spark-submit options explained

ddas.tech/spark-submit-options-explained/

November 22, 2022

Created By: Debasis Das (21-Nov-2022)

In this post we will understand the details of spark-submit options

The `spark-submit` script in Spark's `bin` directory is used to launch applications on a cluster. It can use all of Spark's supported cluster managers through a uniform interface so we don't have to configure our application especially for each one.

–master MASTER_URL	spark://host:port, mesos://host:port, yarn, k8s://https://host:port, or local (Default: local[*]). The master url for the cluster
–deploy-mode DEPLOY_MODE	Whether to launch the driver program locally (“client”) or on one of the worker machines inside the cluster (“cluster”) (Default: client).
— class CLASS_NAME	Your application’s main class (for Java / Scala apps). The entry point for our applications.
— name NAME	Name of our applications
— jars JARS	Comma separated list of jars to include on the driver and executor classpaths
— packages	Comma-separated list of maven coordinates of jars to include on the driver and executor classpaths. Will search the local maven repo, then maven central and any additional remote repositories given by –repositories. The format for the coordinates should be groupId:artifactId:version.
–exclude-packages	Comma-separated list of groupId:artifactId, to exclude while resolving the dependencies provided in –packages to avoid dependency conflicts.
–repositories	Comma-separated list of additional remote repositories to search for the maven coordinates given with –packages.
–py-files PY_FILES	`Comma-separated list of .zip, .egg, or .py files to place on the PYTHONPATH for Python apps.`
–files FILES	Comma-separated list of files to be placed in the working directory of each executor. File paths of these files in executors can be accessed via **SparkFiles.get(fileName).**
–archives ARCHIVES	Comma-separated list of archives to be extracted into the working directory of each executor.
–conf, -c PROP=VALUE	Arbitrary Spark configuration property.
–properties-file FILE	Path to a file from which to load extra properties. If not specified, this will look for **conf/spark-defaults.conf.**
–driver-memory MEM	Memory for driver (e.g. 1000M, 2G) (Default: 1024M).
–driver-java-options	Extra Java options to pass to the driver.

–driver-library-path	Extra library path entries to pass to the driver.
–driver-class-path	`Extra class path entries to pass to the driver. Note that jars added with --jars are automatically included in the classpath.`
–executor-memory MEM	Memory per executor (e.g. 1000M, 2G) (Default: 1G).
–proxy-user NAME	User to impersonate when submitting the application. This argument does not work with –principal / –keytab.
–version	Print the version of current Spark.

spark-submit options

Cluster deploy mode only:	
–driver-cores NUM	Number of cores used by the driver, only in cluster mode (Default: 1).
Spark standalone or Mesos with cluster deploy mode only:	
–supervise	If given, restarts the driver on failure.
Spark standalone, Mesos or K8s with *cluster deploy mode only:*	
–kill SUBMISSION_ID	If given, kills the driver specified.
–status SUBMISSION_ID	If given, requests the status of the driver specified.
Spark standalone, Mesos and Kubernetes only:	
–total-executor-cores NUM	Total cores for all executors.
Spark standalone, YARN and Kubernetes only:	
–executor-cores NUM	Number of cores used by each executor. (Default: 1 in YARN and K8S modes, or all available cores on the worker in standalone mode).
Spark on YARN and Kubernetes only:	
–num-executors NUM	Number of executors to launch (Default: 2). If dynamic allocation is enabled, the initial number of executors will be at least NUM.
–principal PRINCIPAL	Principal to be used to login to KDC.
–keytab KEYTAB	The full path to the file that contains the keytab for the principal specified above.
Spark on YARN only:	
–queue QUEUE_NAME	The YARN queue to submit to (Default: “default”).

spark-submit options

- A common deployment strategy is to submit the application from a gateway machine that is physically co-located with the worker machines (e.g. Master node in a standalone EC2 cluster).
- In this setup, `client` mode is appropriate. In `client` mode, the driver is launched directly within the `spark-submit` process which acts as a *client* to the cluster. The input and output of the application is attached to the console. *Thus, this mode is especially suitable for applications that involve the REPL (e.g. Spark shell).*
- Alternatively, if the application is submitted from a machine far from the worker machines (e.g. locally on your laptop), it is common to use `cluster` mode to minimize network latency between the drivers and the executors. Currently, the standalone mode does not support cluster mode for Python applications.

```
# Run application locally on 8 cores
./bin/spark-submit \
  --class org.apache.spark.examples.SparkPi \
  --master local[8] \
  /path/to/examples.jar \
  100

# Run on a Spark standalone cluster in client deploy mode
./bin/spark-submit \
  --class org.apache.spark.examples.SparkPi \
  --master spark://IP:PORT \
  --executor-memory 20G \
  --total-executor-cores 100 \
  /path/to/examples.jar \
  1000

# Run on a Spark standalone cluster in cluster deploy mode with supervise
./bin/spark-submit \
  --class org.apache.spark.examples.SparkPi \
  --master spark://IP:PORT \
  --deploy-mode cluster \
  --supervise \
  --executor-memory 20G \
  --total-executor-cores 100 \
  /path/to/examples.jar \
  1000

# Run a Python application on a Spark standalone cluster
./bin/spark-submit \
  --master spark://IP:PORT \
  examples/src/main/python/pi.py \
  1000
```

Master URLs

The master URL passed to Spark can be in one of the following formats:

local	Run Spark locally with one worker thread (i.e. no parallelism at all).
local[K]	Run Spark locally with K worker threads (ideally, set this to the number of cores on your machine).
local[K,F]	Run Spark locally with K worker threads and F maxFailures
local[*]	Run Spark locally with as many worker threads as logical cores on your machine.
local[*,F]	Run Spark locally with as many worker threads as logical cores on your machine and F maxFailures.
local-cluster[N,C,M]	Local-cluster mode is only for unit tests. It emulates a distributed cluster in a single JVM with N number of workers, C cores per worker and M MiB of memory per worker.
spark://HOST:PORT	Connect to the given Spark Standalone Cluster master. The port must be whichever one your master is configured to use, which is 7077 by default.
yarn	Connect to a YARN cluster in client or cluster mode depending on the value of –deploy-mode. The cluster location will be found based on the HADOOP_CONF_DIR or YARN_CONF_DIR variable.
k8s://HOST:PORT	Connect to a Kubernetes cluster in client or cluster mode depending on the value of –deploy-mode. The HOST and PORT refer to the Kubernetes API Server. It connects using TLS by default. In order to force it to use an unsecured connection, you can use k8s://http://HOST:PORT.

Master URLs

Reference

https://spark.apache.org/docs/latest/submitting-applications.html