, which contains standard configuration options for the IDOL Agentstore component
- all standard modules and libraries for the main IDOL Server components (such as language files, security modules, and KeyView filters)
- the IDOL Server `help.dat` file, which contains the *IDOL Server Reference*, *IDOL Server Administration Guide*, and *IDOL Expert*
- the `admin.dat` file that contains the IDOL Admin interface
- License Server, and the License Server configuration file and `help.dat` file

Component Installation

The IDOL Server installer **Custom** installation option allows you to individually select and install components, or sets of components. You can run the installer on each machine where you want to install a selection of IDOL components, and choose the appropriate components.

For each component, the installer provides:

- the component executable file
- a configuration file, which contains standard configuration options for the component
- all standard modules and libraries for the component
- the documentation for the component (in the `help.dat` file, which you can access by using the `Help` action)
- the IDOL Admin interface (in the `admin.dat` file, which you can access by using the `Admin` action)

When you run the installation process, you must select the components and any sub-components that you want to installer.

By default, the Controller component is selected. For IDOL Site Admin monitoring, you must include a Controller on every host machine where you install IDOL services. Clear the Controller installation option only if you have already installed Controller on a particular host.

Component ZIP Packages

IDOL components are also available to install as separate ZIP packages.

Each component ZIP package provides a configuration file, and the required modules for the component. It also includes a component-specific Reference, which contains the actions and configuration parameters relevant to that component.

To install the component, you extract the ZIP package to the host where you want to run the component. You must also update the `[License]` section of the configuration file to include appropriate license information. If you have more than one version of a component running on the same host, you might also need to update the port information to ensure that each component uses a different port.

After you install each of the components, you must modify the configuration files to ensure that the components can contact any other necessary components. For example, you must configure:

- the IDOL Category component with the details of the data index (IDOL Content component), and the category index (IDOL Agentstore component).
- the IDOL Community component with the details of the agent index (IDOL Agentstore component).
- the IDOL View component with the details of an IDOL Content component that it can use for highlighting.

IDOL Docker Containers

Micro Focus provides a set of Docker container images for IDOL components, as well as Docker Compose files that allow you to create an IDOL environment automatically from containers.

For more information about installing with Docker, and the available containers and options, see Install IDOL with Docker Images, on page 133.

Other Component Installations

The IDOL Server installer includes most of the common IDOL components and options. However, most IDOL connectors, some of the front-end applications, and Omni Group Server, are available in separate installers.

For more information about these components, refer to the documentation for the component.

Chapter 3: Applications

There are many applications that you can use to access, search, and customize the data in your IDOL system. This section describes some of the available applications that you can use to access and administer your IDOL system.

Front-End Applications

There are a variety of applications that provide a user interface through which you can access, customize, search, and modify data in your IDOL system.

Find and BIFHI are provided as part of the IDOL Server installer. Find provides a basic end-user search interface for IDOL, and BIFHI provides the search interface with some additional functionality. There are also several IDOL applications that use IDOL for different business purposes, such as archiving, data mining, and eDiscovery. For more information, contact your Account Manager.

In addition, you can create custom front-end applications to use your IDOL data.

Find

Find is a basic end-user search interface for IDOL. Find also includes Business Information for Human Intelligence (BIFHI).

Find supports the following functionality:

- **Advanced search**. Refine searches by database, date, or parametric values.
- **Document preview**. Find uses the IDOL View component to render near-native views of the source documents for your search results in a web browser. The Document Preview feature works with View server and documents in source repositories.
- **Query manipulation**. Find applies synonym and blacklist rules when you use the IDOL Query Manipulation Service as a back end for your search. You can use IDOL Data Admin to create and manage these rules.
- **Results manipulation**. Find applies pin-to-position promotions when you use the IDOL Query Manipulation Service as a back end for your search. You can use IDOL Data Admin to create and manage these promotions.
- **Results augmentation**. Find applies spotlight promotions when you use the IDOL Query Manipulation Service as a back end for your search. You can use IDOL Data Admin to create and manage these promotions.

- **Document security**. You can set up document security so that only users with appropriate permissions can access the documents, or even see them in search results.
- **Visualizations**. You can view results as topic maps, sunburst charts, maps, or in table format, as well as in list format.
- **Save searches**. You can save searches and run them again later, save a result set as a snapshot, or compare two or more saved searches.

NOTE: Visualizations, saved searches, and comparisons are available only to users with the FindBI role. For more information, refer to the *Find Installation Guide*.

Custom Front-End Applications

The ACI (Autonomy Content Infrastructure) Client API allows you to create your own custom front-end applications.

The ACI API enables easy communication between custom-built applications and IDOL ACI servers, as well as simple manipulation of the results sets. IDOL SDKs are available for several programming languages, including C, Java, and .NET.

For more information, refer to the *ACI API Programming Guide*.

Administrative Applications

There are several IDOL applications that allow you to manipulate and modify the data in IDOL, and to monitor the IDOL services and applications.

IDOL Admin

IDOL Admin allows you to administer the IDOL Content component. You can use the Web-based application interface to:

- monitor IDOL performance
- search the Content configuration file
- retrieve information about data in the index
- index documents
- modify data fields in index documents
- export indexed documents
- back up and restore the data index
- export diagnostics information
- control the IDOL service

For more information, refer to the *IDOL Admin User Guide*.

IDOL Site Admin

IDOL Site Admin is an administration component used to monitor and manage IDOL services. You can use IDOL Site Admin to start and stop services, and monitor individual services and the overall health of your IDOL services. You can also view all log files and configuration files.

For information on how to install and access IDOL Site Admin, refer to the *IDOL Site Admin Installation Guide*.

For information about how to use IDOL Site Admin to monitor your services, refer to the *IDOL Site Admin User Guide*.

IDOL Data Admin

NOTE: Before IDOL version 11.3, IDOL Data Admin was known as IDOL Search Optimizer.

IDOL Data Admin allows you to manage data indexed in IDOL servers to optimize the search process for your end users. You can create and modify promotions or keywords to predefine the results that are returned to users, and control how users view them. You can also view statistical information to help you to refine the rules that you created, and make them more effective.

You can also use IDOL Data Admin administer an Answer Bank system for Answer Server, to set up your question and answer store.

For information on how to install and access IDOL Data Admin, refer to the *IDOL Data Admin Administration Guide* .

For information about how to use IDOL Data Admin to manage your data, refer to the *IDOL Data Admin User Guide*.

Chapter 4: Security in IDOL

This chapter provides an overview of IDOL Document Security. For more information on security, refer to the *IDOL Document Security Administration Guide*.

Security Overview

Micro Focus provides the software infrastructure that automates operations on unstructured information. This software infrastructure is based on IDOL Server.

Document security protects the information that you index into IDOL Server.

Your organization is likely to store information in many repositories. Many of these repositories have security features that apply permissions to files, so that they only authorized personnel can view them. Document security ensures that when you index information into IDOL Server, IDOL continues to enforce these permissions. In response to a query, IDOL returns only documents that a user is permitted to view.

IDOL includes the following features to protect your data:

- **User authentication**. At the front end, users must log on before they can query IDOL. IDOL can authenticate users against an existing directory service, such as Microsoft Active Directory.
- **Document security**. IDOL checks each document that is returned in response to a query. If the user who submitted the query does not have permission to view the document, the document is removed from the query results before the results are returned.
- **Secure communications**. You can encrypt the communications between ACI servers and any applications that use the Micro Focus IDOL ACI API.

User Authentication

To access front-end applications, users must log on by providing their credentials. If you build your own front-end applications, you can use the IDOL API to customize the logon process.

IDOL supports authentication against industry standard systems, including:

- Windows NT logon.
- LDAP authentication.
- Other third-party authentication (for example, Lotus Notes).

A single logon allows a user to access all the systems for which they have permission. For example, if a user submits a query they receive all documents that they are permitted to view, even though these documents originated from different data repositories. The IDOL Server Community component enables you to store and update user security details for this purpose.

In addition, you can use IDOL in a single sign-on environment that uses Kerberos for authentication.

Document Security

Document security ensures that users can access only those documents for which they have the necessary permissions.

When a user logs on to a front-end application and is authenticated successfully, the IDOL Community component returns an encrypted security string to the front-end application. This string identifies the user and contains information about their group memberships. The front-end application must include this security string in every subsequent query it sends to IDOL.

When a user submits a query, the IDOL Content component compares the user's information with the permissions on each document. IDOL offers two ways of doing this, *Mapped Security* and *Unmapped Security*.

NOTE: Micro Focus strongly recommends using Mapped Security, because IDOL can return query responses significantly faster than when Unmapped Security is used.

Mapped Security

In the *Mapped Security* architecture, IDOL determines whether a user is permitted to view a document by comparing the user's security details against an Access Control List (ACL) that has been added to the document.

With Mapped Security, when connectors fetch information from data repositories they add an encrypted Access Control List (ACL) to a metadata field in each document. The ACL contains information about which users and groups are permitted to access the document. The documents, and therefore the ACLs, are indexed into IDOL Server.

A user might be allowed or denied permission to view a document because they are a member of a security group. This means that IDOL must consider group memberships, in addition to permissions, before it can determine whether a user can view a document. OmniGroupServer collects user and group information, and stores it, so that IDOL Server can access this information.

When a user queries IDOL Content, IDOL sends the user's security string and the documents that match the query to the Mapped Security Plug-In. The Mapped Security Plug-In compares the user's details to the ACL in each document, and determines which documents the user is permitted to view. IDOL only returns those documents in its response.

The advantage of this process is that IDOL can quickly respond to a query, because it does not need to connect to the original data repository to check the ACL for each document. The disadvantage is that there might be a delay between the security settings changing in the original data repository and the information being updated in the IDOL index.

Mapped Security is suitable for most environments, particularly where the security settings for documents do not change often.

Unmapped Security

In the *Unmapped Security* architecture, IDOL determines whether a user is permitted to view a document by connecting directly to data repositories and checking the user's security details against the entitlement information of the documents that matched the query.

The connection between IDOL and the data repositories is made possible by IDOL's unmapped security libraries.

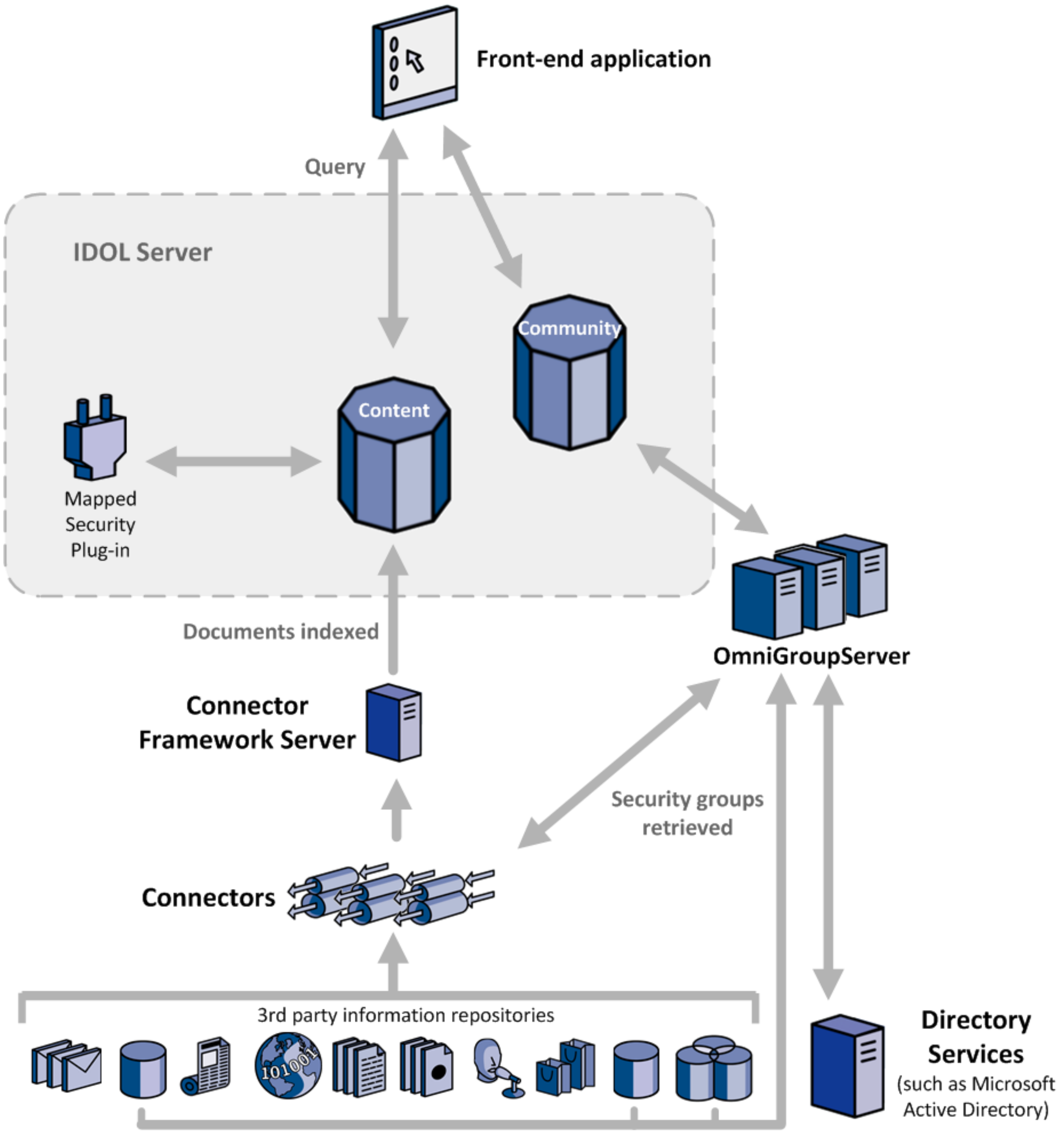
The advantage of unmapped security is that the security information is current. The disadvantage is that IDOL Server has to connect to the original data repositories to check permissions for each result document. This means that there can be a significant delay between a user submitting a query, and IDOL returning its response. For this reason, Micro Focus strongly recommends using Mapped Security.

Unmapped security is suitable for environments where the security settings for documents change frequently.

In some cases, you must also configure a group server. A group server is required because it is impossible for IDOL to retrieve group information from the repositories in a reasonable time. OmniGroupServer extracts user and group information and stores it so that it is available to IDOL immediately.

Mapped Security Example

The following diagram shows the components involved in a Mapped Security architecture:


Front-end application
Query
IDOL Server
Community
Content
Mapped Security Plug-in
Documents indexed
OmniGroupServer
Connector Framework Server
Security groups retrieved
Connectors
3rd party information repositories
Directory Services (such as Microsoft Active Directory)

Connectors extract information from third party repositories so that the information can be indexed into IDOL Server. The connector adds an Access Control List (ACL) to a metadata field in each document. The ACL describes which users and groups are permitted to view the document. CFS indexes the document into IDOL Server.

At the same time, OmniGroupServer retrieves group memberships from the third party repositories and from directories such as Active Directory. OmniGroupServer stores this information until it is needed. In some cases, where a repository uses its own system for storing users and groups, OmniGroupServer queries a connector to retrieve group information.

If the permissions set on a file in a repository are changed, the connector sends the update to CFS and the document's ACL is updated in IDOL Server. If a user's group memberships change, the group information is updated in OmniGroupServer the next time the group server synchronizes with the repository.

To use the front-end application, a user must log on. After authentication is successful, the front-end application sends a query to Community, to retrieve the user's security information. Community returns an encrypted `securityinfo` string that contains the names of the groups the user is a member of. The front-end stores the string, because it must be sent with all queries to IDOL server.

When a user does something in the front-end application that requires information from IDOL server (for example, starting a search), the front-end sends a query to IDOL server. IDOL Server (Content) runs the operation and sends the resulting documents and the user's security information to the Mapped Security plug-in. This compares the user's security information with the ACL of each document and returns the documents that the user is permitted to view. IDOL Server then returns these documents to the front-end.

Secure Communications

You can configure ACI servers to communicate using encryption and Secure Socket Layer (SSL) communications.

SSL Communications

You can configure Secure Socket Layer (SSL) communications for IDOL Server, as well as for other ACI servers, front-end applications, and connectors.

The exact configuration you use depends on the component you are configuring. For more information, refer to the topics about Secure Communications in *IDOL Expert*, and the relevant component Administration Guides.

Part 2: Install and Run IDOL Server

This section describes how to install, run, and configure IDOL Server.

Chapter 5: Install IDOL

This section describes how to install IDOL by using the IDOL Server installer, and the software and hardware requirements to run IDOL.

For information about how to install IDOL containers by using Docker images, see Install IDOL with Docker Images, on page 133.

System Requirements

This section describes the software and hardware requirements to run IDOL.

Basic Requirements

- To install IDOL services on Microsoft Windows operating systems, you must run the IDOL Server installer as a user with administrative privileges.

  NOTE: You do not require special privileges to run the IDOL Server installer on Linux operating systems, or to run IDOL Server on any platform.

- You cannot run IDOL with restricted file system permissions (for example: disk quotas, file handle limits or memory limits).
- Your file system must permit file locking.
- Your network must support TCP/IP.
- If you run anti-virus software, back up software, or any other software that scans the file system on the machine that hosts IDOL server, ensure it does not monitor the IDOL Server directories, which can have a serious impact on IDOL Server performance. Micro Focus recommends that you use the component back up actions to back up your IDOL data.

  In addition some advanced anti-virus software can scan the network and might block some IDOL traffic, which can cause errors. Where possible, exempt the IDOL processes from this kind of network traffic analysis.

Supported Platforms

IDOL runs on a variety of Windows and UNIX platforms. For details of supported platforms, refer to the *IDOL 12.13 Release Notes*.

Recommended Minimum Hardware Specifications

Micro Focus recommends the following minimum hardware specifications for IDOL Server.

- a dedicated SCSI disk
- 4 GB RAM
- 100 GB Disk
- a minimum of 2 dedicated CPU - Intel Xeon or AMD Opteron or above

Windows System Recommendations

This section describes recommendations and requirements for IDOL for Microsoft Windows operating system platforms.

TCP Port Requirements

On Microsoft Windows platforms, if your IDOL service is receiving a large number of actions, or you have a large number of services on the same machine, your machine might run out of available TCP ports.

If your system does not have enough ports, you can edit the following Windows registry parameters:

- `TCPTimedWaitDelay`
- `MaxUserPort`

These parameters are located in:

```
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters\
```

For more information about these registry parameters, refer to the Microsoft Windows documentation.

Linux System Recommendations

This section describes some Linux operating system settings that you might want to modify to improve IDOL performance and function. You might need additional system privileges to make these changes. In all cases, Micro Focus recommends that you test on your own systems to find the optimal settings for your setup.

NOTE: For many of the settings listed in this section, values and example commands are given for Red Hat Enterprise Linux (RHEL). You might need to check the default values and suggested commands for other Linux distributions.

Open File Descriptors

The IDOL Content component can use a large number of files for certain tasks, such as indexing, particularly for large indexes. On Linux systems, the number of file descriptors is limited to prevent them from using too much memory. The memory usage of a single file descriptor is low, so it is usually safe to set the limit to a high value. Micro Focus recommends that you increase the limit value to at least **`65536`** for the user that the IDOL Content component runs as.

You can find an estimate of the number of files that a process (with a specified process ID) is using by using the following command:

```
ls -l /proc/ProcessID/fd | wc -l
```

You can find the current limit for the number of files by using the following commands:

```
Ulimit –Sn
Ulimit -Hn
```

There are two limits to the number of files that processes can use:

- The soft limit is the default limit at the start of a user session, and you can use `ulimit` commands to increase it (up to the hard limit).
- The hard limit is the absolute limit for your system.

To modify the individual process file limit for the user that the IDOL Content component runs as (*`UserName`*), open the `/etc/security/limits.conf` configuration file, and add or modify the following lines to modify the soft and hard file limits (where *`NewValue`* is the limit that you want to set):

```
UserName hard nofile NewValue
UserName soft nofile NewValue
UserName hard nproc NewValue
UserName soft nproc NewValue
```

TIP: On Ubuntu, when you use the start-stop-daemon, you might need to run the `ulimit` command in the init script to update the hard and soft limits when you start the service.

If you start IDOL component executable files directly (rather than using the init script) by using SSH and `su`, you might also need to modify the `/etc/pam.d/su` file to uncomment the following line:

```
# session    required   pam_limits.so
```

File Handle Limits in Services

On systemd init systems, services do not inherit file handle limits from the system limits or user settings. In some cases, this behavior might mean that a component fails to operate when running as a service, because it runs out of file handles.

You can modify the file handle limits for the service by adding the `LimitNOFILE` parameter to the service file, or by creating an `override.conf` file for the service.

For more information, see Install an IDOL Component as a Service on Linux, on page 72.

Transparent Huge Pages

Transparent Huge pages (THP) is the use of an abstraction layer that automates the creation, management and use of Huge Pages for memory management. In RHEL 6 and later, this feature is turned on by default, but Red Hat does not recommend the use of THP for database workloads. For IDOL, turning this setting off can improve the performance for many query operations.

You can change this setting in a terminal by using the following commands:

```
echo never > /sys/kernel/mm/redhat_transparent_hugepage/enabled
echo never > /sys/kernel/mm/redhat_transparent_hugepage/defrag
```

> **TIP:** On Ubuntu, you can use the following commands:
>
> ```
> echo never > /sys/kernel/mm/transparent_hugepage/enabled
> echo never > /sys/kernel/mm/transparent_hugepage/defrag
> ```

You can also turn this setting off at boot time, by adding the following text to the bottom of the `/etc/rc.local` file:

```
#disable THP at boot time
if test -f /sys/kernel/mm/redhat_transparent_hugepage/enabled; then
echo never > /sys/kernel/mm/redhat_transparent_hugepage/enabled
fi
if test -f /sys/kernel/mm/redhat_transparent_hugepage/defrag; then
echo never > /sys/kernel/mm/redhat_transparent_hugepage/defrag
fi
```

> **TIP:** On Ubuntu, the `/etc/rc.local` ends with the line `exit 0`, and you must make any changes to the file above that line.

Tune Virtual Memory

Micro Focus recommends tuning the virtual memory usage for Linux systems for database workloads to improve IDOL performance. In particular, you might want to modify the following settings:

- `vm.swappiness`. This setting controls how much the system favors swapping out runtime memory. The default value on Linux systems is `60`, but for database workloads, the recommended value is `10`. Micro Focus recommends that you set this value to `10` or lower. A value of `0` or `1` configures the system to only use the minimum amount of swapping, but does not turn it off.

- `vm.dirty_ratio`. This setting controls the maximum percentage of memory that can contain pages that have not yet been written to disk. The default value is `20`, but for database workloads the recommended value is `15`.

- `vm.dirty_background_ratio`. This setting controls the percentage of memory that can contain pages that have not yet been written to disk before the system starts to write the data in the background. The default is `10`, but for database workloads the recommended value is `3`.

You can check the current values for these settings by running the following command in the terminal:

```
# sysctl –a | grep "vm.Parameter"
```

To change these settings permanently, update the `/etc/sysctl.conf` configuration file, to add or modify the following lines:

```
vm.swappiness = 10
vm.dirty_ratio = 15
vm.dirty_background_ratio = 3
```

Memory Map Counts

The IDOL Content component can use a large number of memory map areas for certain tasks, such as indexing, particularly for large indexes. On Linux systems, the number of memory mapped areas that a process can use is limited, to a value controlled by the `max_map_count` kernel parameter. To prevent IDOL from reaching this limit, Micro Focus recommends that you increase the value of the `max_map_count` parameter to the amount of memory on the system in KB /16, but no less than 65536.

NOTE: This recommendation applies for a single IDOL Server (or Content component) running on the host machine. If you have multiple Content components running on the same host, you might need to limit this value further.

You can find an estimate of the number of memory mapped areas that a process (with a specified process ID) is using by using the following command:

```
cat /proc/processID/maps | wc -l
```

You can find the current value of `max_map_count` by using the following command:

```
sysctl vm.max_map_count
```

You can change the value of `max_map_count` by using the following command:

```
sysctl vm.max_map_count=NewValue
```

Configure the I/O Scheduler

The Linux kernel can use several different I/O schedulers to prioritize disk input and output. By default, many Linux distributions use the Completely Fair Queuing (CFQ) scheme for I/O scheduling, which gives input and output requests equal priority. For IDOL systems, Micro Focus recommends that you set I/O scheduling to either `deadline` or `noop`:

- The Deadline scheduler gives priority to read requests over write requests. It also imposes a deadline on all requests. After reaching the deadline, these requests gain priority over all other requests. This scheduling method helps prevent processes from becoming starved for I/O access. The Deadline scheduler is recommended for physical media drives (HDD), because it attempts to group requests for adjacent sectors on disk, reducing the amount of time that the disk spends seeking.

- The NOOP scheduler uses a simple first-in first-out approach, placing all input and output requests into a single queue. This scheduler is recommended for solid state drives (SSDs). Because SSDs do not have a physical read head, there is no performance penalty for accessing non-adjacent sectors.

Micro Focus recommends that you always use one of these schedulers, rather than the CFQ scheduler. However, the exact method to use depends on your environment and usage, so you might need to test both of these methods on your system to see which provides the best performance.

You can check whether your disks are SSD or HDD by using the following command:

```
for i in /sys/block/sd*; do cat $i/queue/rotational; done
```

This command returns `1` for HDD, and `0` for SSD.

You can find the scheduler for your mounted devices by using the following command:

```
for i in /sys/block/sd*; do cat $i/queue/scheduler; done
```

NOTE: On RHEL 6 and above, for multipath devices, you must also change the `dm-*` block device.

This command returns a result of the following type, with the current option surrounded with square brackets:

```
noop deadline [cfq]
```

You can change the scheduler by using a command of the following type:

```
for i in /sys/block/sd*; do echo "noop" >  $i/queue/scheduler; done
```

This example changes the system to use the NOOP scheduler.

Install IDOL

The IDOL installation is a simple, single-pass installer that installs IDOL Server and some other common components. This section describes how to use the installer to install IDOL server.

Install IDOL on Windows

Use the following procedure to install IDOL 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 IDOL requires to run.

To install IDOL

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 IDOL (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 IDOL, 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 each component, you can optionally provide port information to specify the ports that the component must use. The port that you choose must not be used by any other service. The installer allows you to set the following port types:

ACI Port	The port that client machines use to send ACI actions to the component.
Index Port	The port that client machines use to send index actions to the component. This port is available only for the Content, Agentstore, and DIH components.
Service Port	The port that client machines use to send service requests to the component. This port is also the port that the component uses for License Server communication.

   You can also optionally choose to install the components as a Windows service. In this case, select **Create Windows service**. You can specify a name to use for the service, or accept the default name.

   If you do not want to install the component as a service at installation time, you can choose to add the services later. See Install an IDOL Component as a Service on Windows, on page 67.

   The following table lists the components and subcomponents that you can install, and the default port that the installer uses if you do not provide an alternative value. You must change the port information if the default port value is already in use.

Component	Default port information
Content	ACI Port: 9100 Index Port: 9101 Service Port: 9102
Agentstore	ACI Port: 9050

	Index Port: 9051 Service Port: 9052
Category	ACI Port: 9020 Service Port: 9022
Community	ACI Port: 9030 Service Port: 9032
View	ACI Port: 9080 Service Port: 9082
QMS	ACI Port: 16000 Service Port: 16002
Statistics Server	ACI Port: 19870 Service Port: 19872
Controller	ACI Port: 41200 Service Port: 41202
Coordinator	ACI Port: 40200 Service Port: 40202
DAH	ACI Port: 9060 Service Port: 9062
DIH	ACI Port: 9070 Index Port: 9071 Service Port: 9072
CFS	ACI Port: 7000 Service Port: 17000
File System Connector	ACI Port: 7002 Service Port: 17002
Web Connector	ACI Port: 7006 Service Port: 17006
Media Server	ACI Port: 14000 Service Port: 14001
Knowledge Graph	ACI Port: 10300 Service Port: 10302
Find	HTTP Port: 8080
IDOL Site Admin	HTTP Port: 8080

For more information about these components, see IDOL Core Components, on page 27.

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 a Unified IDOL Server Setup on Windows

The unified IDOL Server setup is the most basic IDOL setup, for training and test environments. For more information, see Unified Setup, on page 30.

The following procedure describes how to install the unified IDOL Server setup on the Microsoft Windows operating system platform.

To install unified IDOL Server

1. Double-click the appropriate installer package:

   `IDOLServer_`*`VersionNumber`*`_`*`Platform`*`.exe`

   where:

`VersionNumber`	is the version of the installer.
`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 IDOL and License Server in. By default, the system installs on `C:\MicroFocus\IDOLServer-`*`VersionNumber`*. Click to navigate to another location. Click **Next**.

   The Installation Mode dialog box opens.

5. Select **Unified**, and then click **Next**.

   The Service Name dialog box opens.

6. Type the name to use for the IDOL Server Windows service and then click **Next**. By default, the system installs the service `MicroFocus-IDOLServer`.

   The License Server Service Name dialog box opens.

7. Type the name to use for the License Server Windows service and then click **Next**. By default, the system installs the service `MicroFocus-LicenseServer`.

   The License Server dialog box opens.

8. Specify the ports that you want License Server to listen on, and then type the path or click and navigate to the location of your IDOL license key file (`licensekey.dat`), which you obtained when you purchased IDOL. Click **Next**.

9. The Summary dialog box opens. Verify the settings you made and click **Next**.

   The Ready to Install dialog box opens.

10. Click **Next**.

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

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

Windows Services

When you install a unified IDOL Server, the installer automatically installs Windows Services for the License Server and the IDOL Proxy component (which automatically starts up other IDOL components when it starts). You can use these Windows Services to easily start and stop your IDOL Server while you test configurations.

You can also manually install components as Windows Services. See Install an IDOL Component as a Service on Windows, below.

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 IDOL as a Windows service from a command line.

To install a component as a Windows service

1. Open a command prompt with administrative privileges (right-click the icon and select **Run as administrator**).

2. Navigate to the directory that contains the component that you want to install as a service.

3. Send the following command:

   ```
   Component.exe -install
   ```

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

   The `-install` command has the following optional arguments:

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

For example:

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

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

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

To uninstall an IDOL Windows Service

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

   ```
   Component.exe -uninstall
   ```

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

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

   ```
   Component.exe -uninstall -servicename ServiceName
   ```

Install IDOL on UNIX

Use the following procedure to install IDOL in text mode on UNIX platforms.

To install IDOL on UNIX

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

   ```
   ./IDOLServer_VersionNumber_Platform.exe --mode text
   ```

   where:

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

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

   The console installer starts and displays the Welcome screen.

2. Read the information and then press the `Enter` key.

   The license information is displayed.

3. Read the license information, pressing `Enter` to continue through the text. After you finish reading the text, type **`Y`** to accept the license terms.

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

   The Installation Mode screen is displayed.

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

6. Choose whether to install IDOL for OEM usage.

   - To install IDOL for OEM usage

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

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

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

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

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

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

7. Choose whether you have an existing License Server.

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

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

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

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

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

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

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

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

   For each component, you can optionally provide port information to specify the ports that the component must use. The port that you choose must not be used by any other service. The installer allows you to set the following port types:

ACI Port	The port that client machines use to send ACI actions to the component.
Index Port	The port that client machines use to send index actions to the component. This port is available only for the Content, Agentstore, and DIH components.
Service Port	The port that client machines use to send service requests to the component. This port is also the port that the component uses for License Server communication.

   The following table lists the components and subcomponents that you can install, and the default port that the installer uses if you do not provide an alternative value. You must change the port information if the default port value is already in use.

Component	Default port information
Content	ACI Port: 9100 Index Port: 9101 Service Port: 9102

Agentstore	ACI Port: 9050 Index Port: 9051 Service Port: 9052
Category	ACI Port: 9020 Service Port: 9022
Community	ACI Port: 9030 Service Port: 9032
View	ACI Port: 9080 Service Port: 9082
QMS	ACI Port: 16000 Service Port: 16002
Statistics Server	ACI Port: 19870 Service Port: 19872
Controller	ACI Port: 41200 Service Port: 41202
Coordinator	ACI Port: 40200 Service Port: 40202
DAH	ACI Port: 9060 Service Port: 9062
DIH	ACI Port: 9070 Index Port: 9071 Service Port: 9072
CFS	ACI Port: 7000 Service Port: 17000
File System Connector	ACI Port: 7002 Service Port: 17002
Web Connector	ACI Port: 7006 Service Port: 17006
Media Server	ACI Port: 14000 Service Port: 14001
Knowledge Graph	ACI Port: 10300 Service Port: 10302
Find	HTTP Port: 8080
IDOL Site Admin	HTTP Port: 8080

For more information about these components, see IDOL Core Components, on page 27.

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

The Init Scripts screen is displayed.

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

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

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

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

    The Summary screen is displayed.

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

    The Installing screen is displayed.

    This screen indicates the progress of the installation process.

    The Installation Complete screen is displayed.

13. Press `Enter` to finish the installation.

Install an IDOL Component as a Service on Linux

On Linux operating systems, you can 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 IDOL as a service on Linux.

> **IMPORTANT:** These procedures assume that you install IDOL 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 IDOL 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, on the next page.
- For Linux operating system versions that use System V, see System V, on page 74.

systemd

NOTE: If your setup has an externally mounted drive that IDOL 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.

Use Unattended Installation

For many IDOL setups, you might want to install multiple sets of the same components with the same settings, on different machines. In this case, you can use the *unattended* installation mode.

> **NOTE:** Unattended installation is a feature of the BitRock InstallBuilder installers. For more information, refer to the InstallBuilder documentation.
>
> You can use this procedure to install any IDOL component that uses an InstallBuilder installer (for example, connectors).

For an unattended installation mode, you provide all the installation options at the command line, and the installer runs without any further user interaction. You can also supply the installation options in an options file.

To retrieve a full list of installation options for the installer

1. Open a command prompt.
2. Navigate to the directory that contains the installer file.
3. Run the following command:

   ```
   InstallerName.exe --help
   ```

   where `InstallerName` is the full name of the installer file, for example `IDOLServer_11.0.0_WINDOWS_X86_64`.

This option displays the help page for the installer, which includes all the options that you can set on the command line.

To run the installer in unattended mode

1. Open a command prompt.
2. Navigate to the directory that contains the installer file.
3. Run the installer with the `--mode` command-line option set to **`unattended`**. For example:

   ```
   InstallerName.exe --mode unattended
   ```

   where `InstallerName` is the full name of the installer file, for example `IDOLServer_11.0.0_WINDOWS_X86_64`.

   The default option installs a unified IDOL Server. To install a custom set of components, you can specify additional installation parameters on the command line, or you can set the `--optionfile FileName` option to specify a file that contains the installation options. For example:

   ```
   InstallerName.exe --mode unattended --optionfile IDOLinstall.options
   ```

   The option file must contain a list of `key=value` pairs, with one parameter on each line. For example:

   ```
   param1=value1
   param2=value2
   ```

For more information, refer to the BitRock InstallBuilder documentation.

Install Separate IDOL Components

For a component setup, where you install and run each IDOL component separately, you can download and install individual component ZIP packages, if you do not want to use the IDOL Server installer.

Each component ZIP package provides a configuration file, and the required modules for the component. It also includes a component-specific Reference, which contains the actions and configuration parameters relevant to that component.

To install the component, you extract the ZIP package to the host where you want to run the component. You must also update the `[License]` section of the configuration file to include appropriate license information. If you have more than one version of a component running on the same host, you might also need to update the port information to ensure that each component uses a different port.

Component packages are available for the core IDOL Server components, as well as the FileSystem, HTTP, and Web Connectors.

On Microsoft Windows operating systems, you can optionally also add components as a Windows service. See Install an IDOL Component as a Service on Windows, on page 67.

IDOL Component Dependencies

When you install a unified IDOL Server, the installer automatically configures the components with associated dependencies. When you use a custom installation, or install components using ZIP packages, you must ensure that you configure these dependencies in the component configuration files.

For example, you must configure:

- the Category component with the details of the data index (Content component), and the category index (Agentstore component).
- the Community component with the details of the agent index (Agentstore component).
- the View component with the details of a Content component that it can use for highlighting.
- the QMS component with the details of the promotion Agentstore component.

  NOTE: If you install Agentstore with the IDOL Server installer, there are two available configuration files, `agentstore.cfg` and `agentstore_qms.cfg`. The default `agentstore.cfg` file is for use with Category and Community. To use Agentstore with QMS, rename `agentstore_qms.cfg` to `agentstore.cfg` (and rename or remove the default file), or run Agentstore from the command line to choose the configuration file. See Start IDOL from the Command Line, on page 84.

- the Connector and CFS with the details of the component to send data to.
- the DIH and DAH components with details of the components to distribute to.

This list is not exhaustive. For more information about component dependencies, refer to the component documentation.

Install IDOL Admin

IDOL Admin is an administration interface for performing IDOL administration tasks, such as gathering status information, retrieving information about data in the index, and sending index actions to administer the index.

The IDOL Server installer includes the `admin.dat` file when you install any IDOL component. For each component that you install, the installer sets:

- the `AdminFile` configuration parameter in the `[Paths]` section of the configuration file to the path to the installed `admin.dat` file.
- the `Access-Control-Allow-Origin` configuration parameter in the `[Service]` section of the configuration file to allow the local host and the component port to access the IDOL Admin interface.

You might need to modify the `Access-Control-Allow-Origin` configuration parameter to allow access to IDOL Admin from other hosts. For more information, refer to the Reference for your component.

Supported Browsers

IDOL Admin supports the following browsers:

- Edge
- Chrome (latest version)
- Firefox (latest version)

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.

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.

Licenses

To use IDOL solutions, you must have a running License Server, and a valid license key file for the products that you want to use. Contact Micro Focus Big Data Support to request a license file for your installation.

License Server controls the IDOL licenses, and assigns them to running components. License Server must run on a machine with a static, known IP address, MAC address, or host name. The license key file is tied to the IP address and ACI port of your License Server and cannot be transferred between machines. For more information about installing License Server and managing licenses, see the *License Server Administration Guide*.

When you start IDOL, it requests a license from the configured License Server. You must configure the host and port of your License Server in the IDOL configuration file.

You can revoke the license from a product at any time, for example, if you want to change the client IP address or reallocate the license.

CAUTION: Taking any of the following actions causes the licensed module to become inoperable.

You **must not**:

- Change the IP address of the machine on which a licensed module runs (if you use an IP address to lock your license).
- Change the service port of a module without first revoking the license.
- Replace the network card of a client without first revoking the license.
- Remove the contents of the `license` and `uid` directories.

All modules produce a `license.log` and a `service.log` file. If a product fails to start, check the contents of these files for common license errors. See Troubleshoot License Errors, on page 80.

Display License Information

You can verify which modules you have licensed either by using the IDOL Admin interface, or by sending the `LicenseInfo` action from a web browser.

To display license information in IDOL Admin

- In the **Control** menu of the IDOL Admin interface for your License Server, click **Licenses**.

  The **Summary** tab displays summary information for each licensed component, including:

  - The component name.
  - The number of seats that the component is using.
  - The total number of available seats for the component.

- (Content component only) The number of documents that are currently used across all instances of the component.
- (Content component only) The maximum number of documents that you can have across all instances of the component.

The **Seats** tab displays details of individual licensed seats, and allows you to revoke licenses.

To display license information by sending the LicenseInfo action

- Send the following action from a web browser to the running License Server.

```
http://LicenseServerHost:Port/action=LicenseInfo
```

where:

LicenseServerHost	is the IP address of the machine where License Server resides.
Port	is the ACI port of License Server (specified by the `Port` parameter in the `[Server]` section of the License Server configuration file).

In response, License Server returns the requested license information. This example describes a license to run four instances of IDOL Server.

```
<?xml version="1.0" encoding="UTF-8" ?>
<autnresponse xmlns:autn="http://schemas.autonomy.com/aci/">
  <action>LICENSEINFO</action>
  <response>SUCCESS</response>
  <responsedata>
    <LicenseDiSH>
      <LICENSEINFO>
        <autn:Product>
          <autn:ProductType>IDOLSERVER</autn:ProductType>
          <autn:TotalSeats>4</autn:TotalSeats>
          <autn:SeatsInUse>0</autn:SeatsInUse>
        </autn:Product>
      </LICENSEINFO>
    </LicenseDiSH>
  </responsedata>
</autnresponse>
```

Revoke a Client License

After you set up licensing, you can revoke licenses at any time, for example, if you want to change the client configuration or reallocate the license. The following procedure revokes the license from a component.

NOTE: If you cannot contact the client (for example, because the machine is inaccessible), you can free the license for reallocation by sending an `AdminRevokeClient` action to the License Server. For more information, see the *License Server Administration Guide*.

To revoke a license

1. Stop the IDOL solution that uses the license.
2. At the command prompt, run the following command:

   ```
   InstallDir/ExecutableName[.exe] -revokelicense -configfile cfgFilename
   ```

This command returns the license to the License Server.

You can send the `LicenseInfo` action from a web browser to the running License Server to check for free licenses. In this sample output from the action, one IDOL Server license is available for allocation to a client.

```
<autn:Product>
   <autn:ProductType>IDOLSERVER</autn:ProductType>
   <autn:Client>
      <autn:IP>192.123.51.23</autn:IP>
      <autn:ServicePort>1823</autn:ServicePort>
      <autn:IssueDate>1063192283</autn:IssueDate>
      <autn:IssueDateText>10/09/2003 12:11:23</autn:IssueDateText>
   </autn:Client>
      <autn:TotalSeats>2</autn:TotalSeats>
      <autn:SeatsInUse>1</autn:SeatsInUse>
</autn:Product>
```

Troubleshoot License Errors

The table contains explanations for typical licensing-related error messages.

License-related error messages

Error message	Explanation
Error: Failed to update license from the license server. Your license cache details do not match the current service configuration. Shutting the service down.	The configuration of the service has been altered. Verify that the service port and IP address have not changed since the service started.
Error: License for *ProductName* is invalid. Exiting.	The license returned from the License Server is invalid. Ensure that the license has not expired.
Error: Failed to connect to license server using cached licensed details.	Cannot communicate with the License Server. The product still runs for a limited period; however, you should verify whether your License Server is still available.
Error: Failed to connect to license server. Error code is SERVICE: *ErrorCode*	Failed to retrieve a license from the License Server or from the backup cache. Ensure that your License Server can be contacted.

License-related error messages, continued

Error message	Explanation
Error: Failed to decrypt license keys. Please contact Autonomy support. Error code is SERVICE:*ErrorCode*	Provide Micro Focus Big Data Support with the exact error message and your license file.
Error: Failed to update the license from the license server. Shutting down	Failed to retrieve a license from the License Server or from the backup cache. Ensure that your License Server can be contacted.
Error: Your license keys are invalid. Please contact Autonomy support. Error code is SERVICE:*ErrorCode*	Your license keys appear to be out of sync. Provide Micro Focus Big Data Support with the exact error message and your license file.
Failed to revoke license: No license to revoke from server.	The License Server cannot find a license to revoke.
Failed to revoke license from server *LicenseServer Host*:*LicenseServerPort*. Error code is *ErrorCode*	Failed to revoke a license from the License Server. Provide Micro Focus Big Data Support with the exact error message.
Failed to revoke license from server. An instance of this application is already running. Please stop the other instance first.	You cannot revoke a license from a running service. Stop the service and try again.
Failed to revoke license. Error code is SERVICE:*ErrorCode*	Failed to revoke a license from the License Server. Provide Micro Focus Big Data Support with the exact error message.
Your license keys are invalid. Please contact Autonomy Support. Error code is ACISERVER:*ErrorCode*	Failed to retrieve a license from the License Server. Provide Micro Focus Big Data Support with the exact error message and your license file.
Your product ID does not match the generated ID.	Your installation appears to be out of sync. Forcibly revoke the license from the License Server and rename the `license` and `uid` directories.
Your product ID does not match this configuration.	The service port for the module or the IP address for the machine appears to have changed. Check your configuration file.

Find the Number of Licensed Documents

You can send the `LicenseInfo` action to your License Server to find out how many documents your IDOL installation is licensed for.

The `LicenseInfo` response includes a list of the components that you have a license for. For the Content component license information, the following tags show your licensed number of documents:

Tag	Description
`<autn:AggregateLimit name="maxdocuments">`	The maximum number of documents that you can have across all instances of the Content component that connect to the License Server.
`<autn:AggregateUsage name="maxdocuments">`	The number of documents that are currently used across all instances of the Content component that connect to the License Server.

In addition, the `<autn:Product>` section of the response has an `<autn:Client>` subsection that provides details of the services that are using the license. In this section, the `<autn:TrackedLimit name="maxdocuments">` shows how many documents the particular instance of the content component is using.

For example:

```
<autn:Product>
       <autn:ProductType>SUIR</autn:ProductType>
       <autn:ComponentName>content</autn:ComponentName>
       <autn:Client>
               <autn:IP>12.34.56.78</autn:IP>
               <autn:MAC>01.23.45.67.89.AB</autn:MAC>
               <autn:HostName>MyHost.example.com</autn:HostName>
               <autn:ServicePort>9993</autn:ServicePort>
               <autn:IssueDate>1379588409</autn:IssueDate>
               <autn:ProductUID>1706984408</autn:ProductUID>
               <autn:TrackedLimit name="maxdocuments">10001?
               </autn:TrackedLimit>
               <autn:IssueDateText>19/09/2013 11:00:09</autn:IssueDateText>
       </autn:Client>
       <autn:TotalSeats>100</autn:TotalSeats>
       <autn:SeatsInUse>1</autn:SeatsInUse>
       <autn:AggregateLimit name="maxdocuments">50000000</autn:AggregateLimit>
       <autn:AggregateUsage name="maxdocuments">10001</autn:AggregateUsage>
</autn:Product>
```

Chapter 6: Run IDOL Server

When you have installed IDOL server, you are ready to run it. This section describes the basics of using IDOL and verifying that it is running correctly.

Start and Stop IDOL

This section describes how to start and stop your IDOL service.

Start IDOL

The following sections describe the different ways that you can start IDOL and IDOL components.

Before you can start any IDOL component, you must start the License Server. When the License Server is running, you can start your other IDOL components.

> **TIP:** If you install a unified IDOL Server, you start IDOL Server by starting the IDOL Proxy component, which starts up the other IDOL components. The License Server must be running when you start the IDOL Proxy component.
>
> In a unified installation on Microsoft Windows platforms, License Server and IDOL Server are automatically installed as services.

Start an IDOL Component on Microsoft Windows

- Double-click the `*componentname*.exe` file in your component installation directory.
- Start the component Service from a system dialog box. The component must be installed as a Windows Service. See Install an IDOL Component as a Service on Windows, on page 67.
  1. Display the Windows **Services** dialog box.
  2. Select the ***ServiceName*** 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. See Start IDOL from the Command Line, below.

Start an IDOL Component 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 72 You can use one of the following commands to start the service:
  - On systemd Linux platforms:

    ```
    systemctl start ServiceName
    ```

  - On System V Linux platforms:

    ```
    service ServiceName start
    ```

  - On Solaris platforms (using System V):

    ```
    /etc/init.d/ServiceName start
    ```

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

- Start the IDOL component from the command line. See Start IDOL from the Command Line, below.
- (Advanced) Use the start script (`start.sh`).

  A template start script is included with the component installation. However, you must update the template script with details for the component that you want to start. In most cases, Micro Focus recommends that you use the provided init scripts instead.

Start IDOL from the Command Line

You can start IDOL components from the command line. This advanced functionality allows you to adjust certain parts of the component configuration when you start up. You might want to use this method to install components as Windows services, or to start a component with a configuration file that is in a different directory or has a different name.

NOTE: On UNIX platforms, if you start components from the command line (rather than using the init script), you might need to set the `LD_LIBRARY_PATH` to include the `*InstallDir*/common` and `*InstallDir*/common/runtimes` directories, to ensure that the component can access the installed shared libraries.

You can also copy the shared libraries to the component working directory.

This section describes the command line arguments available to specify the component configuration files and ports, and arguments that are available for specific components to specify dependencies.

All command line arguments are optional. By default, the component attempts to read a configuration file in the same directory, and with the same name as the executable file (for example `content.exe` and `content.cfg`). The components attempt to read all the other values from the configuration file.

The following command line arguments are accepted by all the IDOL Server text processing components (Content, Category, Community, DAH, DIH, IDOL Proxy, and View).

Argument	Description	Type
`-aciport`	The port to use to receive ACI actions.	Number
`-configfile`	The path to the configuration file to use.	FilePath
`-idolcomponent`	Specifies that the component runs under the control of an IDOL Proxy component.	None
`-install`	Installs the component as a Windows service. See Install an IDOL Component as a Service on Windows, on page 67.	None
`-revokelicense`	Revokes the component license from the License Server.	None
`-serviceport`	The port to use to receive Service actions.	Number
`-uninstall`	Uninstalls the component Windows service. See Install an IDOL Component as a Service on Windows, on page 67.	None
`-version`	Displays the version and build information for the component.	None

The following command line arguments are accepted by the specified components.

Argument	Description	Type	Components
`-agentstoreport`	The ACI port to use to contact the Agent index (IDOL Agentstore component).	Number	Category, Community
`-communityhost`	The host name or IP address of the IDOL Community component.	Hostname	Category
`-communityport`	The ACI port to use to contact the IDOL Community component.	Number	Category
`-contentport`	The ACI port to use to contact the IDOL Content component.	Number	View
`-datahost`	The host name or IP address of the Data index (IDOL Content componentt).	Hostname	Category, Community
`-dataport`	The ACI port to use to contact the Data index (IDOL Content component).	Number	Category, Community
`-indexport`	The port to use to receive Index actions.	Number	Content, DIH, IDOL Proxy

The following example starts the IDOL Content component as a child component of an IDOL Proxy component. It specifies the ports to use, and the host and port for the Agent and Data indexes:

```
Community.exe -idolcomponent -configfile C:\IDOL\IDOLServer\IDOLServer.cfg -aciport
9030 -serviceport 9032 -datahost localhost -dataport 9010  -agentstoreport 9050
```

Stop IDOL

You can stop an IDOL component 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 IDOL component is running, and *`servicePort`* is the component service port (which is specified in the `[Service]`section of the IDOL configuration file).

  To stop a unified IDOL Server, you can send the Stop service action to the IDOL Proxy component, and it automatically stops the other components.

  > **NOTE:** It can take a short while for the Content process to stop after you have sent the stop action to IDOL.

- 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 **IDOLServer** service, and click **Stop** to stop IDOL.
  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 ServiceName
    ```
  - On system V platforms:

    ```
    service ServiceName stop
    ```
  - On Solaris platforms (using System V):

    ```
    /etc/init.d/ServiceName stop
    ```

Send Actions to IDOL

IDOL 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 IDOL is installed.

port	is the IDOL ACI port. The ACI port is specified by the `Port` parameter in the `[Server]` section of the IDOL configuration file. For more information about the `Port` parameter, see the *IDOL 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 *IDOL Server Reference*.

Verify IDOL Runs Correctly

When you have installed IDOL and are using it, you can run actions to verify that IDOL is running correctly.

GetRequestLog

Send a `GRL` (or `GetRequestLog`) action to any IDOL component to return a log of the requests that have been made to it, including:

- the date and time that a request was made.
- the client IP address that made the request.
- the internal thread that handled the action.

For example:

```
http://IDOLhost:port/action=GRL
```

For further details on the `GRL` action, refer to the *IDOL Server Reference*.

Related Topics

- Send Actions to IDOL, on the previous page

GetLicenseInfo

You can send a `GetLicenseInfo` action to IDOL to return information on your license. This action allows you to check whether your license is valid, which IDOL operations your license includes, and which actions you can run.

For example:

```
http://IDOLhost:port/action=GetLicenseInfo
```

The following result indicates that your license is valid:

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

The following result indicates that your license includes the IDOL Agent operation:

```
-<autn:section>
       <autn:name>Agent</autn:name>
       <autn:licensed>true</autn:licensed>
</autn:section>
```

The following result indicates that you are permitted to run `Query` actions:

```
-<autn:section>
       <autn:name>Query</autn:name>
       <autn:licensed>true</autn:licensed>
</autn:section>
```

GetStatus

Use the `GetStatus` action to check whether the IDOL service is running.

For example:

```
http://IDOLhost:port/action=GetStatus
```

GetVersion

Use the `GetVersion` action to check the version number of an IDOL service.

For example:

```
http://IDOLhost:port/action=GetVersion
```

Display Online Help

You can display *IDOL Expert*, the *IDOL Server Reference*, and the *IDOL Server Administration Guide* by sending an action from your Web browser.

For IDOL to display help, the help data file (`help.dat`) must be available in the same directory as the service instance.

To display help for IDOL

1. Start IDOL server.
2. Send the following action from your Web browser:

   ```
   http://IDOLhost:port/action=Help
   ```

where,

IDOLhost	is the IP address or name of the machine on which IDOL is installed.
port	is the ACI port by which you send actions to IDOL (set by the `Port` parameter in the IDOL configuration file `[Server]` section).

For example:

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

3. On the help landing page, click one of the following options to open the relevant help:

Admin Guide	The *IDOL Server Administration Guide* describes how to configure and use IDOL Server.
Reference	The *IDOL Server Reference* contains details of all the actions and configuration parameters that you can set in IDOL Server.
IDOL Expert	*IDOL Expert* provides conceptual overviews and expert knowledge of IDOL server and its features and functionality.

If you are new to IDOL, you can use IDOL Expert to find out more about the different functions that IDOL can perform. Use the *IDOL Server Administration Guide* to get specific information about configuration and administration procedures. Use the *IDOL Server Reference* to look up specific configuration parameters and actions.

The navigation panel for the IDOL Server Reference lists the following options to display reference information:

Tab	Description
Actions	Describes the actions you can send to IDOL. Actions allow you to query IDOL, and to instruct it to perform a variety of operations.
Configuration Parameters	Describes the parameters that determine how the IDOL operates. Configuration parameters are set in the IDOL configuration file.
Index Actions	Describes the index actions you send to IDOL. Index actions allow you to index content into IDOL, and to administer the IDOL Data index.
Service Actions	Describes service actions. Service actions allow you to return data about the IDOL service, and to control IDOL.

Related Topics

- Send Actions to IDOL, on page 86

Chapter 7: Configure IDOL Server

This section describes how to configure IDOL server by editing the configuration file.

IDOL Configuration Files

When you install a component setup, each IDOL component that you install has a default configuration file. You configure IDOL components by modifying these configuration files.

The configuration file for a component has the same name as the main component executable file. For example, the IDOL Content component is `content.exe` and the configuration file is `content.cfg`.

You can modify the configuration parameters to customize your IDOL configuration.

Unified Configuration

In a unified IDOL Server, you configure IDOL by manually editing the IDOL Server configuration file.

The IDOL configuration file contains the parameters that determine how the main IDOL components operate, except for the IDOL Agentstore component which has a separate configuration file. You can modify the configuration parameters to customize IDOL.

The IDOL Server configuration file is in the following directory of your IDOL installation:

```
InstallDir\idolserver.cfg
```

where *`InstallDir`* is the main installation directory.

Modify Configuration Parameter Values

You modify IDOL 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 IDOL 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.

Encrypt Passwords

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

NOTE: The AES encryption method has been hardened in version 12.9.0 and later. Micro Focus strongly recommends that you reencrypt all passwords in configuration files by using the updated tool.

The older AES encryption format and basic encryption methods are now deprecated. Passwords that you have encrypted with older versions continue to work, but IDOL logs a warning. Support for these older encryption methods will be removed in future.

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 IDOL installation folder.
2. At the command line, type:

   ```
   autpassword -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. IDOL 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 IDOL installation folder.
2. At the command line, type:

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

   where:

Option	Description
-*tEncryptionType*	The type of encryption to use: • **`AES`** - AES256 • **`Basic`** **DEPRECATED:** The basic encryption type is deprecated in version 12.9.0 and later. Use the more secure AES encryption instead. Passwords that you have encrypted with older versions continue to work, but IDOL logs a warning. Support for this older encryption method will be removed in future. For example: **`-tAES`**
-o*KeyFile*	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: **`-oKeyFile=./key.ky`** **NOTE:** The full (absolute) path of the key file is included in the encrypted value, because IDOL requires the key to decrypt the password. If you move or rename the key file, this path becomes invalid and you must update the encrypted value.
-cFILE -sSECTION -pPARAMETER	(Optional) You can use these options to write the password directly into a configuration file. You must specify all three options. • **`-c`**. The configuration file in which to write the encrypted password. • **`-s`**. The name of the section in the configuration file in which to write the password. • **`-p`**. The name of the parameter in which to write the encrypted password. For example: **`-c./Config.cfg -sMyTask -pPassword`**
PasswordString	The password to encrypt.

For example:

```
autpassword -e -tBASIC MyPassword
```

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

```
autpassword -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 IDOL installation folder.
2. At the command line, type:

   ```
   autpassword -d -tEncryptionType PasswordString
   ```

   where:

Option	Description
`-t`*`EncryptionType`*	The type of encryption: • **`Basic`** • **`AES`** - AES256 For example: **`-tAES`**
`PasswordString`	The password to decrypt.

   For example:

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

   ```
   autpassword -d -tAES PasswordString
   ```

   The password is returned in plain text.

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.

CAUTION: You cannot use shared configuration files if you use any actions or index actions that modify the configuration file (for example, the `DRECREATEDBASE` index action writes the new database configuration to the file). If IDOL updates the configuration file in this way, it overwrites the link to the shared configuration file with the imported content. This behavior prevents possible conflicts if the configuration change updates any shared configuration.

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 IDOL 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 page 97.

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 IDOL 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, IDOL uses the values in the `[SharedSSLOptions]` section of the external configuration file as the values in the `[SSLOptions1]` section of the IDOL 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, IDOL 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.

Chapter 8: IDOL Performance

This section discusses methods you can use to improve the performance of your IDOL system.

For further recommendations about improving IDOL performance, refer to *IDOL Expert*.

Performance Overview

IDOL Server performance generally refers to the performance of the following IDOL Content component operations:

- **Index Data**. Documents in IDX or XML file format are added to the IDOL Content component.
- **Query IDOL server**. Users send natural language, keyword or Boolean queries to the IDOL Content component, which analyzes the concept of the query and returns documents that are conceptually similar.

To optimize IDOL Server performance, you must consider the relationship between these operations. The appropriate setup for your system depends on whether your priority is achieving fast query speeds or making new information available to users as quickly as possible.

Schedule Index and Query Operations

The IDOL Content component provides the fastest query responses when it is not indexing, and indexes fastest when it is not being queried.

One way to improve performance for both indexing and querying is to schedule indexing so that it occurs when the query load is at a minimum. For example, if your users send most queries during office hours, you can schedule indexing tasks to run during the night.

When you consider this approach, you must also consider how important up-to-date information is in your system. In some systems, users must be able to access the latest information immediately. For other systems, weekly updates to your IDOL data index are enough.

Use a Component Setup

You can improve the performance of IDOL operations by using a component setup. In this setup, you can place each IDOL component on its own hardware, maximizing the available resources.

Related Topics

- Types of IDOL Systems, on page 27.

Optimize the IDOL Content component

There are many ways to optimize the content that you store in the IDOL Content component. You can improve both index and query performance by using a stop word list, and configuring the IDOL Content component to not index numbers.

Use a Stop Word List

A stop word list is a list of common words that do not convey much meaning to sentences. For example, in English, words such as *the* and *that* do not add extra meaning in a sentence.

By default, the IDOL Content component does not index any stop words, and does not use them to search documents. This option reduces the number of terms that the IDOL Content component must index. It also means that the IDOL Content component retrieves documents based only on the relevant terms in the query.

To optimize index and query performance, ensure that your stop word list includes all common words for your system.

Index Numbers

You can determine how the IDOL Content component treats numbers when indexing documents. By default, the IDOL Content component does not index any numbers. This option reduces the number of terms that the IDOL Content component must index and search.

You can configure the IDOL Content component to index numbers, if you want users to be able to search for numeric terms. You can also configure the IDOL Content component to index numbers only if they occur in a mixed alphanumeric word (for example, IPv6).

Optimize the Index Process

The speed of the indexing process is often less critical than the speed of the query process. However, when indexing large amounts of data into IDOL, it is important to improve the efficiency of the process where possible. In addition, the way you configure the indexing process can affect the efficiency of the query process.

Delayed Synchronization

The indexing process has the following stages:

1. The IDOL Content component creates a representation of the new data in the index cache.
2. The IDOL Content component synchronizes the cache with data that it currently contains, and stores the new data on disk, removing it from the index cache.

Delayed synchronization allows you to select how the IDOL Content component synchronizes the index cache with the IDOL data. This process is useful in systems where you schedule indexing tasks at times when IDOL is also handling queries.

The `DelayedSync` parameter in the `[Server]` section of IDOL Content component configuration file allows you to specify whether the indexing process uses delayed synchronization.

If the `DelayedSync` parameter is not set, IDOL synchronizes the cache as soon as the representation of data is made. New data is available to the user (as query results) quickly. Use this setting in systems where up-to-date data is the priority.

Synchronization uses resources that the IDOL Content component could otherwise use for querying. Delayed synchronization reduces this effect by collecting multiple data representations in the index cache and then synchronizing them all with IDOL data at the same time. This process is useful in systems where query speed is more important than having the most up-to-date data.

> **NOTE:** Micro Focus recommends that you use delayed synchronization if you index a large number of small files (files that are smaller than 100MB).

To configure delayed synchronization

1. Open the IDOL Content component configuration file in a text editor.
2. In the `[Server]` section, set the `DelayedSync` parameter to **`True`**.
3. In the `[Server]` section, set the `MaxSyncDelay` parameter to the maximum length of time between synchronization operations.
4. In the `[IndexCache]` section, set the `IndexCacheMaxSize` parameter to the maximum size that the IDOL Content component index cache can grow to before synchronizing.

   > **NOTE:** When the index cache reaches this maximum size, the IDOL Content component synchronizes the cache, but data is not available for searching until the `MaxSyncDelay` has passed, or you send a `DRESYNC` index action.

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

Distribute IDOL Content component Data Across Multiple Disks

If your IDOL data becomes too big to store on one volume (as the stored terms, references, content, and so on increase in size), you can store the data files across multiple disk partitions.

To distribute IDOL data

1. Open the IDOL Content component configuration file in a text editor.
2. Find the `[Paths]` section, or create one if it does not exist.

3. Set the path parameters to the full paths to the directories where you want to store the corresponding part of the index. The following paths have the largest impact on performance:

Parameter	Description
`DynTermPath`	The directory that contains conceptual data.
`NodeTablePath`	The directory that contains content and structured data.
`StatusPath`	The directory that contains status files, and data that is streamed over the network before indexing.

   For a full list of available parameters, refer to the *IDOL server Reference*.

4. Save and close the configuration file.

5. Restart the IDOL Content component for your changes to take effect.

For example:

```
[Paths]
DyntermPath=C:\idolserver\dynterm
NodetablePath=D:\idolserver\nodetable
StatusPath=E:\idolserver\status
RefIndexPath=F:\idolserver\refindex
MainPath=G:\idolserver\main
TagPath=H:\idolserver\tagindex
```

Related Topics

- Display Online Help, on page 88

Optimize Query Operations

Query operations include any action that requests data from IDOL Server (particularly the IDOL Content component), for example:

- `Query`
- `Suggest`
- `SuggestOnText`
- `GetQueryTagValues`

Query speed can be particularly important because users require a real-time response to any query. This section describes how to optimize the performance of query operations.

The IDOL Content component `Query` action allows you to search for documents using a number of restrictions and criteria. The main types of query are:

- `Text`. The query contains a natural language expression, which can also contain Boolean or Proximity operators. The IDOL Content component finds documents that contain concepts that are similar to the given text, and that match any additional operations.

- `FieldText`. The query contains restrictions that specify values that must occur in specified fields. For example, you can restrict a query to documents that contain the term *Dog* in the *Animal* field.

Simple `Text` queries are generally quicker than `FieldText` queries.

Optimize IDOL Fields

You can optimize `FieldText` queries by configuring field processes. These processes define how the IDOL Content componentstores fields, to optimize the retrieval of the data that the field contains. For example, if the field contains a number, configure the field as a numeric field to allow IDOL server to quickly process and retrieve values for that field.

Micro Focus recommends that you use field processes to identify types of fields appropriately. For details of how to set up field processes, refer to the *IDOL Server Administration Guide*.

The following table lists the field properties that optimize `FieldText` specifiers, and the specifiers that they optimize.

Field Property	**Optimized Field Specifiers**		**Optimized Sort Operations**

`MatchType`	`ARANGE`
	`BIASVAL`
	`EMPTY`
	`EXISTS`
	`EQUALCOVER`
	`MATCH`
	`MATCHALL`
		`MATCHCOVER`
		`NOTMATCH`
		`NOTSTRING`
		`NOTWILD`
		`STRING`
		`STRINGALL`
		`WILD`
			`fieldname``:alphabetical`
			`fieldname``:decreasing`
			`fieldname``:increasing`
			`fieldname``:reversealphabetical`

`NumericType`	`BIAS`
	`EMPTY`
	`EQUAL`
	`EQUALALL`
	`EQUALCOVER`
		`EXISTS`
		`GREATER`
		`LESS`
		`NOTEQUAL`
		`NRANGE`
			`fieldname``:numberdecreasing`
			`fieldname``:numberincreasing`
			`fieldname``:decreasing`
			`fieldname``:increasing`
			`Distcartesian`
			`Distspherical`

`NumericDateType`	`BIASDATE`
	`GTNOW`
		`LTNOW`
		`RANGE`
			`fieldname``:decreasing`
			`fieldname``:increasing`

`GeospatialType`	`BIASDISTCARTESIAN`
	`BIASDISTSPHERICAL`
	`DISTCARTESIAN`
		`GEOCONTAINS`
		`GEOINTERSECTS`
		`GEOWITHIN`
			`Distcartesian`
			`Distspherical`

Field Property	Optimized Field Specifiers		Optimized Sort Operations
	`DISTSPHERICAL`	`POLYGON`	
`CountType`	`EQUALCOVER`	`MATCHCOVER`	
`BitFieldType`	`BITSET`		

Index Fields

The IDOL Content component processes index fields linguistically. It removes stop words, and stems each term before storing the terms. This process allows IDOL to return documents that match a conceptual query or contain keyword search terms.

Define fields that contain document content as index fields. For example, the document title and body. When performing `Text` queries, IDOL checks Index fields for matching terms and concepts.

Micro Focus recommends that you do not store URLs or content that you are unlikely to query as Index fields. Micro Focus also recommends that you use Match fields, rather than index fields, for fields where you query only the whole value of the field.

Match Fields

The IDOL Content component stores the value of match fields in a fast look-up structure in memory. You can define fields as match field when you frequently want to retrieve documents using the whole value of this field. For example, you might define the *Author* field of a document as a Match field so that users can search for the author.

You can also use match fields to allow users to search for values in an alphabetical range. For example, when you define the *Author* field as a Match field, users can search for books alphabetically by author.

Match fields also optimize the `Sort` action parameter when sorting on a the value of a field.

Numeric Fields

The IDOL Content component stores the value of numeric fields in a fast look-up structure in memory. You can use numeric fields to allow users to search for values or ranges of values. For example, you might make a *Price* field a numeric field so that users can restrict results to products within a price range.

Numeric Date Fields

The IDOL Content component stores the value of numeric date fields in a fast look-up structure in memory. You can use numeric date fields to allow users to search for a date or range of dates. For example, you might make a *Date* field a numeric date field so that users can restrict results to those between two dates.

Geospatial Fields

The IDOL Content component stores the value of geospatial fields in a fast look-up structure in memory. These fields can store point locations and regions. You can use geospatial fields to allow users to search for particular locations and regions, and intersections between regions. For example, you might make a *CountryBorders* field a geospatial field, so that users can restrict results to those that occur in a particular country.

Count Fields

The IDOL Content component stores the number of occurrences of count fields. You can use count fields to allow users to search for more than one string in a given field.

Bit Fields

Typically, bit fields contain information about the sets that a document belongs to. You can use bit fields to allow users to search for documents that only occur in a particular set. For example, you might make a *Workbook* field a bit field so that users can search for documents that occur in a particular workbook.

Parametric Fields

You can use parametric fields to allow users to restrict a search to certain parameters. For example, you might make a *Model* field a parametric field so that users can restrict a search to products of a particular model.

You might also want to make a field both `ParametricType` and `MatchType` to allow you to list the available values and then restrict by those values.

> **NOTE:** You can configure a field as simultaneously `MatchType` and `ParametricType` only if you set `ParametricNumericMapping` to `false`.

Field Check Fields

The IDOL Content component stores a checksum hash of the value of a field check fields. This option allows especially fast retrieval when you restrict results to the exact value of the field. For example, you might make a *Category* field a field check field so that you can restrict a search to a particular category.

Field Check fields are also useful for the `Combine` operation. In a `Query` action, you can combine results that have the same value of the field check field. The IDOL Content component then returns only one document for this field check value.

> **NOTE:** Each document that you index into the IDOL Content component must contain only one `FieldCheckType` field.

Hardware Considerations

To maximize IDOL performance, ensure that your IDOL components have adequate resources assigned:

- Ensure that IDOL server has enough memory.
- Monitor disk usage to check input and output.

Related Topics

- Recommended Minimum Hardware Specifications, on page 58

Optimize Distributed Systems

Distributed IDOL systems use a Distributed Action Handler (DAH) and Distributed Index Handler (DIH) to distribute actions and index actions respectively. In these systems, you can use one of two modes:

- **Mirror mode** uses several identical child servers to process client queries. You can use this system when you have large numbers of queries, to minimize the query load on individual child servers.
- **Non-mirror mode** distributes all data across several child servers, which can improve the speed of indexing for the system, and increases the amount of data that child server contains. You can use this system when you have a large amount of data to index, or to minimize the index load on individual servers. As servers might respond more slowly while they are indexing data, this method can also improve query response speed.

In both modes, you can consider other modes in the DIH and DAH to distribute ACI and index actions most efficiently.

Related Topics

- Types of IDOL Systems

Distributed Action Handler Performance

The Distributed Action Handler distributes actions between child servers. It also performs some additional processing on actions. For example, in mirror mode it can apply templates, and recognize and distribute administrative actions to all child servers.

Depending on your system, you can use Distributed Action Handler modes to reduce the amount of additional processing that Distributed Action Handler performs, which can improve the performance of the system.

Fast Mirror Mode

In fast mirror mode, Distributed Action Handler distributes actions between mirrored child servers but does not perform any additional processing to actions. You can use this option to increase the performance of your IDOL system in mirror mode when you use Distributed Action Handler only for requesting information from child servers.

When running the Distributed Action Handler in fast mirror mode:

- You cannot send state-changing actions to the Distributed Action Handler (for example, actions including the `State` and `StoreState` action parameters).
- You cannot request additional processing on actions (for example, actions that use the `Template`, `EncryptResponse` or `Output` action parameters).
- You cannot use document IDs in action parameters.
- You cannot use distributed actions (actions to send to all child servers) or asynchronous actions.

Simple Combinator Mode

In simple combinator mode, Distributed Action Handler does not use virtual databases to distribute and combine actions in non-mirror mode. Distributed Action Handler forwards the `DatabaseMatch` action parameter to child servers, and combines results from all child servers.

This option reduces the amount of processing that Distributed Action Handler must perform to distribute and combine virtual database.

Distributed Index Handler Performance

The DIH distributes index actions to its child servers.

In standard mirror and non-mirror mode, DIH forwards the IDX or XML index file to all its child servers, with instructions to the child servers about which documents it must index. You can use different modes in the DIH to reduce the amount of data that the DIH, or the child servers, must process.

Preserve DREADD

The DIH configuration parameter `PreserveDREADD` allows it to distribute index actions more quickly.

By default, DIH accepts an IDX or XML file input and converts the original `DREADD` index action into a `DREADDDATA` index action, which contains all the IDX or XML data. If you set `PreserveDREADD` to `true` in the configuration file, DIH forwards the `DREADD` action directly. This option reduces the network load.

To use this parameter, each child IDOL Content component must be able to access the original IDX or XML file.

You can use `PreserveDREADD` only in mirror mode, simple non-mirror mode or `DistributeOnBatch` mode. For example, you cannot use `PreserveDREADD` if you have set `DistributeByReference` to `true`. In advanced distribution modes, DIH must parse the IDX or XML file to distribute the documents correctly.

Distribute On Batch

In distribute on batch mode, DIH indexes data to alternating child servers. This method is an efficient way of distributing the data to child servers in non-mirror mode. You can use this method if:

- you index a steady stream of IDX or XML files, with similar numbers of documents. This mode might not be appropriate if you infrequently index very large IDX or XML files.
- you do not need to remove duplicate documents.

Advanced Distribution Modes

DIH has several distribution modes that determine how it distributes documents between child servers in non-mirror mode. Unlike standard non-mirror mode, these methods send each child server only the documents that it must index. This option reduces the amount of data that each child server receives.

> **NOTE:** If you use these methods, the number of DIH child servers is fixed. To add child servers, you must clear all engines, add the new servers, and then re-index using the DIH.

- **Distribute by reference**. DIH distributes documents to child servers based on the document reference. This method ensures that you can prevent duplicate documents that have the same reference.
- **Distribute by date**. DIH distributes documents to child servers according to the document date. You configure a date range for each child server.
- **Distribute by fields or field values**. DIH distributes documents to child servers based on the value of a certain field.
  - In distribute by fields mode, DIH sends documents with the same value of the specified field to the same child server. DIH determines which child server receives a given field value.
  - In distribute by field values mode, you can specify which child servers receive documents that contain certain field values.

Round Robin Mode

The Round-Robin indexing mode rotates indexing between several child servers, so that indexing and querying are performed by different child servers at different times.

The IDOL Content component 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. It then indexes only into this child server, which has optimal indexing performance.

After the specified time period, DIH starts indexing to a different child server and makes the previous server available for querying.

Chapter 9: Tutorial: Index Data into IDOL

This chapter describes how to configure the IDOL Content component index and index data.

Overview

The IDOL Content component data index is a representation of all the data that you can search for in IDOL Server. When the IDOL Content component indexes data, it processes the terms and fields in the new documents. This process allows it to retrieve the documents quickly when you send queries.

Before you index documents, you must configure the data index to process documents according to your requirements. You can then create and index documents, either by using an IDOL connector, or by creating an IDX or XML file and indexing them manually.

Related Topics

- Configure the IDOL Content component Index
- Create Documents
- Pre-Process Documents
- Index Documents

Configure the IDOL Content component Index

Before you begin to index documents into the IDOL Content component, you must configure the data index to process the documents correctly.

You can configure:

- IDOL databases to store your data.
- field processes to correctly identify and process the fields in the documents you index.
- language types to correctly process different languages and encodings.

Related Topics

- Configure Databases
- Configure Fields
- Configure Languages

Configure Databases

The IDOL Content component can store documents in different databases. You can configure up to 65534 databases in an IDOL Content component.

Indexing data into different databases allows you to restrict queries to data in a particular database. This can improve the performance of your queries.

You can add databases to the IDOL Content component by editing the configuration file or by sending an index action. For details of adding a database by editing the configuration file, refer to the *IDOL Server Administration Guide* and the *IDOL Server Reference*.

To add IDOL Content component databases

- Send a `DRECREATEDBASE` action (case sensitive) from your Web browser:

  ```
  http://ContentHost:indexPort/DRECREATEDBASE?DREdbname=databaseName
  ```

  where,

`ContentHost`	is the IP address or host name of the machine on which the IDOL Content component is installed.
`indexPort`	is the IDOL Content component index port (specified as `IndexPort` in the IDOL Content component configuration file `[Server]` section).
`databaseName`	is the name of the database that you want to create.

For example:

```
http://12.3.4.56:20001/DRECREATEDBASE?DREdbname=Archive
```

This action uses port 20001 to create a new database named `Archive` on the IDOL Content component installed on the machine with the IP address `12.3.4.56`.

You can also add other index action parameters to this action, for example to make the new database internal. For details, refer to the *IDOL Server Reference*.

Configure Fields

The documents in the IDOL Content component store data in fields. These fields can contain many different types of information. For example, one field might contain the bulk of the document content, and another field might contain only the name of the author.

When you index data into the IDOL Content component, it is important to set up field processes so that the IDOL Content component treats each field correctly.

Choose Field Properties

Before you set up fields in IDOL, you must consider the fields in the documents that you want to index.

There are a large number of different field properties that you can apply to IDOL fields. Fields can contain information about the document, or values that you want to retrieve in queries.

For a complete list of the types of fields that you can store in the IDOL Content component, refer to the *IDOL Server Administration Guide* and *IDOL Server Reference*.

Fields that contain information about the document	
`ReferenceType`	Reference fields contain a unique document reference, which you can use to remove duplicate documents, and to retrieve a specific document. Each document must contain at least one reference field. The IDOL Content component also uses the reference field to populate the `autn:reference` metadata field.
`DateType`	Date fields contain the document date, which the IDOL Content component uses to populate the `autn:date` metadata field. If a document does not have a date field, Content uses the date that the document was indexed.
`TitleType`	Title fields contain the document title, which the IDOL Content component uses to populate the `autn:title` metadata field.
`DatabaseType`	Database fields contain the IDOL database that the IDOL Content component must index the document into. If the document does not contain a database field, you must specify the database in the index action.
`LanguageType`	Language fields contain the language type of the document, which the IDOL Content component uses to find the appropriate language configuration. The IDOL Content component also uses this field to populate the `autn:languagetype` metadata field.
`SecurityType`	Security fields contain the security type of the document, which the IDOL Content component uses to index the document according to the specified security configuration.
`ACLType`	ACL fields contain document access control lists (ACLs), which determine the access restrictions for that document.
`ExpireDateType`	Expire date fields contain the date that the document expires. On this date, IDOL processes the document according to the expiry rules for the database, for example to delete the document or move it to an archive database. The IDOL Content component also uses this field to populate the `autn:expiredate` metadata field.
`SectionBreakType`	Section break fields contain the section number for a document section,

	which the IDOL Content component uses to populate the `autn:section` metadata field.
`ParametricType`	Parametric fields contain values that you want to use to restrict queries. In a parametric query, you can return all values that occur in a certain parametric field in all documents.
`ParametricRangeType`	Parametric fields contain values that you want the IDOL Content component to dynamically analyze to generate numeric ranges in parametric queries.
Fields that contain document content	
`Index`	Index fields contain document content. The IDOL Content component processes these fields linguistically and stores terms to allow fast data retrieval. Typically, you store the document content and title field as index fields. Do not store data as index fields if: • you do not need to query the field contents. • you query only the whole value of the fields. It is more efficient to store these fields as `MatchType` fields and query the fields using a `FieldText` specifier such as `MATCH`.
`SourceType`	Source fields contain document content that the IDOL Content component uses to create document summaries and suggest conceptually similar documents. If you do not configure source fields, the IDOL Content component uses the configured Index fields.
`LangDetectType`	Language Detection fields contain document content that the IDOL Content component uses to automatically detect the document language. If you do not configure a language detection fields, the IDOL Content component uses the configured source fields.
`HighlightType`	Highlight fields contain document content that IDOL server can highlight when you send the `Highlight` action (or a query action with the `Highlight` parameter set to **`true`**).
Field properties that optimize FieldText query operators	
`MatchType`	See Match Fields, on page 104
`NumericType`	See Numeric Fields, on page 104
`NumericDateType`	See Numeric Date Fields, on page 104
`GeospatialType`	See Geospatial Fields, on page 105
Field properties that determine how to display fields	
`HiddenType`	Hidden fields do not appear
`PrintType`	Print fields are displayed in a `Query` action when the `Print` parameter is set to **`Fields`** (the default value).

Set up Field Processes

To identify properties for different fields, you must define field processes. In a field process, you define:

- the set of fields that the field process applies to
- the property that applies to this process

You can have multiple field processes that share the same property. You must then create a configuration section for each property that you use, and define the field properties.

NOTE: Use the following formats to identify fields:

- `/FieldName` to match root-level fields
- `*/FieldName` to match all fields except root-level
- `/`*`Path`*`/FieldName` to match fields that the specified path points to.

Field names must not contain spaces nor accents, and they must not start with a number. For IDX documents, the IDOL Content component converts these text elements to underscores (_) when it indexes the fields. You must also change any queries that reference these field names to use the modified field name.

To apply processes to fields or documents that contain specific fields

1. Open the IDOL Content component configuration file in a text editor.
2. In the `[FieldProcessing]` section, list the processes to apply to fields. For example:

   ```
   [FieldProcessing]
   0=IndexFields
   1=DateFields
   2=DatabaseFields
   3=SetReferenceFields
   ```

3. Create a configuration section for each process listed.
   - Set `PropertyFieldCSVs` to a comma-separated list of fields that this process applies to.
   - Set `Property` to the name of the property configuration section.

   NOTE: Each property must have a unique configuration section name.

   - (Optional) Set `PropertyMatch` to a comma-separated list of values that the field must have to be processed. For example, you can use this parameter to set up a process that identifies security or language fields.
   - (Optional) Set `PropertyNegativeFieldCSVs` to a comma-separated list of fields that this process does not apply to. For example, if you use a wildcard value in `PropertyFieldCSVs`, you can define any exceptions in `PropertyNegativeFieldCSVs`.

   For example:

```
[IndexFields]
Property=Index
PropertyFieldCSVs=*/DRECONTENT,*/DRETITLE

[DateFields]
Property=Date
PropertyFieldCSVs=*/DREDATE,*/harvest_time

[DatabaseFields]
Property=Database
PropertyFieldCSVs=*/DREDBNAME,*/Database*
PropertyNegativeFieldCSVs=*/DatabaseNumber

[SetReferenceFields]
Property=Reference
PropertyFieldCSVs=*/DREREFERENCE,*/DRETITLE
```

4. Create a section for each property. Specify appropriate configuration parameters for each property. These configuration parameters define the processes that apply to all the fields (or all documents that contain the fields) that you previously associated with the processes.

   For example:

```
[Index]
Index=TRUE

[Date]
DateType=TRUE

[Database]
DatabaseType=TRUE

[Reference]
ReferenceType=TRUE
TrimSpaces=TRUE
```

> **NOTE:** For some properties, you must reindex your content if you want to change the property value after you have indexed content into the IDOL Content component.
>
> For details, refer to the *IDOL Server Reference*.

5. Save and close the configuration file.

6. Restart the IDOL Content component for your changes to take effect.

Related Topics

- Display Online Help

Configure Languages

The IDOL Content component can process documents in multiple languages and encodings. For each language that you want to use, you must define the language types in the IDOL Content component configuration file. You must also configure the IDOL Content component to classify documents, either by automatically detecting the language and encoding, or by reading the language type from a field.

Define Language Types

To run the IDOL Content component in multiple languages, specify the language types you want the IDOL Content component to process. A language type is a combination of the language and encoding.

NOTE: You must specify languages and language types before you index data into the IDOL Content component.

To define language types

1. Open the IDOL Content component configuration file in a text editor.
2. Find the `[LanguageTypes]` section. List the languages that you want the IDOL Content component to process. You must use ASCII characters to specify the language names.

   For example:

   ```
   [LanguageTypes]
   0=English
   1=Afrikaans
   2=General
   ```

3. For each language, create a configuration section that matches the name you defined in the `[LanguageTypes]` section.
4. In this section, specify appropriate settings that determine how the IDOL Content component handles this language. For details on the configuration parameters you can use, refer to the *IDOL Server Reference*.
5. For each section, set the `Encodings` parameter to a list of the encodings and corresponding language types used by the language. List each encoding and language in the format *`encoding`*`:`*`languagetype`*. Separate multiple language types with commas.

   For example:

   ```
   [english]
   Encodings=ASCII:englishASCII,UTF8:englishUTF8
   Stoplist=english.dat
   IndexNumbers=1

   [afrikaans]
   Encodings=ASCII:afrikaansASCII,UTF8:afrikaansUTF8
   IndexNumbers=1
   ```

```
[general]
Encodings=UTF8:generalUTF8,ASCII:generalASCII,CYRILLIC:generalC
YRILLIC
IndexNumbers=1
```

6. Save and close the configuration file.

7. Restart IDOL server for your changes to take effect.

You can now configure the IDOL Content component to associate the language types you defined with documents.

Associate Language Types with Documents

After you define all the language types you want the IDOL Content component to process, set up a field process that allows the IDOL Content component to associate these language types with documents.

The way you configure the field process depends on the documents that you want to index:

- If all the documents contains a field that exactly specifies the language type, configure a field process to define this field as a `LanguageType` field. The language types that appear in this field must exactly match the language types that you define in the `[LanguageTypes]` configuration section. See Configure Fields, on page 110.

- If the documents contain a field that specifies the language, but does not exactly specify the language type, you can configure field processes to detect the language from this field data.

To configure the IDOL Content component to detect the language type from field data

1. Open the IDOL Content component configuration file in a text editor.

2. In the `[FieldProcessing]` section, define a field process for each language that you want to detect.

   For example:

   ```
   [FieldProcessing]
   0=DetectArabic
   1=DetectEnglish
   2=DetectFrench
   ```

3. Create a configuration section with the same name as each of the field processes you defined in the `[FieldProcessing]` section.

4. In this section:

   - Set `Property` to the name of the property for the specified language type.

   - Set `PropertyFieldCSVs` to a comma-separated list of fields that can contain the language data.

   - Set `PropertyMatch` to a comma-separated list of values that this field might contain to identify the specified language type.

   For example:

```
[DetectArabic]
Property=SetArabicProperty
PropertyFieldCSVs=*/DRELANGUAGETYPE,*/LANG
PropertyMatch=arabic

[DetectEnglish]
Property=SetEnglishProperty
PropertyFieldCSVs=*/DRELANGUAGETYPE,*/LANG
PropertyMatch=*eng*,uk,*british

[DetectFrench]
Property=SetFrenchProperty
PropertyFieldCSVs=*/DRELANGUAGETYPE,*/LANG
PropertyMatch=*fre*,fran*
```

5. Create a configuration section for each property that you define in the field processing sections.
6. In the property configuration section, set the `LanguageType` parameter to the language type to use to define documents that match this property (that is, that contain a field with a matching value for the field process). This language type must match one of the language types you configure in the `[LanguageTypes]` configuration section.

   For example:

   ```
   [SetArabicProperty]
   LanguageType=Arabic

   [SetEnglishProperty]
   LanguageType=English

   [SetFrenchProperty]
   LanguageType=French
   ```
7. Save and close the configuration file.
8. Restart the IDOL Content component for your changes to take effect.

Create Documents

Unless your data is in XML format, you must first import it. You can import data using one of two methods.

- **Import with a connector**. The IDOL connectors (for example, File System Connector, HTTP Connector, Database Connector, and so on) retrieve documents from different repositories and import them into IDX or XML file format. For details of how to configure a connector to import data and index it into IDOL, refer to the appropriate Connector Administration Guide.
- **Import manually**. You can create a text file in either XML or IDX format, which contains the information that you want to index into your IDOL Content component in specific IDOL fields.

For details of how to create a manual IDX document, refer to the *IDOL Server Administration Guide*.

Pre-Process Documents

You can use the Connector Framework Server (CFS) or NiFi Ingest to process documents before you index them. For example, you can run Eduction, modify document fields, categorize documents, or write the documents to disk.

For more information about how to use CFS to process your documents, refer to the *Connector Framework Server Administration Guide*.

For information about how to use NiFi Ingest to process your documents, refer to the *NiFi Ingest Help*.

Index Documents

You can index only files in XML or IDX format into the IDOL Content component. If the data that you want to index is in XML format, you can index it directly into the IDOL Content component, without having to first *import* it (convert its content and metadata to IDX).

IDOL connectors use the `DREADD` and `DREADDDATA` index actions to index data into IDOL Server. You can also use these actions to directly index data into the IDOL Content component.

Use DREADD to Index IDX and XML Files Directly

The `DREADD` index action (case sensitive) directly indexes an IDX or XML file that is located on the same machine as the IDOL Content component. For example:

```
http://ContentHost:indexPort/DREADD?filename&optionalParams
```

where,

ContentHost	is the IP address or host name of the machine on which the IDOL Content component is installed.
indexPort	is the indexing port of the IDOL Content component (specified in the *IndexPort* parameter in the `[Server]` section of the IDOL Content component configuration file.
filename	is the file name or location of the IDX or XML file to index.
optionalParams	are any optional index action parameters that you want to add. **NOTE:** You must set the `DREDbName` parameter if your IDOL Content component configuration does not contain a field process to read the database from a field.

For a full list of available parameters for the `DREADD` action, refer to the *IDOL Server Reference*.

Use DREADDDATA to Index Data Over a Socket

The `DREADDDATA` index action (case sensitive) allows you to directly index data over a socket into the IDOL Content component.

NOTE: This index action requires a POST request method. For more information, refer to the *IDOL Server Administration Guide*.

For example:

```
http://
ContentHost
:indexPort/DREADDDATA?OptionalParamsData#DREENDDATAKillDuplicatesOption\n\n
```

where,

ContentHost	is the IP address or host name of the machine on which the IDOL Content component is installed.
indexPort	is the indexing port of the IDOL Content component (specified in the *IndexPort* parameter in the `[Server]` section of the IDOL Content component configuration file.
optionalParams	are any optional index action parameters. `DREADDDATA` accepts the same optional parameters as the `DREADD` index action, except for *KillDuplicatesOption*.
Data	is the content of the IDX or XML document to index. You must add `#DREENDDATA` to the end of your data. You can use gzipped documents, but `#DREENDDATA` must not be compressed. This parameter is required.
KillDuplicatesOption	is the option to use for removing duplicate documents.

For details of all the parameters available in the `DREADDDATA` index action, refer to the *IDOL Server Reference*.

Index Data in a Distributed IDOL Setup

Configuration and indexing in a distributed system is very similar to using an individual IDOL Content component, except that you must configure multiple child servers, and send your index actions to the DIH, instead of Content.

In most cases, you use an identical or very similar configuration for each of your child IDOL Content components, even in non-mirror mode, to ensure that data is processed and indexed consistently.

For example, you usually configure the same fields, languages, and databases, in all your child servers (although you might need to configure different port settings, and paths).

After you configure your child servers, you configure the DIH to distribute data to those child servers. For more information, refer to the *DIH Administration Guide*.

You send index actions (such as `DREADD` and `DREADDDATA`) to the DIH, which distributes the data to its child servers. Depending on your DIH configuration, DIH might also perform some preprocessing.

Related Topics

- Distributed Setup, on page 33

Chapter 10: Query IDOL

This section describes the IDOL action responses, and the basics of query operations in the IDOL Content component.

IDOL ACI Responses

Interactions with IDOL components are structured according to the Autonomy Content Infrastructure (ACI) architecture. The ACI architecture defines how clients communicate with IDOL components and how the IDOL components respond. In these interactions, clients submit ACI requests to IDOL components as GET or POST HTTP requests, and receive ACI responses in the form of XML or JSON.

An ACI request consists of an ACI action and a collection of ACI parameters. See Send Actions to IDOL, on page 86.

For example, the following ACI request sends an action to a local IDOL Content component on port 9100. The action is `Query`, and there are two ACI parameters, `Text` with a value of `IDOL Server`, and `MaxResults` with a value of `20`.

```
http://localhost:9100/action=Query&text=IDOL Server&MaxResults=20
```

The ACI action determines the collection of valid ACI parameters. For a given ACI action, some ACI parameters might be required while others are optional.

The ACI responses in XML use a uniform structure for the top two levels of the document tree. For example:

```
<autnresponse xmlns:autn=”http://schemas.autonomy.com/aci/”>
   <action>QUERY</action>
   <response>SUCCESS</response>
   <responsedata>...</responsedata>
</autnresponse>
```

Every ACI response returns the `<autnresponse>` node, defines the `autn` namespace as shown, and has three children of the document root:

`<action>`	contains the ACI action of the request.
`<response>`	contains either `SUCCESS` or `ERROR` indicating whether the ACI request

	succeeded or failed.
`<responsedata>`	contains the rest of the response, the structure of which depends on the ACI action.

The Query ACI Action

The query action is an ACI action supported by the IDOL Content component that searches the corpus of indexed documents for those matching the restrictions of the query. Query action ACI parameters come in two types:

- Search parameters define restrictions on the result set of documents.
- Response parameters change the information that returns with the result set of documents.

Search Parameters

This section describes many of the most important parameters for the `Query` action that you can use to restrict the document result set.

For more information about each of these parameters, refer to the *IDOL Server Reference*.

Text

The `Text` parameter defines restrictions on the content of matching documents. You can specify restrictions in plain text, or by using Boolean and proximity operators.

If `Text` contains plain text, for example *airplane leases*, the IDOL Content component treats the query as a conceptual query. It *tokenizes* the query text, removes *stop words*, and *stems* the remaining tokens to produce a collection of terms.

NOTE: These functions are language-specific. For more information about language configuration in IDOL, refer to the *IDOL Server Administration Guide*, and *IDOL Expert*.

The IDOL Content component finds documents that contains these terms, and uses statistical algorithms to quantify the relevance of the match to the original query. It assigns a relevancy score to each result document. By default, Content also uses this relevancy score to sort the results of the query.

For example:

```
action=query&text=the airplane leases
```

The IDOL Content component tokenizes this query text into *the*, *airplane*, and *leases*. The term *the* is a stop word, so Content ignores it. It stems the other two terms to produce the terms *airplan* and *leas*. Stemming allows Content to match documents that contain alternate forms of the words airplane and leases (for example, *lease*, *leasing*, and *leased* also stem to *leas*).

If `Text` contains Boolean or proximity operators, such as `AND`, or `OR`, Content treats the query as a Boolean query. As with conceptual queries, Content stems the search terms and ignores stop words, but documents match the query only if they satisfy the Boolean or proximity expression.

FieldText

The `FieldText` parameter defines restrictions on the fields of documents that match the query. `FieldText` restrictions have the following format:

`FIELDOPERATOR``{`*`Value1`*`[,`*`Value2`*`,...]}:`*`FieldName1`*`[:`*`FieldName2`*`:...]`

where,

`FIELDOPERATOR`	is the type of comparison or test to perform on the value of the specified fields, against the specified values.
`Value1` `[,`*`Value2`*`,...]`	is a comma-separated list of field values.
`FieldName1` `[:`*`FieldName2`*`:...]`	is a colon-separated list of field names.

There are more than 40 different field operators. The following examples demonstrate the `FieldText` restriction format and some of the most common field operators. For a full list of field operators, refer to the *IDOL Server Reference*. For more information about fields and optimizations for the `FieldText` queries, see Configure Fields, on page 110.

NOTE: By default, `FieldText` matches are not case sensitive. For more information about case sensitive search, refer to *IDOL Expert*.

The following `FieldText` restriction matches documents whose `DOCUMENT_TYPE` field contains the value *record*:

```
FieldText=MATCH{record}:DOCUMENT_TYPE
```

The following `FieldText` restriction matches documents whose `FILETYPE` field contains one of the values *pdf*, *doc*, or *txt*:

```
FieldText=MATCH{pdf,doc,txt}:FILETYPE
```

The following `FieldText` restriction matches documents whose `DIR` or `DIRECTORY` field exactly matches one of *docs*, *archive*, or *work*:

```
FieldText=MATCH{docs,archive,work}:DIR:DIRECTORY
```

You can combine `FieldText` restrictions by using Boolean operators, such as `AND`, `OR`, and `NOT`, and group the restrictions by using parentheses. For example:

```
FieldText=MATCH{dog}:ANIMAL+AND+(MATCH{celery}:VEGETABLE+OR+NOT+MATCH
{quartz}:MINERAL)
```

The IDOL Content component also includes field operators for numerical comparison. The following `FieldText` restriction matches documents for which `SCORE1` is greater than `.95` and `SCORE2` is less than `.85`:

```
FieldText=GREATER{.95}:SCORE1+AND+LESS{.85}:SCORE2
```

The following `FieldText` restriction matches documents that have a `COUNT` field, regardless of its value:

```
FieldText=EXISTS{}:COUNT
```

The following `FieldText` restriction matches documents whose `BRAND` field contains *tech* as a substring:

```
FieldText=STRING{tech}:BRAND
```

The following `FieldText` restriction matches documents whose `EMOTION` field contains a string that has the same stem as the word *loving*.

```
FieldText=TERM{loving}:EMOTION
```

For example, if the `EMOTION` field has any of the values *love*, *loving*, *loves*, or *lover*, the document matches.

The following `FieldText` restriction matches documents whose position (stored in the `LAT` and `LONG` fields) is within a distance of 5 kilometers from the point with latitude 52.2°N and longitude 0.12° E.

```
FieldText=DISTSPHERICAL{52.2,0.12,5}:LAT:LONG
```

The following `FieldText` restriction matches documents whose `DUE_DATE` is before the current time.

```
FieldText=LTNOW{}:DUE_DATE
```

Combine

The `Combine` parameter contains instructions on how to treat certain groups of documents as a single result for the `Query` action.

The most important value for `Combine` is **`Simple`**. By default, when you index long documents, the IDOL Content component splits the documents into shorter sections. When Content processes a query, it can return each of these sections as a separate search result. If you set `Combine` to **`Simple`**, Content returns only the best matching section from each document.

It is a best practice to set `Combine` to **`Simple`** in every Content `Query` unless there is a specific reason not to.

DatabaseMatch

Every document in the IDOL Content component belongs to exactly one database. The `DatabaseMatch` parameter specifies a comma-separated list of databases that resulting documents can come from.

By default (if `DatabaseMatch` is missing or empty), Content retrieves results from any database. Content applies database restrictions early in the query process, and these restrictions are generally quick, so using a `DatabaseMatch` can improve the performance of your query.

It is a best practice to set `DatabaseMatch` in every Content `Query` to be as restrictive as possible.

> **NOTE:** A DatabaseMatch restriction is equivalent to the following `FieldText` operation:
>
> `MATCH{`*`DatabaseNames`*`}:autn_database`

Start and MaxResults

Together, `Start` and `MaxResults` control the paging of query results.

`MaxResults` indicates the total maximum number of results to retrieve from the index. By default, the IDOL Content component returns the results in order of the relevance score, which indicates how well the document matches the query. You can change the order of the results by setting the `Sort` parameter (see Sort, on page 128). Content returns the top results according to your sort, up to the value of `MaxResults`.

`Start` indicates the first result to return from the set. This option allows you to return results in pages. For example, the following query returns up to 50 results. If there are more than 50 results, it returns the best 50.

```
action=Query&Text=bananas&MaxResults=50
```

For the following query, Content finds the best 50 results that match the query, and then returns these results from the eleventh best onwards. The response therefore contains 40 results (results 11 to 50).

```
action=Query&Text=bananas&MaxResults=50&Start=11
```

MinScore

The `Minscore` parameter specifies the smallest relevancy score that a document must have to the query for the IDOL Content component to return it as a result. For example:

```
action=Query&Text=aircraft&MinScore=60
```

If this query has 10 matching documents, with scores of 100, 90, 80, 70, and so on down to 10, the query returns five results (the documents with relevancy scores of 100, 90, 80, 70, and 60).

StoreState, StateMatchID, and StateDontMatchID

The `StoreState` parameter allows you to save an IDOL Content component search. When you set `StoreState` to **`True`**, Content returns an alphanumeric token with the query. This token is a reference to the original set of search results.

You can set this token as the value of the `StateMatchID` parameter to restrict the results of a query to only those in the original stored search. In this way, you can use `StoreState` and `StateMatchID` together to implement 'search within a search' functionality in IDOL.

The `StateDontMatchID` parameter is the opposite of `StateMatchID`. When you set `StateDontMatchID` to the value of a stored state token , the query excludes any result that was a result in the original search. Effectively, `StoreState` and `StateDontMatchID` implement 'search outside a search' functionality.

UserMetaFields

The `UserMetaFields` parameter allows you to combine the field values in result documents into a new field (a *metafield*) dynamically at query time. This method allows you to create on-the-fly analytics for your documents. The format of `UserMetaFields` is:

`UserMetaFields=`*`OPERATION`*`{`*`UserFieldName`*`}:`*`DocumentFieldName`*

The following examples demonstrate both the `UserMetaFields` format, and some of the existing operators. For a full list of operators, refer to the *IDOL Server Reference* or the *IDOL Server Administration Guide*.

In the following example, the `TOTAL` metafield for a matching document is the sum of all the `PRICE` fields in that document:

```
UserMetaFields=ADD{TOTAL}:PRICE
```

You can create multiple metafields with one UserMetaFields parameter. In the following example, the `TOTALCOST` metafield for a matching document contains the sum of all its `COST` fields, and the `AVGREVENUE` field contains the average (mean) of the `REVENUE` fields:

```
UserMetaFields=ADD{TOTALCOST}:COST+MEAN{AVGREVENUE}:REVENUE
```

You can use metafields in the `Sort` and `FieldText` parameters of the `Query` where you create them. In the following example, the IDOL Content component creates a `MAXPRICE` metafield for each document that matches the initial query stages, which is the maximum of the `PRICE` fields in the document. The query returns only documents for which the `MAXPRICE` metafield is greater than `500`.

```
UserMetaFields=MAX{MAXPRICE}:PRICE&FieldText=GREATER{500}:autn_user_MAXPRICE
```

Response Parameters

This section describes many of the most important parameters for the `Query` action that you can use to modify the query response.

For more information about each of these parameters, refer to the *IDOL Server Reference*.

Predict and TotalResults

The `TotalResults` parameter allows you to find out how many query results exist in the IDOL Content component (when this values is higher than `MaxResults`). To return this information, set `TotalResults` to **`True`**.

For performance reasons, when there are a large number of results, the IDOL Content component does not normally compute the exact number. Instead, Content makes a guess at the total number of results, based on the distribution of the results that it has already found. This guess is almost always of the right order of magnitude, and is usually within 20% of the exact number.

To make Content compute the exact number of results, you can set the `Predict` parameter to **`False`**. However, this might incur a significant performance penalty. When you set `TotalResults` to **`True`** and `Predict` to **`False`**, Content also returns a breakdown of the available query results by database.

Print and Printfields

The `Print` and `Printfields` parameters control the information from result documents that returns. The most important values for `Print` are **`Fields`**, **`All`**, **`None`**, and **`NoResults`**.

The default value for `Print` is **`Fields`**. In this case, the IDOL Content component returns metadata, and either any fields in the document that are configured as `PrintType`, or the fields that you list in the `PrintFields` parameter. If you set `PrintFields`, only the specified fields return.

For example, the following query returns the `PrintType` fields of the results:

```
action=Query&Text=pillows&Print=Fields
```

The following query returns the `FIRMNESS`, `WEIGHT`, and `STUFFING` fields of the results, along with the document metadata. No other fields return.

```
action=Query&Text=pillows&Print=Fields&PrintFields=FIRMNESS,WEIGHT,STUFFING
```

When you set `Print` to **`All`**, Content returns all fields in the results documents (except for hidden fields).

When you set `Print` to **`None`**, Content prints metadata such as the reference, document ID, and date for each result, but not the document content or fields.

When you set `Print` to **`NoResults`**, Content does not print any information about results at all; it returns only the number of results. This option is often used with the `QuerySummary` parameter (see QuerySummary, on the next page) to return just the summary terms and phrases.

QuerySummary

The `QuerySummary` parameter allows you to extract important common words and phrases from result documents. When you set `QuerySummary` to **`True`**, the IDOL Content component returns a list of these summary words and phrases at the top of the response (above the results).

You can use the collection of words and phrases as query guidance, where each phrase is a potential search that you can use to refine the original query. The query summary results might also expose aspects of the result set that you might not have known about originally.

For more information about using query summaries, refer to *IDOL Expert*, and the *IDOL Server Administration Guide*.

ResponseFormat

The `ResponseFormat` parameter is available for all ACI actions. By default, all IDOL responses are in XML format. You can set `ResponseFormat` to **`JSON`** to return JSON, or to **`config`** to return results in IDOL configuration file format (a series of lines in `field=value` format).

Sort

The `Sort` parameter controls the order in which results return. There are many options for sorting, but the most important ones are **`Date`**, **`Relevance`**, and field sort. For field sort, you specify the name of the field whose value you want to sort by, and a sorting method (for example, **`alphabetical`**, or **`numberdecreasing`**).

For example:

```
Sort=relevance+date
```

This example sorts the query results by relevance. If two or more documents have the same relevance score, the IDOL Content componentsorts them by the document date.

For a full list of the available sort and field sort options, refer to the *IDOL Server Reference*.

Highlight, StartTag, and EndTag

The `Highlight` parameter controls how the IDOL Content component highlights match query terms and phrases in the result documents. For example, you can set `Highlight` to **`Terms`** to use highlight tags to surround each instance of a query term in the content of the result documents. You can use the `StartTag` and `EndTag` parameters to specify the tags that surround matching terms.

For example:

```
action=Query&Text=quickly jumping&Highlight=Terms&StartTag=<strong>&EndTag=</strong>
```

If this query matches a document with the content *the quick brown fox jumps over the lazy dog*, the IDOL Content component returns the following modified content in the query response:

```
the <strong>quick</strong> brown fox <strong>jumps</strong> over the lazy dog
```

For information about other available highlight options, refer to the *IDOL Server Reference*.

Summary, Sentences, and Characters

The `Summary` parameter allows you to automatically create a summary of result documents. The main options for `Summary` are **`Quick`**, **`Concept`**, and **`Context`**. A *quick* summary is just the first few sentences of the result document. A *concept* summary contains sentences that are typical of the content of the result document. A *context* summary contains sentences from the document that contain words and phrases that match the query text.

You can use the `Sentences` and `Characters` parameters to control the maximum length of a summary.

Cluster

The IDOL Content component can dynamically cluster the results of a query, grouping similar results together, by including a cluster id with each result document. To turn on dynamic clustering, set `Cluster` to **`True`**.

Query Response

This section describes some of the response tags that a query returns.

For example:

```
action=Query&Text=spiders&Combine=Simple&Print=all&MaxResults=2
```

This query might return the following ACI response:

```
<autnresponse xmlns:autn="http://schemas.autonomy.com/aci/">
   <action>QUERY</action>
   <response>SUCCESS</response>
   <responsedata>
      <autn:numhits>2</autn:numhits>
      <autn:hit>
         <autn:reference>430377369900437504</autn:reference>
         <autn:id>128534</autn:id>
         <autn:section>0</autn:section>
         <autn:weight>85.27</autn:weight>
         <autn:links>SPIDER</autn:links>
         <autn:database>Twitter</autn:database>
         <autn:content>
            <DOCUMENT>
               <DREREFERENCE>430377369900437504</DREREFERENCE>
               <AUTN_GROUP>Connector</AUTN_GROUP>
```

```
                <AUTN_IDENTIFIER>PGlkIHM9IlRXViIgcj0iNDMw</AUTN_IDENTIFIER>
                <CREATED_AT>Mon Feb 03 16:28:55 +0000 2014</CREATED_AT>
                <DOCTRACKINGID>256a3b295232c32c4a3ad4caa5c05d3f</DOCTRACKINGID>
                <DREDBNAME>Twitter</DREDBNAME>
                <DRECONTENT>If spiders can sit on the web all day then so can
I</DRECONTENT>
            </DOCUMENT>
         </autn:content>
      </autn:hit>
      <autn:hit>
         <autn:reference>430369534923374592</autn:reference>
         <autn:id>22023</autn:id>
         <autn:section>0</autn:section>
         <autn:weight>85.27</autn:weight>
         <autn:links>SPIDER</autn:links>
         <autn:database>Twitter</autn:database>
         <autn:content>
            <DOCUMENT>
               <DREREFERENCE>430369534923374592</DREREFERENCE>
               <AUTN_GROUP>Connector</AUTN_GROUP>
               <AUTN_IDENTIFIER>PGlkIHM9IlRXViIgcj0iNDM</AUTN_IDENTIFIER>
               <CREATED_AT>Mon Feb 03 15:57:47 +0000 2014</CREATED_AT>
               <DOCTRACKINGID>2a6e7bbc51001b815efa9b30acf6704e</DOCTRACKINGID>
               <DREDBNAME>Twitter</DREDBNAME>
               <DRECONTENT>eek! spiders!</DRECONTENT>
            </DOCUMENT>
         </autn:content>
      </autn:hit>
   </responsedata>
</autnresponse>
```

For a query action, the `<responsedata>` element contains the `<autn:numhits>` tag, which contains the number of results returned, and a series of `<autn:hit>` tags, which each contain a query result.

A query result consists of metadata for the result document, and the returned portions of the result document under the `<autn:content>` tag (in this example, each document is in a `<DOCUMENT>` subtree).

The important result document metadata tags are described in the following table.

`<autn:reference>`	The identifier for the document in the IDOL index.
`<autn:id>`	A unique number assigned to the document by the Content component where the document is indexed. The ID counts up from the first document indexed, and is unique only in a single Content component. In a distributed index, ID is not a reliable way to refer to a document in the index. As a best practice, always use references to refer to documents in IDOL Server.
`<autn:section>`	The section number of the result.

`<autn:weight>`	The relevancy score of the document to the query.
`<autn:links>`	The matching terms from the query that were found in the document.
`<autn:database>`	The database the result document came from.

The `<DOCUMENT>` subtree consists of field values for the document. The tag is the name of the document field, and the value of the tag is the value of the field in the document. For documents that you indexed in IDX format, this subtree is flat. For documents that you indexed in XML format, this subtree might have additional field structure, corresponding to the original XML.

Commonly, the reference for a document is stored in a field called `DREREFERENCE`, the content of a document is stored in a field called `DRECONTENT`, and the database name is stored in a field called `DREDBNAME`.

Chapter 11: Install IDOL with Docker Images

This chapter describes the Micro Focus IDOL Docker images, and how to use them to create and use simple IDOL example instances.

Introduction

Docker is a platform that allows you to run and use *containers*, which are packages that contain pre-installed and configured software. Containers provide convenient building blocks to allow you to create complex systems more easily.

The IDOL Docker containers allow you to quickly set up a simple IDOL installation, without having to manually install or configure the IDOL components.

> **NOTE:** This chapter assumes that you have installed and set up the Docker platform in your environment. It does not provide general information about how to use Docker.
>
> For more information about Docker, refer to the Docker documentation: https://docs.docker.com/.

The Micro Focus IDOL containers each provide a single IDOL component, with a standard configuration. Micro Focus also provides several Docker Compose files to allow you to link these containers together in simple, standard systems. You can use these containers and Compose files as a starting point for building your own containerized IDOL system.

Available IDOL Containers

The following table describes the available Micro Focus IDOL Docker containers. For more information about the IDOL components, refer to the relevant IDOL documentation.

Container Name	Description
content	IDOL Content component
community	IDOL Community component
agentstore	IDOL Agentstore component, configured to work with the IDOL Community component (for example, the community container)
category	IDOL Category component
categorisation-agentstore	IDOL Agentstore component configured to work with the IDOL Category Component and categorization tasks in IDOL NiFi Ingest (for example, the category, nifi-minimal, and nifi-full containers)
view	IDOL View component
dah	Distributed Action Handler
dih	Distributed Index Handler
nifi-minimal	IDOL NiFi Ingest, with only the IDOL NiFi processors (and not the standard Apache NiFi processors)
nifi-full	IDOL NiFi Ingest, with all the standard processors that are included in the Apache distribution of NiFi, in addition to the IDOL NiFi processors.
find	IDOL Find user interface
mediaserver	IDOL Media Server
mediaserver-enus	IDOL Media Server, configured with the speech packages for US English
mediaserver-playlistserver	IDOL Media Server, with a minimal configuration to serve video for MMAP.
mmap_app	IDOL Media Management and Analysis Platform application
siteadmin	IDOL Site Admin user interface
controller	IDOL Controller component
coordinator	IDOL Coordinator component
dataadmin	IDOL Data Admin user interface

Container Name	Description
statsserver	IDOL Statistics Server
qms	IDOL Query Manipulation Server
qms-agentstore	IDOL Agentstore component, configured to work with the IDOL Query Manipulation Server (for example, the qms container)
omnigroupserver	IDOL OmniGroup Server
eductionserver	IDOL Eduction Server
answerserver	IDOL Answer Server
answerbank-agentstore	IDOL Agentstore component, configured to work with Answer Bank systems in IDOL Answer Server
passageextractor-agentstore	IDOL Agentstore component, configured to work with Passage Extractor systems in IDOL Answer Server

IDOL Docker Compose Files

The IDOL Docker Compose files allow you to quickly set up a system that uses multiple containers, connected together to provide an IDOL setup.

There are several sets of Docker Compose files that you can use:

- `basic-idol`. Set up a simple IDOL system so that you can ingest some data and run queries.
- `data-admin`. Set up IDOL Data Admin and the IDOL components that it requires to run.
- `site-admin`. Set up a simple IDOL system so that you can ingest some data and run queries. This setup includes IDOL Site Admin to administer the IDOL services.

NOTE: The `basic-idol`, `data-admin`, and `site-admin` names are the folder names in the provided IDOL Docker Compose package. The examples in this chapter are based on these standard names. If you change the folder names, or modify the `COMPOSE_PROJECT_NAME` environment variable, it affects the container names that Docker produces, and you must adapt the example commands.

Basic IDOL Docker Compose

The basic-idol Docker Compose setup has a standard `docker-compose.yml` file that creates a small IDOL setup. This basic setup includes the following containers:

- `content`. The IDOL Content component, for indexing and query.
- `nifi-minimal`. A basic version of IDOL NiFi Ingest, configured with a File System Connector, to allow you to process and ingest files into your IDOL Content component.
- `categorization-agentstore`. An IDOL Agentstore component configured to work with categorization tasks in IDOL NiFi Ingest.
- `find`. The Find user interface, to allow you to view results from your Content index.
- `community`. The IDOL Community component, which Find uses to manage users.
- `agentstore`. An IDOL Agentstore component configured to store users and agent data for Community.
- `view`. The IDOL View component, which Find uses to display document previews.

When you use this setup, you can copy files into an internal volume. This volume is monitored by a File System Connector, running within NiFi. Any files that you copy to the volume are ingested and the resulting documents are indexed into the IDOL Content component. You can then log in to Find to send queries to find this data.

By default, this setup does not allow you to access the IDOL component ports directly, and does not enable SSL.

Micro Focus also provides several additional compose files to allow you to extend the basic setup. The following table describes these additional files.

Docker File	Description
`docker-compose.add-docsec.yml`	Adds document security for your IDOL documents. This option includes an OmniGroupServer to retrieve user and group information from an LDAP server.
`docker-compose.bindmount.yml`	Provides a bindmount. This option allows you to copy files to a physical directory for the File System Connector to ingest them, rather than using Docker copy.
`docker-compose.expose-ports.yml`	Exposes the IDOL component ports in the containers, so that you can connect to the components directly.
`docker-compose.ssl.yml`	Enables SSL communications for the basic IDOL components.
`docker-compose.add-mmap.yml`	Adds IDOL Media Server to allow you to process image, audio, and video files in addition to text-based content, and the IDOL MMAP application to allow you to perform more advanced analysis on the audio and video.
`docker-compose.add-mmap.ssl.yml`	Enables SSL communications for MMAP and Media Server.

IDOL Data Admin Docker Compose

The `data-admin` Docker Compose setup has a standard `docker-compose.yml` file that creates the required containers to run IDOL Data Admin.

You can optionally include `docker-compose.ssl.yml` to run IDOL Data Admin and the IDOL components with SSL communications enabled.

You can optionally use `docker-compose.expose-ports.yml`, if you want direct access to all of the component ports.

IDOL Site Admin Docker Compose

The `site-admin` Docker Compose setup has a standard `docker-compose.yml` file that creates a small IDOL setup, including IDOL Site Admin for administering the ACI servers.

You can optionally use `docker-compose.expose-ports.yml`, if you want direct access to all of the component ports.

Deploy IDOL Containers on Kubernetes

Kubernetes is a system for automating the deployment, scaling, and management of containerized applications. The IDOL containers repository on github provides files to allow you to run the IDOL containers by using Helm, which is a package manager for Kubernetes.

You can view and find the Kubernetes and Helm chart files at:

https://github.com/microfocus-idol/idol-containers-toolkit/tree/main/helm

For more information, view the README file: https://github.com/microfocus-idol/idol-containers-toolkit/blob/main/helm/README.MD.

Requirements

This section describes the requirements for you to run the basic IDOL Docker Compose set up.

Before you can use the IDOL Docker containers, you must have:

- an installation of Docker, on Windows or Linux. On Linux you must also install Docker Compose. For details about how to install and run Docker and Docker Compose, refer to the Docker documentation.
- an API key to access the Micro Focus IDOL Docker hub to download the container images. You can obtain this key from Micro Focus Support.

- an IDOL License Server, running on a static machine. The IDOL containers use this License Server to retrieve licenses.

The IDOL Docker Compose files are available from GitHub, in the idol-containers-toolkit repository.

Build IDOL Docker Compose System

This section describes how to build and start your IDOL Docker Compose system.

1. Obtain the Docker Compose files, for example:

   ```
   git clone https://github.com/microfocus-idol/idol-containers-toolkit
   ```

2. Modify the environment file to specify the License Server IP address, and a HTTP Proxy if required. See Modify the Environment Files, below.
3. (`basic-idol` only) Make any required updates to the YAML files for your particular options. You must modify the YAML when you want to use a bindmount, or document security. See Update YAML Files, on the next page.
4. Run the `docker-compose up` command to build and run the containers. See Run Docker Commands, on page 141.

You can also optionally set up SSL/TLS communications between the IDOL components in your containers. In this case, you must provide the required certificates, or a means to generate them. There are also additional environment variables to set. See Use SSL/TLS Communications, on page 142.

Modify the Environment Files

The IDOL Docker Compose ZIP package contains environment files.

For all IDOL Docker Compose options, you must modify the `.env` environment file to provide the details of your license server.

> **NOTE:** To use SSL, you must also modify the `idol-ssl.env` environment file to configure SSL behavior (see SSL Environment Variable Reference, on page 145).

To modify the standard environment file

1. In your IDOL Docker Compose package directory, open the folder for the setup that you want to run (`basic-idol` or `data-admin`).
2. Open the `.env` environment file in a text editor.
3. Update the `LICENSESERVER_IP` variable to give the IP address of your IDOL License Server.

> **NOTE:** To work with the standard IDOL Docker images, License Server must run on port 20000.

4. (Optional) Modify the following variables for your setup:

HTTP_ PROXY	The URL of a HTTP proxy that Docker must use to access the container files on Docker Hub.
IDOL_ SERVER_ VERSION	The IDOL Server version of the images that you want to download. You can use one of the following version string formats: • 12.*X* Download the latest version of the 12.X release containers. For example, 12.6 downloads the latest available 12.6 version (which might be 12.6.0 or 12.6.1, and so on). • 12.*X*.*X* Download the latest version of the 12.X.X release containers. For example, 12.6.0 downloads the latest available 12.6.0 version, where the docker containers have been updated but the IDOL component versions have not changed. • 12.*X*.*X_DockerBuild* Download an exact version of the IDOL Docker containers.

   The other environment variables are for internal use, and you do not need to update them unless instructed by Micro Focus Support.

5. Save and close the environment variables file.

Update YAML Files

> **NOTE:** This section applies only to the `basic-idol` Docker Compose setup.

In most cases, you do not need to modify the YAML files for the IDOL Docker Compose package. However, YAML updates are required when:

- you want to run the IDOL NiFi Ingest container with a bindmount. See Modify the YAML to Use a Bindmount, below.
- you want to use LDAP for Find login or document security. See Modify the YAML to Use Document Security, on the next page.

When you want to use these features, you must update these YAML files before you run the `docker-compose up` command

Modify the YAML to Use a Bindmount

To run the IDOL NiFi Ingest container with a bindmount, you must modify `docker-compose.bindmount.yml` to add the directory to use.

The bindmount means that you have a directory on the computer where you run your Docker containers that you can use to ingest data into the IDOL NiFi Ingest containers. In the default configuration, you upload documents to ingest by using a Docker copy.

To run with a bindmount

1. In your IDOL Docker Compose ZIP package, open the `basic-idol` folder.
2. Open `docker-compose.bindmount.yml` in a text editor.
3. Under `idol-ingest-volume`, update the device: property with the directory that you want to use for ingest. For example:

   ```
   volumes:
      idol-ingest-volume:
         driver_opts:
            type: none
            device: C:\docker\MyIngestDir
            o: bind
            driver: local
   ```

4. Save and close the YAML file.

Modify the YAML to Use Document Security

To run the IDOL Docker containers with document security, you must modify `docker-compose.add-docsec.yml` to add settings for the LDAP server that you want to use.

The document security setup uses an LDAP server to manage user and group details, and an IDOL OmniGroupServer to expose the users and groups to IDOL.

You can use this option to provide LDAP login for Find. You can also include the IDOL document security options that restrict access to documents. When a user logs into Find, IDOL generates a security string, which it uses whenever the user makes a query, to ensure they can only access permitted documents.

> **NOTE:** By default, the connector configurations in the IDOL Docker container do not include document security. You must configure the connectors after you set up the containers. For more details about how to configure the connectors in IDOL NiFi Ingest, refer to the IDOL NiFi Ingest documentation.

To run with document security

1. In your IDOL Docker Compose ZIP package directory, open the `basic-idol` folder.
2. Open `docker-compose.add-docsec.yml` in a text editor.
3. Under `x-args-security`, update the following parameters for your system:

- `LDAP_SERVER`. The IP address or host name of your LDAP server.
- `LDAP_PASSWORD`. The password for the Base DN user in LDAP (that is, the user that can access all documents in your server).

  > **IMPORTANT:** Encrypt this password by using the Micro Focus `Autpassword` command-line tool. This tool can generate the AES encryption keys, and the encrypted password strings.
  >
  > You supply the AES keys to OmniGroupServer by replacing the `basic-idol/omnigroupserver/aes.key` file in your IDOL Docker Compose ZIP package directory. Do not use the `aes.key` file provided; it is a placeholder file only.

```
x-args-security:
  # Put configuration details for the ldap server here
  - &ldap-server "LDAP_SERVER=myldap.example.com"
  ...
  # Don't put the LDAP password here in plain text:
  # encrypt your password via autpassword with omnigroupserver/aes.key
  - &ldap-password "LDAP_PASSWORD=qNvqIYaYxZyOEDrmz/gthg=="
```

4. To provide document security, uncomment (delete the #) for the following parameters:

```
- &document-security-type "DOCUMENT_SECURITY_TYPE=NT_V4"
- &document-security-type-mode "DOCUMENT_SECURITY_TYPE_MODE=AUTONOMY_SECURITY_
V4_NT_MAPPED"
- &document-security-type-propmatch "DOCUMENT_SECURITY_TYPE_PROPMATCH=nt_v4"
```

   Modify the values for your configuration. These parameters are used by the IDOL Content component and IDOL Community component for document security. For more information, refer to the *IDOL Content Component Reference* and *IDOL Community Component Reference*.

Run Docker Commands

After you have updated the environment file with your License Server details, and made any optional changes, you can run the containers.

To run the Docker Compose system

1. Open a command prompt.
2. Log in to docker by typing the following command:

```
docker login -u microfocusidolreadonly
```

   When prompted, type the API key provided by Micro Focus Support.

   This log in allows you to access the IDOL containers in the `microfocusidolserver` organization on Docker Hub.

3. In the command prompt, switch to the directory for the Docker Compose system that you want to run (for example `basic-idol`, `data-admin`, or `site-admin`).

> **NOTE:** Docker uses these folder names to create the names of the container instances. If you change the default `basic-idol`, `data-admin`, or `site-admin` folder names or modify the `COMPOSE_PROJECT_NAME` environment variable, you must adapt some of the examples in later sections of this chapter.

4. Run the `docker-compose up` command to build and run the containers:

   ```
   docker-compose up
   ```

   This option builds the simple system.

   To add any of the additional features, you can use the `-f` argument with the name of each YAML file that you want to use. For example, for `basic-idol`, the following command runs the basic system with a bindmount and exposes the component ports:

   ```
   docker-compose -f docker-compose.yml -f docker-compose.expose-ports.yml -f
   docker-compose.bindmount.yml up
   ```

   For details of the available YAML files, see IDOL Docker Compose Files, on page 135.

Check that the System is Running

After the up command is complete, you can check that your IDOL Docker system is running correctly by accessing one of the available user interfaces.

- For `basic-idol`: `http://`*`DockerHost`*`:8080/find/`
- For `data-admin`: `http://`*`DockerHost`*`:8080/dataadmin/`
- For `site-admin`: `http://`*`DockerHost`*`:8080`

Where *`DockerHost`* is the IP address or host name of the computer where you have run the docker containers.

Use SSL/TLS Communications

IDOL components and front end applications support SSL/TLS.

To configure the docker compose set up to use SSL/TLS

1. Modify the `idol-ssl.env` environment file to configure the SSL behavior.

   The `idol-ssl.env` file is the environment file for all the services that Docker Compose creates. This file configures the environment variable configurations that are common to all services. If required, you can also set environment variables for individual services. See SSL Environment Variable Reference, on page 145.

2. Modify the `docker-compose.ssl.yml` file to define how to set up the deployment with certificates.

You can provide these certificates (recommended), or you can provide an OpenSSL-based Certificate Authority to generate the certificates when you run the containers. See Set Up SSL/TLS Certificates and Trust Stores, below.

3. Modify the `docker-compose.ssl.yml` file to configure the `ssl-volume` bind volume, which provides the certificates from the host machine to the containers. By default this has the following configuration:

```
x-ssl-volume: &ssl-volume
   type: bind
   source: ../ssl/intermediate
   target: /ssl
```

   You must adjust the `source` path to the appropriate location of certificates for your system.

   NOTE: Micro Focus recommends that you leave the `target` path as `/ssl`, which is required for `idol-nifi` to retrieve certificates. In all cases, the `target` path must correspond to the directory that you set in the `IDOL_SSL_CA_MOUNTDIR` environment variable (see SSL Environment Variable Reference, on page 145).

4. Send the docker compose up command, including the `docker-compose.ssl.yml` file. For example:

```
docker-compose -f docker-compose.yml -f docker-compose.ssl.yml up
```

   For the `basic-idol` setup, if you also want to include MMAP, you must also add the `docker-compose.add-mmap.ssl.yml` file to the `up` command.

   NOTE: Between the `ssl-volume` and the environment variables that you provide to the containers, the containers must be able to either find or generate a certificate for the component. If it cannot obtain an appropriate certificate, the `docker-compose up` command exits with an error.

Set Up SSL/TLS Certificates and Trust Stores

When you want to run the IDOL Docker Compose packages with SSL, you must either provide the SSL certificates (recommended), or provide an OpenSSL-based Certificate Authority to generate the certificates when you run the containers.

NOTE: If you configure the containers to generate the certificates, you must provide a certificate for your certificate authority that is able to sign Certificate Signing Requests.

You use the `idol-ssl.env` environment variables file to provide information about your certificates. The `docker-compose.ssl.yml` defines an `ssl-volume` bind mount, which provides the certificates from the host machine to the containers.

Between `idol-ssl.env` and the `ssl-volume`, the containers must be able to either find a certificate provided for the component, or use the Certificate Authority certificate and configuration to generate one. If the containers cannot obtain an appropriate certificate by either method, the `docker-compose up` command exits with an error.

When you use a custom Certificate Authority to generate your certificates, you must also set up a trust store for your user interfaces, to allow them to securely communicate with the SSL-activated IDOL components. See User Interface Trust Stores, below.

The following sections describe how to provide or generate the certificates, and trust stores.

Provide Certificates

Micro Focus recommends that you provide certificates for the container.

When you start a container with SSL activated, the container looks for a certificate at a configurable path for the component. You configure this path by using the environment variables.

The expected location is:

```
${IDOL_SSL_CA_MOUNTDIR}/${USER_SSL_CERTS_DIR}/${IDOL_SSL_COMMON_NAME}.cert.pem
```

where:

- `${IDOL_SSL_COMMON_NAME}` is the common name for the service.
- `${IDOL_SSL_CA_MOUNTDIR}` and `${USER_SSL_CERTS_DIR}` are environment variable you can use to specify the location of the certificates.

For more information about these environment variables, see SSL Environment Variable Reference, on the next page.

Generate Certificates

When you do not provide a certificate, Docker generates them for the component when you start up the container for the first time. In this case, you must have a Certificate Authority that is able to sign Certificate Signing Requests.

You control the location of the Certificate Authority certificate, and the password for it, by using environment variables.

- The certificate is expected to be in the directory `${IDOL_SSL_SSL_CA_MOUNTDIR}/${IDOL_SSL_CACERT}`.
- The private key file is expected to be in the directory `${IDOL_SSL_SSL_CA_MOUNTDIR}/${IDOL_SSL_CAKEY}`.
- The password for the private key is expected to be set in the environment variable `${IDOL_SSL_CAPASS}`.

User Interface Trust Stores

When you use a custom Certificate Authority to generate certificates, you must set up trust stores for your user interfaces (Find, MMAP, and Data Admin), to allow them to securely communicate with the SSL-activated IDOL components.

You construct the trust store from a PKCS 12 file of the issuing certificate and its private key, along with the root certificate.

To construct the trust store

1. Create a `cfg` directory in the `${IDOL_SSL_CA_MOUNTDIR}` directory. The SSL generation scripts require this directory to store the certificate generation configuration.
2. Generate an intermediate PKCS 12 file. The following example command creates a file, `intermediate.pkcs12`:

   ```
   openssl pkcs12 -export -in certs/intermediate.cert.pem -inkey
   private/intermediate.key.pem -passin pass:PRIVATE_KEY_PASSWORD -out
   certs/intermediate.pkcs12 -passout pass:PASSWORD_FOR_PKCS12_FILE
   ```

   where *`PRIVATE_KEY_PASSWORD`* is the password for your private key, and *`PASSWORD_FOR_PKCS12_FILE`* is your password for the PKCS12 file.
3. Copy your root certificate (for example `ca.cert.pem`) to the intermediate `certs` directory to make it available to the trust store.
4. Generate the chain file by concatenating the root certificate PEM and the issuing intermediate certificate PEM. For example:

   ```
   cat certs/ca.cert.pem certs/intermediate.cert.pem > ca-chain.cert.pem
   ```
5. Choose the value of the `IDOL_SSL_SUBJ_ALT_NAME` environment variable.

   For a user interface (that is, something that you expect to access from a Web browser), use a wildcard value to generate a wildcard certificate. For example, if the computer hosting the certificate is accessible on the exampl.com domain, use `*example.com`. This option prevents warnings from your browser because of a difference between the domains that the certificate is valid for, compared with the domains that it is accessible on.

   For a back end component, the value that you choose is less important, because for this demo system the communications between the back end components and the user interfaces are not verified against the server they run on.
6. Import your chain certificate into your browser. This step prevents warnings about the issued certificates being untrusted.

SSL Environment Variable Reference

The following table lists the environment variables that are required for SSL/TLS communications to work in your docker environment.

To configure these values for all services, you set the environment variable in the `idol-ssl.env` file.

If required, you can also modify the environment variables for an individual service by creating an `environment` section in the `docker-compose.ssl.yml` file section for that service. However, in most cases the default values are suitable and this approach is not required.

IMPORTANT: The `idol-nifi` container has its own environment variable, `USE_SSL` to activate SSL, and it uses the `ssl-volume` bind mount to retrieve certificates. NiFi Ingest does not use the other environment variables listed here.

NiFi Ingest uses SSL only to communicate with your IDOL components, and does not use HTTPS to restrict access to the user interface.

Variable	Description
IDOL_SSL	You must set this value to activate SSL.
IDOL_SSL_SUBJ_ALT_NAME	The second DNS Name entry in Subject Alternative Name in the certificate. This value is service-specific.
IDOL_SSL_CACERT	The directory path to the issuing certificate for the CA, relative to IDOL_SSL_SSL_CA_MOUNTDIR. This value is specific to your CA setup.
IDOL_SSL_CAKEY	The directory path to the private key for the issuing certificate for the CA, relative to the IDOL_SSL_SSL_CA_MOUNTDIR. This value is specific to your CA setup.
IDOL_SSL_CAPASS	The password for IDOL_SSL_CAKEY. This value is specific to your CA setup.

The following table describes optional environment variables, which you might need to change to match any differences for your Certificate Authority setup.

Variable	Default	Description
IDOL_SSL_CA_MOUNTDIR	/ssl	The directory where the bind mount for the SSL Certificate Authority is mounted on the container. This value is container-specific.
USER_SSL_CERTS_DIR	certs	The directory path where the containers can generate and find certificates, relative to IDOL_SSL_CA_MOUNTDIR. This value is specific to your CA setup.
IDOL_SSL_COMMON_NAME	idol-${IDOL_COMPONENT}	The COMMON_NAME for the certificate. Docker populates the value of ${IDOL_COMPONENT} internally when it builds the component container. This value is also the first DNS Name entry in Subject Alternative Name, for example idol-content.
USER_SSL_CRL_DIR	crl	The directory path to use to store CRL information, relative to IDOL_SSL_CA_MOUNTDIR. This value is specific to your CA setup.
USER_SSL_NEWCERTS_DIR	newcerts	The directory path to use to generate newcerts, relative to IDOL_SSL_CA_MOUNTDIR. This value is less important than USER_SSL_CERTS_DIR, and is specific to your CA setup.
USER_SSL_DATABASE_FILE	index.txt	The file name of the database of issued certificates for your Certificate Authority. This value is specific to your CA setup.
USER_SSL_	serial	The file name of the serial file for your

Variable	Default	Description
SERIAL_FILE		Certificate Authority. This value is specific to your CA setup.
USER_SSL_ RANDFILE	private/.rand	The directory path to the randfile for your Certificate Authority, relative to IDOL_SSL_CA_ MOUNTDIR. This value is specific to your CA setup.
USER_SSL_ CRLNUMBER_ FILE	crlnumber	The file name of the CRLNumber file for your Certificate Authority. This value is specific to your CA setup.
USER_SSL_ CRL_FILE	crl/intermediate.crl.pem	The directory path to the CRL file for your Certificate Authority, relative to IDOL_SSL_CA_ MOUNTDIR. This value is specific to your CA setup.
USER_ ISSUING_CA_ PKCS12	intermediate.pkcs12	The directory path to the PKCS 12 file of the issuing certificate and private key, relative to IDOL_SSL_CA_MOUNTDIR/USER_SSL_CERTS_DIR. This value is specific to your CA setup.
ROOT_ CERTIFICATE	ca.cert.pem	The directory path to a copy of the root certificate PEM file, relative to IDOL_SSL_CA_ MOUNTDIR/USER_SSL_CERTS_DIR. This value is specific to your CA setup.
CA_CHAIN_ FILE	ca-chain.cert.pem	The directory path to a chain PEM file for the issuing and root certificates, relative to IDOL_ SSL_CA_MOUNTDIR/USER_SSL_CERTS_DIR.

Use the Basic IDOL Docker Setup

This section describes how to use the `basic-idol` Docker Compose setup to index and search files, and how to access the Find and IDOL NiFi Ingest user interfaces.

Index Documents

After you have built your IDOL Docker Compose system, you can index documents.

Index Documents with Docker Copy

In the default IDOL Docker Compose setup, IDOL NiFi Ingest has an internal volume that you can use to index documents. You must transfer documents to this directory by using the docker cp command.

To index a document by using docker copy

1. Open a command prompt.
2. Run the docker cp command to transfer a file to the `basic-idol_idol-nifi_1:/idol-ingest/` volume. For example:

   ```
   docker cp example.pdf basic-idol_idol-nifi_1:/idol-ingest/
   ```

   Where your setup include Media Server, you can use the same process to ingest media files. For example:

   ```
   docker cp example.mp4 basic-idol_idol-nifi_1:/idol-ingest/
   ```

After you have run the copy, IDOL NiFi Ingest retrieves the documents from this directory and ingests them into your system.

Index Documents with a Bindmount

When you use a bindmount in your IDOL Docker Compose setup, you can copy files directly to a directory on the computer. IDOL NiFi Ingest retrieves them from this directory and ingests them into your system.

Access User Interfaces

The IDOL user interfaces available in the `basic-idol` IDOL Docker Compose system allow you to view your ingest stream, and search your documents. Where your setup includes Media Server and MMAP, you can also use the MMAP video player to view the indexed video.

Use IDOL NiFi Ingest

IDOL NiFi Ingest is a user interface that allows you to set up an ingest process. It provides connector processors that can retrieve documents from many different repositories, extract the useful content, and index it into IDOL.

In the IDOL Docker Compose system, you can access it at the following URL:

```
http://DockerHost:8080/nifi/
```

Where *`DockerHost`* is the IP address or host name of the computer where you have run the docker containers.

The IDOL NiFi Ingest user interface allows you to view and modify the configured ingestion chain, and monitor the process.

In the basic system, the default ingest process indexes documents from the internal ingest volume or bindmount. It also performs Eduction to extract personally identifiable information (PII) from the documents that you index.

When you use Media Server as well, it has additional processes to detect media files, and route them to Media Server for analysis, including Speech to Text for video files.

For information about how to modify the ingestion process in the NiFi setup, refer to the IDOL NiFi Ingest documentation.

Use IDOL Find

IDOL Find is a search user interface that allows you to search and view documents. It also provides some graphics and text analytics, such as topic maps and sunburst diagrams to provide more insight into your search results.

In the IDOL Docker Compose system, you can access it at the following URL:

`http://DockerHost:8080/find/`

Where *`DockerHost`* is the IP address or host name of the computer where you have run the docker containers.

In the default setup, you can log in to Find by using the default login details, username **admin** and password **admin**.

> **NOTE:** Find uses the IDOL Community component to store user details. For a more advanced system with multiple users, you can use the document security setup to add an LDAP server. In this case, you can log in to Find by using the user and password for a valid LDAP user. In this case, all users have the basic Find role, rather than the admin role, so the interface has different available features.
>
> You can also add users to Community directly. In this case, you must run the `docker-compose up` command with the component ports exposed, by using the `docker-compose.expose-ports.yml` option. You can then manually add users to Community.

After you log in you can search for documents and view the graphs and options. When you use Media Server, Find can return your media files as well, and uses the MMAP video player to provide more advanced video playback and search options.

Use the Data Admin Docker Setup

This section describes how to use the `data-admin` Docker Compose setup.

The `data-admin` setup provides all the IDOL components required to run IDOL Data Admin. However, the setup does not provide a built-in way to add any additional components or information (for example, it does not provide the ability to index documents into Passage Extractor).

Access the IDOL Data Admin User Interface

The IDOL Data Admin user interface allows you to access and use IDOL search optimization tools such as promotions, and to use IDOL Answer Bank to create and maintain a set of questions and answers for natural language question answering.

In the IDOL Docker Compose system, you can access it at the following URL:

`http://DockerHost:8080/dataadmin/`

Where `DockerHost` is the IP address or host name of the computer where you have run the docker containers.

You can log in to IDOL Data Admin by using the default login details, username **admin** and password **admin**.

Use the Site Admin Docker Setup

This section describes how to use the `site-admin` Docker Compose setup.

The `site-admin` setup does not include a reverse proxy, so the NiFi, Find, and Site Admin interfaces are accessed on different ports as described in the following table.

Interface	URL	Default crendentials
Site Admin	`http://DockerHost:8080`	Username **useradmin**, password **useradmin**
NiFi	`http://DockerHost:8081/nifi/`	
Find	`http://DockerHost:8000/find/`	Username **admin**, password **admin**

You can ingest documents by copying files into a Docker volume, in a similar way to the Basic IDOL Docker Setup. These documents will be detected and ingested by the File System Connector that is running within NiFi. For example:

```
docker cp example.pdf site-admin_idol-nifi_1:/idol-ingest/
```

Troubleshoot Common Problems

This section provides some information about some common problems that might occur when you build and run your Docker Compose setup.

On Windows, IDOL NiFi Ingest or MMAP volumes are not created

During the up command, you might get Windows prompts to allow file sharing access to particular directories. Access to these directories is required for the volumes that IDOL NiFi Ingest and MMAP use.

If you have this issue, re-run the `up` command and ensure that you click these file sharing prompts to allow access.

Applications are very slow or unresponsive

You might need to increase the amount of memory available to your applications, by using the Docker settings.

Customize the IDOL Docker Setup

The IDOL Docker Compose files provide an example system to demonstrate one way to run IDOL in containers. In most cases, to run a containerized IDOL in a production environment you must customize the setup.

The scope of this document does not extend to the kind of advanced customizations that you might want to make.

For general information about docker containers, refer to the Docker documentation.

If you require further assistance modifying your IDOL Docker container environments, contact Micro Focus Support.

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 API
The programming interface for interacting with IDOL ACI Servers. Several IDOL SDKs are available to allow you to develop applications with the ACI API.

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.

Agentstore
An IDOL component that indexes, collects, manipulates, and stores agent and category information. Agentstore is a special configuration of the Content component.

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.

Controller
An IDOL component that monitors IDOL services on a host machine, and communicates with Coordinator to report status information. This component is primarily used by the IDOL Site Admin administrative front-end application.

Coordinator
An IDOL component that manages status information from Controller components and acts as a central point for viewing logs and monitoring the status of the IDOL system. This component is primarily used by the IDOL Site Admin administrative front-end application.

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.

F

fields
The parts of a document that store different sections of the content.

Find
A basic search user interface for IDOL.

G

graph
A set of relationships that describes the connections in a set of data. The IDOL Knowledge Graph component uses graphs to explore connections in IDOL documents.

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.

IDOL SDK
Software Development Kits that allow you to develop applications that use the ACI API to use IDOL component ACI Servers.

index
The store of content in IDOL Server.

indexing
The process of adding content to IDOL Server.

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.

Knowledge Graph
An IDOL component that uses connections between IDOL documents to create a set of relationships called a graph, and allows you to explore these connections in your data.

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

mapped security
A method for configuring document security to ensure that only permitted users can access a document. In mapped security, IDOL compares user security details against an access control list (ACL), which is stored in the document.

Media Server
An IDOL component that analyzes video files and streams, image files, and audio to extract information about their content. Media Server can run analysis operations such as face recognition, number plate recognition, speech-to-text, and speaker identification.

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.

Q

QMS (Query Manipulation Server)
An IDOL component that manages promotions, modifies queries to IDOL Server, and manipulates results from IDOL Server.

query
A search to retrieve documents from IDOL Server.

S

stemming
The process of extracting the morphological root of a word. In languages, some words have a common morphological root. IDOL provides stemming algorithms that reduce words to this form. This process allows IDOL Server to match concepts regardless of the grammatical use of words. In English, for example, the words 'help', 'helpful', 'helping', and 'helped' all reduce to their stem 'help' without significant loss of meaning. IDOL provides as standard a set of stemming algorithms for the most commonly used languages. IDOL Server applies stemming after stop words have been discarded, both at index time (when content is stored in IDOL Server), and at query time (query text is stopped and stemmed before it is matched).

stop word
A very common word that occurs too frequently to be useful for searching. Stop words include articles (for example, the) and prepositions (for example, to or from). Stop words are language-specific. You can use a stop word list in IDOL Server to allow it to discard these words at index and query time to save index space and improve retrieval performance.

T

tokenization
The process of splitting a query string up into individual search terms.

U

unmapped security
A method for configuring document security to ensure that only permitted users can access a document. In unmapped security, IDOL connects to the original repository to check the user permissions for a document. This method is usually a lot slower than mapped security.

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 Micro Focus IDOL 12.13 Getting Started Guide

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!