Anteproyecto TFG

**Despliegue de Kubernetes con Ansible en entornos offline**

Autor: Sergio Picazo Serrano

Tutor: Óscar García Población

Titulación: Grado en ingeniería en sistemas de información

Visto bueno del cotutor (en su caso): David de la Hoz Alías

# DEFINICIÓN DEL ÁMBITO DEL PROYECTO

**- 1 Introducción y objetivos**

- 1.1 Introducción

- 1.2 Motivación

- 1.3 Objetivo

- 1.4 Estructura del documento

**- 2 Antecedentes**

- 2.1 Contenedores

- 2.2 Kubernetes

- 2.3 Ansible

**- 3 Desarrollo**

- 3.1 Prerrequisitos y preparación del entorno

- 3.2 Planificación de uso de recursos (Requisitos de hardware y software)

- 3.2.1 Creación de máquinas independientes (setup-host.sh)

- 3.3 Instalación y configuración de Ansible

- 3.3.1 Creación de playbooks de Ansible

- 3.3.2 Definición de las tareas y roles necesarios

- 3.3.3 Configuración de los hosts y los grupos

- 3.3.4 Creación de variables y templates

- 3.3.5 Instalación de common necesarios

- 3.3.6 Despliegue del clúster de Kubernetes

- 3.3.7 Instalación de los componentes del clúster(master,workers, bastión, registry, loadbalancer, etcd)

- 3.3.8 Configuración de los nodos y los componentes

**- 4 Resultados**

- 4.1 Verificación del funcionamiento del clúster (gestionarlo con Rancher Desktop por ejemplo)

- 4.2 Demostración

**- 5 Conclusiones**

- 5.1 Ventajas de utilizar Ansible para montar un cluster de Kubernetes

**- 6 Futuras tendencias y vista a futuro**

- 6.1 Otras formas de instalar en k8s con Helm

**- 7 Bibliografía**

# INTRODUCCIÓN

En este Trabajo de Fin de Grado se va a montar un clúster de Kubernetes mediante automatización con Ansible en un entorno offline montado on premise.

A diferencia de las máquinas virtuales, los contenedores permiten desplegar, arrancar y parar aplicaciones más rápido, aprovechando mejor los recursos de hardware.

La motivación de este proyecto viene por el papel que están teniendo últimamente las tecnologías Cloud Native, en especial Kubernetes debido a que ayuda a las empresas a crear, escalar y administrar aplicaciones en la nube y mantener sus ciclos de vida dinámicos.

Según un artículo de Dominio de las Ciencias [1], se demuestra como actualmente, la manera más eficiente para el desarrollo y puesta en producción de aplicaciones es la implementación de microservicios contenerizados, orquestados en Kubernetes, y ya no los procesos tradicionales a través de monolitos.

Kubernetes permite el desarrollo y puesta en producción de aplicaciones de manera más rápida y con propiedades de escalamientos y alta disponibilidad.

La conclusión es evidente, desde su nacimiento está ganando una adopción masiva en todas las industrias, ayudando a las empresas a ofrecer soluciones de software con menos énfasis en la infraestructura.

Uno de los casos más destacados es el de Mercedes-Benz [2], hace años, los ingenieros de software se enfrentaban a tiempos difíciles en Mercedes-Benz: operaciones con hojas de cálculo, procesos manuales, infraestructuras crecidas y una gobernanza estricta. Una iniciativa popular de los ingenieros propuso el reto de cambiar las reglas del juego, y su bala de plata fue Kubernetes. Comenzaron con Kubernetes 0.9 en servidores gestionados hasta el día de hoy, donde manejan una plataforma on-premise self-service con cerca de 1000 clústeres en Cluster API. Apostaron por transformar un centro de datos con un equipo joven que, en su mayoría, desconocía los procesos empresariales, pero a través de una mezcla de visiones ingenuas y una fuerte creencia en el código abierto con mucha resistencia hizo que el proyecto fuera un éxito.

La clave de este trabajo reside en la necesidad de conseguir, de la manera más eficiente posible, aprovechando al máximo unos recursos limitados, desplegar un entorno en el que el mantenimiento de este y la gestión de aplicaciones sea lo más sencilla posible.

Esto se debe a que antiguamente la solución era mantener tus servicios y aplicaciones en un solo ordenador físico, llevando a obtener muchos problemas a las empresas.

Los contenedores nacieron para simplificar este proceso y ofreciendo virtualización ligera, generan el entorno mínimo necesario para aprovechar en mayor parte los recursos de la máquina física donde se ejecuta.

# OBJETIVOS DEL TRABAJO DE FIN DE GRADO Y CAMPO DE APLICACIÓN

Este proyecto tiene como objetivo el diseño y construcción de una instalación automatizada de un clúster de Kubernetes, con aprovisionamiento automático, en un entorno de alta seguridad, en el que la conectividad a Internet es limitada o nula.

El objetivo al finalizar el proyecto es ser capaz de ofrecer un entorno totalmente funcional de laboratorio en el que se puedan hacer pruebas que conlleven riesgos y donde se puedan probar todo tipo de instalaciones de aplicativos para seguir aprendiendo sobre dicha plataforma, por tanto, un entorno de laboratorio es su principal campo de aplicación.

# ENTREGABLES

Como resultado de este proyecto tendremos:

- Repositorio en GitHub con el código fuente

- Plantilla de cookiecutter para la personalización del despliegue

- Repositorio con las dependencias necesarias para operar offline

- Demostración del clúster, recursos y funcionalidades

- Documento con el análisis y diseño propuestos, así como con los pasos seguidos para la elaboración

# DESCRIPCIÓN DE LAS TAREAS

(Explicar en detalle qué se va a hacer, cómo y por qué. Sería interesante incluir un esquema/diagrama de bloques, si se puede.)

Con el fin de llegar al objetivo final de este proyecto, la instalación automatizada del clúster, se llevarán a cabo diversas tareas.

Por tanto, debemos familiarizarnos con estas nuevas tecnologías y habrá que seguir los siguientes objetivos paso a paso.

Estudio sobre sistemas de uso de contenedores como Docker y aprender sobre el uso de herramientas de virtualización, automatización y IAC(infraestructura como código).

Dar a conocer las ventajas de los contenedores sobre otras tecnologías y sistemas de orquestación de contenedores, en este caso Kubernetes[3].

Identificar ventajas del empleo de estas plataformas orquestadoras.

Tras haber logrado entender el funcionamiento de estas tecnologías habrá que descubrir las posibles formas de instalar estas plataformas. Por ello habrá que:

- Aprender a desplegar un entorno de orquestación de contenedores llevando a cabo la configuración necesaria

Al mismo tiempo que se elaboran estas tareas, se efectuará la tarea de las pruebas, muy importante en el ciclo de desarrollo de software. Finalmente, se redactará la memoria con los resultados obtenidos.

Con la tecnología entendida e instalada, es decir, ya con la infraestructura en funcionamiento, se comprobará si el funcionamiento es el esperado. Por tanto:

- Estudiar que el funcionamiento es el correcto tras todo el despliegue.

- Realizar la conclusión de lo analizado.

Tras el análisis se llevará a cabo una conclusión de los resultados y de lo aprendido en el desarrollo del proyecto.

Simultáneamente, se irán efectuando las respectivas pruebas del código implementado. Al crear el video, se pueden ir modificando elementos tanto a nivel estético como a nivel funcional.

Por último, se redactará la memoria plasmando los resultados obtenidos en ella. En esta se detallará el diseño de las soluciones adoptadas.

# METODOLOGÍA Y PLAN DE TRABAJO

Para proceder a la realización del trabajo de fin de grado se ha optado por dividirlo en diferentes fases para poder llevar un control y seguimiento de las tareas de manera específica. El total de horas de trabajo será de 350 aproximadamente.

El trabajo se ha dividido en semanas y no en días para una mayor libertad en la consecución de los objetivos. Cada semana comprenderá 17.5 horas de trabajo.

De cara a la consecución de los objetivos del proyecto que se han descrito anteriormente, estas serán las fases a seguir:

1. \*\*Formación inicial (5 semanas):\*\*

\* \*\*Formación en el uso de Ansible, Docker y Kubernetes (3 semanas):\*\* Durante esta fase, se obtendrán los conocimientos que permitan un manejo óptimo de la herramienta. Para ello se tomarán como ayuda la documentación proporcionada por la propia herramienta y las guías de inicio.

\* \*\*Consulta bibliográfica y de la API (2 semanas):\*\* Recopilación de la información necesaria tanto de la comunidad como de la API.

2. \*\*Análisis y diseño del entorno (4 semanas):\*\*

\* \*\*Análisis de los requisitos y arquitecturas(4 semanas):\*\* Se llevará a cabo un análisis de los requerimientos de cada una de las máquinas que formarán el clúster con planteamiento de las arquitecturas que sean necesarias para la posterior implementación de estas.

3. \*\*Implementación y desarrollo de las configuraciones (10 semanas):\*\*

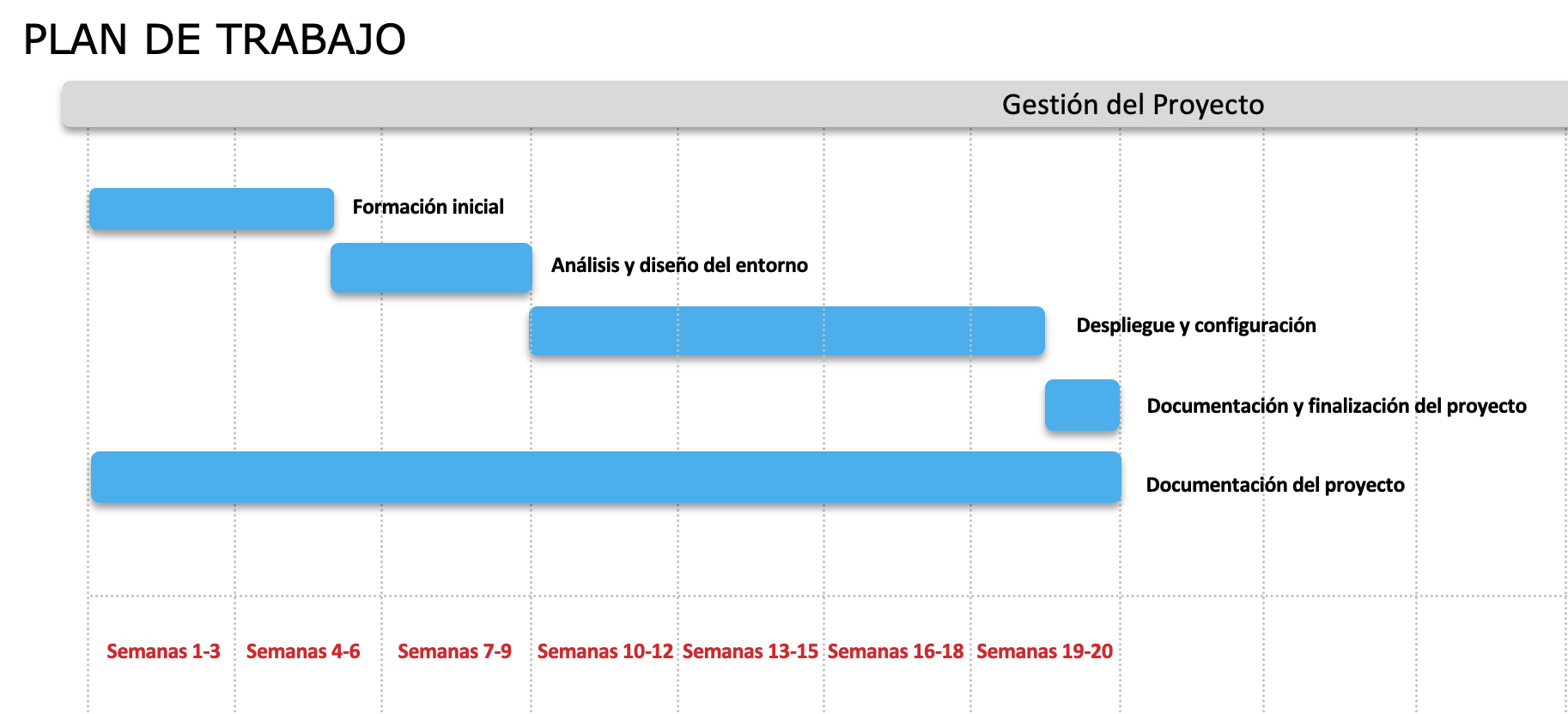
\* \*\*Desarrollo del código fuente (scripts y playbooks) (6 semanas):\*\* Durante esta etapa, se identificarán las dependencias necesarias, se elegirán las versiones adecuadas y se escogerán los roles necesarios. Tras esto, se procederá a materializar todo el conocimiento en el código.

\* \*\*Pruebas (3 semanas):\*\* En esta fase se busca detectar los fallos que se han podido cometer en etapas anteriores y corregirlos.

\* \*\*Correcciones estéticas (0,5 semanas):\*\* Retoques y modificaciones a nivel decorativo y artístico.

\* \*\*Documentación del código (0,5 semanas):\*\* Comentar y documentar de forma adecuada el código de los diferentes programas.

4. \*\*Documentación y finalización del proyecto (1 semanas):\*\* La documentación del proyecto se irá realizando a lo largo de toda la planificación y en la última semana se pondrá especial atención en los últimos detalles.



# MEDIOS

Se utilizarán máquinas RHEL (Red Hat Enterprise Linux) de laboratorio:

| **IP** | **Hostname** | **Rol** |
| --- | --- | --- |
| IP | master1.cin | Kubernetes Master y ETCD |
| IP | worker1.cin | Kubernetes Worker1 |
| IP | worker2.cin | Kubernetes Worker2 |
| IP | registry.cin | Kubernetes Registry |
| IP | loadbalancer1.cin | Kubernetes Load Balancer1 |
| IP | loadbalancer2.cin | Kubernetes Load Balancer2 |
| IP | VIPa | Kubernetes Ingress VIP (solo IP virtual, no es una máquina) |

# 

# REFERENCIAS BIBLIOGRÁFICAS

1. Valladares, C. S. T., & Sacoto, A. S. Q. (2022). Procesos de protección en entornos de ejecución de contenedores Kubernetes para una entidad financiera: una revisión sistemática. Dominio de las Ciencias, 8(4), 619-644.
2. 7 Years of Running Kubernetes for Mercedes-Benz,<https://www.youtube.com/watch?v=UmbjwSK9b3I&list=PLj6h78yzYM2PfD9vkHopnzNNIVicOFtih&index=18>
3. Kubernetes,<https://kubernetes.io/>