Data Exchange Framework 6.0 Tenant Web Service Container Deployment Guide

April 9, 2021

Table of Contents

1. Introduction to Tenant Service for containers

Sitecore Data Exchange Framework Tenant Service (TS) enables you to trigger data synchronization processes multiple systems.

For more information about TS, see the Sitecore developer documentation.

This guide shows you how to deploy TS to Sitecore container installations for Docker and Azure Kubernetes Service.

2. Prepare the installation

This section explains what you need to prepare for installing the Sitecore Data Exchange Framework Tenant Service (TS) in Sitecore containers for Docker and Azure Kubernetes Service (AKS).

2.1. Requirements

Before you deploy TS for Docker or AKS, you must have the following:

- Docker Desktop installed and running. For instructions on how to set up the Docker environment, see the Containers in Sitecore development documentation.
- Sitecore Docker container files on a local machine if the installation is done on Docker. For instructions on how to prepare the Sitecore containers, see the *Installation Guide for Developer Workstation with Containers* on the Sitecore download site.
- Sitecore AKS container files on a local machine if the installation is done on Kubernetes. For instructions on how to prepare a Sitecore environment with Kubernetes, see the *Installation Guide for Production Environment with Kubernetes* on the Sitecore download site.
- A build of the DEF Docker images on your local machine. For instructions on how to build the Data Exchange Framework images, see the *Data Exchange Framework container deployment* guide on the Sitecore download site, and follow the instructions on how to build the Docker images.

3. Add the Tenant Service module to Sitecore in Docker

To add Sitecore Data Exchange Framework (DEF) Tenant Service (TS) module in Docker, you must:

1. Prepare the installation files
2. Deploy the Tenant Service container

3.1. Prepare the installation files

To prepare the files you need for the installation:

1. Download the DEF container deployment package from the Sitecore Developer Portal. Extract it to your local workstation with the folder structure intact.
2. Go to the folder into which you extracted the DEF container deployment package. Locate the folder for the Windows version and topology you are using, for example, `TenantService.Runnable\compose\ltsc2019\xp1`.
3. Copy the contents of the topology folder to a new, temporary folder, for example `c:\tenant-service\xp1`.
4. In the folder, copy the `sample.env` file, and name the new file `.env`. Open the `.env` file in an editor. Scroll to the end of the file, and update the following variables to fit your installation:

```
#The host for the Tenant service website, for example, <topology-to-deploy>ts.localhost
TENANT_SERVICE_HST=<...>

#The version tag for the Tenant service image
MODULE_VERSION=<...>

#The Tenant Id after it is created in Sitecore
CMSITECORE_TENANT_ID=<...>

#The Client Id of Sitecore Identity Server
SITECORE_CLIENT_ID=<...>

#The Client Secret of Sitecore Identity Server
SITECORE_CLIENT_SECRET=<...>
```

5. Save the `.env` file.

3.2. Deploy the Tenant Service container

When you have prepared the installation files, you must deploy the DEF TS container.

To deploy a TS container:

1. In the Windows console, go to the folder containing the `tenant-service.override.yml` file. Run this command:

   ```
   docker-compose -f docker-compose.yml -f docker-compose.override.yml -f tenant-
   service.override.yml up --detach
   ```

2. To test if the tenant service is running, open a browser, and navigate to `https://<topology>ts.localhost`.
3. Add HTTPS/TLS certificates for the tenant service. For instructions on how to do this, see the *Installation Guide for Developer Workstation with Containers*.
4. To update your search indexes, browse to your Sitecore URL, for example, `https://xp0cm.localhost/`. Open the control panel and click **Populate Solr Managed Schema**. After Sitecore has populated the Solr Schema, click **Indexing Manager**.
5. In the Sitecore content management container, create a DEF tenant, and upgrade the tenant.
6. In the DEF tenant, create a DEF tenant service endpoint. Enable the DEF tenant and the DEF tenant service endpoint.
7. In your local deployment folder, open the `tenant-service.override.yml` file in a text editor. Remove the following lines:

   ```
   depends_on:
     id:
       condition: service_healthy
     cm:
       condition: service_healthy
   ```

8. Open a PowerShell window with administrator rights, and navigate to your local deployment folder.
9. Run this command:

   ```
   docker-compose -f tenant-service.override.yml up --detach
   ```

10. To test if the tenant is available, open a browser, and browse to `https://<topology>.localhost/api/tenant`.

NOTE
To learn how to apply configuration transforms, such as adding connection strings or changing the web configuration, see the Sitecore container development documentation.

3.3. Deploy a plugin to the Tenant Service container

If you want to use TS in a connector module such as, for example, Sitecore Connect for Salesforce CRM, you must deploy a plugin for that connector.

To deploy a plugin to the TS container:

1. In the Window console, get the name of the TS container by running this command:

   ```
   docker-compose ps
   ```

2. Stop the TS container by running this command:

   ```
   docker stop <container name>
   ```

3. Add the plugin by running this command:

   ```
   docker cp <source path> <container name>:"c:/inetpub/wwwroot"
   ```

   > **NOTE**
   > Replace `<source path>` with the path to the folder containing the plugin.

4. Start the TS container again by running this command:

   ```
   docker start <container name>
   ```

4. Add Tenant Service module to Sitecore in Azure Kubernetes Service

This section explains how to add Data Exchange Framework (DEF) Tenant Service (TS) in Azure Kubernetes Service (AKS).

> **NOTE**
>
> For AKS deployments, the preferrred URLs for TS connection strings are:
>
> ```
> host=http://cm.default.svc.cluster.local;
> auth endpoint=http://id.default.svc.cluster.local/
> ```

You must;

1. Prepare files and folders for deployment.
2. Deploy the containers using *kubectl* commands.

4.1. Prepare files and folders for deployment

To prepare files and folders in your Kubernetes installation for deployment:

1. Download the Sitecore Experience Platform container deployment package from the Sitecore download page. Extract the package and save it, with the folder structure intact, in a temporary folder on your local machine .
2. Download the Sitecore Data Exchange Framework container deployment package from the Sitecore download page. Extract the package and save it, with the folder structure intact, in a temporary folder on your local machine.
3. Copy the `TenantService.Runnable\\k8s\ltsc2019\<ts.yaml` file from Step 2 to the `k8s\<version>\<topology>` folder from Step 1.
4. In the `k8s\<version>\<topology>` folder from Step 1, open the `kustomization.yaml` file in a text editor.
5. At the end of the `images:` section, add a definition for the TS image. In the `newName` parameter, specify the TS image repository name. In the `newTag` parameter, specify the TS image version. For example:

   ```
   - name: sitecore-ts
     newName: scr.sitecore.com/sxp/modules/sitecore-tenant-service
     newTag: 6.0.0.01518.97-10.0.17763.1757-ltsc2019
   ```

6. At the end of the `resources:` section, add the line `- ts.yaml`. Save the file.
7. Go to the `k8s\<version>\<topology>\ingress-nginx` folder from Step 1. Open the `ingress.yaml` file in a text editor.

8. In the `spec:` section, at the end of the `rules:` section, add the following host and TLS definitions:

```
- host: ts.globalhost
  http:
    paths:
    - path: /
      backend:
        serviceName: ts
        servicePort: 80
tls:
-secretName: global-ts-tls
hosts:
-ts.globalhost
```

9. Save the file.
10. In the `secrets/tls/global-ts` folder, add the TLS certificates `tls.crt` and `tls.key` for the `ts.globalhost` host. To add HTTPS/TLS certificates for the tenant service see the *Installation Guide for Production Environment with Kubernetes* guide.
11. Navigate to the `k8s\<version>\<topology>\secrets\tls` folder. Open the `kustomization.yaml` file in a text editor, and add the following lines at the end:

```
-name: global-ts-tls
    files:
        -tls/global-ts/tls.key
        -tls/global-ts/tls.crt
    type: kubernetes.io/tls
```

12. Save the file.

4.2. Deploy the AKS container

To deploy the container and the necessary Kubernetes components:

1. Prepare the AKS cluster configuration and deploy the ingress controller. For information on how to do this, see the *Installation Guide for Production Environment for Kubernetes* which is available on the Sitecore download page.
2. Open the Windows console, and navigate to the `k8s\<version>\<topology>` folder.
3. Deploy the secrets. Use this command:

```
kubectl apply -k ./secrets/
```

4. Run the `external` folder. Use this command:

```
kubectl apply -k ./external/
```

5. Wait for all containers to have the status *Ok/Running*. You can check the status with this command:

```
kubectl get pods -o wide
```

6. Run the `init` folder. Use this command:

```
kubectl apply -k ./init/
```

7. Wait for all containers to have the status *Completed*. You can check the status with this command:

```
kubectl get pods
```

8. To create persistent volumes, run this command:

```
kubectl apply -f ./volumes/azurefile
```

9. Run the Sitecore containers with the SFCRM changes. Use this command:

```
kubectl apply -k ./
```

10. Wait for all containers to have the status *Ok/Running*. You can check the status with the *kubectl get pods* command. Obtain the external IP address by using this command:

```
kubectl get service -l app=nginx-ingress
```

11. Update the local host file with the external IP address. For information on how to do this, see the *Installation Guide for Production Environment for Kubernetes*, which is available on the Sitecore download page.
12. To check that the service is running, open a browser, and browse to `https:\\ts.globalhost`.

    > **NOTE**
    > If you get a **Your connection isn't private** response, click **Advanced**, then click **Continue to ts.globalhost (unsafe)** to continue.

> **NOTE**
> You can also use a strategic merge patch to deploy a tenant service in AKS.