TIBCO® Managed File Transfer Platform Server for UNIX Docker Container Deployment

Version 8.0.1

April 2022

Contents

Introduction to Docker Container Deployment

TIBCO® Managed File Transfer Platform Server for UNIX can be container enabled. This means that TIBCO Managed File Transfer (MFT) Platform Server for UNIX can have the following capabilities:

- It can be executed in the cloud.
- It supports Docker and Kubernetes.
- It can be used for on-site or off-site cloud environments.
- It simplifies deployment: install, upgrade, and hotfixes.
- It can be used to deploy configuration changes.
- It runs on most 64 bit flavors of Linux that supports Docker and Kubernetes.
- It supports Kubernetes for HA load balancing.

The Docker container is the basis for most cloud implementations, so you must follow the steps to create TIBCO MFT Platform Server for UNIX Docker images. Since each cloud implementation is different, the parameters supplied in the sample Docker file will not work for every customer site. The Docker file provided shows a sample of how TIBCO MFT Platform Server for UNIX can be configured.

You can use this document as a guideline to deploy TIBCO MFT Platform Server for UNIX using Docker container.

Kubernetes is generally only used when load balancing TIBCO MFT Platform Server for UNIX, therefore Kubernetes is not discussed in this document.

Requirements and Recommendation for Docker Container Deployment

You must check the requirements and follow the recommendation provided here for easy deployment.

You must have a solid understanding of Docker and should have installed and tested a current Docker version. Before you start deployment, check the following requirements:

- Docker is installed and operational.
- The necessary firewall ports are opened to allow communications.

It is recommended that you create images for the base version and for each hotfix that is installed. This allows you to easily deploy different versions of TIBCO MFT Platform Server for UNIX across your network.

For example: If you install version 8.0.1 and hotfixes 8.0.1_HF-001 and 8.0.1_HF-002, it is recommended that you create the following Docker images:

- TIBCO MFT Platform Server for UNIX 8.0.1
- TIBCO MFT Platform Server for UNIX 8.0.1 with Hotfix 8.0.1_HF-001
- TIBCO MFT Platform Server for UNIX 8.0.1 with Hotfix 8.0.1_HF-002

If you have already created a base TIBCO MFT Platform Server for UNIX image and want to install a hotfix, you can directly start with the following step of the cloud deployment process: Install TIBCO MFT Platform Server for UNIX Hotfixes.

Docker Container Deployment Process

To deploy TIBCO MFT Platform Server for UNIX using the Docker container, you must follow these steps that are a part of the process. Each step is documented in more detail in the sections that follow.

- Step 1: Select the base operating system.
- Step 2: Create a base TIBCO MFT Platform Server for UNIX installation.

  Sample Docker files are distributed in the following folder:`<PSU Install>/cloud`
- Step 3: Install TIBCO MFT Platform Server for UNIX hotfixes.
- Step 4: Configure the `startall.sh` script.
- Step 5: Configure the Docker file.
- Step 6: Build the Docker image.
- Step 7: Test the Docker image.
- Step 8: Save the Docker image to a repository.

Step 1: Select the Base Operating System

You must install TIBCO MFT Platform Server for UNIX on a Linux environment to create a TIBCO MFT Platform Server for UNIX Docker image. It is recommended to use the same Linux environment for the base installation that will be used for the Docker container.

Step 2: Create a Base TIBCO MFT Platform Server for UNIX Installation

You can install TIBCO MFT Platform Server for UNIX in two ways: as a root user or as a non-root user.

Procedure

1. Install TIBCO MFT Platform Server for UNIX. For installation instructions, see *TIBCO Managed File Transfer Platform Server for UNIX Installation*.

   Note: After installation, TIBCO MFT Platform Server for UNIX configuration files are saved in the `$CFROOT/config` directory.
2. Before you create the Docker images, configure the following components.

Component to be Configured	Program Used
Config.txt	Using a text editor such as “vi”
Nodes	Using the cfnode program
User profiles	Using the cfprofile program
Responder profiles	Using the cfrprofile program
Other config files, such as `AccessControl.cfg`, `CfAlias.cfg`, `CfgPostProc.cfg`, `Cfcos.cfg`, and `Cflist.cfg`	Using a text editor such as “vi"

3. Optional: If you use TLS or TLS Tunneling mode, you must do the following configurations also:

- Create the necessary certificates, private keys, and private key passwords and configure the SSL/TLS parameters in the `config.txt` file in both Server and Client sections.
- Update the trusted authority will as required.
- Update the `SSLAuth,cfg` files as required.
- Update the SSL parameters for the client and the server in the `config.txt` file.

For information on these configurations, see *TIBCO Managed File Transfer Platform Server for UNIX User's Guide*.

4. Configure the following parameters in the [COMMON] section of the `config.txt` file to use persistent storage when running in a production environment. This ensures that audit files are kept permanently, that there are no duplicate transaction ids, and that transfers can restart when the container is restarted.

Parameter	Description
`LogEventFileName`	Defines where the `Log.txt` files are saved. Each completed file transfer is logged in the `Log.txt` file. The permissions for the LogEventFileName directory must be the same as the permissions in the `$CFROOT/log` directory.
`PQFDirectory`	Defines where the PQF files are stored. PQF files save restart information about file transfers. The permissions for the PQF directory must be the same as the permissions in the `$CFROOT/PQF` directory.
`TransnumFileName`	Defines the name of the Transaction Number file. This file is used to keep track of the current active transaction number. It is not recommended to save this file in the config directory. It is recommended to save the `transnum` file in the in the same directory as the `Log.txt` files. Optionally, the `transnum` file can be saved in its own directory. The permissions for the TransnumFileName and the directory where this file is located must be the same as the the permissions in the `$CFROOT` directory.

Multiple Platform Server containers cannot point to the same persistent directories for the three parameter defined above, unless the server is a passive server running in an HA Active/Passive mode. If you point multiple Platform Server containers to the same directory, you will get duplicate transaction numbers and the `Log.txt` files that contain the audit records may be corrupted.

Step 3: Install the TIBCO MFT Platform Server for UNIX Hotfixes

The `readme` file that accompanies a hotfix provides instructions to install the hotfix. You can install the hotfix directly over the base installation that you created in the previous step. It is a good idea to create a

Docker image for each hotfix installation. That makes it easy for you to test and deploy different TIBCO MFT Platform Server for UNIX images.

Step 4: Configure the startall.sh Script

At startup time, the Docker **`run`** command executes the `startall.sh` script. The `startall.sh` script is located in the `$CFROOT/bin` directory.

Procedure

- Configure the `startall.sh` script based on your requirements.

  The `startall.sh` script performs the following functions:

  - Catches the "TERM" signal that is thrown when a Docker container stops. This allows you to copy the Docker Container configuration files to persistent storage. If you do not want to do this, comment out this step.
  - Sets the persistent storage directory names and ensures that the necessary persistent directories have been created.
  - Creates the following PSU environment variables: CFROOT, PATH and LD_LIBRARY_PATH.
  - At startup, copies the configuration files from persistent storage to ephemeral storage (`$CFROOT/config`) so that you can maintain configuration records when the Docker container is restarted. If you are deploying configuration changes contained in the Docker image, then you should comment out this step. When this is done, the configuration changes contained in the Docker image are used.
  - Starts the necessary CyberResp services. The following services can be started:

    1. IPV4, IPV4 with TLS, IPV4 with TLS Tunnel
    2. IPV6, IPV6 with TLS, IPV6 with TLS Tunnel
  - Executes "sleep forever". This is done because the Docker container terminates when PID 1 ends. So the `startall.sh` script is kept running forever.

  For information related to these configurations, see *TIBCO Managed File Transfer Platform Server for UNIX User's Guide*.

Step 5: Configure the Docker File

The Docker file is used to build the TIBCO MFT Platform Server for UNIX container.

Procedure

- Configure the Docker file based on your requirements.

  The Docker file performs the following functions:

  - Installs Linux version and debugging tools.
  - Sets environment variables that can be overridden by the Docker build.
  - Creates MFT groups and users.
  - Copies the Platform Server distribution. If you are running as a non-root user, make sure the directories have the correct rights and attributes.
  - Defines the user that the container runs under.
  - Sets the Platform Server CFROOT environment variable and create the `bash_profile` file.

  For information related to these configurations, see *TIBCO Managed File Transfer Platform Server for UNIX User's Guide*.

Step 6: Build the Docker Image

The Docker **`build`** command creates the Docker image using the Docker file. The command differs slightly depending on whether you are installing as a root user or as a non-root user.

Procedure

- Use one of the following commands based on whether you are a root user or a non-root user.

User	Command
Root User	`docker build -f Dockerfile.psu --build-arg CFROOT=/PSU801 --build-arg CFUSER=root -t library/psu8.0.1 ./`
Non-root User	`docker build -f Dockerfile.psu --build-arg CFROOT=/PSU801NonRoot --build-arg CFUSER=mftuser -t library/psu8.0.1nonroot ./`

Step 7: Test the Docker Image

Now that the Docker image is created, you must test the Docker image using the Docker **`run`** command. The Docker **`run`** command differs slightly for a root user and a non-root user. (The Docker **`run`** command executes the `startall.sh` script.)

The Docker **`run`** command allows you to redirect the container ports to the ports on the Docker host.

Make sure that the ports defined by the **`-p`** parameter are unique and available on the Docker host when running multiple Docker containers on the same Docker host.

Procedure

- To test the docker image, execute the Docker **`run`** command depending on whether you are a root or a non-root user. See the sample provided in the following table.

User	RUN Command Sample
Root User	`docker run -it --rm -p 46464:46464 -p 56565:56565 -p 58585:58585 -v /persiststorage:/psupersist:z library/psu8.0.1`
Non-root User	`docker run -it --rm docker run -it --rm -p 46464:46464 -p 56565:56565 -p 58585:58585 -u mftuser:mftuser -v /persiststorage:/psupersist:z library/psu8.0.1`

The following table describes the parameters in the **`run`** command:

Parameter	Descriptions
-p 46464:46464 -p 56565:56565 -p 58585:58585	Maps the ports used by the container to the ports accessible by the network
-u mftuser:mftuser	Defines the user ID and group ID used by the container. This is required only when running the container as a non-root user. The user defined in this command must match the user created in the `dockerfile`.

Parameter	Descriptions
-v /psupersiststorage:/psupersist:z:	Defines the persistent storage used by the container. This storage is used for saving critical files across container restarts. The persistent storage defined must match the persistent storage directory defined in the `startall.sh` script.
-library/psu8.0.1	Defines the name of the docker image.

If you have made any changes to the Docker image that you want to make permanent, you can use the Docker **`commit`** command to commit the changes to an image. However, make sure to do this before the Docker image is removed.

Step 8: Save the Docker Image to a Repository

Now that the Docker image has been built and tested, you can save the Docker image to a Docker registry. You must save this to a private registry; TIBCO Licensing does not allow you to save TIBCO MFT Platform Server for UNIX Docker images to a public registry.

Procedure

1. Use the Docker **`push`** command to save the image to a registry.
2. After the image is saved in a registry, use the Docker **`pull`** command to retrieve the image from the registry and save it to the local machine.

Result

Now all the TIBCO MFT Platform Server for UNIX images can be deployed using the new Docker container.