AMS 2016: UniCloud, Docker at Unidata

Julien Chastang

<2015-12-08 Tue>

Contents

1 LDM, TDS, and RAMADDA on Microsoft Azure VM

1.1 Quick Start

1.2 Preamble

The following instructions describe how to configure a Microsoft Azure VM serving data with the LDM, TDS, and RAMADDA. This document assumes you have access to Azure resources though these instructions should be fairly similar on other cloud providers (e.g., Amazon). They also assume familiarity with Unix, Docker, and Unidata technology in general. You will have to be comfortable entering command at the Unix command line. We will be using Docker images defined here:

`https://github.com/Unidata/Unidata-Dockerfiles`

in addition to a configuration specifically planned for AMS 2016 demonstrations here:

`https://github.com/Unidata/Unidata-Dockerfiles/tree/master/ams2016`

1.2.1 `docker-machine`

Install `docker-machine` on your local computer. `docker-machine` is a command line tool that gives users the ability to create Docker VMs on your local computer or on a cloud provider such as Azure.

1.3 Preliminary Setup on Azure

The instructions assume we will create an Azure VM called `unidata-server.cloudapp.net` abbreviated to `unidata-server`. Tailor the VM name for your purposes when following this document. This VM will be our **Docker Host** from where we will run Docker containers for the LDM, TDS, and RAMADDA.

1.3.1 Create a VM on Azure.

The following `docker-machine` command will create a Docker VM on Azure in which you will run various Docker containers. It will take a few minutes to run (between 5 and 10 minutes). You will have to supply `azure-subscription-id` and `azure-subscription-cert` path. See these Azure `docker-machine` instructions, if you have questions about this process.

```
docker-machine -D create -d azure \
               --azure-subscription-id="3.141" \
               --azure-subscription-cert="/path/to/mycert.pem" \
               --azure-size="ExtraLarge" unidata-server
```

1.3.2 Configure Unix Shell to Interact with New Azure VM.

Execute the following command in your local computer shell environment.

```
eval "$(docker-machine env unidata-server)"
```

1.3.3 `ssh` into VM with `docker-machine`

```
docker-machine ssh unidata-server
```

1.3.4 Install Package(s) with `apt-get`

At the very least, we will need `unzip` on the Azure Docker host.

```
sudo apt-get -qq update
sudo apt-get -qq install unzip
```

1.3.5 Add `ubuntu` User to `docker` Group and Restart Docker

```
sudo usermod -G docker ubuntu
sudo service docker restart
```

1.3.6 Restart Azure VM

At this point, we want to restart the VM to get a fresh start. This command may take a little while...

```
docker-machine restart unidata-server
eval "$(docker-machine env unidata-server)"
```

1.3.7 `ssh` into VM

```
docker-machine ssh unidata-server
```

1.3.8 Install `docker-compose` on VM

`docker-compose` is a tool for defining and running multi-container Docker applications. In our case, we will be running the LDM, TDS, TDM (THREDDS Data Manager) and RAMADDA so `docker-compose` is perfect for this scenario. Install `docker-compose` on the Azure Docker host.

You may have to update version (currently at `1.5.2`).

```
 curl -L \
https://github.com/docker/compose/releases/download/1.5.2/docker-compose-‘uname -s‘-‘un
      > docker-compose
 sudo mv docker-compose /usr/local/bin/
 sudo chmod +x /usr/local/bin/docker-compose
```

1.4 LDM and TDS Configuration

1.4.1 Background

At this point, we have done the preliminary legwork to tackle the next step in this process. We will now want to clone two repositories that will allow us to configure and start running the the LDM, TDS, and RAMADDA. In particular, we will be cloning:

- `Unidata-Dockerfiles`
- `TdsConfig`

1. `Unidata-Dockerfiles`

   The `Unidata-Dockerfiles` repository contains a number of Dockerfiles that pertain to various Unidata technologies (e.g., the LDM) and also projects (e.g., ams2016). As a matter of background information, a `Dockerfile` is a text file that contains commands to build a Docker image containing, for example, a working LDM. These Docker images can subsequently be run by `docker` command line tools, or `docker-compose` commands that rely on a `docker-compose.yml` file. A `docker-compose.yml` file is a text file that captures exactly how one or more containers run including directory mappings (from outside to within the container), port mappings (from outside to within the container), and other information.

2. `TDSConfig`

The `TDSConfig` repository is a project that captures THREDDS and LDM configuration files (e.g., `catalog.xml`, `pqact.conf`) for the TDS at `http://thredds.ucar.edu`. Specifically, these TDS and LDM configurations were meant to work in harmony with one another. We can re-use this configuration with some minor adjustments for running the TDS on the Azure cloud.

1.4.2 `git clone` Repositories

With that background information out of the way, let's clone those repositories by creating `~/git` directory where our repositories will live and issuing some `git` commands.

```
mkdir -p /home/ubuntu/git
git clone https://github.com/Unidata/Unidata-Dockerfiles \
    /home/ubuntu/git/Unidata-Dockerfiles
git clone https://github.com/Unidata/TdsConfig /home/ubuntu/git/TdsConfig
```

1.4.3 Configuring the LDM

1. LDM Directories on Docker Host

   For anyone who has worked with the LDM, you may be familiar with the following directories:

   ```
   etc/
   var/data
   var/logs
   var/queue
   ```

   The LDM `etc` directory is where you will find configuration files related to the LDM including `ldmd.conf`, `pqact` files, `registry.xml`, and `scour.conf`. We will need the ability to easily observe and manipulate the files from **outside** the running LDM container. To that end, we need to find a home for `etc` on the Docker host. The same is true for the `var/data` and `var/logs` directories. Later, we will use Docker commands that have been written on your behalf to mount these directories from **outside** to within the container. The `var/queues` directory will remain inside the container.

   ```
   mkdir -p ~/var/logs
   mkdir -p ~/etc/TDS
   ```

`var/data` is a bit different in that it needs to be mounted on data volume on the Docker host. We will be handling that step further on.

2. LDM Configuration Files

   There is a generic set of LDM configuration files located here `~/git/Unidata-Dockerfiles/ldm/et` However, we will just grab `netcheck.conf` which will remain unmodified.

   ```
   cp ~/git/Unidata-Dockerfiles/ldm/etc/netcheck.conf ~/etc
   ```

   The rest of the LDM configuration files will come from our `ams2016` project directory.

   Also, remember that these files will be used **inside** the LDM container that we will set up shortly. We will now be working with these files:

   - `ldmd.conf`
   - `registry.xml`
   - `scour.conf`

   (a) `ldmd.conf`

   ```
   cp ~/git/Unidata-Dockerfiles/ams2016/ldmd.conf ~/etc/
   ```

   This `ldmd.conf` has been setup for the AMS 2016 demonstration serving the following data feeds:

   - 13km Rapid Refresh

   In addition, there is a `~/git/TdConfig/idd/pqacts/README.txt` file that may be helpful in writing a suitable `ldmd.conf` file.

   (b) `registry.xml`

   ```
   cp ~/git/Unidata-Dockerfiles/ams2016/registry.xml ~/etc/
   ```

   This file has been set up for the AMS 2016 demonstration. Otherwise you would have to edit the `registry.xml` to ensure the `hostname` element is correct. For your own cloud VMs, work with `support-idd@unidata.ucar.edu` to devise a correct `hostname` element so that LDM statsitics get properly reported. Here is an example `hostname` element `unidata-server.azure.unidata.ucar.edu`.

(c) `scour.conf`

You need to scour data or else your disk will full up. The crontab entry that runs scour is in the LDM Docker container. Scouring is invoked once per day.

```
cp ~/git/Unidata-Dockerfiles/ams2016/scour.conf ~/etc/
```

(d) `pqact.conf` and TDS configuration

In the `ldmd.conf` file we copied just a moment ago there is a reference to a `pqact` file; `etc/TDS/pqact.forecastModels`. We need to ensure that file exists by doing the following instructions. Specifically, explode `~/git/TdsConfig/idd/config.zip` into `~/tdsconfig` and `cp -r` the `pqacts` directory into `~/etc/TDS`. **Note** do NOT use soft links. Docker does not like them.

```
mkdir -p ~/tdsconfig/
cp ~/git/TdsConfig/idd/config.zip ~/tdsconfig/
unzip ~/tdsconfig/config.zip -d ~/tdsconfig/
cp -r ~/tdsconfig/pqacts/* ~/etc/TDS
```

3. Upstream Data Feed from Unidata or Elsewhere

   The LDM operates on a push data model. You will have to find someone who will agree to push you the data. If you are part of the American academic community please send a support email to `support-idd@unidata.ucar.edu` to discuss your LDM data requirements.

1.4.4 Configuring the TDS

1. Edit TDS catalog.xml Files

   The `catalog.xml` files for TDS configuration are contained within the `~/tdsconfig` directory. Search for all files terminating in `.xml` in that directory. Edit the xml files for what data you wish to server. See the TDS Documentation for more information on editing these XML files.

   Let's see what is available in the `~/tdsconfig` directory.

```
find ~/tdsconfig -type f -name "*.xml"

/home/ubuntu/tdsconfig/idd/forecastModels.xml
/home/ubuntu/tdsconfig/idd/radars.xml
```

```
/home/ubuntu/tdsconfig/idd/obsData.xml
/home/ubuntu/tdsconfig/idd/forecastProdsAndAna.xml
/home/ubuntu/tdsconfig/idd/satellite.xml
/home/ubuntu/tdsconfig/radar/CS039_L2_stations.xml
/home/ubuntu/tdsconfig/radar/CS039_stations.xml
/home/ubuntu/tdsconfig/radar/RadarNexradStations.xml
/home/ubuntu/tdsconfig/radar/RadarTerminalStations.xml
/home/ubuntu/tdsconfig/radar/RadarL2Stations.xml
/home/ubuntu/tdsconfig/radar/radarCollections.xml
/home/ubuntu/tdsconfig/catalog.xml
/home/ubuntu/tdsconfig/threddsConfig.xml
/home/ubuntu/tdsconfig/wmsConfig.xml
```

1.5 Setting up Data Volumes

As alluded to earlier, we will have to set up data volumes so that the LDM can write data, and the TDS and RAMADDA can have access to that data. The `/mnt` volume on Azure is a good place to store data. I do not know what kind of assurances Azure makes about the reliability of storing your data there for the long term. (I remember reading about this once, but I cannot remember where.) For the LDM this should not be too much of a problem, but for RAMADDA you may wish to be careful.

Let's first display the free disk space with the `df` command.

```
df -H
```

```
Filesystem  Size   Used  Avail  Use%  Mounted                            on
/dev/sda1   31G    1.8G  28G    6%    /
none        4.1k      0  4.1k   0%    /sys/fs/cgroup
udev        7.4G    13k  7.4G   1%    /dev
tmpfs       1.5G   394k  1.5G   1%    /run
none        5.3M      0  5.3M   0%    /run/lock
none        7.4G      0  7.4G   0%    /run/shm
none        105M      0  105M   0%    /run/user
none        66k       0  66k    0%    /etc/network/interfaces.dynamic.d
/dev/sdb1   640G    73M  607G   1%    /mnt
```

Create a `/data` directory where the LDM can write data soft link to the `/mnt` directory. Also, create a `/repository` directory where RAMADDA data will reside.

```
sudo ln -s /mnt /data
sudo mkdir /mnt/ldm/
sudo chown -R ubuntu:docker /data/ldm
sudo mkdir /mnt/repository/
sudo chown -R ubuntu:docker /data/repository
```

These directories will be used by the LDM, TDS, and RAMADDA docker containers when we mount diretories from the Docker host into these containers.

1.6 RAMADDA Configuration

When you fire up RAMADDA for the very first time, you ,ust have a `password.properties` file in the RAMADDA home directory which is `/data/repository/`. See RAMADDA documentation for more details on setting up RAMADDA.

Here is a `pw.properties` file to get you going. Change password below to something more secure!

```
echo ramadda.install.password=changeme! > /data/repository/pw.properties
```

1.7 Opening Ports

Ensure these ports are open on the VM where these containers will run. Ask the cloud administrator for these ports to be open.

Service	External Port
HTTP	80
TDS	8080
RAMADDA	8081
SSL TDM	8443
LDM	388

Note the TDM is an application that works in conjuction with the TDS. It creates indexes for GRIB data in the background, and notifies the TDS via port 8443 when data have been updated or changed. See here to learn more about the TDM.

1.8 Tomcat Logging for TDS and RAMADDA

It is a good idea to mount Tomcat logging directories outside the container so that they can be managed for both the TDS and RAMADDA.

```
mkdir -p ~/logs/ramadda-tomcat
mkdir -p ~/logs/tds-tomcat
```

Note there is also a logging directory in `~/tdsconfig/logs`. All these logging directories should be looked at periodically, not the least to ensure that log files are not filling up your system.

1.9 Starting the LDM TDS RAMADDA TDM

Edit the `docker-compose.yml` file and change the `TDM_PW` to `MeIndexer`.

At this point you are almost ready to run the whole kit and caboodle. But first pull the relevant docker images to make life easier for the subequent `docker-compose` command.

```
docker pull unidata/ldmtds:latest
docker pull unidata/tdm:latest
docker pull unidata/tds:latest
docker pull unidata/ramadda:latest
```

Now you can run

```
docker-compose -f ~/git/Unidata-Dockerfiles/ams2016/docker-compose.yml up -d
```

1.10 Check What is Running

At this point, you should have these services running:

- LDM
- TDS
- TDM
- RAMADDA

Verify you have the TDS running by navigating to:
`http://unidata-server.cloudapp.net/thredds/catalog.html`
You should now be able to download the Unidata Integrated Data Viewer and point it to this catalog to access and display data.

Verify you have the RAMADDA running by navigating to:
`http://unidata-server.cloudapp.net:8081/repository`
If you are going to RAMADDA for the first time, you will have to do some RAMADDA set up.

Also RAMADDA has access to the `/data/ldm` directory so you may wish to set up server-side view of this part of the file system.