Curso
Herramientas de Computación en la Nube

Clase 6

Febrero 14 de 2026

RustFS

Taller práctico

Este documento se fundamenta en los principios de arquitectura y sistemas de datos descritos en **Designing Data-Intensive Applications**, *aplicados a un caso académico de analítica operativa. Consultar la bibliografía al final.*

Profesor

Alvaro Mauricio Montenegro Díaz
Universidad de la Sabana

Febrero 2026

Esquema del Taller Tutorial: RustFS desde cero (S3-compatible)

En este taller, Rust se instalará para acceso directo en tu computador. En proximas sesiones o instalaremos con Docker, al crear el proyecto de curso.

1. Preflight

- Objetivo del taller y checklist
- Requisitos mínimos (macOS Apple Silicon)
- Carpeta de trabajo del lab
- Puertos y endpoint local
- Herramientas: `rustfs`, `curl`, `awscli` (o `s5cmd` opcional)

2. Instalación y verificación

- Descargar/instalar RustFS (binario ARM64)
- Verificar versión
- Arrancar el servicio local
- Confirmar salud del servicio (endpoint / health)
- Confirmar credenciales (access key / secret key)

Entregable: RustFS corriendo local con endpoint funcional.

3. Configurar cliente S3

- Opción A: AWS CLI (`aws configure`)
- Opción B (opcional): `s5cmd` para operaciones masivas
- Definir endpoint `http://localhost:PORT`
- Probar comando de “list buckets” (debe responder aunque esté vacío)

Entregable: cliente listo apuntando a RustFS.

4. Operaciones básicas con buckets

- Crear bucket

- Listar buckets
- Validar existencia
- Política básica (si RustFS soporta policy/ACL; si no, lo dejamos simple)
- Borrar bucket (primero debe estar vacío)

Entregable: bucket creado y gestionado.

5. Operaciones con objetos (CRUD)

- Subir objetos:
  - `PUT` con `aws s3 cp`
  - Subir un archivo grande para ver performance
  - Subir a prefijos (carpetas lógicas)
- Listar objetos:
  - listado simple
  - listado recursivo
- Descargar/recuperar:
  - `GET` a archivo local
  - verificación con `shasum`
- Borrar objetos:
  - borrar 1 objeto
  - borrar prefijo completo
- Manejo de “metadata”:
  - content-type
  - etiquetas (si aplica)
  - headers

Entregable: CRUD completo probado + verificación de integridad.

6. Versionado y protección (si RustFS local lo soporta)

- Activar versioning en bucket
- Sobrescribir un objeto varias veces
- Listar versiones

- Restaurar una versión
- Borrar una versión específica

Entregable: recuperación de versiones demostrada.

7. Presigned URLs (si RustFS lo soporta)

- Generar URL firmada para upload (PUT)
- Generar URL firmada para download (GET)
- Probar con `curl`

Entregable: compartir acceso temporal sin credenciales.

8. Replicabilidad del laboratorio

- Script “reset del lab”:
  - detener servicio
  - limpiar datos
  - levantar de nuevo
- Checklist final de comandos
- Troubleshooting rápido (403, signature mismatch, endpoint, reloj/hora)

Entregable: laboratorio reproducible en 5 minutos.

9. Mini-reto final (práctico)

- Crear bucket `datalake`
- Subir datos a:
  - `raw/`
  - `bronze/`
  - `silver/`
  - `gold/`
- Recuperar un archivo, validar hash, borrar y vaciar bucket

Desarrollo del Taller Tutorial: RustFS

Las siguientes instrucciones son solamente para MAC. RustFS se instal directamente desde los comandos generales. En todo caso, vaya la Github de RustFS para indicaciones de instalación en tu sistema operativo (Windows o Linux): https://docs.rustfs.com/installation/.

1. Instalación en MAC

1.1. Paso 1 — Crear carpeta del lab

mkdir -p ~/rustfs-lab
cd ~/rustfs-lab

1.2. Instalación nativa de MAC

brew tap rustfs/homebrew-tap
brew install rustfs

Verifica mediante

rustfs --version
rustfs –help

1.3. Levantar el servicio de Rust (server)

Definir credenciales del laboratorio

export RUSTFS_ACCESS_KEY=rustfsadmin
export RUSTFS_SECRET_KEY=rustfsadmin

Arranca el servicio

```
rustfs \
  --address :9000 \
  --console-enable \
  --console-address :9001 \
  ~/rustfs-lab/data
```

En la terminal debes ver algo como:

API: http://localhost:9000
Console: http://localhost:9001

En tu navegador ve a

http://localhost:9001

credenciales por defecto, según el help : rustfs –help

Access key: `rustfsadmin`
Secret key: `rustfsadmin`

2. Verificar que el servidor responde

Abre otra terminal.

Probar endpoint:

curl http://localhost:9000

Debe responder con algo (XML o JSON según implementación).

3. Cliente S3 para operar buckets/objetos (AWS CLI)

Ya estamos listos y listas para comenzar a trabajar con RustFS. Primero instalamos un cliente para acceder por la interface de AWS.

3.1. Instala AWS CLI

brew install awscli
aws --version

3.2. Configura un perfil

aws configure --profile rustfs

Pon:

- *Access key:* `rustfsadmin`
- *Secret key:* `rustfsadmin`
- *Region:* `us-east-1`
- *Output:* `json`

4. Taller practico CRUD

***CRUD** es un acrónimo fundamental en ingeniería de software y bases de datos:*

- ***C** → **Create** (Crear)*
- ***R** → **Read** (Leer)*
- ***U** → **Update** (Actualizar)*
- ***D** → **Delete** (Eliminar)*

*Representa las **cuatro operaciones básicas** que pueden realizarse sobre datos persistentes. Es el núcleo de cualquier sistema transaccional.*

4.1. Crear bucket (balde)

*aws s3api create-bucket *

*--bucket test-bucket *

*--endpoint-url http://localhost:9000 *

--profile rustfs

4.2. Listar objetos

aws s3api list-buckets \
--endpoint-url http://localhost:9000 \
--profile rustfs

4.3. Subir objeto

Crear archivo de prueba:

echo "Hola RustFS" > archivo.txt

Subirlo

```
aws s3 cp archivo.txt s3://test-bucket/ \
  --endpoint-url http://localhost:9000 \
  --profile rustfs
```

4.4. Lista objetos

```
aws s3 ls s3://test-bucket \
  --endpoint-url http://localhost:9000 \
  --profile rustfs
```

4.5. Descargar objeto

```
aws s3 cp s3://test-bucket/archivo.txt descargado.txt \
  --endpoint-url http://localhost:9000 \
  --profile rustfs
```

Verificar contenido
cat descargado.txt

4.6. Borrar objeto

```
aws s3 rm s3://test-bucket/archivo.txt \
  --endpoint-url http://localhost:9000 \
  --profile rustfs
```

5. Borrar bucket

Primero debe estar vacío:

```
aws s3 rb s3://test-bucket \
  --endpoint-url http://localhost:9000 \
  --profile rustfs
```

6. Detener servidor

En la terminal donde corre RustFS:

Control + C

CRUD completo!!