Computación Tolerante a Fallas

Sección D06

Proyecto v1

Fernández Venegas David Guadalupe

Código: 216437352

Profesor: López Franco Michel Emanuel

Se busca realizar un proyecto para la materia de Computación Tolerante a Fallas. Este se basa en una serie de principios y prácticas que buscan garantizar la disponibilidad y confiabilidad de una aplicación incluso en presencia de fallos:

1. División en Servicios Independientes: Este enfoque arquitectónico promueve la modularidad y la independencia de los componentes de la aplicación. Cada microservicio es responsable de una tarea específica y se comunica con otros a través de interfaces bien definidas. Esto facilita el desarrollo, el mantenimiento y la escalabilidad de la aplicación.
2. Contenerización con Docker: La contenerización con Docker permite encapsular cada microservicio junto con sus dependencias en un contenedor ligero y portable. Esto garantiza que cada servicio pueda ejecutarse de manera consistente y aislada en cualquier entorno, lo que facilita el despliegue y la gestión de la aplicación.
3. Orquestación con Kubernetes: Kubernetes es una herramienta poderosa para la gestión de contenedores a escala. Permite automatizar tareas como el despliegue, el escalado y la recuperación ante fallos de los microservicios. Con Kubernetes, se puede garantizar que la aplicación se ejecute de manera eficiente y resiliente en cualquier entorno.
4. Comunicación entre Microservicios: La comunicación entre microservicios es fundamental en una arquitectura basada en servicios. Se pueden utilizar diferentes mecanismos, como APIs REST, mensajería asíncrona o llamadas remotas, dependiendo de los requisitos de la aplicación y las preferencias del equipo de desarrollo.
5. Monitorización y Observabilidad: La monitorización y observabilidad son aspectos críticos para entender el comportamiento de la aplicación y detectar problemas de manera proactiva. Herramientas como Istio o Azure Monitor permiten recopilar métricas, registros y trazas que pueden utilizarse para optimizar el rendimiento y la confiabilidad de la aplicación.
6. Automatización CI/CD: La automatización del proceso de construcción, pruebas e implementación de los microservicios mediante herramientas de CI/CD garantiza una entrega rápida y confiable de la aplicación. Esto reduce el tiempo de comercialización y minimiza el riesgo de errores humanos en el proceso de implementación.
7. Diseño Escalable y Resiliente: El diseño de la arquitectura para que los microservicios sean escalables y resilientes es fundamental para garantizar la disponibilidad de la aplicación. Esto implica diseñar estrategias de escalado horizontal y vertical, así como implementar mecanismos de recuperación ante fallos, como la redundancia y la tolerancia a fallos.
8. Implementación de Seguridad: La seguridad es un aspecto crítico en cualquier aplicación, especialmente en un entorno distribuido como el de microservicios. Es

importante implementar mecanismos de autenticación, autorización y cifrado de comunicaciones para proteger los datos y garantizar la integridad de la aplicación.

9. Ingeniería del Caos: La ingeniería del caos es una práctica que consiste en inyectar intencionadamente fallos en un sistema para probar su resistencia y capacidad de recuperación. Esto permite identificar y mitigar posibles puntos débiles en la arquitectura y mejorar la confiabilidad de la aplicación a largo plazo.

Herramientas a Utilizar

Rancher:

Rancher es una plataforma de gestión de contenedores open-source que facilita la implementación y administración de entornos basados en contenedores Docker y Kubernetes. Ofrece una interfaz de usuario intuitiva y potentes capacidades de automatización para simplificar tareas complejas de implementación, escalado y gestión de aplicaciones en entornos de contenedores.

Características principales:

1. Gestión de Kubernetes: Rancher proporciona una interfaz unificada para gestionar clústeres de Kubernetes, permitiendo la creación, configuración y supervisión de clústeres de Kubernetes de forma sencilla.
2. Implementación Multi-Clúster: Permite administrar múltiples clústeres de Kubernetes desde una sola instancia de Rancher, lo que facilita la gestión de entornos distribuidos y heterogéneos.
3. Catálogo de Aplicaciones: Incluye un catálogo de aplicaciones preconfiguradas que pueden ser desplegadas con unos pocos clics. Esto facilita la implementación de aplicaciones comunes y acelera el proceso de desarrollo.
4. Gestión de Recursos: Ofrece herramientas para gestionar recursos de infraestructura, como almacenamiento, redes y balanceadores de carga, de manera centralizada y automatizada.
5. Monitorización y Alertas: Proporciona capacidades integradas de monitorización y generación de alertas para supervisar el estado y el rendimiento de los clústeres de Kubernetes y las aplicaciones desplegadas.
6. Gestión de Usuarios y Accesos: Permite configurar políticas de acceso y control de usuarios para garantizar la seguridad y la segregación de roles en la plataforma.

7. Automatización y CI/CD: Integra con herramientas de CI/CD para automatizar el proceso de construcción, pruebas e implementación de aplicaciones en entornos de contenedores.
8. Escalado Automático: Permite configurar políticas de escalado automático basadas en métricas de rendimiento para ajustar dinámicamente el número de réplicas de una aplicación en función de la demanda.

Rancher Desktop:

Rancher Desktop es una aplicación de escritorio de código abierto que proporciona Kubernetes, gestión de contenedores y utilidades integradas en el escritorio. Con Rancher Desktop, puede ejecutar un clúster K3s ligero dentro de una máquina virtual. La aplicación también incluye los entornos de ejecución de contenedores containerd y dockerd.

Con Rancher Desktop, puede hacer lo siguiente:

- Configurar Kubernetes usando la interfaz de usuario sencilla de Rancher Desktop.
- Elegir la versión de Kubernetes que desea utilizar.
- Elegir su entorno de ejecución de contenedores preferido.
- Configurar los recursos del sistema para la máquina virtual (en Mac y Linux).
- Reiniciar Kubernetes o el entorno de ejecución de contenedores a su configuración predeterminada con solo presionar un botón.

Istio:

Istio es una plataforma de servicio de malla (Service Mesh) de código abierto que proporciona un control de tráfico sofisticado, seguridad y observabilidad para aplicaciones distribuidas que se ejecutan en entornos de contenedores Kubernetes u otros entornos de orquestación de contenedores. Istio ayuda a resolver desafíos comunes en arquitecturas de microservicios, como la gestión del tráfico, la seguridad de las comunicaciones y la observabilidad del sistema.

Características principales:

1. Control de Tráfico: Istio permite definir reglas de enrutamiento y políticas de tráfico para dirigir el tráfico de red entre diferentes versiones de servicios, implementar canarios y realizar pruebas A/B de manera segura y controlada.

2. Balanceo de Carga: Proporciona capacidades avanzadas de balanceo de carga para distribuir de manera equitativa y eficiente las solicitudes entre las instancias de un servicio, mejorando la disponibilidad y el rendimiento de la aplicación.
3. Seguridad de la Comunicación: Istio facilita la implementación de políticas de seguridad de red como el cifrado de extremo a extremo (mTLS), la autenticación basada en JWT y el control de acceso granular a través de políticas de autorización.
4. Gestión del Tráfico Seguro: Permite aplicar circuitos cerrados y control de errores para gestionar de forma segura situaciones de falla y sobrecarga en la red, minimizando el impacto en la experiencia del usuario.
5. Observabilidad: Istio recopila métricas, logs y trazas de las comunicaciones entre servicios, lo que facilita la detección de problemas de rendimiento, la resolución de problemas y la optimización de la infraestructura de red.

Kiali:

Kiali es una interfaz de usuario gráfica y de código abierto que proporciona una visión en tiempo real de la topología de la aplicación, el tráfico de red y el estado de la malla de servicios gestionada por Istio. Kiali facilita la comprensión y el análisis de la arquitectura de microservicios, permitiendo a los equipos de desarrollo y operaciones visualizar, analizar y solucionar problemas de manera eficiente.

Características principales:

1. Visualización de la Topología: Kiali muestra de forma visual la topología de la aplicación, incluyendo los servicios, las relaciones entre ellos y el flujo de tráfico de red, lo que facilita la comprensión de la arquitectura y las dependencias de la aplicación.
2. Monitorización en Tiempo Real: Proporciona métricas en tiempo real sobre el tráfico de red, el rendimiento de los servicios y el estado de la malla de servicios, lo que permite detectar problemas y tomar medidas correctivas de manera proactiva.
3. Análisis de Rendimiento: Permite analizar el rendimiento de los servicios y las comunicaciones entre ellos, identificando cuellos de botella, latencias elevadas y otros problemas que puedan afectar la experiencia del usuario.
4. Generación de Grafos: Kiali genera gráficos dinámicos y personalizables que representan la topología de la aplicación y el flujo de tráfico de red, lo que facilita el análisis y la resolución de problemas en entornos complejos de microservicios.

Istio y Kiali son herramientas poderosas que trabajan en conjunto para proporcionar control de tráfico, seguridad y observabilidad en entornos de microservicios basados en Kubernetes, permitiendo a los equipos de desarrollo y operaciones gestionar y optimizar de manera eficiente sus aplicaciones distribuidas.