ALM OCTANE DOCKER

Bauer, Idan

MICROFOCUS

Intro

ALM Octane is a Web-based platform that provides Agile project management, as well as application lifecycle management abilities. ALM Octane helps your teams collaborate more easily, manage the software delivery pipeline, fully understand the impact of changes, and track the progress and quality of their development.

IMPORTANT: ALM Octane on Docker is provided for demo purposes only. It does not support production environments.

In sections of this guide we refer to the main ALM Octane Installation Guide for further details. The guide is located here: https://octane-help.saas.hpe.com/en/latest/PDFs/ALM_Octane_Installation_Guide.pdf

System Requirements

ALM Octane on Docker is available as a Linux image only, and requires a compatible Linux host. The following table describes the minimum requirements for setting up ALM Octane on Docker:

Memory (RAM)	The full ALM Octane solution (ALM Octane + databases) requires at least 8 GB of RAM. If you're running database services on separate machines, 4 GB is sufficient for ALM Octane alone.
Free Disk Space	50GB minimum. If you plan to run the environment for a long period of time, 200GB is recommended to avoid running out of disk space.
Operating System	Any Linux operating system that supports Docker 1.17 and higher.
Docker software	Docker 1.17 and above. For details, see the Docker website. Alternatively, to use a package manager, see: https://docs.docker.com/engine/installation.

Notes:

- To avoid a disk flooding security risk, you should move your docker storage to a different partition. For details, see http://www.projectatomic.io/docs/docker-storage-recommendation.
- Software service packs and updates are supported, provided they are compatible with ALM Octane's general availability release.

Quick Installation

ALM Octane uses **Docker Compose** to quickly install ALM Octane and its required services with minimal effort. For details, see: https://docs.docker.com/compose/install/

Note: Deploying ALM Octane on Docker will create an ALM Octane installation without SSL configuration and is not meant to work under load. You should never expose this instance to the internet. If you need a more advanced configuration, see **Manual Installation** below.

Docker Compose service names

When using docker-compose, three services will be created. The service names are constructed from the name of the folder + name of service + instance. For example, if you run docker-compose from a folder named 'octane', the octane service will be named: octane_octane_1

The three services are:

- octane
- octane_es
- octane_oracle

When a service name is requested throughout the guide, make sure to use the full service name.

Install Octane with docker-compose

1. Create a folder anywhere on the server.
2. Copy the files included in the installation package to the new folder:
    - docker-compose.yml
    - octane.env
    - cleanup.sh
3. Edit octane.env and set the following parameters:

Parameter	Description
ADMIN_PASSWORD	The initial password Octane will use for the first user. The password must be at least 8 characters long, and contain at least one of these groups: • Capital letters • Lower case letters • Numbers or symbols
APP_URL	The url you will use to access ALM Octane. For example: http://myapp.mydomain.com:8080

4. Start Octane deployment with the following command from the folder:
   `docker-compose up –d`

Verify installation

Installation should take around 3 minutes, depending on your server's performance.

To follow the installation progress, run:

```
docker logs -f <name of your octane container>
```

When the installation is complete, you will receive a message: 'Creation Successful'.

Press Ctrl+c to stop displaying the log.

Log in to ALM Octane

1. Access your ALM Octane instance by entering its URL in a browser:
   `http://<your octane server>:<your octane port>/ui`
2. Login with sa@nga and the admin password you set in step 3.

Restarting your server

You can restart your server. From the directory in which docker-compose.yml resides, run the following commands:

- Stop the server: `docker-compose stop <octane-service-name>`
- Start the server: `docker-compose start <octane-service-name>`

Uninstalling ALM Octane

If you want to remove ALM Octane from your server, use the cleanup.sh command provided in the installation zip.

The script performs the following:

1. Runs the command **`docker-compose down`** that stops and removes the docker instances created.
2. Cleans up the folders created on the host during installation to store the server files.

If you want to retain your data after removing the server, run **`docker-compose down`** from the installation folder.

Note: if you modified any of the folders' locations, you will need to modify cleanup.sh as well to make it work.

Manual Installation

If you want to install ALM Octane on your existing environment, you can install ALM Octane on Docker manually. This will allow you to use existing database existing database servers, or provide support for SSL or LDAP, for example.

When installed manually, ALM Octane on Docker supports all options detailed in the ALM Octane Installation Guide.

Database requirements

ALM Octane on Docker can run with user-supplied database servers (Oracle / SQL / ES). If you plan to provide your own database servers, refer to the ALM Octane System Requirements.

ALM Octane on Docker does not support upgrading from a previous version. You can only create a new schema. The FILL_EXISTING site action is not supported.

Database permissions and prerequisites

Refer to the ALM Octane Installation Guide for the required database permissions.

Before you install ALM Octane manually, you need to have database servers up and running on your system. The database servers need to be accessible to the ALM Octane deployment.

Create the setup environment

If ALM Octane detects that you had previously set up an environment to run ALM Octane, the installer will use the settings that you previously provided. To allow this situation, we will configure the settings files:

1. Create a folder on the server. For example: /opt/octane/conf
   Note: You can decide where to place this folder by modifying the Docker run volume option.
2. Copy the enclosed setup.xml and octane.yml files to the new folder.
3. Edit the settings in the files according to the ALM Octane Installation Guide.

Run Octane Docker

Run the following command to pull and start Docker:

```
docker run -d -p <8080>:8080 -v </opt/octane/conf>:/opt/octane/conf -v
</opt/octane/log>:/opt/octane/log -v </opt/octane/repo>:/opt/octane/repo -
-name octane  lifecyclemanagement/octane
```

The following is an example of an actual command:

```
docker run -d -p 8080:8080 -v /opt/octane/conf:/opt/octane/conf -v
/opt/octane/log:/opt/octane/log -v /opt/octane/repo:/opt/octane/
lifecyclemanagement/octane
```

Run ALM Octane on Docker with SSL support

To run ALM Octane on Docker with SSL support, you need to first prepare an SSL certificate, and then run Docker:

Prepare SSL certificate

You need to create a certificate in the JKS format and place it in the same configuration folder used to store setup.xml and octane.yml.

You need to add two additional environment variables to the Docker run statement. One tells ALM Octane that it needs to apply your file. The other variable contains the password used to open the private key.

Run Docker

```
docker run -d -p <8443>:8443 -e "SSL_KEYSTORE=<name of file>" -e "SSL_
PASSWORD=<keystore password>" -v </opt/octane/conf>:/opt/octane/conf –v
</opt/octane/log>:/opt/octane/log -v </opt/octane/repo>:/opt/octane/repo -
-name octane lifecyclemanagement/octane
```

Post-installation steps

Follow the steps in the Manual Installation section to verify your installation, connect to ALM Octane, restart the server, and uninstall, if necessary.

Advanced topics

The following sections are not required for a standard installation of ALM Octane on Docker. We provide them in case you run into issues and need additional help.

Docker images versions

As Docker images are used for trying out new features, we provide three channels for ALM Octane on Docker. By default, Docker will pull the "latest" – or no tag – release which is always the latest stable release. We also provide more bleeding-edge releases. There are three release channels:

- **Stable** – Found under the "latest" tag (or no tag), it provides the latest stable release. This version was thoroughly tested and is what we use in production. You should use this channel if you're not sure what to use.
- **Beta** – a version that was well tested but may still contain several known issues. This version is a good compromise between getting the latest features and getting a relatively stable version.
- **Alpha** – a version that is released as soon as basic integration tests are done. This version may contain serious issues and is not considered stable but it generally works. Only use when you need the absolute latest.

To pull a different channel, use Docker tags. Use the following commands to pull the different versions:

- **Stable:** lifecyclemanagement/octane
- **Latest beta:** lifecyclemanagement/octane:beta
- **Latest alpha:** lifecyclemanagement/octane:alpha

We also provide tags for each version we release. For a complete list of all tags available, see:

https://hub.docker.com/r/lifecyclemanagement/octane/tags/

Manual Docker database servers

If you need to run a manual installation but would still like to use the Docker-based database servers, these are the database server images used by the ALM Octane all-in-one solution:

Oracle

For Oracle, we use the sath89 image. It allows us to run Oracle XE, and store the data on the host, in case we need to access it. The image can be found here:

https://hub.docker.com/r/sath89/oracle-xe-11g/

To run this image, use this command:

```
docker run -d -v /opt/oracledata:/u01/app/oracle -p 1521:1521 --name
octane_oracle sath89/oracle-xe-11g
```

Note: since we are not running with docker compose there's no dedicated network for running docker and we are using the hosts' network. For that we need to expose the port (1521) and access the server with localhost:1521 instead of the container's label.

Elastic Search

For Elastic search we use the official Elastic search 2.4 image. The image can be found here:

https://hub.docker.com/_/elasticsearch/

To run Elastic search use the following command:

```
docker run -d -e "ES_HEAP_SIZE=1G" -v
/opt/esdata:/usr/share/elasticsearch/data -p 9200:9200 -p 9300:9300 --name
octane_es  elasticsearch:2.4
```

Note: Since we are not running with Docker Compose, there is no dedicated network for running Docker and we use the hosts' network. For that we need to expose the port (9300) and access the server with localhost:9300 instead of the container's label.

Upgrading Octane Docker

It is possible to upgrade your Octane Docker instance. To upgrade docker, perform the following two steps:

Prepare the environment

Open setup.xml in your conf folder (default: /opt/octane/conf) and change the site upgrade value to UPGRADE.

Update binaries

Docker type	**Run commands**
docker-compose	1. Run `docker-compose pull <octane_service_name>` 2. Run `docker-compose up -d --no-deps <octane_service_name>` **Note:** if you used a non-standard octane docker image you will have to make sure the correct image appears before you pull.
Docker	1. Pull the new image: docker pull <image name> e.g. lifecyclemanagement/octane 2. Remove the existing container: `docker rm -f <octane container name>`

Recreate the container as explained above.

Continue with regular ALM Octane upgrade

For details, see ALM Octane Installation Guide.