DOCKER IMAGE DEPLOYMENT FOR CAN

Version 5.0

AUGUST 10, 2020

AVANSEUS TECHNOLOGIES PVT. LTD.

REVISION HISTORY

Version	Date	Change description	Created by	Updated by	Reviewed by
V 5.0	Aug, 2020	-	Naveen	Sandeep	Chiranjib

TABLE OF CONTENTS

1. FOCUS 3
2. DEPLOYMENT OF IMAGES IN CONTAINER 3

1. Focus

This document focuses on deploying an existing image of CAN application & all other supporting applications. This document is mainly for the Deployment team (Avanseus support/System integrator) who deploys the Docker image. The Docker images needs to be deployed for CAN applications to run are as follows.

1. MongoDB
2. LDAP image (With OpenDS and one default user)
3. VBI python module
4. CAN application (Master node)
5. CAN application (Slave node)

2. Deployment of Images in Container

1. MongoDB

Run the MongoDB using below command:

```
docker run -d -p <physical_port>:27017 -v <path_on_physical_server>:/data/db mongo:3.4.6

Eg: docker run -d -p 37017:27017 -v /Users/avanseustechnologies/DockerTest/DB/data:/data/db mongo:3.4.6
```

- Access Mongo console via physical port assigned and create an admin user with necessary roles with command “mongo localhost:37017/admin”. Example is shown below:

```
MongoDB shell version v3.4.6
connecting to: mongodb://127.0.0.1:27017
MongoDB server version: 3.4.6
Welcome to the MongoDB shell.
For interactive help, type "help".
For more comprehensive documentation, see
        http://docs.mongodb.org/
Questions? Try the support group
        http://groups.google.com/group/mongodb-user
Server has startup warnings:
2020-04-16T05:47:05.352+0000 I CONTROL  [initandlisten]
2020-04-16T05:47:05.352+0000 I CONTROL  [initandlisten] ** WARNING: Access control is not enabled for the database.
2020-04-16T05:47:05.352+0000 I CONTROL  [initandlisten] **          Read and write access to data and configuration is unrestricted.
2020-04-16T05:47:05.352+0000 I CONTROL  [initandlisten]
> use candb
switched to db candb
> use admin
switched to db admin
> db.createUser({user: "admin", pwd: "Avanseus$0", roles: ["userAdminAnyDatabase", "readWriteAnyDatabase", "dbAdminAnyDatabase", "clusterAdmin"]})
Successfully added user: {
        "user" : "admin",
        "roles" : [
                "userAdminAnyDatabase",
                "readWriteAnyDatabase",
                "dbAdminAnyDatabase",
                "clusterAdmin"
        ]
}
> exit
bye
```

- Stop the MongoDB container using "sudo docker stop <container_ID>".
- Run a new MongoDB container using below command:

```
docker run -d -p <physical_port>:27017 -v <path_on_physical_server>:/data/db mongo:3.4.6 --auth

Eg: docker run -d -p 37017:27017 -v /Users/avanseustechnologies/DockerTest/DB/data:/data/db mongo:3.4.6 --auth
```

- Access Mongo console via physical port assigned with admin user credentials to create DB admin user and schemas. An example is shown below:

```
Avanseuss-MacBook-Pro:~ avanseustechnologies$ mongo localhost:37017/admin -u admin -p 'Avanseus$0'
MongoDB shell version v4.0.3
connecting to: mongodb://localhost:37017/admin
WARNING: No implicit session: Logical Sessions are only supported on server versions 3.6 and greater.
Implicit session: dummy session
MongoDB server version: 3.4.6
WARNING: shell and server versions do not match
> use candb
switched to db candb
> db.createUser({user:"canadmin", pwd:"Avanseus$0", roles: ["userAdmin", "dbAdmin", "readWrite"]})
Successfully added user: {
        "user" : "canadmin",
        "roles" : [
                "userAdmin",
                "dbAdmin",
                "readWrite"
        ]
}
> exit
bye
```

Once DB is up & running, MongoDB must be updated with necessary collections needed for CAN application to run. Few prerequisites are:

- Config table must be correctly configured.
- Network IP pattern in Config must be set to 127.0.0.
- PredictionNode table must be updated with necessary node information & their ports. This port information is needed beforehand to run the CAN image.

Once all this is done, the docker container instance can be stopped and started using:
sudo docker stop <container_ID>
sudo docker start <container_ID>

To list the containers, use the command “sudo docker ps -a”. This shows all information on running and stopped containers.

To view container logs, use the command “sudo docker logs <container_ID>”. This is to verify if the application is up and running correctly or not.

2. LDAP

The command to run LDAP image in a container is as follows:

```
sudo docker run -d -p <physical_port>:1389 -p <physical_admin_port>:4444 ldapimage:1

Eg: sudo docker run -d -p 1389:1389 -p 4444:4444 ldapimage:1
```

To stop, start and view logs of the container, commands are already provided in Section-1.

3. VBI Python Module

The command to run VBI python module image in a container is as follows:

```
sudo docker run -d -p <physical_port>:12001 pyvbi:v1

Eg: sudo docker run -d -p 12001:12001 pyvbi:v1
```

To stop, start and view logs of the container, commands are already provided in Section-1.

4. CAN Application (MASTER)

Below is the command to run “canapp:1” image in a container. But before running, please fill up “catalina.properties”, “config.properties” & “setenv.sh” with necessary information. This will change in every new environment.

```
docker run -d -p <can_tomcat_port>:2000 -p <can_ajp_port>:2002 -p <cas_tomcat_port>:2003 -p <cas_ajp_port>:2005 -p <can_master_port>:<can_master_port> -m=4g -v <physical_log_path_on_physical_server>:/data/workspace/logs/ -v <path_to_config_file_on_physical_server>:/data/workspace/tomcatCAS/config.properties -v <path_to_config_file_on_physical_server_for_master>:/data/workspace/tomcatCAN/config.properties -v <path_to_catalina_properties_on_physical_server>:/data/workspace/tomcatCAN/conf/catalina.properties -v <path_to_catalina_properties_on_physical_server>:/data/workspace/tomcatCAS/conf/catalina.properties -v <path_to_setenv_script_on_physical_server>:/data/workspace/tomcatCAN/bin/setenv.sh canapp:1

Eg:
docker run -d -p 2000:2000 -p 2002:2002 -p 2003:2003 -p 2005:2005 -p 3000:3000 -m=4g -v /Users/avanseustechnologies/DockerTest/tomcats/data:/data/workspace/logs/ -v /Users/avanseustechnologies/DockerTest/tomcats/config.properties:/data/workspace/tomcatCAS/config.properties -v /Users/avanseustechnologies/DockerTest/tomcats/config.properties:/data/workspace/tomcatCAN/config.properties -v /Users/avanseustechnologies/DockerTest/tomcats/catalina.properties:/data/workspace/tomcatCAN/conf/catalina.properties -v /Users/avanseustechnologies/DockerTest/tomcats/catalina.properties:/data/workspace/tomcatCAS/conf/catalina.properties -v /Users/avanseustechnologies/DockerTest/tomcats/setenv.sh:/data/workspace/tomcatCAN/bin/setenv.sh canapp:1
```

<can_master_port> is the port where master node or “nodeA” keeps listening on the socket. This must be same as the one given in PredictionNode collection in DB.

To stop, start and view logs of the container, commands are already provided in Section-1.

5. CAN application (SLAVE)

The command to run “canslaveapp:1” image in a container is given below. Before running the command, fill up “catalina.properties”, “config.properties” & “setenv.sh” with the necessary information. This details will change in every new environment.

docker run -d -p **<can_physical_port>**:8080 -p **<can_slave_port>**:**<can_slave_port>** -m=2g -v **<physical_log_path_on_physical_server>**:/data/workspace/logs/ -v **<path_to_config_file_on_physical_server_for_slave_X>**:/data/workspace/tomcatCAN/config.properties -v **<path_to_catalina_properties_on_physical_server_for_slave_X>**:/data/workspace/tomcatCAN/conf/catalina.properties -v **<path_to_setenv_script_on_physical_server_for_slave_X>**:/data/workspace/tomcatCAN/bin/setenv.sh canslaveapp:1

Eg:

docker run -d -p 2006:8080 -p 3001:3001 -m=1g -v /Users/avanseustechnologies/DockerTest/slaves/data:/data/workspace/logs/ -v /Users/avanseustechnologies/DockerTest/slaves/config.properties:/data/workspace/tomcatCAN/config.properties -v /Users/avanseustechnologies/DockerTest/slaves/nodeX/catalina.properties:/data/workspace/tomcatCAN/conf/catalina.properties -v /Users/avanseustechnologies/DockerTest/slaves/nodeX/setenv.sh:/data/workspace/tomcatCAN/bin/setenv.sh canslaveapp:1

The above command can be used to generate many more slave docker containers based on requirement. The image remains the same. The only change is ports, path of config.properties & log path.

<can_slave_port> is the port where slave node or “nodeX” keeps listening on the socket. This must be the same as the one given PredictionNode collection in DB.

To stop, start and view logs of the container, commands are already provided in Section-1.