

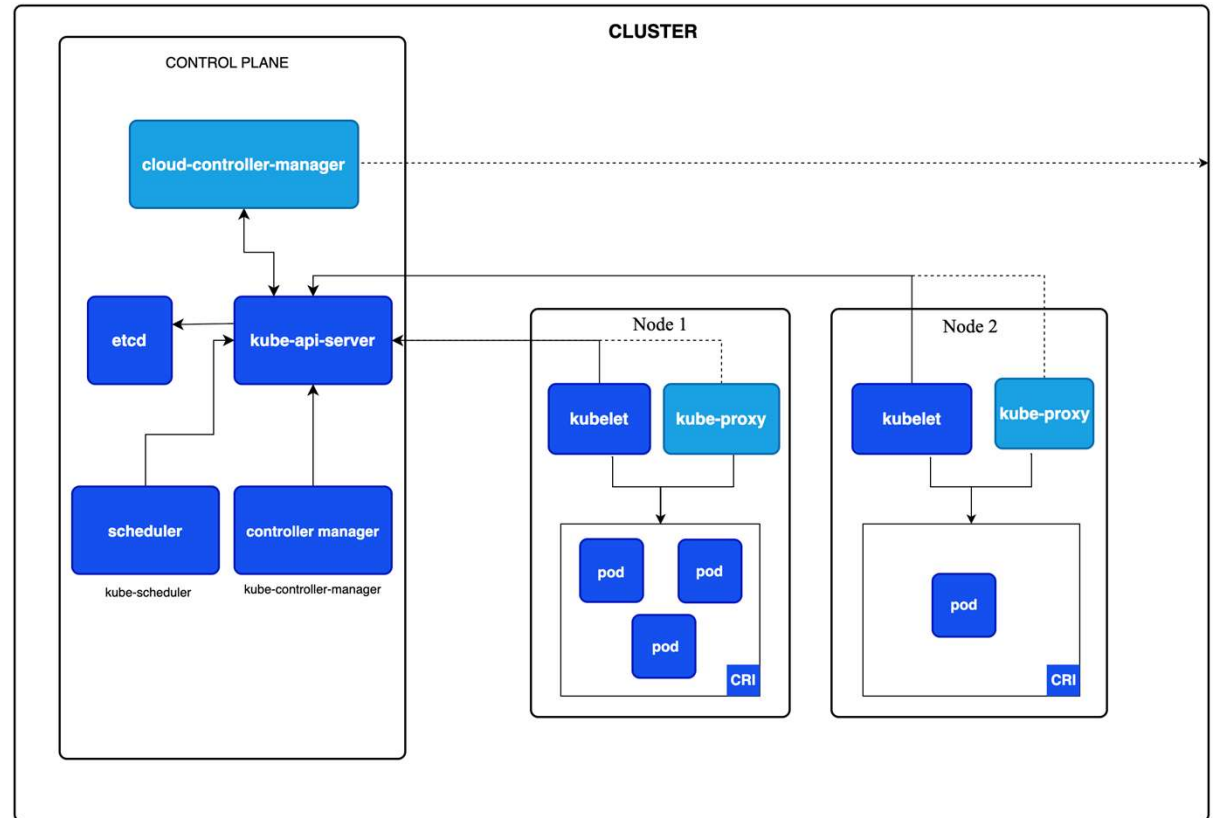
Cluster Architecture

K8s part 1

All rights reserved by OpsDev

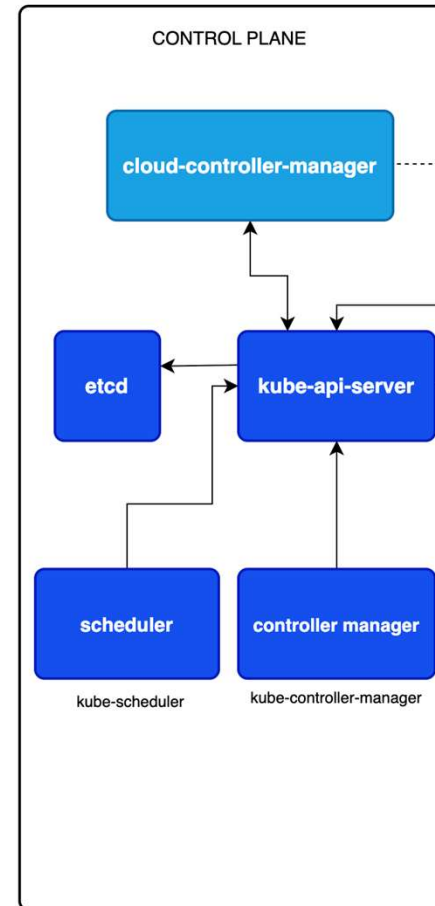
Cluster Architecture

1. Control plane -> ทำหน้าที่เหมือน vCenter ควบคุม resources ต่างๆใน cluster สามารถทำ HA ได้
2. Node -> ทำหน้าที่เหมือน ESX host มีหน้าที่รัน workload ตามที่ Control plane สั่ง



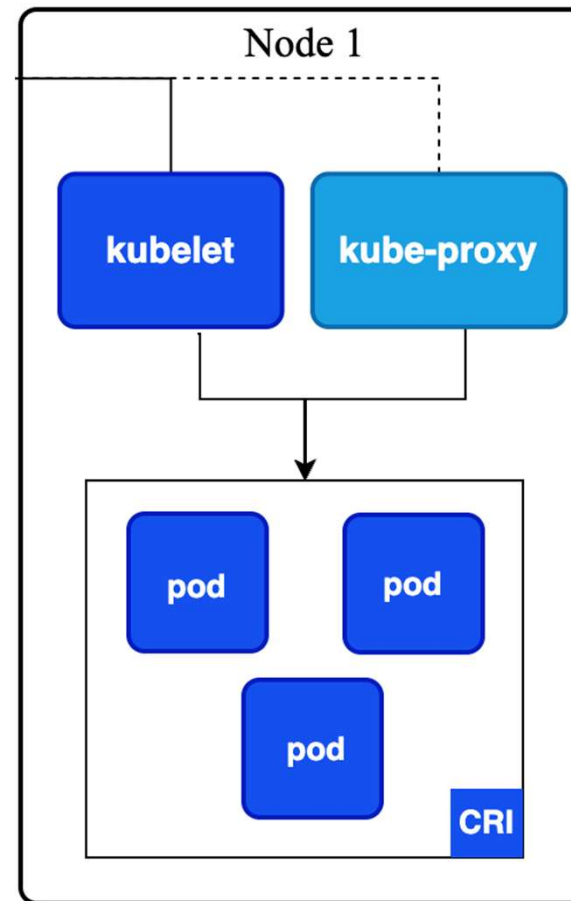
Control plane components

- kube-control-manager -> ทำหน้าที่ควบคุมทรัพยากรต่างๆใน cluster
- kube-schedule -> ทำหน้าที่จัดสรรทรัพยากรต่างๆใน cluster
- kube-api-server -> ทำหน้าที่ตัวกลางติดต่อสื่อสารทั้งหมดใน cluster
- etcd -> เป็นฐานข้อมูลทำหน้าที่เก็บข้อมูลต่างๆใน cluster



Node components

- kubelet -> เป็น agent สื่อสารกับ control plan
- kube-proxy -> เป็นตัวเชื่อม traffic ที่คุยกันภายใน cluster
- CRI -> เป็นตัวกลางระหว่าง kubelet ใช้งาน container runtime



Metric server

ใช้ Monitor resource ของ k8s

Install -> <https://github.com/kubernetes-sigs/metrics-server/releases>

Option

kubectl patch deployment metrics-server -n kube-system \

--type='json' \

-p='[{"op": "add", "path": "/spec/template/spec/containers/0/args/-", "value": "--kubelet-insecure-tls"}]'

```
controlplane:~$ k get po -n kube-system
```

NAME	READY	STATUS	RESTARTS	AGE
calico-kube-controllers-fdf5f5495-vxpw5	1/1	Running	2 (22m ago)	9d
canal-4xxhw	2/2	Running	2 (22m ago)	9d
canal-wlt99	2/2	Running	2 (22m ago)	9d
coredns-6ff97d97f9-4wljj	1/1	Running	1 (22m ago)	9d
coredns-6ff97d97f9-wmpcb	1/1	Running	1 (22m ago)	9d
etcd-controlplane	1/1	Running	3 (22m ago)	9d
kube-apiserver-controlplane	1/1	Running	3 (22m ago)	9d
kube-controller-manager-controlplane	1/1	Running	2 (22m ago)	9d
kube-proxy-d8lhg	1/1	Running	1 (22m ago)	9d
kube-proxy-tx14k	1/1	Running	2 (22m ago)	9d
kube-scheduler-controlplane	1/1	Running	2 (22m ago)	9d
metrics-server-56fb9549f4-qdqqb	1/1	Running	0	11m

```
controlplane:~$ k top nodes
```

NAME	CPU(cores)	CPU(%)	MEMORY(bytes)	MEMORY(%)
controlplane	74m	7%	1096Mi	53%
node01	22m	2%	773Mi	42%

```
controlplane:~$ k top po multi-con-5f9f4d498c-d5r1f
```

NAME	CPU(cores)	MEMORY(bytes)
multi-con-5f9f4d498c-d5r1f	0m	2Mi

CMD

kubectl cluster-info

```
controlplane:~$ kubectl cluster-info
Kubernetes control plane is running at https://172.30.1.2:6443
CoreDNS is running at https://172.30.1.2:6443/api/v1/namespaces/kube-system/services/kube-dns:dns/proxy
```

kubectl get nodes

```
controlplane:~$ kubectl get node
NAME             STATUS    ROLES    AGE    VERSION
controlplane     Ready    control-plane  3d21h  v1.33.2
node01           Ready    <none>      3d20h  v1.33.2
```

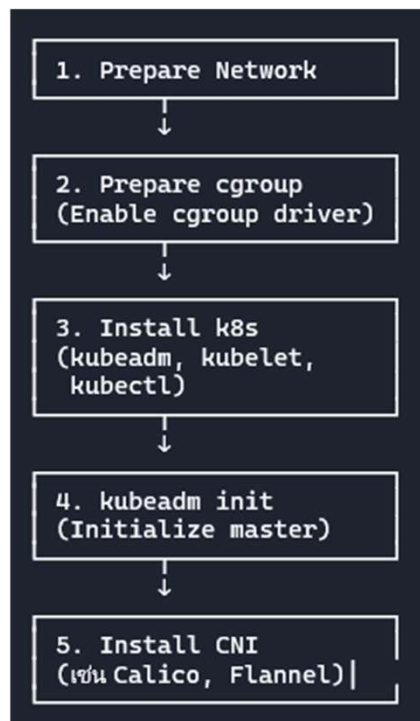
CMD

kubectl get pods -A -o wide

```
controlplane:~$ k get po -n kube-system -o wide
```

NAME	READY	STATUS	RESTARTS	AGE	IP	NODE	NOMINATED	NODE	READINESS	GATES
calico-kube-controllers-fdf5f5495-vxpw5	1/1	Running	2 (36m ago)	5d12h	192.168.0.3	controlplane	<none>		<none>	
canal-4xxhw	2/2	Running	2 (37m ago)	5d11h	172.30.2.2	node01	<none>		<none>	
canal-wlt99	2/2	Running	2 (36m ago)	5d11h	172.30.1.2	controlplane	<none>		<none>	
coredns-6ff97d97f9-4wljj	1/1	Running	1 (37m ago)	5d11h	192.168.1.2	node01	<none>		<none>	
coredns-6ff97d97f9-wmpcb	1/1	Running	1 (37m ago)	5d11h	192.168.1.3	node01	<none>		<none>	
etcd-controlplane	1/1	Running	3 (36m ago)	5d12h	172.30.1.2	controlplane	<none>		<none>	
kube-apiserver-controlplane	1/1	Running	3 (36m ago)	5d12h	172.30.1.2	controlplane	<none>		<none>	
kube-controller-manager-controlplane	1/1	Running	2 (36m ago)	5d12h	172.30.1.2	controlplane	<none>		<none>	
kube-proxy-d8lhg	1/1	Running	1 (37m ago)	5d11h	172.30.2.2	node01	<none>		<none>	
kube-proxy-tx14k	1/1	Running	2 (36m ago)	5d12h	172.30.1.2	controlplane	<none>		<none>	
kube-scheduler-controlplane	1/1	Running	2 (36m ago)	5d12h	172.30.1.2	controlplane	<none>		<none>	

Control plane install



Prepare network

```
cat <<EOF | sudo tee  
/etc/sysctl.d/k8s.conf  
net.ipv4.ip_forward = 1  
EOF
```

```
sudo sysctl --system
```

```
sysctl net.ipv4.ip_forward  
** output =1 ***
```

```
controlplane:~$ cat <<EOF | sudo tee /etc/sysctl.d/k8s.conf  
net.ipv4.ip_forward = 1  
EOF  
net.ipv4.ip_forward = 1
```

Prepare cgroup

vi /etc/containerd/config.toml

systemdCgroup = true

sudo systemctl restart containerd

```
[plugins."io.containerd.grpc.v1.cri".containerd.runtimes.runc.options]
  BinaryName = ""
  CriuImagePath = ""
  CriuPath = ""
  CriuWorkPath = ""
  IoGid = 0
  IoUid = 0
  NoNewKeyring = false
  NoPivotRoot = false
  Root = ""
  ShimCgroup = ""
  SystemdCgroup = true
```

Install K8s

```
sudo apt-get update
```

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

```
curl -fsSL https://pkgs.k8s.io/core:/stable:/v1.33/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:/v1.33/deb/ /' | sudo tee /etc/apt/sources.list.d/kubernetes.list
```

```
sudo apt-get update
```

```
sudo apt-cache madison kubeadm
```

```
sudo apt-get install -y kubelet kubeadm kubectl
```

```
sudo apt-mark hold kubelet kubeadm kubectl
```

kubeadm init

kubeadm init --apiserver-advertise-address=
--kubernetes-version= --pod-network-cidr=

*** พยายามใช้ kubeadm init -h ***

```
Usage:
  kubeadm init [flags]
  kubeadm init [command]

Available Commands:
  phase      Use this command to invoke single phase of the "init" workflow

Flags:
  --apiserver-advertise-address string    The IP address the API Server will advertise it's listening on. If
  --apiserver-bind-port int32             Port for the API Server to bind to. (default 6443)
  --apiserver-cert-extra-sans strings     Optional extra Subject Alternative Names (SANs) to use for the API
  --cert-dir string                       The path where to save and store the certificates. (default "/etc/
  --certificate-key string                 Key used to encrypt the control-plane certificates in the kubeadm-
  --config string                         Path to a kubeadm configuration file.
  --control-plane-endpoint string          Specify a stable IP address or DNS name for the control plane.
  --cri-socket string                     Path to the CRI socket to connect. If empty kubeadm will try to au
rd CRI socket.
  --dry-run                                Don't apply any changes; just output what would be done.
  --feature-gates string                  A set of key=value pairs that describe feature gates for various f
ControlPlaneKubeletLocalMode=true|false (BETA - default=true)
NodeLocalCRISocket=true|false (ALPHA - default=false)
PublicKeysECDSA=true|false (DEPRECATED - default=false)
RootlessControlPlane=true|false (ALPHA - default=false)
WaitForAllControlPlaneComponents=true|false (BETA - default=true)
```

Install CNI

<https://kubernetes.io/docs/concepts/cluster-administration/addons/>

*** Daemonset CNI status Running

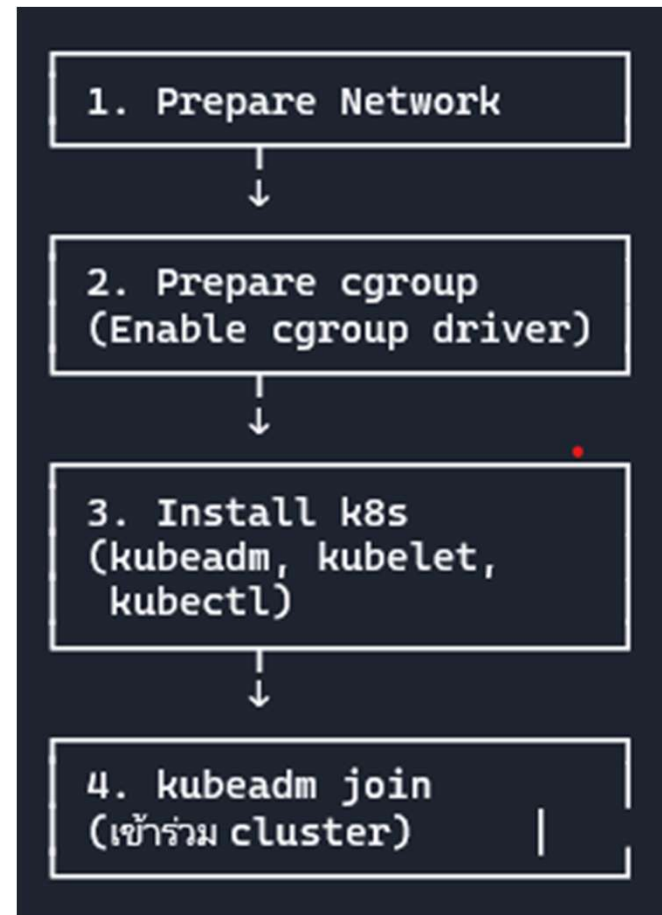
```
controlplane:~$ k get nodes
NAME             STATUS    ROLES    AGE   VERSION
controlplane     Ready    control-plane  5d12h  v1.33.2
node01           Ready    <none>      5d12h  v1.33.2
controlplane:~$ k get pods -n kube-system
NAME                                     READY   STATUS    RESTARTS   AGE
calico-kube-controllers-fdf5f5495-vxpw5 1/1     Running   2 (10m ago) 5d12h
canal-4xxhw                               2/2     Running   2 (10m ago) 5d12h
canal-wlt99                               2/2     Running   2 (10m ago) 5d12h
```

Node install

```
kubeadm join ip-master:6443 --token  
8tsrg6.8hlcdombbk1qwmv2 --discovery-token-ca-cert-  
hash  
sha256:134d379eb4613d3154a0ab5de356979b40b05d8c  
bc2134e088f84965f860d551
```

Note print token

```
kubeadm token create --print-join-command
```



Check running component

Control node

kubectl get pods -A -o wide

```
controlplane:~$ k get po -n kube-system -o wide
```

NAME	READY	STATUS	RESTARTS	AGE	IP	NODE	NOMINATED	NODE	READINESS	GATES
calico-kube-controllers-fdf5f5495-vxpw5	1/1	Running	2 (36m ago)	5d12h	192.168.0.3	controlplane	<none>		<none>	
canal-4xxdw	2/2	Running	2 (37m ago)	5d11h	172.30.2.2	node01	<none>		<none>	
canal-wlt99	2/2	Running	2 (36m ago)	5d11h	172.30.1.2	controlplane	<none>		<none>	
coredns-6ff97d97f9-4wljj	1/1	Running	1 (37m ago)	5d11h	192.168.1.2	node01	<none>		<none>	
coredns-6ff97d97f9-wmpcb	1/1	Running	1 (37m ago)	5d11h	192.168.1.3	node01	<none>		<none>	
etcd-controlplane	1/1	Running	3 (36m ago)	5d12h	172.30.1.2	controlplane	<none>		<none>	
kube-apiserver-controlplane	1/1	Running	3 (36m ago)	5d12h	172.30.1.2	controlplane	<none>		<none>	
kube-controller-manager-controlplane	1/1	Running	2 (36m ago)	5d12h	172.30.1.2	controlplane	<none>		<none>	
kube-proxy-d8lhg	1/1	Running	1 (37m ago)	5d11h	172.30.2.2	node01	<none>		<none>	
kube-proxy-tx14k	1/1	Running	2 (36m ago)	5d12h	172.30.1.2	controlplane	<none>		<none>	
kube-scheduler-controlplane	1/1	Running	2 (36m ago)	5d12h	172.30.1.2	controlplane	<none>		<none>	

Kube-API config

Path

/etc/kubernetes/manifests/kube-apiserver.yaml

สามารถแก้ service-cluster-cidr

```
controlplane:~$ cat /etc/kubernetes/manifests/kube-apiserver.yaml
apiVersion: v1
kind: Pod
metadata:
  annotations:
    kubeadm.kubernetes.io/kube-apiserver.advertise-address.endpoint: 172.30.1.2:6443
  creationTimestamp: null
  labels:
    component: kube-apiserver
    tier: control-plane
  name: kube-apiserver
  namespace: kube-system
spec:
  containers:
    - command:
      - kube-apiserver
      - --advertise-address=172.30.1.2
      - --allow-privileged=true
      - --authorization-mode=Node,RBAC
      - --client-ca-file=/etc/kubernetes/pki/ca.crt
      - --enable-admission-plugins=NodeRestriction
      - --enable-bootstrap-token-auth=true
      - --etcd-cafile=/etc/kubernetes/pki/etcd/ca.crt
      - --etcd-certfile=/etc/kubernetes/pki/apiserver-etcd-client.crt
      - --etcd-keyfile=/etc/kubernetes/pki/apiserver-etcd-client.key
      - --etcd-servers=https://127.0.0.1:2379
      - --kubelet-client-certificate=/etc/kubernetes/pki/apiserver-kubelet-client.crt
      - --kubelet-client-key=/etc/kubernetes/pki/apiserver-kubelet-client.key
      - --kubelet-preferred-address-types=InternalIP,ExternalIP,Hostname
      - --proxy-client-cert-file=/etc/kubernetes/pki/front-proxy-client.crt
      - --proxy-client-key-file=/etc/kubernetes/pki/front-proxy-client.key
      - --requestheader-allowed-names=front-proxy-client
      - --requestheader-client-ca-file=/etc/kubernetes/pki/front-proxy-ca.crt
      - --requestheader-extra-headers-prefix=X-Remote-Extra-
      - --requestheader-group-headers=X-Remote-Group
      - --requestheader-username-headers=X-Remote-User
      - --secure-port=6443
      - --service-account-issuer=https://kubernetes.default.svc.cluster.local
      - --service-account-key-file=/etc/kubernetes/pki/sa.pub
      - --service-account-signing-key-file=/etc/kubernetes/pki/sa.key
      - --service-cluster-ip-range=10.96.0.0/12
      - --tls-cert-file=/etc/kubernetes/pki/apiserver.crt
      - --tls-private-key-file=/etc/kubernetes/pki/apiserver.key
    image: registry.k8s.io/kube-apiserver:v1.33.2
```


Controller config

Path

/etc/kubernetes/manifests/kube-apiserver.yaml

สามารถแก้ pod-network-cidr

```
controlplane:~$ cat /etc/kubernetes/manifests/kube-controller-manager.yaml
apiVersion: v1
kind: Pod
metadata:
  creationTimestamp: null
  labels:
    component: kube-controller-manager
    tier: control-plane
  name: kube-controller-manager
  namespace: kube-system
spec:
  containers:
    - command:
      - kube-controller-manager
      - --allocate-node-cidrs=true
      - --authentication-kubeconfig=/etc/kubernetes/controller-manager.conf
      - --authorization-kubeconfig=/etc/kubernetes/controller-manager.conf
      - --bind-address=127.0.0.1
      - --client-ca-file=/etc/kubernetes/pki/ca.crt
      - --cluster-cidr=192.168.0.0/16
      - --cluster-name=kubernetes
      - --cluster-signing-cert-file=/etc/kubernetes/pki/ca.crt
      - --cluster-signing-key-file=/etc/kubernetes/pki/ca.key
      - --controllers=*,bootstrapsigner,tokencleaner
      - --kubeconfig=/etc/kubernetes/controller-manager.conf
      - --leader-elect=true
      - --requestheader-client-ca-file=/etc/kubernetes/pki/front-proxy-ca.crt
      - --root-ca-file=/etc/kubernetes/pki/ca.crt
      - --service-account-private-key-file=/etc/kubernetes/pki/sa.key
      - --service-cluster-ip-range=10.96.0.0/12
      - --use-service-account-credentials=true
    image: registry.k8s.io/kube-controller-manager:v1.33.2
```

Scheduler config

Path

/etc/kubernetes/manifests/kube-apiserver.yaml

```
controlplane:~$ cat /etc/kubernetes/manifests/kube-scheduler.yaml
apiVersion: v1
kind: Pod
metadata:
  creationTimestamp: null
  labels:
    component: kube-scheduler
    tier: control-plane
  name: kube-scheduler
  namespace: kube-system
spec:
  containers:
  - command:
    - kube-scheduler
    - --authentication-kubeconfig=/etc/kubernetes/scheduler.conf
    - --authorization-kubeconfig=/etc/kubernetes/scheduler.conf
    - --bind-address=127.0.0.1
    - --kubeconfig=/etc/kubernetes/scheduler.conf
    - --leader-elect=true
    image: registry.k8s.io/kube-scheduler:v1.33.2
```

Etcd config

Path

/etc/kubernetes/manifests/etcd.yaml

```
controlplane:~$ ls /etc/kubernetes/manifests/
etcd.yaml kube-apiserver.yaml kube-controller-manager.yaml kube-scheduler.yaml
controlplane:~$ cat /etc/kubernetes/manifests/etcd.yaml
apiVersion: v1
kind: Pod
metadata:
  annotations:
    kubeadm.kubernetes.io/etcd.advertise-client-urls: https://172.30.1.2:2379
  creationTimestamp: null
  labels:
    component: etcd
    tier: control-plane
  name: etcd
  namespace: kube-system
spec:
  containers:
  - command:
    - etcd
    - --advertise-client-urls=https://172.30.1.2:2379
    - --cert-file=/etc/kubernetes/pki/etcd/server.crt
    - --client-cert-auth=true
    - --data-dir=/var/lib/etcd
    - --experimental-initial-corrupt-check=true
    - --experimental-watch-progress-notify-interval=5s
    - --initial-advertise-peer-urls=https://172.30.1.2:2380
    - --initial-cluster=controlplane=https://172.30.1.2:2380
    - --key-file=/etc/kubernetes/pki/etcd/server.key
    - --listen-client-urls=https://127.0.0.1:2379,https://172.30.1.2:2379
    - --listen-metrics-urls=http://127.0.0.1:2381
    - --listen-peer-urls=https://172.30.1.2:2380
    - --name=controlplane
    - --peer-cert-file=/etc/kubernetes/pki/etcd/peer.crt
    - --peer-client-cert-auth=true
    - --peer-key-file=/etc/kubernetes/pki/etcd/peer.key
    - --peer-trusted-ca-file=/etc/kubernetes/pki/etcd/ca.crt
    - --snapshot-count=10000
    - --trusted-ca-file=/etc/kubernetes/pki/etcd/ca.crt
    image: registry.k8s.io/etcd:3.5.21-0
```

Scheduler config

- Path

/etc/kubernetes/manifests/kube-apiserver.yaml

```
controlplane:~$ cat /etc/kubernetes/manifests/kube-scheduler.yaml
apiVersion: v1
kind: Pod
metadata:
  creationTimestamp: null
  labels:
    component: kube-scheduler
    tier: control-plane
  name: kube-scheduler
  namespace: kube-system
spec:
  containers:
  - command:
    - kube-scheduler
    - --authentication-kubeconfig=/etc/kubernetes/scheduler.conf
    - --authorization-kubeconfig=/etc/kubernetes/scheduler.conf
    - --bind-address=127.0.0.1
    - --kubeconfig=/etc/kubernetes/scheduler.conf
    - --leader-elect=true
    image: registry.k8s.io/kube-scheduler:v1.33.2
```

Upgrade

Step 1: Upgrade kubeadm
- ใช้ apt/yum
- ตรวจสอบด้วย `kubeadm version` |



Step 2: Upgrade kubelet
- ใช้ apt/yum
- Restart ด้วย systemctl



Step 3: Upgrade kubectl
- ใช้ apt/yum
- ตรวจสอบด้วย `kubectl version` |

Control plan

kubectl drain control

kubeadm version

sudo apt-mark unhold kubelet kubeadm kubectl

apt update

apt-cache madison kubeadm

apt-get install kubeadm=1.33.0-1.1

kubeadm upgrade plan 1.33.0-1.1

kubeadm upgrade apply 1.33.0-1.1

apt-get install kubelet=1.33.0-1.1 kubelet=1.33.0-1.1

systemctl daemon-reload

systemctl restart kubelet

查看 version -> vi /etc/apt/sources.list.d/kubernetes.list

```
controlplane:~$ kubeadm upgrade plan
[preflight] Running pre-flight checks.
[upgrade/config] Reading configuration from the "kubeadm-config" ConfigMap in namespace "kube-system"...
[upgrade/config] Use 'kubeadm init phase upload-config --config your-config-file' to re-upload it.
[upgrade] Running cluster health checks
[upgrade] Fetching available versions to upgrade to
[upgrade/versions] Cluster version: 1.33.2
[upgrade/versions] kubeadm version: v1.33.2
I0925 15:53:06.464038 14490 version.go:261] remote version is much newer: v1.34.1; falling back to: stable-1.33
[upgrade/versions] Target version: v1.33.5
[upgrade/versions] Latest version in the v1.33 series: v1.33.5

Components that must be upgraded manually after you have upgraded the control plane with 'kubeadm upgrade apply':
COMPONENT  NODE      CURRENT  TARGET
kubelet     controlplane  v1.33.2  v1.33.5
kubelet     node01      v1.33.2  v1.33.5

Upgrade to the latest version in the v1.33 series:

COMPONENT      NODE      CURRENT  TARGET
kube-apiserver  controlplane  v1.33.2  v1.33.5
kube-controller-manager  controlplane  v1.33.2  v1.33.5
kube-scheduler  controlplane  v1.33.2  v1.33.5
kube-proxy      1.33.2      v1.33.5
CoreDNS         v1.12.0     v1.12.0
etcd            controlplane  3.5.21-0 3.5.21-0

You can now apply the upgrade by executing the following command:

    kubeadm upgrade apply v1.33.5

Note: Before you can perform this upgrade, you have to update kubeadm to v1.33.5.
```

The table below shows the current state of component configs as understood by this version of kubeadm. Configs that have a "yes" mark in the "MANUAL UPGRADE REQUIRED" column require manual config upgrade or resetting to kubeadm defaults before a successful upgrade can be performed. The version to manually upgrade to is denoted in the "PREFERRED VERSION" column.

API GROUP	CURRENT VERSION	PREFERRED VERSION	MANUAL UPGRADE REQUIRED
kubeproxy.config.k8s.io	v1alpha1	v1alpha1	no
kubelet.config.k8s.io	v1beta1	v1beta1	no

All rights reserved by OpsDev

Node

```
kubect! drain node01
```

```
kubeadm version
```

```
apt update
```

```
apt-mark unhold kubelet kubeadm kubect!
```

```
apt-cache madison kubeadm
```

```
apt-get upgrade -y kubeadm=1.33.0-1.1
```

```
kubeadm upgrade node
```

```
apt-get install kubelet=1.33.0-1.1 kubelet=1.33.0-1.1
```

```
systemctl daemon-reload
```

```
systemctl restart kubelet
```

```
apt-mark hold kubelet kubeadm kubect!
```

```
ໜ້າ version -> vi /etc/apt/sources.list.d/kubernetes.list
```

All rights reserved by OpsDev

🐙 เปรียบเทียบคำสั่ง Drain, Cordon, Uncordon

คำสั่ง	จุดประสงค์หลัก	ผลกระทบต่อ Pod บน Node	Node รับ Pod ใหม่หรือไม่	ใช้เมื่อ...
kubectl	ป้องกันไม่ให้มีการ schedule Pod ใหม่	ไม่มีผลต่อ Pod ที่รันอยู่	❌ ไม่รับ Pod ใหม่	เตรียม node สำหรับ maintenance
kubectl	เปิดให้ node รับ Pod ใหม่อีกครั้ง	ไม่มีผลต่อ Pod ที่รันอยู่	✅ รับ Pod ใหม่ได้	หลังจาก maintenance เสร็จ
kubectl	ย้าย Pod ออกจาก node เพื่อ maintenance	✅ ลบ Pod ที่สามารถ reschedule ได้	❌ ไม่รับ Pod ใหม่	ต้องการเคลียร์ node เพื่อปิดหรืออัปเดต



Troubleshooting

All rights reserved by OpsDev

Control plan

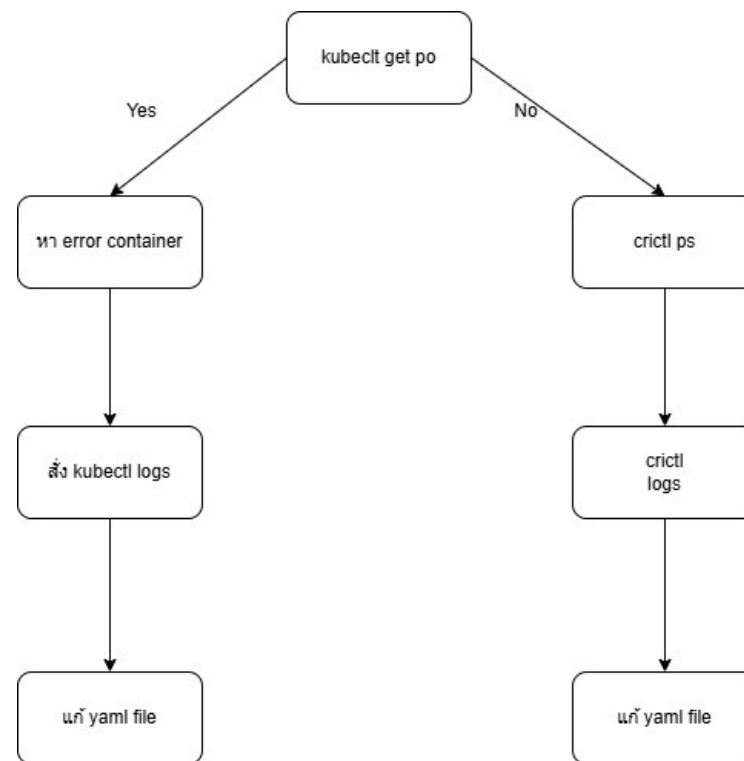
- 1.kubectl get ... อะไรไม่ได้แสดงว่า kube-api มีปัญหา
 2. Pod ไม่ถูก deploy (status pending) ไปที่ node แสดงว่า kube-schedule มีปัญหา
 3. Deployment มีปัญหาแสดงว่า control plan มีปัญหา
 4. ในกรณีที่ kubectl ใช้ไม่ได้ อาจจะต้องใช้ crictl ช่วย debug
- อย่าลืมว่า K8s เป็น Automate แต่ container รันผ่านพวก container run time

Logs path

/var/log/syslog

/var/log/containers

/var/log/pods



Node

1. ให้ check -> `service kubelet status`
2. ดู log error -> `/var/log/syslog`
3. ดู config kubelet -> `/var/lib/kubelet/kubeadm-flags.env`

สามารถใช้คำสั่ง `journalctl -u xxx`



Debug Cert

kubeadm certs check-expiration

Kubeadm certs renew