

How to submit/execute a Spark application

Spark-submit

- Spark programs are executed (submitted) by using the spark-submit command
 - It is a command line program
 - It is characterized by a set of parameters
 - E.g., the name of the jar file containing all the classes of the Spark application we want to execute
 - The name of the Driver class
 - The parameters of the Spark application
 - etc.

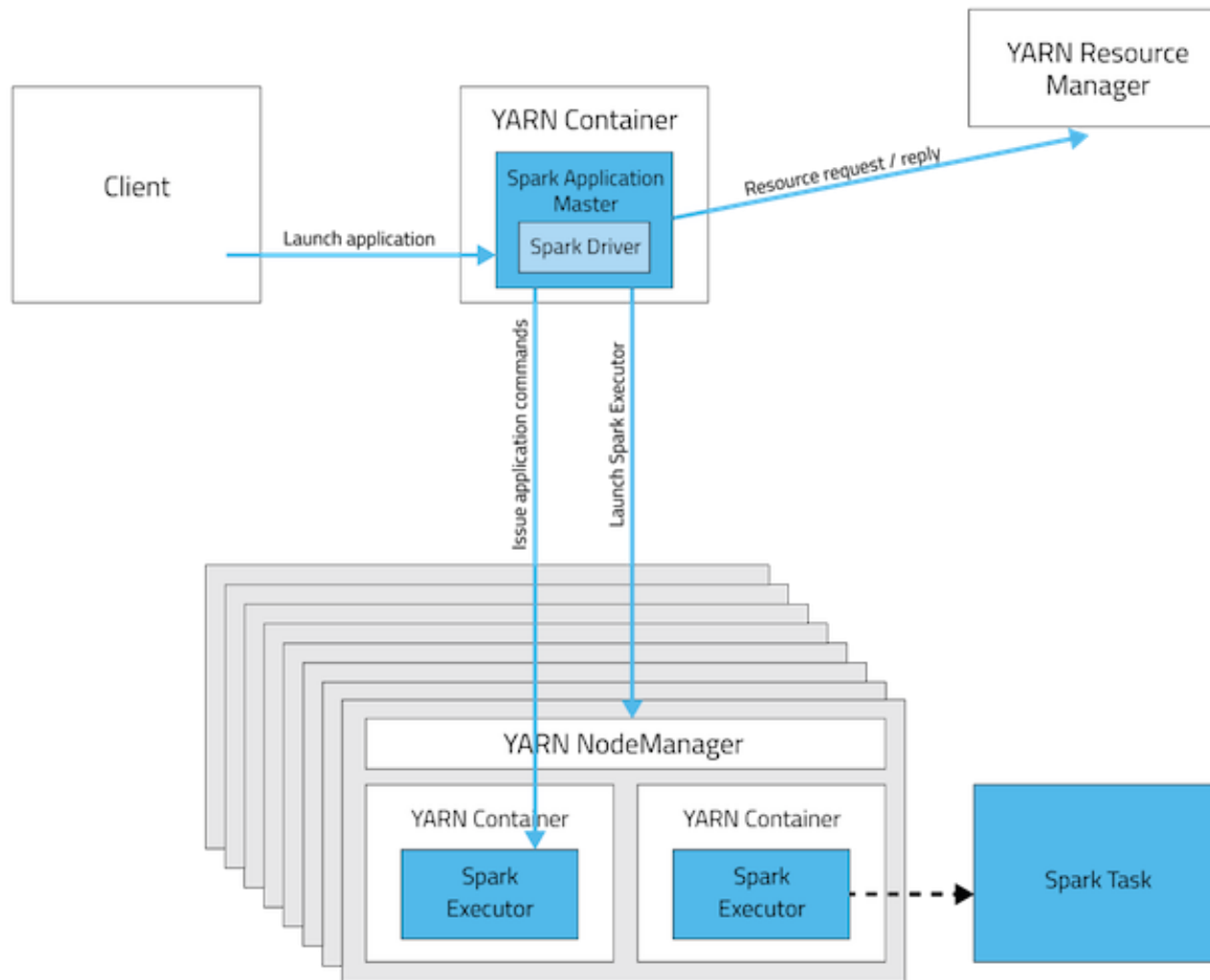
Spark-submit

- spark-submit has also two parameters that are used to specify where the application is executed
 - **--master** option
 - Specify which environment/scheduler is used to execute the application
 - spark://host:port The spark scheduler is used
 - mesos://host:port The memos scheduler is used
 - yarn The YARN scheduler (i.e., the one of Hadoop)
 - local The application is executed exclusively on the local PC

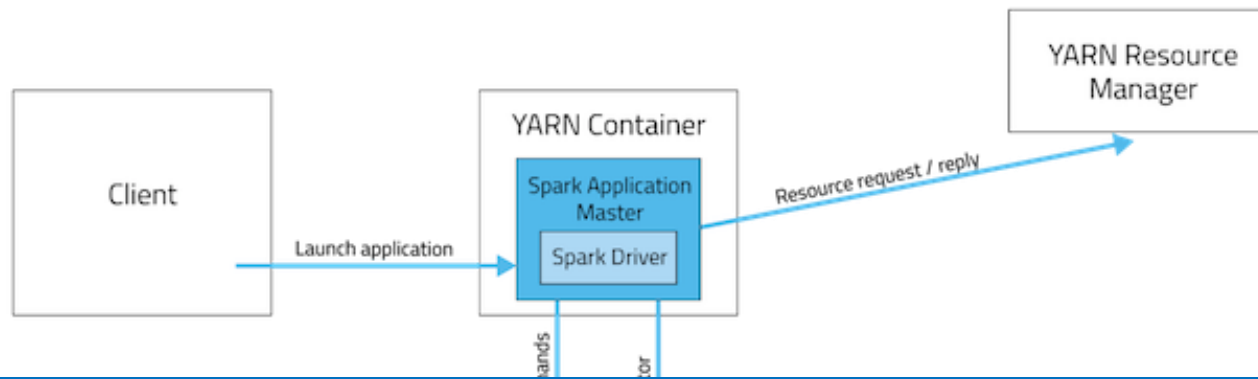
Spark-submit

- **--deploy-mode** option
 - Specify where the Driver is launched/executed
 - client The driver is launched locally (in the "local" PC executing spark-submit)
 - cluster The driver is launched on one node of the cluster

Cluster Deployment Mode

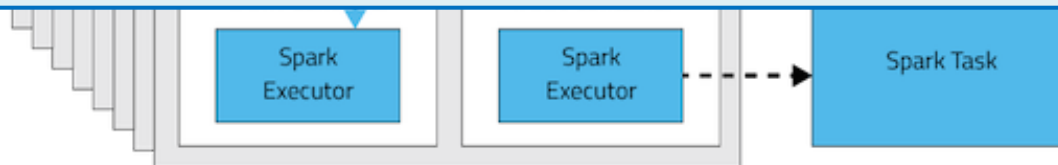


Cluster Deployment Mode

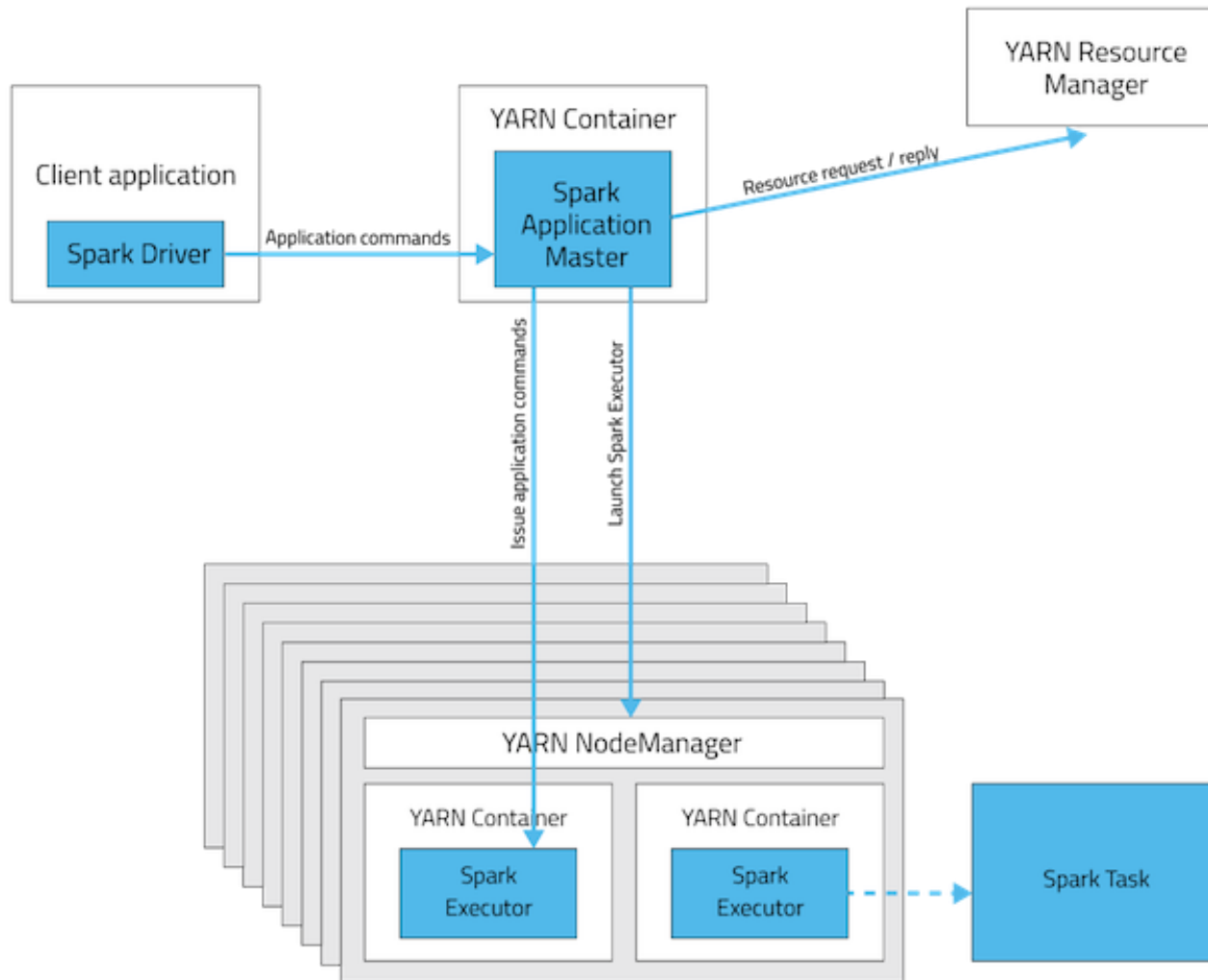


In cluster mode

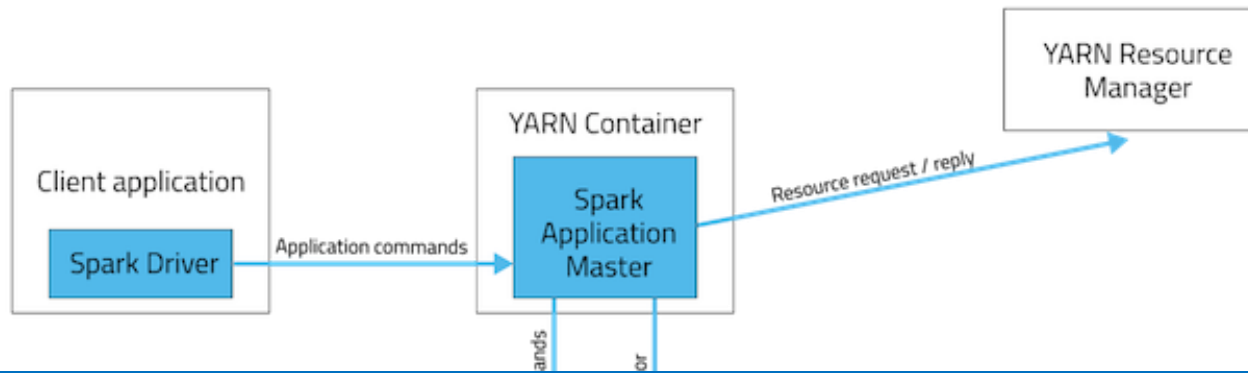
- The Spark driver runs in the ApplicationMaster on a cluster node.
- The cluster nodes are used also to store RDDs and execute transformations and actions on the RDDs
- A single process in a YARN container is responsible for both driving the application and requesting resources from YARN.
- The resources (memory and CPU) of the client that launches the application are not used.



Client Deployment Mode

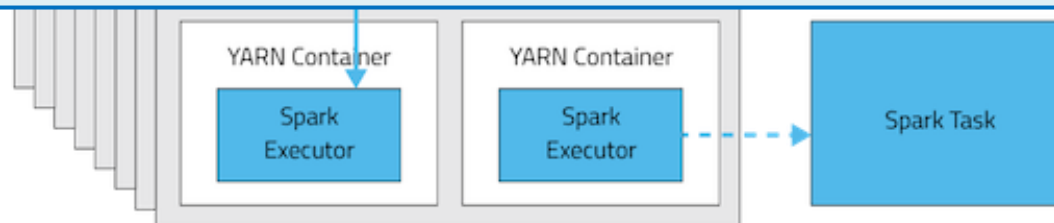


Client Deployment Mode



In client mode

- The Spark driver runs on the host where the job is submitted (i.e., the resources of the client are used to execute the Driver)
- The cluster nodes are used to store RDDs and execute transformations and actions on the RDDs
- The ApplicationMaster is responsible only for requesting executor containers from YARN.



Spark-submit: setting executors

- Spark-submit allows specifying
 - The number of executors
 - `--num-executors NUM`
 - Default value: NUM=2 executors
 - The number of cores per executor
 - `--executor-cores NUM`
 - Default value: NUM=1 core
 - Main memory per executor
 - `--executor-memory MEM`
 - Default value: MEM=1GB
- The maximum values of these parameters are limited by the configuration of the cluster

Spark-submit: setting driver

- Spark-submit allows specifying
 - The number of cores for the driver
 - `--driver-cores NUM`
 - Default value: NUM=1 core
 - Main memory for the driver
 - `--driver-memory MEM`
 - Default value: MEM=1GB
 - Also the maximum values of these parameters are limited by the configuration of the cluster when the `deploy-mode` is set to `cluster`

Spark-submit: Execution on the cluster

- The following command submits a Spark application on a Hadoop cluster

```
spark-submit --class it.polito.spark.DriverMyApplication --  
deploy-mode cluster --master yarn MyApplication.jar  
arguments
```

- It executes/submits the application `it.polito.spark.DriverMyApplication` contained in `MyApplication.jar`
- The application is executed on a Hadoop cluster based on the YARN scheduler
 - Also the Driver is executed in a node of the cluster

Spark-submit: Local execution

- The following command submits a Spark application on a local PC

```
spark-submit --class it.polito.spark.DriverMyApplication --  
deploy-mode client --master local MyApplication.jar  
arguments
```

- It executes/submits the application `it.polito.spark.DriverMyApplication` contained in `MyApplication.jar`
- The application is completely executed on the local PC
 - Both Driver and Executors
 - Hadoop is not needed in this case
 - You only need the Spark software