SOLR DOCKER CONTAINER

Anh Vu - 12/3/20

# PURPOSE

* How to set up a Solr Docker container
* Intended audience is developers

# REQUIREMENTS

* Docker container containing a solr server which runs as standalone or cloud mode. Cloud mode is required in some cases/scenarios such as running Kafka Connect service where a third party Solr Connector/Sink is used.
* The data volume for the Solr server is either on the Docker writable layer or on the host of the Docker container.
* Solr server contains our specific schema for input data.

# CONSIDERATION

## Docker-compose vs Dockerfile

Since we already have a docker-compose.yml in the venom-docker folder which builds all of our modules/components, we continue to use docker-compose.yml to create our Solr container. Dockerfile allows us to pre-populate data to the Solr container (TODO).

## Solr data location

Docker allows 3 ways to access external data: volume, bind mount, and tmpfs mount. “Volume” is the recommended/preferred way to persist data.

* Tmpfs = is used for temporary storage. Not applicable for our purposes.
* Volume = resides on the host or another container
* Bind mount = resides either on the host or the container

|  |
| --- |
|  |

*Figure 1 - Docker data volume*

|  |  |  |
| --- | --- | --- |
| **Type** | **Advantage** | **Disadvantage** |
| Data on container | Easy to set up, perfect for development quick start (i.e. a standalone container for developing/testing) | When the container is destroyed, the data is also gone.  When the container is stopped, you cannot access the data (might be ok for our purposes)  Not distributed/shared: multiple containers will have different data sets  Unscalable: Data size is limited by the container capacity  Dependent on the container Storage Driver: If you change the Driver, you cannot access old data anymore. |
| External data volume | Accessible: you can still access the data when container is destroyed/stopped  Scalable: the data can grow freely (ex. cloud storage)  Distributed: multiple containers can share the same volume  Easier to back up  Faster, better performance | Harder to set up  Need planning and maintenance (ex. backups) |

*Table 1 - Compare Volume and Bind Mount*

Using external volumes is the clear choice. For the sake of completeness, we show both setups.

# ACTIVITY

## TEST ENVIRONMENT

* Host: MacOS Catalina v10.15.3
* Docker v19.03.5

## REQUIRED FILES

* venom\_docs folder: This is the solr “core”/”collection” folder which contains configuration and data for our purposes.
* Schema.xml: This is our venom data structure, stored in gitlab venom project, solr branch, under venom/venom-docker/config/solr/venom\_docs/conf folder.
* Taxi\_top50-dsname.json: This is our test data, stored in gitlab, under venom/venom-docker/testdata/solr folder.

Since we want to apply our schema to our venom\_docs core/collection, we must maintain this folder as we update the schema.xml. The venom\_docs folder used in the following examples (and also kept in our Gitlab) also contains the Taxi data as well as the schema.xml.

## GOAL

We build a standalone Solr Docker Container. Set it to use our schema.xml. Then, load the test data.

There are several ways to build the container as well as setting up the data volume. Creating the container can be done with a Dockerfile or a docker-compose.yml. The following sections describe how it is done with docker-compose.yml. They also show how to configure the data location inside and outside of the container (see Table 1).

## EXAMPLE 1 - DOCKER-COMPOSE WITH DATA ON THE CONTAINER

Figure 2 shows the docker-compose.yml to build the desired container.

|  |
| --- |
| version: '3'  services:  solr:  image: solr:8.6  ports:  - "8983:8983"  volumes:  - data:/var/solr  command:  - solr-precreate  - venom\_docs  volumes:  data: |

*Figure 2 - docker-compose.yml (data on the container)*

* Version 3 is the docker-compose version used to build this container
* We want to download solr v8.6 for the container
* Map solr port 8983 to container port 8983
* Bind “data” volume to /var/solr/data/venom\_docs on the container. This is “data on the container” as explained in Table 1.

Solr server expects some configuration files in the data folder (see Figure 3). I have not found out exactly which files yet.

|  |
| --- |
|  |

*Figure 3 - Content of /var/solr/data of a working solr container*

The “venom\_docs” folder is the solr “core” which will contain all of our test data after ingestion. In standalone mode, such a repository is called “core”. In cloud mode, it is called “collections”. It is noteworthy to remember the difference.

|  |
| --- |
|  |

*Figure 4 - Content of /var/solr/data/venom\_docs*

## STEPS TO BUILD

From a terminal window, cd to your working directory where the docker-compose.yml is located, type “docker-compose up -d” to build and run the container. Point your browser to localhost:8983 to see the admin console of the solr service.

You can log-in to the container by:

docker container ls (this will show you the id of the container)

docker exec -it <id of the container> /bin/bash

There, you can see /var/solr/data/venom\_docs as in Figure 4.

We then can use the solr post utility to load our test data but we need to copy our test data into the container first.

Open another terminal window and type “docker cp ./Taxi\_top50-dsname.json <container id>:/tmp”. Go back to the /bin/bash terminal and type “post -c venom\_docs /tmp/Taxi\_top50-dsname.json”. This loads the content of the taxi data into venom\_docs core. Verify successful loading from the solr admin web page, click Query|Execute Query.

## EXAMPLE 2 - DOCKER-COMPOSE WITH DATA ON THE HOST (EXTERNAL DATA VOLUME)

Figure 5 shows the docker-compose.yml to build the desired container.

|  |
| --- |
| version: '3.2'  services:  solr:  image: solr:8.6  ports:  - "8983:8983"  volumes:  - /tmp/venom\_docs:/var/solr  command:  - solr-precreate  - venom\_docs  volumes:  data: |

*Figure 5 - docker-compose.yml (data outside of container)*

* We need at least version 3.2 of docker-compose to work.
* Bind the host /tmp/venom\_docs folder to /var/solr/ on the container as “data” volume. The final path on the container will be /var/solr/data/venom\_docs (see Figure 3). This is “External data volume” as explained in Table 1.

In this example, I keep my venom\_doc/ in my /tmp which is not suitable for production because /tmp will be wiped clean after my machine rebooted. Please keep it at a more permanent host in your production/development environment. This venom\_docs/ folder is in our Gitlab under solr-anh branch, venom/venom-docker/solr-only folder. Note that my /tmp is already shared with everyone. You need to share your data folder so solr can access it. You don’t need to change directory ownership as before (see Figure 6).

|  |
| --- |
|  |

*Figure 6 - /tmp/venom\_docs content on my machine (external volume to solr container)*

## 

## 

## STEPS TO BUILD

First we need to clean up everything from Example 1 above:

* docker stop <container id>
* docker rm <container id>
* docker images (this shows you the image id)
* docker rmi <image id>

Typing all the above commands every time is tedious. Figure 7 is a helper script to remove all images on your machine.

|  |
| --- |
| docker rm -vf $(docker ps -a -q)  docker rmi -f $(docker images -a -q) |

*Figure 7 - cleanup.sh*

* Download the venom\_docs/ folder from our gitlab, venom project. Copy it to a shared folder on your machine (ex. /tmp). This folder must be shared and accessible to the container.
* Edit the docker\_compose.yml in Figure 5 to bind your shared folder to solr volume.

Then, “docker-compose up -d” will build and run the solr container. You can load the sample data with the same commands in Example 1 above.

## EXAMPLE 3 - DOCKERFILE WITH DATA ON THE CONTAINER

|  |
| --- |
| # Single line Dockerfile  FROM solr:8.6 |

*Figure 8 - Dockerfile (data on the container)*

## STEPS TO BUILD

* Clean up all images from previous examples.
* cd to the folder containing the above Dockerfile, type “docker build -t mysolr:1.0 .” It tags the image as “mysolr”, version “1.0”.
* Type “docker run -p 8983:8983 -v "data:/var/solr" mysolr:1.0 solr-precreate venom\_docs” to run the container. This command runs the version you’ve just built, creates an empty core name “venom\_docs”. Any new data posted to this container will stay on the container.

## VARIATION

Other venom modules such as venom-postgres uses a Dockerfile to build a container, then include it inside the docker-compose.yml. This variation of the Dockerfile attempts to follow that pattern.

|  |
| --- |
| # Base solr image  FROM solr:8.6  # Copy our data to the container  COPY ./venom\_docs /var/solr/data |

*Figure 9 - Dockerfile (data on the container)*

If you build the above as “mysolr:1.0” then you can use it by changing the “image” in docker-compose.yml (see Figure 5) to “image: mysolr:1.0”.

## EXAMPLE 4 - DOCKERFILE WITH DATA ON THE HOST (EXTERNAL DATA VOLUME)

|  |
| --- |
| # Single line Dockerfile  FROM solr:8.6 |

*Figure 10 - Dockerfile (data outside of container)*

## STEPS TO BUILD

* Prepare your external data folder on your machine (host) like in Figure 6
* cd to the folder containing the above Dockerfile, type “docker build .”
* Type “docker run -p 8983:8983 -v "/tmp/venom\_docs:/var/solr" solr solr-precreate venom\_docs'' to run the container. You could replace /tmp/venom\_docs with your external path. Add the parameter “-d” if you want to run it in the background.

# LIMITATION

* We have not addressed schema/schemaless, Zookeeper and running solr in cloud mode
* We have to depend on solr utilities/commands to create a core/collection (i.e. venom\_docs). The issue is that it always overwrites our venom\_docs folder which we copied to the container.

# CONCLUSION

The most efficient solution to reach our goal is Example 2 because:

* It only needs the docker-compose.yml
* Our data volume is on the host (outside of the container) as Solr recommended.
* It solves the limitation of solr-precreate core overwriting our data (see Limitation section above).

# REFERENCES

<https://github.com/docker-solr/docker-solr>

<https://github.com/docker-solr/docker-solr/blob/master/Docker-FAQ.md>

<http://docs.docker.oeynet.com/samples/library/solr/#creating-cores>

Dockerfile does not allow binding: <https://docs.google.com/document/d/1H_IIMJlTVPRS76y27GVMXLk8VkEsNt2SQEoBh690EqY/edit#>

<https://stackoverflow.com/questions/18873474/can-i-specify-host-directory-to-mount-from-dockerfile>