# developer.skatelescope.org Documentation

*Release 0.1.0-beta*

**Marco Bartolini**

**Jan 18, 2022**

# CONTENTS:

This project builds a generic InfluxDB node based on dynamic inventory generated by https://gitlab.com/ska-telescope/sdi/heat-cluster, which in turn relies on ansible collections from https://gitlab.com/ska-telescope/sdi/systems-common-roles.

CHAPTER

ONE

# README

Deploy a single node InfluxDB with Ansible.

## 1.1 Summary

This repo builds a generic InfluxDB node based on the dynamic inventory generated by https://gitlab.com/ska-telescope/sdi/heat-cluster, which in turn relies on ansible collections from https://gitlab.com/ska-telescope/sdi/systems-common-roles

Checkout deploy-influxdb (this repo) and pull the dependent collections with:

```
git clone git@gitlab.com:ska-telescope/sdi/deploy-influxdb.git
cd deploy-influxdb
make install
```

### 1.1.1 Ansible vars for the cluster

In order to define the architecture of the cluster, one needs to describe, at a minimum, the *dbnode_influxdb*.

In influxdb_cluster_vars.yml, it is described a cluster containing 1 *dbnode_influxdb* (there is always just 1).

### 1.1.2 Molecule testing

Before running the build stage to deploy the InfluxDB node on the Openstack environment we encourage testing the ansible collection roles by using molecule.

The first step is to install the required python packages (we strongly recommend that a virtual environment is set). This can be acomplished with:

```
virtualenv venv && . venv/bin/activate
pip3 install -r requirements.txt
```

Molecule tests can then be run by doing:

```
make test
```

These tests do not require access to an openstack platform, they are fully local, but they need a lot of resources, at least 8 Gb of free available memory are recommended. The tests are also time consuming taking several minutes to complete.

For debugging purposes one can run the molecule tests without destroying the test container:

```
make molecule
```

With this option, resuming the tests after fixing any bugs that are found will be faster, but occasionally one may need to destroy the test container by doing:

```
make destroy
```

### 1.1.3 Setting your OpenStack environment

You need to use an adequate Openstack keypair name. This maybe be your usual personal keypair (which is most likely also your username) or a keypair defined just for this purpose. You also need your own RC file for the Openstack platform that you will be using. This file is available in shell or yaml formats. You should be able to retrieve it from the dashboards on the Openstack platform GUI.

If you choose the Openstack RC shell file, you should then source it (your password may be asked) and test your connection to the Openstack API:

```
. openrc.sh
openstack server list
```

Alternatively, if you prefer the yaml file (`cloud.yaml`) you should place it in the `~/.config/openstack/` folder. You may need to add your password to the auth section in that file. Do not forget to test the connection.

### 1.1.4 Referencing custom configuration

In order not to have to specify the `CLUSTER_KEYPAIR`, `PRIVATE_VARS` and `INVENTORY_FILE` variables every time a make command is issued, we recommend that you create a *PrivateRules.mak* file in the deploy-influxdb root directory specifying these three variables (**these need to be changed accordingly to each specific purpose**):

```
CLUSTER_KEYPAIR=your-openstack-key-name
PRIVATE_VARS=./influxdb_cluster_vars.yml
INVENTORY_FILE=./inventory_dev_cluster
```

### 1.1.5 Running the build

The build process is broken into two phases:

- create the VM and do the common install steps
- build influxdb

This can be run simply with:

```
make build
```

Or, in case you didn't define the *PrivateRules.mak* with:

```
make build CLUSTER_KEYPAIR=your-openstack-key-name PRIVATE_VARS=./influxdb_cluster_
↪vars.yml INVENTORY_FILE=./inventory_dev_cluster
```

You can also break the `make build` command into the two steps comprising it:

```
# create the VMs and do the common install
make build_nodes
# build InfluxDB
make build_influxdb
```

### 1.1.6 Destroying the nodes

**Caution: The following command deletes everything that was installed and destroys the cluster specified by the *PRIVATE_VARS* variable.**

In case you want to delete the whole stack issue the `make clean` command.

### 1.1.7 Help

Run `make` to get the help:

```
make targets:
mbuild                         Build nodes and influxdb
mbuild_common                  apply the common roles
mbuild_docker                  apply the docker roles
mbuild_influxdb                Build influxdb
mbuild_nodes                   Build nodes based on heat-cluster
mbuild_vms                     Build vms only based on heat-cluster
mcheck_nodes                   Check nodes based on heat-cluster
mclean                         destroy the nodes - CAUTION THIS DELETES EVERYTHING!!!
mclean_nodes                   destroy the nodes - CAUTION THIS DELETES EVERYTHING!!!
mdestroy                       run molecule destroy - cleanup after make molecule
mhelp                          show this help.
minstall                       Install dependent ansible collections
mlint                          Lint check playbook
mmolecule                      run molecule tests but don't destroy container
mreinstall                     reinstall collections
mrtest                         run make $(STAGE) using gitlab-runner - default: test
mtest                          run molecule tests locally outside gitlab-runner
mvars                          Variables
mverify                        rerun molecule verify - must run make molecule first

make vars (+defaults):
mANSIBLE_USER                  ubuntu## ansible user for the playbooks
mCI_APPLICATION_TAG            $(shell git rev-parse --verify --short=8 HEAD)
mCI_BUILD_TOKEN                ""
mCI_ENVIRONMENT_SLUG           development
mCI_JOB_ID                     job$(shell tr -c -d
↪'0123456789abcdefghijklmnopqrstuvwxyz' </dev/urandom | dd bs=4 count=1 2>/dev/null;
↪echo)
mCI_PIPELINE_ID                pipeline$(shell tr -c -d
↪'0123456789abcdefghijklmnopqrstuvwxyz' </dev/urandom | dd bs=8 count=1 2>/dev/null;
↪echo)
mCLUSTER_KEYPAIR               piers-engage## key pair available on openstack to be␣
↪put in the created VMs
mCOLLECTIONS_PATHS             ./collections
mCONTAINERD                    true
mDEBUG                         false
mDOCKERFILE                    Dockerfile
mDOCKER_VOLUMES                /var/run/docker.sock:/var/run/docker.sock
```

(continues on next page)

(continued from previous page)

```
mEXECUTOR                      docker
mEXTRA_VARS ?=
mGITLAB_USER                   ""
mGITLAB_USER_EMAIL             "nobody@example.com"
mIGNORE_NVIDIA_FAIL            false
mINVENTORY_FILE                ./inventory_influxdb_cluster##inventory file to be␣
↪generated
mNVIDIA                        false
mPRIVATE_VARS                  ./influxdb_cluster_vars.yml##template variable for␣
↪heat-cluster
mREGISTRY_TOKEN                ""
mREPOSITORY_TOKEN              ""
mSTAGE                         test## Molecule stage name
mTAGS ?=
mV ?=
```