Perceptive Integration Server on WebSphere

Installation and Setup Guide

Version: Foundation EP3

Written by: Product Knowledge, R&D
Date: May 2021

Hyland™

Copyright

Information in this document is subject to change without notice. The software described in this document is furnished only under a separate license agreement and may be used or copied only according to the terms of such agreement. It is against the law to copy the software except as specifically allowed in the license agreement. This document or accompanying materials contains certain information which is confidential information of Hyland Software, Inc. and its affiliates, and which is subject to the confidentiality provisions agreed to by you.

All data, names, and formats used in this document's examples are fictitious unless noted otherwise. Complying with all applicable copyright laws is the responsibility of the user. Without limiting the rights under copyright law, no part of this document may be reproduced, stored in or introduced into a retrieval system, or transmitted in any form or by any means (electronic, mechanical, photocopying, recording, or otherwise), or for any purpose, without the express written permission of Hyland Software, Inc. or one of its affiliates.

Hyland® and Hyland Software®, as well as Hyland product names, are registered and/or unregistered trademarks of Hyland Software, Inc. and its affiliates in the United States and other countries. All other trademarks, service marks, trade names and products of other companies are the property of their respective owners.

© 2021 Hyland Software, Inc. and its affiliates. All rights reserved.

Table of Contents

Overview **4**
Prepare to install **4**
System Requirements 4
Compatibility 4
Licensing requirements 4
Install Integration Server using WebSphere **5**
Download the Integration Server files 5
Extract Integration Server 5
Set up Integration Server in WebSphere Management Console *5*
Set up Integration Server as an enterprise application in WebSphere Classic *6*
Setting up SSL *8*
Deploy Integration Server in WebSphere Liberty 8
Configure Integration Server *8*
Setting up SSL *9*
Configure Integration Server for single sign-on 9
Export a certificate *10*
Import a certificate into Perceptive Content Server *10*
Configure Integration Server to use a keystore *11*
Configure Integration Server for OpenID Connect authentication 11
Configure an OpenID Provider *11*
Configure Integration Server to use OpenID Connect Login Profiles *11*
Configure Perceptive Content Server to use an OpenID Connect Login Profile *12*
Test the OpenID Connect *14*
SSL/TLS considerations 14
Configure Integration Server for agent token authentication 14
Import a token signing certificate *14*
Configure Integration Server for agent token authentication *15*
Configure agent token authentication for Forms Server *15*
Appendix A: integrationserver.ini file options **16**
Appendix B: X.509 LDAP authentication translator **30**
Example configuration *31*

Overview

Perceptive Integration Server is a service that provides a window into Perceptive Content functionality. Integration Server exports a rich set of web services that enable customers and third-party application developers to embed Perceptive Content functionality, such as document management and workflow, directly into their applications.

For product technical specifications and system requirements, see the *Product Technical Specifications* for your product version.

Prepare to install

To install Integration Server, use the platform-specific steps for your operating system. The Integration Server Installation section of this document provides all installation methods.

System Requirements

Integration Server follows the same minimum system requirements as Perceptive Content Server. For product technical specifications and system requirements, see the *Technical Specifications* documentation available at docs.hyland.com.

Note Before you install Integration Server, you must have WebSphere successfully installed as the web application server.

Compatibility

Integration Server 7.2 is not backwards compatible with earlier versions of ImageNow or Perceptive Content. Your Integration Server version must be the same version as Perceptive Content Server.

Licensing requirements

Integration Server is licensed on a transaction basis. The use of the agent requires a license that provides a certain number of transactions over a period.

If you are installing Integration Server for use with an ImageNow Interact product that is not licensed by transactions, you only need the license for that specific Interact product. If the Interact product is licensed by transactions, you must have an Integration Server license as well as the specific Interact license. For more licensing information, see that product's installation and setup guide.

The following table contains three example scenarios for Integration Server licensing.

Scenario	Required Licenses	Number of licenses	Description
User developed application	Integration Server license Integration Server transaction pack license	A minimum of one Integration Server license. A minimum of one Integration Server transaction pack license.	The Integration Server transaction pack license must contain enough transactions to cover the number of calls you expect your users to make collectively per transaction period and 300 transactions per hour. For example, if 30 users make 10 calls per hour each, you need at least two transaction packs per hour specified on the license.

Scenario	Required Licenses	Number of licenses	Description
Using an Integration Server compatible product that uses a seat or concurrent license.	Specific license for that product only.	See the product's documentation.	You do not need the Integration Server license or Integration Server transaction pack license, because all of the calls are covered by the product's specific license. Note Integration Server must be installed.
Using an Integration Server compatible Interact product that uses a transaction pack scheme.	Integration Server license Transaction pack license for the product	One Integration Server license. A minimum of one license for the product.	The license for the product must have enough transaction packs for the number of transactions your users need to make. Note For this scenario, you do not need an Integration Server transaction pack license.

Install Integration Server using WebSphere

This document assumes you are installing Perceptive Integration Server for the first time. The steps for installing are listed below.

- Download the Integration Server files
- Extract Integration Server
- Deploy Integration Server in WebSphere Liberty
- Configure single sign-on

Download the Integration Server files

To obtain Perceptive product installation files, contact the Hyland Software Technical Support group. For a list of Technical Support phone numbers, go to hyland.com/pswtscontact.

Extract Integration Server

Use the following procedures to extract the downloaded Integration Server web application in IBM WebSphere Application Server.

Set up Integration Server in WebSphere Management Console

1. Open the **WebSphere Management Console**.
2. Ensure WebSphere is running and navigate to **Administrative Console**. This defaults to **<serverName>:9060/ibm/console** in your browser.
3. In the **Login** dialog box, in the **User ID** box, type a user ID, and then click **OK**.
4. On the left side of the window, expand the **Applications** node, and then click the **New Application** hyperlink.
5. On the right side of the window, under **Install a New Application**, click **New Enterprise Application**.
6. In the **Preparing for the Application installation** window, enter the location of the downloaded **integrationserver** WAR file and click **Next**.

7. Under **How do you want to install the application?**, select either **Fast Path** or **Detailed**.
8. Select the **Generate Default Bindings** check box and then click **Next**.
9. In the **Install New Application** window, click the **Step 3** hyperlink and in the **Context Root** parameter box, type **integrationserver**.
10. Click **Next** and then click **Finish**.
11. The **Installing** window confirms that the WAR file was installed. At the bottom of the window, click the **Save** link.

Set up Integration Server as an enterprise application in WebSphere Classic

1. In the main **Administrative Console**, on the left side of the window, expand the **Servers** node, and then expand **Server Types**.
2. Click the **WebSphere application servers** hyperlink.
3. In the **Application servers** window, click the **<servername>** application hyperlink.
4. On the right side, under **Server Infrastructure**, expand **Java and Process Management** and click the **Process Definition** hyperlink.
5. In the **Process definition** window, click the **Java Virtual Machine** hyperlink.
6. On the **Configuration** tab, under **General Properties**, add the following to the **Generic JVM** arguments.

```
-Dcom.ibm.websphere.webservices.DisableIBMJAXWSEngine=true
```

7. To save to the master configuration, click **Apply** and then click the **Save** hyperlink.
8. In the **Process Definition** window, click **OK**.
9. At the top of the **Application server** window, in the **Messages** section, click **Save**.
10. Expand **Applications** > **Application Types** and click the **WebSphere enterprise applications** hyperlink.
11. Click the **integrationserver.war** hyperlink.
12. On the **Configuration** tab, under **Modules**, click the **Manage Modules** hyperlink.
13. In **Manage Modules**, click the **integrationserver** hyperlink.
14. Under **General Properties**, select the **Class loader order** option of **Classes loaded to local class loader first (parent last)** option.
15. In the **Messages** window, click **Save** to save to the master configuration.
16. Configure the integrationserver.ini settings. To configure the settings, see the Configure Integration Server section.
17. Open a file browser window and navigate to the **{$WEBSPHERE_HOME}/<application server>/profiles/<application server profile>/<application server node>/integrationserver.ear/integrationserver.war/WEB-INF/lib** directory.
18. Remove the following files from the directory.
    - Saaj-api-1.3.4.jar
    - saaj-impl-1.3.12.jar

- serializer-2.7.1.jar
- xalan-2.7.1.jar
- xercesImpl-2.8.1.jar
- xml-apis-1.3.04.jar
- xmlbeans-2.5.0.jar

19. In the **Enterprise Applications** window, Integration Server is disabled. To enable Integration Server, select the **integrationserver.war application** check box and then click **Start**.
20. Restart the web application server.

Configure Integration Server

Within integrationserver.ini file settings, you can configure such things as the IP address, port, and logging for Integration Server. You can use the settings in the integrationserver.ini file to customize how Integration Server works.

Note When you make changes to Integration Server's INI file, restart WebSphere to apply the changes.

In most cases, you need to modify only the IP address and port for the Perceptive Content Server. Integration Server communicates with Perceptive Content Server using these settings.

1. In the **[path]/{$WEBSPHERE_HOME}/<application server>/profiles/<application server>/<application server node>/integrationserver.ear/integrationserver.war/WEB-INF/** directory, select the **integrationserver.ini** file and open it in a text editor.
2. In the **[General]** section, change the settings as needed using the following substeps.
    1. To change the value for the server address, provide an IP address for the **server.ip.address** setting. The default is **127.0.0.1**.
    2. To change the server port, enter a value for the port for the **server.port** setting. The default is **6000**.
3. In the **[Logging]** section, set the logging for the following settings. A detailed explanation of each setting is provided in the **integrationserver.ini** setting table in Appendix A.
    - **log.directory**. Specifies the path to the directory that contains the log files. The default is **logs**.
    - **log.rolling.interval**. This setting specifies the amount of time data is stored in a log file before the data is archived to another file. This keeps log files from becoming too large. The default is **DAY**.
    - **log.max.history**. This setting is a threshold for the maximum number of log files that are retained per day. One file is created each day, hour, or month depending on the log.rolling.interval setting. The default is **28**.

4. In the [**Capture]** section, set the directory to use for document captures and transfers. For example, **capture.work.directory=[*drive*:]/inserver/temp**. The default is **temp**.
5. Save and close the file.

Setting up SSL

If you want to configure SSL to use with Integration Server, you must do so on your WebSphere web application server. SSL configuration cannot be completed within Integration Server. For more information, see the WebSphere Portal help documentation on the IBM website.

Deploy Integration Server in WebSphere Liberty

To deploy Integration Server in IBM WebSphere Application Server Liberty, complete the following steps.

1. Copy the **integrationserver** WAR file to the **[path]/${WEBSPHERE_LIBERTY_HOME}/ /usr/servers/<servername>/apps** directory, where <servername> is the Liberty Server instance name.
2. Extract the **integrationserver** WAR file into the **[path]/${WEBSPHERE_LIBERTY_HOME}/usr/servers/<servername>/apps/** directory.
3. Delete the **integrationserver** WAR file.
4. Rename the **integrationserver_<buildnumber>** folder to **integrationserver.war**.

Note After deploying the WAR file, you must make some changes to the integrationserver.ini configuration file.

Configure Integration Server

Within integrationserver.ini file settings, you can configure such things as the IP address, port, and logging for Integration Server. You can use the settings in the integrationserver.ini file to customize how Integration Server works.

Note When you make changes to Integration Server's INI file, restart WebSphere for the new changes to be picked up.

In most cases, you need to modify only the IP address and port for the Perceptive Content Server. Integration Server communicates with the Perceptive Content Server using these settings.

1. In the **[path]/${WEBSPHERE_HOME}/Liberty/usr/servers/<servername>/apps/integrationserver.war/ WEB-INF/** directory, select the **integrationserver.ini** file and open it in a text editor.
2. In the **[General]** section, change the settings as needed using the following substeps.
    1. To change the value for the server address, provide an IP address for the **server.ip.address** setting. The default is **127.0.0.1**.
    2. To change the server port, enter a value for the port for the **server.port** setting. The default is **6000**.

3. In the **[Logging]** section, set the logging for the following settings. A detailed explanation of each setting is provided in the **integrationserver.ini** setting table in Appendix A.
   - **log.directory**. Specifies the path to the directory that contains the log files. The default is **logs**.
   - **log.rolling.interval**. This setting specifies the amount of time data is stored in a log file before the data is archived to another file. This keeps log files from becoming too large. The default is **DAY**.
   - **log.max.history**. This setting is a threshold for the maximum number of log files that are retained per day. One file is created each day, hour, or month depending on the log.rolling.interval setting. The default is **28**.
4. In the [**Capture]** section, set the directory to use for document captures and transfers. For example, **capture.work.directory=[*drive*:]/inserver/temp**. The default is **temp**.
5. Save and close the file.
6. Navigate to the **[path]/${WEBSPHERE_LIBERTY_HOME}/usr/servers/<servername>** directory and using a text editor, open the **server.xml** file.
7. Under the **server** node, add the following elements.

```
<application id="integrationserver" location="integrationserver.war"
name="integrationserver" />
```

   Note If you deployed the WAR file to a directory other than the suggested directory, change the **location** attribute in the above application element to reflect the directory you specified.
8. In the text editor, save the file.

Setting up SSL

If you want to configure SSL to use with Integration Server, you must do so on your WebSphere web application server. SSL configuration cannot be completed within Integration Server. For more information, see the WebSphere Portal help documentation on the IBM website.

Configure Integration Server for single sign-on

Single sign-on (SSO) allows users to access Integration Server by authenticating with an external identity provider. Currently, Perceptive Experience is the only Perceptive product that supports Integration Server's SSO configuration.

Note When single sign-on is enabled, your single sign-on provider supplies authentication information. Users cannot log in to Integration Server anonymously.

Generate a keystore and key pair

To establish trust between Integration Server and Perceptive Content Server, you must generate a keystore and a key pair. To generate a keystore and a key pair, complete the following steps.

Note Perform these steps on your Integration Server machine.

1. Open a command window and navigate to the bin directory of your JRE installation.
2. Enter the following command and press `ENTER`.

```
keytool -genkeypair -alias [alias] -keyalg [RSA] -keystore [path to keystore] -keysize [key size]
```

Setting	Options	Description
-alias	Any valid alias.	Specifies the identifier associated with the key pair to be generated in the output key store.
-keyalg	RSA	Specifies the algorithm to be used to generate the key pair.
-keystore	Any valid keystore path.	Specifies the keystore path. **Note** Only JKS keystores are supported.
-keysize	Any valid key size.	Specifies the size of each key to be generated. For example, 512, 1024, 2048.

Example

```
keytool -genkeypair -alias PSWProd1 -keyalg RSA -keystore C:\Keystores\PSWProd1.jks -keysize 1024
```

Export a certificate

After generating a keystore and a key pair, a certificate is created. To export your certificate to be imported into Perceptive Content Server, complete the following steps.

1. Open a **Command Prompt** window and navigate to the bin directory of your JRE installation.
2. Enter the following command.

```
keytool -exportcert -keystore [path to keystore] -alias [alias] -file [certificate file to create]
```

Example

```
keytool -exportcert -keystore C:\Keystores\PSWProd1.jks -alias PSWProd1 -file C:\Keystores\PSWProd1.cer
```

Import a certificate into Perceptive Content Server

To import a certificate into Perceptive Content Server, complete the following steps.

1. Copy the certificate and paste it into the **etc** directory on the Perceptive Content Server machine.
2. Open a **Command Prompt** window and navigate to the **bin/bin64** directory.
3. Enter the following command.

```
intool --cmd import-cert --file [drive]:\{certificate location} --type pki-sso
```

Configure Integration Server to use a keystore

To configure Integration Server to use a keystore, complete the following steps.

1. In the **WEB-INF** folder of your Integration Server deployment, open the **integrationserver.ini** file in a text editor.
2. Set **sso.enabled** to **TRUE**.
3. Set **sso.keystore.location** to **C:/Keystores/PSWProd1.jks**.
4. **Optional** Enter the keystore password using the **sso.keystore.password** setting to verify the integrity of the keystore.
5. Enter the key pair alias using the **sso.privatekey.alias** setting.
6. Enter the key password using the **sso.privatekey.password** setting.
7. Set the **sso.authenticator.class** to the fully qualified class used to determine authentication from an external provider.

   Note You can use the built-in authentication class **com.imagenow.authentication.translator.HTTPHeaderTranslator** or create your own authentication class.
8. Configure the authentication class settings.

   Note If using the built-in authentication class, set the **sso.httpheader.name** value to the authorized user header.

Configure Integration Server for OpenID Connect authentication

You can configure Integration Server to support the authorization code flow of OpenID Connect as a method of single sign-on. Configuration for OpenID Providers is managed using Perceptive Content OpenID Connect login profiles. You can configure multiple Perceptive Content OpenID Connect login profiles to support different Perceptive client applications. Each login profile has its own configuration the system can used to integrate with different OpenID Providers, customize redirect behaviors, and manage security policies. We recommend that each login profile represents a unique OpenID client that is configured on the OpenID Provider.

Configure an OpenID Provider

See the OpenID Provider's documentation for details on OpenID client registration. There should be a unique OpenID client for each Perceptive client application that will use OpenID connect authentication. During OpenID client registration you need to register the redirect URIs for Integration Server callbacks for each login profile you want to configure. For example, if you are registering a login profile named **hylandexperience**, the endpoint you need to register for the redirect URI should be **https://example.integrationserver.url/integrationserver/v1/sso/callback/hylandexperience**. You must also allow at a minimum the **openid** scope and will also a **client id** and a **client secret** to configure a login profile in Integration Server. See Appendix A: integrationserver.ini configuration for more information on OpenID login profile configuration options.

Configure Integration Server to use OpenID Connect Login Profiles

To configure Integration Server to use OpenID Connect with login profiles, complete the following steps.

1. In the **WEB-INF** folder of your Integration Server deployment, open the **integrationserver.ini** file in a text editor.

2. In the **[OpenID Connect Login Profiles]** section, set **sso.openid.login.enabled** to **TRUE**.
3. Append a profile name into the **sso.openid.profiles** setting for each profile you would like to configure.
4. Enter the client id configured with the OpenID Provider for each profile using the **sso.openid.profile.[profileName].client.id** setting.
5. Enter the OpenID Provider's authorization endpoint URL for each profile using the **sso.openid.profile.[profileName].authorization.endpoint** setting.
6. For web-based Perceptive clients, such as Perceptive Experience, enter a redirect URI for successful authentication attempts using the **sso.openid.profile.[profileName].post.login.redirect.uri** setting.
7. For web-based Perceptive clients, enter a redirect URI for unsuccessful authentication attempts using the **sso.openid.profile.[profileName].error.redirect.uri** setting.
8. **Optional** Configure any optional login profile settings. A detailed explanation of each login profile setting is provided in the inserver.ini OpenID Connect Login Profiles settings topic in the Content System Online Help.

Example

```
[OpenID Connect Login Profiles]
sso.openid.login.enabled=TRUE
sso.openid.profiles=hylandexperience
sso.openid.profile.hylandexperience.client.id=AAA3983D-9632-4F66-828C-
71EC5535C0BA
sso.openid.profile.hylandexperience.authorization.endpoint=https://sample.onb
ase.com/IdentityProvider/connect/authorize
sso.openid.profile.hylandexperience.post.login.redirect.uri=https://sample.on
base.com/hylandexperience/#oidc/callback
sso.openid.profile.hylandexperience.error.redirect.uri=https://sample.onbase.
com/hylandexperience/#oidc/callback?error={error}
```

Configure Perceptive Content Server to use an OpenID Connect Login Profile

To configure Perceptive Content Server to support OpenID Connect through login profiles, complete the following steps.

1. In the **etc** folder of your Perceptive Content Server deployment, open the **inserver.ini** file in a text editor.
2. Create the **[OpenID Connect Login Profiles]** section and set **sso.openid.profiles** to the same as the corresponding value in the **integrationserver.ini** file.
3. Enter the OpenID client id for each profile into the **sso.openid.profile.[profileName].client.id** setting.
4. Enter the OpenID client secret associated with the client id of each profile into the sso.openid.profile.[profileName].client.secret setting.

   Note OpenID client secrets in inserver.ini can be encrypted by running the **inserver -encrypt-config** command. We recommend you run this command each time you update a client secret. Do not manually update the encrypted client secret settings.

5. The **sso.openid.profile.[profileName].user.claim** setting maps OpenID authenticated end-users to Perceptive Content users. The user claim values should map one-to-one to Perceptive Content usernames. The user claim should be specified as a JSON attribute path. This path should identify a claim in the JSON object returned from the UserInfo endpoint of the OpenID Provider. For more information about the syntax of JSON attribute paths, see the JSON attribute paths topic in the Online Help.
6. If the OpenID Provider supports discovery, enter the discovery endpoint into the **sso.openid.profile.[profileName].discovery.endpoint** setting.

   Note When using OpenID Connect Discovery, OpenID configuration is cached in Perceptive Content. Perceptive Content refreshes OpenID configuration based on the discovery refresh interval setting on login profiles. To force Perceptive Content to invalidate cached OpenID configuration you can run the following command.

   ```
   intool --cmd openidconnect-invalidate-discovery-configuration
   ```
7. If the OpenID Provider doesn't support discovery, configure the following settings.
   1. Set **sso.openid.profile[profileName].auto.discovery** to **FALSE**.
   2. Enter the OpenID Provider's issuer URI into the **sso.openid.profile.[profileName].issuer.uri** setting.
   3. Enter the URL of the OpenID Provider's token endpoint into the **sso.openid.profile.[profileName].token.endpoint** setting.
   4. Enter the URL of the OpenID Provider's userinfo endpoint into the **sso.openid.profile.[profileName].userinfo.endpoint** setting.
   5. Prior to using OpenID Connect authentication, you must establish trust with OpenID Provider by importing one or more token signing keys. Use the following command with the OpenID Provider's signing certificate.

      ```
      intool --cmd import-cert --file <certificate file name> --type openid-connect --login-profile <profile name> --key-id <key ID value>
      ```

      Note You will need to determine what values the OpenID Provider uses for key IDs. Refer to documentation for the OpenID Provider to determine how they identify signing keys. Signing keys and key IDs are automatically imported when using OpenID Connect Discovery.
8. Ensure that the SSL/TLS root CA for the OpenID Provider is imported into the SSL trust store on the machine where Perceptive Content Server is running. Refer to the Perceptive Content Installation Guide for more information on configuring trust for SSL/TLS.
9. OpenID client secrets in inserver.ini can be encrypted by running the **inserver -encrypt-config** command. This command should be ran each time a client secret is updated. Do not manually update the encrypted client secret settings.
10. **Optional**. Configure any optional login profile settings. A detailed explanation of each login profile setting is provided in the inserver.ini OpenID Connect Login Profiles settings topic in the Content System Online Help.

Example

```
[OpenID Connect Login Profiles]
sso.openid.profiles=hylandexperience, defaultWithoutDiscovery

sso.openid.profile.hylandexperience.client.id=d48d37b4-c6ac-43a3-9864-60b9bad5c683
sso.openid.profile.hylandexperience.client.secret=ZJYCqe3GGRvdrudKyZS0XhGv_Z45DuKhCUk0gBR1vZk
```

```
sso.openid.profile.hylandexperience.user.claim=username
sso.openid.profile.hylandexperience.discovery.endpoint=https://sample.onbase.com/Ident
ityProvider/.well-known/openid-configuration

sso.openid.profile.defaultWithoutDiscovery.client.id=f282592c-e5e4-4b9f-a3fb-
022b6c149f04
sso.openid.profile.defaultWithoutDiscovery.client.secret=HJmUFH3GGRvdrudKyZS0XhGv_Z45D
uKhCUk0gJkdnU2
sso.openid.profile.defaultWithoutDiscovery.user.claim=username
sso.openid.profile.defaultWithoutDiscovery.issuer.uri=https://sample.onbase.com/Identi
tyProvider
sso.openid.profile.defaultWithoutDiscovery.token.endpoint=https://sample.onbase.com/Id
entityProvider/connect/token
sso.openid.profile.defaultWithoutDiscovery.userinfo.endpoint=https://sample.onbase.com
/IdentityProvider/connect/userinfo
```

Test the OpenID Connect

Complete the following steps to test the OpenID Connect.

1. To test the OpenID Connect flow, in a web browser go to **[hostname]/integrationserver/v1/sso/login/[profileName]?cookieApplicationId=testApplication**. If configured correctly, the system creates an Integration Server session.
2. To test the session, in a web browser go to **[hostname]/integrationserver/v2/connection?cookieApplicationId=testApplication**. If configured correctly, the system returns the userId and userName of the Perceptive Content user that was successfully authenticated. If an error occurs, refer to log files for Integration Server, Perceptive Content Server and the OpenID Provider.

SSL/TLS considerations

If you want to configure SSL/TLS to use with Integration Server, you must do so on your Tomcat web application server. We recommend using a CA signed certificate. For more information, see the Tomcat help documentation on the Apache website.

To establish trust between Integration Server and Perceptive Content Email Agent, when Integration Server is set up for SSL/TLS, you must import the Integration Server certificate into the Email Agent truststore. To establish trust with Email Agent, see the Email Agent Installation Guide.

Configure Integration Server for agent token authentication

Integration Server can be configured to support token-based authentication for agent connections. To import a token signing certificate and configure Integration Server, complete the following sections.

Import a token signing certificate

In order for Integration Server to validate authentication tokens, a token signing certificate is required. To import a token signing certificate, complete the following steps.

Note Perform these steps on your Integration Server machine.

1. Open a command window and navigate to the bin directory of your JRE installation.
2. Enter the following command and press ENTER.

```
keytool -import -alias [alias] -file [path to certificate] -keystore [path
to keystore] -storetype JKS
```

Setting	Options	Description
-alias	Any valid alias.	Specifies the identifier to associate with the certificate being imported.
-file	Any valid certificate path.	Specifies the path to the certificate.
-keystore	Any valid keystore path.	Specifies the keystore path. **Note** Only JKS keystores are supported.
-storetype	JKS	Specifies the keystore type. **Note** Only JKS is allowed.

Example

```
keytool -import -alias token-signing -file tokensigning.cer -keystore tokenauth.jks -
storetype JKS
```

Configure Integration Server for agent token authentication

To configure Integration Server support for agent token authentication, complete the following steps.

1. In the **WEB-INF** folder of your Integration Server deployment, open the **integrationserver.ini** file using a text editor.
2. Set **agent.token.authentication.method.enabled** to **TRUE**.
3. Enter the path to the token authentication keystore using the **agent.token.keystore.location** setting.
4. Optional. Enter the keystore password using the **agent.token.keystore.password** setting to enable integrity checking of the keystore.
5. Optional. If it is no longer required, disable support for legacy agent authentication by setting the **agent.legacy.authentication.method.enabled** to **FALSE**.
6. Save and close the file.
7. Restart Integration Server.

Configure agent token authentication for Forms Server

Optional. To configure Forms Server support for agent token authentication, complete the following steps.

1. On the Perceptive Content server, generate an authentication token for Forms Server by running the following command.

   ```
   intool --cmd create-authentication-token --lictype ImageNow Forms Server --file
   FormsServerToken.txt
   ```

2. On Integration Server, navigate to the **[path]/${TOMCAT_HOME}/webapps/<integration server subdirectory>/WEB-INF** directory.
3. Open the **integrationserver.ini** file in a text editor.
4. In the **[SSO]** section, set the **forms.server.authentication.token** setting to the contents of the **FormsServerToken.txt**.

   Example

   ```
   forms.server.authentication.token=eyJhbGciOiJFUzI1NiIsImtp...
   ```

5. Save and close the file.
6. Restart Integration Server.

Appendix A: integrationserver.ini file options

The following table provides definitions and sample data for the settings in the integrationserver.ini configuration file. This table displays the INI settings under group headings in brackets, for example, [General], in the order the groups appear in the INI file. Each setting offers two or more options, which are defined in the table below along with a description of each setting and its options. Use this table as a guide when customizing the file.

Group	Setting	Options	Description
General	server.ip.address	Any valid and explicit IP address	The IP address of the Perceptive Content server to use. For example, server.ip.address=localhost
	server.port	Any valid port	Specifies the port number of the Perceptive Content Server. For example, server.port=6000 The default is 6000.
	session.cookie.samesite.policy	UNSET NONE LAX STRICT	Specifies the SameSite policy set on session cookies. The default is UNSET.
	session.timeout	Any positive number	Specifies the connection timeout in minutes. For example, session.timeout=60 The default is 60.
	connection.timeout.logon	Any positive number	Specifies how many seconds to wait for successful logon before terminating the connection. The default is 30.
	connection.timeout.search	Any positive number	Specifies how many seconds to wait for search calls before terminating the connection. The default is 30.
	connection.timeout.erm	Any positive number	Specifies how many seconds to wait for ERM before terminating the connection. The default is 30.
	connection.timeout.default	Any positive number	Specifies how many seconds to wait for server calls before terminating the connection. The default is 30.

Group	Setting	Options	Description
	connection.use.generic.login.error.messages	TRUE FALSE	Specifies whether to return generic error messages when login errors occur. When disabled, allows more detailed error messages to be returned. Detailed error messages are always logged regardless of this setting. TRUE = Enable generic login error messages. FALSE = Disable generic login error messages and allow more detailed error messages. We recommend you enable this setting in production environments. The default is TRUE.
	ratelimiter.tracking.period.max.requests	Any positive number	Specifies the number of times each rate limited endpoint can be called within a thirty minute interval. If this limit is exceeded than all clients will receive a 429 status code until the current time interval has expired. The default is 560.
	stream.cutoff.size	Any positive number	Specifies the maximum size of a file transfer that will be stored within memory. File transfers exceeding this value are written to a temporary file instead. The default is 1048576 bytes.
	stream.transfer.chunk.size	Any positive number	Specifies the chunked data size to use when buffering an uploaded file during an Integration Server call. This setting should be smaller than the stream.cutoff.size. The default is 10240 bytes.

Group	Setting	Options	Description
Logging	log.level	0 through 7	This setting specifies the verboseness of the logging that Integration Server uses to log errors for troubleshooting. The higher the number, the more verbose the logging. Typically, set a minimal logging level unless debugging an issue. If logging is increased, make sure the logging level is reset to a lower level after the debugging is finished. Failure to do so can greatly affect performance and disk space. 0 = Logging is turned off (do not use) 1 = FATAL 2 = ERROR 3 = WARN 4 = INFO 5 = DEBUG 6 = TRACE 7 = ALL A logging level logs all information for that level and the lower levels. For example, if logging is set to 4, logging is turned on for levels 4, 3, 2, and 1. The default is 2.

Group	Setting	Options	Description
	log.level.thirdparties	0 through 7	Specifies the level Integration Server uses to logs messages from third-party libraries, such as the Spring Framework. The higher the number, the more verbose the logging. Typically, set a minimal logging level unless debugging an issue. If logging is increased, make sure the logging level is reset to a lower level after the debugging is finished. Failure to do so can greatly affect performance and disk space. 0 = Logging is turned off (do not use) 1 = FATAL 2 = ERROR 3 = WARN 4 = INFO 5 = DEBUG 6 = TRACE 7 = ALL A logging level logs all information for that level and the lower levels. For example, if logging is set to 4, logging is turned on for levels 4, 3, 2, and 1. The default is 2.
	log.timing.enabled	TRUE FALSE	Specifies if timing logging should be enabled. Timing logging outputs a small message when a request or response is transmitted. FALSE = Timing logging is disabled TRUE = Timing logging is enabled The default is FALSE.
	log.interceptor.enabled	TRUE FALSE	Specifies if interceptor logging should be enabled. Interceptor logging outputs the full request and response messages transmitted, which aids in debugging. FALSE = Interceptor logging is disabled TRUE = Interceptor logging is enabled The default is FALSE.

Group	Setting	Options	Description
	log.line.pattern	Any conversion pattern (including literal text) using valid specifiers and modifiers.	This setting formats the output of a logging event as a string of literal text. Each conversion specifier starts with a percent sign (%) and is followed by optional format modifiers and a conversion pattern name. The conversion pattern name specifies the type of data, such as logger, level, date, and so on. The format modifiers control such things as field width, padding, and justification. **Note** This pattern adheres to the Log4j PatternLayout standard.
	log.directory	Any valid path	Specifies the path to the directory that contains the log files. It is assumed that the Integration Server is installed and running on the same machine as Perceptive Content Server. The default is C:/inserver/logs
	log.rolling.interval	MONTH, DAY, or HOUR	This setting specifies the amount of time data is stored in a log file before the data is archived to another file. This keeps log files from becoming too large. The default is DAY.
	log.max.history	Any positive number.	This setting is a threshold for the maximum number of log files that are retained. One file is created each day, hour, or month depending on the log.rolling.interval setting. Note This setting excludes the log file that is currently being used to record events. For example, if log.max.history is set to 3 and the log.rolling.interval is set to DAY, then today's log file plus three additional log files exist for four log files. The default is 28. Giving this setting a value of 0 indicates that there should be no maximum number of log files retained. In other words, log files will not be automatically deleted.
Capture Settings	capture.work.directory	Any valid directory	The temporary directory to use for document captures and transfers. For example, capture.work.directory=C:/inserver/temp
Forms	forms.transform.output.method	html xml	Specifies the output method to use for forms transform. html = forms transform uses the html output method. xml = forms transform uses the xml output method. The default is xml.

Group	Setting	Options	Description
	forms.fallback.transform.output.method	html xml	Specifies the output method to use if forms transform fails. The fallback only occurs if this value differs from forms.transform.output.method. html = forms transform uses the html output method. xml = forms transform uses the xml output method. The default is xml.
Render Settings	render.fileimageservice.url	URL	This setting specifies the base URL for the File Conversion Service. For example, render.fileimageservice.url=http://localhost:1337 You can request the rendering of raster content from Perceptive Content through the GET: /document/{id}/page/{pageID}/rendition Integration Server call. The call retrieves a PNG rendition of a document page. **Note** You must install the File Conversion Service prior to using the Integration Server call. For installation instructions, see the Integration Server 7.2 > File Conversion Service Help.

Group	Setting	Options	Description

	render.fileimagesservice.maxrequests	Any positive number	This setting limits the number of simultaneous requests that can be made to the File Conversion Service from each instance of Integration Server. If this limit is exceeded then the operation will be blocked until the current operations are completed. You can set this value greater than the maximum number of concurrent requests that your File Conversion Service cluster can handle if you want Integration Server to immediately return when the pooled resources are exhausted.
			
			You must ensure that the maximum number of File Conversion Service requests made by Integration Server does not exceed the maximum concurrent conversions that File Conversion Service can handle. If the request exceeds that number, File Conversion Service fails the conversion with 500 server status code and reports the page as not available. You can query File Conversion Service for the MaxConcurrentConversions by sending a GET called to the File Conversion Service server.
			
			Example:
			
			`http://FCSserverAddress:1337/api/2018-01-06/status`
			`{"ActiveJobs":0,"MaxConcurrentConversions":8,"LoadRatio":0}`
			
			The default is 8.
			
			Note The value that File Conversion Service uses can be set in the File Conversion Service configuration. For more information, review the File Conversion Service Installation Guide.

	render.fileimageservice.connectionpool.enabled	TRUE FALSE	This setting enables or disables connection pooling for File Conversion Service requests. When enabled, connections to File Conversion Service are pooled and can be reused. These connections are cleaned up after the render.fileimageservice.connectionpool.connection.cleanup.after.idle.seconds interval. When disabled, a new connection to File Conversion Service is made for each request and all render.fileimageservice.connectionpool.* settings are ignored. The default is TRUE.
	render.fileimageservice.connectionpoool.connection.cleanup.after.idle.seconds	Any positive number	This setting controls the cleanup interval, in seconds, for expired and idle File Conversion Service pooled connections. The default is 3.

Group	Setting	Options	Description
	render.fileimageservice.connectionpool.connection.cleanup.after.idle.seconds	Any positive number	This setting controls the period of time, in seconds, that idle connections can remain in the File Conversion Service connection pool. The default is 25.
SSO	sso.enabled	TRUE FALSE	This setting specifies whether Single Sign On is enabled. The default is FALSE.
	sso.keystore.location	Any valid file path location	This setting specifies the file path location of the key store to be used for Single Sign On authentication. A forward slash should be used as the file separator.
	sso.keystore.password	At least 6 characters	Optional. This setting specifies the password to verify the integrity of the key store.
	sso.privatekey.alias	Any valid string	This setting specifies the alias for the key pair to be used for Single Sign On authentication.
	sso.privatekey.password	At least 6 characters	This setting specifies the password for the private key to be used for Single Sign On authentication.
	sso.authenticator.class	Any valid fully qualified class	This setting specifies the fully qualified class used to determine authorization from an external provider. This class must be in the application class path. Optional built-in classes provided for this purpose are: **com.imagenow.authentication.translator.HTTPHeaderTranslator com.imagenow.authentication.translator.x509ldap.X509LDAPAuthenticationTranslator**
	sso.httpheader.name	Any valid HTTP header field name	This setting specifies the name of the HTTP request header to be used by the built-in class HTTPHeaderTranslator plugin to map authentication information from an external provider.
	agent.legacy.authentication.method.enabled	TRUE FALSE	This setting specifies whether the legacy authentication method for agent connections is enabled. Default = TRUE
	agent.token.authentication.enabled	TRUE FALSE	This setting specifies whether token-based authentication for agent connections is enabled. Default = FALSE
	agent.token.keystore.location	Any valid path	This setting specifies the file path location of the key store to be used for token-based authentication for agent connections. A forward slash should be used as the file separator.

Group	Setting	Options	Description
	agent.token.keystore.password	Any valid password	Optional. This setting specifies the password to verify the integrity of the key store.
	forms.server.authentication.token	Any valid token	Optional. This setting specifies an authentication token for Forms Server to use when making connections.
OpenID Connect Login Profiles	sso.openid.login.enabled	TRUE FALSE	Specifies whether OpenID Connect authentication is enabled. FALSE = OpenID Connect authentication is disabled. TRUE = OpenID Connect authentication is enabled. The default is FALSE.
	sso.openid.profiles	Any alphanumeric characters	A comma separated list of OpenID Connect authentication profile names. Profile names must only contain alphanumeric characters.
	sso.openid.profile.<profileName>.authorization.endpoint	URL	The OpenID Provider authorization endpoint URL. The authorization endpoint performs authentication of the end user.
	sso.openid.profile.<profileName>.client.id	Any valid client identifier	A valid client identifier that is registered with the OpenID Provider.
	sso.openid.profile.<profileName>.grant.type	authorization_code	Optional. The authorization grant type to use for OpenID Connect authentication requests. authorization_code = Authorization Code grant type. The default is authorization_code.
	sso.openid.profile.<profileName>.use.pkce	TRUE FALSE	Optional. Specifies whether OpenID Connect authentication requests use a Proof Key for Code Exchange (PKCE). FALSE = Authentication requests do not use a Proof Key for Code Exchange (PKCE). TRUE = Authentication requests use a Proof Key for Code Exchange (PKCE). The default is TRUE.
	sso.openid.profile.<profileName>.scope	openid	Optional. A list of space separated, case sensitive strings that specify the scope of OpenId Connect authentication requests. openid = Required. Specifies an OpenID Connect authentication request. The default is openid.

Group	Setting	Options	Description
	sso.openid.profile.<profileName>.max.age	Any positive number or -1	Optional. Specifies the allowable elapsed time in seconds since the last time the end user was actively authenticated by the OpenID Provider. A value of -1 specifies that a max age should not be enforced. The default is -1.
	sso.openid.profile.<profileName>.acr.values	Any valid ACR values	Optional. A list of space separated Authentication Context Class Reference values that are added to authentication requests.
	sso.openid.profile.<profileName>.response.mode	query form_post	Optional. Specifies the mechanism for returning responses from the OpenID Provider. query = Response parameters are encoded in the query string. form_post = Response parameters are encoded as HTML form values and utilize the HTTP POST method. The default is form_post.
	sso.openid.profile.<profileName>.post.login.redirect.uri	URL	Optional. The URL to redirect end users to following a successful OpenID Connect login. The following values are added onto the redirect URL: state – Optional. Application state, provided it is included in the login request. By default, the system returns values as query parameters. Alternatively, you can use path variables that the system replaces with a value at run time. Path variables are denoted using the value name surrounded by curly braces. **Examples** Redirect URL using default semantics, values are added onto a query string. https://www.hostname.com/app/callback/ Redirect URL using a path variable, the {state} path variable are replaced at run time. https://www.hostname.com/app/callback/{state} If the post login redirect URL is not set, the following HTTP headers are returned in the callback response: X-IntegrationServer-SSO-App-State - Optional. Application state, provided it is included in the login request.

Group	Setting	Options	Description
	sso.openid.profile.<profileName>.error.redirect.uri	URL	Optional. The URL to redirect to following an unsuccessful OpenID Connect login. The following values are added onto the redirect URL: error - An error code for the unsuccessful login. state - Optional. Application state, provided it is included in the login request. By default, values will be returned as query parameters. Alternatively, path variables may be used which will be replaced with a value at run time. Path variables are denoted using the value name surrounded by curly braces. **Examples** Redirect URL using default semantics, values will be added onto a query string. https://www.hostname.com/app/callback/ Redirect URL using path variables, the {state} and {error} path variables will be replaced at run time. https://www.hostname.com/app/callback/{state}?error={error} If an error occurs and the error redirect URL is not set, the following HTTP headers are returned in the callback response: X-IntegrationServer-Error-Code - An error code for the unsuccessful login. X-IntegrationServer-Error-Message - An error message. X-IntegrationServer-SSO-App-State - Optional. Application state, provided it is included in the login request.
	sso.openid.profile.<profileName>.state.cookie.domain	Any valid domain	Optional. Specifies the domain to use when setting state cookies. The default is not to specify the domain when setting state cookies.
	sso.openid.profile.<profileName>.state.cookie.path	Any valid URI path	Optional. Specifies the path to use when setting state cookies. The default is the Integration Server SSO resource path.
	sso.openid.profile.<profileName>.state.cookie.expiration	Any positive number	Optional. Specifies the maximum lifetime in seconds of a state cookie. Ensure this value is long enough for end users to complete a valid login. The default is 300.

Group	Setting	Options	Description
	sso.openid.profile.<profileName>.same.site.cookies.enabled	TRUE FALSE	Optional. Specifies whether state cookies have the SameSite directive set to Strict. FALSE = State cookies are not explicitly set with the SameSite directive. TRUE = State cookies have the SameSite directive set to Strict. The default is TRUE. **Note** If the machine running Integration Server is different than the machine running the OpenID Provider, set sso.openid.profile.<profileName>.same.site.cookies.enabled to FALSE.
	sso.openid.profile.<profileName>.strict.callback.uri.validation	TRUE FALSE	Optional. Specifies whether callback URIs must match the hostname and profile name associated with the originating login request. FALSE = Do not validate callback URIs. TRUE = Enable callback URI validation. The default is TRUE.
	sso.openid.profile.<profileName>.validate.hostname	TRUE FALSE	Optional. Specifies whether to disable validation of the hostname specified on the request. This should not be disabled in production environments. TRUE = Always validate hostname values. FALSE = Do not validate hostname values. The default is TRUE.
	sso.openid.profile.<profileName>.max.response.value.size	Any positive number	Optional. Specifies the maximum allowable size in bytes of each parameter or authentication state value. The default is 1024. **Note** If your profile name, the URL to Integration Server including **/integrationserver/v1/sso/**, or the authorization codes produced by the OpenID Provider are longer than 1024 characters, update the maximum response value size setting to the appropriate value.
	sso.openid.profile.<profileName>.allowed.app.types	web desktop	Optional. Comma separated list of one or more application types that are allowed to login through this profile. If no valid application type is specified, then the default value is used. web = Perceptive Content web clients desktop = Perceptive Content desktop clients The default is web.

Group	Setting	Options	Description
Bearer Token Login Profiles	sso.bearer.profiles	Any alphanumeric characters	A comma separated list of bearer token login profile names. Profile names must only contain alphanumeric characters.
	sso.bearer.profile.<profileName>.bearer.token.character.validation	TRUE FALSE	Optional. Specifies whether to validate that bearer tokens conform to the expected bearer credentials syntax. FALSE = Do not validate bearer token characters. TRUE = Enable bearer token character validation. The default is TRUE.
	sso.bearer.profile.<profileName>.max.bearer.token.size	Any non-negative number	Optional. Specifies the maximum allowable size of bearer tokens. A value of 0 specifies that a maximum bearer token size should not be enforced. The default is 1024 characters.
ShareBase	sharebase.api.url	URL	Specifies the location of the ShareBase API.
	sharebase.api.bearer.token	Any valid token	Specifies the bearer token that will be used when interacting with the ShareBase API.
	sharebase.api.library.name	Any valid name	Specifies the name of the library to which documents will be uploaded. The default is My Library.
	sharebase.api.max.requests	Any positive number	Specifies the maximum amount of concurrent requests that can be made to the ShareBase API. The default is 8.
	sharebase.api.upload.chunk.threshold	Any positive number	Specifies a file size in bytes. Page files of this size or larger will be uploaded in chunks. The default is 5242880 (5 MB).
	sharebase.api.upload.chunk.size	Any positive number	Specifies the size, in bytes, of the chunks that will be uploaded for page files larger than the sharebase.api.upload.chunk.threshold setting. The default is 524288 (512 KB). The maximum allowed is 2097152 (2 MB).

Group	Setting	Options	Description
	sharebase.api.connectionpool.enabled	TRUE FALSE	When enabled, connections to ShareBase API are pooled and can be reused. These connections are cleaned up after the `sharebase.api.connectionpool.connection.cleanup.after.idle.seconds` interval. When disabled, a new connection to ShareBase API is made for each request and all `sharebase.api.connectionpool.*` settings are ignored. The default is TRUE.
	sharebase.api.connectionpool.connection.cleanup.interval.seconds	Any positive number	This setting controls the cleanup interval, in seconds, for expired and idle ShareBase API pooled connections. The default is 3.
	sharebase.api.connectionpool.connection.cleanup.after.idle.seconds	Any positive number	This setting controls the period of time, in seconds, that idle connections can remain in the ShareBase API connection pool. The default is 25.

Appendix B: X.509 LDAP authentication translator

The following table defines the settings for the X.509 LDAP authentication translator plugin. The plugin uses attributes to map a client X.509 certificate to an LDAP user. To map successfully, you must use unique attributes, such as an email address or user ID. If multiple LDAP entries match the X.509 client certificate, then mapping is unsuccessful.

The sso.authenticator.class setting for this plugin is com.imagenow.authentication.translator.x509ldap.X509LDAPAuthenticationTranslator.

Note To use this plugin, you must configure your application server to use SSL/TLS with client certificate authentication.

Setting	Options	Description
sso.x509ldap.ldap.server	Any valid address.	The LDAP server address.
sso.x509ldap.ldap.port	Any valid port number.	Optional. The LDAP server port number. If SSL is enabled, the default is 636. If SSL is not enabled, the default is 389.
sso.x509ldap.max.connections	Any valid number.	Optional. The maximum number of connections that can be made to the LDAP server. The default is 10.
sso.x509ldap.ldap.ssl	TRUE FALSE	Optional. Specifies whether to enable SSL/TLS for connections to the LDAP server. The default is FALSE.
sso.x509ldap.truststore.location	Any valid path.	Path to the truststore to use to validate the LDAP server certificates.
sso.x509ldap.truststore.password	Any valid password.	Optional. The password used to verify the integrity of the truststore.
sso.x509ldap.validate.server.certificates	TRUE FALSE	Optional. Specifies whether LDAP server certificates should be validated when SSL is enabled. The default is TRUE
sso.x509ldap.bind.dn	Any valid name.	The distinguished name of an account to use for LDAP server authentication.
sso.x509ldap.bind.password	Any valid password.	The password to be used for LDAP server authentication.

Setting	Options	Description
sso.x509ldap.anonymous.bind	TRUE FALSE	Optional. Specifies whether to use an anonymous login for LDAP server authentication. The default is FALSE.
sso.x509ldap.base.dn	Any valid name.	The distinguished name of the directory tree to be searched.
sso.x509ldap.x509.mapping.attribute	Any valid name.	The name of the X.509 attribute to be used for mapping client certificates to LDAP users.
sso.x509ldap.ldap.mapping.attribute	Any valid name.	The name of the LDAP attribute to be used for mapping client certificates to LDAP users.
sso.x509ldap.ldap.user.attribute	Any valid name.	The name of the LDAP attribute to use as the user's principal name when a client certificate can be mapped to an LDAP user.
sso.x509ldap.log.level	CONFIG, INFO, WARNING, or SEVERE.	Optional. The log level specifying which message levels will be logged. The default is SEVERE.
sso.x509ldap.log.directory	Any valid directory.	Optional. The directory to where log files will be written. The default is LOGS.

Example configuration

The following is an example of an X.509 LDAP authentication translator plugin configuration.

```
sso.authenticator.class=com.imagenow.authentication.translator.x509ldap.X509LDAPAuthen
ticationTranslator
sso.x509ldap.ldap.server=server.example.org
sso.x509ldap.ldap.ssl=true
sso.x509ldap.truststore.location=truststore.jks
sso.x509ldap.bind.dn=cn=ldapuser,ou=accounts,dc=example,dc=org
sso.x509ldap.bind.password=userPassword
sso.x509ldap.base.dn=ou=people,dc=example,dc=org
sso.x509ldap.x509.mapping.attribute=emailAddress
sso.x509ldap.ldap.mapping.attribute=mail
sso.x509ldap.ldap.user.attribute=sAMAccountName
```