SparkSteps Documentation

Release 0.3.4.dev1+g7490783

Kamil Sindi

Dec 05, 2018

Contents

CHAPTER 1

Spark Steps

SparkSteps allows you to configure your EMR cluster and upload your spark script and its dependencies via AWS S3. All you need to do is define an S3 bucket.

1.1 Install

```
pip install sparksteps
```

1.2 CLI Options

```
Prompt parameters:
  app                           main spark script for submit spark (required)
  app-args:                     arguments passed to main spark script
  aws-region:                   AWS region name
  bid-price:                    specify bid price for task nodes
  bootstrap-action:             include a bootstrap script (s3 path)
  cluster-id:                   job flow id of existing cluster to submit to
  debug:                        allow debugging of cluster
  defaults:                     spark-defaults configuration of the form key1=val1␣
↪key=val2
  dynamic-pricing-master:       use spot pricing for the master nodes.
  dynamic-pricing-core:         use spot pricing for the core nodes.
  dynamic-pricing-task:         use spot pricing for the task nodes.
  ebs-volume-size-core:         size of the EBS volume to attach to core nodes in GiB.
  ebs-volume-type-core:         type of the EBS volume to attach to core nodes␣
↪(supported: [standard, gp2, io1]).
  ebs-volumes-per-core:         the number of EBS volumes to attach per core node.
  ebs-optimized-core:           whether to use EBS optimized volumes for core nodes.
  ebs-volume-size-task:         size of the EBS volume to attach to task nodes in GiB.
  ebs-volume-type-task:         type of the EBS volume to attach to task nodes.
```

(continues on next page)

(continued from previous page)

```
  ebs-volumes-per-task:          the number of EBS volumes to attach per task node.
  ebs-optimized-task:            whether to use EBS optimized volumes for task nodes.
  ec2-key:                       name of the Amazon EC2 key pair
  ec2-subnet-id:                 Amazon VPC subnet id
  help (-h):                     argparse help
  keep-alive:                    whether to keep the EMR cluster alive when there are␣
↪no steps
  log-level (-l):                logging level (default=INFO)
  instance-type-master:          instance type of of master host (default='m4.large')
  instance-type-core:            instance type of the core nodes, must be set when num-
↪core > 0
  instance-type-task:            instance type of the task nodes, must be set when num-
↪task > 0
  maximize-resource-allocation: sets the maximizeResourceAllocation property for the␣
↪cluster to true when supplied.
  name:                          specify cluster name
  num-core:                      number of core nodes
  num-task:                      number of task nodes
  release-label:                 EMR release label
  s3-bucket:                     name of s3 bucket to upload spark file (required)
  s3-dist-cp:                    s3-dist-cp step after spark job is done
  submit-args:                   arguments passed to spark-submit
  tags:                          EMR cluster tags of the form "key1=value1 key2=value2"
  uploads:                       files to upload to /home/hadoop/ in master instance
```

1.3 Example

```
AWS_S3_BUCKET = <insert-s3-bucket>
cd sparksteps/
sparksteps examples/episodes.py \
  --s3-bucket $AWS_S3_BUCKET \
  --aws-region us-east-1 \
  --release-label emr-4.7.0 \
  --uploads examples/lib examples/episodes.avro \
  --submit-args="--deploy-mode client --jars /home/hadoop/lib/spark-avro_2.10-2.0.2-
↪custom.jar" \
  --app-args="--input /home/hadoop/episodes.avro" \
  --tags Application="Spark Steps" \
  --debug
```

The above example creates an EMR cluster of 1 node with default instance type *m4.large*, uploads the pyspark script episodes.py and its dependencies to the specified S3 bucket and copies the file from S3 to the cluster. Each operation is defined as an EMR "step" that you can monitor in EMR. The final step is to run the spark application with submit args that includes a custom spark-avro package and app args "–input".

1.4 Run Spark Job on Existing Cluster

You can use the option `--cluster-id` to specify a cluster to upload and run the Spark job. This is especially helpful for debugging.

1.5 Dynamic Pricing (alpha)

Use CLI option `--dynamic-pricing-<instance-type>` to allow sparksteps to dynamically determine the best bid price for EMR instances within a certain instance group.

Currently the algorithm looks back at spot history over the last 12 hours and calculates `min(50% * on_demand_price, max_spot_price)` to determine bid price. That said, if the current spot price is over 80% of the on-demand cost, then on-demand instances are used to be conservative.

Note: code depends on ec2instances for getting demand price.

1.6 Testing

```
make test
```

1.7 Blog

Read more about sparksteps in our blog post here: https://www.jwplayer.com/blog/sparksteps/

1.8 License

Apache License 2.0

CHAPTER 2

Indices and tables

- genindex
- modindex
- search