Getting 100k+ Tweets

Data Warehousing, MSc Data Analytics

Last Updated: November 6, 2018

1 Introduction

Getting a large number of Tweets requires that we have a machine that is up and running on permanent basis. The easiest way to do this is to make a virtual machine on the cloud – or NCI's OpenStack infrastructure: `https://openstack.cloudenci.ie`, alternatively you can use AWS accessible here: `https://cloud.ncirl.ie` if you prefer (the VM set up is the same once you launch and connect to an AWS instance).

You do not need a big machine for this, a medium-sized instance is fine. The setup will take you about an hour provided you have at least passing knowledge of Linux.

In short, we will be doing the following:

1. building an Linux VM on openstack
2. setting it up as a LAMP (Linux, Apache, MySQL, PHP) server
3. installing a PHP library for connecting to the Twitter streaming API
4. creating a developer account on Twitter
5. pulling tweets in accordance to a set of specific keywords and storing them in a MySQL database

2 Building the VM

Log into OpenStack, and select instances (top of menu on the left). At the top, hit Launch instance, see Figure 1.

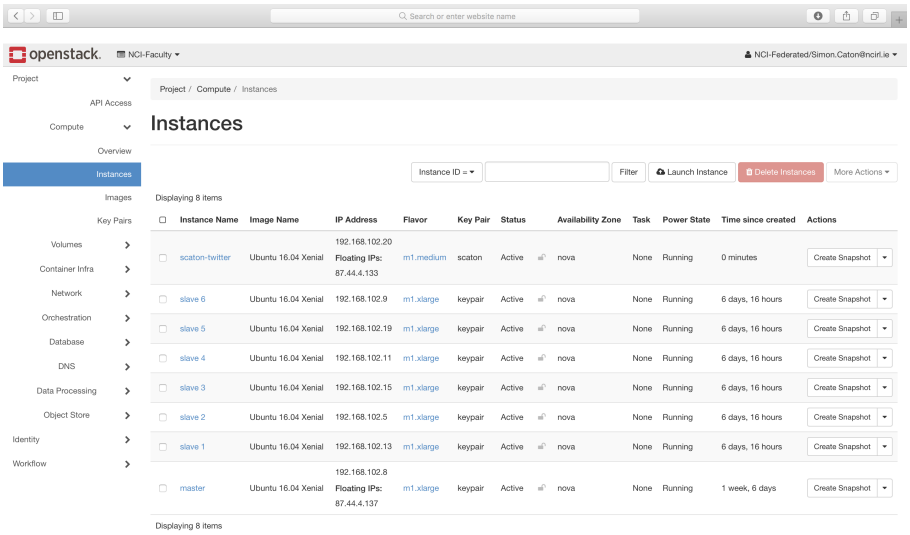

Figure 1: Open Stack instances interface, where you create a new instance (next to the red button)

Open stack has a bug, so the first time you hit launch instance you will not get the complete set of options you need (so hit cancel), the interface should look the same as that in Figure 2.

Now, define a VM with the following properties:

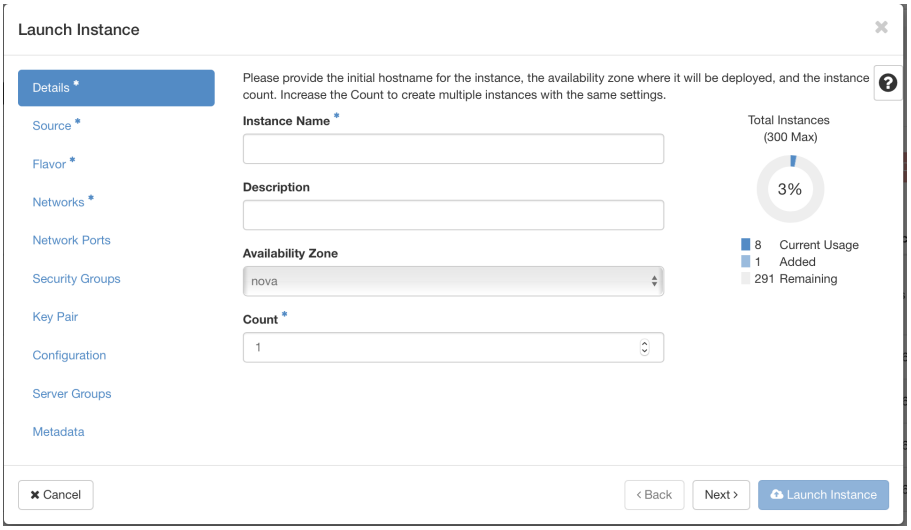

Figure 2: Open Stack launch interface, where you define new instances, if it doesn't look like this, hit cancel and then hit the "launch instance" button again.

- a name prefixed with your student id, e.g. x12345678-DWBI
- using the image Ubuntu 16.04 Xenial (if you prefer a different version of ubuntu then use a different one – my preference is 16.04)
- use the size "m1.medium"
- use the default network (probably called mscdata-net)
- skip network ports – you don't need to do anything here
- use the default security group (it's probably already selected)
- create a key pair (if you don't have one already from DSM), and copy the private key to your machine
- hit launch instance – we don't need the remaining options

You should now see your VM in the list of VMs – you will also see everyone else's so it's important you use a meaningful name!! On the right hand side, you can now associate a floating IP to you machine (click the down arrow next to "create snapshot" and associate a public IP. Whenever you are not using your VM disassociate your public IP, it's only needed when you want to log into the VM. Now you are ready to log into the VM. On windows you will need to use Putty see:`https://support.rackspace.com/how-to/logging-in-with-an-ssh-private-key-on-windows/`, on a Mac / Linux you can just use the terminal see: `https://help.github.com/articles/connecting-to-github-with-ssh/`

You can log into the VM (if using a terminal) as follows (once you have set up your SSH cert using the links above) and obviously replace <ip address> with the IP address of your VM.

```
ssh ubuntu@<ip address>
```

Once logged in we need to execute the following **DO NOT COPY AND PASTE ANY CODE IN THIS DOCUMENT** the pdf encoding will add characters that break the install. In the MySQL setup use a strong password, as this will be publically accessible whenever you have a public IP associated with your VM. Execute the following from the command line:

```
sudo apt-get update
sudo apt-get install -y lamp-server^ git
```

The "^" in the above is intentional and necessary! Once everything here has installed, we need to enable some php extensions:

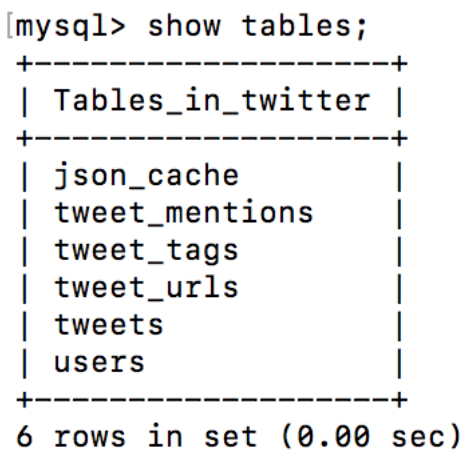

Figure 3: Tables created for storing Tweets

- cd into /etc/php/7.0/apache2
- edit php.ini with sudo access, e.g. sudo nano php.ini
- find the line: ;extension=php_mysqli.dll and insert extension=mysqli.so under it
- save the file
- restart apache: sudo /etc/init.d/apache2 restart

3 Setting up Twitter

Log into `https://developer.twitter.com/content/developer-twitter/en.html` if you don't have a Twitter account you will need to make one. If you have a Twitter account but not a developer account you will need convert your account into a developer account. This can take a while to come through.

3.1 Building the DB

To build the database that will store your tweets:

1. log into the database: mysql -u root -p and enter your password
2. create a new database (name is irrelevant)
3. run the commands here `http://140dev.com/free-twitter-api-source-code-library/twitter-database-server/mysql-database-schema/` to create the necessary tables

You should end up with something similar to Figure 3. Now we need a slight adjustment to the MySQL configuration. cd into /etc/mysql/mysql.conf.d and edit mysqld.cnf (e.g. sudo nano mysqld.cnf), navigate to the line starting bind-address and change 127.0.0.1 to 0.0.0.0 this means you can access the DB from anywhere, not just from the command line of the VM. We will also need to add a new user to the DB we don't want to do everything with root:

```
CREATE USER 'twitter'@'localhost' IDENTIFIED BY 'password';
GRANT ALL PRIVILEGES ON *.* TO 'twitter'@'localhost' WITH GRANT OPTION;
CREATE USER 'twitter'@'%' IDENTIFIED BY 'password';
GRANT ALL PRIVILEGES ON *.* TO 'twitter'@'%' WITH GRANT OPTION;
```

Obviously change "password" to something more secure!! Now restart MySQL using:

```
sudo /etc/init.d/mysql restart
```

3.2 Setting up the PHP code

Now we have a database where we can store our tweets, so we need to now concentrate on pulling tweets from the streaming API. This section is a slightly adapted version of the guide here: `http://140dev.com/free-twitter-api-source-code-library/twitter-database-server/install/`.

Step-1 return to your home directory: cd /home/ubuntu (or just cd)

Step-2 we need to clone the git repository for 140dev as follows:

```
git clone https://github.com/fennb/phirehose.git
```

Step-3 make a directory called twitter, i.e. mkdir twitter, cd into it

Step-4 add the DB config from here: `http://140dev.com/free-twitter-api-source-code-library/twitter-database-server/db-config-php/` this must be called: db_config.php. Add your database name, username and password (from above) in the correct fields. Then add db_lib.php from here: `http://140dev.com/free-twitter-api-source-code-library/twitter-database-server/db-lib-php/`

Step-5 once your twitter developer account is setup, go to: `https://developer.twitter.com/apps`, make a new app, and get your key, secrets and token. Until you have your developer account you will not be able to progress beyond this point.

Step-6 setup the config for tweet handling from here: `http://140dev.com/free-twitter-api-source-code-library/twitter-database-server/140dev-config-php/`, this should be called 140dev-config.php. Enter the email address you want to receive error messages at (the code will email you if things go wrong!), and then also add your Twitter credentials where instructed to do so.

Step-7 now we need to get the script that will listen for tweets, from here: `http://140dev.com/free-twitter-api-source-code-library/twitter-database-server/get-tweets-php/` this should be called get_tweets.php. You need to the change the line near the bottom: $stream -> setTrack(array('recipe')); such that it is a list of the keywords you are interested in. We then need the parser for the tweets, this will change the json object returned by the Twitter API into a structured format and insert each tweet into the DB, we get this here: `http://140dev.com/free-twitter-api-source-code-library/twitter-database-server/parse-tweets-php/` it should be called parse_tweets.php. We then need to change these lines in get_tweets.php:

- require_once('../libraries/phirehose/Phirehose.php');
- require_once('../libraries/phirehose/OauthPhirehose.php');

To (as the guide is a little out of date):

- require_once('../phirehose/lib/Phirehose.php');
- require_once('../phirehose/lib/OauthPhirehose.php');

Step-8 Finally, we can execute the code to start pulling tweets. First test to see it's working by running: php get_tweets.php (wait for 20 seconds) then kill it (ctrl+c) then run php parse_tweets.php (wait 20 seconds) then kill it (ctrl+c). Then go back to your database and run select * from tweets; if you have rows, it worked. Now, to run the code and leave it running:

```
nohup php get_tweets.php > /dev/null &
nohup php parse_tweets.php > /dev/null &
```

Step-9 You can then run:

```
ps ux | grep php
```

to make sure that it is running. Now if you go back to your database, you'll see that tweets are being populated in it. Provided that your keywords are popular enough, you can easily gather 100k tweets per week using this approach.

3.3 Summary

If you have set everything up correctly, your home directory should look as follows:

```
ubuntu@scaton-twitter:~$ ls -l
total 8
drwxrwxr-x 5 ubuntu ubuntu 4096 Nov  6 11:17 phirehose
drwxrwxr-x 2 ubuntu ubuntu 4096 Nov  6 14:22 twitter
```

And the contents of each directory as follows:

```
ubuntu@scaton-twitter:~$ cd phirehose/
ubuntu@scaton-twitter:~/phirehose$ ls -l
total 36
-rw-rw-r-- 1 ubuntu ubuntu   494 Nov  6 11:17 composer.json
drwxrwxr-x 2 ubuntu ubuntu  4096 Nov  6 11:17 example
-rw-rw-r-- 1 ubuntu ubuntu 17987 Nov  6 11:17 gpl.txt
drwxrwxr-x 2 ubuntu ubuntu  4096 Nov  6 11:17 lib
-rw-rw-r-- 1 ubuntu ubuntu  2360 Nov  6 11:17 Readme.md
```

and

```
ubuntu@scaton-twitter:~/phirehose$ cd ..
ubuntu@scaton-twitter:~$ cd twitter/
ubuntu@scaton-twitter:~/twitter$ ls -l
total 100
-rw-rw-r-- 1 ubuntu ubuntu   924 Nov  6 13:04 140dev_config.php
-rw-rw-r-- 1 ubuntu ubuntu   505 Nov  6 13:04 db_config.php
-rw-rw-r-- 1 ubuntu ubuntu  3496 Nov  6 13:47 db_lib.php
-rw-rw-r-- 1 ubuntu ubuntu   471 Nov  6 13:44 db_test.php
-rw-rw-r-- 1 ubuntu ubuntu  1944 Nov  6 14:22 get_tweets.php
-rw-rw-r-- 1 ubuntu ubuntu  5398 Nov  6 14:09 parse_tweets.php
```

4 Closing

See the unstructured R data lab on moodle for how to now use your database. Do make sure that you check in on your VM from time to time, MySQL tables can crash when they get large, and in that time you won't be collecting any tweets.

If you need to stop collecting for any reason, you can run the following to find processes that are running php:

```
ubuntu@scaton-twitter:~$ ps ux | grep php
ubuntu   12945  0.0  0.4 138736 16488 pts/1    S    18:48   0:03 php get_tweets.php
ubuntu   12946  0.0  0.4 134388 16276 pts/1    S    18:48   0:05 php parse_tweets.php
ubuntu   13185  0.0  0.0  12944   944 pts/0    S+   20:32   0:00 grep --color=auto php
```

To terminate the collection processes get the process numbers associated with get_tweets.php (12945) and parse_tweets.php(12946) and kill them as follows:

```
ubuntu@scaton-twitter:~$ sudo kill 12945
ubuntu@scaton-twitter:~$ sudo kill 12946
```