Documentation Library

aXes-Cloud

Administrator Guide

Contents summary

Contents

List of Figures

List of Tables

Welcome to aXes-Cloud for administrators

aXes-Cloud is software that gives you browser access to new and existing IBM i (System i, iSeries or AS/400) applications without you having to install any new software on your IBM i servers and without having to change any of your existing applications. aXes-Cloud installs on one IBM i server and users can reach applications and operating system services on any other IBM i servers.

Users of aXes services (Terminal Server (TS), Data Explorer (DE) and Web Spooled File Manager (WSFM)) will not be aware of aXes-Cloud.

Administrator tasks are installing aXes-Cloud, configuring aXes-Cloud and services provided by Terminal Server, Data Explorer and Web Spooled File Manager.

This guide explains aXes-Cloud for an administrator, describing how to plan a deployment, configure the deployment, and manage aXes-Cloud.

Road Map

Use the road map to find information about aXes-Cloud.

If you want to…	Refer to…
What is aXes-Cloud?	Section 1 – What is aXes-Cloud? This section provides an overview of aXes-Cloud and explains the deployment options.
Administrator activities and responsibilities	Section 2 – Administration This section explains administrative concepts, responsibilities and tasks.
Managing day-to-day activities and troubleshooting	Section 3 – Housekeeping and trouble shooting This section explains administrator tasks such as archiving logs and tracking down problems.
Set values for configuration items and parameters	Section 4 – Configuration This section explains how to configure aXes-Cloud by setting values for the configuration items and parameters.
Read reference information	Appendices

For information about installing aXes refer to the aXes Quick Start and Reference guides.

aXes-Cloud administrators will need to be familiar with the administrator guides for Data Explorer and Web Spooled File Manager.

Section 1 – What is aXes-Cloud?

This section provides an overview of aXes-Cloud, describing its architecture, deployment options and issues to consider when planning an installation of aXes-Cloud.

Architecture

aXes-Cloud is a licensed extension to aXes that enables companies to use aXes features on multiple IBM i servers connected to an aXes installation on one IBM i server. Figure 1 (page 9) illustrates the architecture of aXes-Cloud and shows the components of a deployment:

- Client computers are desktop, laptop or mobile devices equipped with a browser.
- aXes-Cloud gateway server is an IBM i server where aXes-Cloud resides.
- Corporate IBM i application servers are the servers connected to aXes-Cloud.

The networks are intranets and/or the Internet. The aXes-Cloud server acts as a gateway between users and applications on corporate servers.

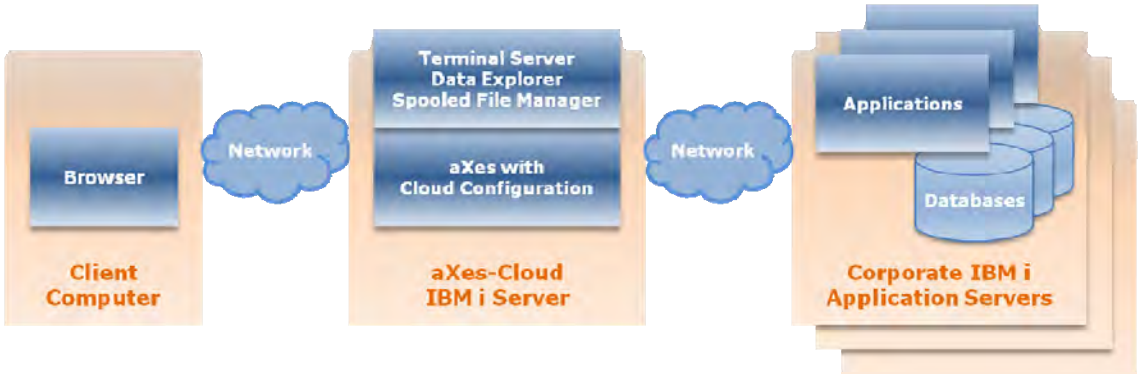


Figure 1: aXes-Cloud Architecture

In deployments of aXes without cloud extensions, aXes resides on the same server as the applications. Deployments of aXes-Cloud assume a minimum of two IBM i servers. One server is used for aXes-Cloud (the cloud gateway) and applications reside on the second corporate server. aXes-Cloud connects to one or multiple corporate servers.

aXes in a cloud deployment operates as if aXes were installed on every connected corporate server. Pages displayed by a browser will be HTML and JavaScript equivalents of 5250 screens from corporate servers where applications reside.

Deployment options for aXes-Cloud

aXes-Cloud offers several deployment options designed to satisfy requirements of small, medium and large companies. aXes-Cloud supports hosting service providers managing corporate servers on behalf of several customers. Topics in this section are as follows.

- Cloud gateway server
- Private cloud deployments
- Public cloud deployments
- SaaS cloud deployments for ISVs

Cloud gateway server

A typical deployment of aXes-Cloud requires a dedicated IBM i server (known as the cloud gateway server) and the aXes-Cloud software. The server may reside in the DMZ or on an internal network. In cases where users require access from outside a corporate network, the cloud gateway server must be accessible from the Internet.

You should employ appropriate security measures to protect the cloud gateway server and enforce access control and authentication measures for people who will use aXes-Cloud.

Private cloud deployments

An aXes private cloud deployment is aXes-Cloud installed and operated by one company. Typically, companies will operate multiple corporate servers but aXes-Cloud also operates with only one connected corporate server. Figure 2 (page 10) illustrates a private cloud deployment. In this example, a company has two branch offices and operates a corporate server in each branch office.

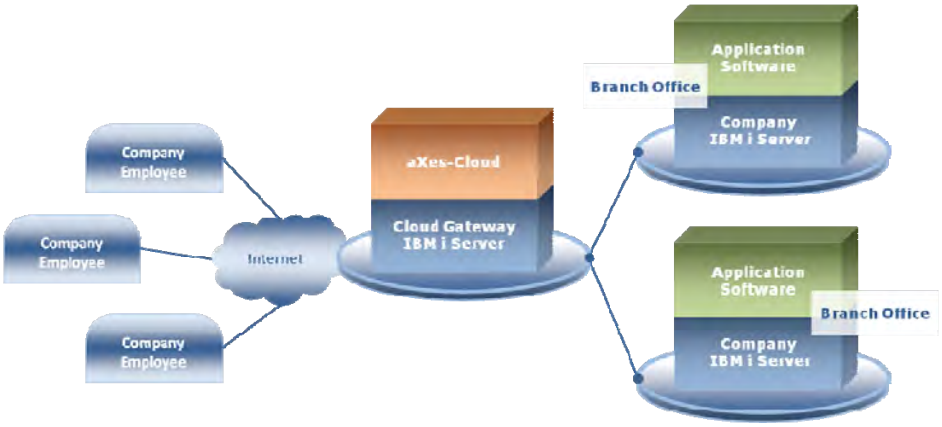


Figure 2: aXes-Cloud: Private Cloud Deployment

The deployment consists of aXes-Cloud on a cloud gateway server and the connected corporate servers. Users work with applications on corporate servers through aXes-Cloud.

The deployment shown in Figure 2 (page 10) assumes that all users share the same instance of aXes-Cloud. An alternate deployment is a dedicated instance of aXes-Cloud for each corporate server (that is, an aXes-Cloud instance for each branch office).

Companies can use an aXes private cloud as way to migrate applications from an older corporate server to a new corporate server. aXes-Cloud provides access to applications remaining on the older corporate server during the migration. Retire the older server when migration is complete. aXes-Cloud will then provide browser access to 5250 applications operating on the new corporate server.

Public cloud deployments

An aXes public cloud deployment is aXes-Cloud operated by a hosting service provider on behalf of multiple companies each with one or more corporate servers. Company personnel, customers and suppliers use applications from their respective corporate servers by logging into aXes-Cloud which routes them to their connected corporate servers.

Hosting service providers can deploy one or more instances of aXes-Cloud. Each hosted company may have a dedicated instance of aXes-Cloud or share the same instance of aXes-Cloud. Figure 3 (page 11) illustrates a deployment where each company operates in a dedicated instance of aXes-Cloud.

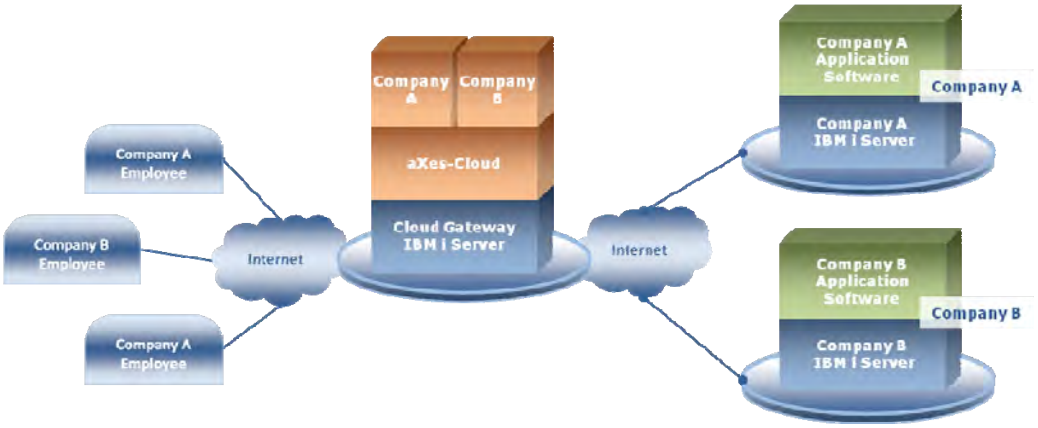


Figure 3: aXes-Cloud Configured with Instances Dedicated to Each Company

Use a dedicated instance of aXes-Cloud for each company to ensure the separation of companies sharing a cloud gateway server. Apart from company separation, the advantages of this deployment are the ability to customise sign on pages and enhancements to applications using aXes-eXtensions.

SaaS cloud deployments for ISVs

aXes-Cloud provides independent software vendors (ISVs) with tools to offer their packaged application software to their customers using a Software-as-a-Service (SaaS) model. An ISV can install aXes-Cloud on a cloud gateway server and instances of their application software on one or more connected corporate servers.

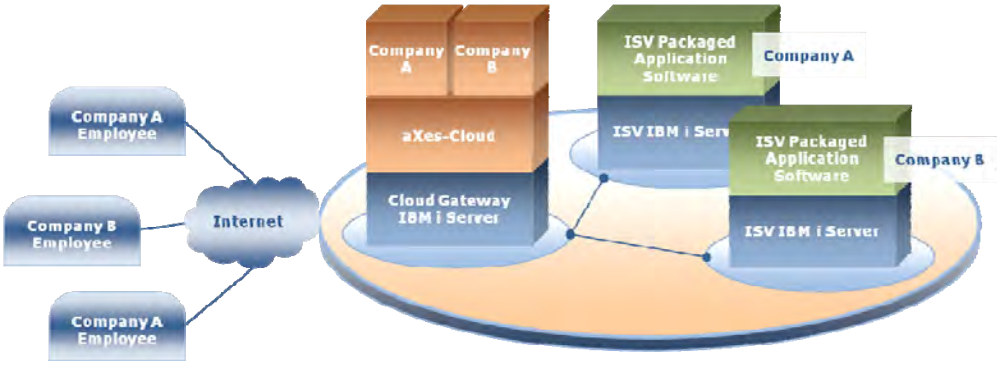


Figure 4: aXes-Cloud: ISV Deployment for SaaS

The ISV's customers sign on to the cloud gateway server and are connected to the appropriate server running their instance of the application software.

ISVs install an instance of aXes-Cloud for each of their customers on the cloud gateway server. This configuration provides ISVs with the optimum separation of customer environments and facilitates customisation of graphics and colour schemes for individual customers.

Planning a deployment

The things to consider when planning an aXes-Cloud deployment are:

- Single or multiple instances of aXes-Cloud
- Instance naming conventions
- IP addresses and port numbers
- Sub-systems, devices and queues
- Customising sign on pages

aXes-Cloud software versions

Whether you deploy one instance or multiple instances of aXes-Cloud, all instances share the same version of the aXes-Cloud software.

The cloud gateway server allows only one installation of a specific version of the aXes-Cloud software. Installing hot fixes or upgrading to a new software version will apply to all instances of aXes-Cloud on the cloud gateway server.

Deploying single or multiple instances of aXes-Cloud

aXes-Cloud supports both single and multiple instances of an aXes cloud and configuration and there are advantages and disadvantages for each deployment model.

The recommended deployment model for best separation is an instance for each connected server.

Table 1 (page 12) presents the characteristics of the single instance deployment model.

Table 1: Characteristics of Single Instance Deployments of aXes-Cloud

Characteristics
All users share the same aXes-Cloud instance
The instance supports one set of aXes-eXtensions and all applications share these eXtensions.
aXes-eXtensions apply to applications on any connected corporate servers.
The instance has one URL/URI.

Table 2 (page 12) presents the characteristics of the multiple instance deployment model.

Table 2: Characteristics of Multiple Instance Deployments of aXes-Cloud

Characteristics
Each instance has its own URL/URI which separates the users of each connected corporate server.
Each instance supports its own set of aXes-eXtensions.
aXes-eXtensions apply to applications on servers connected to the instance (rather than any connected corporate servers).

The multiple-instance deployment model offers the most flexibility for configuration and customisation.

Examples of the advantages of separation are dedicated sub-systems ensure that all instance jobs run in the sub-system assigned to the instance and each instance may have a customised sign on screen (QSIGNON). Separate sub-systems allow administrators to shut down an instance without affecting other instances.

For hosting service providers, separate instances of aXes-Cloud insulate companies using the cloud gateway from each other. The insulation provides operational separation. Administrators will be able to close operations for one company without influencing or compromising the operations of other companies sharing the cloud gateway.

Implementing a naming convention that associates the resources with an aXes-Cloud instance will assist administrators when looking for queues, sub-systems or devices on work with administration screens. For example add a prefix to the names of resources associated with each instance.

Planning for the aXes-Cloud gateway server

Table 3 (page 13) describes the policies for assigning IP addresses and ports for a cloud gateway server.

Table 3: Planning IP Addresses for an aXes-Cloud Gateway Server

Rules
A cloud gateway server can have multiple IP addresses.
A cloud gateway server can have multiple LPARs with separate IP addresses.
A cloud gateway server can share an IP address by using different ports with the IP address.

Planning for subsystems, devices and queues

Deployments of aXes-Cloud require the following resources (Table 4, page 13.

Table 4: Resources required for aXes-Cloud Deployments

Resources	
aXes-Cloud instance	aXes-Cloud requires a minimum of one instance. Multiple instances are copies of an original instance with additional configuration and customisation. One example of a deployment is an instance for each connected corporate server. Individual instances provide the most flexible configuration and customisation opportunities.
aXes-Cloud software	Mandatory. Shared by all instances of aXes-Cloud.
Job queue	Recommended for each instance
Message queue	Recommended for each instance
Output queue	Recommended for each instance
Sub-system	Recommended for each instance
User profile	Mandatory for each instance
Virtual devices	Recommended for each instance. Using a unique naming convention will assist administrators to identify devices associated with an aXes-Cloud instance.

Instances of aXes-Cloud may share some or all of the resources. However, for the optimum separation and insulation of aXes-Cloud instances, it is best not to share any of the resources (except for the aXes-Cloud software).

aXes Terminal Server as a cloud service

This section explains the concepts for configuring aXes Terminal Server for use with aXes-Cloud.

How users access aXes-Cloud

Users access aXes-Cloud from their browser. They operate a supported browser on their computer or mobile device and access aXes-Cloud by typing a URL in the address bar and pressing the Go button (or its equivalent).

After a successful authentication with the cloud gateway, aXes-Cloud connects the user to a corporate server.

Accessing applications on corporate servers

Applications on corporate servers

Users can access their applications on corporate servers after they sign on. Once a user has signed on, their interaction with the applications is the same as if they had signed on directly to the corporate server from their local network.

Applications on the aXes-Cloud gateway server

Users can access applications on the gateway server, provided that the aXes-Cloud configuration includes an instance that points to the cloud gateway server as a corporate server.

Responsibility for user authentication on connected servers

Authentication and authorisation for connected corporate servers is the responsibility of corporate server administrators. Users may sign on automatically or be asked to present their credentials to sign on. The choice of how users will sign on to corporate servers is independent of aXes-Cloud.

Sign on and authentication options

User authentication

When using aXes-Cloud, users sign on to two servers. The first sign on is to the cloud gateway server, where aXes-Cloud resides, and the second sign on is to a corporate server. Therefore, there are multiple options for how users may sign on. One option is an explicit sign on to aXes-Cloud and an explicit sign on to a corporate server. A second option is an implicit sign on to aXes-Cloud and an explicit sign on to a corporate server.

Implicit sign on to both aXes-Cloud and a corporate server is not recommended in networks accessible from the Internet.

User credentials for aXes-Cloud do not need to be the same as user credentials for corporate servers.

Explicit sign on

Figure 5 (page 15) shows the steps in the sign on process for an explicit sign on to aXes-Cloud and an explicit sign to a corporate server. Table 5 (page 15) explains the steps in the process and step numbers in Figure 5 correspond to step numbers in Table 5.

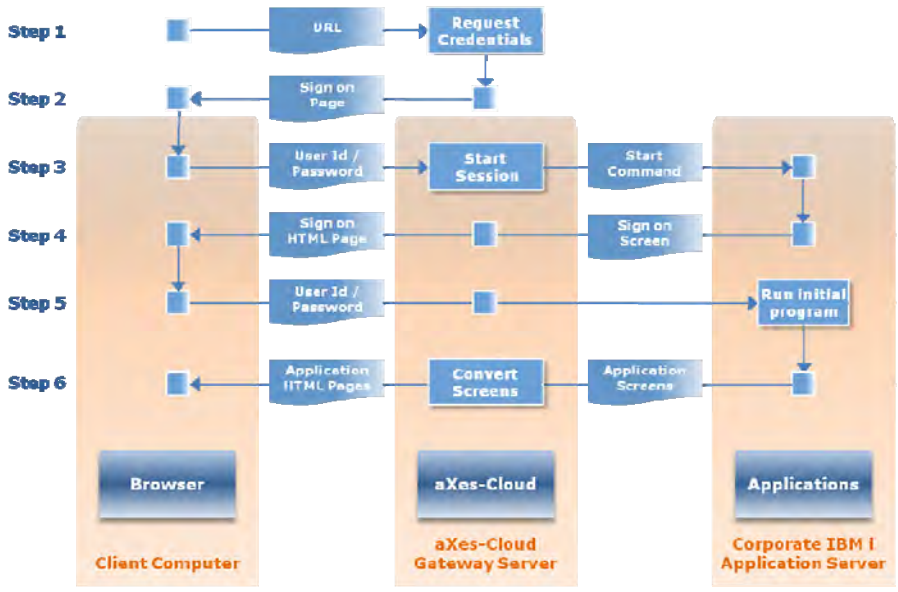


Figure 5: Explicit Sign on for both aXes-Cloud and a Corporate Server

Table 5 (page 15) explains what happens at each step.

Table 5: Explicit aXes-Cloud and Corporate Server Sign On

Steps	What Happens
1.	To start the sign on process a person types the URL for aXes-Cloud in a browser address bar and presses go.
2.	When the cloud gateway server receives the request it sends back a sign on page asking the person for credentials (user identification and password).
3.	The person types a user identification and password and presses a sign on button to send the credentials to the cloud gateway server. When the cloud gateway server receives the request it signs the person on to the cloud gateway server and starts a session with a corporate server.
4.	The corporate server sends back a sign on screen. aXes-Cloud prepares the HTML sign on page from the sign on screen and returns the page to the browser. The browser receives the response and displays the page to the person.
5.	The person types a user identification and password for the corporate server and presses the sign on button. aXes-Cloud passes the credentials to the corporate server. The corporate server signs the person on and runs the associated initial program or menu.
6.	Applications running on the corporate server generate screens from which aXes-Cloud creates equivalent pages consisting of HTML and JavaScript and sends the pages to the browser.

The last step operates while the person remains signed on to a corporate server. aXes-Cloud terminates the session between the cloud gateway and the corporate server when the person signs off.

Implicit sign on

Figure 6 (page 16) shows the steps in the sign on process for an implicit sign on to aXes-Cloud and an explicit sign to a corporate server. Table 6 (page 16) explains the steps in the process and step numbers in Figure 6 correspond to step numbers in Table 6.

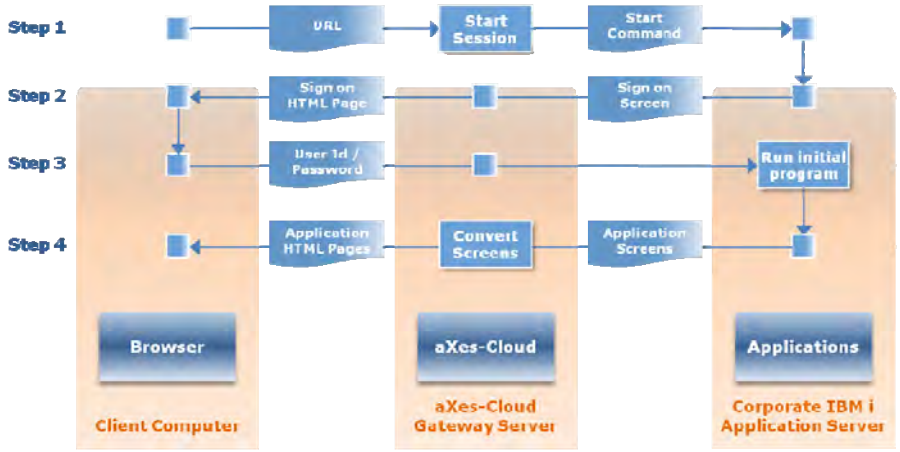


Figure 6: Implicit Sign On for aXes-Cloud and Explicit Sign On for a Corporate Server

Table 6 (page 16) explains what happens at each step.

Table 6: Implicit aXes-Cloud and Explicit Corporate Server Sign On

Steps	What Happens
1.	To start the sign on process a person types the URL for aXes-Cloud in a browser address bar and presses go. When the cloud gateway server receives the request it signs the person on to the cloud gateway server and starts a session with the corporate server.
2.	The corporate server sends back a sign on screen. aXes-Cloud prepares the HTML sign on page from the sign on screen and returns the page to the browser. The browser receives the response and displays the page to the person.
3.	The person types a user identification and password for the corporate server and presses the sign on button. aXes passes the credentials to the corporate server. The corporate server signs the person on and runs the associated initial program or menu.
4.	Applications running on the corporate server generate screens from which aXes creates equivalent pages consisting of HTML and JavaScript and sends the pages to the browser.

The last step operates while the person remains signed on to a corporate server. aXes-Cloud terminates the session between the cloud gateway and the corporate server when the person signs off.

Configuring connections to corporate servers

When a user signs on to the cloud gateway (explicitly or implicitly) an exit program runs. The exit program starts a Telnet session with a connected corporate server. Typically, the next page presented to the user is the sign on screen from the connected corporate server. The advantage of this approach is that no change is necessary at the connected corporate server. Users sign on and operate their sessions as if they had signed on without the aXes-Cloud.

If users wish to connect to multiple corporate servers, they can open a browser page for each server to which they wish to connect.

Using aXes-eXtensions with aXes-Cloud

aXes-Cloud comes with all of the features of aXes including aXes-eXtensions. Developers use aXes-eXtensions tools to modernise and enhance 5250 applications.

Table 7 (page 17) presents policies for using aXes-eXtensions with aXes-Cloud.

Table 7: Policies for using aXes-eXtensions with aXes-Cloud

Rules
The aXes-eXtensions projects must reside on the cloud gateway server.
Developers work on the projects on the cloud gateway server.
Nothing needs to be installed on the corporate servers that support the applications.

aXes-Cloud applies the enhancements built with aXes-eXtensions to the applications in real time from the cloud gateway server.

User authorisation for DE and WSFM

Define users who are allowed or denied access to aXes-Cloud

Users of DE and WSFM must be authorised to use aXes-Cloud. This authorisation does not replace the user profile on connected corporate servers.

The configuration parameters, service.user.allow and service.user.deny, determine the authorisation of a user to sign on. The service.user.allow parameter authorises users to sign on and the service.user.deny parameter denies sign on authorisation. To determine whether to authorise a user, aXes-Cloud examines the service.user.deny parameters first to remove users that should not be authorised and then examines the service.user.allow parameters.

User identifications beginning with the letter Q

aXes-Cloud prohibits users logging on with a user identification that begins with the letter Q as its default policy. If you wish to override this policy and allow the use of these user identifications, you must configure service.user.allow parameters for any user identifications, beginning with the letter Q, you wish to allow.

Qualified user identification

aXes-Cloud allows users to specify a server name at sign on and when they do, the sign on process establishes a connection with the nominated server automatically.

The syntax of a qualified user identification is server name and user.

[Server Name]/[User Identification]

An example of a qualified user identification is,

MyServer/MyUserId

The user types server name and user identification into the user identification on the sign on page. For example, typing MyServer/MyUserIdentification signs the user on to the server named MyServer. The server name is optional and users who do not name a server will sign on to their default server (as defined in the configuration).

The server name is the corporate server (or service host) where DE queries will run and where WSFM will manage spooled files.

Administrators define the names of corporate servers available for access in this way using the database.host, database.library and service.host configuration parameters.

Qualified user identification is available for aXes Data Explorer and aXes Web Spooled File Manager but not aXes Terminal Server.

How to deploy DE and WSFM databases

This section describes options for deploying DE and WSFM databases, policies governing deployments, and how to secure access to the databases.

About DE and WSFM databases

DE and WSFM store information in discrete databases, one for DE and another for WSFM. The databases hold information about queries, searches, folders and options associated with individual users. The parameters, database.host and database.library, determine where the databases reside. The service.host parameter and the server name from the user sign on (browser host) determine the server where DE queries and WSFM searches will run. The database host and the service host can be the same server or different servers.

aXes-Cloud offers flexible options for deploying single orf multiple instances of DE and WSFM databases on the cloud gateway server or connected corporate servers.

- You can have one database in a library on one server and run DE queries and WSFM searches on each corporate server.
- You can have databases and run DE queries and WSFM searches on each connected corporate server.
- You can have multiple databases in libraries on one corporate server and run DE queries and WSFM searches on each connected corporate server.

Policies for deploying DE and WSFM databases

The following tables define policies for deploying databases belonging to DE and WSFM.

Table 8 (page 18) presents policies for deploying DE databases.

Table 8: Policies for Deploying DE Databases

Policies for deploying DE databases
The database resides on one server.
The server can be the cloud gateway server or one of the connected corporate servers.
The database may have multiple instances, where each instance represents a corporate server. Multiple instances reside in different libraries on the same server.
The DE database may reside, but does not need to reside, on the same server as the WSFM database.

Table 9 (page 18) presents policies for deploying the WSFM database.

Table 9: Policies for Deploying WSFM Databases

Policies for deploying WSFM databases
The database resides on one server.
The server can be the cloud gateway server or one of the connected corporate servers.
The database may have multiple instances, where each instance represents a corporate server. Multiple instances reside in different libraries on the same server.
The WSFM database may reside, but does not need to reside, on the same server as the DE database.

Policies for deriving values for database and service host parameters

This section describes how aXes-Cloud interprets parameters in the configuration file. The parameters are database.host, database.library, service.host and service.library. The configuration contains a set of parameters for DE and an equivalent set for WSFM.

Deriving values for database.host and database.library

aXes-Cloud uses the database parameter values to locate DE and WSFM databases on a server. Table 10 (page 19) explains the policies used to derive values for the database host parameter.

Table 10: Policies for Deriving database.host Values

Policies for deriving or resolving values for database.host
Database host is a fixed value, configured as database.host="MyServer"
Database host is derived from the service host parameter, configured as database.host="{SERVICEHOST}"
Browser host overrides the service host when users include a server name at sign on, MyServer/MyUserId.
Browser host does not override a fixed value.

Table 11 (page 19) explains the policies used to derive values for the database library parameter.

Table 11: Policies for Deriving database.library Values

Policies for deriving or resolving values for database.library
Database library is a fixed value, configured as database.library="MyLibrary"
Database library is derived from the service host parameter, configured as database.library="{SERVICEHOST}"
Browser host overrides the service host when users include a server name at sign on, MyServer/MyUserId.
Browser host does not override a fixed value for the database library.
Libraries included in the configuration as a fixed value, or substituted by {SERVICEHOST} or provided as the browser host must exist on the database host server.

Deriving values for service.host and service.library

aXes-Cloud uses the service host parameter values to determine the server where DE queries and WSFM searches will run. Table 12 (page 19) explains the policies used to derive values for the service host parameter.

Table 12: Policies for Deriving service.host Values

Policies for deriving or resolving values for service.host
Service host is a fixed value, configured as service.host="MyServer"
Browser host overrides the service host when users include a server name at sign on, MyServer/MyUserId.
If the service host parameter is missing from the configuration, its value defaults to service.host="LOCALHOST" unless overridden by browser host.

The service.library parameter defaults to QGPL.

Deployment: cloud gateway server with one corporate server

This section provides example configurations based on a deployment (Figure 7, page 20) consisting of a cloud gateway server named Gateway and one connected corporate server named Apollo and explains how aXes-Cloud resolves the database host and service host parameters.

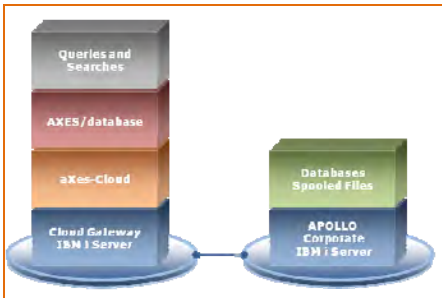
All examples assume that DE and WSFM databases reside on the same server and that DE and WSFM services are provided by the same server. The configuration also allows separation of the databases and where services run.

Figure 7: Deployment: Gateway Server with One Corporate Server

The configuration of database.host, database.library, service.host and user sign on determines where the databases reside and where DE queries and WSFM searches run.

Table 13 (page 20) shows the configuration for both databases and services on the same server. This example shows what to configure to use the cloud gateway server as the server for database instances and running DE queries and WSFM searches.

Table 13: Databases, Queries and Searches on the Gateway Server

	Configuration: database.host = "LOCALHOST" database.library = "AXES" service.host="LOCALHOST"
User signs on as	MyUserId
Database resides on	Gateway (derived from database.host)
Database library resolves to	AXES (derived from database.library)
Queries run on	Gateway (derived from service.host)

DE and WSFM databases reside on the cloud gateway server and the user runs DE queries and WSFM searches on the cloud gateway server.

Table 14 (page 20) shows how to use server name from the user sign on override the configured value of service host to a server name chosen by the user at sign on.

Table 14: Databases on Gateway, Queries and Searches on a Corporate Server

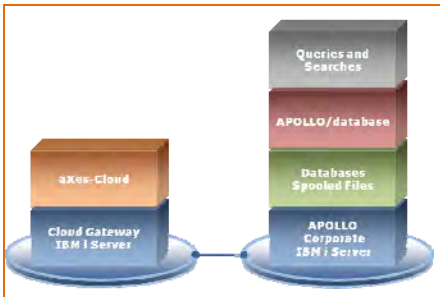
	Configuration: database.host = "LOCALHOST" database.library = "AXES" service.host="LOCALHOST"

User signs on as	APOLLO/MyUserId
Database resides on	Gateway (derived from database.host)
Database library resolves to	AXES (derived from database.library)
Queries run on	APOLLO (derived from server name in user sign on)

DE and WSFM databases reside on the cloud gateway server while the user runs DE queries and WSFM searches on the corporate server named Apollo.

Table 15 (page 21) shows how to configure the databases on a corporate server (Apollo) and also run queries and searches on the corporate server.

Table 15: Databases, Queries and Searches on a Corporate Server

	Configuration: database.host = "APOLLO" database.library = "{SERVICEHOST}" service.host="APOLLO"
User signs on as	MyUserId or APOLLO/MyUserId
Database resides on	APOLLO (derived from database.host)
Database library resolves to	APOLLO (derived from service.host)
Queries run on	APOLLO (derived from service.host)

DE and WSFM databases reside on Apollo and the user runs DE queries and WSFM searches on Apollo. If the user signs on as APOLLO/MyUserId, the server name overrides the value of service host. In this example, the override has the same effect as the configuration because all values resolve to the Apollo. This configuration will operate in the same manner using the parameter database.host="{SERVICEHOST}".

Deployment: cloud gateway server with two corporate servers

This section provides example configurations based on a deployment consisting of a cloud gateway server named Gateway and two corporate servers named Apollo and Zeus (Figure 8, page 22) and explains how aXes-Cloud resolves the database host and service host parameters.

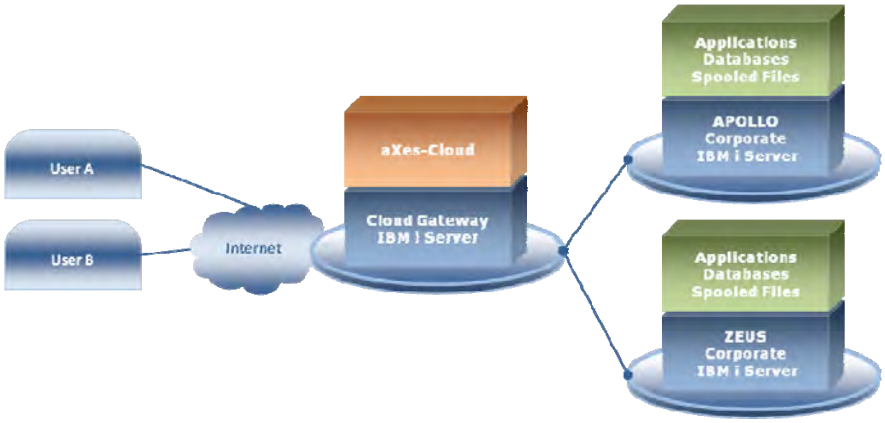
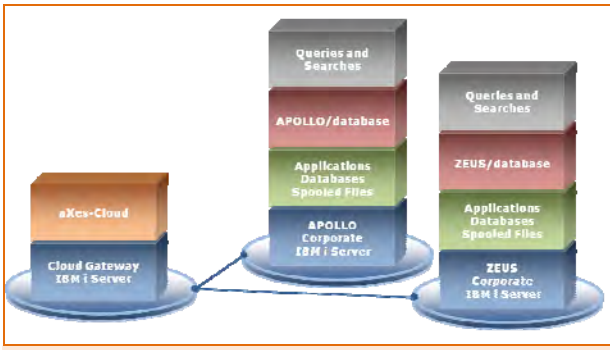


Figure 8: Deployment: Gateway Server with Two Corporate Servers

The configuration shown in Table 16 (page 22) places DE and WSFM databases on each server in a library named after the server. Users who sign on as APOLLO/MyUserId will use the databases on Apollo in the library name Apollo. Users who sign on as ZEUS/MyUserId will use the databases on Zeus in the library name Zeus. The use of {SERVICEHOST} as the value in the database host parameter allows placement of the DE and WSFM databases on the servers that will run queries and searches.

Table 16: Databases, Queries and Searches on Two Corporate Servers



Configuration:

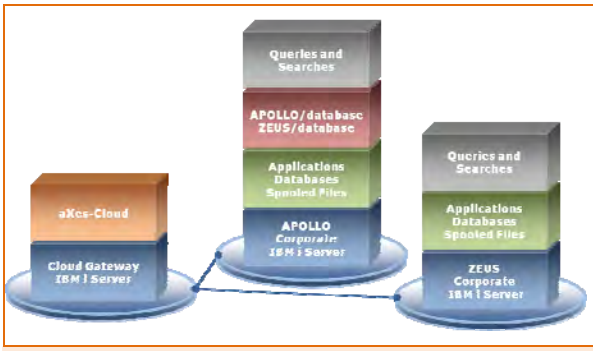
database.host = "{SERVICEHOST}"
database.library = "{SERVICEHOST}"
service.host="APOLLO"

User signs on as	APOLLO/MyUserId
Database resides on	APOLLO (derived from server name from sign on)
Database library resolves to	APOLLO (derived from server name from sign on)
Queries run on	APOLLO (derived from server name from sign on)

User signs on as	ZEUS/MyUserId
Database resides on	ZEUS (derived from server name from sign on)
Database library resolves to	ZEUS (derived from server name from sign on)
Queries run on	ZEUS (derived from server name from sign on)

The configuration shown in Table 17 (page 23) places all DE and WSFM database instances on the server named Apollo in libraries named after the servers. Users who sign on as APOLLO/MyUserId will use the database on Apollo in the library name Apollo and run DE queries and WSFM searches on Apollo. Users who sign on as ZEUS/MyUserId will use the database on Apollo in the library named Zeus and run DE queries and WSFM searches on the server named Zeus. Using a specific server name as the value in the database host parameter places all database instances on the same server.

Table 17: Databases on One Corporate Server, Queries and Searches on Two Corporate Servers



Configuration:

database.host = "APOLLO"
database.library = "{SERVICEHOST}"
service.host="APOLLO"

User signs on as	APOLLO/MyUserId
Database resides on	APOLLO (derived from database.host)
Database library resolves to	APOLLO (derived from server name from sign on)
Queries run on	APOLLO (derived from server name from sign on)

User signs on as	ZEUS/MyUserId
Database resides on	APOLLO (derived from database.host)
Database library resolves to	ZEUS (derived from server name from sign on)
Queries run on	ZEUS (derived from server name from sign on)

Controlling access to DE and WSFM databases

DE and WSFM databases consist of a number of tables (or files) that are accessible to any user with appropriate authority. To restrict access to the tables by any user create a specific database user. Create a user profile with limited operating system authority and full access to the tables. Configure table access properties to deny public access and allow full access to the database user profile.

Table 18: database.user and database.password Parameters Syntax

Database user and password parameters syntax
<parameter name="database.user" value="database user profile"/> <parameter name="database.password" value="database password"/>

Add the database.user and database.password parameters to the configuration file. Table 18 (page 23) shows the generic syntax of the parameters and Table 19 (page 23) shows an example where the user is CLOUDDB and the password is CLOUDPW.

Table 19: database.user and database.password Parameters Example

Database user and password parameters example
<parameter name="database.user" value="CLOUDDB "/> <parameter name="database.password" value="CLOUDPW"/>

You may use the same user profile for both DE and WSFM or create separate profiles. Insert the database.user and database.password parameters in the DE section of the configuration file and the WSFM section of the configuration file (even when you use the same user profile for both).

aXes-Cloud will use the database user's access authority to perform insert, update and delete actions on DE and WSFM databases.

Warning	Using dedicated user profile(s) to access the DE and WSFM databases is essential when you deploy these databases on the cloud gateway server.

Section 2 — Administration

This section provides information that will help administrators manage aXes-Cloud.

Administrator responsibilities

Administrator responsibilities are:

- Installing aXes-Cloud on the cloud gateway server
- Configuring aXes and aXes-Cloud server components
- Configuring connections between the cloud gateway server and corporate servers
- Managing users
- Ensuring regular backups occur and archiving log files

Installation and configuration are once only or occasional tasks. Housekeeping is an ongoing task. Troubleshooting is an occasional task. Managing users is an ongoing task.

Before you start

This section describes issues you need to think through and decisions you must take before installing and configuring DE.

Prerequisites

The prerequisites for aXes-Cloud:

- aXes installation media
- Licences
- An aXes-Cloud deployment design and deployment plan

The prerequisite aXes software is:

- aXes-Cloud, including the application server and Terminal Server
- aXes Data Explorer (optional)
- aXes Web Spooled File Manager (optional)

Install the prerequisite software before configuring aXes-Cloud.

Pre-requisites for the cloud gateway server:

- Java virtual machine
- IBM Toolbox for Java

Pre-requisites for connected corporate servers are:

- Telnet server configured and operational

Planning your deployment is an essential prerequisite to installing and configuring aXes-Cloud.

Decisions you need to make

User authorisation and automatic sign on

Users must provide their credentials (user identification and password) to access aXes-Cloud and corporate servers.

Sign on can be automatic (implicit) or manual (explicit) for corporate servers. Automatic sign on requires that administrators allow automatic sign on to the cloud gateway server and

in this case, users only need to present their credentials once (to the connected corporate servers). Manual sign on means that users must present their credentials twice. The first credential presentation occurs at the initial sign on page for aXes-Cloud and the second occurs during session initiation for the corporate server.

aXes-Cloud caters for automatic and manual sign on. Choosing which option to use is an administrator responsibility.

DE and WSFM database instances and locations

Choose single or multiple instance deployments for DE database(s).

Decide where to place DE database(s).

Choose single or multiple instance deployments for WSFM database(s).

Decide where to place WSFM database(s).

Telnet servers

aXes-Cloud requires active Telnet servers on corporate servers connected to the cloud gateway server.

Port numbers

The default port numbers will be adequate for single instance installations of aXes-Cloud, provided that they are compatible with your environment.

You will need additional port numbers when deploying multiple instances of aXes-Cloud.

Resource naming conventions

Design a set of conventions for naming resources associated with an instance of aXes-Cloud.

Installation and configuration process

The steps in the installation and configuration process are:

1.	Read the release documentation. Read the administrator guides for Terminal Server, DE and WSFM.
2.	Install the prerequisite software on the server
3.	Install aXes-Cloud on the IBM i server chosen as the cloud server.
4.	Apply aXes licences.
5.	Apply aXes-Cloud remote service activation for either or both DE and WSFM.
6.	Configure aXes-Cloud instances.
7.	Configure DE (if licensed to use this feature).
8.	Configure WSFM (if licensed to use this feature).

You will reduce your workload if you are familiar with the options for configuration before you start.

Install server components

aXes-Cloud is part of the aXes suite of products and installing aXes also installs aXes-Cloud.

Install the aXes-Cloud software.

Configure the base instance.

Create and configure additional instances, if you intend to use multiple instances.

Licences

To use aXes-Cloud you must obtain and apply licenses for aXes-Cloud.

aXes-Cloud requires a license and remote activation code for both DE and WSFM.

Remote service activation

DE and WSFM each require a remote service activation code inserted into the configuration files. A remote service activation is a hexadecimal code (or hash) derived from the serial number of the IBM i server that is the cloud gateway server and the service (DE or WSFM).

To enable aXes cloud services for DE and WSFM, insert the remote service activation code into the configuration file for each service you have licensed. For example,

```
<parameter name="service.remote.activation"
value="815223DB1E993AEDDC8C84F93A074D61B39522DF"/>
```

The value in this example is the activiation code.

Creating aXes-Cloud instances

The default installation of aXes-Cloud creates a directory in the IFS named, /axes, and a library named AXES. All aXes-Cloud instances use the AXES library, i.e., the same aXes-Cloud software. For multiple instance deployments, each instance has it own copy of the /axes directory.

Table 20 (page 27) describes how to create an instance of aXes-Cloud.

Table 20: Create an Instance of aXes-Cloud

Steps	Action
1.	Create a directory in the IFS, for example, /AGSMyServer.
2.	Copy the contents of the /axes directory into the new instance directory.
3.	Add the instance to aXes-Cloud using this command: ADDAXESW3 INSTANCE(AGSMyServer) CFG('/AGSMyServer')

You need to change the configuration of the new instance, including IP address, port number, and customise the sign on page.

Operating aXes Terminal Server as a cloud service

This section explains what administrators need to do to configure aXes Terminal Server services.

Configuration

Determine the way you wish to operate aXes Terminal Server. The options are as follows.

- One aXes-Cloud instance for all connected corporate servers
- A dedicated aXes-Cloud instance for each connected corporate server
- A combination of the single instance or multiple instances

You need to configure aXes-Cloud for each instance.

Create subsystems, devices and queues

Table 21 (page 28) contains a list of resources required for each instance of aXes-Cloud.

Table 21: Resources required for aXes-Cloud Deployments

Resources	
aXes-Cloud instance	aXes-Cloud requires a minimum of one instance. Multiple instances are copies of an original instance with additional configuration and customisation.
aXes-Cloud software	Shared by all instances of aXes-Cloud.
Job queue	Recommended for each instance
Message queue	Recommended for each instance
Output queue	Recommended for each instance
Sub-system	Recommended for each instance
User profile	Mandatory for each instance
Virtual devices	Recommended for each instance. Using a unique naming convention will assist administrators to identify devices associated with an aXes-Cloud instance.

Design naming conventions and create the objects on the cloud gateway server for each instance of aXes-Cloud you wish to operate.

Configuring connections for aXes Terminal Server

When users sign on to aXes-Cloud using aXes Terminal Server (aXes-TS2), they expect to be routed to a corporate server. Table 22 (page 28) describes how to set up connections for aXes Terminal Server sessions.

Table 22: Configuring Connections for aXes Terminal Server with aXes-Cloud

Steps	Actions
1.	Create an instance of aXes-Cloud for each corporate server that will connect to the cloud gateway.
2.	Create a URL for each instance of aXes-Cloud.
3.	If you want to use the implicit sign on method: Customise the aXes-Cloud sign on page (for each instance) so that the cloud gateway will sign the user on automatically. This requires a user identification and password in the URL or in the customised sign on page. If you want to use the explicit sign on method: Customise the aXes-Cloud sign on page (for each instance) so that the cloud gateway will request user credentials. Sign on to aXes-Cloud is not automatic.
4.	Create an exit program and associate it with the aXes-Cloud user profile.

If users wish to connect to multiple corporate servers, they can open a browser page for each server to which they wish to connect.

Exit program for aXes-Cloud sign on

Create an exit program associated with the aXes-Cloud sign on.

The exit program starts a Telnet connection to a connected corporate server.

Connected corporate servers

Configuration

The configuration of aXes Terminal Server as a cloud service requires no additional software for, or change to, connected corporate servers. However, on each connected corporate server you might consider a different sign on page, a sign on exit program and device naming if you need to differentiate users who sign on from an internal network and users who sign on via aXes-Cloud.

Telnet servers

aXes-Cloud requires active Telnet servers on each connected corporate server.

Consult IBM guides to discover how to configure Telnet servers.

User sign on policy

Determine the user sign on policy. There are two options. Option one is explicit sign on to the cloud gateway server and explicit sign on to connected corporate servers. Option two, is implicit sign on to the cloud gateway server and explicit sign on to connected corporate servers.

Implicit sign on to both the cloud gateway and connected corporate servers may have security implications.

Operating Data Explorer as a cloud service

This section explains what administrators need to do to configure DE services.

Configuration

You need to be sure that the default configuration for DE is suitable for your environment and if not, you need to adjust the default configuration. The DE administration guide explains the configuration. aXes-Cloud requires additional configuration items for DE to operate as a cloud service.

Choosing where to locate the DE database

DE uses a database to store information about registered users, folders users create, and filters and queries they build.

Administrators are responsible for determining the optimum location for DE databases and creating a database user profile.

User registration

Users need to be registered to operate DE.

The DE administration guide describes the registration procedure.

Operating Spooled File Manager as a cloud service

This section explains what administrators need to do to configure WSFM services.

Configuration

You need to be sure that the default configuration for WSFM is suitable for your environment and if not, you need to adjust the default configuration. The WSFM administration guide explains the configuration. aXes-Cloud requires additional configuration items for WSFM to operate as a cloud service.

Choosing where to locate the WSFM database

WSFM uses a database to store information about registered users, folders users create, and filters and searches they build.

Administrators are responsible for determining the optimum location for WSFM databases and creating a database user profile.

User registration

Users need to be registered to operate WSFM.

The WSFM administration guide describes the registration procedure.

Installation and connection for users of aXes-Cloud

Installation for users of aXes-Cloud

Users do not need to install any software on their computer or mobile device to use aXes-Cloud.

The prerequisites for are:

- A supported browser — Chrome, Firefox, Internet Explorer and Safari
- A network connection to the Internet and/or corporate network (fixed line or wireless)
- The address (URL) for aXes-Cloud at your site

Administrators are responsible for providing users with the address.

Starting a connection with aXes-Cloud

Users initiate a connection with aXes-Cloud from a browser:

1.	Open the browser.
2.	Type the address (URL) into the location bar and press the Enter key or the Go icon.
3.	Wait for aXes-Cloud to display the sign on page and then follow the sign on instructions.

When users sign off from a corporate server, they should end the connection, SIGNOFF ENDCNN(*YES) is an example of the sign off command, or use a program to end the connection.

How users sign on to aXes-Cloud

When users initiate a connection with aXes-Cloud, it responds by sending a sign on page to the browser.

User identification and password

To sign on users must provide a user identification and password (credentials). The steps in the sign on procedure are:

1.	Type your user identification.
2.	Type your password.
3.	Press the sign on button
4.	Wait while aXes-Cloud verifies your credentials. If sign on is successful, aXes-Cloud will display the next page. If sign on is unsuccessful, aXes-Cloud will display an error message.

What users should do if sign on fails

Sign on will fail if you make a mistake typing the user identification and/or the password. Check you have the correct user identification and re-type the password.

If users do not know or forget their user identification and/or password, they need to contact an administrator.

Documentation: administrator guides

Table 23 (page 31) lists the documentation available for aXes-Cloud.

Table 23: Documentation for aXes-Cloud

Document	Purpose
Administrator Guide	The aXes-Cloud Administrator Guide explains how to configure aXes-Cloud and to manage people using aXes-Cloud. This guide is intended for use by administrators. The aXes-Cloud Administrator Guide is this document.

Administrators should be familiar with administrator guides for aXes, DE and WSFM.

Section 3 – Housekeeping and Troubleshooting

This section describes how to perform housekeeping tasks, track down problems and contact support.

Logging

This section explains administrator tasks for managing access log files and error log files. These files accumulate with use and administrators should archive and/or clear the files periodically.

Log files: names and locations

In a typical installation log files reside in a logging folder.

Location	axes/jsm/instance/www/instance/logs/
Log File Names	error.log access.log

Examining the content of these files will reveal information about activity and errors.

Enable and disable logging

The errorlog and accesslog directives in the configuration file control logging activity.

To enable logging for both error and access logs, set the value of the enabled parameter as true:

```
<errorlog enabled="true" file="www/instance/logs/error.log"/>
<accesslog enabled="true" file="www/instance/logs/access.log"/>
```

To disable logging for both error and access logs, set the value of the enabled parameter as false:

```
<errorlog enabled="false" file="www/instance/logs/error.log"/>
<accesslog enabled="false" file="www/instance/logs/access.log"/>
```

Error and access logs operate independently. For example you can enable error logging and disable access logging.

Archive log files

Archive log files by copying them to another location.

Delete log files

Do not delete the active error.log and access.log files.

DE and WSFM will archive the active log files periodically. Before deleting the archived log files, ensure you either have a backup or archive copy of the files.

Tracing

This section explains the administrator tasks for collecting data to assist with tracking errors. Administrators need to enable and disable tracing services and clear trace files when they are no longer needed.

Tracing allows you to collect information about the activity of DE and WSFM. When enabled, tracing directs STDOUT and STDERR to the trace files STDOUT.TXT and STDERR.TXT.

The Java Virtual Machine and instance information is logged to a MANAGER.TXT file.

Trace files: names and locations

In a typical installation trace files reside in two folders.

The first location provides high level trace information.

Location	axes/jsm/instance/trace/[job number]/
Log File Names	CLASSPATH.TXT MANAGER.TXT STDERR.TXT STDOUT.TXT
Sample	axes/jsm/instance/trace/189002/STDOUT.TXT

Where [job number] is the job number of the active job when tracing occurred.

The second location provides detailed trace information.

Location	axes/jsm/instance/trace/[job number]/[date]/
Log File Directory Names	HTTP00000000 HTTP00000001 HTTP00000002 HTTP00000003
Sample	axes/jsm/instance/trace/309001/2009-11-24/HTTP00000004

Enable and disable tracing

The trace and clienttrace directives in the configuration file control tracing activity. These directives are in the virtual host section of and are associated with services.

To enable tracing for a service, set the value of the trace and/or clienttrace parameter as true:

```
trace="true" clienttrace="true"
```

To disable tracing for a service, set the value of the trace and/or clienttrace parameter as false:

```
trace="false" clienttrace="false"
```

Tracing directives operate independently. For example you can enable tracing for one service or all services.

Clear trace files

Use the command CLRJSM command to clear trace files and subdirectories. The command has the following parameters:

INSTANCE	The instance defaults to a value of *DEFAULT. This is the recommended value.
TRACEDIR	The option defaults to a value of *YES. Valid values are *YES, *NO. The value *YES removes files and subdirectories in the trace directory.

TEMPDIR	The option defaults to a value of *YES. Valid values are *YES, *NO. The value *YES removes all files and sub-directories in the TEMP directory. Do not clear the TEMP directory while JSM services are running, as this action will delete temporary files used by the services. Always use *NO when clearing trace files from a running JSM instance.
KEEP	Valid values are in the range 0 to 99 days. Zero means do not keep the files. Running the command will delete ALL files and sub-directories. Use 1 to keep only today's files. Use 2 to keep today's and yesterday's files.

Archive trace files

Archive trace files by copying them to another location, before clearing the files.

Certificate management

Use the Java keytool command to create a private key and certificate suitable for the TLS/SSL server credentials.

```
%JDKPATH%\bin\keytool  -genkey —alias sslkey —keyalg RSA —keysize 1024
                       -sigalg SHA1withRSA
                       -keypass password —keystore wwwssl.jks
                       -storepass password —validity 365
                       -dname "CN=Web Applications,OU=Web Services,O=My
                       Company,L=My Location,S=My State,C=US"
```

Substitute your equivalent values for My Company, My Location, My State and C (country).

Troubleshooting

The troubleshooting section suggests the parts of aXes-Cloud to check when it does not operate as expected.

Installation and configuration

JSM HTTP server does not start

1.	Check the manager.properties httpd directive: httpd=system/httpd.xml Ensure that the name specified for the httpd configuration file is the name of the actual httpd configuration file.
2.	Check the JSM trace file MANAGER.TXT for JSM HTTP server start up messages.
3.	Review the configuration directives in the instance tag.
4.	Confirm that the JSM HTTP server is started by using a browser to access the home index.html file.

Ensure that the configuration is complete before attempting the start the JSM HTTP server.

IBM Toolbox for Java not installed

1.	Ensure that the IBM Toolbox for Java is installed. Verify that the file, "jt400.jar", resides in the JSM jar directory.

Installing the IBM Toolbox for Java is an essential prerequisite for operating DE.

File ownership and permissions for files in the IFS

1.	The JSM HTTP server by default runs as the JSM job description user profile. Any files and directories created should be owned by that user profile.
2.	Verify the user profile, file and directory ownership.
3.	Change ownerships if necessary.

DE database

1.	DE uses tables to store its information. The tables must exist in the library specified in the database.library directive: <parameter name="database.library" value="AXES"/> In this example the library is AXES
2.	Verify that the tables are in the library defined in the database.library directive. The tables are: DBMUSER (PF-DTA) DBMUSER1 (LF) DBMUSER2 (LF) DBMFOLDER (PF-DTA) DBMFOLDER1 (LF) DBMFILTER (PF-DTA) DBMFILTER1 (LF) DBMFILTER2 (LF)
3.	Change the database.library directive to the library where the files reside. Or, move the files to the library defined in the database.library directive.

WSFM database

1.	WSFM uses tables to store its information. The tables must exist in the library specified in the database.library directive: <parameter name="database.library" value="AXES"/> In this example the library is AXES

2.	Verify that the tables are in the library defined in the database.library directive. The tables are: SFMUSER (PF-DTA) SFMUSER1 (LF) SFMUSER2 (LF) SFMFOLDER (PF-DTA) SFMFOLDER1 (LF) SFMFILTER (PF-DTA) SFMFILTER1 (LF) SFMFILTER2 (LF)
3.	Change the database.library directive to the library where the files reside. Or, move the files to the library defined in the database.library directive.

Before you contact aXes support

Table 24 (page 36) contains the information you need to send to Support if you experience a problem with aXes.

Table 24: Information to Send to Support

Context	Information
Contact and customer details	Your company name Your name Customer name
aXes version(s)	What version of aXes being is being reported? What aXes PTFs are installed and applied to this version? Are other versions of aXes also installed on the same IBM i? Is Terminal Server (TS) installed? Is the Web Spooler (WS) installed? Is the Intersession Option installed as well? Has another version of aXes been successfully deployed on the same IBM i? (If so, please provide details). How is aXes licensed — processor or session based? If processor based, how many processors are licensed? If session based, how many sessions are licensed?
IBM hardware	Model number (QMODEL) Serial number (QSRLNBR) Processor Feature Number (QPRCFEAT) Processor Group (Tier) Is aXes running in an LPAR? If yes, how many Processors are allocated to that LPAR?

Context	Information
IBM operating system software	What operating system version is running aXes? (Version, Release and Modification) What operating system version of the Licensed Internal Code (LIC) is installed? What IBM operating system PTFs have been installed? Are the latest PTFs applied?
Problem report	Provide a detailed description of the problem. Include a full history of events leading up to the occurrence of the problem. For example, did the problem occur after you installed new software, applied PTFs or upgraded the operating system? The more detailed the information, the easier it is for Support to understand your problem.

Section 4 — Configuration

This section explains the tools and configuration options available to administrators who will manage DE and WSFM as cloud services.

The configuration process

The steps in the configuration process are:

1.	Copy and rename the configuration file: httpd-template.xml. The recommended name for the copied file is: system/httpd.xml You can use any valid name but you should choose a name that indicates the file content.
2.	Change the httpd property in the manager.properties file to the new name allocated when you copied the original httpd configuration file. For example, if you used the recommended name for the copied httpd configuration file, change the httpd property in manager.properties to: httpd=system/httpd.xml If not, use the name allocated to the copied file.
3.	Configure axinfo.json
4.	Configure the JSM HTTP server instance.
5.	Configure the services.
6.	Configure the database connections.
7.	Configure the user registration properties.
8.	Insert remote service activations for DE and/or WSFM.

The configuration process assumes you have completed the installation successfully.

Important reminders	Always make a copy of the httpd configuration file before you edit its contents. Keep a separate copy of the httpd configuration file that is active in your production environment.

Use a text editor or XML editor when changing configuration items and parameters in the httpd configuration file. Exercise care when editing and make sure you do not change tag names.

About the configuration files

The configuration details reside in three files: manager.properties, httpd.xml and axinfo.json. This section describes the content of the configuration files. Later sections describe how to change individual configuration items.

File locations

Table 25 (page 39) provides the folder locations of the configuration files.

Table 25: Configuration File Locations

Configuration Files	Locations
axinfo.json	myaxes/jsm/
httpd.xml	myaxes/jsm/instance/system/
httpd-template.xml	myaxes/jsm/instance/system/
manager.properties	myaxes/jsm/instance/system/

The root folder (myaxes) in the location is the folder name chosen on installation.

Manager.properties

The properties file, manager.properties, contains information about the installed instance of DE. Table 26 (page 39) explains the properties manager.properties.

Table 26: Explanation of manager.properties

Properties	Definitions
httpd=system/httpd.xml	Name of the httpd configuration file
httpd.axes.release=V2R1M0	Version of aXes installed (used for licence checking)
httpd.axes.instance=AXES	Name of the instance (used for licence checking)

After updating the httpd configuration file property in manager.properties, the configuration file will look like the following example (Table 27, page 39).

Table 27: Example of an Updated manager.properties File

Properties
httpd=system/httpd-template.xml httpd=system/httpd.xml httpd.axes.release=V2R1M0 httpd.axes.instance=AXES # tcp.port=5560 console.tcp.port=5561 studio.client.address=*none console.client.address=*none

The line beginning with a hash (#) is a comment line.

Httpd configuration file

The configuration file named httpd.xml contains the configuration items and parameters for DE. The file exists in two forms. The files httpd-template.xml contains default setting and is the template for the operational file named httpd.xml.

Table 28: Structure of the httpd Configuration File

Structure of httpd Configuration File
<?xml version="1.0" encoding="UTF-8"?> <configuration> <instance name="HTTP Instance" active="true" root="www/instance/htdocs" index="index.html"> <errorlog enabled="true" file="www/instance/logs/error.log"/> <accesslog enabled="true" file="www/instance/logs/access.log"/> <listen secure="false" store="pki/wwwssl.jks" password="password" port="5563" sslport="5564" interface="*ALL" backlog="256" buffersend="-1" bufferreceive="-1" nodelay="false" timeout="5" /> +<access> +<mimetype> +<virtual host="*" active="true"> </instance> </configuration>

Table 28 (page 40) shows the structure of the httpd configuration file.

The configuration contains an instance of the JSM HTTP server. The instance has a set of configuration parameters (instance, errorlog, accesslog and listen tags). The instance includes configuration items for access, MIME types (or Internet Media Types) and virtual hosts.

Table 29 (page 40) provides a list of recommendations for changing configuration items and parameters in the httpd configuration file.

Table 29: Mandatory and Optional Changes to Configuration Items and Parameters

Configuration Items	Change Mandatory/Optional
JSMHTTPServiceFile (match URI, class)	Do not change these items
MIME types (server instance)	Optional, but not recommended
MIME types (virtual host)	Optional, but not recommended
Ports	You must choose a port number
Realm (protect tag, virtual host)	Do not change any items in the protect tag
Virtual host name and active	Do not change these items unless you choose multiple instances and/or multiple virtual hosts

The default values of many configuration items and parameters in the httpd configuration will be appropriate for operating DE and WSFM. Some values you must change and some are optional.

Important reminders	Copy the original httpd configuration file. Do not use the original httpd configuration file for your configuration settings. Version upgrades and fixes may alter the parameters in the httpd configuration file and will over-write your configuration settings when installed.

Axinfo configuration file

The axinfo configuration file defines whether DE and/or WSFM are enabled, the ports and URIs for the services.

Table 30: axinfo.json Configuration File

axinfo.json Configuration Example

```
{
    "dbm":
        {
            "enabled":    true,
            "port":       5563,
            "sslport":    5564,
            "uri":        "/axes/database-manager.html"
        }
}
{
    "sfm":
        {
            "enabled":    true,
            "port":       5563,
            "sslport":    5564,
            "uri":        "/axes/spooled-file-manager.html"
        }
}
```

Table 31 (page 41) explains the items in axinfo.json.

Table 31: Configuration Items in axinfo.json

Configuration Items	Definitions
DBM	DBM represents Data Explorer.
SFM	SFM represents Web Spooled File Manager
Enabled	"enabled":true indicates that the service is active. "enable":false indicates that the service is inactive (that is, the service is unavailable)
Ports	Port = the non SSL port (not encrypted) SSLPort = the SSL port (encrypted)
URIs	http://mycompany:5563/axes/database-manager.html https://mycompany:5564/axes/database-manager.html http://mycompany:5563/axes/spooled-file-manager.html https://mycompany:5564/axes/spooled-file-manager.html To allow cross port access to the HTTP servers, the protocol and domain name need to match.

Data Explorer and Web Spooled File Manger are examples of JSM services.

Configure ports

Default ports

The axinfo and httpd configuration files contain items to define the ports to use. Table 32 (page 42) shows the default ports.

Table 32: Default Ports

Port Numbers	Definitions
5560	JSM Server TCP port is used internally and does not accept connections.
5561	JSM Console TCP port is used internally and does not accept connections.
5563	JSM HTTP Server (default port without TLS/SSL)
5564	JSM HTTPS Server (default port with TLS/SSL)

Change default ports

To change ports follow these steps:

1.	Edit axinfo.json.
2.	Change the port parameter. Change the sslport parameter.
3.	Save axinfo.json
4.	Edit httpd.xml (or the equivalent file if you use a different name).
5.	Under the listen tag of the instance: Change the port parameter. Change the sslport parameter.
6.	Save httpd.xml

The configuration files shipped are configured to use the default ports. You do not need to change the port configurations if the default ports are suitable for your installation.

Configure JSM HTTP server

This part of the administrator guide explains how to configure the sections of the httpd configuration file that apply to the JSM HTTP server. There are separate sections that explain how to configure individual services that the JSM HTTP server supports.

Server instance configuration

Table 33 (page 42) shows the set of configuration items and parameters that apply to the JSM HTTP server instance.

Table 33: Configure Server Instance

Configure Server Instance Example
<?xml version="1.0" encoding="UTF-8"?>

Configure Server Instance Example

```
<configuration>
    <instance name="HTTP Instance" active="true" root="www/instance/htdocs"
            index="index.html">
        <errorlog enabled="true" file="www/instance/logs/error.log" />
        <accesslog enabled="true" file="www/instance/logs/access.log" />
        <listen secure="false" store="pki/wwwssl.jks" password="password"
            port="5563"
            sslport="5564"
            interface="*ALL" backlog="256" buffersend="-1" bufferreceive="-1"
            nodelay="false" timeout="5" />
        +<access>
        +<mimetype>
        +<virtual host="*" active="true">
    </instance>
</configuration>
```

In this example, the instance has error logging and access logging enabled (the value of the parameter is "true"). The server will place errors into error.log and access events into access.log. The instance will listen on port 5563.

Table 34: What to Configure for the Server Instance

To configure	Apply these settings	Change default
Access logging	Set accesslog enabled to true. Define the access log file name.	Optional
Active	Active must be true	Do not change
Backlog	Choose the depth of the TCP/IP queue.	Optional
Buffer receive	Set the size (bytes) of the receive buffer	Optional
Buffer send	Set the size (bytes) of the send buffer	Optional
Error logging	Set errorlog enabled to true. Define the error log file name.	Optional
Index	Define the index document	Optional
Instance name	Choose a name	Optional for single instances. Mandatory for more than one instance.
Interface	Use *ALL for all addresses or choose the interface address. Default is all interfaces.	Optional
No delay	Set to True to enable TCP/IP no delay. False uses the operating system settings.	Optional

To configure	Apply these settings	Change default
No TLS/SSL	Set listen secure to false. When listen secure is false, the JSM HTTP server ignores store and password values.	Optional
Port	Insert the port number you want to use.	Mandatory The server expects either port and/or sslport.
Root	Define the root directory (folder)	Optional
Secure connection with TLS/SSL	Set listen secure to true. Define the name of the store. Insert the password.	Optional
SSLport	Insert the port number you want to use.	Mandatory The server expects either port and/or sslport.
Timeout	Set the connection timeout in seconds.	Optional

Whether you need to change the optional items depends on the environment of your installation and your performance requirements.

Controlling access to the server instance

The access directive (at the instance level) specifies rules for accessing the instance of the JSM HTTP server. Using the access directive you can:

- Allow addresses
- Deny addresses

You use combinations of allow and deny directives to control access to the server instance. Addresses can be specific (10.2.45.1), masks (10.2) or generic (indicated by the asterisk (*)).

Do not use directives for user agents or content lengths in this section.

Table 35 (page 44) shows an example of allow and deny directives (at the instance level).

Table 35: Control Access to the Server Instance

Server Instance Access (Allow and Deny) Configuration Example
<?xml version="1.0" encoding="UTF-8"?> <configuration> <instance name="HTTP Instance" active="true" root="www/instance/htdocs" index="index.html"> <errorlog enabled="true" file="www/instance/logs/error.log"/> <accesslog enabled="true" file="www/instance/logs/access.log"/> <listen secure="false" store="pki/wwwssl.jks" password="password" port="5563" sslport="5564" interface="*ALL" backlog="256" buffersend="-1" bufferreceive="-1" nodelay="false" timeout="5"/>

Server Instance Access (Allow and Deny) Configuration Example

```
        <access>
            <allow address="*"/>
                <!—the following allow and deny directives are comments
                    <allow address="10.2.1.45"/>
                    <deny address="*"/>
                    <deny address="10.2.1.45"/>
                ->
        </access>
        +<mimetype>
        +<virtual host="*" active="true">
    </instance>
</configuration>
```

In this example <allow address= "*"/> allows access from any address. The other allow and deny directives are inside comments and ignored by the JSM HTTP server.

Table 36: What to Configure for Server Instance Access

To configure	Apply these settings	Change defaults
Allow any address	Use allow address=*	Optional
Access for specific addresses	Add new a directive for each allowed address.	Optional
Deny any address	Use deny address=*	Optional
Deny access for specific addresses	Add new a directive for each denied address.	Optional

MIME types for the server instance

Table 37 (page 45) shows the configuration for MIME types. This part of the httpd configuration file defines MIME types applicable to the whole instance.

The MIME type directives allow the JSM HTTP server to correctly understand the nature of files. Administrators do not need to change these directives.

Table 37: MIME Types for the Server Instance

Server Instance MIME Types Configuration Example

```
<?xml version="1.0" encoding="UTF-8"?>
<configuration>
    <instance name="HTTP Instance" active="true" root="www/instance/htdocs"
            index="index.html">
        <errorlog enabled="true" file="www/instance/logs/error.log"/>
        <accesslog enabled="true" file="www/instance/logs/access.log"/>
        <listen secure="false" store="pki/wwwssl.jks" password="password"
            port="5563"
            sslport="5564"
            interface="*ALL" backlog="256" buffersend="-1" bufferreceive="-1"
            nodelay="false" timeout="5"/>
```

Server Instance MIME Types Configuration Example

```
        +<access>
        <mimetype>
            <map extension="png" type="image/png"/>
            <map extension="gif" type="image/gif"/>
            <map extension="jpg" type="image/jpeg"/>
            <map extension="jpeg" type="image/jpeg"/>
            <map extension="tiff" type="image/tiff"/>
            <map extension="ico" type="image/x-icon"/>
            <map extension="pdf" type="application/pdf"/>
            <map extension="css" type="text/css; charset=utf-8"/>
            <map extension="xsl" type="text/xls; charset=utf-8"/>
            <map extension="xml" type="text/xml; charset=utf-8"/>
            <map extension="htm" type="text/html; charset=utf-8"/>
            <map extension="html" type="text/html; charset=utf-8"/>
            <map extension="js" type="application/x-javascript; charset=utf-8"/>
        </mimetype>
        +<virtual host="*" active="true">
    </instance>
</configuration>
```

MIME type directives in the virtual host section of the configuration file override MIME types specified in this section.

To determine allowed MIME types, the JSM HTTP server looks at MIME types in the virtual host section of the httpd configuration file and then at MIME types in the server instance. Place MIME types in the server instance that will apply all virtual hosts. Place MIME types that are unique to a virtual host in the MIME types section of the virtual host.

Table 38: What to Configure for the Server Instance MIME Types

To configure	Apply these settings	Change defaults
MIME types	Use the default list	Not recommended
Add MIME types	Add new a directive for each MIME type.	Optional

Virtual host configuration

The JSM HTTP server is capable of managing multiple virtual hosts. Table 39 (page 46) shows the virtual host configuration for DE.

Table 39: Virtual Host Configuration

Virtual Host Configuration Example

```
<?xml version="1.0" encoding="UTF-8"?>
<configuration>
    <instance name="HTTP Instance" active="true" root="www/instance/htdocs"
            index="index.html">
        <errorlog enabled="true" file="www/instance/logs/error.log"/>
        <accesslog enabled="true" file="www/instance/logs/access.log"/>
        <listen secure="false" store="pki/wwwssl.jks" password="password"
```

Virtual Host Configuration Example

```
            port="5563"
            sslport="5564"
            interface="*ALL" backlog="256" buffersend="-1" bufferreceive="-1"
            nodelay="false" timeout="5"/>
        +<access>
        +<mimetype>
        <virtual host="*" active="true">
            +<access>
            +<protect>
            +<script>
            +<mimetype>
        </virtual>
    </instance>
</configuration>
```

The host is the name of the virtual host to match with the HTTP host property. In this case the asterisk ("*") indicates acceptance of requests from any host.

Table 40: What to Configure for the Virtual Host

To configure	Apply these settings	Change defaults
Virtual host	Use the default value asterisk (*)	Do not change
Active	Use the value true	Do not change

The active parameter has two values "true" and "false". The value of the active parameter must be "true" for the service to operate.

Access, protect, script and MIME type are sub sections of the virtual section in the httpd configuration file.

Virtual host access

The access directives in the virtual host section in the httpd configuration file control access to services provided by the JSM HTTP server. These directives override the access directives defined for the server instance. Using the virtual host access configuration you can:

- Allow addresses
- Deny addresses
- Allow user agents
- Deny user agents
- Allow content lengths
- Deny content lengths

Table 41 (page 47) shows an example of configuring virtual host access directives.

Table 41: Virtual Host Access Directives

Virtual Host Access Directives Configuration Example

```
<?xml version="1.0" encoding="UTF-8"?>
<configuration>
    <instance name="HTTP Instance" active="true" root="www/instance/htdocs"
            index="index.html">
```

Virtual Host Access Directives Configuration Example

```
        <errorlog enabled="true" file="www/instance/logs/error.log"/>
        <accesslog enabled="true" file="www/instance/logs/access.log"/>
        <listen secure="false" store="pki/wwwssl.jks" password="password"
            port="5563"
            sslport="5564"
            interface="*ALL" backlog="256" buffersend="-1" bufferreceive="-1"
            nodelay="false" timeout="5"/>
        +<access>
        +<mimetype>
        <virtual host="*" active="true">
            <access>
                <allow address="*"/>
                <deny useragent="webos"/>
                <deny useragent="android"/>
                <deny useragent="ipad"/>
                <deny useragent="iphone"/>
                <allow useragent="*"/>
                <allow useragent="?"/>
                <deny contentlength="4096"/> <!—deny content GT value →
            </access>
            +<protect>
            +<script>
            +<mimetype>
        </virtual>
    </instance>
</configuration>
```

In this example, requests from any address are acceptable, deny access is explicit for several user agents, other user agents are acceptable and content length greater than 4096 is unacceptable.

Table 42: What to Configure for Virtual Host Access

To configure	Apply these settings	Change defaults
Allow any address	Use allow address=*	Optional
Access for specific addresses	Add a directive for each address.	Optional
Allow any user agent	Use allow useragent=*	Optional
Allow specific user agents	Add a directive for each user agent	Optional
Allow content lengths	Use allow contentlength= "value"	Optional
Deny any address	Use deny address=*	Optional

To configure	Apply these settings	Change defaults
Deny access for specific addresses	Add a directive for each address.	Optional
Deny any user agent	Use deny useragent=*	Optional
Deny specific user agents	Add a directive for each user agent	Optional
Deny content lengths	Use deny contentlength= "value"	Optional

Access directives in the virtual host section of the configuration file override access directives specified in the access section of the server instance.

Virtual host protect

Important reminders	You do not need to configure this section of the httpd configuration file to use DE or WSFM. DE and WSFM use RSA 1024 bit public key encryption which offers stronger protection than basic or digest authentication.

The protection section of the httpd configuration file maps authentication methods to parts of the Web site or application. The realm describes the authentication method and the match URI associates the realm with the protected part of the Web site or application.

Table 43 (page 49) shows an example of the protect configuration.

Table 43: Virtual Host Protect Configuration

Virtual Host Protect Configuration Example

```
<?xml version="1.0" encoding="UTF-8"?>
<configuration>
    <instance name="HTTP Instance" active="true" root="www/instance/htdocs"
            index="index.html">
        <errorlog enabled="true" file="www/instance/logs/error.log"/>
        <accesslog enabled="true" file="www/instance/logs/access.log"/>
        <listen secure="false" store="pki/wwwssl.jks" password="password"
            port="5563"
            sslport="5564"
            interface="*ALL" backlog="256" buffersend="-1" bufferreceive="-1"
            nodelay="false" timeout="5"/>
        +<access>
        +<mimetype>
        <virtual host="*" active="true">
            +<access>
            <protect>
                <realm name="Area 51">
                    <user name="user" access="bb644a9819425bfd8586b408896a1031"/>
                </realm>
                <match uri="/restricted" realm="Area 51" authentication="basic,digest"/>
            </protect>
```

Virtual Host Protect Configuration Example
+<script> +<mimetype> </virtual> </instance> </configuration>

In this example the realm is "Area 51", the user name is "user", the access is a digest of the user information and the configuration uses both basic and digested authentication methods. The match uri = "/restricted" associates the realm with URIs including the match URI.

Table 44: What to Configure for Virtual Host Protect

To configure	Apply these settings	Change defaults
Realm	Define the realm name. Add one or more user names with their access digests. You need to generate the access digest for each user and include them this section the configuration.	Do not change
Match URIs	Add one or more full or partial URIs with their associated realms.	Do not change

You do not need to configure this section of the httpd configuration file.

Virtual host script

The script section of the httpd configuration file contains configuration items associated with the available services. For example, DE uses the following services:

- Query service (HTTPServiceQuery)
- File service (JSMHTTPServiceFile)

Each service has one or more parameters that control the way the service operates. Examples of parameters are:

- Autoregister
- Allow.query.clause.into
- Database.library

Table 45 (page 50) shows an example of a script configuration.

Table 45: Virtual Host Script Configuration – Data Explorer Example

Virtual Host Script Configuration Example
<?xml version="1.0" encoding="UTF-8"?> <configuration> <instance name="HTTP Instance" active="true" root="www/instance/htdocs" index="index.html"> <errorlog enabled="true" file="www/instance/logs/error.log"/> <accesslog enabled="true" file="www/instance/logs/access.log"/> <listen secure="false" store="pki/wwwssl.jks" password="password" port="5563" sslport="5564" interface="*ALL" backlog="256" buffersend="-1" bufferreceive="-1"

Virtual Host Script Configuration Example

```
        nodelay="false" timeout="5"/>
    +<access>
    +<mimetype>
    <virtual host="*" active="true">
        +<access>
        +<protect>
        <script>
            <match uri="/axes/dbmservice.jsp"
                class="com.lansa.jsm.service.HTTPServiceQuery"
                trace="false" clienttrace="false">
                <parameter name="autoregister" value="false"/>
                <parameter name="allow.query.clause.into" value="false"/>
                <parameter name="database.host" value="LOCALHOST"/>
                <parameter name="database.library" value="AXES"/>
                <parameter name="service.host" value="LOCALHOST"/>
            </match>
            <match uri="/" class="com.lansa.jsm.JSMHTTPServiceFile"
                trace="false" clienttrace="false">
                <parameter name="cache.maxage" value="28800"/>
                <parameter name="cache.maxage.pdf" value="28800"/>
                <parameter name="cache.maxage.image" value="28800"/>
            </match>
        </script>
        +<mimetype>
    </virtual>
</instance>
</configuration>
```

This guide includes individual sections that describe how to configure the services; see the sections under "Configure services", (from page 52).

Virtual host MIME types

Table 46 (page 51) shows the configuration for MIME types applicable to the virtual host.

Table 46: Virtual Host MIME Type Configuration

Virtual Host MIME Type Configuration Example

```
<?xml version="1.0" encoding="UTF-8"?>
<configuration>
    <instance name="HTTP Instance" active="true" root="www/instance/htdocs"
        index="index.html">
        <errorlog enabled="true" file="www/instance/logs/error.log"/>
        <accesslog enabled="true" file="www/instance/logs/access.log"/>
        <listen secure="false" store="pki/wwwssl.jks" password="password"
            port="5563"
            sslport="5564"
            interface="*ALL" backlog="256" buffersend="-1" bufferreceive="-1"
```

Virtual Host MIME Type Configuration Example

```
            nodelay="false" timeout="5"/>
        +<access>
        +<mimetype>
        <virtual host="*" active="true">
            +<access>
            +<protect>
            +<script>
            <mimetype>
                <map extension="pdf" type="application/pdf"/>
            </mimetype>
        </virtual>
    </instance>
</configuration>
```

The MIME types defined in this section of the httpd configuration file override MIME types defined for the server instance.

Table 47: What to Configure for Virtual Host MIME Types

To configure	Apply these settings	Change defaults
MIME types	Use the default list	Do not change
Add MIME types	Add new a directive for each MIME type.	Optional

Configure services

The configure services section provides explanations for configuring each of the services that support DE and WSFM. The configuration items and parameters reside in the script section of the httpd configuration file.

JSM HTTP file service configuration

The JSM HTTP server uses the service titled, com.lansa.jsm.JSMHTTPServiceFile, to retrieve files associated with the requested page.

Table 48 (page 52) shows the configuration for this service.

Table 48: Configure JSM HTTP File Service

Configure JSM HTTP File Service

```
<?xml version="1.0" encoding="UTF-8"?>
<configuration>
    <instance name="HTTP Instance" active="true" root="www/instance/htdocs"
            index="index.html">
        <errorlog enabled="true" file="www/instance/logs/error.log"/>
        <accesslog enabled="true" file="www/instance/logs/access.log"/>
        <listen secure="false" store="pki/wwwssl.jks" password="password"
            port="5563"
            sslport="5564"
            interface="*ALL" backlog="256" buffersend="-1" bufferreceive="-1"
```

Configure JSM HTTP File Service

```
            nodelay="false" timeout="5"/>
        +<access>
        +<mimetype>
        <virtual host="*" active="true">
            +<access>
            +<protect>
            <script>
                +<match uri="/axes/dbmservice.jsp"
                <match uri="/" class="com.lansa.jsm.JSMHTTPServiceFile"
                    trace="false" clienttrace="false">
                    <parameter name="cache.maxage" value="28800"/>
                    <parameter name="cache.maxage.pdf" value="28800"/>
                    <parameter name="cache.maxage.image" value="28800"/>
                </match>
            </script>
            +<mimetype>
        </virtual>
    </instance>
</configuration>
```

In this example the match URI is set to "/" to indicate the match applies to all URIs. The cache parameters set the maximum period for caching files and images. In this case the value used for both parameters is 28,800 seconds (or 8 hours).

Table 49: What to Configure for JSM HTTP File Service

To configure	Apply these settings	Change defaults
Match URI	Use the default "/"	Do not change
Class	Use com.lansa.jsm.JSMHTTPServiceFile	Do not change
Cache Maxage	The default maximum age for cached files is 28,800 seconds (8 hours). Do not remove this parameter.	Optional
Cache Maxage Image	The default maximum age for cached image files is 28,800 seconds (8 hours). Do not remove this parameter.	Optional

Do not remove this service from the httpd configuration file.

Allow and deny user access to services

The parameter, service.user.allow, defines who is allowed to use DE or WSFM. The parameter, service.user.deny, defines who is denied access to DE or WSFM services.

The service.user.allow parameter defines acceptable users and the service.user.deny parameter defines users not permitted to use the services. Table 50 (page 54) explains the syntax of these parameters. You may include multiple instances of the service.user.allow and service.user.deny parameters.

Table 50: User Identification/Profile Service Allow and Deny Parameter Syntax

Parameter Syntax	Action by aXes-Cloud
<parameter name= "service.user.allow" value= "UserId"/>	The parameter value defines allowed (or acceptable) user identifications.
<parameter name= "service.user.allow" value= "UserId,UserId,UserId,UserId"/>	The parameter value is a list of user identifications separated by commas. The service.user.allow parameter allows (or accepts) user identifications in the list.
<parameter name= "service.user.deny" value= "UserId"/>	The parameter value denies (excludes) access to one user.
<parameter name= "service.user.deny" value= "UserId,UserId,UserId,UserId"/>	The parameter value is a list of user identifications separated by commas. The service.user.deny parameter denies (excludes) access to user identifications in the list.
<parameter name= "service.user.allow" value= "*USER"/>	The value *USER is a special case. It is a collective value that allows all user identifications.
<parameter name= "service.user.deny" value= "*USER"/>	The value *USER is a special case. It is a collective value that denies all user identifications.

The user authorisation process looks for instances of the service.user.deny parameter and then instances of the service.user.allow parameter.

By default, aXes-Cloud denies access to user identifications beginning with the letter Q. To enable access for these user identifications, configure each user identification explicitly using a service.user.allow parameter.

Example configuration for DE

Table 51 (page 54) illustrates a configuration that includes the service.user.allow and service.user.deny parameters.

Table 51: Example DE Configuration — service.user Allow and Deny Parameters

DE Configuration for service.user.allow and service.user.deny Parameters
<?xml version="1.0" encoding="UTF-8"?> <configuration> <instance name="HTTP Instance" active="true" root="www/instance/htdocs" index="index.html"> <errorlog enabled="true" file="www/instance/logs/error.log"/> <accesslog enabled="true" file="www/instance/logs/access.log"/> <listen secure="false" store="pki/wwwssl.jks" password="password" port="5563" sslport="5564" interface="*ALL" backlog="256" buffersend="-1" bufferreceive="-1" nodelay="false" timeout="5"/> +<access> +<mimetype> <virtual host="*" active="true">

DE Configuration for service.user.allow and service.user.deny Parameters

```
            +<access>
            +<protect>
            <script>
                <match uri="/axes/dbmservice.jsp"
                    class="com.lansa.jsm.service.HTTPServiceQuery"
                    trace="false" clienttrace="false">
                    <parameter name="service.user.deny" value="JohnS,MaryB"/>
                    <parameter name="service.user.deny" value="GeorgeD,WendyF"/>
                    <parameter name="service.user.allow" value="*USER"/>
                    <parameter name="service.user.allow" value="QSYSOPR "/>
                    <parameter name="autoregister" value="false"/>
                    <parameter name="allow.query.clause.into" value="false"/>
                    <parameter name="database.host" value="LOCALHOST"/>
                    <parameter name="database.library" value="AXES"/>
                </match>
                +<match uri="/" class="com.lansa.jsm.JSMHTTPServiceFile"
            </script>
            +<mimetype>
        </virtual>
    </instance>
</configuration>
```

Table 52 (page 55) provides examples and explanations for configuring the service.user.allow and service.user.deny parameters.

Table 52: Controlling User Access to Services - Examples

Parameter Examples	Explanation
<parameter name= "service.user.allow" value= "*USER"/>	This value allows every user identification, except for those beginning with the letter Q.
<parameter name= "service.user.deny" value= "*USER"/>	This value denies access to every user identification, including those beginning with the letter Q. Using the *USER value on a service.user.deny parameter locks out every user. It overrides all values for the service.user.allow parameter.
<parameter name= "service.user.allow" value= "JohnS,MaryB,GeorgeD,WendyF"/>	This configuration allows all user identifications in the list.
<parameter name= "service.user.deny" value= "JohnS,MaryB"/> <parameter name= "service.user.allow" value= "*USER"/>	This configuration denies access to the users JohnS and MaryB, but allows all other users, except those beginning with the letter Q. This example illustrates the optimum method for allowing most users and denying a small number of users.

Parameter Examples	Explanation
<parameter name="service.user.allow" value= "JohnS,MaryB"/>	This configuration allows only users JohnS and MaryB.
<parameter name="service.user.allow" value= "*USER,QSECOFR,QSYSOPR"/>	This service.user.allow value allows all users and both QSECOFR and QSYSOPR. It is unnecessary to include service.user.deny parameters for user identifications beginning with the letter Q; aXes-Cloud denies access to these user identifications by default.

Example configuration for WSFM

Table 53 (page 56) illustrates a configuration for WSFM that includes the service.user.allow and service.user.deny parameters.

Table 53: Example WSFM Configuration — service.user Allow and Deny Parameters

WSFM Configuration for service.user.allow and service.user.deny Parameters

```
<?xml version="1.0" encoding="UTF-8"?>
<configuration>
    <instance name="HTTP Instance" active="true" root="www/instance/htdocs"
            index="index.html">
        <errorlog enabled="true" file="www/instance/logs/error.log"/>
        <accesslog enabled="true" file="www/instance/logs/access.log"/>
        <listen secure="false" store="pki/wwwssl.jks" password="password"
            port="5563"
            sslport="5564"
            interface="*ALL" backlog="256" buffersend="-1" bufferreceive="-1"
            nodelay="false" timeout="5"/>
        +<access>
        +<mimetype>
        <virtual host="*" active="true">
            +<access>
            +<protect>
            <script>
                <match uri="/axes/sfmservice.jsp"
                    class="com.lansa.jsm.service.HTTPServiceSpool"
                    trace="false" clienttrace="false">
                    <parameter name="service.user.deny" value="JohnS,MaryB"/>
                    <parameter name="service.user.deny" value="GeorgeD,WendyF"/>
                    <parameter name="service.user.allow" value="*USER"/>
                    <parameter name="service.user.allow" value="QSYSOPR "/>
                    <parameter name="autoregister" value="false"/>
                    <parameter name="database.host" value="LOCALHOST"/>
                    <parameter name="database.library" value="AXES"/>
                </match>
                +<match uri="/" class="com.lansa.jsm.JSMHTTPServiceFile"
```

WSFM Configuration for service.user.allow and service.user.deny Parameters
</script> +<mimetype> </virtual> </instance> </configuration>

The example configuration shown in Table 53 (page 56) is the equivalent example for WSFM as the DE configuration example shown in Table 51 (page 54).

To use DE and/or WSFM services, users must be authorised by an implicit or explicit service.user.allow parameter and also registered as a DE and/or WSFM user.

User registration

User registration is one of the tasks managed by the services dbmservice.jsp and sfmservice.jsp. Table 54 (page 57) describes the values for the auto-register parameter associated with these services.

Table 54: Auto-register Parameter for dbmservice.jsp and sfmservice.jsp

Parameter Syntax	Action by DE
<parameter name= "autoregister" value= "true"/>	Any person who logs on will be registered with an active profile on the server.
<parameter name= "autoregister" value= "false"/>	Setting the value to false prevents registering users automatically. Administrators will register users manually.

For manual user registration change the value of the autoregister parameter to "false".

For automated user registration change the value of the autoregister parameter to "true".

Table 55 (page 57) shows an example of a configuration file using manual user registration.

Table 55: Auto-register Parameter for Automated or Manual User Registration

Configuration for the Auto Register Parameter
<?xml version="1.0" encoding="UTF-8"?> <configuration> <instance name="HTTP Instance" active="true" root="www/instance/htdocs" index="index.html"> <errorlog enabled="true" file="www/instance/logs/error.log"/> <accesslog enabled="true" file="www/instance/logs/access.log"/> <listen secure="false" store="pki/wwwssl.jks" password="password" port="5563" sslport="5564" interface="*ALL" backlog="256" buffersend="-1" bufferreceive="-1" nodelay="false" timeout="5"/> +<access> +<mimetype> <virtual host="*" active="true"> +<access> +<protect> <script>

Configuration for the Auto Register Parameter

```
            <match uri="/axes/dbmservice.jsp"
                class="com.lansa.jsm.service.HTTPServiceQuery"
                trace="false" clienttrace="false">
                <parameter name="autoregister" value="false"/>
                <parameter name="allow.query.clause.into" value="false"/>
                <parameter name="database.host" value="LOCALHOST"/>
                <parameter name="database.library" value="AXES"/>
            </match>
            +<match uri="/" class="com.lansa.jsm.JSMHTTPServiceFile"
        </script>
        +<mimetype>
    </virtual>
</instance>
</configuration>
```

Table 55 (page 57) shows an example httpd configuration file <script> section and the configuration of the dbmservice. In this example the value of the auto-register parameter is "false", indicating manual user registration. The WSFM service sfmservice .jsp has an equivalent configuration for the autoregister parameter.

Users cannot set the value of the autoregister parameter from the user interface.

Locating DE and WSFM databases

These sections explain the configuration directives and parameters that control the location of the databases used by DE and WSFM. The directives are <script> and <match> and the parameters are "database.host" and "database.library".

On the aXes-Cloud gateway server

Table 56 (page 58) shows database.host and database.library parameters configured so that DE and WSFM databases reside on the cloud gateway server.

Table 56: Configuration for DE and WSFM on the aXes-Cloud Gateway Server

Virtual Host Script Configuration Example

```
<?xml version="1.0" encoding="UTF-8"?>
<configuration>
    <instance name="HTTP Instance" active="true" root="www/instance/htdocs"
            index="index.html">
        <errorlog enabled="true" file="www/instance/logs/error.log"/>
        <accesslog enabled="true" file="www/instance/logs/access.log"/>
        <listen secure="false" store="pki/wwwssl.jks" password="password"
            port="5563"
            sslport="5564"
            interface="*ALL" backlog="256" buffersend="-1" bufferreceive="-1"
            nodelay="false" timeout="5"/>
        +<access>
        +<mimetype>
        <virtual host="*" active="true">
            +<access>
```

Virtual Host Script Configuration Example

```
            +<protect>
            <script>
                <match uri="/axes/dbmservice.jsp"
                    class="com.lansa.jsm.service.HTTPServiceQuery"
                    trace="false" clienttrace="false">
                    <parameter name="autoregister" value="false"/>
                    <parameter name="allow.query.clause.into" value="false"/>
                    <parameter name="database.host" value="LOCALHOST" />
                    <parameter name="database.library" value="AXES" />
                    <parameter name="service.host" value="LOCALHOST" />
                    <parameter name="service.library" value="QGPL" />
                </match>
                <match uri="/axes/sfmservice.jsp"
                    class="com.lansa.jsm.service.HTTPServiceSpool"
                    trace="false" clienttrace="false">
                    <parameter name="autoregister" value="false"/>
                    <parameter name="database.host" value="LOCALHOST" />
                    <parameter name="database.library" value="AXES" />
                    <parameter name="service.host" value="LOCALHOST" />
                    <parameter name="service.library" value="QGPL" />
                </match>
                <match uri="/" class="com.lansa.jsm.JSMHTTPServiceFile"
                    trace="false" clienttrace="false">
                    <parameter name="cache.maxage" value="28800"/>
                    <parameter name="cache.maxage.pdf" value="28800"/>
                    <parameter name="cache.maxage.image" value="28800"/>
                </match>
            </script>
            +<mimetype>
        </virtual>
    </instance>
</configuration>
```

This configuration is the default and the files will reside on the aXes-Cloud server unless you change the default configuration.

On corporate servers

Table 57 (page 59) shows the database.host and database.library parameters configured so that the DE and WSFM databases reside on different servers.

The DE database resides on a corporate server named APOLLO in the library named USRQRYLIB.

The WSFM database resides on a corporate server named ZEUS in the library named ZEUS.

Table 57: Configuration for DE and WSFM on a Corporate Server

Virtual Host Script Configuration Example

```
<?xml version="1.0" encoding="UTF-8"?>
```

Virtual Host Script Configuration Example

```
<configuration>
    <instance name="HTTP Instance" active="true" root="www/instance/htdocs"
            index="index.html">
        <errorlog enabled="true" file="www/instance/logs/error.log"/>
        <accesslog enabled="true" file="www/instance/logs/access.log"/>
        <listen secure="false" store="pki/wwwssl.jks" password="password"
            port="5563"
            sslport="5564"
            interface="*ALL" backlog="256" buffersend="-1" bufferreceive="-1"
            nodelay="false" timeout="5"/>
        +<access>
        +<mimetype>
        <virtual host="*" active="true">
            +<access>
            +<protect>
            <script>
                <match uri="/axes/dbmservice.jsp"
                    class="com.lansa.jsm.service.HTTPServiceQuery"
                    trace="false" clienttrace="false">
                    <parameter name="autoregister" value="false"/>
                    <parameter name="allow.query.clause.into" value="false"/>
                    <parameter name="database.host" value="APOLLO"/>
                    <parameter name="database.library" value="USRQRYLIB"/>
                    <parameter name="service.host" value="ZEUS"/>
                    <parameter name="service.library" value="QGPL"/>
                </match>
                <match uri="/axes/sfmservice.jsp"
                    class="com.lansa.jsm.service.HTTPServiceSpool"
                    trace="false" clienttrace="false">
                    <parameter name="autoregister" value="false"/>
                    <parameter name="database.host" value="{SERVICEHOST}"/>
                    <parameter name="database.library" value="{SERVICEHOST}"/>
                    <parameter name="service.host" value="ZEUS"/>
                    <parameter name="service.library" value="QGPL"/>
                </match>
                <match uri="/" class="com.lansa.jsm.JSMHTTPServiceFile"
                    trace="false" clienttrace="false">
                    <parameter name="cache.maxage" value="28800"/>
                    <parameter name="cache.maxage.pdf" value="28800"/>
                    <parameter name="cache.maxage.image" value="28800"/>
                </match>
            </script>
            +<mimetype>
        </virtual>
    </instance>
```

Virtual Host Script Configuration Example

```
</configuration>
```

The database locations for DE and WSFM are independent, as illustrated by this example.

Remote service activation

DE and WSFM require their own remote service activation codes and the configuration must include a remote service activation parameter under dbmservice.jsp for DE and under sfmservice.jsp for WSFM.

Table 58 (page 61) shows an example configuration for the remote service activation parameter, where only WSFM is enabled for remote services.

Table 58: Configuring Remote Service Activation

Virtual Host Script Configuration Example – Remote Service Activation

```
<?xml version="1.0" encoding="UTF-8"?>
<configuration>
    <instance name="HTTP Instance" active="true" root="www/instance/htdocs"
            index="index.html">
        <errorlog enabled="true" file="www/instance/logs/error.log"/>
        <accesslog enabled="true" file="www/instance/logs/access.log"/>
        <listen secure="false" store="pki/wwwssl.jks" password="password"
            port="5563"
            sslport="5564"
            interface="*ALL" backlog="256" buffersend="-1" bufferreceive="-1"
            nodelay="false" timeout="5"/>
        +<access>
        +<mimetype>
        <virtual host="*" active="true">
            +<access>
            +<protect>
            <script>
                <match uri="/axes/dbmservice.jsp"
                    class="com.lansa.jsm.service.HTTPServiceQuery"
                    trace="false" clienttrace="false">
                    <parameter name="autoregister" value="false"/>
                    <parameter name="allow.query.clause.into" value="false"/>
                    <parameter name="database.host" value="LOCALHOST"/>
                    <parameter name="database.library" value="AXES"/>
                    <parameter name="service.host" value="LOCALHOST"/>
                    <parameter name="service.library" value="QGPL"/>
                </match>
                <match uri="/axes/sfmservice.jsp"
                    class="com.lansa.jsm.service.HTTPServiceSpool"
                    trace="false" clienttrace="false">
                    <parameter name="autoregister" value="false"/>
                    <parameter name="database.host" value="LOCALHOST"/>
                    <parameter name="database.library" value="AXES"/>
```

Virtual Host Script Configuration Example – Remote Service Activation

```
                <parameter name="service.host" value="LOCALHOST"/>
                <parameter name="service.library" value="QGPL"/>
                <parameter name="service.remote.activation"
                    value="815223DB1E993AEDDC8C84F93A074D61B39522DF"/>
            </match>
            <match uri="/" class="com.lansa.jsm.JSMHTTPServiceFile"
                trace="false" clienttrace="false">
                <parameter name="cache.maxage" value="28800"/>
                <parameter name="cache.maxage.pdf" value="28800"/>
                <parameter name="cache.maxage.image" value="28800"/>
            </match>
        </script>
        +<mimetype>
    </virtual>
  </instance>
</configuration>
```

Reference guide: JSM HTTP server configuration

The reference guide provides explanations of the individual items in the httpd configuration files.

Configuration item reference

Table 59 (page 62) provides explanations of configuration items in the JSM HTTP server httpd configuration file. The items in the table are in name sequence. The configuration item name is the name and position of the parameters in the httpd configuration file. For example, the item named "listen port" is the port on the listen directive.

Table 59: Server Reference: Configuration Item Reference

Configuration Items	Definitions
Access allow	List of addresses, content lengths and/or user agents allowed access to the JSM HTTP server. Example: <allow address="10.2.1.45">
Access deny	List of addresses, content lengths and/or user agents denied access to the JSM HTTP server. Example: <deny address="10.2.1.45">
Access log enabled	Log all access events when access log enabled is "true". Log no access events when access log enabled is "false".
Access log file	Name and location of the access log file. Example: "www/instance/logs/access.log"
Cache maxage	Maximum age for cached files in seconds.
Cache maxage image	Maximum age for cached image files in seconds.

Configuration Items	Definitions
Cache maxage pdf	Maximum age for cached PDF files in seconds.
Database host	Database host is the name of the server where DE and/or WSFM database tables reside. Parameter syntax is: database.host="{SERVICEHOST}" The value derives from the service.host parameter or is overridden by the browser host (i.e. the server name provided as part of the user sign on). database.host="MyServerName" The value derives from the server name enclosed by the double quotes. database.host="LOCALHOST" The default value is LOCALHOST. If the parameter is missing from the configuration, aXes-Cloud assumes the default value.
Database library	Database library is the name of the library in which DE or WSFM database tables reside on the database host. The library must exist on the database host. aXes-Cloud will not create the library. Parameter syntax is: database.library="{SERVICEHOST}" The value derives from the service.host parameter or is overridden by the browser host (i.e. the server name provided as part of the user sign on). database.library="MyLibrary" The value derives from the library name enclosed by the double quotes. Browser host does not override explicitly specified library names. database.library="AXES" The default value is AXES. If the parameter is missing from the configuration, aXes-Cloud assumes the default value.
Error log enabled	Log all errors when error log enabled is "true". Log no errors when error log enabled is "false".
Error log file	Name and location of the error log file. Example: "www/instance/logs/error.log"
Instance active	To operate services supported by the instance, this parameter will always be "true". The JSM HTTP server supports multiple instances and needs this parameter to indicate which instances to activate at run time.
Instance index	Name of the index page. Example: "index.html"

Configuration Items	Definitions
Instance name	Name allocated the JSM HTTP server. Example: "HTTP Instance"
Instance root	Name of the directory from which the documents will be served by the JSM HTTP server. Also known as document root. Example: "www/instance/htdocs"
Listen backlog	The backlog defines the depth of the TCP/IP queue.
Listen buffer receive	TCP/IP receive buffer size in bytes. Special case value "-1" means use operating system default.
Listen buffer send	TCP/IP send buffer size in bytes. Special case value "-1" means use operating system default.
Listen interface	TCP/IP interface address that the JSM HTTP server will bind to and accept connections on. Default value is *ALL *ALL will bind to all interfaces on the server.
Listen no delay	True enables TCP/IP no delay option. False means use the operating system setting for this parameter.
Listen password	Password that will open the store file used for TLS/SSL configuration. Example: "password"
Listen port	TCP/IP port number the server will use to accept connections on.
Listen secure	When true, the JSM HTTP server uses TLS/SSL When false, the JSM HTTP server uses plain sockets.
Listen sslport	TCP/IP port number the server will use to accept connections when using TLS/SSL.
Listen store	Path for the store file that contains the private key and public certificates (used when TLS/SSL enabled). Fully qualified name and location of the store file. Example: "pki/wwwssl.jks"
Listen timeout	Time out count in seconds (integer).
Mimetype map extension	File extension used to identify the MIME type. Example: "png"
Mimetype map type	The type describes the nature of the MIME type. Examples: "image/png"

Configuration Items	Definitions
Protect match authentication	Type of authentication, values are: "basic", "digest" or "basic,digest". Basic: Basic authentication is a concatenation of user name, a colon and the password encoded with the Base64 algorithm. Digest: Digest authentication is an application of MD5 cryptographic hashing of user credentials. It provides stronger encoding than basic authentication.
Protect match realm	Name of the realm used for authentication to parts of the Web site or application. The realm is associated with the matched URIs.
Protect match uri	URIs used to match against requests. When a match occurs the associated realm provides the authentication details.
Protect realm name	Name assigned to a realm.
Protect realm user access	Access is a digest of user information (including the password).
Protect realm user name	User name for authentication.
Script match	The match definitions describe services that the JSM HTTP server will use depending on the match criteria in the URI.
Script match class	Class is the name of a service.
Script match client trace	Use this parameter to trace activity associated with specific clients. Tracing will slow performance. When "true" tracing will occur from the client URI using the "?trace=true" query string parameter. The value "false" disables tracing.
Script match trace	Use this parameter to trace activity associated with all clients. Broader scope than client trace. The value "true" enables tracing for all services and user agents that match the match URI. The value "false" disables tracing.
Script match URI	The match URI is the match criteria the JSM HTTP server uses to determine the services to use.
Service allow	The service allow directive defines user identifications (or profiles) that are permitted to sign on to a server.
Service deny	The service deny directive defines user identifications (or profiles) that are not permitted (or denied access) to sign on to a server.

Configuration Items	Definitions
Service host	Service host is the name the server that runs the DE or WSFM services. For DE, service host defines the server where the tables used in SQL queries reside and where queries run. For WSFM service host defines the server where the spooled files reside and WSFM searches run. The service host is a corporate server connected to the cloud gateway server or may be the cloud gateway server itself. Parameter syntax is: service.host ="MyServer" The value derives from the server name enclosed by the double quotes. service.host ="LOCALHOST" The value derives from the server name enclosed by the double quotes. The default value is LOCALHOST.
Service library	The service library is the library name aXes-Cloud passes to the JDBC driver. Parameter syntax is: service.library ="QGPL" The value derives from the library name enclosed by the double quotes. The default value is QGPL. If the parameter is missing aXes-Cloud assumes the default value.
Virtual access allow	List of addresses, content lengths and/or user agents allowed access to the JSM HTTP server. Example: <allow useragent="safari">
Virtual access deny	List of addresses, content lengths and/or user agents denied access to the JSM HTTP server. Example: <deny useragent="safari">
Virtual active	When the value is "true" this virtual host is active. When the value is "false" the virtual host is inactive. To operate services the value must be "true". Services configured in the virtual host are unavailable when the virtual host is inactive.
Virtual host	Name of the virtual host to match with the HTTP host property. This allows multi homing. HTTP requests can be directed to different virtual host sections of the configuration in the server instance. If a virtual host is not found then the connection request is rejected. The special case value "*" accepts requests from any HTTP host. Specific names take precedence.
Virtual mimetype map extension	File extension used to identify the MIME type. Example: "png"

Configuration Items	Definitions
Virtual mimetype map type	The type describes the nature of the MIME type. Examples: "image/png"

MIME types

MIME type describes the nature of content of file for the JSM HTTP server. Table 60 (page 67) provides examples of MIME types.

Table 60: JSM HTTP Server Reference: MIME type Examples

Extension	Type
css	text/css; charset=utf-8
gif	image/gif
htm	text/html; charset=utf-8
html	text/html; charset=utf-8
ico	image/x-icon
jpeg	image/jpeg
jpg	image/jpeg
js	application/x-javascript; charset=utf-8
pdf	application/pdf
png	image/png
tiff	image/tiff
xls	text/xls; charset=utf-8
xml	text/xml; charset=utf-8

Access allow and deny directives

Table 61 (page 67) shows examples of the access allow/deny directive for addresses.

Table 61: Server Reference: Access Allow and Deny Addresses

Allow / Deny	Syntax and Examples
Allow any address	<allow address="*">
Allow addresses in a range	<allow address="10.2.1">
Allow a specific address	<allow address="10.2.1.45">
Deny any address	<deny address="*">
Deny addresses in a range	<deny address="10.2.1">
Deny a specific address	<deny address="10.2.1.45">

Table 62 (page 68) shows examples of the access allow/deny directive for content length.

Table 62: Server Reference: Access Allow and Deny Content Length

Allow / Deny	Syntax and Examples
Allow access for content less than or equal to the specified length	<allow contentlength="4096">
Zero content length is a special case to allow access for no content connections from the browser	<allow contentlength="0">
Deny access for content greater than the specified length	<deny contentlength="4096">

User agents are applications or services that act on behalf of the user. When a user requests a web page (or URL), the browser acts as a user agent by sending the page request to the JSM HTTP server. Examples of user agents are browsers, web crawlers, link checkers, bots and email clients. Access allow and deny directives control which user agents the JSM HTTP server will allow or deny access.

Table 63 (page 68) shows examples of user agents and the syntax of the allow access and deny access directives.

Table 63: Server Reference: Access Allow and Deny User Agents

Allow / Deny	Syntax and Examples
Allow access for any user agent	<allow useragent="*"
Allow access if no user agent provided	<allow useragent="?"
Allow access to Chrome	<allow useragent="chrome"
Allow access to the Internet Explorer	<allow useragent="explorer"
Allow access to Firefox	<allow useragent="firefox"
Allow access to Safari	<allow useragent="safari"
Deny access for any user agent	<deny useragent="*"
Deny access if no user agent provided	<deny useragent="?"
Deny access to Chrome	<deny useragent="chrome"
Deny access to Internet Explorer	<deny useragent="explorer"
Deny access to Firefox	<deny useragent="firefox"
Deny access to Safari	<deny useragent="safari"

The evaluation of the directives starts with the first item in the list and continues until it finds a true condition. Any combinations of address, user agent and content length are acceptable. However, it is possible to negate the effect of a directive by its position in the list. For example, placing an allow any user agent (<allow useragent="*"/>) ahead of a deny for a specific user agent (<deny useragent="webos"/>) will cause the JSM HTTP server to ignore the deny directive.

Table 64 (page 68) presents lists of user agents.

Table 64: Server Reference: Sample Lists of User Agents

Browser User Agents
android

Bots and Device User Agents
googlebot

Browser User Agents
chrome
explorer
firefox
opera
safari
webos

Bots and Device User Agents
googletoolbar
ipad
iphone
lansaua
msnbot
yahoobot

Sample configurations

Data Explorer

Table 65 (page 69) shows the default httpd configuration file including MIME types, allow and deny directives.

Table 65: Server Reference: Sample httpd Configuration for DE

Sample Configuration for DE

```
<?xml version="1.0" encoding="UTF-8"?>
<configuration>
    <instance name="HTTP Instance" active="true" root="www/instance/htdocs"
            index="index.html">
        <errorlog enabled="true" file="www/instance/logs/error.log"/>
        <accesslog enabled="true" file="www/instance/logs/access.log"/>
        <listen secure="false" store="pki/wwwssl.jks" password="password"
            port="5563"
            sslport="5564"
            interface="*ALL" backlog="256" buffersend="-1" bufferreceive="-1"
            nodelay="false" timeout="5"/>
        <access>
            <allow address="*"/>
        </access>
        <mimetype>
            <map extension="png"  type="image/png"/>
            <map extension="gif"  type="image/gif"/>
            <map extension="jpg"  type="image/jpeg"/>
            <map extension="jpeg" type="image/jpeg"/>
            <map extension="tiff" type="image/tiff"/>
            <map extension="ico"  type="image/x-icon"/>
            <map extension="pdf"  type="application/pdf"/>
            <map extension="css"  type="text/css; charset=utf-8"/>
            <map extension="xsl"  type="text/xls; charset=utf-8"/>
            <map extension="xml"  type="text/xml; charset=utf-8"/>
            <map extension="htm"  type="text/html; charset=utf-8"/>
            <map extension="html" type="text/html; charset=utf-8"/>
            <map extension="js"   type="application/x-javascript; charset=utf-8"/>
```

Sample Configuration for DE

```
        </mimetype>
        <virtual host="*" active="true">
            <access>
                <deny useragent="webos"/>
                <deny useragent="opera"/>
                <deny useragent="android"/>
                <deny useragent="ipod"/>
                <allow useragent="*"/>
                <allow useragent="?"/>
            </access>
            <protect>
                <realm name="Area 51">
                    <user name="user" access="bb644a9819425bfd8586b408896a1031"/>
                </realm>
                <match uri="/restricted" realm="Area 51" authentication="basic,digest"/>
            </protect>
            <script>
                <match uri="/axes/dbmservice.jsp"
                    class="com.lansa.jsm.service.HTTPServiceQuery"
                    trace="false" clienttrace="false">
                    <parameter name="autoregister" value="false"/>
                    <parameter name="allow.query.clause.into" value="false"/>
                    <parameter name="database.host" value'"LOCALHOST"/>
                    <parameter name="database.library" value="AXES"/>
                    <parameter name="service.host" value'"LOCALHOST"/>
                </match>
                <match uri="/" class="com.lansa.jsm.JSMHTTPServiceFile"
                    trace="false" clienttrace="false">
                    <parameter name="cache.maxage" value="28800"/>
                    <parameter name="cache.maxage.pdf" value="28800"/>
                    <parameter name="cache.maxage.image" value="28800"/>
                </match>
            </script>
            <mimetype>
                <map extension="pdf" type="application/pdf"/>
                <!—
                    Defaults to instance mimetype
                ->
            </mimetype>
        </virtual>
    </instance>
</configuration>
```

Web Spooled File Manager

Table 66 (page 71) shows a sample httpd configuration file including MIME types, allow and deny directives.

Table 66: Server Reference: Sample httpd Configuration for WSFM

Sample Configuration for WSFM

```
<?xml version="1.0" encoding="UTF-8"?>
<configuration>
    <instance name="WebServer" active="true" root="www/instance/htdocs"
            index="index.html">
        <errorlog enabled="true" file="www/instance/logs/error.log"/>
        <accesslog enabled="true" file="www/instance/logs/access.log"/>
        <listen secure="false" store="pki/wwwssl.jks" password="password"
            port="5563"
            sslport="5564"
            interface="*ALL" backlog="256" buffersend="-1" bufferreceive="-1"
            nodelay="false" timeout="5"/>
        <access>
            <allow address="*"/>
        </access>
        <mimetype>
            <map extension="png"  type="image/png"/>
            <map extension="gif"  type="image/gif"/>
            <map extension="jpg"  type="image/jpeg"/>
            <map extension="jpeg" type="image/jpeg"/>
            <map extension="tiff" type="image/tiff"/>
            <map extension="ico"  type="image/x-icon"/>
            <map extension="pdf"  type="application/pdf"/>
            <map extension="css"  type="text/css; charset=utf-8"/>
            <map extension="xsl"  type="text/xls; charset=utf-8"/>
            <map extension="xml"  type="text/xml; charset=utf-8"/>
            <map extension="htm"  type="text/html; charset=utf-8"/>
            <map extension="html" type="text/html; charset=utf-8"/>
            <map extension="js"   type="application/x-javascript; charset=utf-8"/>
        </mimetype>
        <virtual host="*" active="true">
            <access>
                <deny useragent="webos"/>
                <deny useragent="opera"/>
                <deny useragent="android"/>
                <deny useragent="ipod"/>
                <allow useragent="*"/>
                <allow useragent="?"/>
            </access>
            <protect>
                <realm name="Area 51">
                    <user name="user" access="bb644a9819425bfd8586b408896a1031"/>
                </realm>
                <match uri="/restricted" realm="Area 51" authentication="basic,digest"/>
            </protect>
```

Sample Configuration for WSFM

```
            <script>
                <match uri="/axes/sfmservice.jsp"
                    class="com.lansa.jsm.service.HTTPServiceSpool"
                    trace="false" clienttrace="false">
                    <parameter name="autoregister" value="false"/>
                    <parameter name="database.host" value="LOCALHOST"/>
                    <parameter name="database.library" value="AXES"/>
                    <parameter name="service.host" value="LOCALHOST"/>
                </match>
                <match uri="/" class="com.lansa.jsm.JSMHTTPServiceFile"
                    trace="false" clienttrace="false">
                    <parameter name="cache.maxage" value="28800"/>
                    <parameter name="cache.maxage.pdf" value="28800"/>
                    <parameter name="cache.maxage.image" value="28800"/>
                </match>
            </script>
            <mimetype>
                <map extension="pdf" type="application/pdf"/>
            </mimetype>
        </virtual>
    </instance>
</configuration>
```

Appendices

Glossary

Table 67 (page 73) presents definitions for abbreviations and terms used in this guide.

Table 67: Glossary of Abbreviations and Terms

Abbreviations and Terms	Definitions and Explanations
aXes-Cloud	aXes-Cloud refers to the software and configuration items that provide the cloud services in aXes.
Browser host	The browser host is a term that describes the server name in a qualified user identification. A qualified user identification is: [server name]/[user identification] Or [browser host]/[user identification] Browser host and server name are equivalent.
CCSID	Coded Character Set Identifier
Cloud gateway	The IBM i server that hosts aXes and aXes-Cloud software and configuration.
Configuration file	The term configuration file refers to the copy of the httpd.xml file used to configure aXes-Cloud.
Corporate server	Corporate server is the term used in this guide to refer to IBM i servers that host corporate applications. You do not install aXes-Cloud on corporate servers.
DBCS	Double Byte Character Set
DE	Data Explorer
Directives	Configuration directives are the parameters and settings that control the behaviour of the JSM HTTP server.
DMZ	A DMZ (demilitarised zone) is a sub-network placed between a corporate internal network (for example, a local area network) and external networks such as the Internet. The DMZ provides a layer of insulation between the internal and external networks. External parties access the DMZ rather than the internal network.

Abbreviations and Terms	Definitions and Explanations
Qualified user identification	A qualified user identification includes both a server name and a user identification (or profile). The syntax is [server name]/[user identification] MyServer/MyUserId is an example. MyServer is the server name (or browser host), i.e. the corporate server where DE queries will run and WSFM will manage spooled files.
IFS	The Integrated File System is a part of the IBM i operating system. It supports stream input/output and storage management capabilities like personal computer and UNIX operating systems.
Internet Media Type	Internet Media Type (IMT) is an identifier for file formats on the Internet. An example is the IMT for the Portable Document Format: application/pdf See also MIME type.
JSM	Java Services Manager
MIME	Multipurpose Internet Mail Extensions MIME is an Internet standard that describes content. An example is PNG for Portable Network Graphic files. See also Internet Media Types.
Policies	Policies (or rules) govern the behaviour of aXes-Cloud. An example of a policy is prohibiting sign on for user identifications beginning with the letter Q. Administrators can alter aXes-Cloud behaviour by changing configuration parameters.
Realm	A realm is a database containing a list of valid users of a web application. The user information in the database is user name, password and a set of roles associated with the user.
SBCS	Single Byte Character Set
Service host	Service host refers to a corporate IBM i server connected to the cloud gateway server.
Sign on	Users sign on to corporate IBM i servers and aXes-Cloud.
System directory	System directory is a synonym for system folder.
System folder	The location of the system folder is: axes/jsm/instance/system
WS	Abbreviation of Web Spooled File Manager
WSFM	Web Spooled File Manager

Assumed and prerequisite knowledge

Table 68 (page 75) defines the prerequisite knowledge needed to use the guide.

Table 68: Assumed and Prerequisite Knowledge

Subject Matter	Explanations
aXes administrator knowledge	This guide assumes that, as an administrator, you understand how to install and configure aXes features and services including aXes Terminal Server, aXes Data Explorer and aXes Web Spooled File Manager.
aXes installation	This guide does not include information about aXes installation, except where it pertains to configuring aXes-Cloud. Refer to the aXes guides for detailed installation instructions.
aXes configuration	This guide does not include information about aXes configuration, except where it pertains to configuring aXes-Cloud. Refer to the aXes guides for detailed configuration instructions.

Example configurations

Database.host, database.library, service.host

Table 69 (page 76) illustrates the resolution of parameter values (database.host, database.library and service.host) based on a deployment of two servers. aXes-Cloud resides on a server named Gateway and the second server is a corporate server named Apollo.

Table 69: Example database.host, database.library, service.host with one corporate server

Configuration file	User signs on as	Database resides on	Database library resolves to	Database library derived from	Queries run on	Where queries run derived from
database.host = "LOCALHOST" database.library = "AXES" service.host="LOCALHOST"	MyUserId	Gateway	AXES	database.library	Gateway	service.host
	APOLLO/MyUserId	Gateway	AXES	database.library	APOLLO	browser host
database.host = "LOCALHOST" database.library = "{SERVICEHOST}" service.host= "APOLLO"	MyUserId	Gateway	APOLLO	service.host	APOLLO	service.host
	APOLLO/MyUserId	Gateway	APOLLO	browser host	APOLLO	browser host
database.host = "APOLLO" database.library = "APOLLO" service.host= "APOLLO"	MyUserId	APOLLO	APOLLO	database.library	APOLLO	service.host
	APOLLO/MyUserId	APOLLO	APOLLO	database.library	APOLLO	browser host
database.host = "APOLLO" database.library = "{SERVICEHOST}" service.host= "APOLLO"	MyUserId	APOLLO	APOLLO	service.host	APOLLO	service.host
	APOLLO/MyUserId	APOLLO	APOLLO	browser host	APOLLO	browser host

In Table 69 (page 76), LOCALHOST refers to the server named Gateway. The substitution value for {SERVICEHOST} is the server name provided by the user when signing on with a qualified user identification (also known browser host); in this case the server name is Apollo.

Table 70 (page 77) illustrates the resolution of parameter values (database.host, database.library and service.host) based on a deployment of three servers. aXes-Cloud resides on a server named Gateway, the second server is a corporate server named Apollo and the third server is ZEUS.

Table 70: Example database.host, database.library, service.host with two corporate servers

Configuration file	User signs on as	Database resides on	Database library resolves to	Database library derived from	Queries run on	Where queries run derived from
database.host = "LOCALHOST"	MyUserId	Gateway	AXES	database.library	APOLLO	service.host
database.library = "AXES"	APOLLO/MyUserId	Gateway	AXES	database.library	APOLLO	browser host
service.host= "APOLLO"	ZEUS/MyUserId	Gateway	AXES	database.library	ZEUS	browser host
database.host = "LOCALHOST"	MyUserId	Gateway	APOLLO	service.host	APOLLO	service.host
database.library = "{SERVICEHOST}"	APOLLO/MyUserId	Gateway	APOLLO	browser host	APOLLO	browser host
service.host= "APOLLO"	ZEUS/MyUserId	Gateway	ZEUS	browser host	ZEUS	browser host
database.host = "APOLLO"	MyUserId	APOLLO	APOLLO	service.host	APOLLO	service.host
database.library = "APOLLO"	APOLLO/MyUserId	APOLLO	APOLLO	database.library	APOLLO	browser host
service.host= "APOLLO"	ZEUS/MyUserId	APOLLO	APOLLO	database.library	ZEUS	browser host
database.host = "APOLLO"	MyUserId	APOLLO	APOLLO	service.host	APOLLO	service.host
database.library = "{SERVICEHOST}"	APOLLO/MyUserId	APOLLO	APOLLO	browser host	APOLLO	browser host
service.host= "APOLLO"	ZEUS/MyUserId	APOLLO	ZEUS	browser host	ZEUS	browser host

In Table 70 (page 77), LOCALHOST refers to the server named Gateway. The substitution value for {SERVICEHOST} is the server name provided by the user when signing on with a qualified user identification (also known browser host); in this case the server name resolves to Apollo or Zeus.