opentext™

# OpenText™ Output Transformation Server

## Cloud Deployment Guide

This document provides information about the deployment of docker images for OpenText Output Transformation Server.

VDTOTS240200-ACD-EN-1

**OpenText™ Output Transformation Server**
**Cloud Deployment Guide**
VDTOTS240200-ACD-EN-1
Rev.: 2024-June-05

**This documentation has been created for OpenText™ Output Transformation Server CE 24.2.**
It is also valid for subsequent software releases unless OpenText has made newer documentation available with the product, on an OpenText website, or by any other means.

**Open Text Corporation**

275 Frank Tompa Drive, Waterloo, Ontario, Canada, N2L 0A1

Tel: +1-519-888-7111
Toll Free Canada/USA: 1-800-499-6544 International: +800-4996-5440
Fax: +1-519-888-0677
Support: https://support.opentext.com
For more information, visit https://www.opentext.com

# Table of Contents

Chapter 1

# Overview

OpenText Output Transformation Server enables an organization to streamline their information supply chains by eliminating both the deficiencies and unnecessary links when sharing content between multiple sources. This process allows for the high speed integration, composition, translation, transformation, indexing, repurposing, archiving, retrieval, routing, printing, delivery, and presentment of structured and unstructured data throughout the enterprise and beyond. It makes possible the flexible sharing of previously inaccessible strategic information assets with customers, employees and also trading partners.

OpenText provides container images that can be used to install Output Transformation Server with Configuration Manager, Alternate Text Manager, and Document Accessibility in a generic Kubernetes cluster.

**Tip:** Kubernetes is a container orchestration platform for deploying and managing Docker containers. It provides many features, including automatic scaling, high availability, and fault tolerance, and simplifies the deployment of Output Transformation Server.

The Output Transformation Server container deployment includes the following:

- An Output Transformation Server (**OTS**) instance with access to Configuration Manager (**CM**), Alternate Text Manager (**ATM**), and Document Accessibility (**DA**)
- Postgres database servers with configuration settings for the server that houses the database for Configuration Manager (CM), Alternate Text Manager (ATM), and Document Accessibility (DA)
- An Ingress controller configured for Output Transformation Server

OpenText provides the container images and Helm Charts for Output Transformation Server and OpenText Directory Services. You can download them from OpenText My Support.

**Important**

Prior to using and configuring the software, every user on the system must be duly licensed for Output Transformation Server, Configuration Manager, Alternate Text Manager, and Document Accessibility. Using this software indicates acknowledgement of this requirement and your certification that your organization is compliant with this requirement per the terms of the OpenText End User License Agreement (EULA) signed between the parties or if no such agreement is signed between the parties then per the terms of the OpenText End User License Agreement found at www.opentext.com/

agreements for the applicable region. Each user for which a product feature will be enabled is duly licensed for this functionality.

## 1.1 About this guide

This guide explains how to deploy your OpenText Output Transformation Server product using container images provided by OpenText. The software tools that you need to do this job are listed in "Requirements" on page 9. The container images and Helm charts that you need are listed in "Obtain Container Images and Helm Charts" on page 15.

You will deploy your container images in a Kubernetes cluster. Basic instructions for the creation of a Kubernetes cluster on your organization's cloud platform are provided in "Create a Kubernetes Cluster" on page 19. For more detailed information, refer to your cloud provider's documentation.

If you deploy your product using the default parameters, Output Transformation Server will use Apache Tomcat as its web server and PostgreSQL as its database. Configuration Manager, Alternate Text Manager, and Document Accessibility will use a PostgreSQL database. All of these products will run in a Kubernetes cluster. The default deployment of Output Transformation Server is described in "Deploy Output Transformation Server" on page 21

If the default deployment does not meet your organization's needs, you can modify the deployment of Output Transformation Server by using additional Helm command-line arguments. This is explained in "Customize the Deployment" on page 33.

Once your deployment is up and running, you can refer to "Verify the Deployment" on page 43 for basic ways to test that each product in the deployment is up and running. For more detailed information, you should refer to the Installation and Configuration guide for your product.

One of the principal benefits of containerization is ease of scaling and upgrading. These are described in "Scale the Deployment" on page 47.

Chapter 2

# Requirements

To perform the steps outlined in this document, you will need the items listed below. Refer to the Output Transformation Release Notes for information on the specific platforms and versions that are supported.

**Note:** The instructions in this document include examples of how to use the required third-party applications. To obtain additional in-depth information, consult the application's official documentation.

**Docker**
You will use Docker, running on a Windows or Linux computer, to tag container image files and push them to your cloud platform repository.

**Kubernetes**
Kubernetes is a container orchestration platform for deploying and managing Docker containers. You may use Kubernetes commands (`kubectl.exe`) during the deployment of the Output Transformation Server container to validate or tweak the deployment, but the main deployment should be performed using Helm.

**Helm**
Helm is used to deploy the container images. Make sure that you use at least Helm 3.6. Previous versions of Helm will not work with the containerized deployment of Output Transformation Server.

**OpenText Directory Services**
OpenText Directory Services (OTDS) is required for authentication of users. If your security protocol uses SSL certificates for encryption, you must be ready to provide the required certificates.

**An SSL certificate**
If your components communicate with your OTDS instance through a secure protocol, you must provide an SSL certificate from your organization's infrastructure. You will add these files to the contents of the Helm charts that you downloaded in "The ots-server Helm Chart" on page 15 and then use them to create a Kubernetes Secret object in a Kubernetes cluster mounted on your organization's cloud platform.

For a non-production installation of Output Transformation Server, you can use a self-generated certificate. Free self-generated certificates are available at several sites on the Internet, including at https://letsencrypt.org.

**PostgreSQL database**
Access to a PostgreSQL database is required for all components. To create users for the Configuration Manager, Alternate Text Manager, and Document Accessibility applications in the database, you must have the PostgreSQL

administrator user credentials. The database Administrator should also have the ability to create roles and databases.

A PostgreSQL database is also required for OTDS. In this database, you must have the `pg_trgm` extension installed or alternatively, temporarily provide the **otds** user with permissions to install the extension. Additionally, this database Administrator should have the ability to create roles and databases. For more information about installing the `pg_trgm` extension, see "Installing the pg_trgm module in PostgreSQL for OTDS" on page 12.

## 2.1 Pre-deployment requirements

In preparation for deployment of the cloud images, you must complete some pre-deployment steps. PostgreSQL users and databases for Output Transformation Server and OTDS must be created as well as establishing passwords for some administrator users.

### 2.1.1 Required passwords for ots-server image

When deploying images and running assorted commands during the configuration phase, passwords for various roles are required. To prepare, you must make sure the following users and their passwords are created before starting the deployment process so that they can be substituted into the Helm chart commands when required. (For more information about creating these passwords, see their respective topics in this document.) The following passwords must be created:

**Table 2-1: Required Database user passwords**

| Password owner | Variable | Notes |
|---|---|---|
| OTDS database user | *<otdsDbPassword>* | Required for all deployments |
| Configuration Manager database user | *<cmUserDbPassword>* | Required for all deployments |
| Alternate Text Manager database user | *<atmUserDbPassword>* | Only needed for Document Accessibility deployments |
| Document Accessibility database user | *<daUserDbPassword>* | Only needed for Document Accessibility deployments |

**Table 2-2: Required Application user passwords**

| Password owner | Variable | Notes |
|---|---|---|
| OTDS Administrator | *<otdsAdminPassword>* | For OTDS Administrator |

| | | |
|---|---|---|
| OTDS Crypt key | *<otdsCryptKey>* | The 16 character ASCII string that has been Base64 encoded from the OTDS Helm chart.<br><br>This key is only required when performing new OTDS installations. |
| OTS Administrator | *<otsAdminPassword>* | Administrator user for Output Transformation Server Manager and Configuration Manager; Required for all deployments |
| OTS JobRunner user | *<otsJobRunnerPassword>* | For Output Transformation Server deployments |
| OTS JobRunner token | *<otsJobRunnerToken>* | Only needed for Documentum integrations, otherwise leave blank |
| OTS Billing user | *<otsBillingPassword>* | For Output Transformation Server Manager |
| OTS Developer user | *<otsDevPassword>* | For Output Transformation Server Manager |
| Alternate Text Manager Administrator | *<atmAdminPassword>* | For Alternate Text Manager Administrator |
| Document Accessibility Administrator | *<daAdminPassword>* | For Document Accessibility Administrator |
| Document Accessibility Example Trainer | *<daTrainerPassword>* | For sample Document Accessibility Trainer user |
| Document Accessibility Example User | *<daUserPassword>* | For sample Document Accessibility regular user |
| JGroups Cluster token | *<jgroupsToken>* | Token used to join the JGroups cluster |

### 2.1.2 Setting up PostgreSQL for OTDS

Before you can get started with deploying the images, you must create an OTDS database user and set up the OTDS database.

To create the OTDS database user, execute a command similar to the following:

```
create user otds with createdb password '<otdsDbPassword>' valid until
'infinity';
grant otds to <dbAdminUser>;
```

When running this command, substitute *<dbAdminUser>* with the logged in user you are using to create the OTDS user.

To create the OTDS database:

```
create database otds with owner='otds';
grant connect on database otds to otds;
```

The pg_trgm module must be installed in the PostgreSQL database for OTDS. For more information about installing the module, see "Installing the pg_trgm module in PostgreSQL for OTDS" on page 12.

### 2.1.3 Setting up PostgreSQL for OTS

Database users must be created in the PostgreSQL database. The database users to create vary depending on the products you are deploying.

#### All deployments

For all deployments, you must create a user named `cmuser` and choose a password. To create the user and their password, substitute *<cmUserDbPassword>* with the password you want while executing a command similar to the following:

```
create user cmuser with createdb password '<cmUserDbPassword>' valid until
'infinity';
```

#### Document Accessibility deployments

When deploying Document Accessibility, you must create users named `atmuser` and `dauser` and choose passwords for them.

To create `atmuser` and their password, substitute *<atmUserDbPassword>* with the password you want while executing a command similar to the following:

```
create user atmuser with createdb password '<atmUserDbPassword>' valid until
'infinity';
```

To create `dauser` and their password, substitute *<daUserDbPassword>* with the password you want while executing a command similar to the following:

```
create user dauser with createdb password '<daUserDbPassword>' valid until
'infinity';
```

### 2.1.4 Installing the pg_trgm module in PostgreSQL for OTDS

In the PostgreSQL database used by OTDS, you must have the pg_trgm extension installed or alternatively, temporarily provide the `otds` user with **superuser** privileges to install the extension themselves.

To install the pg_trgm module in the PostgreSQL database instance used by OTDS, connect to the OTDS database and run the following command:

```
create extension if not exists pg_trgm
```

If you opted to grant the otds user with superuser privileges, make sure you remove these high-level privileges from their account once the extension has been installed.

Chapter 3

# Obtain Container Images and Helm Charts

To install your Output Transformation Server product, you require container images and Helm charts.

You can obtain the container images from the OpenText Container Registry and the Helm charts from My Support.

## 3.1 Output Transformation Server Container Images

The container images that you use to deploy your Output Transformation Server product are available in the OpenText Container Registry. The image names and tags for the pods deployed in a default deployment of Output Transformation Server can be found in the `values.yaml` file in the `ots-server` Helm chart. You have the choice of obtaining these images manually using Docker, or configuring Helm to obtain them automatically. (For more information on obtaining the images, see "OpenText Container Registry" on page 16.)

**Output Transformation Server products**

Output Transformation Server includes Configuration Manager, which is enabled by default. Alternate Text Manager, and Document Accessibility can be enabled by setting their enabled flags (`ATM.enabled` and `DA.enabled`, respectively) to true. The `values.yaml` file, located in the `ots-server` Helm chart, contains the name (`ots-server`) and tag for the compatible version of the Output Transformation Server product.

Configuration Manager, Alternate Text Manager, and Document Accessibility do not have separate image names and tags because they are part of the default deployment, but you can manually turn off these features in the `values.yaml` file.

## 3.2 The ots-server Helm Chart

The Helm chart that you require is available on My Support. It is stored in compressed ZIP format, with a name similar to `ots-server-##.#-helm-otk-cfcr.zip`. For example, the Helm Chart for Output Transformation Server is in a file named `ots-21.2-helm.zip`. This file contains the Helm charts that orchestrate the deployment of Output Transformation Server.

### Extract Helm Chart

Once you have obtained the Helm chart zip file, upload it to the computer that you will use to run the Helm commands. This could be your Windows computer running Docker, or it could be a cloud shell on your organization's cloud platform. Extract the file contents using your preferred extraction utility. Following extraction, there will be an `ots-server-##.#-helm-otk-cfcr` subdirectory in your current directory.

## 3.3 Select your Container Registry

You can use the OpenText Container Registry (`registry.opentext.com`) as the source of your container images, or you can push them to your organization's container registry and use that as the source.

### 3.3.1 OpenText Container Registry

To use the OpenText Container Registry (`registry.opentext.com`) as your `imagesRepository`, create a Docker-Registry secret and specify it as the `imagePullSecrets` in your `default-k8s.yaml` file. For more information on setting up the registry secret, see "Docker-Registry Secret for the OpenText Container Registry" on page 21.

### 3.3.2 Other Container Registry

To use a different container registry, obtain the images from the OpenText Container Registry, re-tag them, and then push them to your container registry.

**Notes**

- In "Set Deployment Parameters" on page 22, you will specify your Container Registry so that Helm can use the container images there to deploy your Output Transformation Server product. To enable this, you should structure your registry so that the images reside at a common location. You will specify this common location of your Output Transformation Server images as the value of `imagePath`.

  For example, if your Container Registry has the following folder:

  – `my_registry_name.my_cloud.com/my_registry/ots`

  You would specify `my_registry_name.my_cloud.com/my_registry/` as the value of `imagePath` in your `default-k8s.yaml` file.

#### 3.3.2.1 Configuring Helm charts to access your container registry

If the container images are being pulled from your own container registry instead of the default OpenText Container Registry location, you must update the `default-k8s.yaml` file with appropriate values for the `imagesRepository`, `imagePath`, and `imagePullSecrets` variables to specify the new location. The Helm chart uses these variables to pull the Output Transformation Server container images with the following format:

```
<imagesRepository>/<imagePath>/<OTSServer.image.name>:<OTSServer.image.tag>
```

Alternatively, the Helm chart values can be overridden on the command line with the following Helm set commands during deployment:

```
--set images.pullSecrets[0]=<Pull_Secret_Name>
--set images.repository=<My_Cloud_Repository>
--set images.path=<Image_Path (If required)>
--set OTSServer.image.name=<OTSServer_Image_Name>
--set OTSServer.image.tag=<OTSServer_Image_Tag>
```

#### 3.3.2.2 Obtain OpenText Container Images

Complete the following steps to obtain the OpenText container images that you require.

**To obtain OpenText Container Images:**

1. Log on to the OpenText Container Registry by executing the following command: `docker login registry.opentext.com`. Provide your My Support logon ID and password when you are prompted.
2. Download each image that you require by executing a `docker pull registry.opentext.com/<image_name>:<image_tag>` command. For example, to download the Output Transformation Server 21.4.00_1234 container image, execute `docker pull registry.opentext.com/ots-server:21.4.00_1234`

#### 3.3.2.3 Push Container Images

The following steps show how to prepare the Output Transformation Server container image and push it to your private container registry. Repeat these steps for the OTDS images.

**To push the container images to your private container registry:**

1. Verify that the image has loaded successfully by running a docker images command. The output should list the images that you obtained in "Output Transformation Server Container Images" on page 15. It should appear similar to the following:

```
REPOSITORY                          TAG          IMAGE ID      CREATED       SIZE
registry.opentext.com/ots-server  21.4.0_1234  f93e211dabc9  29 hours ago  925MB
```

2. Retag the image to suit your environment by running a command similar to the following:

   `docker tag <IMAGE_ID> <My_Cloud_Repository>/ots:21.4.0_1234`

   After you run the command, an additional row, similar to the following, should appear when you run a docker images command again.

```
REPOSITORY                 TAG          IMAGE ID      CREATED       SIZE
<My_Cloud_Repository>/ots  21.2.4_1234  d5a47b8d5dd3  29 hours ago  925MB
```

3. Push the image to your container registry by running a command similar to the following:

   `docker push <My_Cloud_Repository>/ots:21.4.0_1234`

   You should see output that is similar to the example below.

```
The push refers to repository [<My_Cloud_Repository>/ots]
0f5b0e3c99fb: Pushed
96f34c47d792: Pushed
32f0767c0ded: Pushed
25069c0651cb: Pushed
43e0062e571e: Pushed
492b994cb74b: Pushed
4ab452b98d33: Pushed
ded7b91e20ca: Pushed
latest: digest:
sha256:6087335ff1bb4f90e5dd6bd28844c96924016509449d64d949a6fa00df79a887 size: 4699
```

   Once the push completes, the image appears in your container registry.

Chapter 4

# Create a Kubernetes Cluster

A Kubernetes cluster consists of at least one cluster master and multiple worker machines called nodes. These master and node machines run the Kubernetes cluster orchestration system. The Kubernetes objects that represent your containerized applications all run within a cluster.

To deploy your Output Transformation Server product, your Kubernetes cluster should include:

- An Ingress Controller (such as Nginx) to expose an Output Transformation HTTP(s) endpoint outside of the cluster
- If using SSL, a valid SSL certificate to enable HTTPS endpoint encryption between your OTDS and OTS instances and the users who log on to them from outside the cluster, such as, over the public Internet.
- If using Document Accessibility, ensure a storage class is installed that supports ReadWriteMany operations for Persistent Volume Claims (i.e. NFS storage)

**Tip:** The instructions in this section create a Kubernetes cluster in the default namespace. To set `kubectl` to operate in a specific namespace, use the following command:

```
kubectl config set-context --current --namespace=<namespace>
```

Kubernetes namespaces are a way to divide cluster resources between multiple users. For more information on Kubernetes namespaces, see the Kubernetes official documentation.

Chapter 5

# Deploy Output Transformation Server

To deploy your Output Transformation Server product on your organization's cloud platform, you will set user credentials, basic deployment parameters for your cloud platform, and adjust default server resources in the YAML file before deploying your product using a Helm command.

This section describes how to deploy Output Transformation Server and related products using minimal settings; the ones that are required or recommended for initial deployment. Refer to "Customize the Deployment" on page 33 for information on additional optional settings that you can use to modify Output Transformation Server after it has been deployed.

## 5.1 Docker-Registry Secret for the OpenText Container Registry

In your `values.yaml` file, you specify the registry that has the Output Transformation Server container images. When specifying the OpenText Container Registry, you need to create a Docker-Registry secret.

The OpenText Container Registry requires you to log on before you can pull any container images. To enable the container runtime to obtain the images automatically from the OpenText Container Registry, create a Docker-Registry secret that permits the container runtime to log on to the OpenText Container Registry using your My Support credentials, and specify the name of the Docker-Registry secret in your `values.yaml` file.

**OpenText Container Registry**

To use the OpenText Container Registry as your image source, execute the following command to create a Docker-Registry Secret, and then specify *<secret_name>* as the value of `imagePullSecrets` in your `values.yaml` file.

**Other Container Registry**

To use a registry other than the OpenText Container Registry, follow the instructions in "Obtain Container Images and Helm Charts" on page 15 and "Push Container Images" on page 17. (You do not need to create a Docker-Registry Secret.)

## 5.2 Set Deployment Parameters

Set the default deployment parameters for a generic deployment by editing the `ots-server/values.yaml` file.

### 5.2.1 Set Deployment Parameters for a Generic Deployment

Before you deploy the Output Transformation Server containers, edit the `platform/default-k8s.yaml` file so that it contains values appropriate for your deployment. The top of the file has the following appearance:

```
##########################################################################
# Default generic k8s platform template can be used as basis for other platforms.
##########################################################################

# CHANGE HERE ↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓
global:
  platform:             &platform              default-k8s
  # Default is to pull from registry.opentext.com by default.  Requires valid Open Text
account to login (specify image pull secret)
  # Modify values below to match your system if you have loaded image to a private
registry
  # Full path for image is <imagesRepository>/<imagePath>/
<OTSServer.image.name>:<OTSServer.image.tag>
  # -- The Docker registry to pull OTS Server image from
  imagesRepository:     &images_repository     registry.opentext.com
  # -- The path in the docker registry to use.  Keep Blank if using registry.opentext.com
  imagePath:            &images_path
  # -- The imagePullSecret to use if pulling from a secure Docker registry.
  imagePullSecrets:     &image_pull_secrets    []
# To define pull secret, comment above line and uncomment following two lines, replacing
'imagePullSecretName' with the appropriate secret name
# imagePullSecrets:     &image_pull_secrets
#   - name: imagePullSecretName
  # -- The image pull policy
  imagePullPolicy:      &image_pull_policy     IfNotPresent
  # -- Specify a storage class the supports RWM if using Document Accessibility.
Otherwise this is not used.
  storageClassNameRWM:  &storage_class_nfs     nfs
  # -- Is ingress enabled for Output Transformation Server
  ingressEnabled:       &ingress_enabled       true
  # -- Ingress class to use (if enabled) for Output Transformation server
  ingressClass:         &ingress_class
  # -- The public hostname for Output Transformation Server
  OTSServerPublicHostname:                     localhost
```

Make the following changes below the # **CHANGE HERE** ↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓ heading in the `default-k8s.yaml` file:

**`imagesRepository`**

If you are not pulling the Output Transformation Server images from the OpenText registry, replace `registry.opentext.com` with the URL of the Docker registry that contains your Output Transformation Server images. (See "Select your Container Registry" on page 16.) If you leave the OpenText Container Registry as your `imageRepository`, you must provide a value for `imagePullSecret`.

**imagePullSecrets**
To use the OpenText Container Registry as your imagesRepository, add the name of your Docker-Registry Secret here. For more information, see "Docker-Registry Secret for the OpenText Container Registry" on page 21.

**ingressClass**
Replace with the Ingress class to use for Output Transformation Server.

**OTSServerPublicHostname**
Indicates the public host name for Output Transformation Server.

## 5.2.2 Set Deployment Parameters for OTSServer

The base OTSServer parameters in the `ots-server/values.yaml` file include user credentials, database information, and JGroups clustering parameters. The `values.yaml` file is located in the `<localFolder>/ots-server-22.4-helm-otcloud/ots-server/` folder.

### User credentials

The default user credentials for various users must be specified. This section of the file has the following appearance:

```
OTSServer:
     adminPass: "<otsAdminPassword>"
     jobRunnerPass: "<otsJobRunnerPassword>"
     jobRunnerToken: "<otsJobRunnerToken>"
     billingPass: "<otsBillingPassword>"
     devPass: "<otsDevPassword>"
```

Make the following changes in the `values.yaml` file.

**adminPass**
Sets the password of the Output Transformation Server administrator user, *<otsAdminPassword>*. This is required for all deployments.

**jobRunnerPass**
Sets the password of the JobRunner user, *<otsJobRunnerPassword>*.

**jobRunnerToken**
Sets the password for the JobRunner user, *<otsJobRunnerToken>*.

**billingPass**
Sets the password of the Billing user, *<otsBillingPassword>*.

**devPass**
Sets the password of the Developer user, *<otsDevPassword>*.

### Database information

The required database administrator user credentials should have been created during the pre-deployment stage. The database host name and port number must be defined. This section of the file has the following appearance:

```
OTSServer:
     dbAdminUser: ""
     dbAdminPass: ""
     dbHost: "<dbHostname>"
     dbPort: "<dbPort>"
```

Make the following changes in the `values.yaml` file.

**dbAdminUser**
The user name for the database administrator. Because these user credentials should have been created during the pre-deployment stage, the value must be empty so that the system does not attempt to recreate the user account.

**dbAdminPass**
Sets the password of the database administrator user. Because these user credentials should have been created during the pre-deployment stage, the value must be empty so that the system does not attempt to recreate the user account.

**dbHost**
The public hostname for the OTSServer database.

**dbPort**
The port number for the OTSServer database.

## JGroups clustering

JGroups clustering is enabled by default. This section of the file has the following appearance:

```
OTSServer:
     clusterManagerEnabled: true
     clusterManagerJoinToken: “<jgroupsToken>”
     clusterManagerTrace: false
```

Make the following changes in the `values.yaml` file.

**clusterManagerEnabled**
Defines whether JGroups clustering is used. By default, this is enabled.

**clusterManagerJoinToken**
The JGroups token to use for the cluster. Do not set this to the default value, the token must be a generated or user-specific alphanumeric string value.

**clusterManagerTrace**
Enables verbose logging for JGroups. Log messages are printed to the Standard Out (console). By default, logging is turned off.

### 5.2.3 Set Deployment Parameters for CM

The base CM parameters in the `ots-server/values.yaml` file include parameters to initialize the Configuration Manager as well as security settings. This section of the file has the following appearance:

```
OTSServer:
     enabled: true
     dbPass: “<cmUserDbPassword>”
     dbOpts:
        # dbOpts: '?ssl=true&sslmode=verify-
full&sslfactory=org.postgresql.ssl.DefaultJavaSSLFactory'
```

Make the following changes in the `values.yaml` file.

**`enabled`**
Defines whether the Configuration Manager option for Output Transformation Server is enabled using a **True** or **False** value. The value must always be true because this feature is required.

**`dbPass`**
Sets the password of the Configuration Manager database user. The *<cmUserDbPassword>* should have been created during the pre-deployment stage.

This parameter should only be used for setting the password when creating new users. If a user already exists, this parameter will not update their existing password.

**`dbOpts`**
If you are connecting to the database via SSL, you must configure the SSL settings. If your database connection is not via SSL, you must leave the value blank. A sample comment for dbOpts shows a potential value that can be set if SSL is enabled for PostgreSQL.

### 5.2.4 Set Deployment Parameters for ATM

The base ATM parameters in the `ots-server/values.yaml` file include parameters to initialize the Alternate Text Manager option, user credentials, and security settings. This section of the file has the following appearance:

```
ATM:
     enabled:
     dbPass: “<atmUserDbPassword>”
     adminPass: “<atmAdminPassword>”
     dbOpts:
     # dbOpts: '?ssl=true&sslmode=verify-
full&sslfactory=org.postgresql.ssl.DefaultJavaSSLFactory'
```

Make the following changes in the `values.yaml` file.

**`enabled`**
Defines whether the Alternate Text Manager option is enabled using a **True** or **False** value.

**dbPass**
Sets the password of the Alternate Text Manager user. The *<atmUserDbPassword>* should have been created during the pre-deployment stage.

This parameter should only be used for setting the password when creating new users. If a user already exists, this parameter will not update their existing password.

**adminPass**
Sets the password of the Alternate Text Manager administrator user. Make note of the *<atmAdminPassword>* as it may be required for subsequent authentications.

This parameter should only be used for setting the password when creating new users. If a user already exists, this parameter will not update their existing password.

**dbOpts**
If you are connecting to the database via SSL, you must configure the SSL settings. If your database connection is not via SSL, you must leave the value blank. A sample comment for dbOpts shows a potential value that can be set if SSL is enabled for PostgreSQL.

## 5.2.5 Set Deployment Parameters for DA

The base DA parameters in the `ots-server/values.yaml` file include parameters to initialize the Document Accessibility option, user credentials, and security settings. This section of the file has the following appearance:

```
DA:
     enabled:
     dbOpts:
     # dbOpts: '?ssl=true&sslmode=verify-
full&sslfactory=org.postgresql.ssl.DefaultJavaSSLFactory'
     dbPass:
     adminPass:
```

Make the following changes in the `values.yaml` file.

**enabled**
Defines whether the Document Accessibility option is enabled using a **True** or **False** value.

**dbOpts**
If you are connecting to the database via SSL, you must configure the SSL settings. If your database connection is not via SSL, you must leave the value blank. A sample comment for dbOpts shows a potential value that can be set if SSL is enabled for PostgreSQL.

**dbPass**
Sets the password of the Document Accessibility user. The *<daUserDbPassword>* should have been created during the pre-deployment stage.

This parameter should only be used for setting the password when creating new users. If a user already exists, this parameter will not update their existing password.

**adminPass**
Sets the password of the Document Accessibility administrator user. Make note of the <*daAdminPassword*> as it may be required for subsequent authentications.

This parameter should only be used for setting the password when creating new users. If a user already exists, this parameter will not update their existing password.

### 5.2.6 Set Deployment Parameters for OTDS

The base OTDS parameters in the `ots-server/values.yaml` file include parameters to define the connection to OTDS. This section of the file has the following appearance:

```
OTDS:
     enabled: true
     host: “<otdsPublicHostname>”
     port: 443
     protocol: https
     otadminUser: “otadmin@otds.admin”
     otadminPass: “<otdsAdminPassword>”
```

Make the following changes in the `values.yaml` file.

**enabled**
Defines whether the OTDS connection is enabled using a **True** or **False** value. The value must always be true because this feature is required.

**Host**
The public hostname for the OTDS instance.

**Port**
The port number for the OTDS instance.

**Protocol**
The network protocol used for the OTDS instance. The value must always be set to `https`.

**otadminUser**
The user name for the OTDS administrator. For standard OTDS installations, `otadmin@otds.admin` is the default administrator user name used, however, you must modify this value if you are using a different administrator user. The **otadminUser** should match the value set when OTDS was deployed.

**otadminPass**
The password of the OTDS administrator user, <*otdsAdminPassword*>. The **otadminPass** should match the value set when OTDS was deployed.

## 5.3 OTDS Helm chart

Before you can deploy OTDS, you must set up a PostgreSQL user and database for OTDS as well as the required users for Output Transformation Server and any additional products being deployed. For more information about creating the OTDS PostgreSQL user and database, see "Setting up PostgreSQL for OTDS" on page 11. For more information about creating the necessary database users for each product, see "Setting up PostgreSQL for OTS" on page 12.

Furthermore, OTDS requires a 16 character string value to be encrypted as Base64 for the cryptKey parameter. You must provide this value when you run the command to deploy the OTDS Helm chart.

### 5.3.1 Generating the OTDS Encryption Key

The OTDS encryption key is a shared key used by OTDS instances to secure content in the OTDS database.

You must manually generate a 16-character alphanumeric string (for example, abcdefg123456789) and then encrypt it in Base 64 using Powershell or Linux. This value is then used as the *<otdsCryptKey>* value when running the OTDS Helm chart deployment command.

#### Encrypting with PowerShell

To encrypt the 16-character string value, you must execute a command similar to the following:

```
$Text = '<cryptKeyString>' ; `
$Bytes = [System.Text.Encoding]::UTF8.GetBytes($Text) ;`
$EncodedText =[Convert]::ToBase64String($Bytes) ;`
$EncodedText
```

When running the command, substitute *<cryptKeyString>* where *<cryptKeyString>* is your generated string.

#### Encrypting with Linux

To encrypt the 16-character string value, you must execute a command similar to the following:

```
echo -n <cryptKeyString> | base64
```

When running the command, substitute *<cryptKeyString>* where *<cryptKeyString>* is your generated string.

### 5.3.2 Deploy OTDS Helm chart

Go to the OTDS folder in your extracted Helm chart and run the following command:

```
helm install otds-deployment ./otds/ `
--set otdsws.image.source='registry.opentext.com ' `
--set otdsws.otdsdb.url='jdbc:postgresql://<otdsDbHostname>:5432/otds' `
--set otdsws.otdsdb.username=<otdsDbUser> `
--set otdsws.otdsdb.password=<otdsDbPassword> `
--set otdsws.publicHostname=<otdsPublicHostname> `
--set otdsws.otadminPassword=<otdsAdminPassword>  `
--set otdsws.cryptKey='<otdsCryptKey>' `
--set otdsws.replicas=1 `
--set ingress.enabled=true `
--set ingress.class=<ingressClass>
```

By default, the OTDS Helm chart sets the `otdsws.image.source` value to `registry.opentext.com`, which you must replace if you are pulling the `otds-server` image from a local registry instead.

For the `otdsws.otdsdb.url` parameter, you must use the JDBC URL for connecting to the OTDS database. In the example value shown, a connection to a PostgreSQL database named `otds` at `5432`. You must replace this value as required for your environment.

Before running the command, you must also replace the following values:

- <*otdsDbHostname*> with your OTDS database host name
- <*otdsDbUser*> with your OTDS database user name
- <*otdsDbPassword*> with your OTDS database password
- <*otdsPublicHostname*> with your Public OTDS host name
- <*otdsAdminPassword*> with your OTDS administrator password
- <*otdsCryptKey*> with the Base64 encrypted string value
- <*ingressClass*> with the Ingress class used for your Kubernetes environment (for example, nginx)

## 5.4 Adjust Default Server Resources

The default deployment of Output Transformation Server allocates sufficient compute resources for a demonstration environment. It does not provide sufficient resources for a production environment. Before you deploy Output Transformation Serverfor production use, you should review and adjust the default resource allocations for each pod.

You can allocate compute resources for your environment by editing the relevant sections of the `values.yaml` directly.

### 5.4.1 Resource parameters

This section explains how to change the resource limits and requests listed in the `values.yaml` file so that it contains values appropriate for your deployment. The resources section has the following appearance and applies to each `ots-server` pod:

```
resources:
     # -- If enabled, enforce Kubernetes resource limits for current deployment size.
     enabled: false
     # -- Resource configuration for OTS default tier
     OTS:
        default:
           resources:
              limits:
                 cpu: 3000m
                 memory: 6Gi
              requests:
                 cpu: 3000m
                 memory: 6Gi
           pvcSizes:
              da: 13Gi
           maxJobs: 5
```

**`limits` and `requests`**
Controls how much CPU and memory resources to allocate to the pods. The limits section indicates the highest amount of CPU and memory resources the container can use. Once a container reaches these limits, the container is restricted. The requests section indicates the amount of each resource that the container requires to run. Kubernetes will only schedule pods on nodes that can provide the needed resources. You must be aware that the limit settings cannot be smaller than the request settings because the conflicting values will result in the container not running and an error being thrown.

**`pvcSizes`**
Indicates a request for storage, which is also known as a PersistentVolumeClaim. If you opted to deploy Document Accessibility (da), you must specify the amount of storage to provide for the deployment.

**`maxJobs`**
Specifies the maximum number of jobs that can be run simultaneously.

## 5.5 Deploy Output Transformation Server Containers

This section explains how to create a new deployment of Output Transformation Server.

It presumes that your deployment has the following default settings:

- PostgreSQL database server runs in your Kubernetes cluster. It provides the database for Output Transformation Server.
- OTDS runs in your Kubernetes cluster, providing authentication services to the other OpenText components in the deployment.
- Output Transformation Server runs in your Kubernetes cluster on Apache Tomcat.

You must customize the deployment commands by typing the variable property values, represented with angle brackets, for your own environment.

**Note:** The commands shown below contain the backslash (\), which is only recognized in Linux environments. If you want to execute the commands with Windows PowerShell instead, replace the backslash (\) with a backtick (`).

### 5.5.1 Required deployment values for typical installations

At a minimum, a typical installation should specify a platform file to configure deployment parameters for your environment. To get started, the `default-k8s.yaml` can be used and tweaked as necessary for the target environment. For more information about the parameters, consult the embedded comments in the `default-k8s.yaml` file.

### 5.5.2 Deploying Output Transformation Server using Helm

To deploy Output Transformation Server, open the folder where you extracted the Helm chart files. In this folder, you must execute a command similar to the following:

```
helm install ots-server \
--values platform/default-k8s.yaml \
./ots-server \
```

**Note:** Notice that the `default-k8s.yaml` file is specified as an additional parameter. You must include this in your installation command.

After you execute the command, output similar to the following appears:

```
NAME: ots-server
LAST DEPLOYED: Fri Jan  1 17:00:28 2021
NAMESPACE: default
STATUS: deployed
REVISION: 1
TEST SUITE: None
NOTES:
**********************************
OTS Helm Chart 22.4
**********************************
OTS ingress configured for host ots.localhost

Deployment Size: default

OTSManager
URL: ots.localhost/OTSManager
```

It will normally take a few moments to start up every component in the deployment and have them connect to each other. You can monitor the progress of the deployment of the Kubernetes pods by executing `kubectl get pods -w` and by monitoring the deployment logs (see "Verify the Deployment" on page 43).

Chapter 6

# Customize the Deployment

This chapter describes modifications that you can make to the default deployment of your Output Transformation Server product before you deploy it in the cloud using Helm.

You can modify the deployment of your Output Transformation Server product in two principal ways:

**Edit the files in the Helm charts**

A Helm chart is a folder hierarchy containing YAML files, XML files, and other assets. You can edit the values in these files to modify your Helm deployment.

If you have made modifications to any values in the Helm charts and want to apply them to an existing deployment, you must run the following command:

```
helm upgrade <deployment-name> ./ots-server
```

**Apply Helm command-line arguments**

When you run your Helm deployment command, you can apply command-line arguments. Applying a command-line argument does not modify your Helm charts. It only modifies your Helm deployment.

To make modifications to a deployment using the Helm command line, run the Helm command with the new values. For example:

```
helm upgrade <deployment-name> ./ots-server
--set property1=value1
--set property2=value2
```

OpenText recommends that, as a general rule, you do not change the files in the Helm charts. Instead you should customize the deployment of your Output Transformation Server product using Helm command-line arguments (not by directly editing the Helm chart files). For that reason, this section of the guide describes how to make modifications using Helm command-line arguments, wherever possible.

The majority of the changes in this section can also be made by editing the `values.yaml` file, which is included in the `ots-server` Helm chart. A structured values file, such as `values.yaml`, is a standard part of a Helm deployment. By examining the structure of the command-line arguments in this section, you can infer the location of the property in the `values.yaml` file and make changes to the property there, if you prefer. For more information on this topic, refer to the Helm documentation. For example, see the Helm documentation on Values Files (https://helm.sh/docs/chart_template_guide/values_files/).

# 6.1 Deployment Logging

You can set the log levels of various containers to provide varying amounts of information on their deployment.

## 6.1.1 OpenText Domain Services (OTDS) Logging

To set verbose REST client logging for the OTDS container in the deployment, use the following command argument:

```
--set OTSServer.OTDSInitJob.verboseLog=true
```

By default, verbose logging is turned off and should be turned on for debugging purposes.

# 6.2 Passwords

A default deployment of Output Transformation Server and its related products sets the password of the various users involved in the deployment with basic defaults. You can change the passwords of these users after startup, but if you prefer to deploy Output Transformation Server with more secure user passwords, you can.

There are several ways that you can set passwords for your deployment at the time you run your Helm command:

- Use suitable Helm command-line arguments. See "Helm Command-line Arguments for Individual Passwords" on page 35.

**Notes**

- Any password that you specify must conform to the default password complexity rules in OTDS: it must have at least eight characters, and it must include one uppercase letter, one lowercase letter, one digit, and one symbol.
- You can set different passwords using Helm command-line arguments and Kubernetes Secrets concurrently, but if you set the same password using both methods, the password set in the Kubernetes Secret is the one that is applied.

### 6.2.1 Helm Command-line Arguments for Individual Passwords

The users that are listed in the `values.yaml` file fulfill varying roles in a deployment of Output Transformation Server. You can set their passwords individually by applying a Helm command-line argument when you deploy your product.

**Note:** In some cases, the same user password may need to be updated in more than one command-line argument.

This section explains how to change the passwords of the users listed in the `values.yaml` file of the `ots-server` Helm chart. (You can also change the *names* of users in the `values.yaml` file, but this is not normally advisable or necessary.) This section of the documentation is organized by chart. It lists users and passwords in the order that they are found in the `values.yaml` file.

#### 6.2.1.1 Passwords for Alternate Text Manager (ATM)

The passwords of the Alternate Text Manager database user and administrator are set in the `ATM` chart.

##### 6.2.1.1.1 Alternate Text Manager Database User

This is the Alternate Text Manager database user (`dbUser`) that interacts with the Alternate Text Manager database.

To change the password of the Alternate Text Manager database user, use the Helm command-line argument `--set ATM.dbPass=<Password>`

##### 6.2.1.1.2 Alternate Text Manager Admin User

This is the Alternate Text Manager administrative user (`adminUser`) that exists in every Alternate Text Manager deployment.

To change the password of the Alternate Text Manager admin user, use the Helm command-line argument `--set ATM.adminPass=<Password>`

#### 6.2.1.2 Passwords for Configuration Manager (CM)

The passwords of the Configuration Manager database user and administrator are set in the `CM` chart.

#### 6.2.1.2.1 Configuration Manager Database User

This is the Configuration Manager database user (`dbUser`) that interacts with the Configuration Manager database.

To change the password of the Configuration Manager database user, use the Helm command-line argument `--set CM.dbPass=<Password>`

#### 6.2.1.2.2 Configuration Manager Admin User

This is the Configuration Manager administrative user (`cmadmin`) that exists in every Configuration Manager deployment.

To change the password of the Configuration Manager admin user, use the Helm command-line argument `--set CM.adminPass=<Password>`

### 6.2.1.3 Passwords for Document Accessibility (DA)

The passwords of the Document Accessibility database user and administrator are set in the `DA` chart.

#### 6.2.1.3.1 Document Accessibility Database User

This is the Document Accessibility database user (`dbUser`) that interacts with the Document Accessibility database.

To change the password of the Configuration Manager database user, use the Helm command-line argument `--set DA.dbPass=<Password>`

#### 6.2.1.3.2 Document Accessibility Admin User

This is the Document Accessibility administrative user (`adminUser`) that exists in every Document Accessibility deployment.

To change the password of the Document Accessibility admin user, use the Helm command-line argument `--set DA.adminPass=<Password>`

### 6.2.1.4 Passwords for Output Transformation Server

The passwords of the Output Transformation Server database administrator user and main Output Transformation Server Manager administrator are set in the `OTSServer` chart.

#### 6.2.1.4.1 Output Transformation Server Database Administrator User

Output Transformation Server can use a database administrator user (`dbAdminUser`) to create the initial tables and database users for the Configuration Manager, Document Accessibility, and Alternate Text Manager deployments.

To change the password of the Output Transformation Server database administrator user, use the Helm command-line argument `--set OTSServer.dbAdminPass=<Password>`

However, in some environments, due to security policy, use of the database administrator account is not allowed. In this case, Output Transformation Server must use existing database users to create the databases and tables for Configuration Manager, Document Accessibility, and Alternate Text Manager as required.

To configure Output Transformation Server to use existing users for database and table creation, set `dbAdminUser` and `dbAdminPass` with blank values. Subsequently, the database users specified by `ATM.dbUser`, `CM.dbUser`, and `DA.dbUser` are assumed to already exist. Furthermore, you must ensure that these users hold the proper permissions to create databases.

#### 6.2.1.4.2 Output Transformation Server Manager Administrator User

This is the Output Transformation Server Manager administrator user (`adminUser`) that interacts with the Output Transformation Server Manager console. This administrative user exists in every Output Transformation Server deployment.

To change the password of the Output Transformation Server Manager administrator user, use the Helm command-line argument `--set OTSServer.adminPass=<Password>`

## 6.3 Modify Output Transformation Server

This section describes changes that you can make to the deployment of Output Transformation Server.

### 6.3.1 Output Transformation Server Port

In a default deployment of Output Transformation Server, the `ots-server` service runs on port 8080.

To set the port of the `ots-server` service to a number other than 8080, use the following Helm command-line argument:

```
--set otsserver.port=<Port_Number>
```

**Important**

The `ots-server` port can be modified only on the initial deployment. Do not change the `ots-server` port after you have deployed Output Transformation Server.

### 6.3.2 Add Alternate Text Manager

On top of the default deployment of Output Transformation Server, you can also add an optional deployment of Alternate Text Manager.

**To enable Alternate Text Manager in a deployment:**

```
--set ATM.enabled=true
```

Using this Helm command-line argument deploys Alternate Text Manger with the default passwords for administrator and database users. If you prefer to configure your own passwords, see "Helm Command-line Arguments for Individual Passwords" on page 35.

### 6.3.3 Add Document Accessibility

On top of the default deployment of Output Transformation Server, you can also add an optional deployment of Document Accessibility.

**To enable Document Accessibility in a deployment:**

```
--set DA.enabled=true
```

Using this Helm command-line argument deploys Document Accessibility with the default passwords for administrator and database users. If you prefer to configure your own passwords, see "Helm Command-line Arguments for Individual Passwords" on page 35.

## 6.4 Modify the Database Server Information

A default deployment of your Output Transformation Server product uses an instance of PostgreSQL database server to house the Configuration Manager, Alternate Text Manager, and Document Accessibility database.

**To specify the connection to the Postgres database server**

```
--set OTSServer.dbHost=<Database_Host_Name> `
```

**To set the port number for the Postgres database server**

```
--set OTSServer.dbPort=<Database_Port> `
```

**To set the admin user name and password for the Postgres database server**

```
--set OTSServer.dbAdminUser=<Administrator_User> `

--set OTSServer.dbAdminPass=<Administrator_Password> `
```

**Note:** The Administrator user name and password values are only required if you want to create new tables and database users for Configuration Manager, Alternate Text Manager, and Document Accessibility. If these values are left blank, it is assumed that the required database users already exist.

## 6.5 Modify OTDS

A default deployment of your Output Transformation Server product deploys an OTDS instance that provides authentication services to the other OpenText components in the deployment.

To use your organization's OTDS server rather than the OTDS container deployed by default, include the following Helm command-line parameters, and edit the `values.yaml` file, as described below.

**To connect to a different OTDS server**

```
--set OTDS.host=<OTDS_URL>
```

**To set the name of the OTDS user**

```
--set OTDS.otadminUser=<OTDS_user_name>
```

**To set the password of the OTDS user**

```
--set OTDS.otadminPass=<OTDS_Password>
```

## 6.6 Enabling SSL

SSL/TLS protocols are not required for deployment but can be used to establish encrypted communications for Output Transformation Server deployments.

The `ots-server` Helm chart contains a TLS client option that enables use of a custom Output Transformation Server keystore when initiating a TLS connection instead of the default JVM keystore. This option should only be used when the Output Transformation Server JVM cannot validate a server certificate or CA certificate based on its internal cacerts database of well-known Certificate Authorities.

When `OTSServer.tls.client=false`, the Output Transformation Server JVM uses its built-in cacerts database to validate TLS server certificates. This is the default setting.

When `OTSServer.tls.client=true`, the Output Transformation Server JVM uses a custom truststore specified in the Helm chart location `ots-server/truststore.pkcs12` to validate the TLS server certificates.

If you are uncertain about enabling the TLS client option, it is recommended you initially try working with the option turned off, as this uses the JVM cacerts database

for certificate validation. Subsequently, if the application begins to encounter TLS errors, then custom certificates may be required. For example, this option may need to be enabled for PostgreSQL SSL functionality, and/or HTTPS connections to OTDS.

In summary, when enabling the TLS client option, you must:

- Set the `OTSServer.tls.client=` option to true.
- Import the required certificates into a PKCS12 truststore in the `ots-server/truststore.pkcs12` file.
- Configure the `security.tls.client.trustStorePassword=` option with the truststore password you previously configured when creating a truststore. For more information, see "Creating a Truststore and Importing truststore certificates" on page 40.

### 6.6.1 Creating a Truststore and Importing truststore certificates

The following components may require certificates:

- If using TLS for OTDS, a certificate for OTDS identified by the public hostname, `otds.crt`
- If using TLS for Postgres, a certificate for Postgres identified by the internal hostname, `postgres.crt`

If a `ots-server\truststore.pkcs12` already exists in the Helm chart directory, you must delete it before continuing. This ensures that a new truststore is created when the following commands are run.

The following commands assume that all certificates mentioned above are placed in the `ots-server` Helm chart directory and that the keytool commands are being run from this file path location. These commands will import certificates into a PKCS12 keystore file, `ots-server\truststore.pkcs12`.

**Note:** If the truststore file doesn't exist, running these commands will create a new file.

Choose a password for *<truststore_password>* and set this as the value for the `security.tls.client.trustStorePassword` option in your Helm chart. Make note of the password because it is required when running commands.

To import an OTDS TLS certificate, assuming the certificate is named `otds.crt`, run the following command:

```
keytool -import -keystore ots-server\truststore.pkcs12 -storepass
<truststore_password> -file otds.crt -storetype pkcs12 -noprompt -alias otds
```

To import a Postgres TLS certificate, assuming the certificate is named `postgres.crt`, run the following command:

```
keytool -import -keystore ots-server\truststore.pkcs12 -storepass
<truststore_password> -file postgres.crt -storetype pkcs12 -noprompt -alias
postgres
```

Chapter 7

# Verify the Deployment

## 7.1 Output Transformation Server

An Output Transformation Server pod is deployed.

**Output Transformation Server**

An Output Transformation Server pod is deployed with the name `ots-server`. To verify the operation of the container, you can log on to Output Transformation Server Manager as an Output Transformation Server Administrator. Construct the URL for Output Transformation Server Manager using the host name that you set up in "Set Deployment Parameters for a Generic Deployment" on page 22, as follows:

`https://<OTSServerPublicHostname>/OTSManager`

For example: `https://localhost/OTSManager`

When the Output Transformation Server Manager logon page appears, log on as `Administrator`.

### 7.1.1 Output Transformation Server Install Verification Test

Following installation of your Output Transformation Server product, an Install Verification Test is available to validate basic functionality of the product by running some sample projects. The test can be run in Output Transformation Server Manager, which is an administrator console for Output Transformation Server.

**To run the Install Verification Test:**

1. In your preferred browser, navigate to your Output Transformation Server Manager instance. The console is located on the server specified in your `global.OTSServerPublicHostname=` value followed by `/OTSManager`.

   For example, if the value of `global.OTSServerPublicHostname=` is `https://myotpublicurl.example.com`, enter `https://myotpublicurl.example.com/OTSManager` to access the console.

2. On the **Account Login** page, log in using the Administrator account credentials. The Administrator's user name and password are set by the `OTSServer.adminUser` and `OTSServer.adminPass` parameters, respectively, in your Helm chart. By default, the values are set to `Administrator` for the user name and `Xenos@123!` as the password.

3. On the main Output Transformation Server Manager page, click **Resources**.

4. On the **Resources** tab screen, click **Component Pool**.

5. On the **Component Pool** screen, the **installVerificationTest** component is shown in the list. Click **Execute** to run the component.
6. In the **Execute Job** dialog box, click **Execute**.
7. When the job is finished running, verify that the **Status** shown in the summary displays **FINISHED**.

### 7.1.2 Output Transformation Server Deployment Log File

You can obtain information on the deployment of Output Transformation Server by using the kubectl logs command on your Kubernetes host. For example, to monitor log output during the deployment of Output Transformation Server, run `kubectl logs ots-server --follow`.

## 7.2 OTDS

To verify operation of OTDS, log on to it as an OTDS user. Construct the URL of the OTDS web interface, using the host name that you set up in "Deploy Output Transformation Server Containers" on page 30, as follows:

```
https://otds.<hostname>/otds-admin
```

For example: `https://otds.localhost/otds-admin`

When the logon page appears, log on to OTDS as `otadmin@otds.admin`, using the password you set during installation with the OTDS Helm chart.

### 7.2.1 OTDS Deployment Log File

You can obtain information on the deployment of OTDS by using the `kubectl logs` command on your Kubernetes host. For example, to monitor log output during the deployment of OTDS, run `kubectl logs <pod_name> --follow`. Replace *<pod_name>* with the appropriate OTDS container name you want to review. For OTDS 21.3, pod names begin with `opendj*` or `otdsws*`.

## 7.3 Alternate Text Manager

If your deployment includes Alternate Text Manager, perform the steps in this section to verify the integration of Output Transformation Server and Alternate Text Manager.

### 7.3.1 Verify functionality of Alternate Text Manager

Following installation of your Alternate Text Manager product, you can validate basic functionality by running a sample file in the console.

**To verify basic functionality of Alternate Text Manager:**

1. In your preferred browser, navigate to your Alternate Text Manager instance. The console is located on the server specified in your `global.OTSServerPublicHostname=` value followed by `/csalttextadmin`.

   For example, if the value of `global.OTSServerPublicHostname=` is `https://myotpublicurl.example.com`, enter `https://myotpublicurl.example.com/csalttextadmin` to access the console.

2. On the **Account Login** page, log in using the Administrator account credentials. The Administrator's user name and password are set by the `ATM.adminUser` and `ATM.adminPass` parameters, respectively, in your Helm chart. By default, the values are set to `ATMAdmin` for the user name and `Xenos@123!` as the password.

3. On the **Upload** page, click **Choose File**.

4. In the file browser, navigate to the `<ots-helm-chart-directory>\samples` directory and select the `alttext.arp` file.

5. On the **Preview** page that appears, verify that a new image is shown.

6. Next, on the **Preview** page, select the image and click **Delete**.

   Once you confirm that the image was deleted and no longer appears, functionality has been verified.

## 7.4 Document Accessibility

If your deployment includes Document Accessibility, perform the steps in this section to verify the integration of Output Transformation Server and Document Accessibility.

### 7.4.1 Verify functionality in Document Accessibility

Following installation of your Document Accessibility product, you can validate basic functionality by running a sample file in the console.

**To verify basic functionality of Document Accessibility:**

1. In your preferred browser, navigate to your Document Accessibility instance. The console is located on the server specified in your `global.OTSServerPublicHostname=` value followed by `/remediation`.

   For example, if the value of `global.OTSServerPublicHostname=` is `https://myotpublicurl.example.com`, enter `https://myotpublicurl.example.com/remediation` to access the console.

2. On the login page, log in using the Administrator account credentials. The Administrator's user name and password are set by the `DA.adminUser` and `DA.adminPass` parameters, respectively, in your Helm chart. By default, the values are set to `DAAdmin` for the user name and `Xonos@123!` as the password.
3. On the **Home** screen, click **Add Item**,  , and select **Document**.
4. In the file browser, navigate to the `<ots-helm-chart-directory>\samples` directory and select the `sample.pdf` file.
5. Verify that the **Preflight successful** dialog box appears and then click **OK**.
6. On the Home screen, select the `sample.pdf` document and click **Delete**.

   Once you confirm that the document was deleted and no longer appears, functionality has been verified.

Chapter 8

# Scale the Deployment

When deploying the `ots-server` pods using Helm and Kubernetes, you define how many replicas of the application you'd like to run.

The `replicas` section has the following appearance:

```
replicas:
  # -- number of replicas for OTS default tier
  OTS:
    default: 2
```

By default, the `ots-server` deployment is deployed with two replicas. To scale an application, you increase or decrease the number of replicas to reflect the number of `ots-server` pod replicas to deploy.

Chapter 9

# Upgrade the Deployment

To upgrade your Output Transformation Server deployment from a previous version, run a helm upgrade command with similar parameters to the helm install that you used to deploy it.

For example, presume that your initial deployment command looked like this:

```
helm install ots-server \
--values platform/default-k8s.yaml \
./ots-server
```

To upgrade your deployment to the next version of Output Transformation Server, execute the following command:

```
helm upgrade ots-server \
 --values platform/default-k8s.yaml \
 ./ots-server \
--set OTSServer.image.tag=<OTSUpgradeVersion>
```

When running the upgrade command, make sure to set the values for <*OTSUpgradeVersion*> where <*OTSUpgradeVersion*> indicates the image tag name of the version to update to.

**Tip:** For a list of your running Helm releases, run: `helm ls`.

## 9.1 Upgrade OTDS to version 22.4

To upgrade your OTDS database for use with deployments in version 22.4, additional steps are required.

If you are updating OTDS from a version earlier than 22.1, you must migrate your backend from OpenDJ to PostgreSQL because OpenDJ is no longer supported. For more information about the migration process, see "Migrate OTDS backend from OpenDJ to PostgreSQL" on page 50.

If you are using OTDS 22.1 or later, you only need to run the upgrade OTDS database command. For more information about upgrading OTDS, see "Upgrade the OTDS database" on page 51.

**Note:** For more information about upgrading OTDS from previous versions, see *OpenText Directory Services Cloud Deployment Guide*.

### 9.1.1 Migrate OTDS backend from OpenDJ to PostgreSQL

If you are updating OTDS from a version earlier than 22.1, you must migrate your backend from OpenDJ to PostgreSQL because OpenDJ is no longer supported. The following overview lists the multiple parts involved in the migration:

1. Create the OTDS user and OTDS database in PostgreSQL. For more information about creating the OTDS user and database, see "Setting up PostgreSQL for OTDS" on page 11.

2. Generate the crypt key string required for OTDS. In the command, *<otdsCryptKey>* represents the string value of the crypt key. For more information about generating the string, see "Generating the OTDS Encryption Key" on page 28.

3. Determine the namespace and service name of the existing OpenDJ service as these are required when running the migration command. For more information about finding these OpenDJ service names, see "Identify OpenDJ service names" on page 51.

4. Extract the OTDS Helm chart into a new folder. Make note of this folder as it is required as the *<otdsHelmChartFolder>* value when running subsequent commands.

5. Run the migration command:

```
helm install otds-migration <otdsHelmChartFolder> `
--set global.otdsUseReleaseName=true `
--set otdsws.image.source='<imageHost>/<imagePath>' `
--set otdsws.otdsdb.url='jdbc:postgresql://<otdsDbHostname>:5432/otds' `
--set otdsws.otdsdb.username=<otdsDbUser> `
--set otdsws.otdsdb.password=<otdsDBPassword> `
--set otdsws.publicHostname=<otdsPublicHostname> `
--set otdsws.otadminPassword=<otdsAdminPassword> `
--set otdsws.cryptKey='<otdsCryptkey>' `
--set otdsws.replicas=1 `
--set ingress.enabled=false `
--set otdsws.migration.enabled=true `
--set otdsws.migration.servicename=<opendj-ServiceName>.<opendj-
NameSpace>.svc.cluster.local `
--set otdsws.migration.password=otds
```

6. Run `kubectl get pods` to display a list of the currently running pods and locate the name of the OTDS migration pod. The name should start with `otds-migration*`. Make note of the migration pod name as it is required as the *<otdsMigrationPodName>* value when running subsequent commands.

7. Run `kubectl logs <otdsMigrationPodName>` to display the `otds-migration` pod logs.

   Examine the logs for any errors. In the logs, a successful migration from OpenDJ generates messages similar to the following:

```
2022-11-09 20:27:59.756|OTDS        |INFO
||[main]||TenantImporter||Import from OpenDJ completed with 0 errors and 0 warnings
```

8. Lastly, uninstall the `otds-migration` release. Run `helm uninstall otds-migration` to initiate the uninstall process.

#### 9.1.1.1 Identify OpenDJ service names

To identify OpenDJ service names:

1. Run `kubectl get service` to display a listing of the existing OTDS namespaces, which will show the namespace and service name of your existing OpenDJ service.

2. A list of services similar to the following appears:

```
NAME                                TYPE        CLUSTER-
IP

kubernetes                          ClusterIP   10.99.0.1
opendj                              ClusterIP   10.999.201.47
otdsws                              ClusterIP   10.99.177.242
ots-server-service                  ClusterIP   10.99.41.25
```

   Locate the name of your OpenDJ service in the list. In this example, the OpenDJ service is named **opendj**. Make note of your service name as it is required as the *<opendj-ServiceName>* value when running the migration command.

### 9.1.2 Upgrade the OTDS database

If you are using OTDS with an earlier version than 22.1, then you must migrate your information from OpenDJ to PostgreSQL before continuing with the OTDS upgrade. For more information about the migration, see "Migrate OTDS backend from OpenDJ to PostgreSQL" on page 50.

If you are using OTDS 22.1 or later, you can upgrade OTDS using the following command.

To upgrade OTDS:

1. Run the Helm upgrade command:

```
helm upgrade otds-deployment ./otds/ `
--set otdsws.image.source='<imageHost>/<imagePath>' `
--set otdsws.otdsdb.url='jdbc:postgresql://<otds_db_hostname>:5432/otds' `
--set otdsws.otdsdb.username=otds `
--set otdsws.otdsdb.password=<otdsDbPassword> `
--set otdsws.publicHostname=<otdsPublicHostname> `
--set otdsws.otadminPassword=<otdsAdminPassword>  `
--set otdsws.cryptKey='<otdsCryptkey>' `
--set otdsws.replicas=<otdsPodCount> `
--set ingress.enabled=true `
--set ingress.class=<ingressClass>
```

2. Restart the Output Transformation Server pods by running:

```
kubectl  scale deployment ots-server-deployment --replicas=0
kubectl  scale deployment ots-server-deployment --replicas=<replicaCount>
```

Chapter 10

# Configuring docker containers

Docker container images are provided to install Output Transformation Server on a cloud platform. Depending on the options selected, you can deploy a default installation of Output Transformation Server, which only has core Output Transformation Server functionality, or opt to include the Configuration Manager, Document Accessibility, or Alternate Text Manager modules.

## 10.1 Output Transformation Server container configuration

Many configuration options are available to be applied to the Output Transformation Server image to define how the application initializes.

### Environment variables

Listed in the following table are all available database types that can be used as values for DB_TYPE.

**Table 10-1: Database type values for the DB_TYPE variable**

| Name | Default Value |
|---|---|
| POSTGRES | PostgreSQL |
| MYSQL | MySQL |
| MSSQL | Microsoft SQL Server |
| ORACLE | Oracle |

Listed in the following table are all database connection types that can be used to specify the database connection type for the various databases.

**Table 10-2: Database connection type variables**

| Name | Default Value | Description |
|---|---|---|
| DB_JDBC_CON | JDBC | JDBC connection type |
| DB_JNDI_CON | JNDI | JNDI connection type |
| DB_PERSISTENT_STORAGE_CON | PersistentStorage | Persistent Storage connection type |

Listed in the following table are variables to specify database information, which is used for all services.

**Table 10-3: Database information variables**

| Name | Default Value | Description |
|---|---|---|
| DB_TYPE | $POSTGRES | Denotes the database type. Currently, Output Transformation Server only supports PostgreSQL and Document Accessibility only supports PostgreSQL and Oracle. |
| DB_PORT | 5432 | Indicates the default PostgreSQL port. |
| DB_ADMIN_USER | postgres | Specifies the default user name for Administrator. |
| DB_ADMIN_USER_PWD | postgres | Specifies the default user password for Administrator. |

Listed in the following table are the Output Transformation Server environment variables that specify various aspects about the application's settings.

**Table 10-4: Output Transformation Server environment variables**

| Name | Default Value | Description |
|---|---|---|
| OTS_BASEREPOSITORY | `/OpenText/ots/BaseRepositories/tomcat/ots-tc` | Specifies the base repository location on the container. |
| OTS_PORT | 8080 | Denotes the default port that Tomcat uses. |
| OTS_HOST | localhost | Denotes the default host that all webservices use. |
| OTS_PROTOCOL | http | Denotes the default protocol that all webservices use. |
| OTS_CLUSTER_ENABLE | true | Indicates whether the network cluster cache is enabled, which allows nodes to communicate with each other via a shared network cache. |
| OTS_SYSTEM_ID | ots | Designates the name for the cluster. All Output Transformation Server instances with the same ID will join this cluster. |

Listed in the following table are Configuration Manager environment variables. By default, Configuration Manager is disabled.

**Table 10-5: Configuration Manager environment variables**

| Name | Default Value | Description |
|---|---|---|
| CM_ENABLE | false | Specifies whether to use Configuration Manager. If set to true, when starting Output Transformation Server the application is automatically configured and the database is created. |
| CM_DB_NAME | cm | Denotes the name of the database. |
| CM_DB_EXISTS | false | Indicates that the existing database is used.<br><br>**Note:** Do not try to create the database. |
| CM_DB_USER_EXISTS | false | Indicates that the existing database user is used.<br><br>**Note:** Do not try to create the user. |
| CM_AUTO_SYNC | true | Denotes whether to automatically update resources if a newer version is available in Configuration Manager. This variable works in conjunction with CM_CHECK_INTERVAL. |
| CM_CHECK_INTERVAL | 60 | Designates the interval, in minutes, that Configuration Manager checks if a newer version of a resource is available. This variable is only used with CM_AUTO_SYNC is set to true. |
| CM_CACHE_ENABLE | false | Specifies whether the network shared cache is used across separate Configuration Manager instances. This notifies other nodes in the cluster when a new version of a resource is available. |
| CM_DB_USER | cmuser | Specifies the Configuration Manager user name to use. For Oracle, a schema is created using this name. |

| Name | Default Value | Description |
|---|---|---|
| CM_DB_USER_PWD | cmuser | Specifies the password for the corresponding Configuration Manager user account. |
| CM_DB_NAME | cm | Denotes the name for the Configuration Manager database. This is used in the JDBC connection string. |
| CM_ADMIN_PWD | cmadmin | Specifies the Configuration Manager Administrator user name. |
| CM_ADMIN_OLD_PWD | cmadmin | Specifies the Configuration Manager Administrator password. |
| CM_OTS_AUTH_ENABLE | false | Denotes whether to use Output Transformation Server authentication instead of the built-in Configuration Manager authentication. |

Listed in the following table are Document Accessibility environment variables. By default, Document Accessibility is disabled.

**Table 10-6: Document Accessibility environment variables**

| Name | Default Value | Description |
|---|---|---|
| DA_ENABLE | false | Specifies whether to set up and start Document Accessibility. |
| DA_DB_NAME | otda | Denotes the name of the database. |
| DA_STORAGE_ROOT_PATH | /OpenText/ots/DA/storage | Designates the folder where all user projects are stored. The selected directory should be a volume override to a shared network location that all instances of Document Accessibility can access. |
| DA_TRAINING_DATA | /OpenText/ots/DA/training-data | Designates the folder where all training samples and models are stored. The selected directory should be a volume override to a shared network location that all instances of Document Accessibility can access. |

| Name | Default Value | Description |
|---|---|---|
| DA_DB_USER | dauser | Specifies the Document Accessibility user name to use. For Oracle, a schema is created using this name. |
| DA_DB_USER_PWD | dauser | Specifies the password for the corresponding Document Accessibility user account. |
| DA_DB_NAME | otda | Denotes the name for the Document Accessibility database. This is used in the JDBC connection string. |
| DA_CONFIG_DIR | otda | Designates the folder in the base repository where Document Accessibility configurations are stored.<br><br>**Note:** This may be used as the solution name that is used to store configurations in Configuration Manager. |
| DA_CONFIG_NAME | otda-1 | Specifies the base name to use for all Document Accessibility-related configurations. |
| DA_PERSISTENT_POOL_NAME | OTDAStorage | Designates the persistent storage pool name to use when creating a persistent storage entry in the `default.xSystemConfig` file. |
| DA_PREFLIGHT_CONNECTION_TYPE | local | Indicates how Preflight is invoked. |
| DA_PREFLIGHT_REMOTE_HOST | localhost | Denotes the host to use if Preflight is remote. |
| DA_PREFLIGHT_REMOTE_PORT | 8080 | Denotes the port to use if Preflight is remote. |
| DA_SERVICE_AUTOSTART | true | Automatically initializes the Document Accessibility service when Output Transformation Server is started. |

Listed in the following table are Alternate Text Manager environment variables. By default, Alternate Text Manager is disabled.

**Table 10-7: Alternate Text Manager environment variables**

| Name | Default Value | Description |
|---|---|---|
| ATM_ENABLE | false | Specifies whether to set up and start Alternate Text Manager. |
| ATM_DB_NAME | Atm | Denotes the name of the database. |
| ATM_DB_USER | atmuser | Specifies the Alternate Text Manager user name to use. For Oracle, a schema is created using this name. |
| ATM_DB_USER_PWD | atmuser | Specifies the password for the corresponding Alternate Text Manager user account. |
| ATM_DB_NAME | atm | Denotes the name for the Alternate Text Manager database. This is used in the JDBC connection string. |
| ATM_CONFIG_DIR | atm | Designates the folder in the base repository where Document Accessibility Alternate Text Manager configurations are stored.<br><br>**Note:** This may be used as the solution name that is used to store configurations in Configuration Manager. |
| ATM_CONFIG_NAME | atm-1 | Specifies the base name to use for all Alternate Text Manager-related configurations. |
| ATM_AUDIT_LOD_FILE | `/OpenText/ots/BaseRepositories/tomcat/ots-tc/common/output/logs/atm.log` | Designates the folder where the Alternate Text Manager audit logs are written. |
| ATM_CONN_TYPE | $DB_JDBC_CON | Denotes the database connection type to configure for Alternate Text Manager. |

| Name | Default Value | Description |
|---|---|---|
| ATM_PERSISTENT_POOL_NAME | ATMStorage | Designates the persistent storage pool name to use when creating a persistent storage entry in the `default.xSystemConfig` file. This variable only applies if ATM_CONN_TYPE= $DB_PERSISTENT_STORAGE_CON. |
| ATM_SERVICE_AUTOSTART | true | Automatically initializes the Alternate Text Manager service when Output Transformation Server is started. |

**Note:** On Oracle databases, the database name *_DB_NAME values typically all point to the same database (for instance, XE). Different database tables are usually partitioned by the user or schema name.

## Volumes

These volumes are folders used by Output Transformation Server that can be overridden to point to different path locations. By default, the I/O is performed on the docker container in the locations specified by the volume. When remapping to a new path, the I/O can be re-directed to a folder outside of the container so that the data is persisted and does not get deleted when the container is stopped.

**Table 10-8: Output Transformation Server container volumes**

| Volume | Description |
|---|---|
| $OTS_BASEREPOSITORY | Indicates where the Output Transformation Server base repository is created if it doesn't already exist. |
| $DA_STORAGE_ROOT_PATH | Designates the folder where Document Accessibility user projects are stored. The selected directory should be a shared network location if multiple instances of Output Transformation Server are deployed. |
| $DA_TRAINING_DATA | Indicates the folder where Document Accessibility training samples and models are stored. The selected directory should be a shared network location if multiple instances of Output Transformation Server are deployed. |

| Volume | Description |
|---|---|
| /OpenText/ots/custom/jars | Designates the folder where custom component jars can be stored.<br><br>**Note:** Developers can also add modified code here for testing purposes. |

## 10.2 Tomcat container configuration

A single environment variable is available for use with Tomcat.

**Table 10-9: Tomcat environment variables**

| Name | Default Value | Description |
|---|---|---|
| JAVA_OPTS | `-Xmx2048M` | Specifies additional Java arguments to use when the JVM starts. The default is to set the max heap size to 2 GB. Among other arguments that can be set is configuring remote debugging by setting an argument like this example:<br>`-Xdebug -Xrunjdwp:transport=dt_socket,server=y,suspend=n,address=8000` |