



UNIVERSIDAD NACIONAL DE EDUCACIÓN A DISTANCIA  
ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA INFORMÁTICA

Proyecto de fin de Grado en Ingeniería Informática

# **Virtualización ligera y cloud computing para el despliegue de servidores streaming para la docencia online de ingeniería.**

Ángel Roberto García Serpa

Dirigido por: Agustín Carlos Caminero Herráez

Curso 2025/2026, convocatoria diciembre





# Virtualización ligera y cloud computing para el despliegue de servidores streaming para la docencia online de ingeniería.

Proyecto de fin de Grado en Ingeniería Informática  
de modalidad genérica

Realizado por: Ángel Roberto García Serpa  
Dirigido por: Agustín Carlos Caminero Herráez

Fecha de lectura y defensa: Fecha de lectura

# Agradecimientos

*Incluir todos los agradecimientos*



# Resumen

Este proyecto se centra en configurar varios servidores que sean capaces de manejar streaming de video sobre contenedores Docker. Esta configuración será tanto en local como en AWS . Otro aspecto clave es el uso de kubernetes para ayudar con el despliegue y escalado de las aplicaciones consiguiendo un manejo eficiente de los recursos en cada momento. Para esto último es necesaria la monitorización de los contenedores. Se han elegido dos tecnologías muy utilizadas en este momento en el streaming de video y las videollamadas, como son HLS y WebRTC.



# Abstract

This project focuses on configuring several video streaming servers using Docker containers. This configuration will be both in local and on AWS. Another key aspect is the use of kubernetes that is in charge of automating the deploying, scaling and management of the containerized applications. For achieving this last part it is necessary monitoring the applications. Two technologies, such as HLS and WebRTC, were choosed for this Project because they are widely used at the moment in video streaming.





# Palabras clave

servidores de video, HLS, WebRTC, Docker, Kubernetes



# Keywords

video servers, HLS, WebRTC, Docker, Kubernetes



# Índice

Índice de tablas . . . . .	XIII
Índice de figuras . . . . .	XV
<b>1. Introducción</b>	<b>1</b>
<b>2. Estado del arte</b>	<b>3</b>
<b>3. Propuesta</b>	<b>5</b>
<b>4. Diseño</b>	<b>7</b>
<b>5. Desarrollo del proyecto</b>	<b>9</b>
5.1. opción A . . . . .	9
5.2. opción B . . . . .	9
<b>6. Pruebas</b>	<b>23</b>
<b>7. Conclusiones</b>	<b>27</b>
<b>Bibliografía</b>	<b>29</b>
<b>A.</b>	<b>31</b>
A.1. Fe de erratas . . . . .	31
<b>B.</b>	<b>33</b>
B.1. VirtualBox . . . . .	33
B.1.1. instalación . . . . .	33
B.1.2. Guest Additions . . . . .	33
B.1.3. configuración de la red de los equipos: . . . . .	33
B.2. Instalación de Ansible and sshpass . . . . .	33
B.3. Instalación de de kubeadm . . . . .	38
B.3.1. instalación kubectl . . . . .	38
B.3.2. Instalación de docker Engine . . . . .	40
B.4. Creación del cluster . . . . .	42
B.5. Instalar programas de obs . . . . .	45
B.5.1. install obs studio . . . . .	45



# Índice de tablas





# Índice de figuras

5.1. Generar gpg key . . . . .	10
5.2. Rancher . . . . .	11
5.3. versiones de Docker y kubernetes . . . . .	12
5.4. Docker . . . . .	12
5.5. Instalacion Docker . . . . .	13
5.6. Instalacion Docker . . . . .	13
5.7. Instalacion Docker . . . . .	14
5.8. Instalacion Docker . . . . .	14
5.9. Instalacion Docker . . . . .	15
5.10. Instalacion Kubectl . . . . .	15
5.11. Instalacion Kubectl . . . . .	16
5.12. Instalacion Kubectl . . . . .	16
5.13. Instalacion Docker Compose . . . . .	17
5.14. Configurar Virtualización . . . . .	17
5.15. Configurar Virtualización . . . . .	18
5.16. Docker Desktop en ejecución . . . . .	18
5.17. Construyendo el proyecto . . . . .	19
5.18. Construyendo el proyecto . . . . .	19
5.19. Proyecto construido . . . . .	20
5.20. Ejecutando el proyecto . . . . .	20
5.21. Ejecutando el proyecto . . . . .	21
B.1. Configuración Adaptador 1. . . . .	42
B.2. Configuración Adaptador 2. . . . .	42
B.3. Direcciones de los adaptadores en la Máquina Virtual. . . . .	43
B.4. kube-flannel.yml. . . . .	45
B.5. Ejecución del comando get nodes. . . . .	45



# Capítulo 1

## Introducción

La utilización de "nuevas tecnologías" para la educación es uno de los temas centrales de las universidades y centros educativos en todo el mundo. La aparición de ARPANET a finales de la década de los 60 inicia la carrera para la democratización del conocimiento poniendo a disposición de todo aquel que tenga un dispositivo capaz de conectarse a internet información sobre casi cada tema que una persona pueda encontrar interesante. El siguiente paso ha sido el de impartir clases de forma remota. A partir de la pandemia de 2020 Muchas universidades empezaron a utilizar las tecnologías de la información como una herramienta más a la hora de impartir clases. **Revisar**

La UNED ha conseguido acercar la experiencia de la formación presencial a la modalidad remota mediante la utilización de herramientas que se han ido modernizando con el paso del tiempo. Desde el uso de la radio y vídeos, pasando por televisión y más recientemente las clases online, la UNED siempre ha intentado poner a disposición de sus alumnos material audiovisual que ayudara al alumnado a la obtención de los conocimientos necesarios. Actualmente la utilización de diversas plataformas online ayudan a conseguir dicho objetivo. Una de las maneras en que se puede mejorar la calidad de la educación a distancia es mediante las clases online y el streaming de vídeo online. En este momento esta es una de las herramientas utilizadas por nuestra universidad y por ello es interesante conocer herramientas que puedan ayudar a mejorar la calidad de las enseñanzas.

En este TFG he intentado revisar algunas tecnologías que ayuden a realizar el lema de nuestra UNED "Que la sabiduría se mueva más que las cosas que se mueven". Para ello he investigado e implementado el uso de servidores de streaming relevantes en este momento y he llevado a cabo pruebas de rendimiento sobre los mismos que puedan ayudar a realizar eficientemente streaming de vídeo y que esto se convierta en una herramienta más que ayude a la formación de los alumnos.

Voy a revisar tecnologías que se utilizan para el streaming de video como son servidores de emisión de video unidireccionales que se han utilizado tradicionalmente codificando señales de video enviadas y servidas en formatos que puedan consumirse desde un navegador web o un reproductor de video. También servidores web basados en webRTC que mejoran la latencia de los anteriores y cuya comunicación bidireccional hace que se pueda dar conversación entre los participantes.**Redundante, mirar si es necesario mencionar las tecnologías en la introducción o eliminar éste párrafo**



# Capítulo 2

## Estado del arte

### 1. Streaming de vídeo: fundamentos y tecnologías actuales

- Codificación y codecs modernos: formatos como AV1, VP9 o H.264, sus ventajas respecto a calidad y eficiencia energética.
- Adaptive Bitrate Streaming (ABR): protocolos como MPEG-DASH, HLS o Smooth Streaming y cómo funcionan los algoritmos del cliente para ajustar calidad según ancho de banda.
- Arquitecturas de extremo a extremo: revisión holística del pipeline de streaming, incluyendo captura, ingesta, transcodificación, entrega (CDN) y reproducción cliente.
- Edge & MEC (Mobile Edge Computing): almacenamiento y procesamiento en nodos cerca del usuario para mejorar latencia y eficiencia, uso de IA para caching y offloading.
- Modelos híbridos P2P-CDN: arquitectura híbrida que mejora la latencia, el consumo energético y la calidad para streaming en vivo
- Codificación adaptativa energéticamente eficiente: estrategias como LADRE con algoritmos ML para balancear calidad, latencia y consumo energético.

### 2. Contenerización y orquestación de servidores de contenidos

- Virtualización basada en contenedores: diferencias clave entre contenedores y máquinas virtuales (recursos, arranque, aislamiento, portabilidad).
- Uso práctico en broadcast y streaming: microservicios, contenedorización y orquestación en infraestructuras de radiodifusión modernas, con el fin de mejorar la flexibilidad y resiliencia operativa.
- Servidor de streaming de baja latencia en contenedor: OvenMediaEngine (OME), proyecto open-source con Docker, optimizado para streaming de baja latencia en arquitecturas contenedorizadas.
- Monitoreo y registro en pipelines contenedorizados: prácticas clave en DevOps para asegurar rendimiento, detección de errores y optimización de servicios despliegues en contenedores.

### 3. Retos actuales y líneas futuras: **Revisar porque ya se habla de varios de estos temas en el primer punto**

- latencia ultra-baja
- uso de IA
- green streaming
- Moby (open source Docker)
- Docker Compose



# Capítulo 3

## Propuesta

Se pretende seleccionar 2 ó 3 tecnologías para servidores de medios, crear la estructura distribuida mediante el uso de Docker y Kubernetes y comparar las diferencias en cuanto a latencia y escalabilidad de los distintos servidores. Una vez elegido uno de ellos por sus mejores características, se elegirá un proveedor de servicios de los más utilizados en la actualidad como son AWS, Azure o Google Cloud y se documentará el despliegue en las mismas.





# Capítulo 4

## Diseño

Estas son los criterios seguidos para la selección del programa a utilizar para el desarrollo de este TFG.

- Capacidad de Streaming: ¿Soporta Live Streaming (en vivo) y Video On Demand (VOD) (grabado)?
- Protocolos Soportados: RTMP (ingesta), HLS y DASH (distribución a clientes), WebRTC (baja latencia).
- Transcodificación/Transmuxing: ¿Puede convertir videos a diferentes calidades y formatos sobre la marcha?
- Escalabilidad: ¿Cómo se escala en un entorno distribuido (contenedores, Kubernetes)? ¿Soporta balanceo de carga?
- Documentación y Comunidad: Facilidad para encontrar ayuda y recursos.
- Licenciamiento/Costo: ¿Es de código abierto o comercial?
- Soporte de Contenedores: ¿Hay imágenes Docker oficiales o bien mantenidas? ¿Ejemplos de despliegue en Kubernetes?



# Capítulo 5

## Desarrollo del proyecto

### 5.1. opción A

Se seguira el libro Mastering kubernetes **Revisar ToDo** en su cuarta edicion. Para la creacion del cluster es necesario instalar Rancher desktop [Instalar Rancher](#). Ésta es una aplicacion de escritorio que permitira ejecutar Docker. Rancher Desktop instalara otras herramientas como:

- Helm
- Kubectl
- Nerdctl
- Moby (open source Docker)
- Docker Compose

Se instala pass, necesario para guardar credenciales que se pasan via docker login y nerdctl login.

Una vez instalado Rancher es recomendable generar una GPG key para securizar los secrets. para ello hacmoes lo siguiente:

Se instala docker desktop siguiendo las instrucciones de la página oficial [Instalar Docker](#) comprobamos que se han instalado correctamente tanto kubectl como docker el usuario debe tener permisos de docker y de video

```
sudo usermod -aG video $USER
sudo usermod -aG docker $USER
```

### 5.2. opción B

de momento sigue esta Se utiliza una distribución basada en Ubuntu como es Mint en su versión XFCE que es una versión ligera y se realiza la instalación en una MV para comprobar que la instalación funciona se deb instalar: Docker: Siguiendo las instrucciones de la Documentación de Docker:: [Instalar Docker](#)

añadimos el usuario al grupo de Docker y probamos qeu se puede ejecutar sin sudo:

kubectl [Instalar Kubectl](#).

Instalamos y comprobamos versión de docker compose

Para poder utilizar Docker Desktop en una MV se debe configurar en VirtualBox:

Tras configurar la virtualización y reiniciar, se puede ver que se ejecuta correctamente Docker Desktop

Una vez instalado todo podemos empezar a ejecutar el servidor:

Se utiliza el comando :



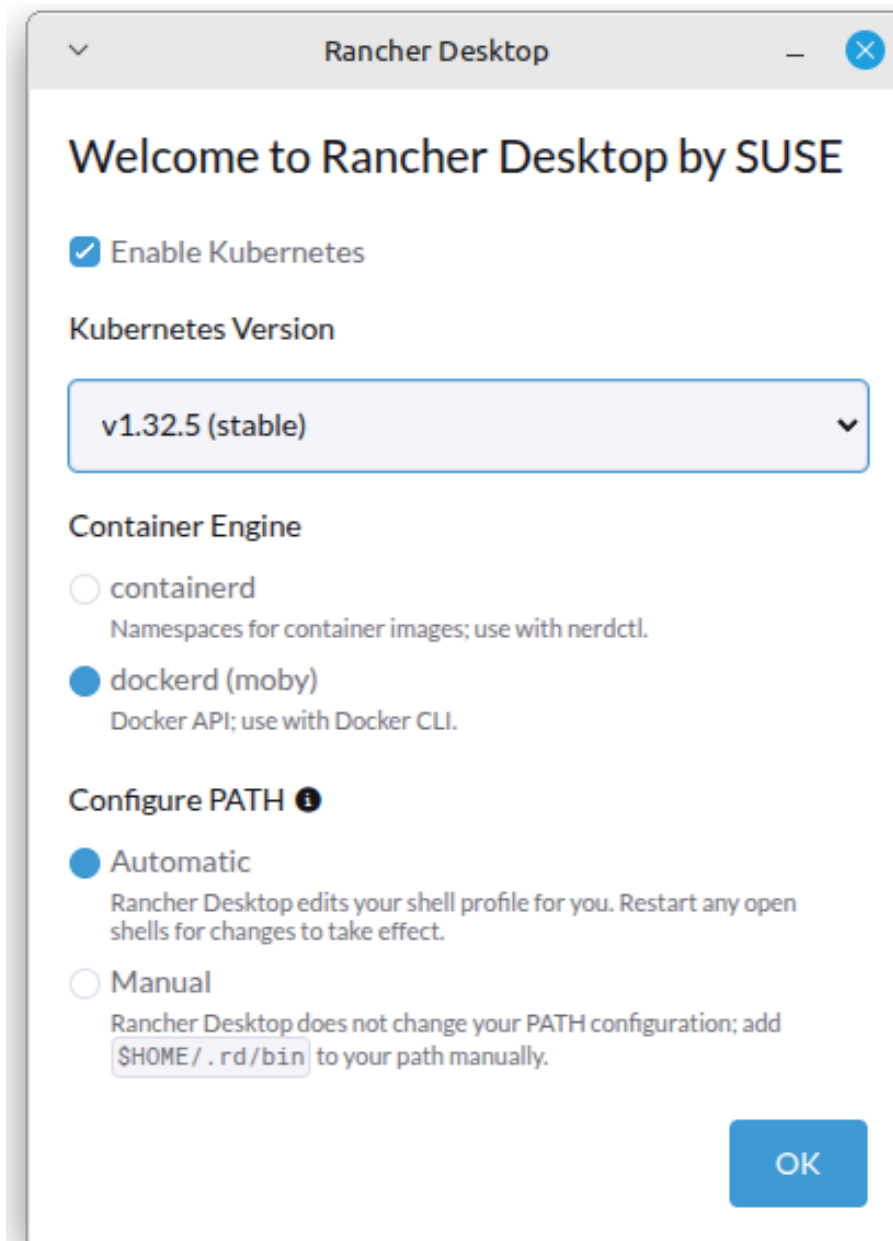
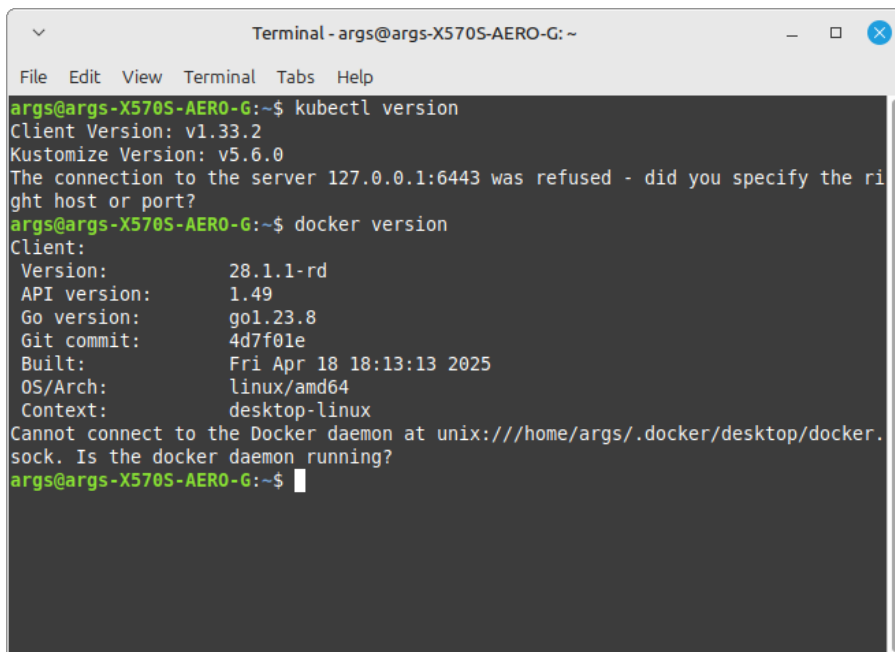


Figura 5.2: Rancher

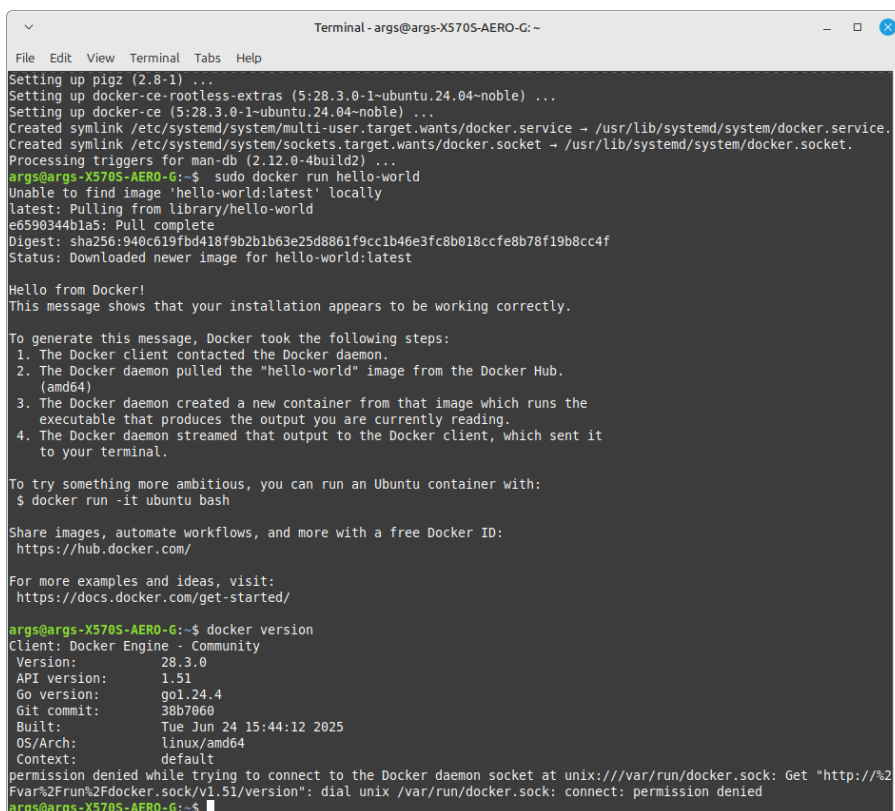


```

Terminal - args@args-X570S-AERO-G: ~
File Edit View Terminal Tabs Help
args@args-X570S-AERO-G:~$ kubectl version
Client Version: v1.33.2
Kustomize Version: v5.6.0
The connection to the server 127.0.0.1:6443 was refused - did you specify the right host or port?
args@args-X570S-AERO-G:~$ docker version
Client:
Version:      28.1.1-rd
API version:  1.49
Go version:   go1.23.8
Git commit:   4d7f01e
Built:        Fri Apr 18 18:13:13 2025
OS/Arch:      linux/amd64
Context:      desktop-linux
Cannot connect to the Docker daemon at unix:///home/args/.docker/desktop/docker.sock. Is the docker daemon running?
args@args-X570S-AERO-G:~$

```

Figura 5.3: versiones de Docker y kubernetes



```

Terminal - args@args-X570S-AERO-G: ~
File Edit View Terminal Tabs Help
Setting up pigz (2.8-1) ...
Setting up docker-ce-rootless-extras (5:28.3.0-1-ubuntu.24.04~noble) ...
Setting up docker-ce (5:28.3.0-1-ubuntu.24.04~noble) ...
Created symlink /etc/systemd/system/multi-user.target.wants/docker.service → /usr/lib/systemd/system/docker.service.
Created symlink /etc/systemd/system/sockets.target.wants/docker.socket → /usr/lib/systemd/system/docker.socket.
Processing triggers for man-db (2.12.0-4build2) ...
args@args-X570S-AERO-G:~$ sudo docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
e6590344b1a5: Pull complete
Digest: sha256:940c619fbd418f9b2b1b63e25d8861f9cc1b46e3fc8b018ccfe8b78f19b8cc4f
Status: Downloaded newer image for hello-world:latest

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
1. The Docker client contacted the Docker daemon.
2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
   (amd64)
3. The Docker daemon created a new container from that image which runs the
   executable that produces the output you are currently reading.
4. The Docker daemon streamed that output to the Docker client, which sent it
   to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
$ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
https://hub.docker.com/

For more examples and ideas, visit:
https://docs.docker.com/get-started/

args@args-X570S-AERO-G:~$ docker version
Client: Docker Engine - Community
Version:      28.3.0
API version:  1.51
Go version:   go1.24.4
Git commit:   38b7060
Built:        Tue Jun 24 15:44:12 2025
OS/Arch:      linux/amd64
Context:      default
permission denied while trying to connect to the Docker daemon socket at unix:///var/run/docker.sock: Get "http://%2Fvar%2Frun%2Fdocker.sock/v1.51/version": dial unix /var/run/docker.sock: connect: permission denied
args@args-X570S-AERO-G:~$

```

Figura 5.4: Docker

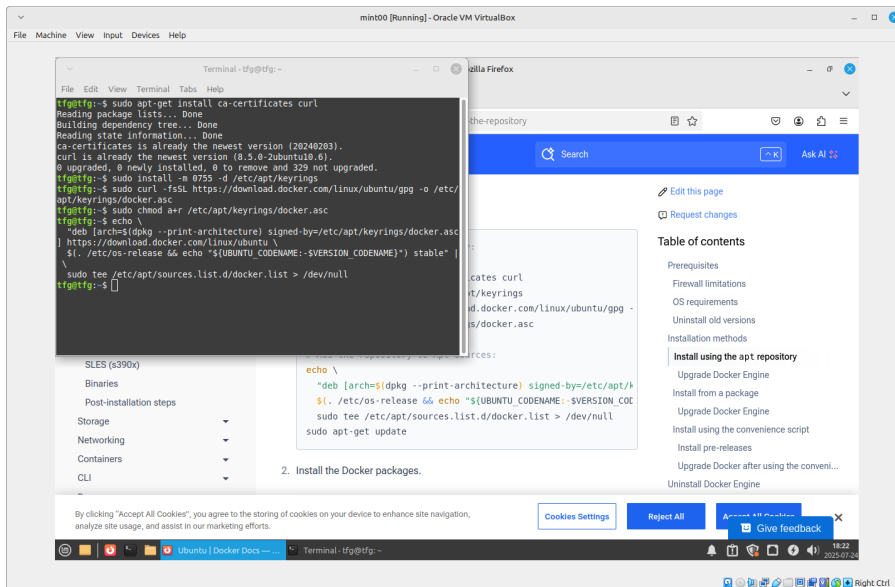


Figura 5.5: Instalacion Docker

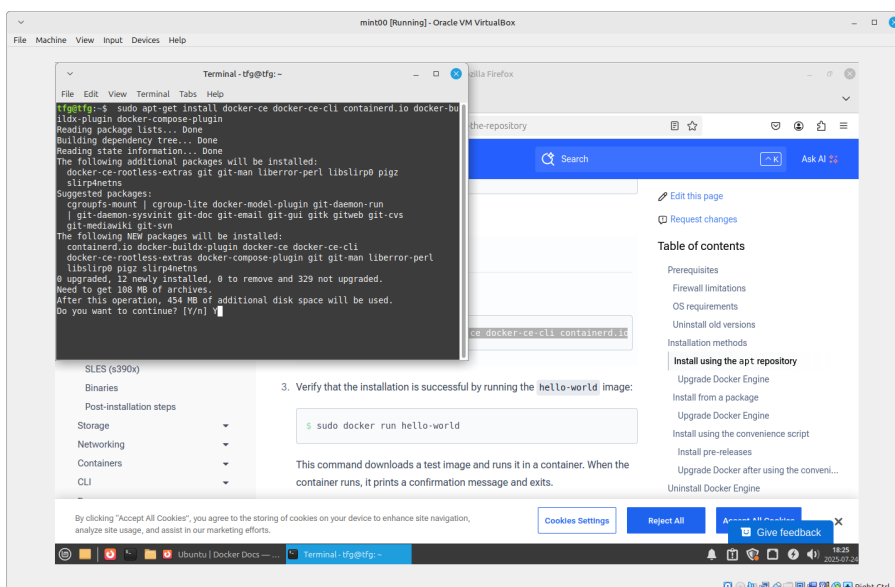


Figura 5.6: Instalacion Docker





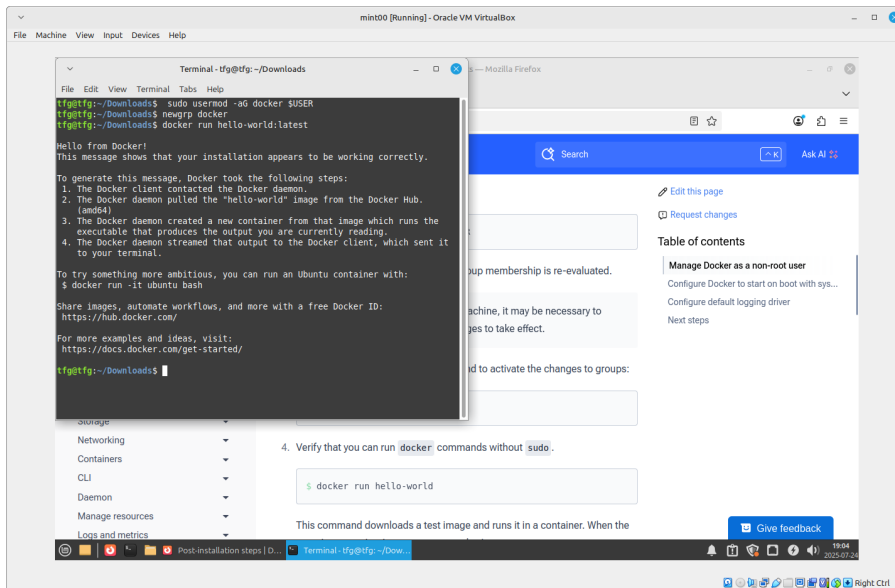


Figura 5.9: Instalacion Docker

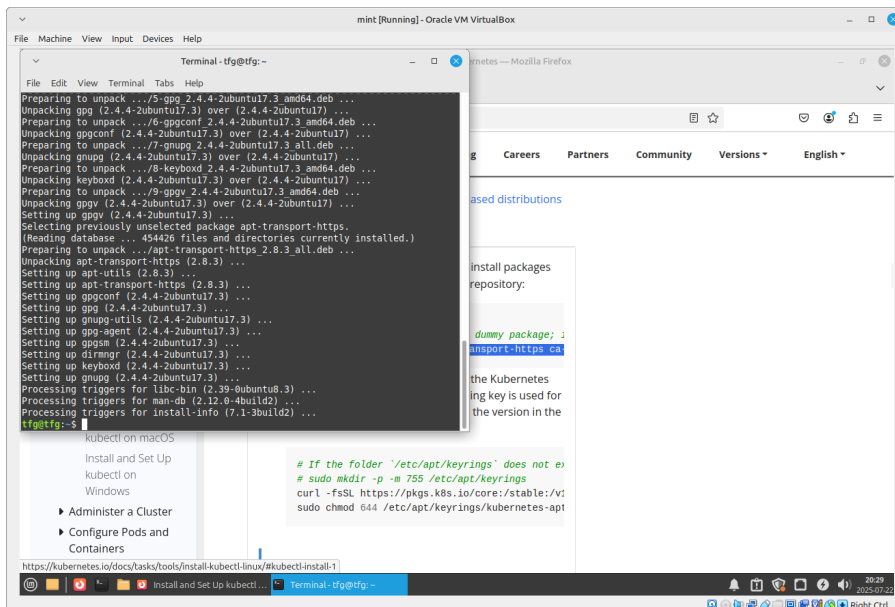


Figura 5.10: Instalacion Kubectl

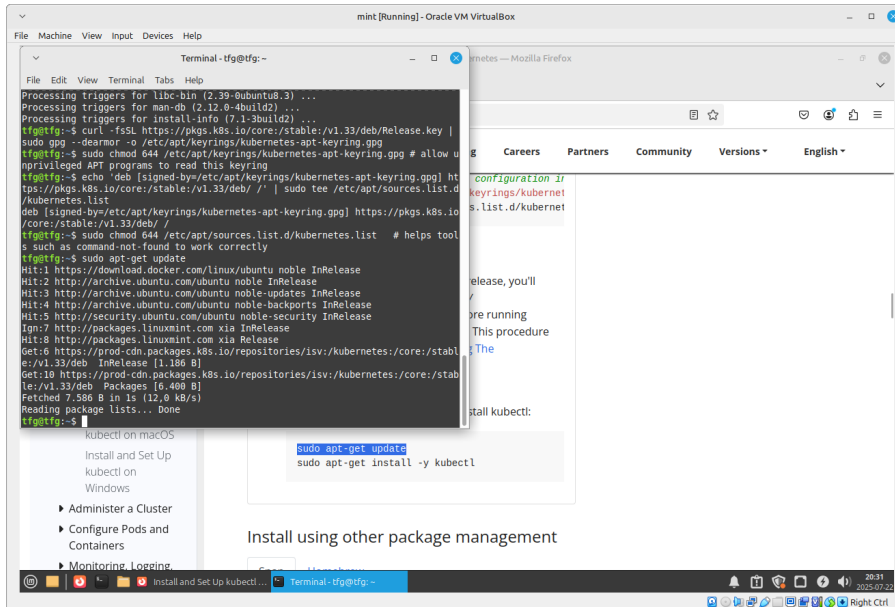


Figura 5.11: Instalacion Kubectl

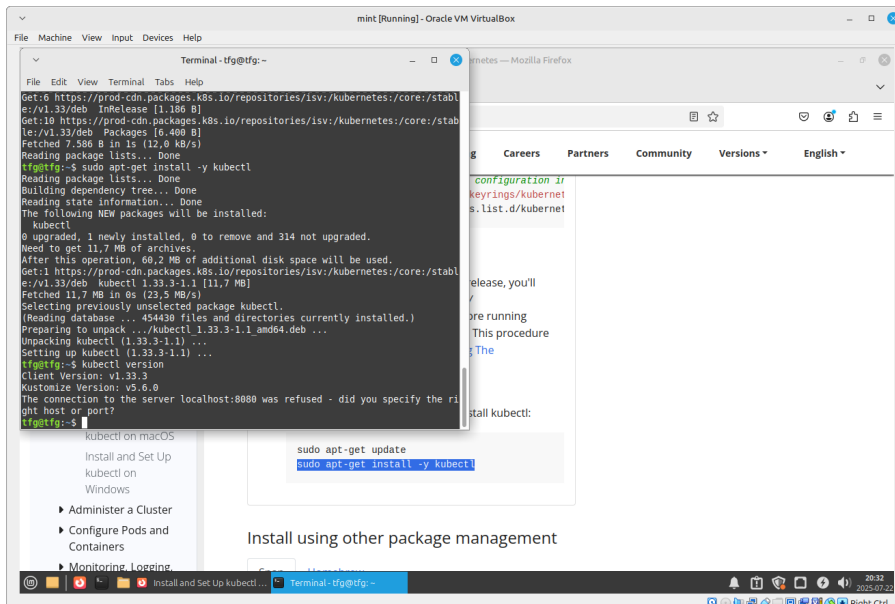


Figura 5.12: Instalacion Kubectl

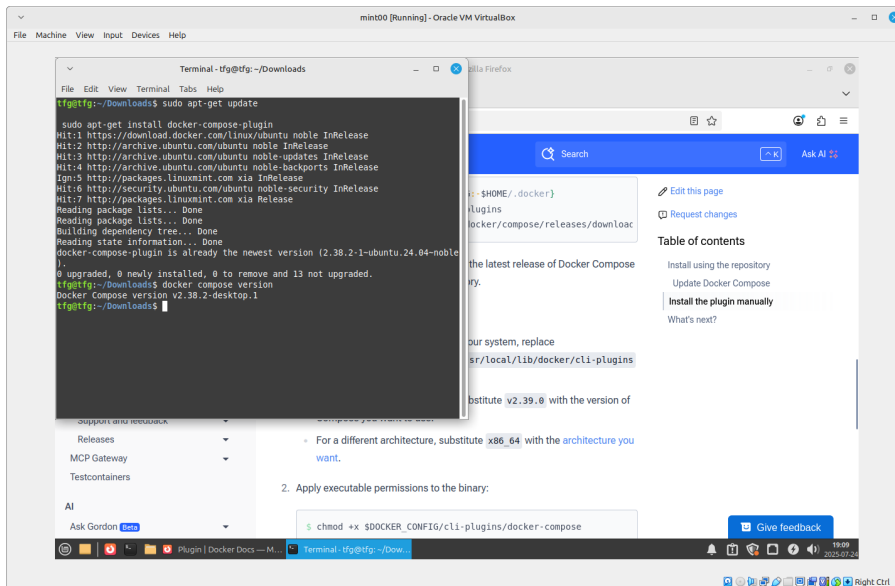


Figura 5.13: Instalacion Docker Compose

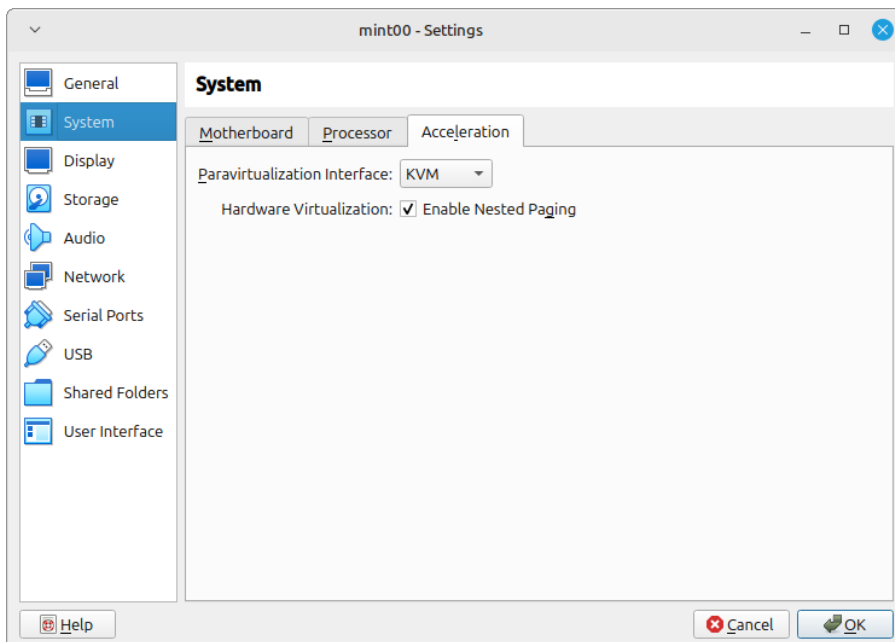


Figura 5.14: Configurar Virtualización

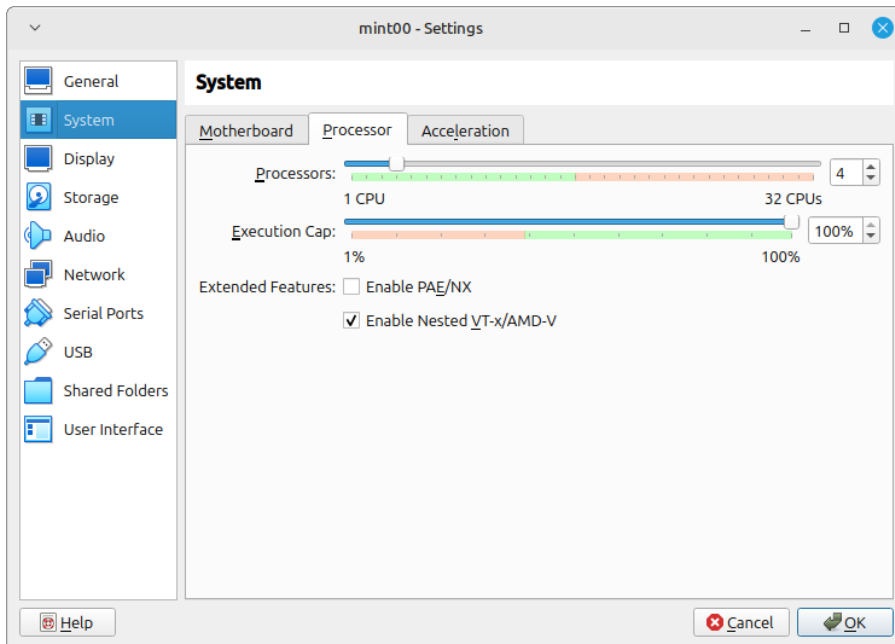


Figura 5.15: Configurar Virtualización

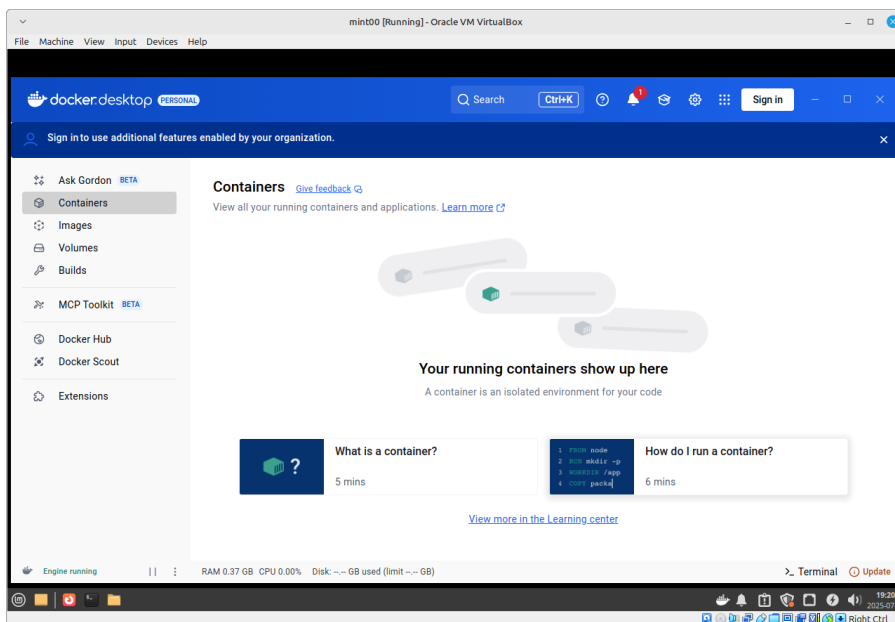


Figura 5.16: Docker Desktop en ejecución

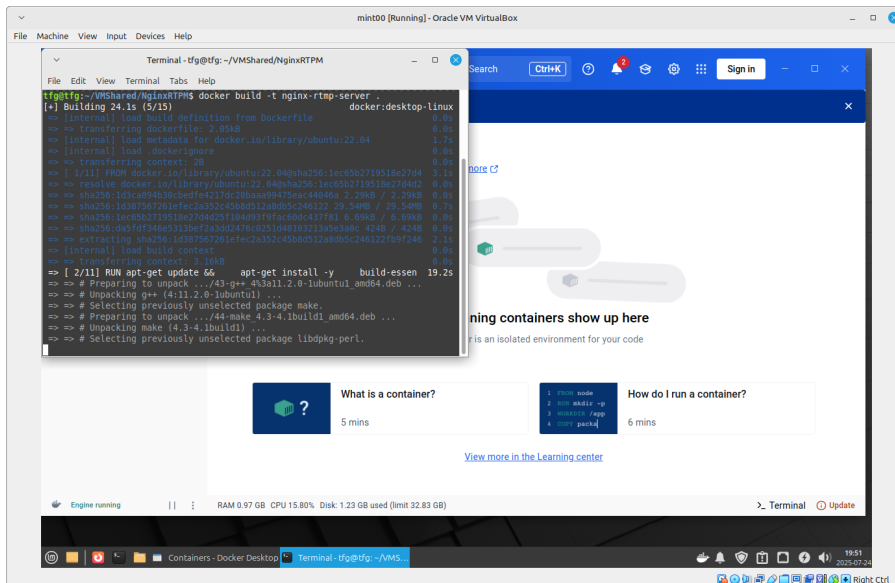


Figura 5.17: Construyendo el proyecto

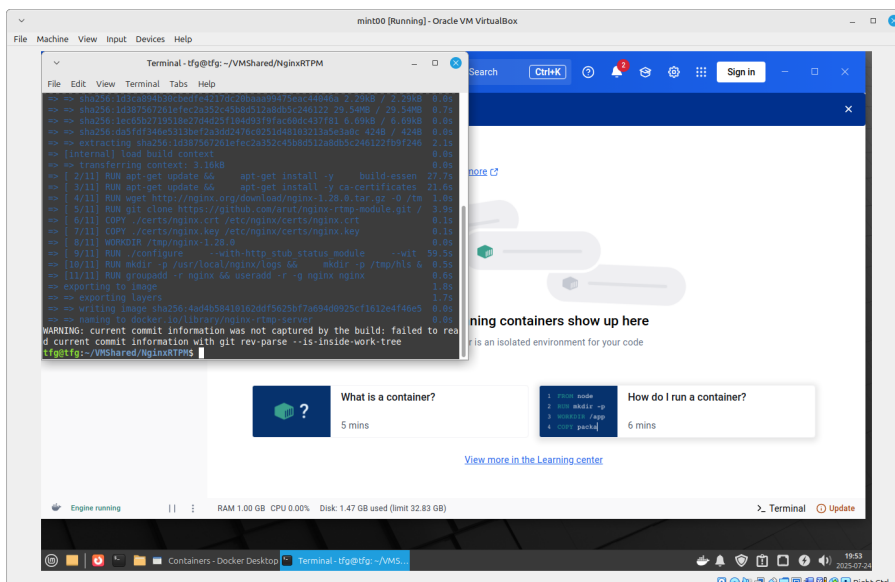


Figura 5.18: Construyendo el proyecto

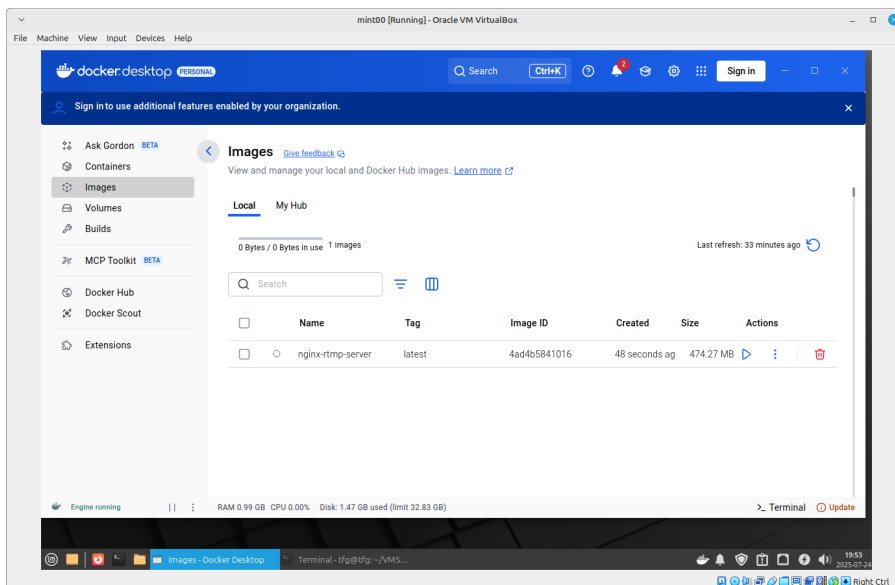


Figura 5.19: Proyecto construido

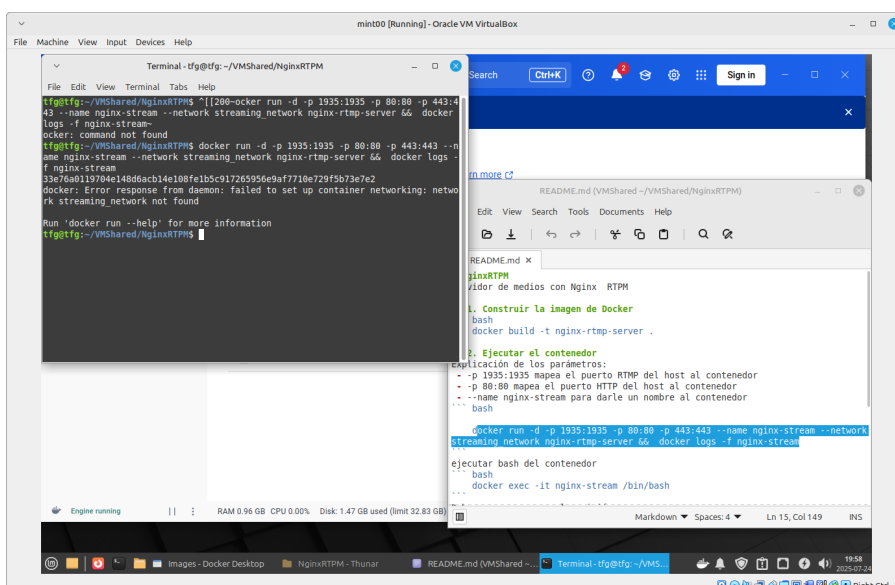


Figura 5.20: Ejecutando el proyecto

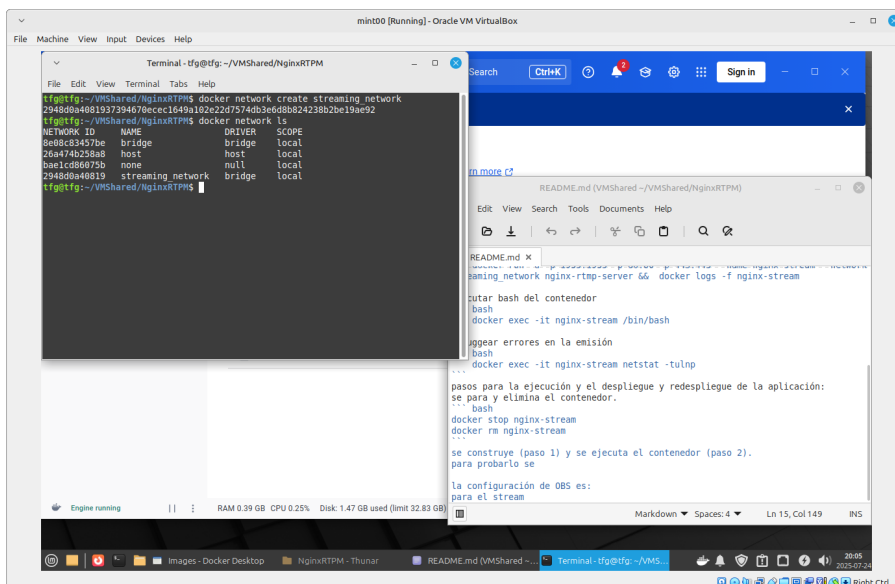


Figura 5.21: Ejecutando el proyecto





# Capítulo 6

## Pruebas

mejorar la latencia:

hay mucho retraso entre lo grabado y la emisión eso es la latencia? se puede mejorar?

¡Totalmente! Ese retraso entre lo grabado y la emisión se llama efectivamente latencia. Es una métrica crucial en el streaming en vivo y, sí, ¡se puede mejorar!

¿Qué es la Latencia en el Streaming?

La latencia en el streaming es el tiempo que transcurre desde que un evento ocurre en la realidad (por ejemplo, alguien habla frente a una cámara) hasta que ese mismo evento es visto o escuchado por el espectador. Se mide generalmente en segundos o milisegundos.

Existen diferentes niveles de latencia:

Latencia alta/estándar: Típicamente entre 15-30 segundos. Es común en YouTube Live o Twitch si no se configuran modos de baja latencia. Ofrece mayor estabilidad y calidad, ya que permite más tiempo para el buffering.

Latencia baja: Generalmente entre 5-7 segundos. Es un equilibrio entre calidad y cercanía al tiempo real.

Latencia ultrabaja (near real-time): Menos de 1 segundo (incluso milisegundos). Es crucial para interacción en tiempo real como videoconferencias, juegos en línea o transmisiones donde la interacción instantánea es clave.

Factores que Influyen en la Latencia

Varios elementos en la cadena de streaming contribuyen a la latencia:

Codificación (OBS Studio):

Calidad de video/audio: Mayor resolución, bitrate o complejidad de códecs (como H.265 vs H.264) requieren más procesamiento y pueden añadir latencia.

Configuración del codificador: Los "presets" o perfiles de codificación (ej. veryfast, medium, slow en x264) afectan directamente la latencia. Los más rápidos usan menos CPU pero generan archivos más grandes o de menor calidad, y viceversa.

Intervalo de Keyframes (GOP size): Cuanto más grande es este intervalo (tiempo entre fotogramas completos), más tiempo se necesita para que el reproductor empiece a decodificar el stream. 2-4 segundos es lo habitual, pero intervalos más pequeños (1-2s) reducen la latencia.

Búferes: OBS y el codificador usan búferes para garantizar una transmisión fluida. Búferes más grandes aumentan la latencia.

Protocolos de Transmisión:

RTMP (Real-Time Messaging Protocol): Excelente para la ingesta (de OBS al servidor) por su baja latencia inherente. Sin embargo, no es compatible directamente con navegadores modernos para la distribución.

HLS (HTTP Live Streaming) / DASH (Dynamic Adaptive Streaming over HTTP): Son los protocolos dominantes para la distribución a la mayoría de los dispositivos. Funcionan segmentando el video en pequeños trozos (.ts o .mp4) y listas de reproducción (.m3u8 o .mpd). Cada segmento es una unidad de latencia. Por defecto, pueden tener latencia alta (15-30s) debido al tamaño y número de segmentos en el búfer.

LL-HLS (Low-Latency HLS) y CMAF (Common Media Application Format): Son extensiones de HLS/DASH diseñadas específicamente para reducir la latencia (a menudo por debajo de los 2-3 segundos) mediante el uso de segmentos más pequeños y la entrega parcial de estos.

WebRTC (Web Real-Time Communication): Ofrece la latencia más baja (sub-segundo) al permitir la comunicación directa peer-to-peer entre navegador y servidor. Es el estándar para videoconferencias. Red5 Pro tiene un fuerte enfoque en WebRTC.

SRT (Secure Reliable Transport): Un protocolo más nuevo y robusto, ideal para el transporte de video de alta calidad y baja latencia en redes inestables (común en transmisiones profesionales).

Red (Internet y CDN):

Ancho de banda: Una conexión de internet insuficiente (tanto para el que emite como para el que recibe) provocará buffering y aumentará la latencia.

Distancia geográfica: Cuanto más lejos estén el servidor de streaming y los espectadores, mayor será la latencia debido al tiempo que tardan los datos en viajar.

Congestión de la red: El tráfico en la red puede ralentizar la entrega de paquetes.

CDNs (Content Delivery Networks): Ayudan a reducir la latencia al almacenar y entregar contenido desde servidores más cercanos al espectador.

Servidor de Streaming (Nginx, Red5 Pro):

Configuración del búfer del servidor: Los servidores de streaming también tienen sus propios búferes para gestionar el flujo de datos.

Transcodificación/Transmuxing: Si el servidor necesita convertir el stream a diferentes formatos o calidades sobre la marcha, esto añade tiempo y, por lo tanto, latencia.

Reproductor (Video.js, VLC):

Búfer del reproductor: Los reproductores tienen sus propios búferes para suavizar la reproducción y evitar interrupciones. Búferes más grandes significan más latencia.

¿Cómo Mejorar la Latencia con Nginx y Red5 Pro?

Para Nginx con RTMP/HLS/DASH

Nginx es muy configurable para reducir la latencia, especialmente en la parte de HLS/DASH.

Reducir el `hls_fragment` y `hls_playlist_length`:

`hls_fragment`: Duración de cada segmento .ts. Un valor más bajo significa segmentos más pequeños y, por lo tanto, menor latencia. Un buen punto de partida para baja latencia es 1 segundo (1s) o incluso 0.5 segundos (0.5s).

`hls_playlist_length`: Duración total de los segmentos listados en el .m3u8. Un valor más bajo significa que el reproductor tiene menos segmentos para buffer. Puede ser de 3s a 5s.

Modifica tu `nginx.conf` así:

```
1  Nginx
2
3  rtmp {
4      server {
5          listen 1935;
6          chunk_size 4096;
7
8          application live {
9              live on;
10             record off;
11         }
```

```

12     hls on;
13     hls_path /tmp/hls/live;
14     hls_fragment 1s;           # <-- Reducir a 1 segundo
15     hls_playlist_length 3s;    # <-- Reducir a 3 segundos
16     # hls_sync 1ms;           # <-- Opcional:
17                               sincronizacion mas precisa
18
19     dash on;
20     dash_path /tmp/hls/live;   # 0 /tmp/dash/live si quieres
21                               una ruta separada para DASH
22     # dash_fragment 1s;        # Puedes anhadir esto para
23                               DASH tambien si no esta ya
24     # dash_playlist_length 3s; # Similar a HLS para DASH
25
26     add_header 'Access-Control-Allow-Origin' '*';
27 }
28
29 }
30
31 }

```

¡Importante! Al reducir la duración de los fragmentos, aumenta la cantidad de peticiones HTTP al servidor y la complejidad de la gestión de la playlist. Esto puede aumentar la carga del servidor y ser más sensible a problemas de red, potencialmente causando interrupciones si la conexión no es estable. Es un equilibrio.

Configuración de OBS Studio (Fuente del Stream):

Tasa de bits (Bitrate): Ajusta el bitrate de video y audio a un valor óptimo para tu conexión a internet. Un bitrate muy alto puede saturar tu subida.

Codificador: Usa H.264 para video y AAC para audio.

Intervalo de Keyframes (Keyframe Interval): Es CRÍTICO para la latencia en HLS/DASH. En OBS, ve a Configuración > Salida > Retransmisión. Pon el Intervalo de Keyframes.<sup>a</sup> 2 segundos. Si lo bajas a 1 segundo, la latencia será menor, pero la carga de CPU de tu máquina y de Nginx puede aumentar. Asegúrate de que este valor coincida con o sea un múltiplo del hls\_fragment.

Búfer del Reproductor (Video.js): Video.js y los navegadores tienen sus propios mecanismos de buffering. Puedes influir en esto, aunque generalmente sus valores por defecto son buenos.

Para baja latencia, a veces puedes jugar con liveSyncDuration o liveSyncDurationCount en Video.js, pero es más avanzado y podría causar inestabilidad si no se hace bien. La configuración del servidor suele tener un mayor impacto.

Para Red5 Pro (Enfoque WebRTC)

Red5 Pro brilla en baja latencia, especialmente con WebRTC. Para latencia ultrabaja, la clave es usar WebRTC.

Configuración de WebRTC en Red5 Pro:

WebRTC es nativamente de baja latencia. Red5 Pro está diseñado para esto. La configuración principal se realiza en los archivos de propiedades de la aplicación Red5 Pro (ej. red5-web.properties o similar en la carpeta de la aplicación live o webrtc).

Deberás configurar los puertos UDP necesarios para WebRTC (los STUN/TURN servers) y asegurar que estén abiertos en tu firewall y en el contenedor (si los usas).

La arquitectura para una producción de baja latencia con WebRTC es más compleja, a menudo involucra clústeres de Red5 Pro y servidores STUN/TURN para manejar NAT y fuegos de artificio.

Aplicación de Ejemplo de Red5 (WebRTC):

Cuando pruebes Red5 (incluso la versión open source), busca las aplicaciones de ejemplo de WebRTC que suelen venir preinstaladas (accede vía <http://localhost:5080/> y navega por las webapps). Estas te mostrarán cómo publicar y ver streams WebRTC desde el navegador directamente.

OBS Studio para WebRTC (más complejo):

OBS Studio no tiene soporte nativo para emitir directamente a WebRTC. Necesitarías un plugin de OBS para WebRTC o usar una herramienta como FFmpeg para transcodificar RTMP a un stream compatible con WebRTC y enviarlo a Red5 Pro. Esto añade complejidad. Lo más común es usar un cliente web basado en JavaScript para publicar directamente a WebRTC.

Consideraciones Adicionales para Reducir la Latencia

Hardware y Red: Asegúrate de que tu máquina (donde corre Docker/OBS) y tu conexión a internet sean potentes y estables. La latencia de tu red doméstica y la de tu ISP son factores fundamentales.

Ubicación del Servidor: Si en el futuro despliegas en la nube, elige una región cercana a tus espectadores para minimizar la latencia de red.

Evitar Transcodificación Excesiva: Cada paso de transcodificación (cambiar códecs, resolución) añade latencia. Si puedes, envía el stream en un formato lo más cercano posible al deseado para el consumo.

Búferes: Entiende que los búferes son necesarios para la estabilidad. Reducirlos demasiado puede llevar a interrupciones (buffering) si la red o el servidor tienen fluctuaciones. Siempre es un equilibrio entre latencia y estabilidad.

Sí, el retraso.<sup>es</sup> latencia. Y sí, con las configuraciones adecuadas en Nginx (reduciendo `hls_fragment` y `hls_playlist_length`) y optimizando OBS (intervalo de keyframes), junto con la exploración de WebRTC en Red5 Pro, puedes reducirla significativamente.

¿Tienes algún objetivo de latencia específico en mente (ej. menos de 5 segundos, menos de 1 segundo)? Eso nos ayudaría a afinar más las recomendaciones.

# Capítulo 7

## Conclusiones

Contenido del capítulo...



# Bibliografía

- [1] G. Sayfan, *Compiladores: Principios, técnicas y herramientas*. Pearson Addison-Wesley, 2008.
- [2] J. M. Díaz Martínez, *Fundamentos básicos de los sistemas operativos*. Sanz y Torres S.L., 2011.
- [3] A. project Contributors. (2023, 03) Installing ansible.
- [4] T. K. Authors. (2017) Installing kubernetes.
- [5] D. Inc. (2017) Install docker engine.





# Apéndice A

## A.1. Fe de erratas

a la hora de hacer los listados con Latex los guiones los toma como otro carácter similar pero diferente, lo que hace inútil el querer copiar y pegar directamente desde este trabajo a una terminal. Además añade aleatoriamente espacios que puede cambiar el uso de los comandos. Por ello se recomienda pasar los comandos a un editor de texto y borrar los espacios extra y reemplazar los guiones copiados por los reales.

comandos útiles:

ver quién está usando el puerto xx `sudo netstat -tulnp | grep :80` parar el servicio que está usando el puerto `sudo systemctl stop nginx.service`

puntos clave de la monitorización. Correcta configuración de los volúmenes, redes y organización de los ficheros



# Apéndice B

## Instalaciones y Configuraciones

### B.1. VirtualBox

#### B.1.1. instalación

#### B.1.2. Guest Additions

<https://www.makeuseof.com/tag/virtualbox-guest-additions-what-they-are-and-how-to-install-them/>

#### B.1.3. configuración de la red de los equipos:

se crea una conexión de tipo *bridge* y se le da una IP fija a cada equipo, de esta manera las VM tienen conexión a Internet y una red dedicada.

- mint00 192.168.1.100
- mint01 192.168.1.101
- mint02 192.168.1.102
- mint03 192.168.1.103

### B.2. Instalación de Ansible and sshpass

Installing Ansible and sshpass. checking ansible version

```
tfg@mint01:~$ sudo apt update && sudo apt install software-properties-common
&& sudo add-apt-repository --yes ppa:ansible/ansible && sudo apt install
ansible
Hit:1 http://mirror.tedra.es/ubuntu jammy InRelease
Hit:2 http://mirror.tedra.es/ubuntu jammy-updates InRelease
Hit:3 http://mirror.tedra.es/ubuntu jammy-backports InRelease
Get:4 http://security.ubuntu.com/ubuntu jammy-security InRelease [110 kB]
Ign:5 https://mirrors.ptisp.pt/linuxmint vera InRelease
Hit:6 https://mirrors.ptisp.pt/linuxmint vera Release
Hit:8 https://download.sublimetext.com apt/stable/ InRelease
Get:9 http://security.ubuntu.com/ubuntu jammy-security/main amd64 DEP-11
Metadata [41,4 kB]
Get:10 http://security.ubuntu.com/ubuntu jammy-security/universe amd64 DEP-11
Metadata [18,5 kB]
Fetched 170 kB in 1s (214 kB/s)
Reading package lists... Done
```

```

Building dependency tree... Done
Reading state information... Done
7 packages can be upgraded. Run 'apt list --upgradable' to see them.
W: https://download.sublimetext.com/apt/stable/InRelease: Key is stored in
  legacy trusted.gpg keyring (/etc/apt/trusted.gpg), see the DEPRECATION
  section in apt-key(8) for details.
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following NEW packages will be installed:
software-properties-common
0 upgraded, 1 newly installed, 0 to remove and 7 not upgraded.
Need to get 9.712 B of archives.
After this operation, 16,4 kB of additional disk space will be used.
Get:1 https://mirrors.ptisp.pt/linuxmint vera/upstream amd64 software-
  properties-common all 2.2.1 [9.712 B]
Fetched 9.712 B in 0s (26,5 kB/s)
Selecting previously unselected package software-properties-common.
(Reading database ... 610456 files and directories currently installed.)
Preparing to unpack .../software-properties-common_2.2.1_all.deb ...
Unpacking software-properties-common (2.2.1) ...
Setting up software-properties-common (2.2.1) ...
You are about to add the following PPA:
Ansible is a radically simple IT automation platform that makes your
  applications and systems easier to deploy. Avoid writing scripts or custom
  code to deploy and update your applications automate in a language that
  approaches plain English, using SSH, with no agents to install on remote
  systems.

http://ansible.com/

If you face any issues while installing Ansible PPA, file an issue here:
https://github.com/ansible-community/ppa/issues
More info: https://launchpad.net/~ansible/+archive/ubuntu/ansible
gpg: directory '/root/.gnupg' created
gpg: keybox '/root/.gnupg/pubring.kbx' created
gpg: /root/.gnupg/trustdb.gpg: trustdb created
gpg: keybox '/etc/apt/keyrings/6125E2A8C77F2818FB7BD15B93C4A3FD7BB9C367.
  keyring' created
gpg: key 93C4A3FD7BB9C367: public key "Launchpad PPA for Ansible, Inc."
  imported
gpg: Total number processed: 1
gpg:      imported: 1
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
python-babel-localedata python3-argcomplete python3-babel python3-distutils
python3-dnspython python3-jinja2 python3-jmespath python3-kerberos
python3-lib2to3 python3-libcloud python3-lockfile python3-ntlm-auth
python3-pycryptodome python3-requests-kerberos python3-requests-ntlm
python3-requests-toolbelt python3-selinux python3-simplejson python3-tz
python3-winrm python3-xlrd
Suggested packages:
cowsay sshpass python3-sniffio python3-trio python-jinja2-doc
python-lockfile-doc
The following NEW packages will be installed:
ansible python-babel-localedata python3-argcomplete python3-babel

```

```

python3-distutils python3-dnspython python3-jinja2 python3-jmespath
python3-kerberos python3-lib2to3 python3-libcloud python3-lockfile
python3-ntlm-auth python3-pycryptodome python3-requests-kerberos
python3-requests-ntlm python3-requests-toolbelt python3-selinux
python3-simplejson python3-tz python3-winrm python3-xlrd
0 upgraded, 22 newly installed, 0 to remove and 7 not upgraded.
Need to get 26,1 MB of archives.
After this operation, 259 MB of additional disk space will be used.
Do you want to continue? [Y/n] Y
Get:1 http://mirror.tedra.es/ubuntu jammy/main amd64 python-babel-localedata
  all 2.8.0+dfsg.1-7 [4.982 kB]
Get:2 http://mirror.tedra.es/ubuntu jammy-updates/main amd64 python3-tz all
  2022.1-1ubuntu0.22.04.0 [33,4 kB]
Get:3 http://mirror.tedra.es/ubuntu jammy/main amd64 python3-babel all 2.8.0+
  dfsg.1-7 [85,1 kB]
Get:4 http://mirror.tedra.es/ubuntu jammy/main amd64 python3-jinja2 all
  3.0.3-1 [108 kB]
Get:5 http://mirror.tedra.es/ubuntu jammy/main amd64 python3-pycryptodome
  amd64 3.11.0+dfsg1-3build1 [1.027 kB]
Get:6 http://mirror.tedra.es/ubuntu jammy-updates/main amd64 python3-lib2to3
  all 3.10.6-1~22.04 [77,6 kB]
Get:7 http://mirror.tedra.es/ubuntu jammy-updates/main amd64 python3-distutils
  all 3.10.6-1~22.04 [139 kB]
Get:8 http://mirror.tedra.es/ubuntu jammy/main amd64 python3-dnspython all
  2.1.0-1ubuntu1 [123 kB]
Get:9 http://mirror.tedra.es/ubuntu jammy/universe amd64 ansible all 2.10.7+
  merged+base+2.10.8+dfsg-1 [17,5 MB]
Get:10 http://mirror.tedra.es/ubuntu jammy/universe amd64 python3-argcomplete
  all 1.8.1-1.5 [27,2 kB]
Get:11 http://mirror.tedra.es/ubuntu jammy/main amd64 python3-jmespath all
  0.10.0-1 [21,7 kB]
Get:12 http://mirror.tedra.es/ubuntu jammy/universe amd64 python3-kerberos
  amd64 1.1.14-3.1build5 [23,0 kB]
Get:13 http://mirror.tedra.es/ubuntu jammy/main amd64 python3-lockfile all
  1:0.12.2-2.2 [14,6 kB]
Get:14 http://mirror.tedra.es/ubuntu jammy/main amd64 python3-simplejson amd64
  3.17.6-1build1 [54,7 kB]
Get:15 http://mirror.tedra.es/ubuntu jammy/universe amd64 python3-libcloud all
  3.2.0-2 [1.554 kB]
Get:16 http://mirror.tedra.es/ubuntu jammy/universe amd64 python3-ntlm-auth
  all 1.4.0-1 [20,4 kB]
Get:17 http://mirror.tedra.es/ubuntu jammy/universe amd64 python3-requests-
  kerberos all 0.12.0-2 [11,9 kB]
Get:18 http://mirror.tedra.es/ubuntu jammy/universe amd64 python3-requests-
  ntlm all 1.1.0-1.1 [6.160 B]
Get:19 http://mirror.tedra.es/ubuntu jammy/main amd64 python3-requests-
  toolbelt all 0.9.1-1 [38,0 kB]
Get:20 http://mirror.tedra.es/ubuntu jammy/universe amd64 python3-selinux
  amd64 3.3-1build2 [159 kB]
Get:21 http://mirror.tedra.es/ubuntu jammy/universe amd64 python3-xlrd
  all 0.12.0-2 [12,6 kB]
Get:22 http://mirror.tedra.es/ubuntu jammy/universe amd64 python3-winrm all
  0.3.0-2 [21,7 kB]
Fetched 26,1 MB in 3s (8.628 kB/s)
Selecting previously unselected package python-babel-localedata.
(Reading database ... 610459 files and directories currently installed.)
Preparing to unpack .../00-python-babel-localedata_2.8.0+dfsg.1-7_all.deb ...
Unpacking python-babel-localedata (2.8.0+dfsg.1-7) ...

```

```

Selecting previously unselected package python3-tz.
Preparing to unpack .../01-python3-tz-2022.1-1ubuntu0.22.04.0_all.deb ...
Unpacking python3-tz (2022.1-1ubuntu0.22.04.0) ...
Selecting previously unselected package python3-babel.
Preparing to unpack .../02-python3-babel-2.8.0+dfsg.1-7_all.deb ...
Unpacking python3-babel (2.8.0+dfsg.1-7) ...
Selecting previously unselected package python3-jinja2.
Preparing to unpack .../03-python3-jinja2-3.0.3-1_all.deb ...
Unpacking python3-jinja2 (3.0.3-1) ...
Selecting previously unselected package python3-pycryptodome.
Preparing to unpack .../04-python3-pycryptodome-3.11.0+dfsg1-3build1_amd64.deb
...
Unpacking python3-pycryptodome (3.11.0+dfsg1-3build1) ...
Selecting previously unselected package python3-lib2to3.
Preparing to unpack .../05-python3-lib2to3-3.10.6-1~22.04_all.deb ...
Unpacking python3-lib2to3 (3.10.6-1~22.04) ...
Selecting previously unselected package python3-distutils.
Preparing to unpack .../06-python3-distutils-3.10.6-1~22.04_all.deb ...
Unpacking python3-distutils (3.10.6-1~22.04) ...
Selecting previously unselected package python3-dnspython.
Preparing to unpack .../07-python3-dnspython-2.1.0-1ubuntu1_all.deb ...
Unpacking python3-dnspython (2.1.0-1ubuntu1) ...
Selecting previously unselected package ansible.
Preparing to unpack .../08-ansible-2.10.7+merged+base+2.10.8+dfsg-1_all.deb
...
Unpacking ansible (2.10.7+merged+base+2.10.8+dfsg-1) ...
Selecting previously unselected package python3-argcomplete.
Preparing to unpack .../09-python3-argcomplete-1.8.1-1.5_all.deb ...
Unpacking python3-argcomplete (1.8.1-1.5) ...
Selecting previously unselected package python3-jmespath.
Preparing to unpack .../10-python3-jmespath-0.10.0-1_all.deb ...
Unpacking python3-jmespath (0.10.0-1) ...
Selecting previously unselected package python3-kerberos.
Preparing to unpack .../11-python3-kerberos-1.1.14-3.1build5_amd64.deb ...
Unpacking python3-kerberos (1.1.14-3.1build5) ...
Selecting previously unselected package python3-lockfile.
Preparing to unpack .../12-python3-lockfile-1%3a0.12.2-2.2_all.deb ...
Unpacking python3-lockfile (1:0.12.2-2.2) ...
Selecting previously unselected package python3-simplejson.
Preparing to unpack .../13-python3-simplejson-3.17.6-1build1_amd64.deb ...
Unpacking python3-simplejson (3.17.6-1build1) ...
Selecting previously unselected package python3-libcloud.
Preparing to unpack .../14-python3-libcloud-3.2.0-2_all.deb ...
Unpacking python3-libcloud (3.2.0-2) ...
Selecting previously unselected package python3-ntlm-auth.
Preparing to unpack .../15-python3-ntlm-auth-1.4.0-1_all.deb ...
Unpacking python3-ntlm-auth (1.4.0-1) ...
Selecting previously unselected package python3-requests-kerberos.
Preparing to unpack .../16-python3-requests-kerberos-0.12.0-2_all.deb ...
Unpacking python3-requests-kerberos (0.12.0-2) ...
Selecting previously unselected package python3-requests-ntlm.
Preparing to unpack .../17-python3-requests-ntlm-1.1.0-1.1_all.deb ...
Unpacking python3-requests-ntlm (1.1.0-1.1) ...
Selecting previously unselected package python3-requests-toolbelt.
Preparing to unpack .../18-python3-requests-toolbelt-0.9.1-1_all.deb ...
Unpacking python3-requests-toolbelt (0.9.1-1) ...
Selecting previously unselected package python3-selinux.

```

```

Preparing to unpack .../19-python3-selinux_3.3-1build2_amd64.deb ...
Unpacking python3-selinux (3.3-1build2) ...
Selecting previously unselected package python3-xltdict.
Preparing to unpack .../20-python3-xltdict_0.12.0-2_all.deb ...
Unpacking python3-xltdict (0.12.0-2) ...
Selecting previously unselected package python3-winrm.
Preparing to unpack .../21-python3-winrm_0.3.0-2_all.deb ...
Unpacking python3-winrm (0.3.0-2) ...
Setting up python3-lockfile (1:0.12.2-2.2) ...
Setting up python3-requests-toolbelt (0.9.1-1) ...
Setting up python3-ntlm-auth (1.4.0-1) ...
Setting up python3-pycryptodome (3.11.0+dfsg1-3build1) ...
Setting up python3-kerberos (1.1.14-3.1build5) ...
Setting up python3-tz (2022.1-1ubuntu0.22.04.0) ...
Setting up python-babel-localedata (2.8.0+dfsg.1-7) ...
Setting up python3-simplejson (3.17.6-1build1) ...
Setting up python3-xltdict (0.12.0-2) ...
Setting up python3-jmespath (0.10.0-1) ...
Setting up python3-requests-kerberos (0.12.0-2) ...
Setting up python3-dnspython (2.1.0-1ubuntu1) ...
Setting up python3-selinux (3.3-1build2) ...
Setting up python3-argcomplete (1.8.1-1.5) ...
Setting up python3-lib2to3 (3.10.6-1~22.04) ...
Setting up python3-distutils (3.10.6-1~22.04) ...
Setting up python3-requests-ntlm (1.1.0-1.1) ...
Setting up python3-babel (2.8.0+dfsg.1-7) ...
update-alternatives: using /usr/bin/pybabel-python3 to provide /usr/bin/
pybabel
(pybabel) in auto mode
Setting up python3-libcloud (3.2.0-2) ...
Setting up python3-jinja2 (3.0.3-1) ...
Setting up python3-winrm (0.3.0-2) ...
Setting up ansible (2.10.7+merged+base+2.10.8+dfsg-1) ...
Processing triggers for man-db (2.10.2-1) ...
tfg@mint00:~$ ansible --version
ansible 2.10.8
config file = None
configured module search path = ['/home/tfg/.ansible/plugins/modules', '/usr/
share/ansible/plugins/modules']
ansible python module location = /usr/lib/python3/dist-packages/ansible
executable location = /usr/bin/ansible
python version = 3.10.6 (main, Mar 10 2023, 10:55:28) [GCC 11.3.0]

tfg@Mint03:~$ sudo apt-get install sshpass
[sudo] password for tfg:
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following NEW packages will be installed:
sshpass
0 upgraded, 1 newly installed, 0 to remove and 8 not upgraded.
Need to get 11,7 kB of archives.
After this operation, 35,8 kB of additional disk space will be used.
Get:1 http://mirror.tedra.es/ubuntu jammy/universe amd64 sshpass amd64 1.09-1
[11,7 kB]
Fetched 11,7 kB in 0s (214 kB/s)
Selecting previously unselected package sshpass.
(Reading database ... 649950 files and directories currently installed.)

```



```
Preparing to unpack .../sshpas_1.09-1_amd64.deb ...
Unpacking sshpass (1.09-1) ...
Setting up sshpass (1.09-1) ...
Processing triggers for man-db (2.10.2-1) ...
```

instalar dependencias de ansible

```
sudo apt-get install ansible-core
```

## B.3. Instalación de de kubeadm

Para poder utilizar docker necesitamos instalar varios paquetes. Estos son Docker Engine, kubectl, kubelet y kubeadm. Pasamos a instalar cada uno individualmente.

### B.3.1. instalación kubectl

Hay que descargarse la última release:

```
curl -LO https://storage.googleapis.com/kubernetes-release/release/$(curl -s
https://storage.googleapis.com/kubernetes-release/release/stable.txt)/bin/
linux/amd64/kubectl
```

Para que el archivo sea ejecutable utilizamos chmod.

```
chmod +x kubectl
```

se mueve el archivo binario al path del usuario para ejecutarlo directamente desde la consola.

```
sudo mv kubectl /usr/local/bin/kubectl
```

Se comprueba que la instalación ha sido correcta mediante el comando kubectl. Debe devolver la ayuda del mismo.

```
tfg@mint01:~$ kubectl
kubectl controls the Kubernetes cluster manager.

Find more information at: https://kubernetes.io/docs/reference/kubectl/

Basic Commands (Beginner):
create          Create a resource from a file or from stdin
expose          Take a replication controller, service, deployment or pod and
expose it as a new Kubernetes service
run             Run a particular image on the cluster
set             Set specific features on objects

Basic Commands (Intermediate):
explain         Get documentation for a resource
get             Display one or many resources
edit           Edit a resource on the server
delete         Delete resources by file names, stdin, resources and names, or
by resources and label selector

Deploy Commands:
rollout         Manage the rollout of a resource
scale          Set a new size for a deployment, replica set, or replication
```

```

controller
autoscale      Auto-scale a deployment, replica set, stateful set, or
replication controller

Cluster Management Commands:
certificate     Modify certificate resources.
cluster-info   Display cluster information
top            Display resource (CPU/memory) usage
cordon         Mark node as unschedulable
uncordon       Mark node as schedulable
drain          Drain node in preparation for maintenance
taint          Update the taints on one or more nodes

Troubleshooting and Debugging Commands:
describe       Show details of a specific resource or group of resources
logs           Print the logs for a container in a pod
attach         Attach to a running container
exec           Execute a command in a container
port-forward   Forward one or more local ports to a pod
proxy          Run a proxy to the Kubernetes API server
cp             Copy files and directories to and from containers
auth           Inspect authorization
debug          Create debugging sessions for troubleshooting workloads and
nodes
events         List events

Advanced Commands:
diff           Diff the live version against a would-be applied version
apply          Apply a configuration to a resource by file name or stdin
patch          Update fields of a resource
replace        Replace a resource by file name or stdin
wait           Experimental: Wait for a specific condition on one or many
resources
kustomize      Build a kustomization target from a directory or URL

Settings Commands:
label          Update the labels on a resource
annotate       Update the annotations on a resource
completion     Output shell completion code for the specified shell (bash,
zsh, fish, or powershell)

Other Commands:
api-resources  Print the supported API resources on the server
api-versions   Print the supported API versions on the server, in the form of
"group/version"
config         Modify kubeconfig files
plugin         Provides utilities for interacting with plugins
version        Print the client and server version information

Usage:
kubectl [flags] [options]

Use "kubectl <command> --help" for more information about a given command.
Use "kubectl options" for a list of global command-line options (applies to all
commands).
```

Se pueden ejecutar los 4 comando a la vez mediante la instrucción:

```
curl -LO https://storage.googleapis.com/kubernetes-release/release/$(curl -s
https://storage.googleapis.com/kubernetes-release/release/stable.txt)/bin/
linux/amd64/kubectl && chmod +x ./kubectl && sudo mv ./kubectl /usr/local/
bin/kubectl && kubectl
```

## B.3.2. Instalación de docker Engine

### B.3.2.1. Configuración del repositorio

Primero, se actualiza el paquete apt y se instalan los certificados necesarios

```
sudo apt-get update
sudo apt-get install \
ca-certificates \
curl \
gnupg
```

Se debe añadir la GPG key oficial de docker a los keyrings.

```
sudo install -m 0755 -d /etc/apt/keyrings
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o
/etc/apt/keyrings/docker.gpg
sudo chmod a+r /etc/apt/keyrings/docker.gpg
```

Se añade el repositorio para la descarga y configuración de docker al archivo de configuración de repositorios para docker.

```
echo \
"deb [arch="$(dpkg --print-architecture)" signed-by=/etc/apt/keyrings/docker.gpg
] https://download.docker.com/linux/ubuntu \
"$(. /etc/os-release && echo "$VERSION_CODENAME")" stable" | \
sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
```

Una vez configurado el repositorio, se ejecuta del comando apt-get update.

```
sudo apt-get update
```

En este punto puede ser que haya un error al no existir una versión de Docker para la distribución de linux sobre la que estamos trabajando.

```
sudo apt-get update
Ign:1 https://ftp.crifo.org/mint-packages vera InRelease
Hit:2 https://ppa.launchpadcontent.net/ansible/ansible/ubuntu jammy InRelease
Hit:3 https://ftp.crifo.org/mint-packages vera Release
Hit:4 http://archive.ubuntu.com/ubuntu jammy InRelease
Hit:5 http://security.ubuntu.com/ubuntu jammy-security InRelease
Ign:7 https://download.docker.com/linux/ubuntu vera InRelease
Hit:8 http://archive.ubuntu.com/ubuntu jammy-updates InRelease
Hit:9 http://archive.ubuntu.com/ubuntu jammy-backports InRelease
Err:10 https://download.docker.com/linux/ubuntu vera Release
404 Not Found [IP: 18.67.240.122 443]
Hit:11 https://download.sublimetext.com apt/stable/ InRelease
Reading package lists... Done
E: The repository 'https://download.docker.com/linux/ubuntu vera Release' does
not have a Release file.
```

```
N: Updating from such a repository can't be done securely, and is therefore
disabled by default.
N: See apt-secure(8) manpage for repository creation and user configuration
details.
```

De darse el caso, se debe editar el archivo `/etc/apt/sources.list.d/docker.list` con la versión de docker disponible más cercana a la que está en ejecución. Para ello hay que buscar en <https://download.docker.com/linux/ubuntu/dists/> las distribuciones para las que docker tiene releases y se elige la que más convenga. Para saber qué distribución es la del SO en uso, se revisa el archivo `/etc/os-release` y se elige el `VERSION_CODENAME` o el `UBUNTU_CODENAME`. Finalmente se instalan los paquetes necesarios para que docker funcione correctamente:

```
sudo apt-get install docker-ce docker-ce-cli containerd.io docker-buildx-plugin
docker-compose-plugin
```

Se puede verificar la correcta instalación ejecutando un *hello world* docker style.

```
sudo docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
2db29710123e: Pull complete
Digest: sha256:4e83453afed1b4fa1a3500525091dbfca6ce1e66903fd4c01ff015dbcb1ba33e
Status: Downloaded newer image for hello-world:latest

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
1. The Docker client contacted the Docker daemon.
2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
   (amd64)
3. The Docker daemon created a new container from that image which runs the
   executable that produces the output you are currently reading.
4. The Docker daemon streamed that output to the Docker client, which sent it
   to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
$ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
https://hub.docker.com/

For more examples and ideas, visit:
https://docs.docker.com/get-started/
```

### B.3.2.2. Instalación de kubelet y kubeadm

El último paso antes de empezar a crear el clúster es instalar kubelet y kubeadm, para ello se ejecutan los siguientes comandos en la terminal **con un usuario con permisos de root**.

```
apt-get update && apt-get install -y apt-transport-https

curl -s https://packages.cloud.google.com/apt/doc/apt-key.gpg | apt-key add -

cat <<EOF >/etc/apt/sources.list.d/kubernetes.list
deb http://apt.kubernetes.io/ kubernetes-xenial main
```

EOF

`apt-get update``apt-get install -y kubelet kubeadm`

## B.4. Creación del cluster

El master va a ser el mint00. Accedemos a MV y ejecutamos la siguiente instrucción donde la IP configurada como *apiserver-advertise-address* es la del adaptador que hace de bridge (Ver dirección MAC) en B.1 y B.3 y la que queda como *-pod-network-cidr* es la NAT. B.2

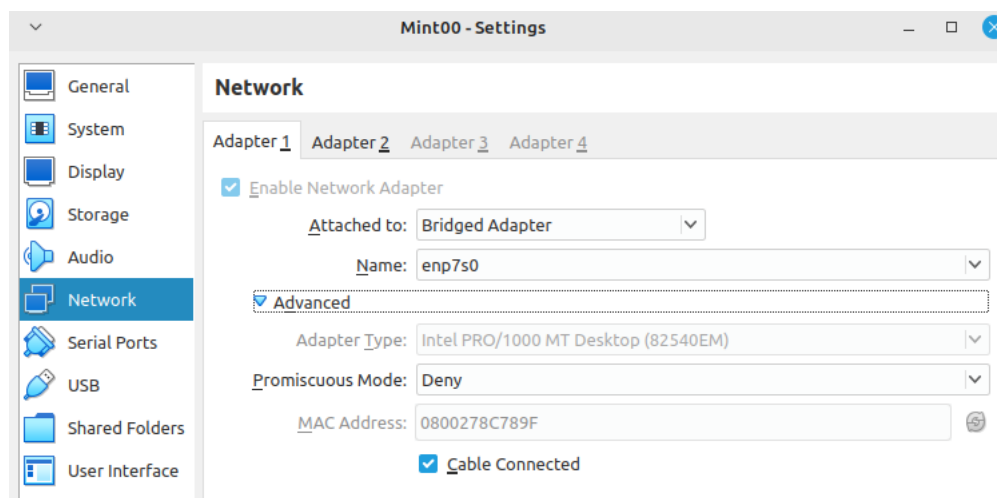


Figura B.1: Configuración Adaptador 1.

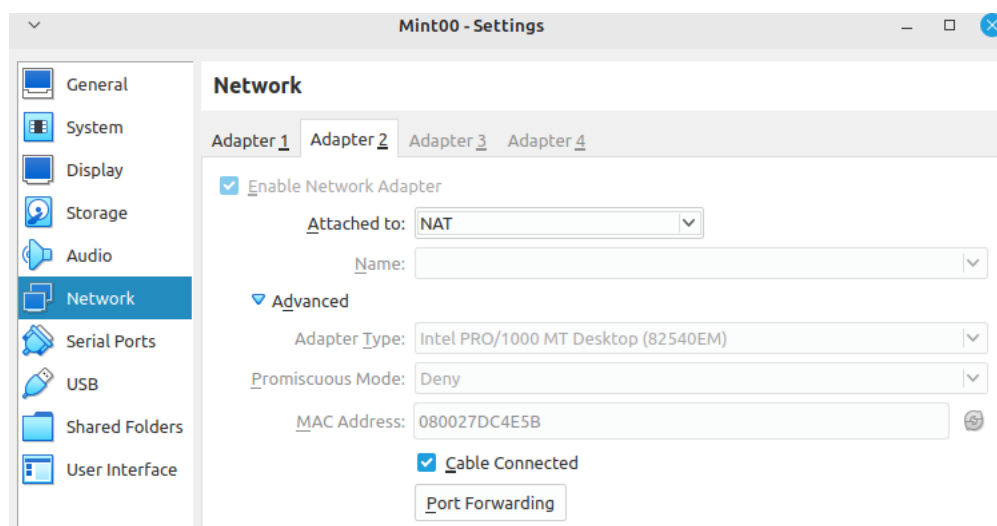


Figura B.2: Configuración Adaptador 2.

```
sudo kubeadm init --apiserver-advertise-address 192.168.1.100 --pod-network-cidr=10.0.3.15
```

En este momento puede surgir un error.

```
tfg@mint00:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:8c:78:9f brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.100/24 brd 192.168.1.255 scope global noprefixroute enp0s3
        valid lft forever preferred_lft forever
    inet6 fe80::48e6:a9a0:ce72:2682/64 scope link noprefixroute
        valid lft forever preferred_lft forever
3: enp0s8: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:dc:4e:5b brd ff:ff:ff:ff:ff:ff
    inet 10.0.3.15/24 brd 10.0.3.255 scope global dynamic noprefixroute enp0s8
        valid lft 80415sec preferred_lft 80415sec
    inet6 fe80::51d9:3916:76cb:c804/64 scope link noprefixroute
        valid lft forever preferred_lft forever
4: docker0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc noqueue state DOWN group default
    link/ether 02:42:fb:4f:8b:aa brd ff:ff:ff:ff:ff:ff
    inet 172.17.0.1/16 brd 172.17.255.255 scope global docker0
        valid lft forever preferred_lft forever
tfg@mint00:~$
```

Figura B.3: Direcciones de los adaptadores en la Máquina Virtual.

```
sudo kubeadm init --apiserver-advertise-address 192.168.1.100 --pod-network-cidr=10.0.3.15
[sudo] password for tfg:
[init] Using Kubernetes version: v1.27.1
[preflight] Running pre-flight checks
[WARNING Swap]: swap is enabled; production deployments should disable swap unless testing the NodeSwap feature gate of the kubelet
error execution phase preflight: [preflight] Some fatal errors occurred:
[ERROR CRI]: container runtime is not running: output: time="2023-05-06T17:35:28+02:00" level=fatal msg="validate service connection: CRI v1 runtime API is not implemented for endpoint \"unix:///var/run/containerd/containerd.sock\": rpc error: code = Unimplemented desc = unknown service runtime.v1.RuntimeService"
, error: exit status 1
[preflight] If you know what you are doing, you can make a check non-fatal with '--ignore-preflight-errors=...'
To see the stack trace of this error execute with --v=5 or higher
```

Esto es un error del paquete containerd que se puede resolver de la siguiente manera: como usuario root, se elimina el paquete containerd, se actualiza e instala de nuevo el paquete, se elimina la configuración por defecto y se reinicia el programa.

```
apt remove containerd
apt update && apt install containerd.io
rm /etc/containerd/config.toml
systemctl restart containerd
```

Una vez resuelto este inconveniente podemos iniciar el master. Para ello, usamos el comando y esperamos a que arranque satisfactoriamente. A continuación vemos algunas de las líneas de salida de la ejecución.

```
sudo kubeadm init --apiserver-advertise-address 192.168.1.100 --pod-network-cidr=10.0.3.15
[init] Using Kubernetes version: v1.27.1
[preflight] Running pre-flight checks
[preflight] Pulling images required for setting up a Kubernetes cluster
```

```
[preflight] This might take a minute or two, depending on the speed of your
internet connection

[certs] apiserver serving cert is signed for DNS names [kubernetes kubernetes
.default kubernetes.default.svc kubernetes.default.svc.cluster.local mint00]
and IPs [10.96.0.1 192.168.1.100]
[apiclient] All control plane components are healthy after 4.001839 seconds

Your Kubernetes control-plane has initialized successfully!

To start using your cluster, you need to run the following as a regular user:

mkdir -p $HOME/.kube
sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
sudo chown $(id -u):$(id -g) $HOME/.kube/config

Alternatively, if you are the root user, you can run:

export KUBECONFIG=/etc/kubernetes/admin.conf

You should now deploy a pod network to the cluster.
Run "kubectl apply -f [podnetwork].yaml" with one of the options listed at:
https://kubernetes.io/docs/concepts/cluster-administration/addons/

Then you can join any number of worker nodes by running the following on each
as root:

kubeadm join 192.168.1.100:6443 --token k79hza.63kxu2jlx6f6qzdb \
--discovery-token-ca-cert-hash sha256:83
c9ec0ea6a42546598b01fa1a54f52f2bcdb2109395f3605772740922d788a0
```

Además muestra el comando a ejecutar para unir otros nodos a la red. Antes de añadir los demás pods se deben ejecutar los siguientes comando sólo en el nodo master para guardar la configuración:

```
mkdir -p $HOME/.kube && sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/
config && sudo chown $(id -u):$(id -g) $HOME/.kube/config
```

Puede ser que al intentar agregar los demás nodos muestre el error del paquete CRI como en el nodo master. Se resuelve de la misma manera. Deshabilitamos la memoria swap para que no muestre el warning y ejecutamos el comando.

```
sudo swapoff -a
sudo kubeadm join 192.168.1.100:6443 --token k79hza.63kxu2jlx6f6qzdb \
--discovery-token-ca-cert-hash sha256:83
c9ec0ea6a42546598b01fa1a54f52f2bcdb2109395f3605772740922d788a0
```

Para comprobar que se han creado correctamente, se ejecuta el comando:

```
kubectl get nodes
```

Hace falta un paso extra que se ha realizado en este punto de la configuración. Se va a trabajar con flannel para configurar la red. Para ello es necesario descargarse la configuración escribiendo:

```
wget https://raw.githubusercontent.com/coreos/flannel/master/Documentation/
kube-flannel.yml
```

Una vez descargado, se debe modificar el archivo B.4, añadiendo a la sección *flanneld*, la interfaz correspondiente, en este caso se añade la línea - *-iface=enp0s3* B.3 con el identificador del adaptador configurado como bridge.

```

159     - -f
160     - /etc/kube-flannel/cni-conf.json
161     - /etc/cni/net.d/10-flannel.conflist
162     volumeMounts:
163     - name: cni
164       mountPath: /etc/cni/net.d
165     - name: flannel-cfg
166       mountPath: /etc/kube-flannel/
167   containers:
168   - name: kube-flannel
169     image: docker.io/flannel/flannel:v0.21.5
170     #image: docker.io/rancher/mirrored-flannelcni-flannel:v0.21.5
171     command:
172     - /opt/bin/flanneld
173     args:
174     - --ip-masq
175     - --kube-subnet-mgr
176     - --iface=enp0s3
177   resources:

```

Figura B.4: kube-flannel.yml.

Una vez realizada toda la configuración correctamente, se observan los nodos agregados al cluster.

```

tfg@mint00:~$ kubectl get nodes
NAME      STATUS   ROLES    AGE     VERSION
mint00    Ready    control-plane   98m    v1.27.1
mint01    Ready    <none>         97m    v1.27.1
mint02    Ready    <none>         97m    v1.27.1
mint03    Ready    <none>         97m    v1.27.1

```

Figura B.5: Ejecución del comando get nodes.

Se puede configurar

## B.5. Instalar programas de obs

### B.5.1. install obs studio

```

tfg@tfg:\$ docker pull somatorio/obs-studio
Using default tag: latest
latest: Pulling from somatorio/obs-studio
b3e1c725a85f: Pull complete
4daad8bdde31: Pull complete
63fe8c0068a8: Pull complete
4a70713c436f: Pull complete
bd842a2105a8: Pull complete
ad2d0b4ee955: Pull complete
Digest: sha256:583
       b32f6719ff422f70cc553d97968b34025f0bc3cc1e5495bcb46c7ea672fb3
Status: Downloaded newer image for somatorio/obs-studio:latest
docker.io/somatorio/obs-studio:latest

```

```

tfg@tfg:/home/tfg# sudo apt-get update && apt-get install -y obs-studio curl
ffmpeg

```