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[8] 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[3] infrastructure simulator, which is itself built upon Simgrid[11]. 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[7]. This project aims at adapting Batsim so it can evaluate Kuberenetes schedulers.

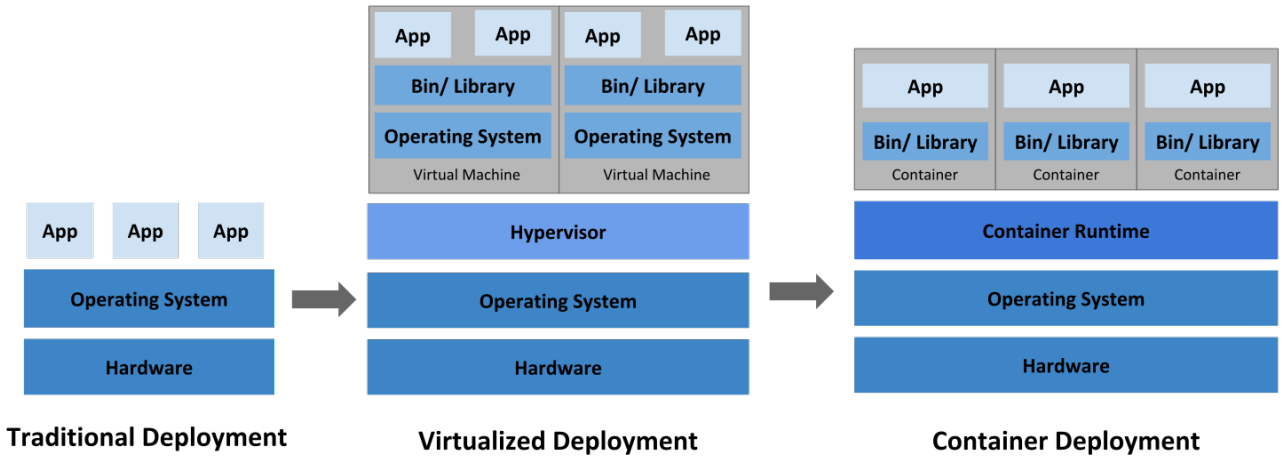

Figure 1.1: Evolution of application deployment.
Source: *https://kubernetes.io/docs/concepts/overview/what-is-kubernetes/*

Problematic

2.1 Objectives

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.

2.2 Kubernetes concepts

2.3 Batsim concepts

2.3.1 Limitations

2.4 Translation

2.5 Synchronization

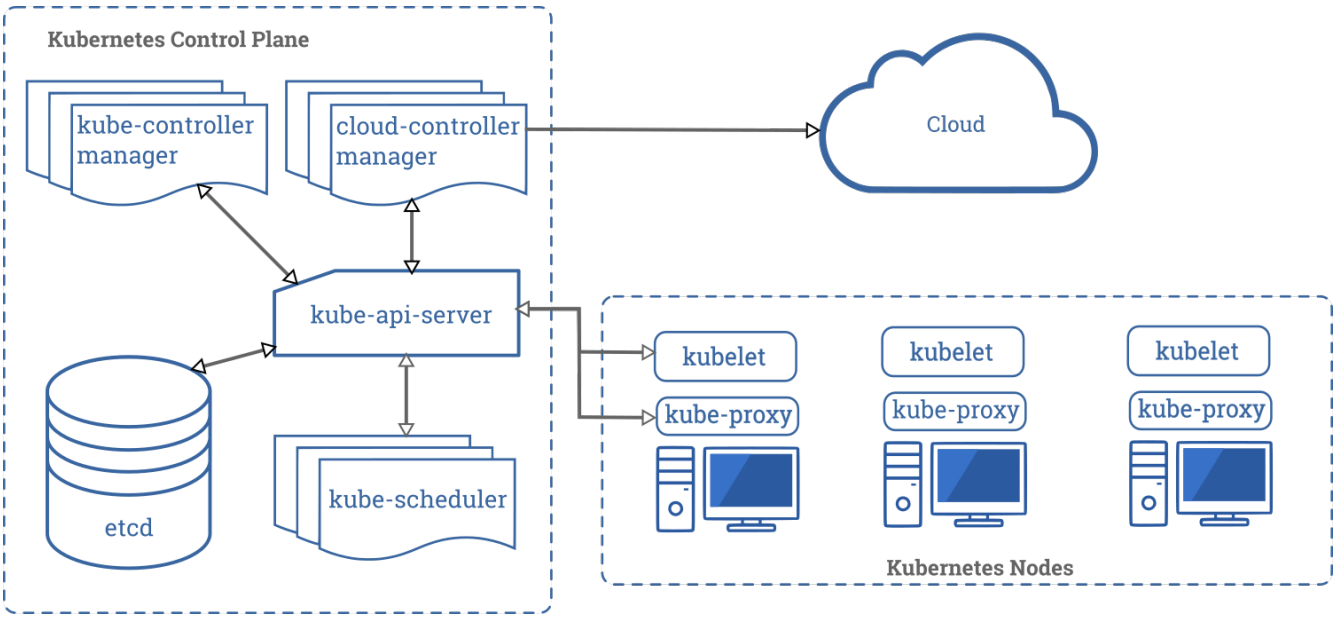

Figure 2.1: Components of Kubernetes
Source: *https://kubernetes.io/docs/concepts/overview/components/*

State of the art

3.1 Simulating Kubernetes

Simulating the Kubernetes ecosystem is a difficult task that hasn't been done many times. In fact, we have been able to find only two references on the subject. One is a from an internship, and the other does not have any open source release.

To our knowledge, there isn't any other project about adapting Simgrid to Kubernetes.

3.1.1 k8s-cluster-simulator

k8s-cluster-simulator[4] is an internship project written in Go. It aims at simulating a Kubernetes clusters in order to evaluate schedulers with basic workloads, with as little changes to schedulers implementation as possible.

The user can specify resource usage for some pods in a yaml config file (cpu, memory, gpu) and submit them through an interface. The scheduler is also submitted through an interface, and that makes the code not exactly plug and play : the scheduler has to comply with the interface to be evaluated.

This is a simple project useful for understanding some Kubernetes core concepts and technical details.

3.1.2 JoySim simulator from JD.com

This is a project led by the company JD.com, presented in the **KubeCon + CloudNativeCon North America 2019**[1]. It looks like a very complete, fully fledged Kubernetes cluster simulator, however it is not (yet) open-source : "Planning to work with CNCF SIG-Scheduling for an open source release".

3.2 Kubernetes schedulers

Gathering knowledge on the scheduling ecosystem in Kubernetes is a first step in deciding what direction to take with Batkube. It came out that the most useful resources we could find were the very simple schedulers that were written as examples or tutorials about writing custom schedulers. These basic schedulers made the process of writing proof of concepts much easier.

3.2.1 Industry grade schedulers

Kubernetes scheduler

kube-scheduler is the de facto scheduler for any Kubernetes cluster (at least with the native Kubernetes distribution) and can be found in Kubernetes repository[9].

It is a generic scheduler and very effective in most cases.

kube-batch

kube-batch[5] is a batch scheduler developed by the Kubernetes Scheduling SIG[10].

It has proven reliable on an industrial size and serves as a base for other scheduling projects (e.g. Volcano[12]).

Poseidon

3.2.2 Example schedulers

These schedulers helped a great deal in understanding how a scheduler interacts with the Kubernetes api-server, thanks to their simplicity.

Bash scheduler

The bashScheduler[2] is a tiny project created demonstrating a very basic implementation of a random scheduler using only bash, so as to break down exchanges between the api server and the scheduler with simple http requests.

It has served as a base upon which the first POC was created.

Random scheduler

This project is taken from a tutorial[13] redacted by Banzai Cloud to guide us through the implementation of a custom scheduler using the go client[6] from kubernetes. Like the bashScheduler, it randomly binds pods on available nodes. Although, it takes things to the next level by using the go client thus introducing the concept of kubernetes informers and using https instead of plain http.

The second POC was built using this scheduler as a base.

Implementation

4.1 Technical challenges

4.2 Batkube architecture

Evaluation

Conclusion

Bibliography

[1] *A Toolkit for Simulating Kubernetes Scheduling at Scale - Yuan Chen, JD.com.* URL: https://kccncna19.sched.com/event/UaVk/a-toolkit-for-simulating-kubernetes-scheduling-at-scale-yuan-chen-jdcom (visited on 04/21/2020).

[2] *bashScheduler repository on Github.* URL: https://github.com/rothgar/bashScheduler.

[3] *Batsim docs.* URL: https://batsim.readthedocs.io/en/latest/.

[4] *k8s-cluster-simulator: A simulator for evaluating Kubernetes schedulers.* URL: https://tech.preferred.jp/en/blog/k8s-cluster-simulator-release/ (visited on 04/21/2020).

[5] *kube-batch repository on Github.* URL: https://github.com/kubernetes-sigs/kube-batch.

[6] *Kubernetes Go Client repository on Github.* URL: https://github.com/kubernetes/client-go.

[7] *Kubernetes Meets High-Performance Computing.* URL: https://kubernetes.io/blog/2017/08/kubernetes-meets-high-performance/ (visited on 04/21/2020).

[8] *Kubernetes official website.* URL: https://kubernetes.io/.

[9] *Kubernetes repository on Github.* URL: https://github.com/kubernetes/kubernetes.

[10] *Kubernetes Scheduling SIG.* URL: https://github.com/kubernetes/community/blob/master/sig-scheduling/README.md.

[11] *Simgrid official website.* URL: https://simgrid.frama.io/.

[12] *Volcano repository on Github.* URL: https://github.com/volcano-sh/volcano.

[13] *Writing custom Kubernetes schedulers.* URL: https://banzaicloud.com/blog/k8s-custom-scheduler/ (visited on 04/21/2020).