MÓDULO 4

Arquitectura Kubernetes

CONCEPTOS DE KUBERNETES

En Kubernetes se utilizan los objetos de la API de Kubernetes para describir el estado deseado del clúster:

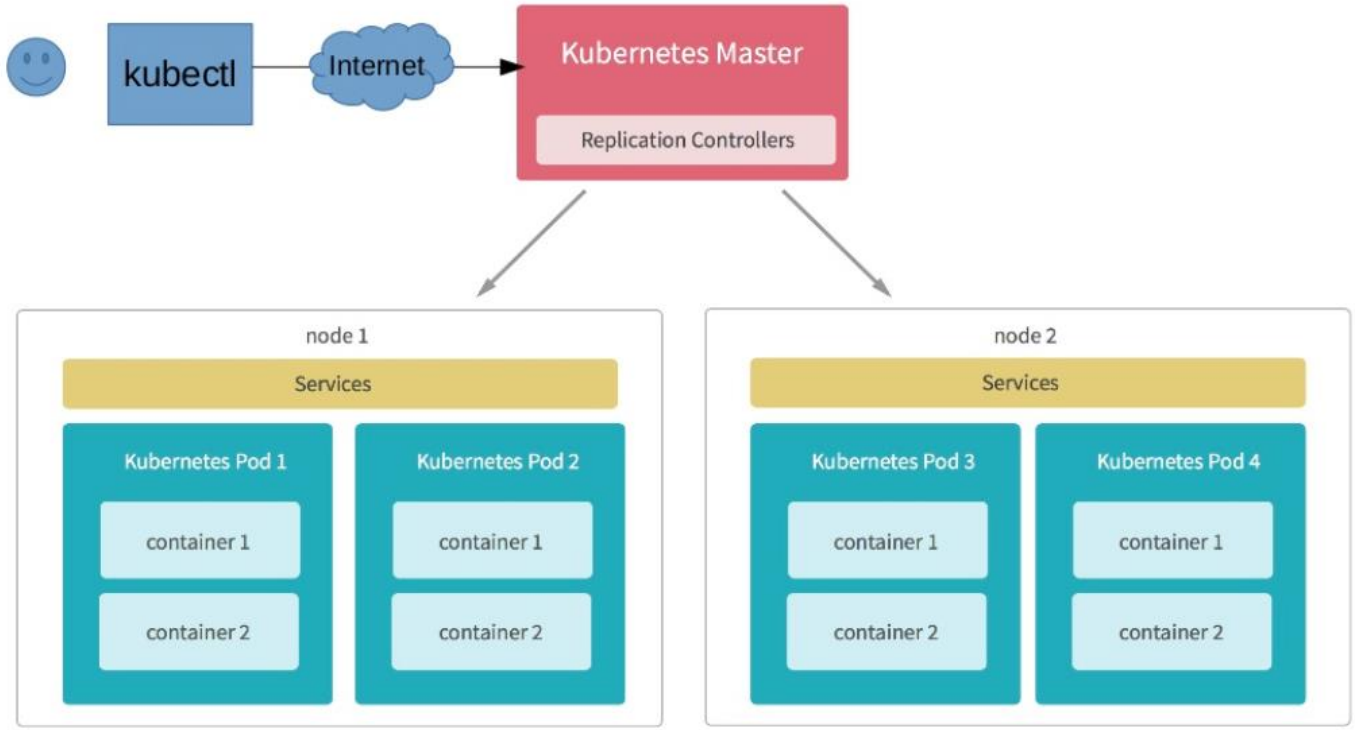
- Qué aplicaciones u otras cargas de trabajo se quieren ejecutar, qué imagenes de contenedores usan, el número de replicas, qué red y qué recursos de almacenamiento quieres que tengan disponibles, etc.

- Se especifica el estado deseado del clúster mediante la creación de objetos usando la API de Kubernetes, típicamente mediante la interfaz de línea de comandos, kubectl.

- También se puede usar la API de Kubernetes directamente para interactuar con el clúster y especificar o modificar tu estado deseado.

PLANO DE CONTROL DE KUBERNETES

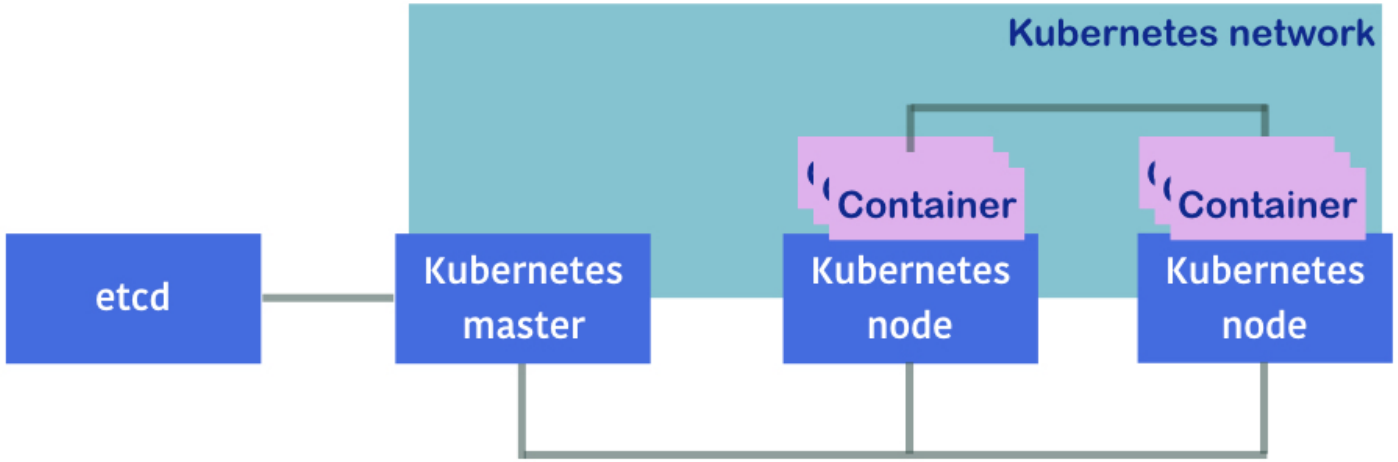
- Una vez que se especifica el estado deseado, el Plano de Control de Kubernetes realizará las acciones necesarias para que el estado actual del clúster coincida con el estado deseado.
- Para ello, Kubernetes realiza diferentes tareas de forma automática, como pueden ser: parar o arrancar contenedores, escalar el número de réplicas de una aplicación dada, etc.
- El Plano de Control de Kubernetes consiste en un grupo de procesos que corren en tu clúster.



COMPONENTES DE KUBERNETES

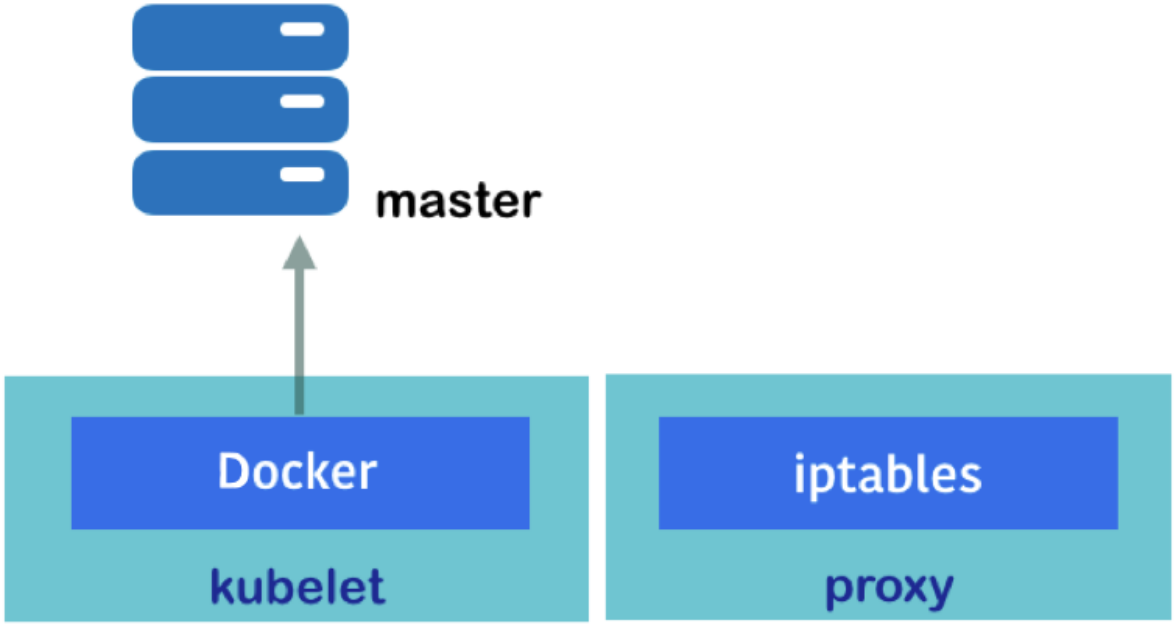
Estos componentes están conectados a través de una red, como se muestra en el siguiente diagrama:

- Master Kubernetes: se conecta a etcd a través de HTTP o HTTPS para almacenar los datos.
- Nodos de Kubernetes: se conecta al maestro de Kubernetes a través de HTTP o HTTPS para obtener un comando e informar el estado.
- Red de Kubernetes: L2, L3 o superposición hacen una conexión de sus aplicaciones de contenedor



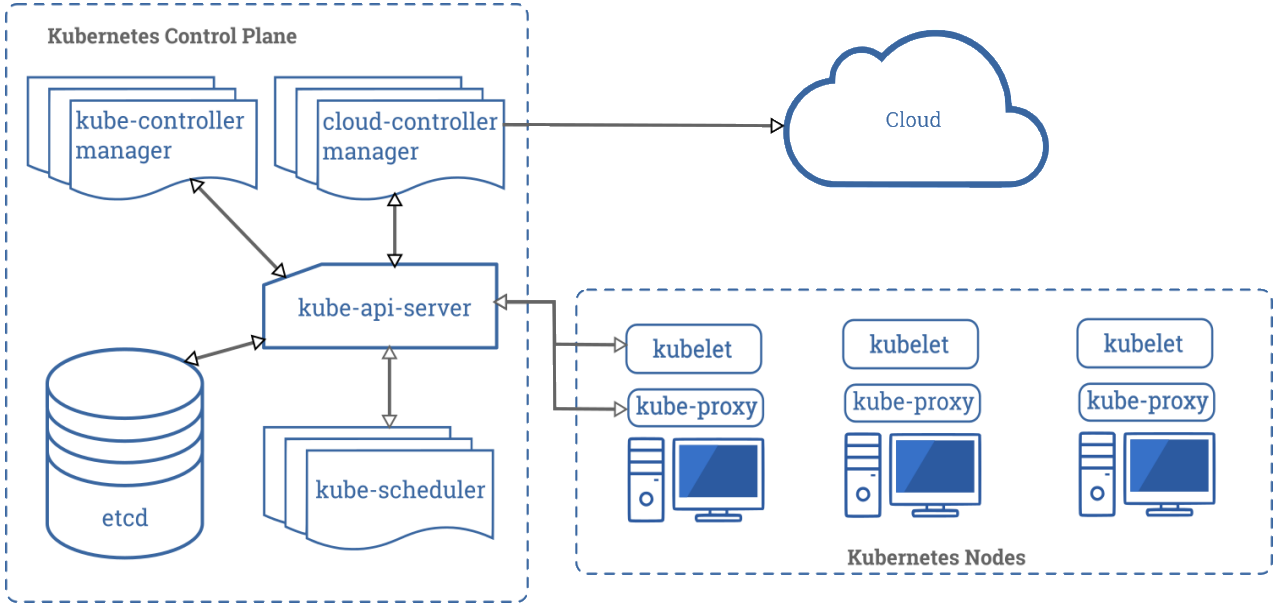
MASTER DE KUBERNETES

- El Master de Kubernetes es el responsable de mantener el estado deseado de tu clúster.
- Cuando interactuas con Kubernetes, como por ejemplo cuando utilizas la interfaz de línea de comandos kubectl, te estás comunicando con el master de tu clúster de Kubernetes.
- Por “master” entendemos la colección de procesos que gestionan el estado del clúster.
- Típicamente, estos procesos se ejecutan todos en un único nodo del clúster, y este nodo recibe por tanto la denominación de master.
- El master puede estar replicado por motivos de disponibilidad y redundancia.
- El Master de Kubernetes es un conjunto de tres procesos que se ejecutan en un único nodo del clúster, kube-apiserver, kube-controller-manager y kube-scheduler.



COMPONENTES DE MASTER NODE API SERVER

- API Server valida y configura datos para los objetos de la API que incluyen pods, servicios, controladores de replicación y otros.
- El servidor API sirve para operaciones REST y proporciona la interfaz para el estado compartido del clúster a través del cual interactúan todos los demás componentes.

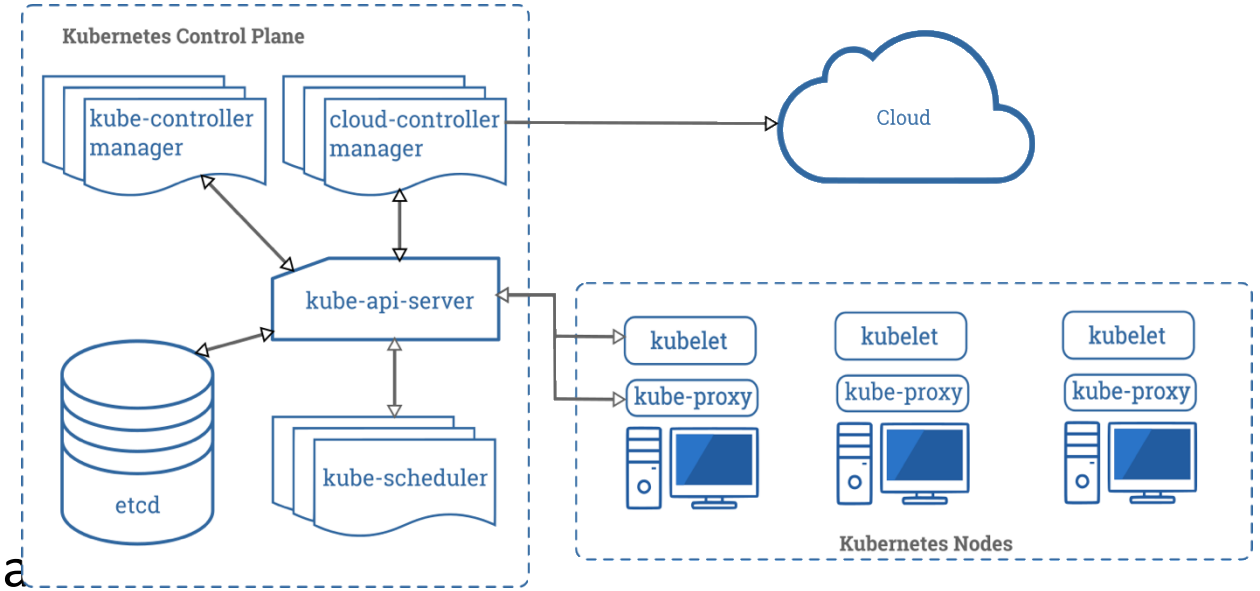


COMPONENTES DE MASTER NODE KUBE-SCHEDULER

- El Kube-Scheduler es una función rica en políticas, con reconocimiento de topología y específica de la carga de trabajo que afecta significativamente la disponibilidad, el rendimiento y la capacidad.

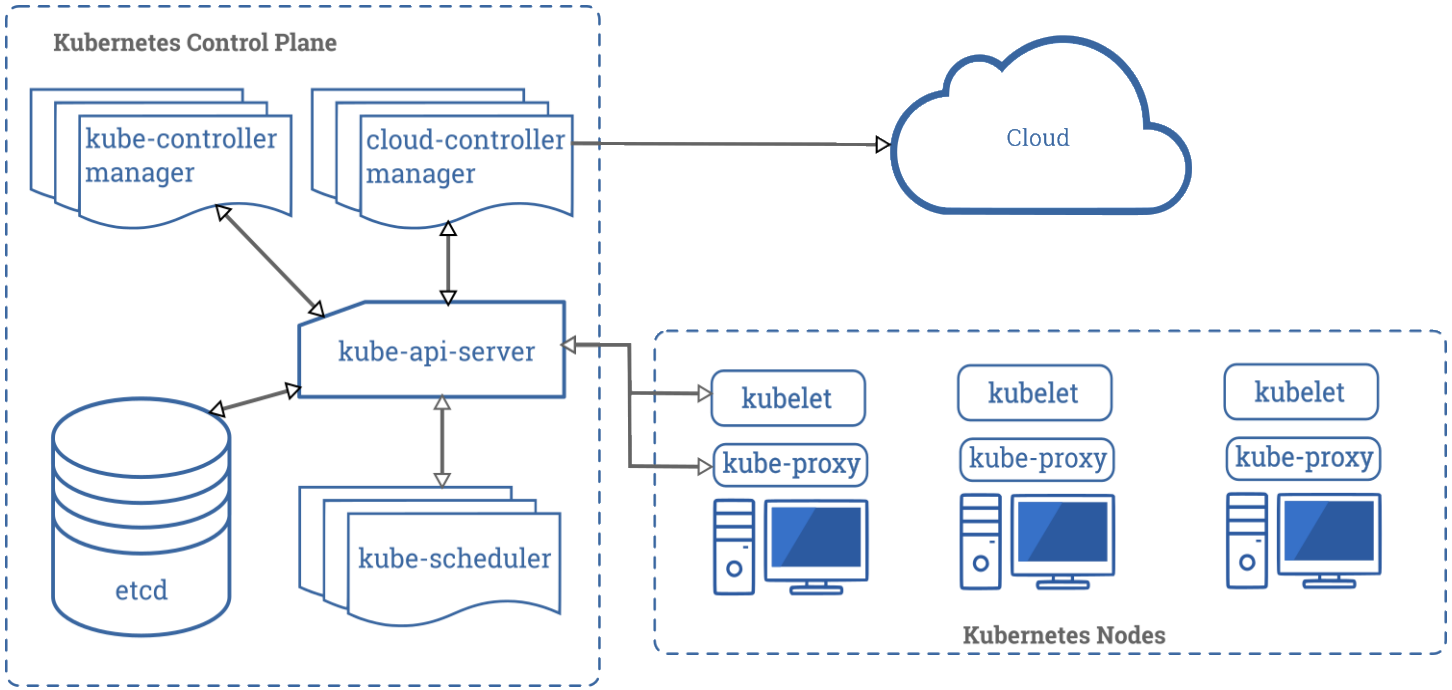
- El planificador debe tener en cuenta los requisitos de recursos individuales y colectivos, los requisitos de calidad de servicio, las restricciones de hardware / software / política, las especificaciones de afinidad y antiafinidad, la ubicación de los datos, la interferencia entre cargas de trabajo, los plazos, etc.

- Los requisitos específicos de la carga de trabajo se expondrán a través de la API según sea necesario.



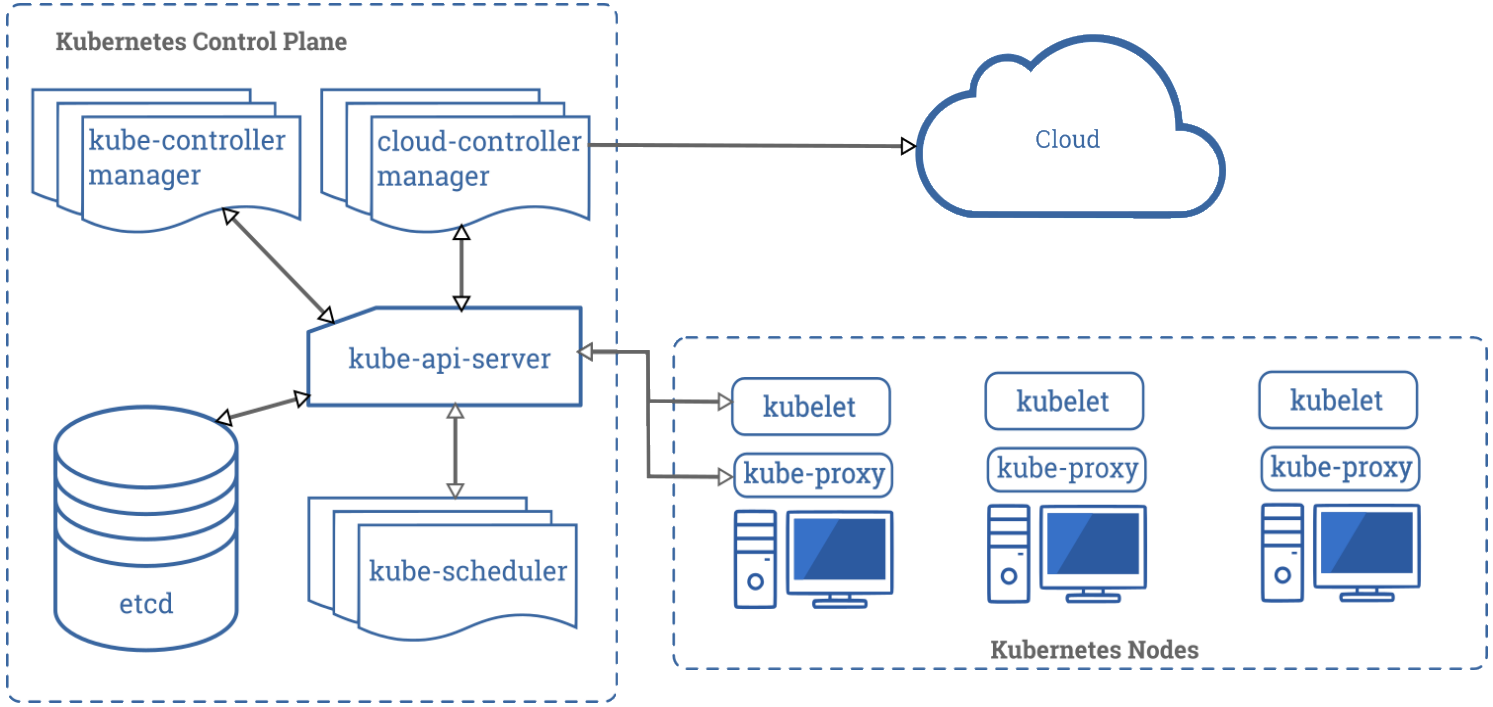
COMPONENTES DE MASTER NODE KUBE-CONTROLLER-MANAGER

- Es un daemon que se ejecuta en los nodos master y que se encarga de gestionar «los controladores».
- Un controlador en Kubernetes es un proceso en segundo plano que ejecuta determinadas tareas ordinarias para el funcionamiento del clúster.



COMPONENTES DE MASTER NODE ETCD

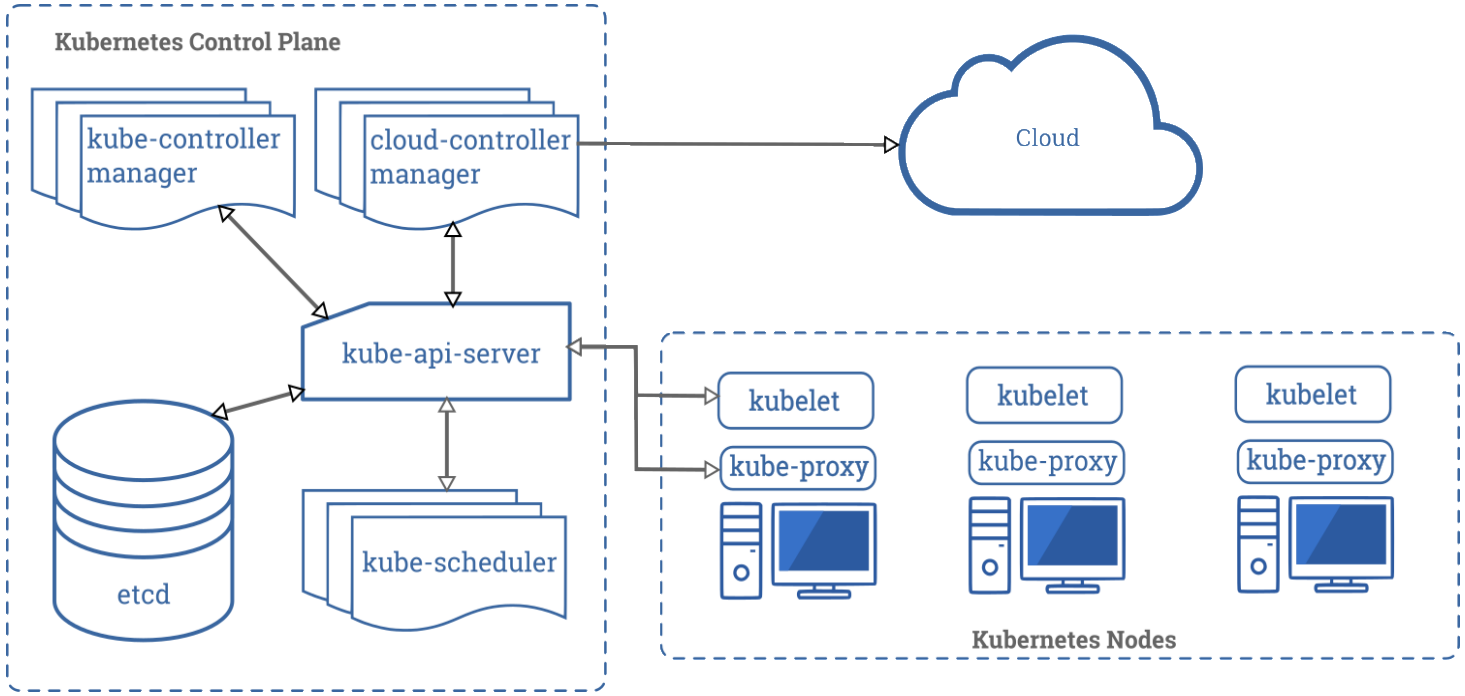
- El estado de un clúster se almacena en dos lugares.
- El estado deseado de todos los objetos se almacena de forma persistente en el etcd.
- El estado actual de los objetos se almacena en el etcd, y parcialmente en memoria dentro de los kubelets en los nodos individuales.



COMPONENTES DE MASTER NODE CLOUD-CONTROLLER-MANAGER

- Cloud-controller-manager se ejecuta en controladores que interactúan con la funcionalidad específica del proveedor de la nube.

- Los ejemplos incluyen un controlador de servicio que crea y administra instancias de “cloud load Balancer”, de un controlador de volumen que crea, adjunta y gestiona volúmenes de almacenamiento específicos de la nube.

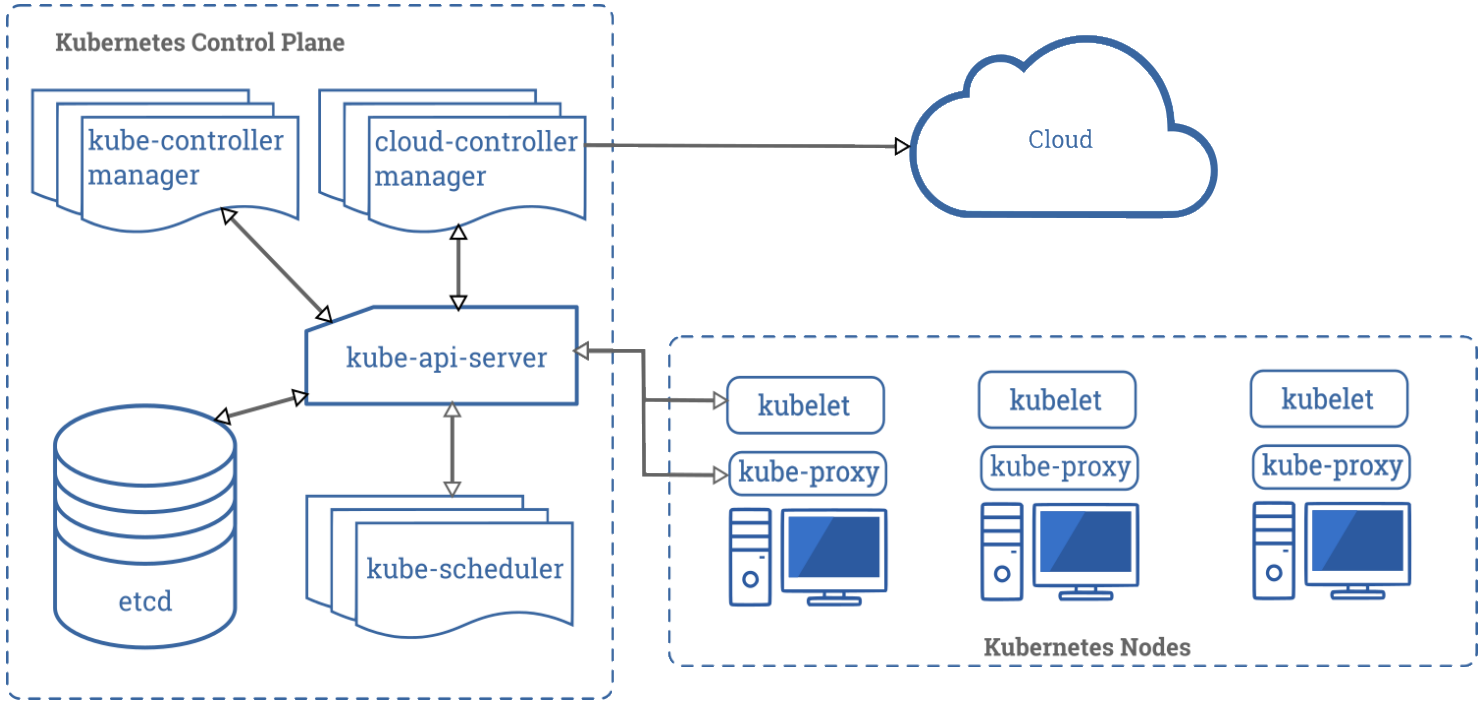


KUBERNETES NODES

- En un clúster de Kubernetes, los nodos son las máquinas (máquinas virtuales, servidores físicos, etc) que ejecutan tus aplicaciones y flujos de trabajo .
- Son Los Llamados Workers.
- El master de Kubernetes controla cada nodo, por lo que en raras ocasiones interactuarás con los nodos directamente.
- Los nodos no master contenidos en el clúster, ejecutan dos procesos: kubelet, kube-proxy.

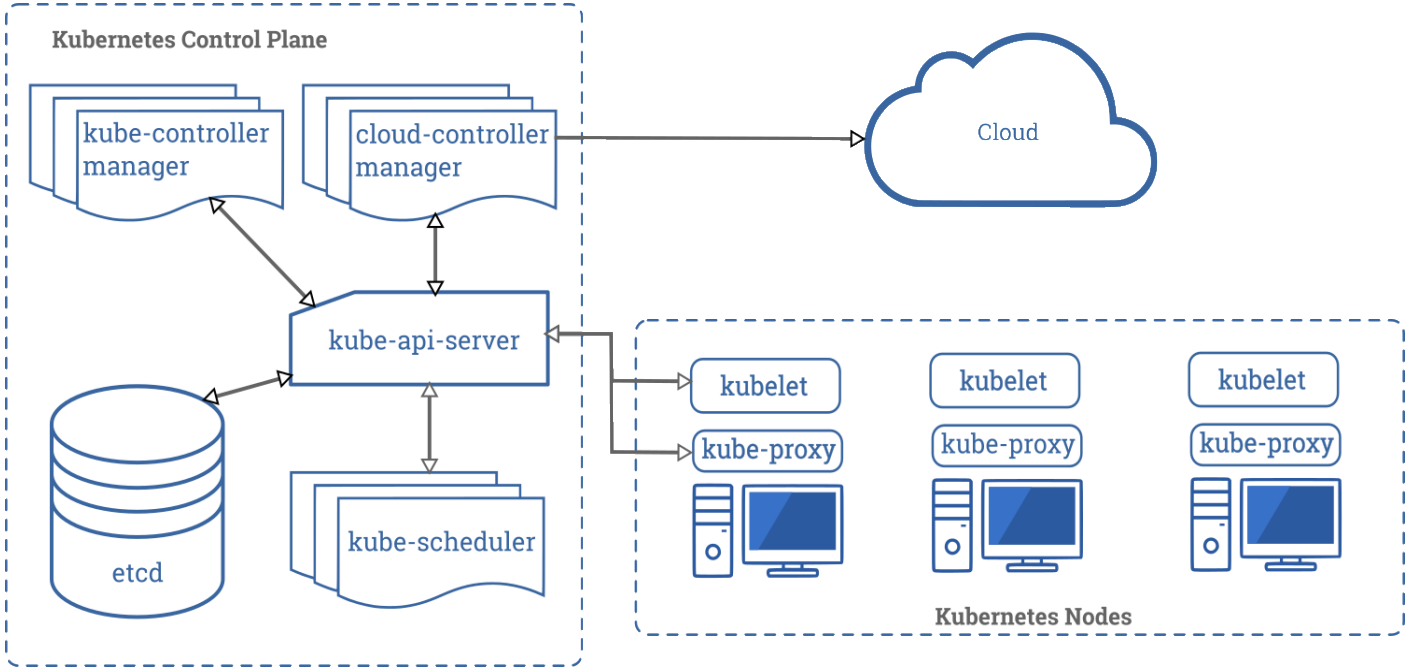
COMPONENTES DE KUBERNETES NODES KUBELET

- kubelet es el ”node agent" primario que se ejecuta en cada nodo.
- Se comunica con el Master de Kubernetes .
- Lee los manifiestos del contenedor, y garantiza que los contenedores definidos estén iniciados y ejecutándose.



COMPONENTES DE KUBERNETES NODES KUBE-PROXY

- El proxy de red de Kubernetes se ejecuta en cada nodo.
- Esto refleja los servicios definidos en la API de Kubernetes en cada nodo y puede realizar reenvíos de flujo TCP, UDP y SCTP simples o reenvío de TCP, UDP y SCTP en un conjunto de backends.
- Las direcciones IP y los puertos del clúster de servicio se encuentran actualmente a través de variables de entorno compatibles con enlaces Docker que especifican los puertos abiertos por el proxy de servicio.
- Hay un complemento opcional que proporciona DNS de clúster para estas IP de clúster.



COMPONENTES DE KUBERNETES NODE CONTAINER RUNTIME INTERFACE

- Es el software actual que se inicia y gestiona contenedores individuales en el nodo.
- Kubernetes ahora define el estándar de Interfaz de Container Runtime Interface(CRI) que permite diferentes tiempos de ejecución compatibles con CRI para ser utilizados fácilmente dentro de un cluster; por ejemplo: Docker, containerd, CRI-O, rklet.