National College of Ireland

Configuration Manual

MSc Research Project
Cloud Computing

Himanshi Painuly
Student ID: x22143220

School of Computing
National College of Ireland

Supervisor: Prof Jitendra Kumar Sharma

National College of Ireland
Project Submission Sheet
School of Computing

Student Name:	Himanshi Painuly
Student ID:	x22143220
Programme:	Cloud Computing
Year:	2024
Module:	MSc Research Project
Supervisor:	Prof Jitendra Kumar Sharma
Submission Due Date:	12/08/2024
Project Title:	Configuration Manual
Word Count:	539
Page Count:	7

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature:	
Date:	12th August 2024

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST:

Attach a completed copy of this sheet to each project (including multiple copies).	☐
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	☐
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	☐

Assignments that are submitted to the Programme Coordinator office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Configuration Manual

Himanshi Painuly
x22143220

1 Introduction

The setup of a Kubernetes cluster on an Ubuntu server is explained in detail in this article. The technologies and instructions required to assemble the cluster will be covered in Part 2. We will look at how to use custom schedulers in addition to the default scheduler in the next section so that we may test both at the same time. To learn more about how our custom scheduler works, we'll dig into the code. Lastly, we will go through configuring Prometheus and Node Exporter, two monitoring tools.

2 Tools and Details of System

Tools and Technologies	Details/Edition
Cluster Creation Platform	AWS EC2
OS version	UBUNTU SERVER 22.04 LTS
Application Container	USER DEFINED MICROSERVICES
Containerization Orch Software	KUBERNETES 1.28.4
Software for Containerization	DOCKER 24.0.7
Monitoring Tools	PROMETHUS,NODE EXPORTER AND GRAFANA
Number of CPUs for Worker and Master	2 FOR EACH
Storage	16GB FOR MASTER AND 8GB FOR WORKERS
Coding Language	GO LANG
File comminutor for Pods and Nodes	YAML

Figure 1: System stack

3 Clustering using Kubernetes

In order to make use of cloud computing, I am using AWS EC2 services for my study. Due to its enhanced compatibility with the most recent Kubernetes features and improvements, as well as its updated kernel support and more recent software versions, I decide to use Ubuntu Server 22.04.

3.1 Node Creation

- Step1: Set unique hostnames for the master and node machines.

```
sudo hostnamectl set-hostname "k8s-master"
exec bash  && sudo bash
sudo hostnamectl set-hostname "k8s-node1"
exec bash  && sudo bash
sudo hostnamectl set-hostname "k8s-node2"
exec bash  && sudo bash
```

- Step2:Add the IP addresses and hostnames of all nodes to the hosts file on each machine.

```
cat <<EOF | sudo tee -a /etc/hosts
172.31.5.116 k8s-master
172.31.7.18 k8s-node1
172.31.0.104 k8s-node2
EOF
```

- Step3: Now Turn off the swap space.

```
sudo swapoff -a
sudo sed -i '/ swap / s/^\(.*\)$/#\1/g' /etc/fstab
```

- Step4: Refresh the system package list and install the required packages for Container-D on all nodes.

  Configure required modules

```
cat <<EOF | sudo tee /etc/modules-load.d/k8s.conf
overlay
br_netfilter
EOF

sudo modprobe overlay
sudo modprobe br_netfilter
```

```
cat <<EOF | sudo tee /etc/sysctl.d/k8s.conf
net.bridge.bridge-nf-call-iptables  = 1
net.bridge.bridge-nf-call-ip6tables = 1
net.ipv4.ip_forward                 = 1
EOF
```

Apply the sysctl parameters to the current running environment without needing to reboot.

```
sudo sysctl --system
```

- Step5: Now Install Docker In All Nodes

```
# Add Docker's official GPG key:
sudo apt-get update -y
sudo apt-get install ca-certificates curl gnupg -y
sudo install -m 0755 -d /etc/apt/keyrings
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --
dearmor -o /etc/apt/keyrings/docker.gpg
sudo chmod a+r /etc/apt/keyrings/docker.gpg

# Add the repository to Apt sources:
echo \
  "deb [arch=$(dpkg --print-architecture) signed-
by=/etc/apt/keyrings/docker.gpg] https://download.docker.com/linux/ubuntu
\
  $(. /etc/os-release && echo "$VERSION_CODENAME") stable" | \
  sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
sudo apt-get update -y
sudo apt-get install docker-ce docker-ce-cli containerd.io docker-buildx-
plugin docker-compose-plugin -y
```

Configure Containerd To Start Using systemd as group

```
containerd config default | sudo tee /etc/containerd/config.toml
>/dev/null 2>&1
sudo sed -i 's/SystemdCgroup \= false/SystemdCgroup \= true/g'
/etc/containerd/config.toml
```

- Step6: Start the ContainerD services and verify their status.

```
sudo systemctl start containerd
sudo systemctl enable containerd
sudo systemctl restart containerd
sudo systemctl daemon-reload
sudo systemctl status containerd.service --no-pager
```

- Step7: Next, install kubectl, kubeadm, and Kubernetes CNI.

```
sudo apt-get update -y
sudo apt-get install -y apt-transport-https ca-certificates curl
curl -fsSL https://pkgs.k8s.io/core:/stable:/v1.28/deb/Release.key | sudo
gpg --dearmor -o /etc/apt/keyrings/kubernetes-apt-keyring.gpg
echo 'deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg]
https://pkgs.k8s.io/core:/stable:/v1.28/deb/ /' | sudo tee
/etc/apt/sources.list.d/kubernetes.list
sudo apt-get update -y
sudo apt-get install -y kubelet kubeadm kubectl kubernetes-cni
sudo apt-mark hold kubelet kubeadm kubectl && sudo apt-mark hold docker
kubectl version --client && docker --version
```

And check their status of installation

```
sudo systemctl daemon-reload
sudo systemctl start kubelet
sudo systemctl enable kubelet.service
sudo systemctl status kubelet.service --no-pager
```

- Step8: On the Master Node, switch to the root user, initialize kubeadm, and set the Kubernetes directory path.

```
sudo su -

kubeadm init

mkdir -p $HOME/.kube
sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
sudo chown $(id -u):$(id -g) $HOME/.kube/config
```

- Step9: On Node1 and Node2, execute the token command as the root user.

```
You should now deploy a pod network to the cluster.
Run "kubectl apply -f [podnetwork].yaml" with one of the options listed at:
  https://kubernetes.io/docs/concepts/cluster-administration/addons/

Then you can join any number of worker nodes by running the following on each as root:

kubeadm join 172.31.27.184:6443 --token 86tce3.5b3gkwvx1olluxtf \
        --discovery-token-ca-cert-hash sha256:47836fb0fd831853babd6fb5f326a9472a91d3d3b3f9d0776791ac0a64d5ef85
root@k8s-master:~# exot
Command 'exot' not found, did you mean:
  command 'exo' from snap exoscale-cli (v1.22.2)
  command 'emot' from deb ruby-emot (0.0.4-2)
See 'snap info <snapname>' for additional versions.
```

- Step10: Install The Flannel pod network network

```
kubectl apply -f https://github.com/flannel-
io/flannel/releases/latest/download/kube-flannel.yml
```

- Step11: Now, let's verify: On the Master Node, use a non-root user.

```
NAME          STATUS   ROLES           AGE     VERSION
k8-node1      Ready    <none>          3d16h   v1.28.12
k8-node2      Ready    <none>          3d16h   v1.28.12
k8s-master    Ready    control-plane   3d16h   v1.28.12
```

4 Prometheus, Grafana and Node Exporter Installation

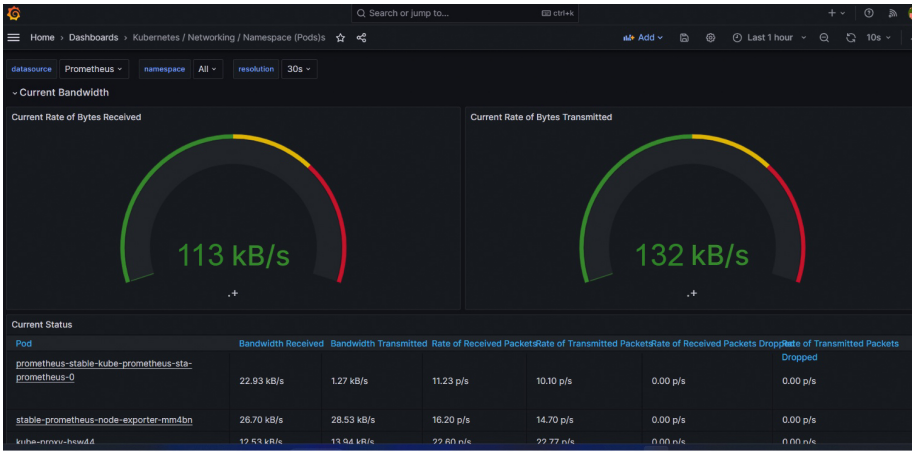
To install Prometheus, Grafana, and Node Exporter, we use Helm. Helm simplifies this process by packaging your configuration files into a single, reusable unit, which automates the development, packaging, configuration, and deployment of Kubernetes applications.

```
helm repo add prometheus-community https://prometheus-
community.github.io/helm-charts
helm repo update prometheus-community
helm search repo prometheus-community
kubectl create namespace prometheus
helm install stable prometheus-community/kube-prometheus-stack -n
prometheus
```

```
kubectl get pods -n prometheus
kubectl get svc -n prometheus
kubectl edit service/stable-grafana -n prometheus
kubectl edit service/stable-kube-prometheus-sta-prometheus -n prometheus
kubectl edit service/stable-kube-prometheus-sta-alertmanager -n prometheus
kubectl edit service/stable-kube-state-metrics -n prometheus
kubectl edit service/stable-prometheus-node-exporter -n prometheus
kubectl edit service/stable-kube-prometheus-sta-operator -n prometheus
kubectl edit service/alertmanager-operated -n prometheus
kubectl edit service/prometheus-operated -n prometheus
kubectl describe secret stable-kube-prometheus-sta-prometheus -n prometheus
kubectl get svc -n prometheus
```

Finally, we will be able to view the Prometheus dashboard with all EC2 instances listed as targets, as shown below.

For Grafana, after accessing the website and logging in, you need to import the Kubernetes monitoring dashboard using the import ID 22855. This will display a dashboard with metrics related to the scheduler, organized by different namespaces to evaluate various parameters.



5 Custom Scheduler Implementation

Following Commands needs to execute our custom scheduler, and our microservice will execute our scheduler using deployment file,

Firstly we need to import our custom scheduler,
https://github.com/Peacemaker1999/k8s/blob/master/quick/latency-aware-scheduler

Followings are the deployment file and scheduler code,

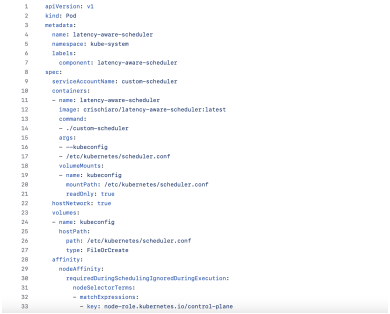


Figure 2: Latency Aware Deployment Script

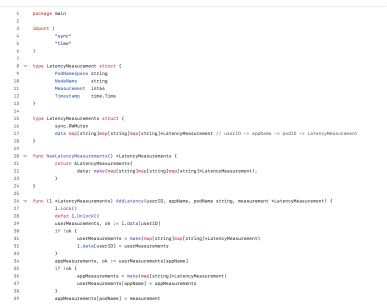


Figure 3: Deployment Script

References

Documentation — Grafana Labs — grafana.com (n.d.). `https://grafana.com/docs/`. [Accessed 08-12-2023].

Getting Started - Cloud Computing Tutorials for Building on AWS — aws.amazon.com (n.d.). `https://aws.amazon.com/getting-started/`. [Accessed 08-12-2023].

Poniszewska-Maranda, A. and Czechowska, E. (2021). Kubernetes cluster for automating software production environment, *Sensors* **21**: 1910.

Prometheus (n.d.). Overview — Prometheus — prometheus.io, `https://prometheus.io/docs/introduction/overview/`. [Accessed 08-12-2023].