End of Studies Master's Project

Master's Thesis

Simulation of a Kubernetes Cluster with Validation in Real Conditions

Author
Théo Larue

Evaluator
Surname NAME

February 24th to August 7th 2020

Contents

Introduction

In the early stages of application development, organizations used to run their services on physical servers. With this direct approach came a few problematics: resources allocation, maintainability, scalability for exemple. Developers then went on with virtualized machines to run their services regardless of physical infrasctucture, which then led to the concept of containers.

Containers can be thought of as lightweight virtual machines. Containerized applications are applications running inside of a container, completely independently of the underlying physical infrastructure. There are many benefits to this : separating the development from deployment, portability, easy resource allocation, breaking large services into smaller micro-services or support of continuous integration tools, to cite a few.

Kubernetes[4] allows the automation of the process of deploying, maintaining and scaling containerized applications. It is an open source platform originally designed by Google and now maintained and developed by the Cloud Native Computing Foundation. It is industry grade and widely used by web services, big or small, for its ease of use and its reliability.

However, eventhough it allows the automation of technical tasks that used to be done manually, it doesn't take away other problematics like scheduling. Scheduling is the NP-complete problem of allocating containers on physical machines : one has to take into account resource availability, latency between servers, storage locality, etc. Kubernetes schedulers have to handle this difficult task of allocating resources in the most efficient way.

Thanks to the architecture of Kubernetes which is built around a central component, the API server, schedulers can be developed independently in order to fit particular needs. As a matter of fact, one universal scheduler that would efficiently operate on any infrastructure will most likely never exist and organizations often develop their own scheduler to better fit their needs.

This raises the question of scheduler development. Developing a scheduler implies being able to test its performances throughout the development process, however, testing in real conditions is time consuming and expensive. Organizations can either have enough resources to cover these costs, or test their scheduler against a simulation.

Kubernetes cluster simulations is an open problem and is the subject of this master project. Our approach relies on the Batsim[1] infrastructure simulator, which is itself built upon Simgrid[2]. Batsim is currently mostly used to simulate HPC infrastructures but was designed to be able to simulate any kind of infrastructure and therefore is theoretically able to simulate any Kubernetes cluster, moreover, Kubernetes was designed to run services but is capable of handling High-Performance Computing[3]. This project aims at adapting Batsim so it can evaluate Kuberenetes schedulers.

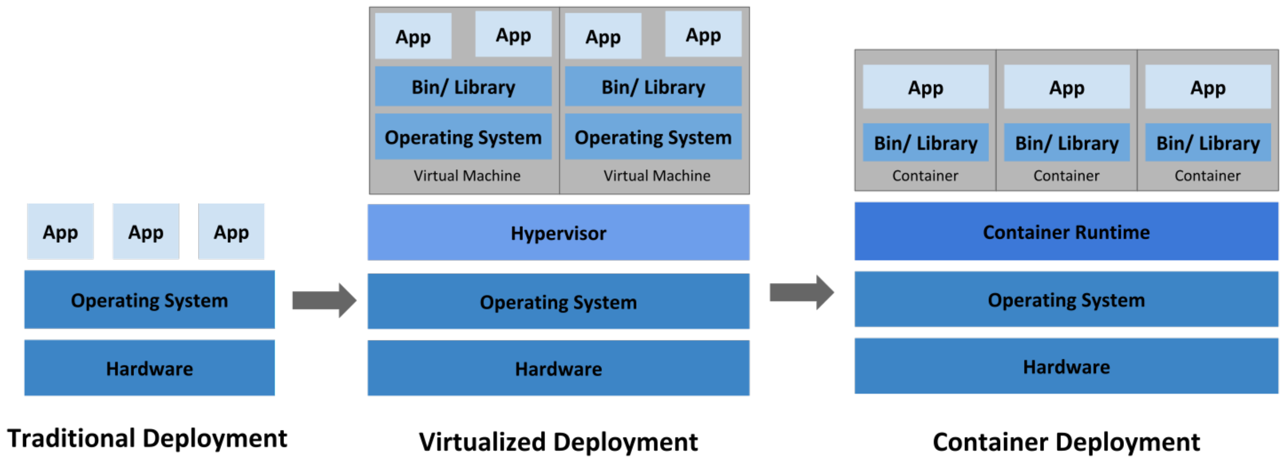

Figure 1.1: Evolution of application deployment.
Source: *https://kubernetes.io/docs/concepts/overview/what-is-kubernetes/*

Problematic

The goal of this project is to design and implement Batkube, which will be an interface between Batsim and Kubernetes schedulers. With this interface, we want to compare Batsim results gainst data from a real Kubernetes cluster, given HPC workloads.

2.1 Kubernetes concepts

2.2 Batsim concepts

2.2.1 Limitations

2.3 Translation

2.4 Synchronization

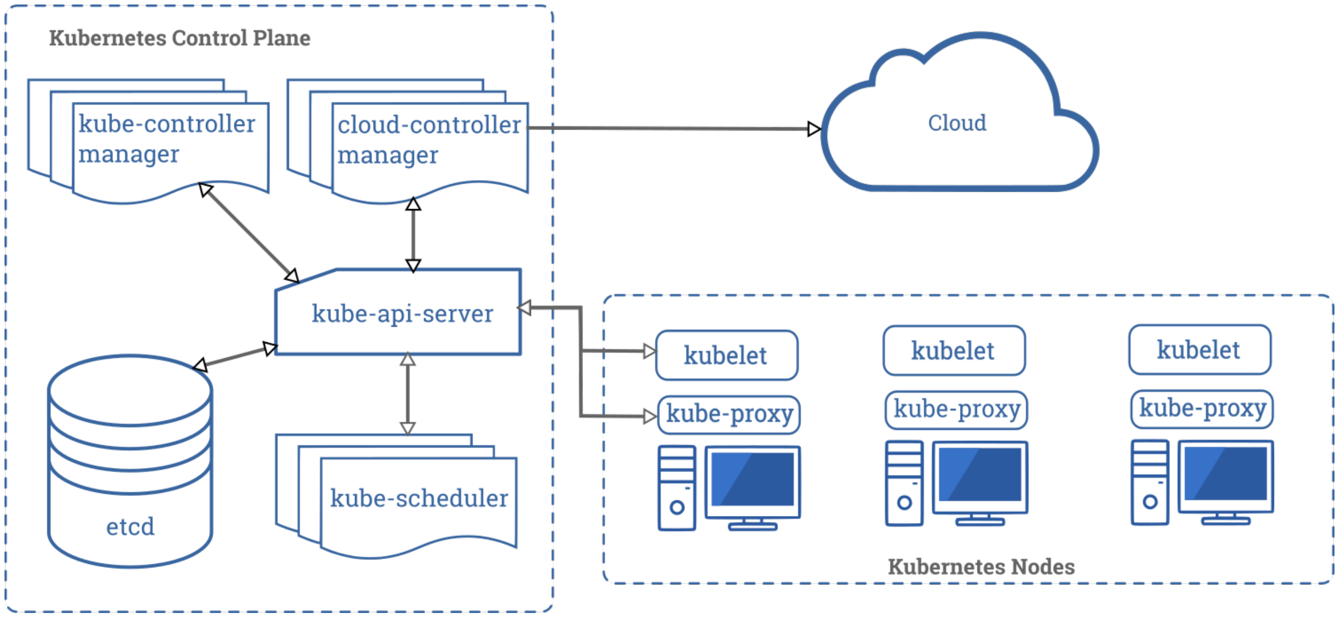

Figure 2.1: Components of Kubernetes
Source: *https://kubernetes.io/docs/concepts/overview/components/*

State of the art

3.1 Kubernetes schedulers

3.2 Simulating Kubernetes

3.3 Technical challenges

Implementation

Batkube architecture

Evaluation

Conclusion