Kubernetes Implementation into HPC

Livermore Computing

August 10, 2017

Jessica Gomes, Jonathan Victorino, Harry Yee

HPC CEA Interns
Meet Team Kubes

Jessica Gomes
University of Illinois at Urbana-Champaign

Jonathan Victorino
Stanford University

Harry Yee
UCLA
Objectives

I. Setup a Kubernetes cluster using Docker containers

II. Run LINPACK benchmark tests comparing Kubernetes and bare metal cluster

III. Determine performance overhead of running the cluster in Kubernetes

IV. Automate Kubernetes builds and deployments using Puppet
Our Project

1) Create docker containers with OpenMPI
2) Launch the containers on Kubernetes
3) Run benchmark tests and gather data
Docker

- World’s leading software container platform
- **Container**: stand-alone package that includes everything needed to run it
- Easily create images for containers using Dockerfiles
- Use Docker Hub to automatically update the containers
What is Kubernetes?

- Open-source system for automating deployment, scaling, and management of containerized applications
- Schedule and deploy any number of Docker container replicas onto a node cluster
Kubernetes Container Visualization

**The old way:** Applications on host

```
App  App
App  App
Libraries
Kernel
```

- Heavyweight, non-portable
- Relies on OS package manager

**The new way:** Deploy containers

```
App  App
Libraries  Libraries
App  App
Libraries  Libraries
Kernel
```

- Small and fast, portable
- Uses OS-level virtualization
Kubernetes Architectural Overview
Installation Process

Docker Installation Script

Kubernetes Packages Installation Script

Puppet Manifest File
.yaml Files

Executable File

```
apiVersion: v1
kind: Service
metadata:
  name: mpi-master
spec:
  ports:
  - port: 22
  - port: 30001
```

Executable File

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: mpi-worker
spec:
  replicas: 5
  template:
    metadata:
      labels:
      # Unique key of the Deployment instance
      name: mpi-worker
    spec:
      # 3 Pods should exist at all times.
      # Apply this label to pods and default
      # Deployment label selector to this value
      # mpi-master-deployment
    containers:
      - name: mpi-worker
        image: centos:7
        ports:
          - containerPort: 22
```

LINPACK Benchmarks

<table>
<thead>
<tr>
<th></th>
<th>Time (seconds)</th>
<th>G-flops</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bare Metal</td>
<td>15.39</td>
<td>3.47E+02</td>
</tr>
<tr>
<td>Kubernetes</td>
<td>1796.654</td>
<td>2.99E+00</td>
</tr>
</tbody>
</table>
Conclusion

- Kubernetes
  - Large performance overhead on clusters
  - Extra setup steps
  - Not ideal for HPC

- Use bare metal cluster

- After further testing, the bottleneck was found to be due to a networking issue

- Time and G-flops were comparable for containers running on the same node to the bare metal configuration
Future Plans

- Use Puppet to automate builds and deployments
- Automatically ssh from worker to master
- Run more Benchmark tests on Kubernetes cluster
- Find out how to fix the bottleneck and improve communication between containers