**Cài đặt Control Plane và Worker node:**

Dưới đây là hướng dẫn **đầy đủ và được cập nhật 2024**, có thể áp dụng cho Ubuntu 20.04, 22.04 (Jammy) và mới hơn:

**1. Cài đặt công cụ hỗ trợ HTTPS cho APT**

sudo apt-get update

sudo apt-get install -y apt-transport-https ca-certificates curl gpg

**2. Thêm khóa GPG và tạo source list đúng cách**

sudo mkdir -p /etc/apt/keyrings

curl -fsSL https://pkgs.k8s.io/core:/stable:/v1.29/deb/Release.key | gpg --dearmor | sudo tee /etc/apt/keyrings/kubernetes-apt-keyring.gpg > /dev/null

echo "deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg] https://pkgs.k8s.io/core:/stable:/v1.29/deb/ /" | sudo tee /etc/apt/sources.list.d/kubernetes.list > /dev/null

**3. Cập nhật và cài đặt Kubernetes:**

sudo apt-get update

sudo apt-get install -y kubelet kubeadm kubectl

sudo apt-mark hold kubelet kubeadm kubectl

**Kiểm tra sau khi cài đặt**

kubectl version --client

kubeadm version

kubelet –version

***Tiếp theo là triển khai Kubernetes cluster multi-node (1 control plane + 1 worker)***

***Bước 1: Cấu hình hệ thống (trên cả 2 VM Ubuntu)***

***Tắt swap****:*

sudo swapoff -a

sudo sed -i '/ swap / s/^/#/' /etc/fstab

Bật các kernel module cần thiết:

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

br\_netfilter

EOF

sudo modprobe br\_netfilter

***Cấu hình sysctl:***

cat <<EOF | sudo tee /etc/sysctl.d/k8s.conf

net.bridge.bridge-nf-call-iptables = 1

net.ipv4.ip\_forward = 1

net.bridge.bridge-nf-call-ip6tables = 1

EOF

sudo sysctl –system

***Bước 2: Cài container runtime (nếu chưa có)***

***Kubernetes cần một container runtime. Dùng containerd là ổn định và phổ biến nhất:***

sudo apt-get install -y containerd

sudo mkdir -p /etc/containerd

containerd config default | sudo tee /etc/containerd/config.toml

***Chỉnh cấu hình để SystemdCgroup = true:***

sudo nano /etc/containerd/config.toml

***Tìm dòng:***

[plugins."io.containerd.grpc.v1.cri".containerd.runtimes.runc.options]

***Và chỉnh:***

SystemdCgroup = true

***Sau đó:***

sudo systemctl restart containerd

sudo systemctl enable containerd

***Bước 3: Tạo cluster trên control plane node***

***Trên VM control plane:***

sudo kubeadm init --pod-network-cidr=10.244.0.0/16

***Dùng lệnh:*** kubeadm join ... dành cho worker. hoặc có thể lấy lại token này bằng lệnh:

sudo kubeadm token create --print-join-command

***Sau đó cấu hình kubectl:***

mkdir -p $HOME/.kube

sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config

sudo chown $(id -u):$(id -g) $HOME/.kube/config

***Bước 4: Cài Flannel CNI (hoặc Calico, Cilium, etc.)***

kubectl apply -f https://raw.githubusercontent.com/coreos/flannel/master/Documentation/kube-flannel.yml

***Bước 5: Thêm worker node vào cluster***

Trên node worker, dùng lệnh kubeadm join mà bạn đã nhận từ bước 3. Ví dụ:

sudo kubeadm join 192.168.1.100:6443 --token abcdef.0123456789abcdef \

--discovery-token-ca-cert-hash sha256:xxxxxxxxxxxxxxxxxxxx

***Kiểm tra cluster (trên control plane)***

kubectl get nodes

***Mặc định, kubernetes không cho phép chạy pod trên nodo control plane, để cho phép pod chạy trên node control-plane (dành cho môi trường học tập, lab, máy ảo test)***

***Chạy lệnh sau để xóa taint:***

kubectl taint nodes <node-name> node-role.kubernetes.io/control-plane-

***Ví dụ, nếu node bạn tên là control-plane, thì:***

kubectl taint nodes control-plane node-role.kubernetes.io/control-plane-

Dấu - ở cuối lệnh là để xóa taint.

**Cài đặt Ingress Control (NodePort) cho Kubernetes**

kubectl apply -f <https://raw.githubusercontent.com/kubernetes/ingress-nginx/controller-v1.10.0/deploy/static/provider/baremetal/deploy.yaml>

***Kiểm tra Ingress Control***

kubectl get svc -n ingress-nginx

Hoặc

kubectl get serice –all-namespaces

***Xác định ingressClass để cấu hình cho ingress service***

kubectl get ingressclass

***Kiểm tra cấu hình của một Ingress***

kubectl get ingress ingress-app1 -o yaml

kubectl get ingress rancher -n cattle-system -o yaml

***Kiểm tra logs Ingress***

kubectl logs -n ingress-nginx -l app.kubernetes.io/name=ingress-nginx

***Dùng forward để kiểm tra một ứng dụng trong pod với Cluster-IP:***

kubectl port-forward svc/cluster-ip-app1 18080:8080

Sau khi forword có thể kiểm tra ứng dụng thông qua localhost:18080

**Cài đặt cloudflared**

***1. Thêm repo Cloudflare***

sudo apt install -y curl gnupg lsb-release

curl -fsSL https://pkg.cloudflare.com/cloudflare-main.gpg | sudo tee /usr/share/keyrings/cloudflare-main.gpg >/dev/null

echo "deb [signed-by=/usr/share/keyrings/cloudflare-main.gpg] https://pkg.cloudflare.com/cloudflared $(lsb\_release -cs) main" | sudo tee /etc/apt/sources.list.d/cloudflared.list

***2. Cài đặt***

sudo apt update

sudo apt install cloudflared

***Đăng nhập vào Cloudflare và tạo Tunnel***

cloudflared login

***Lệnh này sẽ mở trình duyệt → bạn đăng nhập Cloudflare → chọn domain bạn muốn dùng.***

***Sau khi đăng nhập thành công, Cloudflare sẽ lưu chứng chỉ xác thực tại:***

~/.cloudflared/cert.pem

***Tạo tunnel mới***

cloudflared tunnel create my-tunnel

***Cloudflare sẽ tạo tunnel và lưu thông tin vào:***

~/.cloudflared/my-tunnel.json

***Chạy foreground để test:***

cloudflared tunnel run my-tunnel

***Hoặc cấu hình chạy dưới dạng systemd (background):***

sudo cloudflared service install

***Tạo một service kích hoạt tunnel cloudflare chạy nền***

touch /etc/systemd/system/cloudflared-tunnel.service

vim /etc/systemd/system/cloudflared-tunnel.service

*Nội dung file cloudflared-tunnel.service:*

[Unit]

Description=Cloudflare Tunnel

After=network.target

[Service]

Type=simple

ExecStart=/usr/local/bin/cloudflared tunnel --config /tung/cloudflare-tunnel-config.yml run

Restart=on-failure

User=phan

Environment=HOME=/home/phan

[Install]

WantedBy=multi-user.target

***Chạy lệnh để tiến hành thực thi “cloudflared-tunnel.service”, sau khi lệnh chạy có thể tắt terminal, khởi động máy, hoặc ngay khi tiến trình lỗi, nó sẽ tự khởi động lại “cloudflared-tunnel.service”:***

sudo systemctl daemon-reexec

sudo systemctl daemon-reload

sudo systemctl enable cloudflared-tunnel.service

sudo systemctl start cloudflared-tunnel.service

Để xem trạng thái của “***cloudflared-tunnel.service***”:

sudo systemctl status cloudflared-tunnel.service

***Chạy lại sử dụng lệnh:***

sudo systemctl daemon-reexec

sudo systemctl restart cloudflared-tunnel

**Cài đặt Jenkins trong kubernetes**

**Jenkins được cài với Helm chart.**

curl https://raw.githubusercontent.com/helm/helm/main/scripts/get-helm-3 | bash

**Kiểm tra:**

helm version

**Cài đặt Jenkins với helm**

helm install jenkins ./jenkins-chart -n jenkins

**Gỡ cài đặt Jenkins**

helm uninstall jenkins -n jenkins

***Chuẩn bị trước khi cài Jenkins vào kubernetes***

**1. Tạo namespaces cho jenkins**

kubectl create namespace jenkins

Xem secret đang tồn tại:

Kubectl get secret -n jenkins

**2. Tạo secret để tạo kubeconfig cho kubect trong pod jenkins thông qua secret:** *(secret đọc file, cung cấp cho tạo aws/credentials trong pod jenkins trên kubernetes, cho phép sử dụng aws thông qua thông tin đăng nhập lưu ở aws/ credentials cho jenkins. Trong lệnh dùng cần dùng “$HOME” thay vì “~” bới vì “~” là shell nên không được hiểu trong câu lệnh kubectl, còn “$HOME” là biến môi trường nên sẽ luôn đúng cho các trường hợp).*

kubectl create secret generic aws-credentials --from-file=credentials=$HOME/.aws/credentials -n jenkins

**3. Chuẩn bị image cho jenkins bằng docker, lưu trên docker hub.**

docker build -t vothinhuydt/jenkins:txu .

docker push vothinhuydt/jenkins:txu

**4. Tạo thư mục để mout vào jenkins lưu trữ dữ liệu lâu dài khi pod jenkins bị xóa tạo lại…**

sudo mkdir -p /mnt/data/jenkins

sudo chown -R 1000:1000 /mnt/data/jenkins

ls -ldn /mnt/data/jenkins

ls -ld /mnt/data/jenkins

**Cài đặt và kiểm tra docker:**

***Kiểm tra người dùng trong nhóm “docker”***

getent group docker

***Nếu chưa có nhóm “docker” có thể thêm nhóm:***

sudo groupadd docker

***Thêm người dùng võ vào nhóm “docker”***

sudo usermod -aG docker vo

***Hoặc thêm người dùng hiện tại vào nhóm “docker”***

sudo usermod -aG docker $USER

***Sau khi thêm user vào nhóm “docker” có thể chạy lệnh sau và kiểm tra lại nhóm “docker”***

newgrp docker

**1. Kiểm tra CoreDNS có hoạt động đúng không**

kubectl get pods -n kube-system -l k8s-app=kube-dns -o wide

Bạn nên thấy 2 pod coredns-\* đang Running.

**2. Kiểm tra logs của CoreDNS**

kubectl logs -n kube-system -l k8s-app=kube-dns

Tìm xem có dòng lỗi nào liên quan jenkins.jenkins.svc.cluster.local hoặc các truy vấn bị từ chối.

**3. Kiểm tra lại resolv.conf của Pod curl-test**

kubectl exec -n jenkins curl-test -- cat /etc/resolv.conf

Output chuẩn nên là:

search jenkins.svc.cluster.local svc.cluster.local cluster.local

nameserver 10.96.0.10

options ndots:5

**Sau đó kiểm tra lại DNS:**

nslookup jenkins.jenkins.svc.cluster.local

curl -v <http://jenkins.jenkins.svc.cluster.local:8080>

**Tạo secret gắn cho pod kaniko sử dụng cho đăng nhập aws và push code lên ecr:**

kubectl create secret generic aws-creds-kaniko --from-literal=AWS\_ACCESS\_KEY\_ID=AKIATCKANMZM53PY4HWK --from-literal=AWS\_SECRET\_ACCESS\_KEY=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 --from-literal=AWS\_REGION=ap-southeast-1 -n jenkins