Vale 👍. El PDF que has subido es la guía del **Laboratorio 4: NFS y MQTT**.  
 Lo que tienes que hacer es montar un sistema distribuido con **4 máquinas virtuales (MV)**, cada una con un rol distinto. Te resumo los pasos principales:

### **🔹 MV1: Recolección de datos**

* Crear un **script** que:  
  + Genere una temperatura aleatoria (10.0 a 24.0 °C) con un decimal y añada fecha y hora.
  + Envíe el dato a un **topic MQTT** llamado campo/temperatura en MV2.
  + Use el usuario **sensor** para autenticación en el broker.
* Configurar el script para que se ejecute **cada minuto** (ej: con cron).

### **🔹 MV2: Broker MQTT**

* Instalar y configurar **Mosquitto** como broker MQTT.  
  + No debe permitir conexiones anónimas.
  + Crear usuario **sensor** con permisos de **publicar** en campo/temperatura.
* Montar carpeta **NFS** exportada desde MV3 en /tmp/carpetaRemota.
* Escribir cada mensaje recibido en HistoricoTemperatura.txt dentro de esa carpeta.

### **🔹 MV3: Servidor NFS**

* Configurar un servidor NFS que exporte /datosNfs.  
  + MV2 → permisos de **lectura y escritura**.
  + MV4 → permisos de **solo lectura**.

### **🔹 MV4: Servidor Web**

* Montar carpeta NFS (/datosNfs de MV3) en /tmp/carpetaRemota.
* Crear un fichero **index.html** que muestre los últimos datos.
* Levantar un servidor web con:  
    
   sudo python3 -m http.server 80
* Hacer un **script** que, cada minuto, lea el último dato de temperatura del fichero de la carpeta NFS y actualice index.html.
* La web debe mostrarse en http://<IP-pública-MV4>.

### **🔹 Extensión opcional**

* Añadir **humedad** (valores entre 50 y 60).
* Nuevo topic campo/humedad.
* Nuevo fichero de datos.
* Modificar index.html para mostrar **temperatura y humedad**.

👉 En resumen:

1. **MV1 genera datos y los manda por MQTT.**
2. **MV2 (broker) recibe y guarda en NFS.**
3. **MV3 es el servidor NFS que centraliza los datos.**
4. **MV4 muestra en una web los datos leídos desde NFS.**

¿Quieres que te prepare los **scripts listos para usar** (para MV1, MV2 y MV4) o prefieres que te dé solo la explicación conceptual de cómo configurarlo?

Perfecto 🚀. Vamos a configurar **MV2 (Broker MQTT + cliente que escribe en NFS)** paso a paso.

# **🔹 Instrucciones para MV2**

## **1. Instalar dependencias**

Primero actualiza el sistema e instala Mosquitto, clientes MQTT y NFS:

sudo apt update && sudo apt upgrade -y

sudo apt install -y mosquitto mosquitto-clients nfs-common

## **2. Configurar el broker Mosquitto**

Edita la configuración de Mosquitto:

sudo nano /etc/mosquitto/mosquitto.conf

Añade (o asegúrate de tener):

allow\_anonymous false

password\_file /etc/mosquitto/passwd

listener 1883

### **Crear usuario sensor**

sudo mosquitto\_passwd -c /etc/mosquitto/passwd sensor

👉 Te pedirá contraseña (por ejemplo: 1234).

Reinicia el servicio:

sudo systemctl restart mosquitto

sudo systemctl enable mosquitto

## **3. Montar la carpeta NFS**

Suponiendo que **MV3 exporta /datosNfs**, crea el punto de montaje:

sudo mkdir -p /tmp/carpetaRemota

Edita /etc/fstab:

sudo nano /etc/fstab

Añade esta línea (cambia IP\_MV3 por la IP real de MV3):

IP\_MV3:/datosNfs /tmp/carpetaRemota nfs defaults 0 0

Monta:

sudo mount -a

Comprueba:

df -h | grep carpetaRemota

## **4. Suscribirse al topic y guardar en fichero**

Queremos que cada mensaje recibido en campo/temperatura se guarde en:  
 /tmp/carpetaRemota/HistoricoTemperatura.txt

Crea un **servicio con systemd** para automatizarlo:

sudo nano /etc/systemd/system/mqtt-logger.service

Pon esto dentro (ajustando la contraseña):

[Unit]

Description=MQTT to NFS logger

After=network.target

[Service]

ExecStart=/usr/bin/mosquitto\_sub -h localhost -p 1883 -u sensor -P 1234 -t campo/temperatura -v >> /tmp/carpetaRemota/HistoricoTemperatura.txt

Restart=always

[Install]

WantedBy=multi-user.target

Guarda y cierra. Luego:

sudo systemctl daemon-reload

sudo systemctl enable mqtt-logger

sudo systemctl start mqtt-logger

Comprueba que funciona:

tail -f /tmp/carpetaRemota/HistoricoTemperatura.txt

Si MV1 ya envía datos, deberías ver líneas nuevas cada minuto.

# **✅ Resumen**

1. Instalar Mosquitto y NFS.
2. Configurar Mosquitto con usuario sensor.
3. Montar carpeta NFS desde MV3 en /tmp/carpetaRemota.
4. Crear servicio que escucha campo/temperatura y escribe en HistoricoTemperatura.txt.

¿Quieres que también te prepare los **comandos exactos de configuración en MV3** para que MV2 pueda montar la carpeta NFS sin problemas?

Perfecto, gracias por darme el error 👀.

El mensaje **mount.nfs: Connection refused** en **MV2** significa que la máquina **MV3 (servidor NFS)** no está sirviendo todavía la carpeta o el firewall está bloqueando la conexión.

Vamos a solucionarlo:

## **🔹 Pasos en MV3 (Servidor NFS)**

### **1. Instalar y habilitar NFS**

En MV3:

sudo apt update && sudo apt install -y nfs-kernel-server

### **2. Crear carpeta compartida**

sudo mkdir -p /datosNfs

sudo chown nobody:nogroup /datosNfs

### **3. Configurar exportaciones**

Edita /etc/exports:

sudo nano /etc/exports

Añade (reemplaza IP\_MV2 y IP\_MV4 con sus direcciones reales):

/datosNfs IP\_MV2(rw,sync,no\_subtree\_check) IP\_MV4(ro,sync,no\_subtree\_check)

Guarda y aplica cambios:

sudo exportfs -ra

### **4. Reiniciar el servicio NFS**

sudo systemctl restart nfs-kernel-server

sudo systemctl enable nfs-kernel-server

## **🔹 Comprobar desde MV3**

En MV3, revisa qué exporta:

showmount -e

Debe mostrar algo como:

/datosNfs IP\_MV2

/datosNfs IP\_MV4

## **🔹 Intentar montar de nuevo en MV2**

Ahora vuelve a probar en MV2:

sudo mount -a

Si quieres probar manualmente:

sudo mount -t nfs IP\_MV3:/datosNfs /tmp/carpetaRemota

Y comprueba:

df -h | grep carpetaRemota

## **🔹 Posibles problemas**

* Si sigue dando error, puede ser **firewall** en MV3. Abre el puerto NFS:

sudo ufw allow from IP\_MV2 to any port nfs

sudo ufw allow from IP\_MV4 to any port nfs

o directamente:

sudo ufw allow nfs

Vale, ya estamos más cerca 🚀.  
 Ese error significa que la carpeta **sí se montó**, pero el fichero **HistoricoTemperatura.txt todavía no existe** en el NFS.

En tu **servicio de MV2** (mqtt-logger.service) configuraste que mosquitto\_sub escriba en:

/tmp/carpetaRemota/HistoricoTemperatura.txt

Si el fichero no existe, no se creará automáticamente al arrancar tail, sino cuando **llegue el primer mensaje MQTT** desde MV1.

## **🔹 Qué hacer ahora**

### **1. Crear el fichero vacío en MV3 (opcional)**

En **MV3** (servidor NFS), puedes crearlo a mano para evitar el error de tail:

sudo touch /datosNfs/HistoricoTemperatura.txt

sudo chmod 666 /datosNfs/HistoricoTemperatura.txt

👉 Así ya existirá en /tmp/carpetaRemota en MV2 y podrás hacer tail -f.

### **2. Probar si MV2 está escribiendo bien**

En **MV2**, ejecuta manualmente:

mosquitto\_sub -h localhost -p 1883 -u sensor -P 1234 -t campo/temperatura -v

👉 Si MV1 está funcionando y publicando, deberías ver algo como:

campo/temperatura 2025-10-01 12:34:56 Temperatura: 19.7

### **3. Probar publicación desde MV1 (o manualmente)**

Si aún no configuraste MV1, prueba desde **MV2** a publicar manualmente:

mosquitto\_pub -h localhost -p 1883 -u sensor -P 1234 -t campo/temperatura -m "20.5"

Después revisa en el fichero NFS:

cat /tmp/carpetaRemota/HistoricoTemperatura.txt

Debería aparecer tu mensaje.

### **4. Verificar el servicio**

Asegúrate de que el servicio que creamos está activo:

sudo systemctl status mqtt-logger

Si está en marcha, cada mensaje que llegue al broker se añadirá al fichero.

👉 Pregunta: ¿quieres que te prepare también el **script de MV1** para que empiece a mandar datos de prueba automáticamente cada minuto y así confirmar que MV2 los escribe en NFS?

Muy bien, ahora vemos el problema exacto 👌.

El servicio **mqtt-logger.service** falla con **status=1/FAILURE**. Eso suele pasar porque:

1. El comando de ExecStart no es aceptado por systemd (el >> no funciona directamente en servicios).
2. La autenticación MQTT está mal (usuario/contraseña).
3. La carpeta o el fichero aún no existen.

## **🔹 Solución**

En systemd no se puede usar >> directamente.  
 Necesitamos usar **sh -c** para redirigir la salida.

### **1. Edita el servicio**

sudo nano /etc/systemd/system/mqtt-logger.service

Reemplaza la sección [Service] por esta:

[Service]

ExecStart=/bin/sh -c '/usr/bin/mosquitto\_sub -h localhost -p 1883 -u sensor -P 1234 -t campo/temperatura -v >> /tmp/carpetaRemota/HistoricoTemperatura.txt'

Restart=always

Guarda y cierra.

### **2. Recargar y reiniciar**

sudo systemctl daemon-reload

sudo systemctl restart mqtt-logger

sudo systemctl status mqtt-logger

### **3. Verificar funcionamiento**

Haz un publish manual:

mosquitto\_pub -h localhost -p 1883 -u sensor -P 1234 -t campo/temperatura -m "22.5"

Luego revisa:

tail -f /tmp/carpetaRemota/HistoricoTemperatura.txt

Deberías ver:

campo/temperatura 22.5

👉 Pregunta: ¿quieres que te prepare también una **versión más robusta del servicio** que cree el fichero si no existe y añada timestamp automáticamente a cada línea?

Perfecto 💡, ahora toca la **extensión**: además de temperatura, el sistema también debe manejar **humedad**.

Te detallo los cambios en cada máquina:

# **🔹 MV1 – Generador de datos**

Actualmente generas solo **temperatura**. Ahora debes:

1. **Generar humedad**: un número entero aleatorio entre 50 y 60.
2. **Publicar en un nuevo topic**: campo/humedad.
3. El script quedaría así (ejemplo en Python):

#!/usr/bin/env python3

import random

import datetime

import paho.mqtt.publish as publish

import time

BROKER = "IP\_MV2" # sustituir por la IP de MV2

USER = "sensor"

PASSWORD = "1234"

while True:

fecha = datetime.datetime.now().strftime("%Y-%m-%d %H:%M:%S")

temperatura = round(random.uniform(10.0, 24.0), 1)

humedad = random.randint(50, 60)

publish.single("campo/temperatura", f"{fecha} Temperatura: {temperatura}", hostname=BROKER, auth={'username':USER, 'password':PASSWORD})

publish.single("campo/humedad", f"{fecha} Humedad: {humedad}", hostname=BROKER, auth={'username':USER, 'password':PASSWORD})

time.sleep(60) # cada minuto

# **🔹 MV2 – Broker y logger**

Ahora necesitas **dos procesos** de mosquitto\_sub:

* Uno que guarde en /tmp/carpetaRemota/HistoricoTemperatura.txt
* Otro que guarde en /tmp/carpetaRemota/HistoricoHumedad.txt

👉 Puedes hacerlo con **dos servicios systemd**, por ejemplo:

### **Servicio para temperatura**

/etc/systemd/system/mqtt-temp-logger.service

[Unit]

Description=MQTT Temperature Logger

After=network.target

[Service]

ExecStart=/bin/sh -c '/usr/bin/mosquitto\_sub -h localhost -p 1883 -u sensor -P 1234 -t campo/temperatura -v >> /tmp/carpetaRemota/HistoricoTemperatura.txt'

Restart=always

[Install]

WantedBy=multi-user.target

### **Servicio para humedad**

/etc/systemd/system/mqtt-hum-logger.service

[Unit]

Description=MQTT Humidity Logger

After=network.target

[Service]

ExecStart=/bin/sh -c '/usr/bin/mosquitto\_sub -h localhost -p 1883 -u sensor -P 1234 -t campo/humedad -v >> /tmp/carpetaRemota/HistoricoHumedad.txt'

Restart=always

[Install]

WantedBy=multi-user.target

Activarlos:

sudo systemctl daemon-reload

sudo systemctl enable --now mqtt-temp-logger mqtt-hum-logger

# **🔹 MV3 – Servidor NFS**

No hay que cambiar nada, solo asegúrate de que /datosNfs sigue exportándose para MV2 (RW) y MV4 (RO).  
 Los nuevos ficheros (HistoricoHumedad.txt) se crearán solos cuando MV2 escriba.

# **🔹 MV4 – Servidor Web**

Ahora debes mostrar **ambos valores** en la web.

1. El script que actualiza index.html debe leer **el último dato de cada fichero**:  
   * /tmp/carpetaRemota/HistoricoTemperatura.txt
   * /tmp/carpetaRemota/HistoricoHumedad.txt
2. Crear un index.html con el siguiente formato mínimo:

<html>

<head><title>Monitor del Campo</title></head>

<body>

<h1>Datos Meteorológicos</h1>

<p>Última temperatura: 20.5 ºC</p>

<p>Última humedad: 55 %</p>

</body>

</html>

1. Script de actualización en Python (cada minuto con cron o systemd):

#!/usr/bin/env python3

import os

temp\_file = "/tmp/carpetaRemota/HistoricoTemperatura.txt"

hum\_file = "/tmp/carpetaRemota/HistoricoHumedad.txt"

html\_file = "/var/www/html/index.html" # o la carpeta que uses con http.server

def ultima\_linea(fichero):

try:

with open(fichero, "r") as f:

return f.readlines()[-1].strip()

except:

return "No disponible"

temp = ultima\_linea(temp\_file)

hum = ultima\_linea(hum\_file)

html = f"""

<html>

<head><title>Monitor del Campo</title></head>

<body>

<h1>Datos Meteorológicos</h1>

<p>{temp}</p>

<p>{hum}</p>

</body>

</html>

"""

with open(html\_file, "w") as f:

f.write(html)

# **✅ Resumen de extensión**

* **MV1** ahora publica temperatura y humedad.
* **MV2** tiene dos servicios systemd para guardar ambos en ficheros separados.
* **MV3** sigue siendo el servidor NFS central.
* **MV4** actualiza la web leyendo ambos ficheros y mostrando los dos datos.

¿Quieres que te escriba directamente los **dos scripts listos para MV4** (uno que actualiza y otro que lanza el servidor web), o prefieres hacerlo con un solo script que combine ambas cosas?