|  |
| --- |
|  |
| **ΜΔΑ-210 Cloud Computing with Hadoop** |
| **Spark deployment on Apache Hadoop Yarn cluster in cloud IAAS okeanos.grnet.gr** |
| **ORESTIS FOTIADIS - ME1825**  **GIORGOS KARGAS - ME1825** |
| Semester Group Assignment |
| **20/02/2019** |

Table of Contents

[Introduction 2](#_Toc1603727)

[1 Cluster configuration 3](#_Toc1603728)

[2 Starting the Hadoop Yarn cluster 5](#_Toc1603729)

[3 Monitoring Spark and Yarn via UI 7](#_Toc1603730)

[4 Submitting an application in cluster mode 7](#_Toc1603731)

# Introduction

This objective of this assignment was to successfully deploy, configure and operate Apache Spark using Apache Hadoop Yarn for application, scheduling and resource management on a three-node virtual machine cluster in [okeanos.grnet.gr](https://okeanos.grnet.gr).

This document outlines the cluster’s architecture, Spark and Yarn configuration and instructions on how to connect, submit and monitor spark deployments.

# Cluster configuration

The cluster consists of three VMs hosted and managed in “Cyclades”. Cyclades is the Compute and Network Service of “Okeanos”, the IaaS infrastructure of the Greek Research and Technology Network[[1]](#footnote-1). Cyclades provides a web UI to provision, manage and scale virtual machines.

As shown in Figure 1‑1, one VM (snf-859854) plays the role of the Node Master and two VMs (snf-859920 and snf-859921) act as workers (node1 and node2). Node Master is connected to the internet via a virtual network card (also provisioned by Okeanos’s infrastructure). The Node Master and the worker VMs are joined in a Virtual Network through which they are accessible via SSH.

The user account which operates the cluster is hadoopmanager. Hadoop and Spark binaries exist under that user’s home directory, in the paths: home/hadoopmanager/hadoop and home/hadoopmanager/spark.

The following versions of Hadoop and Spark are currently installed:

* Hadoop 3.2.0
* Spark 2.4

The following versions of Java and Scala languages are currently installed:

* OpenJDK Runtime Environment (build 1.8.0\_191-8u191-b12-2ubuntu0.18.04.1-b12)
* Scala 2.11

The cluster has the following memory allocation configured for Yarn (values in megabytes):

yarn.nodemanager.resource.memory-mb 6144

yarn.scheduler.maximum-allocation-mb 6144

yarn.scheduler.minimum-allocation-mb 512

yarn.app.mapreduce.am.resource.mb 512

mapreduce.map.memory.mb 512

mapreduce.reduce.memory.mb 512

Spark is configured with the following memory allocation:

spark.driver.memory 1024m

spark.yarn.am.memory 512m

spark.executor.memory 1024m

A close up of a map

Description automatically generated

Figure ‑: Cluster architecture

Table 1‑1 presents the hardware configuration of the VMs that form the cluster. Table 1‑2 outlines the OS setup in each VM. Effectively, all VMs run Ubuntu server and the Node Master also has the ubuntu-desktop package installed to allow it to act as the cluster’s “driver”.

|  |  |  |  |
| --- | --- | --- | --- |
| VM | CPUs | RAM | HDD |
| Node master | 8 | 8192 | 60Gb |
| Node 1 | 8 | 8192 | 60Gb |
| Node 2 | 8 | 8192 | 60Gb |

Table 1‑1: VM Hardware configuration

|  |  |
| --- | --- |
| VM | OS |
| Node master | Ubuntu 18.04.2 LTS with ubuntu-desktop package |
| Node 1 | Ubuntu 18.04.2 LTS |
| Node 2 | Ubuntu 18.04.2 LTS |

Table 1‑2: VM Operating Systems

# Starting the Hadoop Yarn cluster

The following steps outline the process to successfully start the Yarn and Spark in the cluster:

1. SSH into Node Manager’s port 22 using the supplied credentials.
2. Start Spark and Yarn:

$ start-all.sh

1. To verify that the processes have successfully started:

$ jps

13347 NodeManager

13140 ResourceManager

12069 NameNode

13912 Jps

12798 SecondaryNameNode

1. To start spark interactive shell, run:

$ spark-shell

hadoopmanager@snf-859854:~$ spark-shell

2019-02-20 08:15:22 WARN Utils:66 - Your hostname, snf-859854 resolves to a loopback address: 127.0.1.1; using 83.212.97.118 instead (on interface eth1)

2019-02-20 08:15:23 WARN Utils:66 - Set SPARK\_LOCAL\_IP if you need to bind to another address

Setting default log level to "WARN".

To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).

2019-02-20 08:15:49 WARN Client:66 - Neither spark.yarn.jars nor spark.yarn.archive is set, falling back to uploading libraries under SPARK\_HOME.

Spark context Web UI available at http://snf-859854.vm.okeanos.grnet.gr:4040

Spark context available as 'sc' (master = yarn, app id = application\_1550643306133\_0001).

Spark session available as 'spark'.

Welcome to

\_\_\_\_ \_\_

/ \_\_/\_\_ \_\_\_ \_\_\_\_\_/ /\_\_

\_\ \/ \_ \/ \_ `/ \_\_/ '\_/

/\_\_\_/ .\_\_/\\_,\_/\_/ /\_/\\_\ version 2.4.0

/\_/

Using Scala version 2.11.12 (OpenJDK 64-Bit Server VM, Java 1.8.0\_191)

Type in expressions to have them evaluated.

Type :help for more information.

1. To start the Spark History Server, run:

$ $SPARK\_HOME/sbin/start-history-server.sh

starting org.apache.spark.deploy.history.HistoryServer, logging to /home/hadoopmanager/spark/logs/spark-hadoopmanager-org.apache.spark.deploy.history.HistoryServer-1-snf-859854.out

1. jps should now show the history server process running:

13347 NodeManager

13140 ResourceManager

12069 NameNode

**18731 HistoryServer**

22507 Jps

12798 SecondaryNameNode

# Monitoring Spark and Yarn via UI

The following resources are available via web UI. They are accessible by remoting on Node Manager and opening a browser pointing to the related URL.

|  |  |
| --- | --- |
| Resource | URL |
| HDFS Overview | <http://10.0.0.2:9870> |
| Hadoop Yarn | <http://10.0.0.2:8088> |
| Spark History Server | <http://10.0.0.2:18080> |
| Spark Web UI | <http://10.0.0.2:37231> |

# Submitting an application in cluster mode

After starting the cluster, an example application can be submitted in cluster mode. For this purpose we use a Pi calculator jar which is bundled with Spark. To submit in cluster mode we enter the following command, where the memory allocation for the application can also be specified:

$ spark-submit --class org.apache.spark.examples.SparkPi \

--master yarn \

--deploy-mode cluster \

--driver-memory 1g \

--executor-memory 1g \

--executor-cores 1 \

/home/hadoopmanager/spark/examples/jars/spark-examples\_2.11-2.4.0.jar \

10

While the application is being executed, its state can be monitored via the Hadoop Yarn UI and the Spark web UI.

Once the application has finished executing, the output can be inspected in the stdout log file which is accessible, besides directly in HDFS, in the Hadoop Yarn UI:

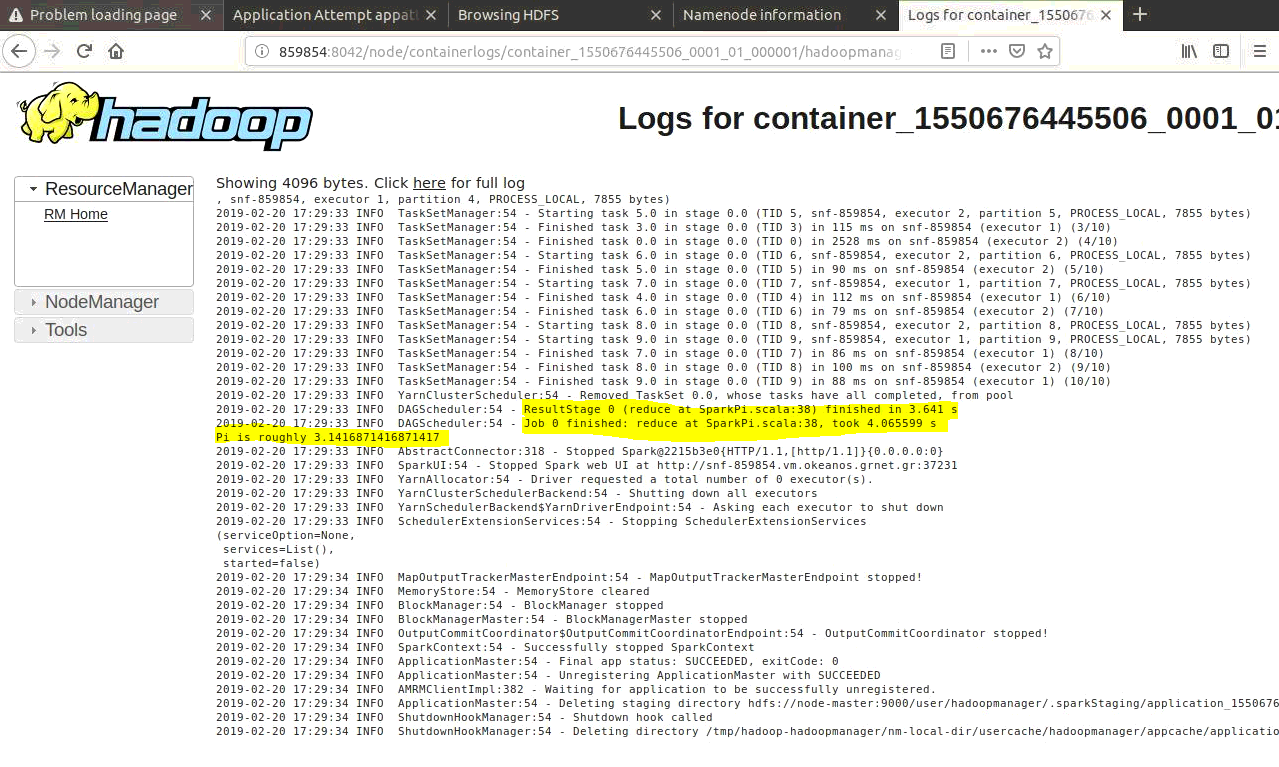


Figure ‑

1. The Greek Research and Technology Network or GRNET is the national research and education network of Greece. For further information about Okeanos: <https://okeanos.grnet.gr/about/what/>. [↑](#footnote-ref-1)