
**REPÚBLICA DE GUATEMALA**

**UNIVERSIDAD DE SAN CARLOS DE GUATEMALA**

**FACULTAD DE INGENIERÍA**

**INGENIERÍA EN CIENCIAS Y SISTEMAS**

SOFTWARE AVANZADO - LABORATORIO - 1S 2024

CATEDRÁTICO: EVEREST DARWIN MEDINILLA

AUXILIAR: DIEGO RENE MOLINA ROLDAN

# Práctica 2 Documentación

REALIZADO POR:

201830313 DENILSON DE LEÓN

GUATEMALA, PRIMER SEMESTRE 5 DE MARZO 2024

# ÍNDICE

**PRÁCTICA 2 - Descripciones** **4**
Componentes de Kubernetes 4
Nodos Trabajadores: 4
Plano de Control (Control Plane): 4
Componentes del Plano de Control 5
kube-apiserver: 5
etcd: 5
kube-scheduler: 5
kube-controller-manager 6
cloud-controller-manager 7
Controladores con Dependencias en Proveedores de Servicios en la Nube: 7
Componentes del Nodo (Node Components) 8
kubelet: 8
kube-proxy: 8
Container Runtime (Tiempo de Ejecución de Contenedores): 9
Addons: 9
Algunos Addons Selecionados: 9
Docker y sus Componentes 11
Docker: 11
Creación de Imágenes 11
Gestión de Contenedores 11
Instrucciones para Ejecutar, Detener y Eliminar Contenedores: 11
Interacción con el Sistema de Archivos del Contenedor: 12
Redes y Volúmenes 12
Configuración de Redes y Volúmenes: 12
Docker Compose 12
Uso de Docker Compose para Definir y Gestionar Aplicaciones Multicontenedor: 12
Seguridad 13
Prácticas Recomendadas para la Seguridad en Docker: 13
Configuración de Privilegios y Control de Acceso: 13
Escenarios Comunes 13
Ejemplos Prácticos de Uso Común de Docker: 13
Contratos de Microservicios 14
Acuerdo de Nivel de Servicio y Términos - Docker 14
Horario de Operación: 14
Detalles del SLA según la Suscripción: 14
Soporte para Docker Desktop: 15
Acuerdo de Nivel de Servicio (SLA) de Google Kubernetes Engine 15
Crédito Financiero Máximo: 17

Exclusiones del SLA: 17
Customer Debe Solicitar el Crédito Financiero: 18
**Aplicaciones a la Práctica** **19**
**Bibliografía** **21**

# PRÁCTICA 2 - Descripciones

## Componentes de Kubernetes

Cuando despliegas Kubernetes, obtienes un clúster. Un clúster de Kubernetes consta de un conjunto de máquinas trabajadoras, llamadas nodos, que ejecutan aplicaciones en contenedores. Cada clúster tiene al menos un nodo trabajador.

### *Nodos Trabajadores:*

- Definición: Las máquinas que ejecutan aplicaciones en contenedores y albergan los Pods que componen la carga de trabajo de la aplicación.
- Funcionamiento: Los nodos trabajadores ejecutan las aplicaciones en forma de Pods y son la parte operativa del clúster donde se ejecuta la lógica de negocio de las aplicaciones.

### *Plano de Control (Control Plane):*

- Definición: La parte de Kubernetes que toma decisiones globales sobre el clúster y responde a eventos del clúster, como la programación de Pods.
- Funcionamiento: El plano de control se encarga de gestionar los nodos trabajadores y los Pods en el clúster. En entornos de producción, el plano de control generalmente se ejecuta en múltiples computadoras para proporcionar tolerancia a fallos y alta disponibilidad.

***Componentes del Plano de Control***

kube-apiserver:

- Definición: El servidor de API es un componente del plano de control de Kubernetes que expone la API de Kubernetes. Es la interfaz principal para el plano de control de Kubernetes.
- Funcionamiento: El kube-apiserver escala horizontalmente, lo que significa que puedes ejecutar varias instancias de kube-apiserver y equilibrar el tráfico entre esas instancias. Es el punto de entrada para realizar operaciones en el clúster.

etcd:

1. Definición: Almacén de clave-valor consistente y altamente disponible utilizado como almacenamiento de respaldo para todos los datos del clúster Kubernetes.
2. Funcionamiento: etcd garantiza que los datos del clúster sean coherentes y estén siempre disponibles. Es fundamental para la persistencia y la coherencia del estado del clúster.

kube-scheduler:

- Definición: Componente del plano de control que observa los Pods recién creados sin un nodo asignado y selecciona un nodo para ejecutarlos.
- Funcionamiento: El kube-scheduler toma decisiones de programación basadas en requisitos de recursos individuales y colectivos, restricciones de hardware/software/políticas, especificaciones de afinidad y anti-afinidad, localidad de datos, interferencia entre cargas de trabajo y plazos.

kube-controller-manager

- Definición: Componente del plano de control que ejecuta procesos de controladores.

- Funcionamiento:Cada controlador es lógicamente un proceso separado, pero para reducir la complejidad, todos se compilan en un solo binario y se ejecutan en un solo proceso.

Tipos de Controladores:

- Node Controller (Controlador de Nodos):
    - Responsabilidad: Detecta y responde cuando los nodos dejan de funcionar.
- Job Controller (Controlador de Trabajos):
    - Responsabilidad: Observa objetos de tipo Job que representan tareas únicas, luego crea Pods para ejecutar esas tareas hasta su finalización.
- EndpointSlice Controller (Controlador de EndpointSlice):
    - Responsabilidad: Puebla objetos de tipo EndpointSlice (para establecer un vínculo entre los Servicios y los Pods).
- ServiceAccount Controller (Controlador de ServiceAccount):
    - Responsabilidad: Crea ServiceAccounts predeterminadas para nuevos espacios de nombres (namespaces).
- Nota: Esta lista no es exhaustiva; hay muchos otros controladores disponibles.

cloud-controller-manager

- Definición: Componente del plano de control de Kubernetes que incorpora lógica de control específica de la nube.
- Funcionamiento: El cloud-controller-manager permite vincular tu clúster con la API específica de tu proveedor de servicios en la nube, separando los componentes que interactúan con esa plataforma en la nube de los componentes que solo interactúan con tu clúster.

Solo ejecuta controladores específicos para tu proveedor de servicios en la nube. Si estás ejecutando Kubernetes en tus propias instalaciones o en un entorno de aprendizaje en tu PC, el clúster no tendrá un cloud-controller-manager.

Al igual que con kube-controller-manager, el cloud-controller-manager combina varios bucles de control lógicamente independientes en un solo binario que se ejecuta como un único proceso. Puedes escalar horizontalmente (ejecutar más de una copia) para mejorar el rendimiento o tolerar fallas.

Controladores con Dependencias en Proveedores de Servicios en la Nube:

- Node Controller (Controlador de Nodos):
    - Dependencia: Verifica el proveedor de servicios en la nube para determinar si un nodo ha sido eliminado en la nube después de que deja de responder.
- Route Controller (Controlador de Rutas):
    - Dependencia: Configura rutas en la infraestructura de nube subyacente.
- Service Controller (Controlador de Servicios):
    - Dependencia: Crea, actualiza y elimina balanceadores de carga proporcionados por el proveedor de servicios en la nube.

***Componentes del Nodo (Node Components)***

Los componentes del nodo se ejecutan en cada nodo, manteniendo los pods en ejecución y proporcionando el entorno de ejecución de Kubernetes.

kubelet:

- Definición: Agente que se ejecuta en cada nodo del clúster. Se asegura de que los contenedores se estén ejecutando en un Pod.

- Funcionamiento: El kubelet toma un conjunto de PodSpecs proporcionados a través de varios mecanismos y garantiza que los contenedores descritos en esos PodSpecs estén en ejecución y saludables. El kubelet no gestiona contenedores que no fueron creados por Kubernetes.

kube-proxy:

- Definición: Proxy de red que se ejecuta en cada nodo en tu clúster, implementando parte del concepto de Servicio de Kubernetes.
- Funcionamiento: Mantiene reglas de red en los nodos que permiten la comunicación de red a tus Pods desde sesiones de red dentro o fuera de tu clúster. Utiliza la capa de filtrado de paquetes del sistema operativo si está disponible; de lo contrario, kube-proxy reenvía el tráfico él mismo.

Container Runtime (Tiempo de Ejecución de Contenedores):

- Definición: Componente fundamental que permite a Kubernetes ejecutar contenedores de manera efectiva. Se encarga de gestionar la ejecución y el ciclo de vida de los contenedores dentro del entorno de Kubernetes.
- Funcionamiento: Kubernetes admite runtimes de contenedores como containerd, CRI-O y cualquier otra implementación de la Interfaz de Runtimes de Contenedores (CRI) de Kubernetes.

Addons:

- Definición: Los addons utilizan recursos de Kubernetes (DaemonSet, Deployment, etc.) para implementar características del clúster. Pertenecen al espacio de nombres kube-system porque proporcionan funciones a nivel de clúster.

Algunos Addons Selecionados:

- DNS:
    - Función: Servidor DNS adicional que proporciona registros DNS para servicios de Kubernetes. Los contenedores iniciados por Kubernetes incluyen automáticamente este servidor DNS en sus búsquedas de DNS.
- Web UI (Dashboard):
    - Función: Interfaz de usuario basada en web para clústeres de Kubernetes. Permite a los usuarios gestionar y solucionar problemas de aplicaciones en ejecución en el clúster, así como el clúster en sí.
- Monitorización de Recursos de Contenedores:
    - Función: Registra métricas genéricas de series temporales sobre contenedores en una base de datos central y proporciona una interfaz de usuario para explorar esos datos.
- Registro a Nivel de Clúster:
    - Función: Mecanismo de registro a nivel de clúster que guarda los registros de contenedores en un almacén de registros central con una interfaz de búsqueda/navegación.
- Plugins de Red:
    - Función: Componentes de software que implementan la especificación de la Interfaz de Red de Contenedores (CNI). Son responsables de asignar direcciones IP a los pods y permitirles comunicarse entre sí dentro del clúster.

**Docker y sus Componentes**

***Docker:***

- **Definición**: Docker es una plataforma de código abierto que permite el desarrollo, envío y ejecución de aplicaciones en contenedores. Los contenedores son entornos ligeros y portátiles que incluyen todo lo necesario para ejecutar una aplicación, incluidas bibliotecas, dependencias y código.

***Creación de Imágenes***

- **Proceso de Creación de Imágenes Docker:** Se realiza a través de la definición de un archivo llamado Dockerfile, que especifica los pasos necesarios para construir la imagen. Esto incluye la selección de la imagen base, la instalación de dependencias y la configuración del entorno de ejecución.
- **Uso de Dockerfile y Mejores Prácticas**: El Dockerfile es una receta para construir una imagen. Se deben seguir mejores prácticas, como minimizar capas, utilizar imágenes base seguras y solo incluir lo esencial en la imagen.

***Gestión de Contenedores***

Instrucciones para Ejecutar, Detener y Eliminar Contenedores:

- Para ejecutar un contenedor:

```
docker run [opciones] nombre_imagen.
```

- Para detener un contenedor en ejecución:

```
docker stop ID_o_nombre_contenedor.
```

- Para eliminar un contenedor:

```
docker rm ID_o_nombre_contenedor.
```

Interacción con el Sistema de Archivos del Contenedor:

Se puede acceder al sistema de archivos del contenedor mediante el comando para abrir un terminal interactivo.

```
docker exec -it ID_o_nombre_contenedor bash
```

***Redes y Volúmenes***

Configuración de Redes y Volúmenes:

Para la comunicación entre contenedores, se pueden crear redes utilizando docker network. Para persistir datos, se utilizan volúmenes, que pueden ser montados en los contenedores.

***Docker Compose***

Uso de Docker Compose para Definir y Gestionar Aplicaciones Multicontenedor:

Docker Compose permite definir y gestionar aplicaciones compuestas por varios contenedores. Se utiliza un archivo YAML para describir la configuración de la aplicación, incluyendo servicios, redes y volúmenes.

***Seguridad***

Prácticas Recomendadas para la Seguridad en Docker:

Se deben seguir prácticas como escanear imágenes en busca de vulnerabilidades, limitar privilegios, y asegurarse de que solo se ejecuten comandos necesarios en los contenedores.

Configuración de Privilegios y Control de Acceso:

Docker permite configurar privilegios específicos para los contenedores y controlar el acceso a recursos del sistema.

### *Escenarios Comunes*

Ejemplos Prácticos de Uso Común de Docker:

Incluyen la creación de entornos de desarrollo reproducibles, despliegue de aplicaciones en producción, y el uso de imágenes oficiales de Docker Hub para servicios como bases de datos, servidores web, etc.

Docker proporciona un entorno flexible y eficiente para el desarrollo y despliegue de aplicaciones, permitiendo la creación de entornos consistentes y la gestión sencilla de contenedores.

**Contratos de Microservicios**

***Acuerdo de Nivel de Servicio y Términos - Docker***

Horario de Operación:

El soporte de Docker está disponible de lunes a viernes, de 8:00 a 16:00 CDT.

Detalles del SLA según la Suscripción:

Los siguientes tiempos de SLA representan los marcos temporales para la respuesta inicial del soporte de Docker. Si bien se hará todo lo posible para resolver los problemas de manera rápida, estos tiempos de SLA no deben interpretarse como el tiempo esperado para la resolución.

Personal:

- *Problemas con facturación, compras, suscripciones:* Contactar con Soporte.
- *Otros problemas:* Foros de Docker o la Comunidad de Docker en Slack.

Pro:

- *Tiempo de Respuesta:* 5 días hábiles
- *Contactar con Soporte*

Team:

- *Tiempo de Respuesta:* 2 días hábiles
- *Contactar con Soporte*

Business:

- *Tiempo de Respuesta:* 1 día hábil
- *Contactar con Soporte*

Soporte para Docker Desktop:

Para clientes Pro, Team o Business, se puede solicitar soporte para los siguientes problemas relacionados con el uso de Docker Desktop:

- Problemas de actualización de Desktop
- Problemas de instalación de Desktop
- Habilitar la virtualización en la BIOS
- Habilitar características de Windows
- Cierres inesperados durante la instalación

Docker se compromete a proporcionar respuestas oportunas y esfuerzos de resolución dentro de los tiempos de SLA especificados. Se recomienda a los clientes que se pongan en contacto con el canal de soporte correspondiente a su nivel de suscripción para una eficiente resolución de problemas. Es importante tener en cuenta que los tiempos de SLA representan los tiempos de respuesta iniciales y no el tiempo esperado para la resolución completa.

***Acuerdo de Nivel de Servicio (SLA) de Google Kubernetes Engine***

Durante el plazo del acuerdo mediante el cual Google se compromete a proporcionar Google Cloud Platform a Customer (según corresponda, el "Acuerdo"), el Servicio Cubierto ofrecerá un Porcentaje de Disponibilidad Mensual a Customer de la siguiente manera (el "Objetivo del Nivel de Servicio" o "SLA"):

| **Servicio Cubierto** | **Porcentaje de Disponibilidad Mensual** |
|---|---|
| Zonal Cluster (plano de control) | 99.5% |
| Regional Cluster (plano de control) | 99.95% |

| | |
|---|---|
| Autopilot Cluster (plano de control) | 99.95% |
| Autopilot Pods en Múltiples Zonas | 99.9% |

Si Google no cumple con el SLA y si Customer cumple con sus obligaciones según este SLA, Customer será elegible para recibir los Créditos Financieros descritos a continuación. El Porcentaje de Disponibilidad Mensual y el Crédito Financiero se determinan mensualmente por clúster. Este SLA establece el único y exclusivo remedio de Customer por cualquier incumplimiento de Google con respecto al SLA. Los términos capitalizados utilizados en este SLA, pero no definidos en este SLA, tienen el significado establecido en el Acuerdo.

**Definiciones:**

- ***Autopilot Pods en Múltiples Zonas:*** Capacidad informática provista por el Servicio Autopilot de Google Kubernetes Engine para programar pods de usuario, donde los pods se programan en dos o más Zonas en la misma Región.
- ***Servicio Cubierto:*** Autopilot Pods en Múltiples Zonas; o, para cada Zonal Cluster (plano de control), Regional Cluster (plano de control) y Autopilot Cluster (plano de control), la API de Kubernetes proporcionada por los clústeres de Customer, siempre y cuando la versión de Google Kubernetes Engine desplegada en el clúster sea una versión actualmente ofrecida en los Canales Estable o Regular.
- ***Downtime:*** Pérdida de conectividad del plano de control de Autopilot a todas las instancias de pod en ejecución aplicables. Este Downtime no incluye problemas relacionados con el equilibrio de carga y el túnel VPN, que están cubiertos por los SLA respectivos de Compute Engine y Cloud VPN.

- ***Período de Downtime:*** Período de cinco minutos o más de Downtime consecutivo. El Downtime intermitente por un período de menos de cinco minutos no se contará como parte de ningún Período de Downtime.
- ***Crédito Financiero:*** Porcentaje de la factura mensual del Servicio Cubierto que no cumple con el SLA y que se acreditará en las facturas mensuales futuras de Customer.

Crédito Financiero Máximo:

El número máximo de Créditos Financieros emitidos por Google a Customer por todos los Períodos de Downtime en un solo mes de facturación no superará el 50% del monto adeudado por Customer por el Servicio Cubierto para el mes correspondiente.

Exclusiones del SLA:

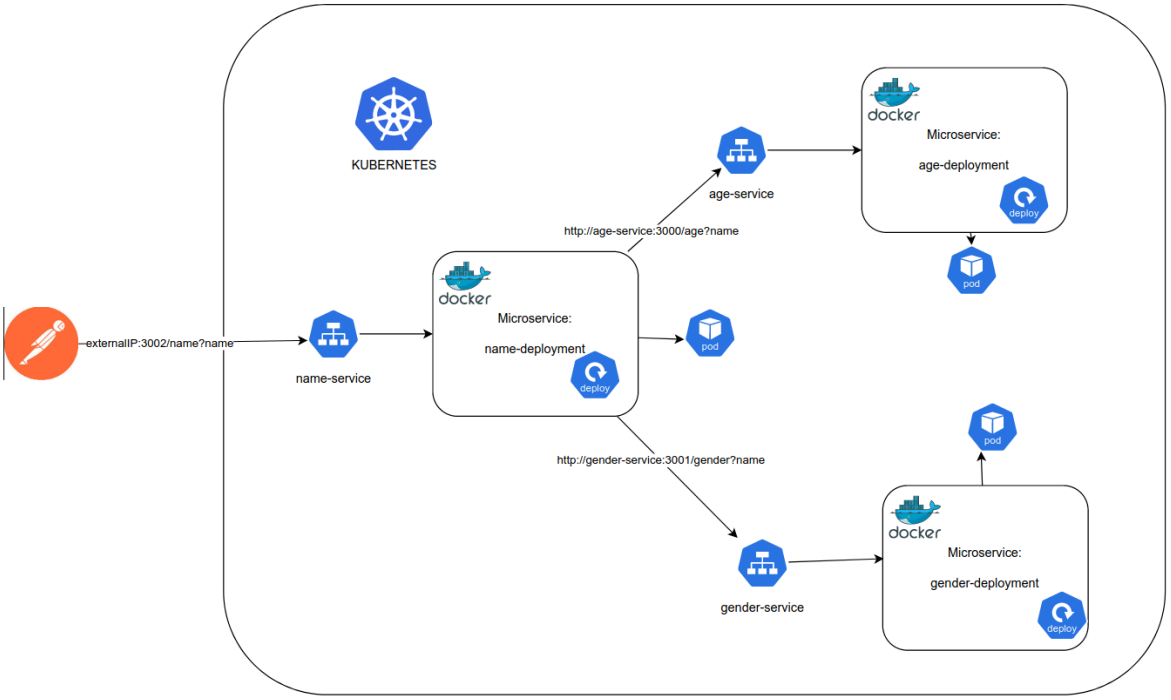
El SLA no se aplica a ninguna (a) características designadas como pregenerales (a menos que se indique lo contrario en la Documentación asociada); (b) características excluidas del SLA (en la Documentación asociada); (c) clústeres de Customer donde la versión desplegada de Google Kubernetes Engine no se ofrece a través de los Canales Estable o Regular; o (d) errores: (i) causados por factores fuera del control razonable de Google; (ii) que resultaron del software o hardware de Customer o de terceros, o ambos; (iii) que resultaron de abusos u otros comportamientos que violan el Acuerdo; o (iv) que resultaron de cuotas aplicadas por el sistema o enumeradas en la Consola de Administración.

Customer Debe Solicitar el Crédito Financiero:

Para recibir cualquiera de los Créditos Financieros descritos anteriormente, Customer debe notificar al soporte técnico de Google dentro de los 30 días desde el momento en que Customer sea elegible para recibir un Crédito Financiero. Customer también debe proporcionar a Google archivos de registro del servidor que muestren errores de pérdida de

conectividad externa y la fecha y hora en que ocurrieron esos errores. Si Customer no cumple con estos requisitos, perderá el derecho a recibir un Crédito Financiero.

## Aplicaciones a la Práctica




La aplicación consta de 3 microservicios:

1. **name-deployment:**
    a. Sirve de middleware, consume age-deployment y gender-deployment mediante los services asociados, mediante la red interna.
    b. **Services:** tiene un service asociado que expone un puerto externo.
        i. name-service
    c. **Pods:** de forma implícita contiene un pod (de momento por cuestiones financieras).
    d. **Contenedores:** apunta a un contenedor docker que contiene el codigo fuente, la imagen docker está en “node_microservice_name” en Container Registry de GCP.
2. **age-deployment:**
    a. Servicio consumida por name-deployment y name-service, mediante la red interna expuesta por su service asociado.

b. **Services:** tiene un service asociado que expone un puerto interno posteriormente a usar por name-deployment.
    i. age-service
c. **Pods:** de forma implícita contiene un pod (de momento por cuestiones financieras).
d. **Contenedores:** apunta a un contenedor docker que contiene el codigo fuente, la imagen docker está en “node_microservice_name” en Container Registry de GCP.

3. **gender-deployment:**
    a. Servicio consumida por name-deployment y name-service, mediante la red interna expuesta por su service asociado.
    b. **Services:** tiene un service asociado que expone un puerto interno posteriormente a usar por name-deployment.
        i. gender-service
    c. **Pods:** de forma implícita contiene un pod (de momento por cuestiones financieras).
    d. **Contenedores:** apunta a un contenedor docker que contiene el codigo fuente, la imagen docker está en “node_microservice_name” en Container Registry de GCP.

## Bibliografía

*Glossary*. (n.d.). Kubernetes. Recuperado el 1 de marzo de 2024, desde: https://kubernetes.io/docs/reference/glossary/?all=true#term-control-plane

*Google Kubernetes Engine Service Level Agreement (SLA)*. (n.d.). Google Cloud. Recuperado el 1 de marzo de 2024, desde: https://cloud.google.com/kubernetes-engine/sla-20210224

*Kubernetes components*. (n.d.). Kubernetes. Recuperado el 1 de marzo de 2024, desde: https://kubernetes.io/docs/concepts/overview/components/

*Support*. (2022, March 3). Docker. https://www.docker.com/support/