Running Spark On A Cluster

Running Spark On A Cluster

- In this chapter, you will learn
  - Spark clustering concepts and terminology
  - Spark deployment options
  - How to run a Spark application on a Spark Standalone cluster

Chapter Topics

- Overview
- A Spark Standalone Cluster
- The Spark Standalone Web UI
- Spark Deployment Options
- Summary
- Review
- References
- Hands-On Exercise: Running The Spark Shell On A Cluster

Spark Cluster Options

- Spark can run
  - Locally on a single machine, in a single JVM
    - No distributed processing
    - Can use the local drive or HDFS
    - Analogous to local mode in Hadoop
    - Good For testing
  - On a cluster
    - Single use cluster
      - Spark Standalone
      - Dedicated Spark cluster
    - Mixed use cluster
      - Apache Hadoop YARN (Yet Another Resource Negotiator)
      - Apache Mesos
      - Both capable of managing resources across a cluster using both Spark and Hadoop
    - Other options may be available in the future

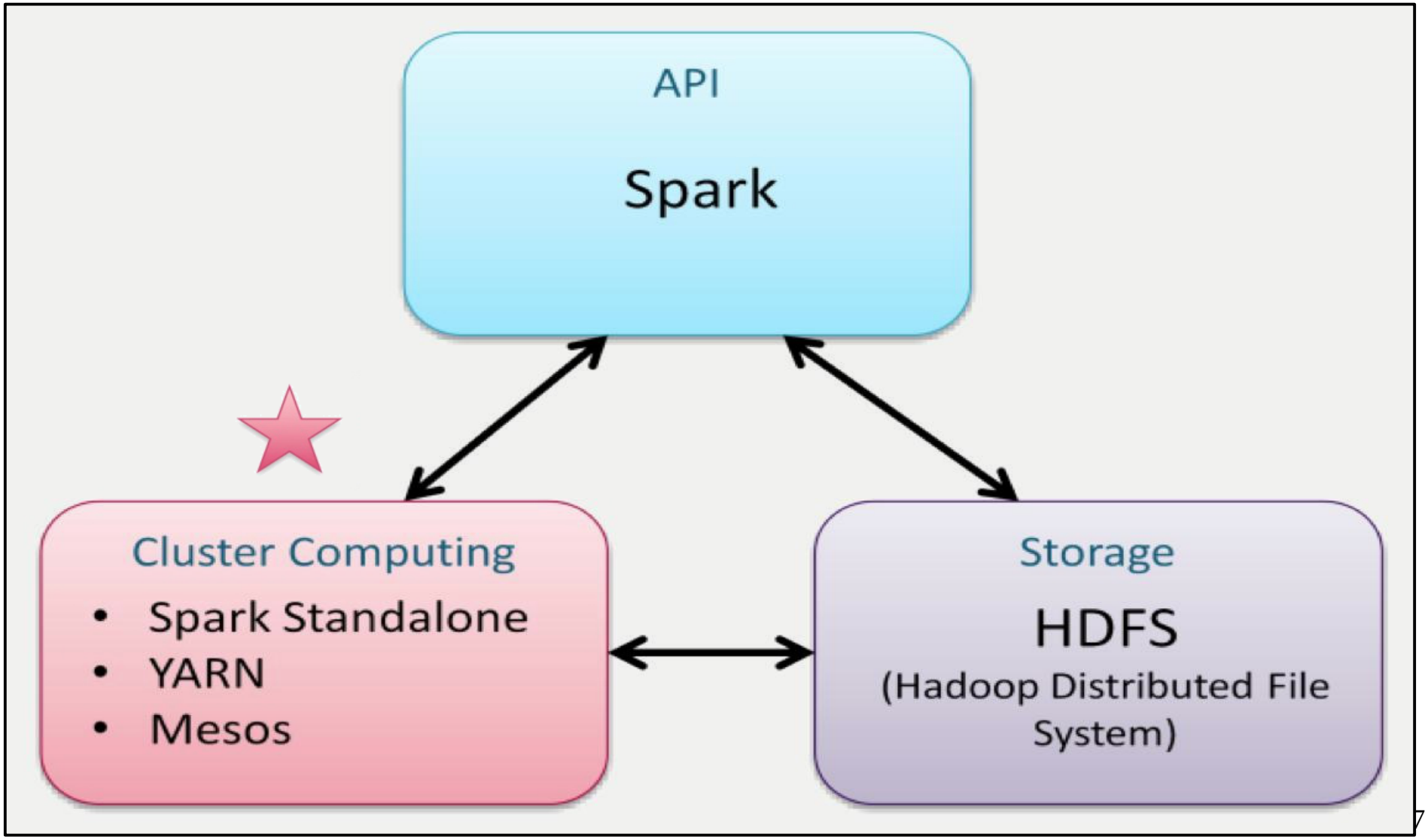
Supported Cluster Resource Managers

- Spark Standalone
  - Included with Spark
  - Easy to install and run
  - Limited configurability and scalability
    - About 12 configurable properties (as opposed to over 200 for Hadoop YARN)
    - Does not share cluster resources dynamically with other applications (the way YARN does)
      - e.g. You cannot deploy both Spark and Hadoop applications in Spark Standalone, as they do not play well together
  - Useful for testing, development, or small production systems
- Hadoop YARN
  - https://hadoop.apache.org/docs/current/hadoop-yarn/hadoop-yarn-site/YARN.html
  - lncluded in CDH
  - Most common for production sites
  - Scalable to 7000+ nodes
  - Very configurable
  - Capable of sharing cluster resources across platforms (MapReduce, Impala, etc.)
- Apache Mesos
  - https://mesos.apache.org/
  - First platform supported by Spark
  - Scalable to “10,000s of nodes"
  - Also capable of sharing cluster resources across platforms (MapReduce, Impala, etc.)
  - Now used less often
  - Mesos is used by Apple (Siri), Twitter, Airbnb, Google, PayPal and OpenTable
    - http://www.slideshare.net/caniszczyk/apache-mesos-at-twitter-texas-linuxfest-2014

Why Run On A Cluster?

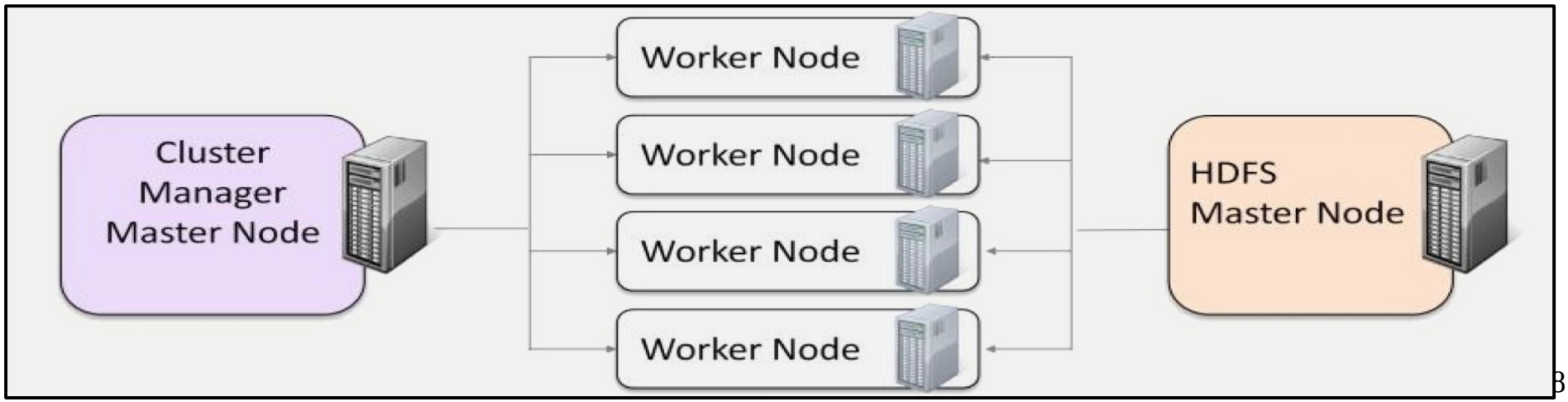
- To get the benefits of distributed processing
  - Parallelism
  - Fault tolerance
  - Scalability
- Local mode is useful for development and testing
- Production use is almost always on a cluster

Distributed Processing With The Spark Framework



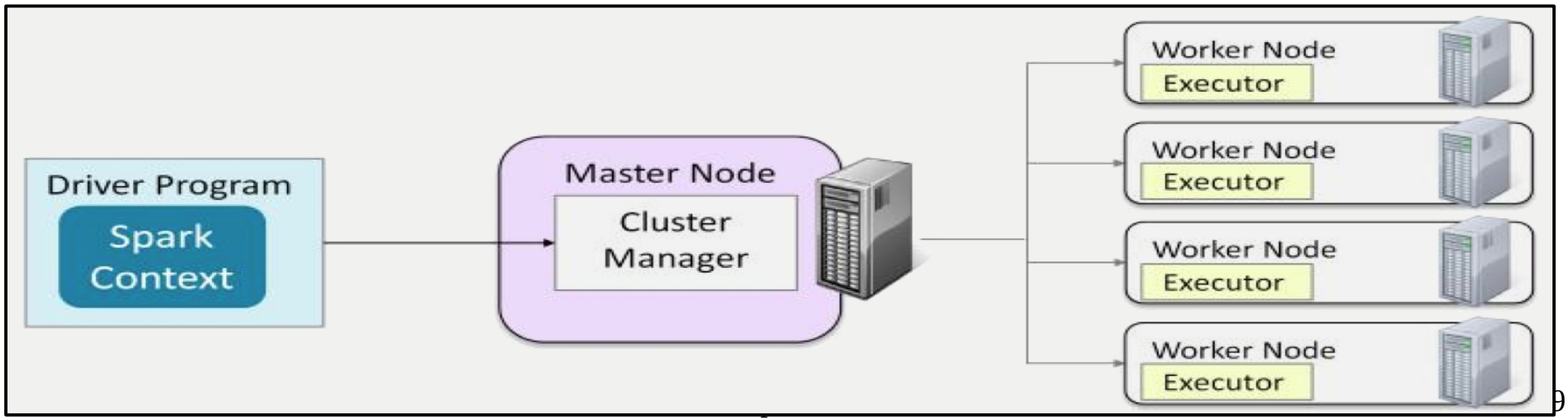
Spark Cluster Terminology

- A cluster is a group of computers working together on a single problem
  - Usually running HDFS (for storage) in addition to Spark Standalone, YARN or Mesos (for processing)
- A node is an individual computer in the cluster
  - Master nodes manage the distribution of work and data to the worker nodes
  - Workers store and process the data on the node itself
- A daemon is a program running on a node
  - Each daemon performs different functions in the cluster



The Spark Driver Program

- A Spark Driver
  - The "main" program
    - Can be either the Spark Shell itself, or a Spark application
  - Creates a Spark Context configured with the URL for the cluster Master
  - Communicates with Cluster Manager to distribute tasks to executors, dedicated to that particular application
    - One executor (JVM) per Job (unlike Hadoop, which has one JVM per task)



Starting The Spark Shell On A Cluster

- Set the Spark Shell master

Property	Description
url	The url of the cluster manager
local[*]	Run locally with as many threads as cores (default)
local[n]	Run locally with n worker threads
local	Run locally without distributed processing (default)

Python Shell: pyspark	Spark Shell: spark-shell
`$ MASTER=spark://masternode:7077 pyspark`	`$ spark-shell --master spark://masternode:7077`

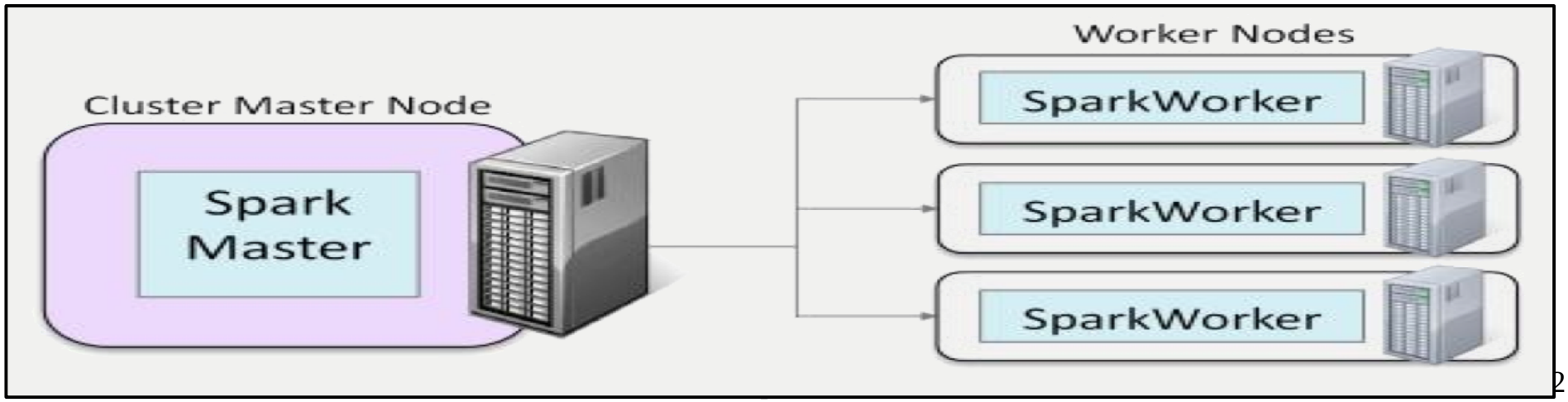
```
$ pyspark --master local[*]                    # Python
$ spark-shell --master local[*]                # Spark
```

Chapter Topics

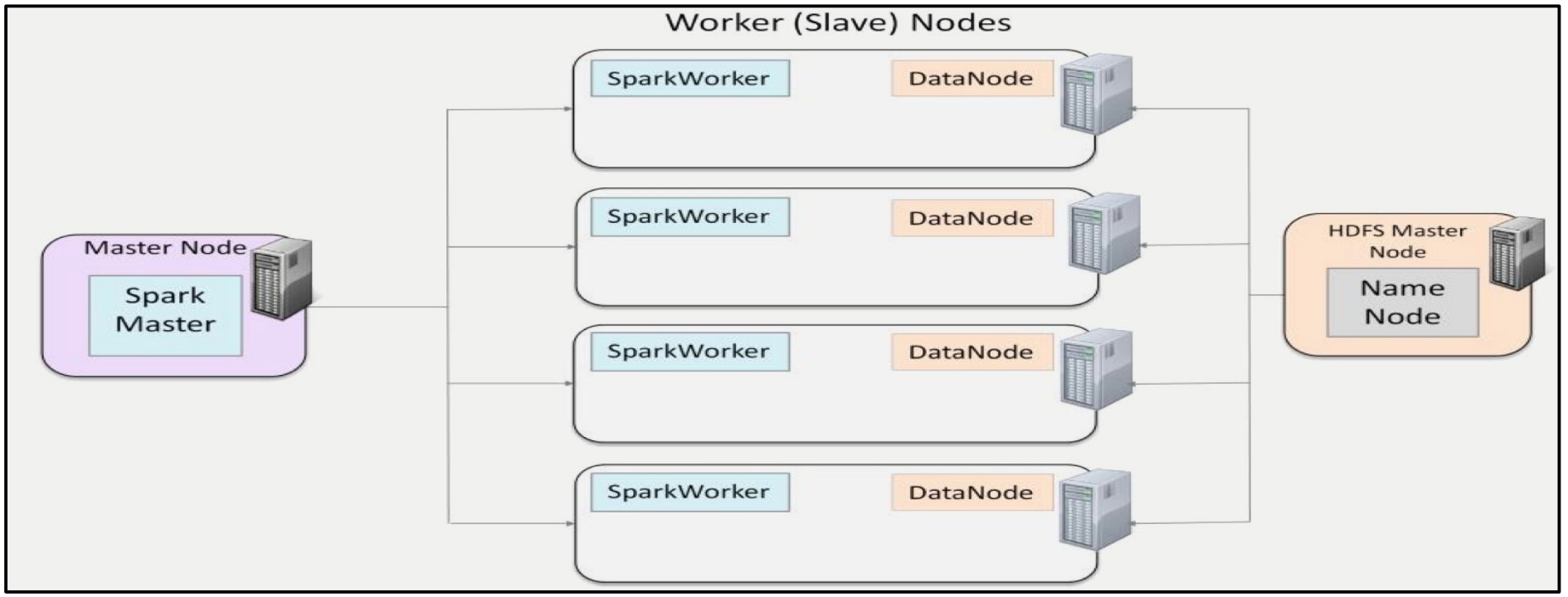
- Overview
- A Spark Standalone Cluster
- The Spark Standalone Web UI
- Spark Deployment Options
- Summary
- Review
- References
- Hands-On Exercise: Running The Spark Shell On A Cluster

Spark Standalone Daemons

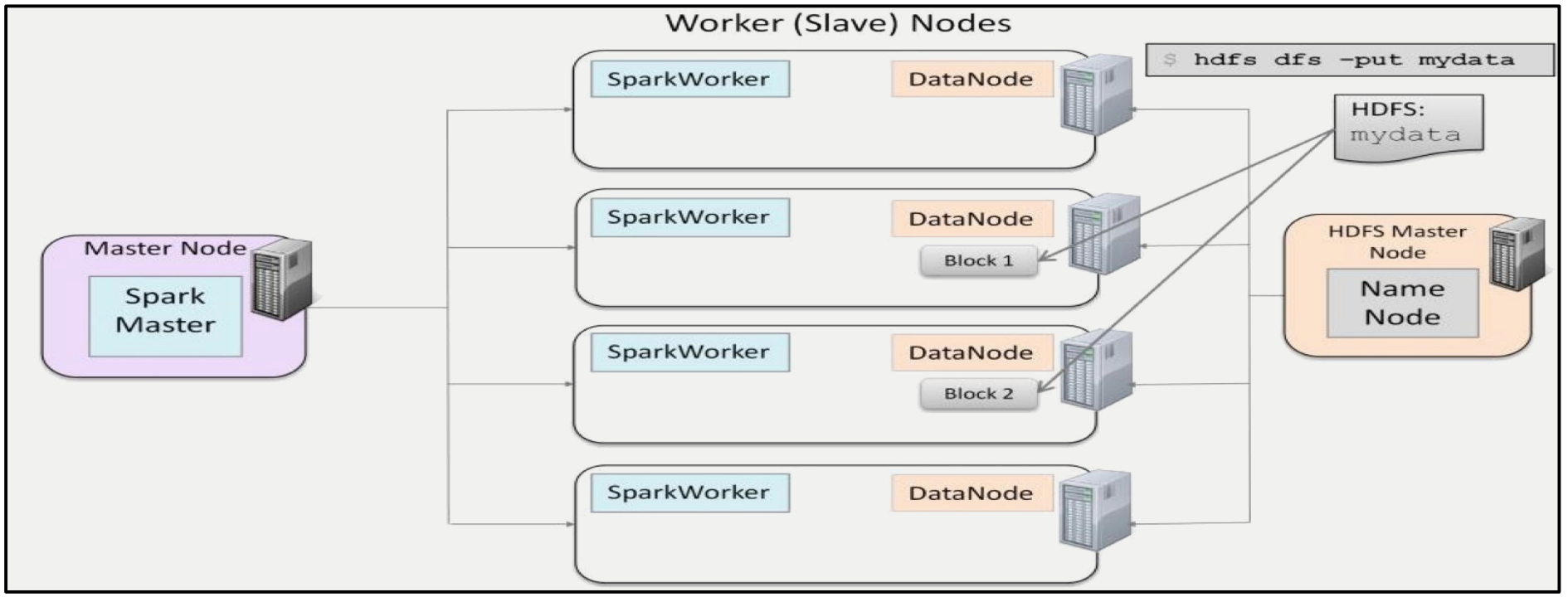
- Spark Master
  - One per cluster (like MR1 Hadoop Job Tracker)
  - Manages applications, distributes individual tasks to Spark Workers
- Spark Worker
  - Typically one per worker node (like MR1 Hadoop Task Tracker)
    - Unless running multiple daemons on a single host (e.g. Pseudo-distributed mode)
  - Starts and monitors Executors for applications
- Each daemon (process) runs in its own Java Virtual Machine (JVM)



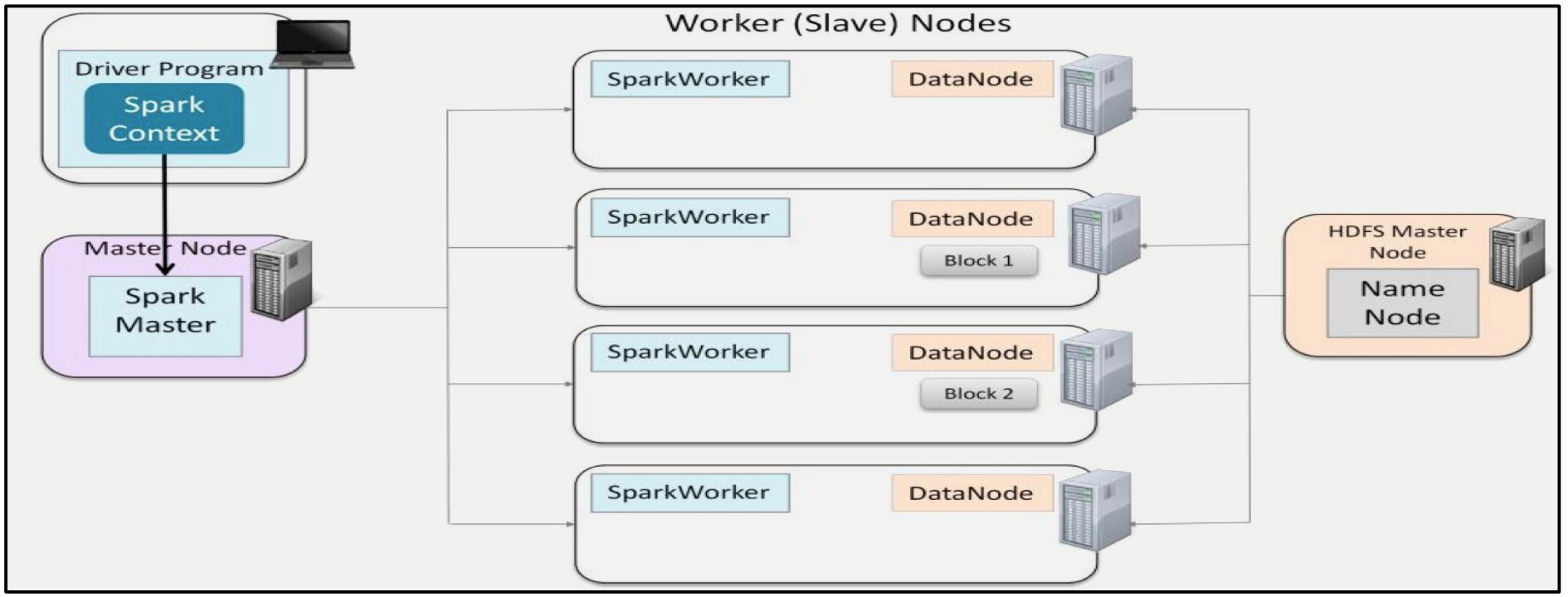
Running Spark On A Standalone Cluster (1 Of 5)



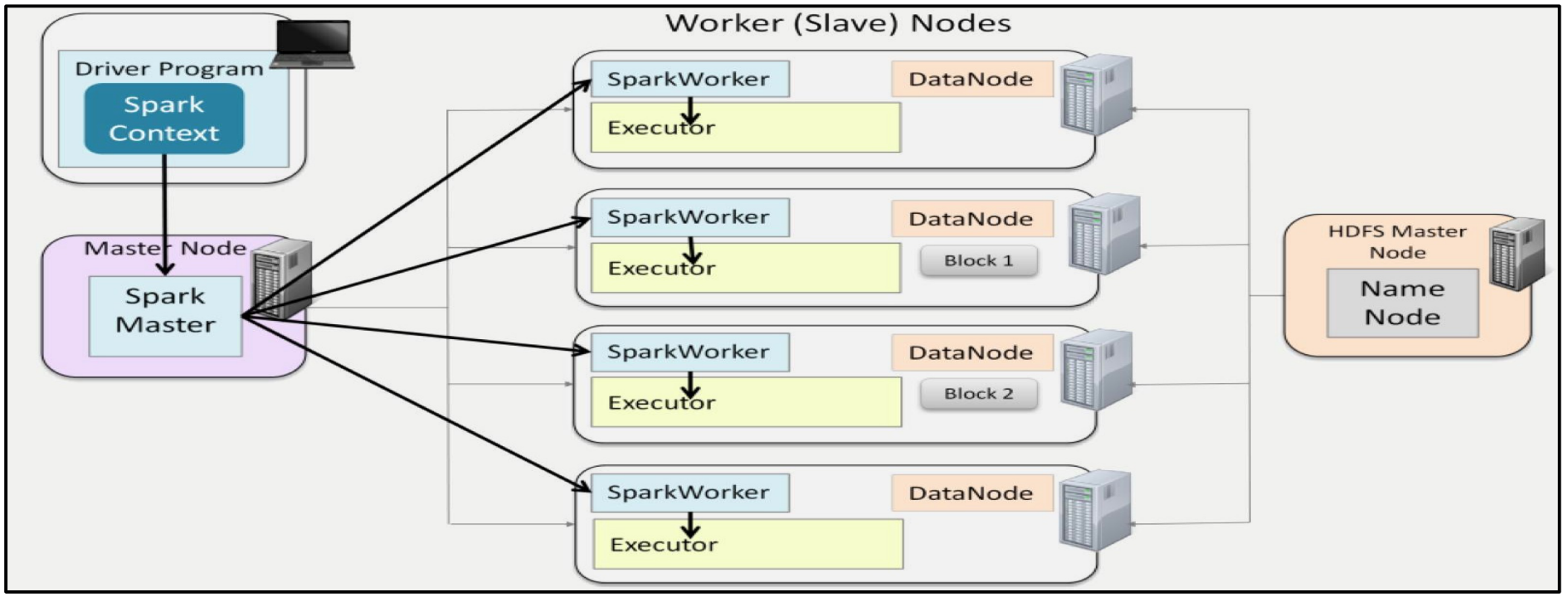
Running Spark On A Standalone Cluster (2 Of 5)



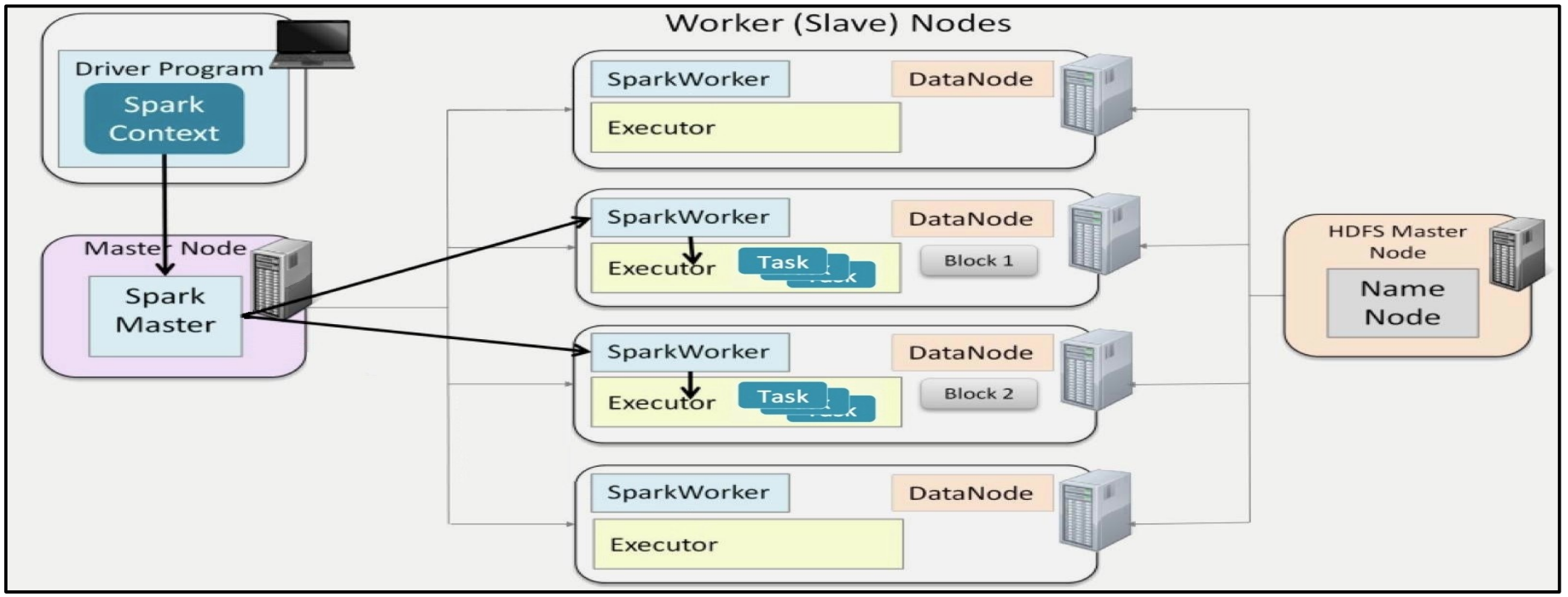
Running Spark On A Standalone Cluster (3 Of 5)



Running Spark On A Standalone Cluster (4 Of 5)



Running Spark On A Standalone Cluster (5 Of 5)



Spark Scheduling

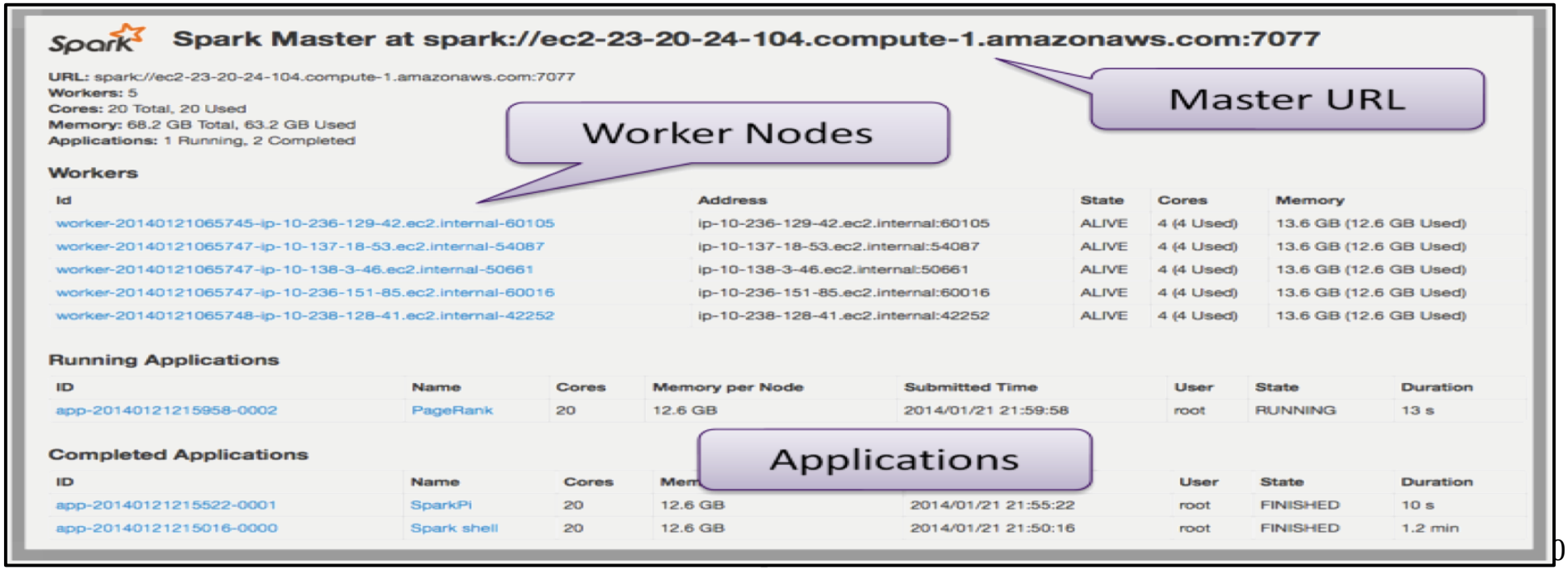
- Task scheduling within the application:
  - Managed by the Driver
  - Determines when tasks execute
- Executor / resource scheduling across nodes of the cluster:
  - Managed by the Spark Master
  - Determines where tasks execute

Chapter Topics

- Overview
- A Spark Standalone Cluster
- The Spark Standalone Web UI
- Spark Deployment Options
- Summary
- Review
- References
- Hands-On Exercise: Running The Spark Shell On A Cluster

Spark Standalone Web UI (1 Of 2)

- Spark Standalone clusters offer a Web UI to monitor the cluster (as do YARN and Mesos) for each daemon
  - http://masternode:uiport
    - e.g. In our class environment, localhost:18080
    - Note, 18080 is the port to access the UI
    - 7077 is the value assigned to SparkContext's MASTER property to access the master



Spark Standalone Web UI (2 Of 2)

- Spark's Standalone Web UI provides layered access to various aspects of the Spark environment
  - The Master URL (to which the MASTER variable is set)
  - A list of currently available worker nodes, any of which if selected takes you to ...
    - A list of applications currently running on that worker node, any of which if selected takes you to ...
      - An application detail page consisting of
        - The name and ID of the application
        - The number of cores and amount of memory dedicated to the application (across the cluster)
        - A list of currently running executors, any of which if selected takes you to ...
          - Details about the executor such as
            - The number of CPU cores allocated to it
            - How much memory it is using
            - Its current status

Hands-On

- Exit any open Spark shells
- Fire up the internal browser built into your Virtual Machine
- Let's try to access the Spark Master
  - http://localhost:18080
- Let's try to access a Spark Worker
  - http://localhost:18081
- Let's try to access an application
  - http://localhost:4040
- Nothing works, because we are not yet running any Spark daemons

Spark Local Mode Startup (Hands-On)

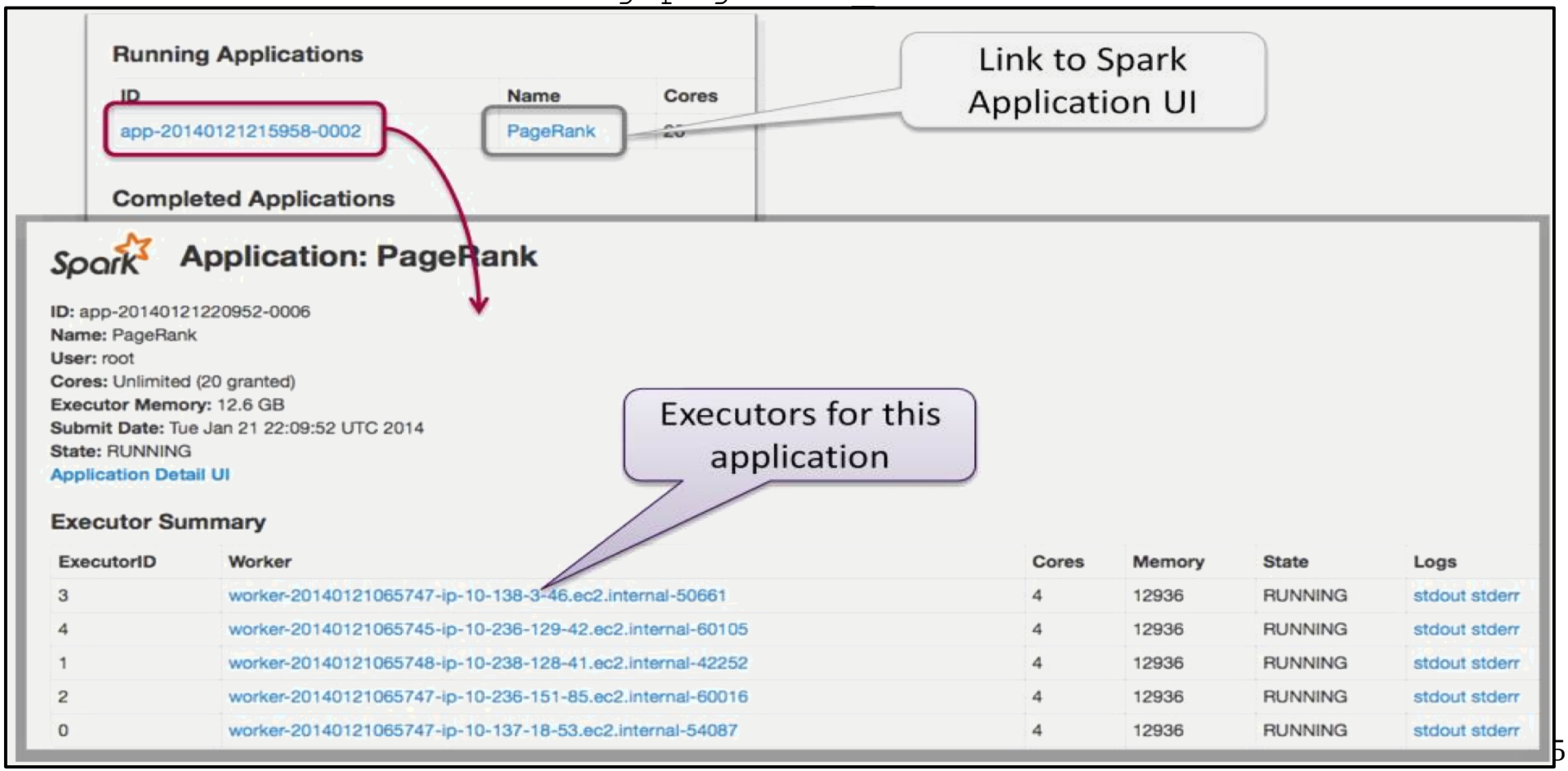
- To launch the master
  - http://localhost:18080
  - sudo service spark-master start
  - http://localhost:18080
- To launch a worker
  - http://localhost:18081
  - sudo service spark-worker start
  - http://localhost:18081
- To launch an application (Local mode)
  - http://localhost:4040
  - pyspark
    - Read the spark shell boot messages
  - http://localhost:4040
  - Execute 1+ Spark actions
  - http://localhost:4040
  - Exit the shell
- Shutdown both daemons
  - sudo service spark-master stop
  - sudo service spark-worker stop

Spark Standalone Mode Startup (Hands-On)

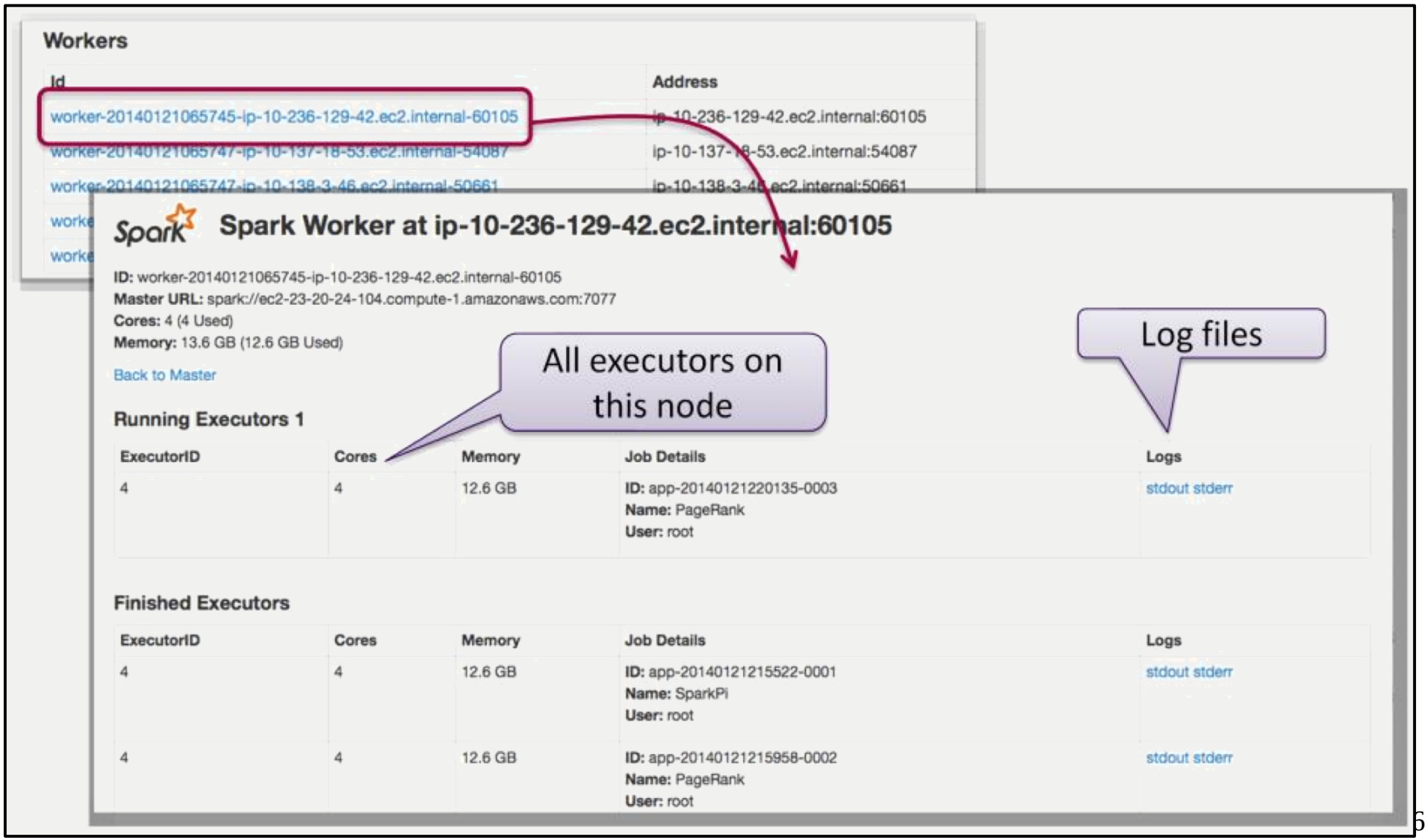
- To launch the master
  - http://localhost:18080
  - sudo service spark-master start
  - http://localhost:18080
- To launch a worker
  - http://localhost:18081
  - sudo service spark-worker start
  - http://localhost:18081
- To launch an application (Standalone mode)
  - http://localhost:4040
  - MASTER=spark://localhost:7077 pyspark
    - Read the spark client boot messages
  - Execute 1+ Spark actions
  - http://localhost:4040
  - exit
- To launch an application (YARN)
  - MASTER=yarn-cluster pyspark
- To launch an application (Mesos)
  - MASTER=mesos://localhost:5050 pyspark

Spark Standalone Web UI: Application Overview

```
$ cd /usr/lib/spark
$ ./bin/run-example SparkPageRank \
      hdfs:/user/training/pagerank_data.txt 10 local
```



Spark Standalone Web UI: Worker Node Overview



Chapter Topics

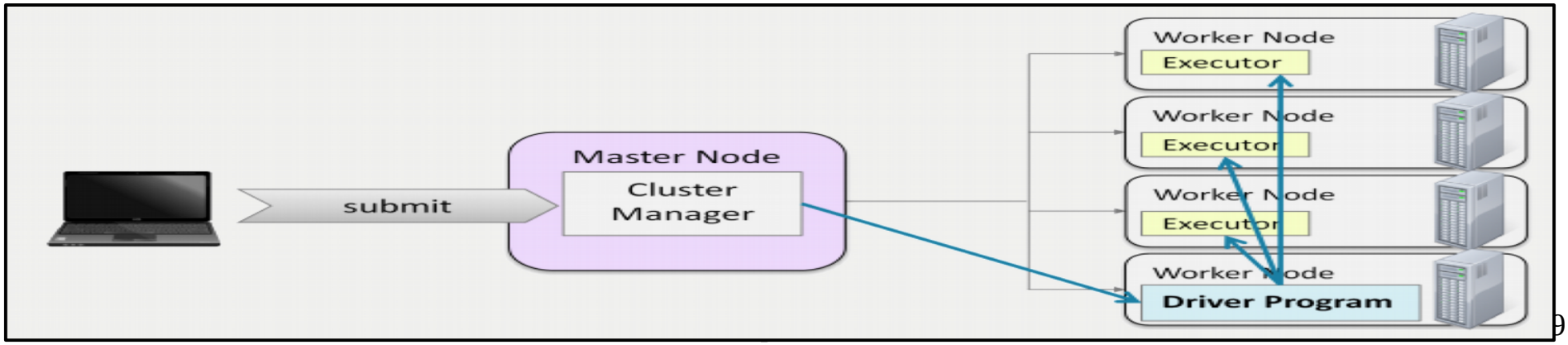
- Overview
- A Spark Standalone Cluster
- The Spark Standalone Web UI
- Spark Deployment Options
- Summary
- Review
- References
- Hands-On Exercise: Running The Spark Shell On A Cluster

Security Implications

- One important consequence of the architecture just outlined in earlier slides (known as Client Mode) is that it requires direct communication between the Driver, running outside of the cluster and the slave nodes running inside of the cluster
- This can be an issue in some environments
  - e.g. For security reasons
- So what's the alternative?

Client Mode And Cluster Mode

- However, it is also possible to run the driver program on a worker node in the cluster
  - Called "cluster" deployment mode
  - Offers the client a more secure environment in which to communicate with the workers
  - Also offers performance benefits when the client and worker reside on the same rack (e.g. Copying data to the workers)
  - Also the only choice when the client node lacks the horsepower or access to resources required to support the job locally



Installing A Spark Cluster (1 Of 2)

- Production cluster installation and administration are usually performed by a system administrator
  - Which is outside of the scope of this course
- But, developers should understand how the components of a cluster work together
- Developers often test first locally, then on a small test cluster

Installing A Spark Cluster (2 Of 2)

- Difficult:
  - Download and install Spark and HDFS directly from Apache
- Easier: CDH
  - Preferably CDH 5
  - Cloudera's Distribution, including Apache Hadoop
  - Includes HDFS, Spark API, Spark Standalone, and YARN
  - Includes many patches, backports, bug fixes
  - Includes useful tools from the Hadoop ecosystem
- Easiest: Cloudera Manager
  - Wizard based UI to install, configure, and manage a cluster
  - Included with Cloudera Express (free) or Cloudera Enterprise
  - Supports Spark deployment as Standalone or YARN

Setting Up A Spark Standalone Cluster On Amazon EC2

- Spark includes support to easily set up and manage a Spark Standalone cluster on Amazon Web Services EC2
  - Create your own AWS account
  - Use the spark-ec2 script to
    - /usr/lib/spark-1.0.0-bin-hadoop2/ec2/spark-ec2
    - Start, pause, and stop a cluster
    - Launch an application on the cluster
    - Specify regions, resource requirements, whether you are using standard or spot pricing, Spark version, and other options
- Uses distributed files stored on Amazon S3 (Simple Storage Service) rather than storing the files in HDFS
  - A distributed file system similar to HDFS, designed to work with MapReduce and Spark via the HDFS API
  - s3://path/to/file
- http://spark.apache.org/docs/latest/ec2-scripts.html
- http://wiki.apache.org/hadoop/AmazonS3

Chapter Topics

- Overview
- A Spark Standalone Cluster
- The Spark Standalone Web UI
- Spark Deployment Options
- Summary
- Review
- References
- Hands-On Exercise: Running The Spark Shell On A Cluster

Summary

- Spark is designed to run on a cluster
  - Spark includes a basic cluster management platform called Spark Standalone
  - Can also run on Hadoop YARN and Mesos
- The master distributes tasks to individual workers in the cluster
  - Tasks run in executors
    - JVMs running on worker nodes
- Spark clusters work closely with HDFS
  - Tasks are assigned to workers where the data is physically stored when possible
- Spark Standalone provides a UI for monitoring the cluster
  - YARN and Mesos have their own UIs

Chapter Topics

- Overview
- A Spark Standalone Cluster
- The Spark Standalone Web UI
- Spark Deployment Options
- Summary
- Review
- References
- Hands-On Exercise: Running The Spark Shell On A Cluster

Review

- Which of the following are benefits of running Spark on a cluster (Choose all that apply)
  - Security
  - The ability to process large quantities of data efficiently
  - Fault tolerance
  - Scalability

Review Answer

- Which of the following are benefits of running Spark on a cluster (Choose all that apply)
  - Security
  - Parallelism
  - Fault tolerance
  - Scalability

Review

- What are the names of the 2 Spark standalone daemons? How many are there of each? What are the primary responsibilities of each?

Review Answer

- What are the names of the 2 Spark standalone daemons? How many are there of each? What are the primary responsibilities of each?
  - Spark Master
    - One per cluster
    - Manages applications, distributes individual tasks to Spark Workers
  - Spark Worker
    - One per worker node
    - Starts and monitors Executors for applications

Review

- What is the primary difference between a Task in the MapReduce architecture and an Executor in the Spark architecture?

Review Answer

- What is the primary difference between a Task in the MapReduce architecture and an Executor in the Spark architecture?
  - In MapReduce, there is one JVM for each Task
  - In Spark, a single Executor might run several Tasks, avoiding the cost of having to repeatedly instantiate new JVMs

Review

- When you run locally by default, how many worker threads are created?

Review Answer

- When you run locally by default, how many worker threads are created?
  - One per core

Review

- When running in Cluster mode, what must you do to make sure that competing applications don't "freeze" one another?

Review Answer

- When running in Cluster mode, what must you do to make sure that competing applications don't "freeze" one another?
  - Limit the number of cores required to run the application

Chapter Topics

- Overview
- A Spark Standalone Cluster
- The Spark Standalone Web UI
- Spark Deployment Options
- Summary
- Review
- References
- Hands-On Exercise: Running The Spark Shell On A Cluster

References

- http://ampcamp.berkeley.edu/exercises-strata-conf-2013/launching-a-cluster.html
- http://docs.sigmoidanalytics.com/index.php/Installing_Spark_and_Setting_Up_Your_Cluster

Chapter Topics

- Overview
- A Spark Standalone Cluster
- The Spark Standalone Web UI
- Spark Deployment Options
- Summary
- Review
- References
- Hands-On Exercise: Running The Spark Shell On A Cluster

Hands-On Exercise

- Running Spark On A Cluster
  - Start the Spark Standalone daemons (Spark Master and Spark Worker) on your local machine (a simulated Spark Standalone cluster)
  - Run the Spark Shell on the cluster
  - View the Spark Standalone Ul
- Please refer to the Hands-On Exercise Manual