Building & Deploying SDC on Linux for Development

This guide outlines how to build and run SDC locally on linux. This can also be achieved on a linux virtual machine.

Table of Contents

- Prerequisites
- Build SDC
- Set up 'data' directory
- Run SDC, SDC simulator and Sanity Tests
- Setup if not building SDC locally first
- Setting up the Webseal-Simulator
- Configure Docker Engine
- Deploying local dockers to a VM

Prerequisites

1. Ubuntu 14.04, 16.04 or 18.04 (CentOS may well also work but it's not confirmed yet)
2. 6 cores and 8G of ram
3. Java 11 and Apache Maven (min. version 3.6.0)
4. Docker (min. version 20.10.x)

Build SDC

This guide will assume you have cloned the SDC repository to **~/workspace/ONAP/sdc**

```
$ cd ~/workspace/ONAP/sdc
$ mvn clean install -P docker
Note: in first run you cannot skip tests. All dependencies must be created and installed in the local maven
repository.
Note: to speed up build process, add "skip-test" parameters (-DskipTests=true -DskipUICleanup=true -Djacoco.
skip=true -DskipPMD -Dmaven.test.skip=true -Dcheckstyle.skip)
```

Run '*docker images*' to verify the existence of recently built ***onap/sdc-**** images

Set up 'data' directory

```
$ sudo mkdir /data && sudo chmod 777 /data
$ mkdir /data/environments
$ cp <sdc-project-folder>/sdc-os-chef/environments/Template.json /data/environments/AUTO.json
$ IP=`ip route get 8.8.8.8 | awk '/src/{ print $7 }'`
$ echo ${IP}
192.168.1.13    # this will probably be different for you
$ sed -i "s/yyy/${IP}/g" /data/environments/AUTO.json
$ sed -i 's/xxx/AUTO/g' /data/environments/AUTO.json
```

Run SDC, SDC simulator and Sanity Tests

> **Dynamic IP**
>
> Remember to update AUTO.json file with your current IP! Most of us have dynamic IP! It could be changed after computer restart. See instruction above.

First cd into the main SDC repository

```
  $ cd ~/workspace/ONAP/sdc
```

Build the SDC docker images

```
  $ mvn clean install -P docker
```

```
If you have built the image previously, you can use the fast-build profile
  $ mvn clean install -P docker, fast-build
```

Run SDC container

```
$ mvn install -P start-sdc
```

You can now open the SDC UI locally

```
$ firefox http://localhost:8285/login
```

For more info view sdc/README.md

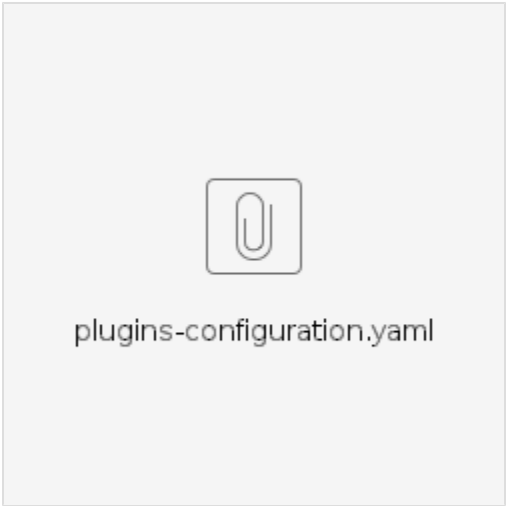
Setup if not building SDC locally first

Note: From here on this guide may be slightly outdated as the SDC git repository may have diverged from the instructions below.

1. in order for the dockers to start correctly on the VM the following folders need to be created on the VM.
2. you need the /opt folder:
    a. in it create /opt/config/
    b. create files in this /opt/config/ folder as follows:

```
$ echo AUTO > /opt/config/env_name.txt
$ echo nexus3.onap.org:10001 > /opt/config/nexus_docker_repo.txt
$ echo docker > /opt/config/nexus_username.txt
$ echo docker > /opt/config/nexus_password.txt
```

3. you need a /data folder
    a. in it create /data/environments/ (**Note:** Do not forget to configure your (BE/FE/Kibana/Elastic search) machine IP address in the below file.If they are all planned to run in the same machine, configure the same IP address) copy the file from here: https://git.onap.org/sdc/tree/sdc-os-chef/environments/Template.json Note: this file may change from time to time so keep it updated with the changes done.
    b. rename it to AUTO.json
    c. change all the yyy to ip of the vm you are running on.
    d. change all the xxx to 'AUTO'
    e. For plugins configuration copy the file from https://git.onap.org/sdc/tree/sdc-os-chef/environments/plugins-configuration.yaml and update the plugins urls
    f. Below is the example of plugins-configuration.yaml



4. now create a /data/scripts/ folder
5. in folder /data/scripts/ place the following scripts from the sdc repo sdc repo
    - docker_run.sh
    - docker_login.sh
    - docker_health.sh
6. provide execution permissions to the scripts

```
chmod 777 /data/scripts/*
```

7. as root user, launch the following script to test the setup is working (it will pull dockers from the LF repo and start them)

```
/data/scripts/docker_run.sh -r 1.1-STAGING-latest
```

on script completion, you will see a successful health check. Depending on you VM performance, some containers may take times to be ready and up : sdc-cs takes about 2/3 mn, sdc-BE takes about 7/8 mn.

8. SDC API are available : http://yourIP:8080 for "internal API"
9. in order to access SDC you will need to enable SDC Simulator

Setting up the Webseal-Simulator

In order to set up the Webseal-Simulator in your local environment, you can use the following guide: SDC Simulator

Configure Docker Engine

1. in order to allow building dockers from dev machine to the VM you will need to enable tcp communication to the docker engine.
   use this link to enable tcp communication to the docker https://docs.docker.com/engine/reference/commandline/dockerd/
2. once enabled you can execute netstat -nap to check the docker d is listening to 2375.

Deploying local dockers to a VM

In order to check our code from our local environment, we'll compile our code locally and upload our local images to the vagrant

This is how to do so:

1. Add **DOCKER_HOST** to environment variables with the value: **tcp://<ip of the vm where >:<port>**
2. Run **mvn clean install** and build the whole project.
3. After the build is finished you'll need to run the **sdc-os-chef** project with profile docker in order to build docker from your local code and upload them to your local vagrant.
   Run the following command in order to do so: **(Need to check the docker repository credentials)**

```
mvn clean install -pl sdc-os-chef -P docker
```

4. After this process is done go to your vagrant and run **docker images** to check that your local images are on the machine
5. In order to deploy your local images without pulling the latest images from the nexus you'll need to run the docker_run script with the local flag like so:

```
/data/scripts/docker_run.sh -l
```