TIBCO® Managed File Transfer Platform Server for UNIX

Docker Container Deployment

Version 8.1.0
August 2021

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.
- Persistent storage can be used to save configuration and audit files.
- It runs on most 64-xxxm.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. Hence, 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 Dockerfile does not work for every customer site. The Dockerfile 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.

Note: 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 is 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 8.1.0 and hotfixes 8.1.0_HF-001 and 8.1.0_HF-002, it is recommended that you create the following Docker images:

- TIBCO MFT Platform Server for UNIX 8.1.0
- TIBCO MFT Platform Server for UNIX 8.1.0 with Hotfix 8.1.0_HF-001
- TIBCO MFT Platform Server for UNIX 8.1.0 with Hotfix 8.1.0_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 Docker container, you must follow these steps that are a part of the process. Each step in the process 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

> **Note:** 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: Create Persistent Storage to Save Configuration and Audit Files.

Step 8: Test the Docker image.

Step 9: 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 is 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.

> **Note: Note:** Sample Docker files are distributed in the following folder: *`PSU Install`*`/cloud`

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 as required.
- Update the `SSLAuth.cfg` files as required.
- Update the SSL parameters for the client and the server in the `config.txt` file. The SSL parameters are defined as described in the table below:

Parameter	Defined Section
ClientVerification	Server
CertificateFileName	Server and Client
PrivateKeyFileName	Server and Client
PrivateKeyPwdFileName	Server and Client
TrustedAuthorityFileName	Server and Client

If you want the ability to update these files without rebuilding the Docker images, then you should consider saving these files to persistent storage. 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.

> **Note:** The values for these parameters should be used in Step 7: Creating Persistent Storage to Save Configuration and Audit Files.

Parameter	Description
LogEventFileName	Defines where the Log.txt files are saved. Each completed file transfer is logged in the Log.txt file. **Note:** 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. **Note:** The permissions for the PQFDirectory 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. **Note:** The permissions for the TransnumFileName and the directory where this file is located must be the same as the permissions in the $CFROOT directory.
ConfigDirectory	Defines the directory in persistent storage where configuration files are located. **Note:** The permissions for the ConfigDirectory must be the same as the permissions in the $CFROOT/config directory.
LogMessageFileName	Defines where message files are saved in persistent storage.

Parameter	Description
	Note: The permissions for the LogMessageFileName directory must be the same as the permissions in the `$CFROOT/log` directory.
`LogAdminFileName`	Defines where admin message files are saved in persistent storage. **Note:** The permissions for the `LogAdminFileName` directory must be the same as the permissions in the `$CFROOT/log` directory.

Note: Unless running in HA mode, each Platform Server container should point to different persistent storage directories. 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 get corrupted.

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

The `readme` file, which 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

1. 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 execute any commands when the Docker container stops.
   - The `config.txt` is parsed to extract permanent `config/audit` files and directories that should be saved in persistent storage.
   - Creates the following PSU environment variables: `CFROOT`, `PATH`, and `LD_LIBRARY_PATH`.
   - At startup, verifies that the persistent storage has the necessary files and directories. If not, it creates the necessary files and directories. If the config directory does not exist, it copies the config files to persistent storage.
   - Starts the necessary `CyberMgr` and `CyberResp` services. The following `CyberResp` services can be started:
     a. IPV4, IPV4 with TLS, IPV4 with TLS Tunnel
     b. 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

1. Configure the Docker file based on your requirements.

   The Docker file performs the following functions:

   - Installs the Unix 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 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 Dockerfile. The command differs slightly depending on whether you are installing as a root user or as a non-root user.

Procedure

1. 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=/PSU810 --build-arg CFUSER=root -t library/psu8.1.0 ./`
Non-root User	`docker build -f Dockerfile.psu --build-arg CFROOT=/PSU810NonRoot --build-arg CFUSER=mftuser -t library/psu8.1.0nonroot ./`

Step 7: Create Persistent Storage to Save Configuration and Audit Files

Create the persistent storage directories defined in the `config.txt` file in prior steps. Make sure that the rights allow necessary access to these files:

Parameter	Permission
`CertificateFileName`	Transfer users require read access.
`PrivateKeyFileName`	Transfer users require read access.
`PrivateKeyPwdFileName`	Transfer users require read access.
`TrustedAuthorityFileName`	Transfer users require read access.
`PQFDirectory`	Transfer users require read and write access.
`TransnumFileName`	Users executing `CyberMgr` require read and write access.
`LogEventFileName`	Users executing `CyberMgr` require read and write access.
`ConfigDirectory`	Users who update the `cfnode`, `cfprofile`, or `cfrprofiles` configuration files require read and write access. Transfer users require read permissions to configure files in the `ConfigDirectory`.

Step 8: 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.

Note: 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

1. 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/psu8.1.0`
Non-root User	`docker run -it --rm -p 46464:46464 -p 56565:56565 -p 58585:58585 -u mftuser:mftuser -v /persiststorage:/psupersist:z library/psu8.1.0`

User	RUN Command Sample in HA Configuration Setup
Root User	`docker run -it --rm -p 46678:46678 -p 46464:46464 -p 56565:56565 -p 58585:58585 -v /persiststorage:/psupersist:z library/psu8.1.0`
Non-root User	`docker run -it --rm -p 46678:46678 -p 46464:46464 -p 56565:56565 -p 58585:58585 -u mftuser:mftuser -v /persiststorage:/psupersist:z library/psu8.1.0`

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 `docker file`.
-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

Parameter	Descriptions
	directory defined in the `startall.sh` script.
-library/psu8.1.0	Defines the name of the Docker image.

Note: 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 9: 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.

TIBCO Documentation and Support Services

For information about this product, you can read the documentation, contact TIBCO Support, and join TIBCO Community.

How to Access TIBCO Documentation

Documentation for TIBCO products is available on the TIBCO Product Documentation website, mainly in HTML and PDF formats.

The TIBCO Product Documentation website is updated frequently and is more current than any other documentation included with the product.

Product-Specific Documentation

The following documentation TIBCO® Managed File Transfer Platform Server for UNIX is available on the TIBCO® Managed File Transfer Platform Server for UNIX Product Documentation page.

- *TIBCO® Managed File Transfer Platform Server for UNIX Release Notes*
- *TIBCO® Managed File Transfer Platform Server for UNIX Managed File Transfer Overview*
- *TIBCO® Managed File Transfer Platform Server for UNIX Installation*
- *TIBCO® Managed File Transfer Platform Server for UNIX User's Guide*
- *TIBCO® Managed File Transfer Platform Server for UNIX Security Guide*
- *TIBCO® Managed File Transfer Platform Server for UNIX Docker Container Deployment*

How to Contact TIBCO Support

Get an overview of TIBCO Support. You can contact TIBCO Support in the following ways:

- For accessing the Support Knowledge Base and getting personalized content about products you are interested in, visit the TIBCO Support website.
- For creating a Support case, you must have a valid maintenance or support contract with TIBCO. You also need a user name and password to log in to TIBCO Support website. If you do not have a user name, you can request one by clicking **Register** on the website.

How to Join TIBCO Community

TIBCO Community is the official channel for TIBCO customers, partners, and employee subject matter experts to share and access their collective experience. TIBCO Community offers access to Q&A forums, product wikis, and best practices. It also offers access to extensions, adapters, solution accelerators, and tools that extend and enable customers to gain full value from TIBCO products. In addition, users can submit and vote on feature requests from within the TIBCO Ideas Portal. For a free registration, go to TIBCO Community.

Legal and Third-Party Notices

This software may be available on multiple operating systems. However, not all operating system platforms for a specific software version are released at the same time. See the readme file for the availability of this software version on a specific operating system platform.

THIS DOCUMENT IS PROVIDED “AS IS” WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT.

THIS DOCUMENT COULD INCLUDE TECHNICAL INACCURACIES OR TYPOGRAPHICAL ERRORS. CHANGES ARE PERIODICALLY ADDED TO THE INFORMATION HEREIN; THESE CHANGES WILL BE INCORPORATED IN NEW EDITIONS OF THIS DOCUMENT. TIBCO SOFTWARE INC. MAY MAKE IMPROVEMENTS AND/OR CHANGES IN THE PRODUCT(S) AND/OR THE PROGRAM(S) DESCRIBED IN THIS DOCUMENT AT ANY TIME.

THE CONTENTS OF THIS DOCUMENT MAY BE MODIFIED AND/OR QUALIFIED, DIRECTLY OR INDIRECTLY, BY OTHER DOCUMENTATION WHICH ACCOMPANIES THIS SOFTWARE, INCLUDING BUT NOT LIMITED TO ANY RELEASE NOTES AND "READ ME" FILES.

This and other products of TIBCO Software Inc. may be covered by registered patents. Please refer to TIBCO's Virtual Patent Marking document (https://www.tibco.com/patents) for details.