AS2 : Proyecto 2 (Parte 3 de 3)

Objetivo

Diseñar y desplegar un servicio de maestro Kubernetes en alta disponibilidad, un servicio de monitorización y alertas con Prometheus y un sistema Dashboard para Kubernetes.

Evaluación

Para esta segunda parte se deberá **entregar una memoria**, de un **máximo de 6 paginas,** en la que, al menos, se incluya:

1.- Resumen
2.- Arquitectura de elementos relevantes del despliegue Kubernetes Aplicaciones.
3.- Explicación de las diferentes elementos desplegados sobre Kubernetes con la explicación de los conceptos y los recursos Kubernetes utilizados y explicación básica de los conceptos y recursos utilizados por Ceph.
5.- Explicación de los métodos utilizados para la validación de la operativa.
4.- Problemas encontrados y su solución.

Además, se deberá mostrar el funcionamiento del despliegue manual, concertando cita previa con el profesor.

Fecha límite de entrega de memoria : 22 de junio de 2020.

Enunciado

En esta 3ª parte del proyecto se plantean 3 aspectos :

- Diseño, configuración y despliegue de los elementos necesarios para obtener un servicio maestro de Kubernetes con tolerancia de fallo de 1 nodo maestro sin para el servicio.
- Poner en marcha un solución de monitorización y alertas del cluster Kubernetes con Prometheus, y el sistema Dashboard que os parezca más adecuado, entre los disponibles para Kubernetes. Ambos mediante Helm Charts.

Utilizar el mínimo de VMs necesario para validar estas soluciones.

Definir los requerimientos de recursos que necesiteis (RAM, disco, CPU, etc). Podeis utilizar elementos de las 2 primeras 2 partes del proyecto, o planteados a lo largo de la asignatura. Estimar, previamente el tiempo necesario para la solución que planteeis.

Para la alta disponibilidad en K3S, teneis diferentes backends disponibles, incluida la opción oficial de Kubernetes, Etcd :

https://rancher.com/docs/k3s/latest/en/installation/ha/

Tener especial cuidado con la sección 3 (Configure the Fixed Registration Address), Los workers (o agents en K3s) necesitan tener un contacto estable con el servicio maestro, independientemente d la dirección IP de cada una de sus réplicas. En esta ocasión, no merece la pena complicarse on uan balanceador de carga de nivel 4 (transporte), ya que debería estar también replicado. Os sugiero la opción "Virtual IP address". Tener en cuenta el servicio kube-API.

Para gestionar la IP Virtual entre las réplicas de maestros Kubernetes podeis utilizar la herramienta Pacemaker/Corosync que hemos visto en clase . Podeis inspiraros de :

https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/8/html/configuring_and_managing_high_availability_clusters/index

También teneis la posibilidad de utilizar DQLite como backend altamente disponible en lugar de Etcd :

https://rancher.com/docs/k3s/latest/en/installation/ha-embedded/

Opcional

1. Diseñar y desplegar una solución DNS tolerante a fallos en y para Kubernetes.

2. Realizar la misma solución planteada en este guión, pero sobre Cloud público como Azure, Amazon, Google, IBM u otros, mediante Packer y Terraform u otras herramientas que considereis oportunas.

3. Otras propuestas complementarias que considereis de interés.