Docker Amazon Kinesis Twitter Feed

Kirk S. Kalvar

12/31/2017

# Overview

I’ve wanted to play with a Twitter Feed as a source for a Big Data Project for ages. Not that there isn’t interesting data to work with out there but with live feeds and tons of it, what’s not to like? Twitter is kind enough to let anyone have access to this feed simply by creating a feed configuration on their website. I saw a Tutorial by Daniel Blazevski and I had to at least give it a try. One small difference, instead of using an Amazon AMI I wanted to do this locally in a docker container. And of course then it could be run virtually anywhere.

One additional feature I wanted to add. At least do some exploration, Managing Secrets. You know the database password we all have hardcoded into configuration files or code? Well I’m definitely aware this isn’t a best practice but wanted to move from acknowledgment to working with potential solutions. So even though we don’t incorporate anything like Amazon System Manager we will at least remove any hardcoding of passwords or keys in the container configuration or code.

So the order of events will be to create a twitter feed, gather all variables we’ll need for the twitter feed, gather all the AWS variables we need to create the Kinesis Stream, build the container, start the container using the variables we’ve collected for Twitter and AWS, and then stream the feed via Amazon Kinesis using python using the container we built.

# Twitter Feed Access

You will need an active Twitter Account

1. Go to <https://apps.twitter.com/app/new>
2. Name: <Your Application Name>
3. Description: <Your Application Name>
4. Website: <Your Website>
5. Callback URL: Not Needed
6. Developer Agreement: Checked
7. Click on “Create Your Twitter application”

Your application has been created.

1. Copy Consumer Key (API Key) for future reference
2. Copy Secret (API Secret) for future reference
3. Scroll down to “Token Actions”
4. Click on “Generate Access Tokens”

Your Access Token has been created.

1. Copy Access Token for future reference
2. Copy Access Token Secret for future reference

export twitter\_consumer\_key=" <your consumer key>"

export twitter\_consumer\_secret=" <your consumer secret>"

export twitter\_access\_token\_key=" <your access token>"

export twitter\_access\_token\_secret=" <your access token secret>"

# AWS Access

You will need an active AWS Account

export amazonec2\_access\_key=”<your access key>”

export amazonec2\_secret\_key=”<your secret key>”

# Checkout Dockerfile from github and Build Container

1. git clone https://github.com/kskalvar/docker-amazon-kinesis-twitter.git
2. cd docker-amazon-kinesis-twitter
3. docker build -t twitter-aws-kinesis-runtime .

# Run twitter-aws-kinesis-runtime

1. docker run --name twitter-aws-kinesis \

-e AWS\_ACCESS\_KEY\_ID=$amazonec2\_access\_key \

-e AWS\_SECRET\_ACCESS\_KEY=$amazonec2\_secret\_key \

-e AWS\_DEFAULT\_REGION=$amazonec2\_region \

-e CONSUMERKEY=$twitter\_consumer\_key \

-e CONSUMERSECRET=$twitter\_consumer\_secret \

-e ACCESSTOKEN=$twitter\_access\_token\_key \

-e ACCESSTOKENSEC=$twitter\_access\_token\_secret \

-it --rm twitter-aws-kinesis-runtime /bin/bash

# Create AWS Kinesis Stream and Start the Twitter Feed

You’ll already be logged inside the container from the above step

1. cd /kinesis
2. python create-stream.py
3. python kinesis.py

# Start the Simple Consumer

The container is already started; we’ll attach and start the consumer in a separate terminal

1. docker exec -it twitter-aws-kinesis /bin/bash
2. cd /kinesis
3. python simple-consumer.py

# Delete the AWS Kinesis Stream

1. python delete-stream.py

# Reference

Tutorial on AWS serverless architecture using Kinesis, DynamoDB and Twitter by Daniel Blazevski

<https://blog.insightdatascience.com/getting-started-with-aws-serverless-architecture-tutorial-on-kinesis-and-dynamodb-using-twitter-38a1352ca16d>