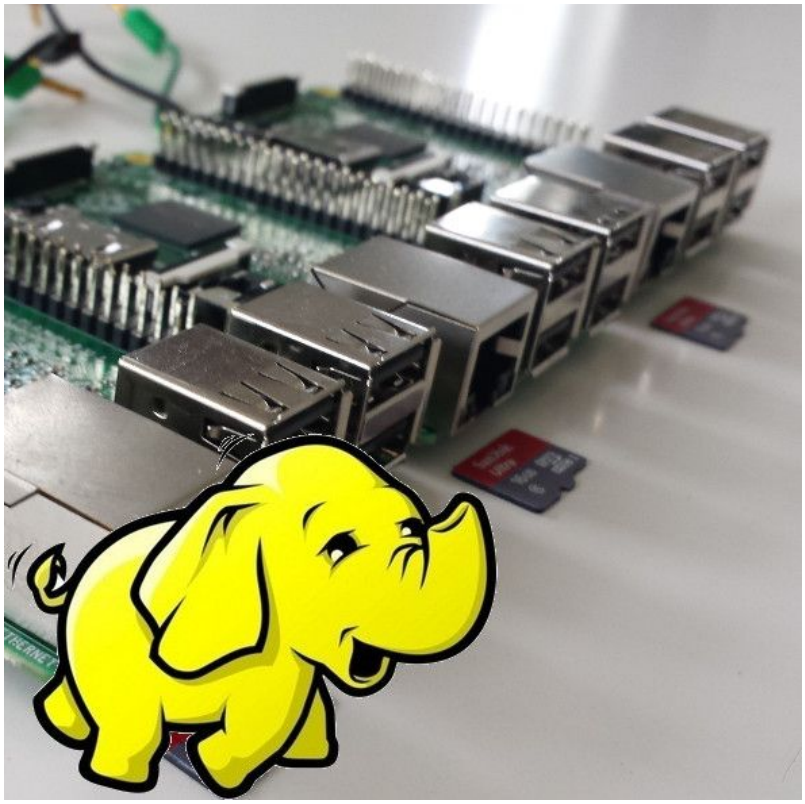
CS 441: Running Hadoop on Raspberry Pi's using Container Orchestration



Due: November 16, 2017

Team Members:

Matthew Grider
Robert Hull
Kaeyan Jones
Daniel Kloza
Michael Koutsostamatis
Ibrahiem Mohammad
Priyam Patel
Nikolay Zakharov

Abstract

Welcome to the Map/Reduce with Kubernetes on OctaPi. If you're into big data/distributed systems and a Raspberry Pi enthusiast this is the perfect project for you. If not, there is still a lot you can learn here. Let's start with our objective.

Objective

Our objective is to create a cluster of 8 Raspberry Pi's that can be used as master/slave system. Afterwards, the master will employ Kubernetes or a different containerized application management system to orchestrate docker containers and run mapreduce on them.

Planning Process

Initially we began with getting our equipment together. Of course it would take some time for everyone to gather all of their materials. Thus we took about 3 or 4 Pi's together and began with installing Raspbian OS and following the guide provided:
https://projects.raspberrypi.org/en/projects/build-an-octapi

Once we completed this, we know that our nodes can communicate, and that the master can communicate the with the slaves. What he had to figure out now is how to distribute the mapreduce work onto the Pi's.

Our plan was to install kubernetes and then use that to create a few docker containers and test our mapreduce, which was already written. Simple in theory, but as you'll see, we ran into a few problems along the way.

Hardware

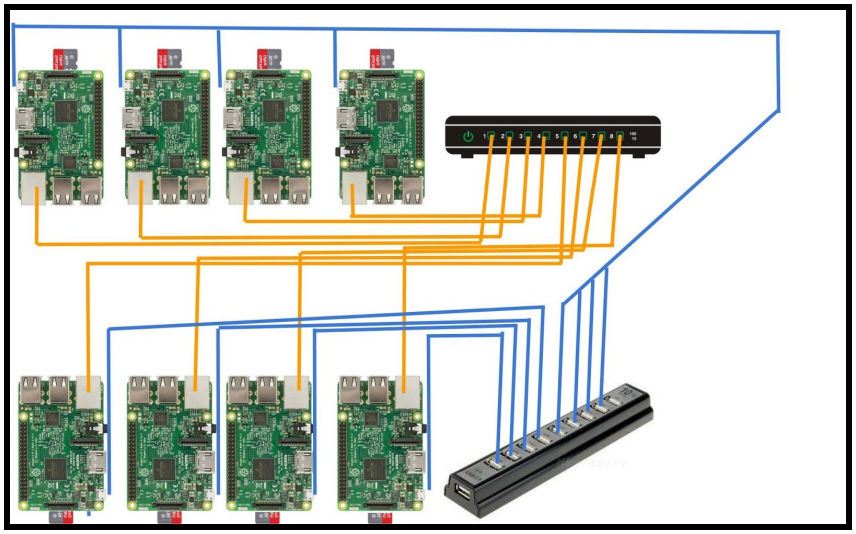
8 - Raspberry Pi 3 Model B
8 - Micro SD cards (16gb minimum storage size)
8 - Ethernet cables
8 - Micro USB cables
1 - TP-Link 8 Port switcher with power adapter
8 - 10 watt USB wall adapters

Optional:
1 - HDMI cable
1 - TV/monitor to set up each Raspberry Pi

1 - mouse
1 - keyboard

Hardware Assembly

Assembly is pretty straight forward, connect each Raspberry Pi to the network switch via an ethernet cable plug in the Micro USB cable into the Micro USB adapter on the Raspberry Pi. For further assistance, refer to the diagram below where the orange lines depict ethernet cables and blue lines depict USB cables.



Implementation Process

To begin with, we had to install Kubernetes to handle the container orchestration that was needed for our project. Initially, we tried to merge YARN and Kubernetes together, and that did not build correctly because the architectures were incompatible. Afterwards, trying to install a Kubernetes pod with a Docker container inside, holding a Hadoop image. The problems we ran into trying to install Kubernetes that every time we try to run it to create a container for a Hadoop image, it would only work with AMD x64/x86 architectures and give us an error shown below, because the Pi has ARM architecture. This was an issue we struggled with for a long time.

Error: standard_init_linux.go:185: exec user process caused "exec format error"
(Arm vs x86 issue)

We wanted to make sure this wasn't a problem with Kubernetes itself, so using Hypriot, we were able to create a Kubernetes cluster. We then tried to run Hadoop MapReduce onto this cluster and realized that this issue was going to be more difficult than we thought to get Kubernetes and Hadoop to work together.
Therefore, we turned to Docker Swarm, another nautical-themed container orchestration tool, which would orchestrate the container that Docker created an image on, and assign resources to it, hoping for the best. However, Kubernetes was a very cool tool to work with because it allowed for the automation of container creation, and this was interesting to see. It was also an interesting finding to be aware of issues with incompatible architectures, because in the future, we will always consider this before making decisions to incorporate multiple programs together.

Our next idea was to try an apache spark image, but that also ended up giving us the same error.

Another attempt was trying to install Hortonworks onto the Kubernetes cluster, which also did not work due to the incompatible architectures.

It was at this point we realized that there was no way to bypass the issue of trying to run x86 software on an ARM device. We then turned our attention to address this.

To help solve this issue of the incompatible architectures, we decided to look into various softwares that would emulate an x86/64 architecture on the ARM device. One of these tools which is Wine (*Wine Is Not an Emulator)*, an open-source compatibility layer that allows programs that can run on Windows to run on Unix-like environments.

Wine:
https://www.winehq.org/

By the time we realized that we would have to write our own Dockerfile that would deploy Hadoop on an ARM device, it was very late and we decided to simply focus on having the cluster and dockerswarm container management setup when presenting to the professor.