RA5.4

K3’s & K9’s

Ciberseguridad en entornos de las tecnologías de la información

Incidentes de ciberseguridad

Índice

1. Kubernetes

En los últimos años, la evolución del desarrollo de software y la creciente complejidad de las aplicaciones modernas han impulsado la necesidad de herramientas potentes y flexibles para la gestión de infraestructura. En este contexto, **Kubernetes** se ha consolidado como uno de los pilares fundamentales de la computación en la nube, la orquestación de contenedores y el DevOps.

Kubernetes, a menudo abreviado como *K8s*, es una plataforma de código abierto diseñada por Google y ahora mantenida por la *Cloud Native Computing Foundation* (CNCF). Su propósito principal es automatizar el despliegue, la escalabilidad y la gestión de aplicaciones en contenedores. Fue creado para resolver los desafíos que surgieron con el uso extensivo de contenedores, como Docker, a gran escala.

A diferencia de las máquinas virtuales tradicionales, los contenedores son ligeros y rápidos, lo que los hace ideales para aplicaciones distribuidas. Sin embargo, con cientos o miles de contenedores en funcionamiento, la gestión manual se vuelve inviable. Ahí es donde entra Kubernetes.

Kubernetes sigue una arquitectura cliente-servidor. Sus principales componentes son:

Master Node (Plano de control): Coordina el clúster y toma decisiones globales (como programar pods, responder a eventos, mantener el estado deseado).

Worker Nodes (Nodos de trabajo): Ejecutan las aplicaciones en contenedores. Cada nodo contiene un kubelet, un agente que gestiona los pods, y un runtime de contenedores (como containerd o Docker).

Pod: Es la unidad más pequeña desplegable en Kubernetes. Un pod puede contener uno o varios contenedores que comparten red y almacenamiento.

Servicios (Services): Abstracciones que permiten acceder a los pods sin importar su ubicación física en el clúster.

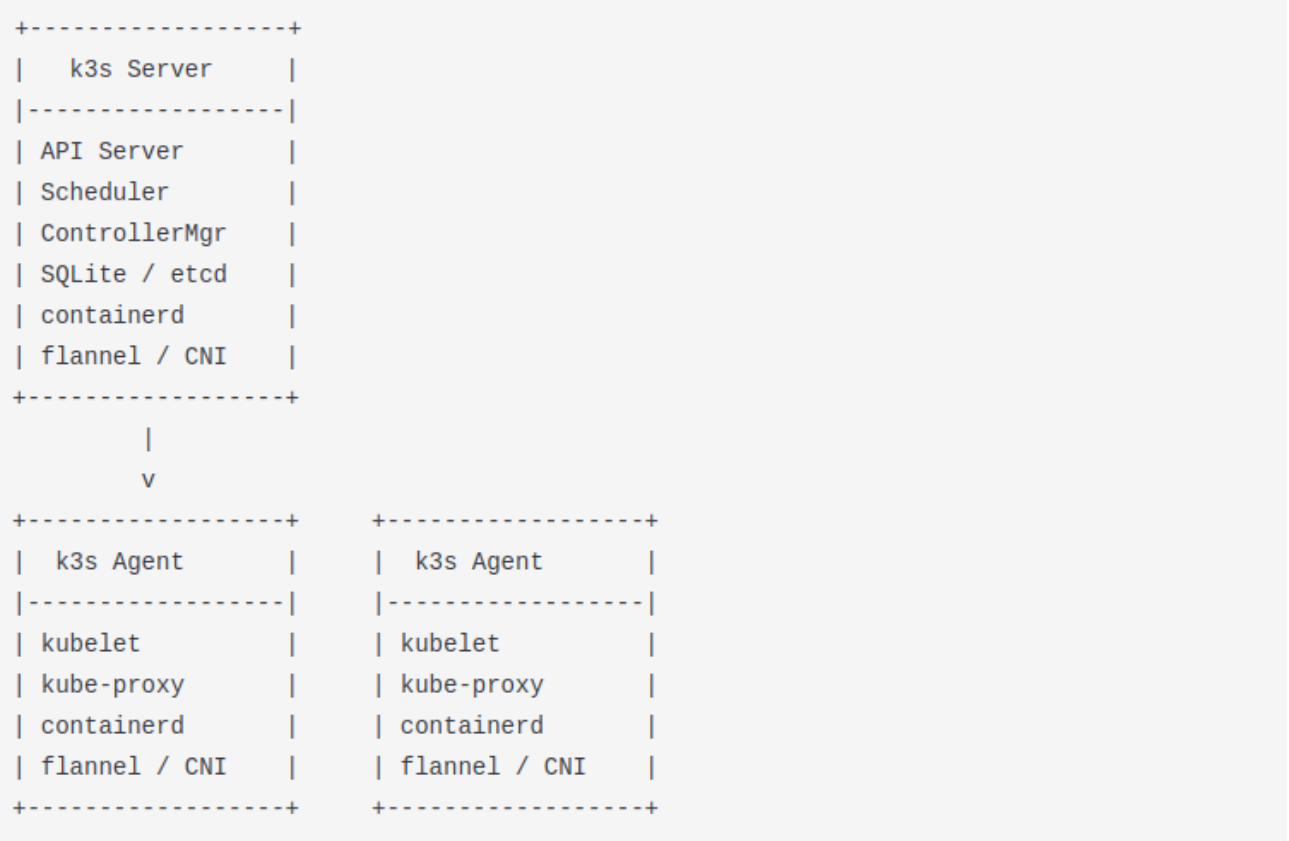
Controladores (Deployments, StatefulSets, DaemonSets): Se encargan de crear y mantener conjuntos de pods con propiedades específicas.

2. K3s

k3s es una distribución ligera de Kubernetes, diseñada para ser más fácil de instalar, operar y mantener en entornos con recursos limitados como **IoT**, **edge computing**, **dispositivos embebidos** o **desarrollos locales**. Fue creada por **Rancher Labs** como una alternativa simplificada a Kubernetes tradicional (k8s), manteniendo su compatibilidad con la mayoría de herramientas y funcionalidades. Kubernetes es potente, pero su complejidad y requisitos de recursos pueden ser excesivos para pequeños dispositivos o desarrollos rápidos. **k3s** soluciona esto con: Instalación rápida con un solo binario, ocupa menos de 100 MB, requiere menos requerimientos y compatible con diferentes arquitecturas.

2.1. Arquitectura Single-Node

k3s sigue el modelo básico de Kubernetes: **control plane + nodos worker**, pero lo simplifica en varios aspectos para reducir peso, dependencias y consumo de recursos.



2.2. Arquitectura HA

Cuando necesitas un clúster **resistente a fallos**, puedes configurar k3s en modo HA. Existen dos modos de funcionamiento: Interna (embedded etcd) y Externa (base de datos externa).

Externa: Se instalan múltiples nodos "server" (control plane), usan una base de datos externa **PostgreSQL**, **MySQL** o **etcd** y muy escalable.

Interna: Desde k3s v1.19+, puedes usar **etcd embebido** entre servidores (similar a k8s estándar), requiere al menos 3 nodos "server" para **quorum** y menos dependencias externas, por tanto más simple.

2.3. Componentes

Componente	Función
`k3s server`	Control plane, puede ser parte de HA
`k3s agent`	Nodos worker que ejecutan contenedores
`etcd / SQLite / DB`	Almacenan el estado del clúster
`kubectl`	Herramienta para interactuar con el clúster
`LoadBalancer/Traefik`	Expone servicios fuera del clúster

2.4. kubectl

`kubectl` es la **línea de comandos oficial de Kubernetes**. Te permite comunicarte con el **API Server** del clúster para: crear y administrar recursos (pods, servicios, deployments, etc.), ver el estado de tu clúster, aplicar cambios a tus aplicaciones y depurar problemas.

`kubectl` usa un archivo de configuración (por defecto `~/.kube/config`) para saber **a qué clúster conectarse**.

Comando	Descripción
`kubectl get nodes`	Lista los nodos del clúster
`kubectl get pods`	Muestra los pods actuales
`kubectl get svc`	Lista los servicios
`kubectl describe pod POD_NAME`	Muestra detalles completos de un pod
`kubectl logs POD_NAME`	Muestra los logs de un pod
`kubectl exec -it POD_NAME -- bash`	Ejecuta comandos dentro del contenedor
`kubectl apply -f archivo.yaml`	Aplica una configuración desde un archivo YAML
`kubectl delete -f archivo.yaml`	Elimina los recursos definidos en un YAML
`kubectl edit deployment NOMBRE`	Abre un editor para modificar el recurso
`kubectl scale deployment NOMBRE --replicas=3`	Escala un deployment a 3 réplicas

3. K9s

K9s es una interfaz de usuario basada en terminal para interactuar con sus clústeres de Kubernetes. El objetivo de este proyecto es facilitar la navegación, la observación y la gestión de sus aplicaciones implementadas en tiempo real. K9s monitorea constantemente Kubernetes para detectar cambios y ofrece comandos posteriores para interactuar con sus recursos observados.

Tecla	Acción
`:`	Comando directo (por ejemplo `:pod`, `:svc`)
↑ ↓	Moverte entre recursos
`Enter`	Entrar a los detalles de un recurso
`l`	Ver logs del pod seleccionado
`d`	Describir el recurso (`kubectl describe`)
`e`	Editar el YAML del recurso (`kubectl edit`)
`s`	Escalar un deployment
`x`	Eliminar un recurso
`q`	Salir o volver atrás

4. Resultado de aprendizaje y criterios de evaluación

Esta unidad didáctica contribuye a trabajar los siguientes criterios de evaluación:

RA5	Implanta sistemas seguros de desplegado de software, utilizando herramientas para la automatización de la construcción de sus elementos
CA.A	0.5
CA.B	0.5
CA.C	0.5

5. Actividades

5.1. Realizar la instalación, configuración y validación de K3s en modo single-node y realizar el deploy de un servicio nginx con 2 réplicas. Realizar la instalación, configuración y validación de K9s del punto 3.1.1.

5.2. Realizar la instalación, configuración y validación de K3s en modo HA y realizar el deploy de un servicio nginx con 2 réplicas. Realizar la instalación, configuración y validación de K9s del punto 3.2.1.

5.3. Realizar el despliegue en K3s y validación mediante K9s del docker-compose:

https://aulasoftwarelibre.github.io/taller-de-docker/dockerfile/#balanceo-de-carga

6. Bibliografía

Orquestación de contenedores para producción. (s. f.). Kubernetes. https://kubernetes.io/es/

K3s. (s. f.). https://k3s.io/

K9S - Manage your Kubernetes clusters in style. (s. f.). https://k9scli.io/

Rúbrica

ACTIVIDAD	50%	70%	80%	90%	100%
5.1	Funcionamiento correcto del K3s 'single-node'	Despliegue servicio	Validación k9s	Análisis técnico	Extra
5.2	Funcionamiento correcto del K3s 'ha'	Despliegue servicio	Validación k9s	Análisis técnico	Extra
5.3	Funcionamiento correcto del K3s	Despliegue servicio	Validación k9s	Análisis técnico	Extra