Relación entre EC2 (máquina virtual) y los almacenamientos:

1.- EBS (Elastic Block Store)

Es un disco en bloques, parecido a un disco duro físico.
Se adjunta (attach) directamente a una instancia EC2.
La EC2 lo ve como un disco local (/dev/xvda, /dev/sdf, etc.).

Se usa para:

Sistema operativo de la instancia.
Bases de datos.
Aplicaciones que necesitan baja latencia y lecturas/escrituras rápidas.

Es persistente, aunque apagues o elimines la instancia (si no borras el volumen).

2.- EFS (Elastic File System)

Es un sistema de archivos compartido en red (tipo NFS).
Una o varias EC2 lo pueden montar al mismo tiempo.
Se ve como una carpeta (/mnt/efs/miapp).
Se usa para:
Aplicaciones que necesitan compartir archivos entre varias instancias.
Servidores web en cluster que usan los mismos datos.
Almacenamiento dinámico que crece automáticamente.

3.- S3

No se adjunta como un disco.
La instancia accede mediante API/CLI/SDK para subir o bajar archivos.
Es almacenamiento de objetos, no un sistema de archivos tradicional.
Se usa para:
Almacenamiento masivo, copias de seguridad.
Distribución de contenido.
Archivos estáticos (logs, imágenes, vídeos).

Nota

EBS → Tu disco duro personal, conectado directamente al PC.
EFS → Una carpeta compartida en red (tipo servidor NAS) que varios PCs pueden usar al mismo tiempo.
S3 → Una gran bodega de archivos, a la que accedes por internet con llaves.

Una máquina virtual siempre necesita un disco EBS (para arrancar y guardar datos).
Puede además montar un EFS si necesita compartir archivos.
Y usar S3 como almacenamiento externo para objetos, aunque no se adjunta como disco.

Práctica 1. Desplegar una web en EC2
Objetivo:

Montar tu aplicación web en una instancia EC2.
Usar EBS como disco para el sistema operativo y la app.
Configurar el Security Group para acceso público (HTTP/HTTPS).
Conectarse a S3 para backups simples.
Beneficio:
Aprendes a desplegar aplicaciones en la nube.
Configuras seguridad básica y almacenamiento local.

Objetivo:
Montar tu aplicación web en una instancia EC2, usando EBS para el sistema y archivos locales, y configurar el Security Group para que sea accesible desde internet.

Pasos detallados
1.-Crear la instancia EC2

Ve a EC2 → Launch Instance.
Elige una AMI (Linux Ubuntu recomendado para web simple).
Selecciona un tipo de instancia (t3.micro sirve para pruebas gratuitas).
Configura el volumen EBS (ya se crea automáticamente para el sistema operativo).

2.- Configurar el Security Group (*nota)

HTTP (80) y HTTPS (443) desde cualquier lugar (0.0.0.0/0) para que la web sea accesible.
Permitir SSH (22) solo desde tu IP para administrar la instancia.
Mantener el puerto 22 abierto es suficiente para SSH, pero no para navegar tu web.

3.-Conectarte a la EC2

Descarga la clave PEM al crear la instancia.
Conéctate vía terminal (Linux/macOS) o Putty (Windows):

```
ssh -i "tu-clave.pem" ubuntu@IP-de-tu-EC2
```

4.-Instalar servidor web
Ejemplo con Apache en Ubuntu:

```
sudo apt update
sudo apt install apache2 -y
sudo systemctl start apache2
sudo systemctl enable apache2
```

-- Ese comando sirve para que Apache2 se inicie automáticamente cada vez que arranques tu instancia EC2 (Ubuntu/Linux)

5.-Subir tu aplicación web

Copia tus archivos HTML/CSS/JS al directorio del servidor web:

5.1.- sal de la máquina virtual

```
exit
```

5.2.- Usa scp para copiar el archivo a tu EC2

```
scp -i "tu-clave.pem" "ruta index.html" ubuntu@IP-de-tu-EC2
```

```
scp -r -i "tu-clave.pem" "C:\Users\Desktop\miweb" ubuntu@98.81.248.252
```

5.3.- Muevelo a var/www/html

```
sudo cp -r /home/ubuntu/miweb/* /var/www/html/
```

* → significa “todos los archivos y carpetas dentro de esta carpeta”.
sudo cp /home/ubuntu/index.html /var/www/html/
sudo → necesitas permisos de administrador para escribir en /var/www/html/.

cp → copia el archivo (puedes usar mv si quieres moverlo y eliminar el original).
/home/ubuntu/index.html → origen.
/var/www/html/ → destino (donde Apache sirve los archivos web).

5.4.- Verifica que esté en el destino

```
ls -l /var/www/html/
```

Deberías ver index.html listado ahí.

6.-Probar la web

Abre un navegador y escribe la IP pública de tu EC2:

```
http://<IP-de-tu-EC2>
```

Debes ver tu página web funcionando.

¿Por qué no funciona?

El grupo de seguridad que hemos hecho solo tiene abierto el puerto 22 es solo para SSH (conectarte a la EC2).

Apache sirve páginas web por defecto en el puerto 80 (HTTP) y 443 (HTTPS).

Si solo tienes abierto el 22, los navegadores no podrán acceder a tu web.

Vamos a modificar las reglas de nuestro grupo de seguridad. Ojo porque podríamos haber abierto estos puertos al crear el grupo de seguridad y habernos evitado este paso.

Abrir el puerto 80 en el Security Group

Ve a la consola de AWS → EC2 → Security Groups.
Selecciona el Security Group asociado a tu instancia.
En Inbound rules (reglas de entrada):
Haz click en Edit inbound rules.
Añade una regla:
Type: HTTP
Protocol: TCP
Port range: 80
Source: Anywhere (0.0.0.0/0) si quieres acceso público, o tu IP si solo quieres ac ceso desde tu máquina.

Guarda los cambios.

Probar la web con esta nueva regla de seguridad

Ahora abre en tu navegador:
`http://<IP-PÚBLICA-DE-TU-EC2>`
Deberías ver tu página cargada.

Notas adicionales

Si quieres HTTPS, tendrías que instalar un certificado (ej: con Let's Encrypt) y abrir también el puerto 443.

“En esta práctica hemos desplegado una página web en nuestra instancia EC2 y la hemos hecho accesible mediante HTTP. Más adelante veremos cómo registrar un dominio gratuito o de pago y configurarlo para usar HTTPS con certificados válidos, de manera que nuestra web sea segura y confiable para los usuarios.”

7.- Conexión a S3

Guardar backups en S3, asigna un rol IAM a tu EC2 con permisos de acceso al bucket.

Un backup = una copia de seguridad de tus datos. Te sirve para recuperar información si algo falla en tu EC2, EBS o aplicación.
En este caso, podemos hacer un backup de: Archivos de la aplicación (/var/www/html)

Nota:
En los laboratorios (como AWS Academy, etc.) los permisos están limitados para que los alumnos no toquen configuraciones críticas de IAM.
Lo que suele estar restringido

Crear usuarios IAM: no se permite.
Crear o modificar roles globales: tampoco.
Asignar políticas personalizadas: muchas veces está bloqueado.
Cambiar configuración de la cuenta raíz:siempre prohibido.

Lo que sí puedes hacer normalmente

Usar un rol prediseñado (ej. LabRole) que ya tiene permisos básicos sobre S3, EC2, DynamoDB, etc.
Asociar ese rol a tu EC2 en los laboratorios (a veces ya viene por defecto).
Trabajar con S3, EC2, EBS, EFS sin problemas, siempre dentro de lo que el rol permite.

¿Por qué lo hacen?

Para evitar que un alumno rompa la cuenta (IAM controla toda la seguridad de AWS).
Para que todos los estudiantes trabajen en condiciones iguales.
Porque los laboratorios son entornos compartidos y temporales, y los roles personalizados no son necesarios para aprender los fundamentos.

Solución práctica=>Usar un rol existente:
Tenemos un rol llamado LabRole.
Normalmente ese rol ya viene con permisos suficientes para acceder a S3.
En vez de crear un rol nuevo, lo que tienes que hacer es asignar ese rol a tu instancia EC2.
Cómo asignarlo en EC2:
Ve a EC2 → Instances → selecciona tu instancia.
En la pestaña Instance details, busca IAM Role.
Haz clic en Modify IAM Role.
Selecciona el rol existente (ej: LabRole).
Guarda

Con eso, tu instancia ya tendrá permisos de acceso a S3 sin necesidad de claves.

Pasos para asignar un rol IAM a la EC2 (nosotros lo tenemos limitado por laboratorio)

1.-Crear un rol IAM:

Ve a IAM → Roles → Create Role.
Selecciona AWS Service → EC2.
Adjunta la política de acceso a S3. Por ejemplo:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:PutObject",
        "s3:GetObject",
        "s3:ListBucket"
      ],
      "Resource": [
        "arn:aws:s3:::empresa-backups-logs",
        "arn:aws:s3:::empresa-backups-logs/*"
      ]
    }
  ]
}
```

2.-Asignar el rol a la EC2 (mismo que hemos visto en el apartado anterior, una vez creado el rol se asigna de la misma manera): LabRole

Al crear la instancia EC2, selecciona el rol en IAM Role.
Si la instancia ya existe, ve a Actions → Security → Modify IAM Role y asigna el rol.

Pasos:

2.1.-Instala el AWS CLI en la EC2 (si no está ya):

AWS CLI = Amazon Web Services Command Line Interface.

Es una herramienta de línea de comandos que te permite gestionar todos los servicios de AWS desde tu terminal, sin usar la consola web.
Básicamente, puedes hacer todo lo que harías en la interfaz gráfica de AWS usando comandos.

```
sudo apt update
sudo apt install awscli -y
```

Si el paquete no está disponible, entonces hay que descargarlo directamente de AWS:

Descarga el instalador oficial:

```
curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o "awscliv2.zip"
```

instala la herramienta necesaria para poder abrir archivos .zip en tu EC2.

```
sudo apt update
sudo apt install unzip -y
```

Descomprime el paquete:

```
unzip awscliv2.zip
```

Entrar en la carpeta e instalar

cd aws

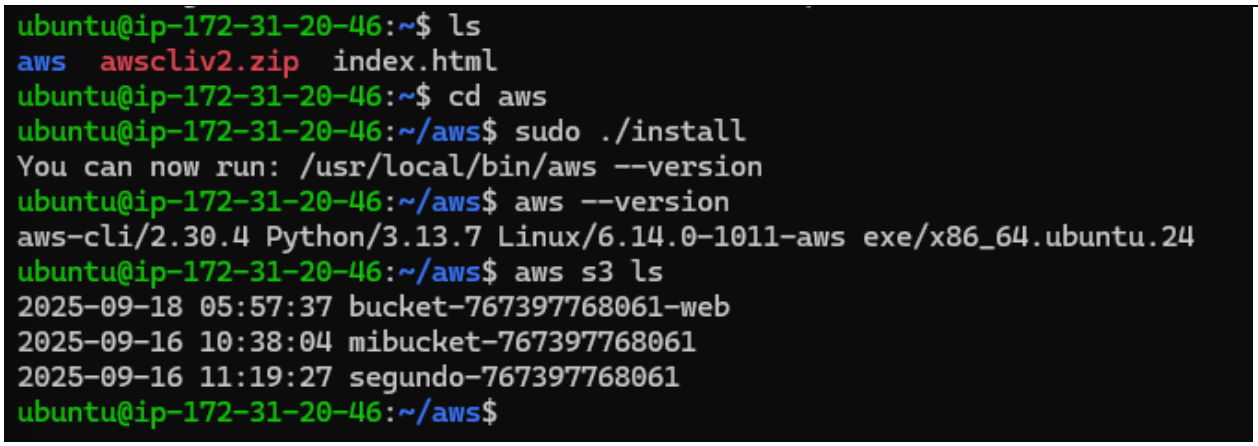
```
sudo ./install
```

Comprobar instalación

```
aws --version
```

Deberías ver algo como:
```
aws-cli/2.x.x Python/3.x Linux/x86_64
```

Si todo sale bien, ya podrás ejecutar y ver tus buckets:
```
aws s3 ls
```



2.2.- Configura AWS CLI sin claves, porque la EC2 obtiene credenciales automáticamente del rol:
```
aws s3 ls
```
Deberías ver tu bucket (empresa-backups-logs).

2.3.- Hacer un backup (por ejemplo, de tu web):
```
aws s3 cp /var/www/html/ s3://bucket-767397768061-web/mi-backup/ --recursive
```

Esto sube todos los archivos de /var/www/html al bucket en S3.

aws s3 cp → comando de copia en S3.
/var/www/html → la carpeta origen (donde está tu web).
s3://empresa-backups-logs/mi-backup/ → el destino en S3.
empresa-backups-logs = el nombre de tu bucket.

mi-backup/ = una carpeta dentro del bucket (puede ser cualquier nombre).
--recursive → indica que copie todos los archivos y subcarpetas dentro de /var/www/html.

2.3.- Desde consola

Ver todos los objetos en el bucket
```
aws s3 ls s3://bucket-767397768061-web/
```
Ver todos los objetos dentro de la carpeta mi-backup
```
aws s3 ls s3://bucket-767397768061-web/mi-backup/
```
Ver absolutamente todo (recursivo, con subcarpetas)
```
aws s3 ls s3://bucket-767397768061-web/ --recursive
```

Beneficios

Seguro: Los archivos quedan en S3, fuera de la EC2.
Persistente: Si la EC2 falla, tus datos están a salvo.
Automatizable: Puedes programar backups diarios con cron.

Resumen práctico

Backup = copia de tus archivos importantes.
Rol IAM = permite a EC2 acceder a S3 sin claves manuales.
Con AWS CLI y el rol, puedes subir archivos al bucket de forma segura.

Qué es cron

cron es un programa de Linux/Unix que permite ejecutar tareas automáticamente en hora rios o intervalos que tú determines.
Es como un temporizador o planificador de tareas.

Para qué sirve

Ejecutar scripts de backup automáticamente cada día, semana o mes.
Enviar correos automáticamente.
Limpiar archivos temporales.
Ejecutar cualquier comando o script sin tener que hacerlo manualmente.

Cómo funciona

Se configuran tareas (cron jobs) con una sintaxis que indica:

```
minuto hora día-del-mes mes día-de-la-semana comando
```

Ejemplo: hacer un backup a S3 todos los días a las 2:30 AM:

```
30 2 * * * aws s3 cp /var/www/html s3://empresa-backups-logs/mi-backup/ --recursive
```

30 2 * * * → significa 2:30 AM todos los días

Lo que sigue es el comando que cron ejecutará automáticamente.

8.- Resultado esperado

Una instancia EC2 corriendo tu aplicación web.

El Security Group permite acceso público a la web.

La aplicación puede conectarse a S3 mediante un rol IAM.

9.- Entrega.

Entregar solo la IP pública de la EC2, que permita acceder al index.html.
captura/s de pantalla o logs que confirmen el backup en S3.

Nota Security Group

Entender qué es un Grupo de Seguridad es clave antes de diseñar cualquier arquitectura en AWS.

Un Grupo de Seguridad (Security Group) en AWS es como un firewall virtual que controla el tráfico de red que entra y sale de una instancia EC2 (o de otros recursos como RDS, Load Balancers, etc.).

Características principales

1.-Funciona a nivel de instancia (no de toda la red)

Cada instancia EC2 puede tener uno o varios grupos de seguridad.

Los grupos definen qué tráfico se permite entrar o salir.

2.-Reglas de entrada (Inbound)

Controlan el tráfico que llega a la instancia.

Ejemplo: permitir solo conexiones HTTP (puerto 80) o SSH (puerto 22)

3.-Reglas de salida (Outbound)

Controlan el tráfico que sale desde la instancia hacia afuera.

Por defecto, AWS permite todo el tráfico de salida.

4.-Son stateful

¿Qué significa que un Security Group sea stateful?

Stateful = "con memoria del estado de la conexión".

Significa que si permites una conexión de entrada, la respuesta de salida se permite auto máticamente, y viceversa.

Ejemplo:

Tienes una regla de entrada (Inbound) que permite HTTP (puerto 80) desde cualquier lugar (0.0.0.0/0). => Un usuario accede a tu web.

La respuesta de salida (Outbound) de la instancia hacia ese usuario se permite automáticamente, aunque no haya una regla de salida específica para ese tráfico.

Si fuera stateless (sin memoria de estado), tendrías que crear reglas de salida para cada tipo de tráfico de respuesta, lo que sería muy tedioso.

Los Security Groups son stateful, lo que significa que no necesitas reglas duplicadas para que las respuestas de las conexiones permitidas salgan de tu instancia.

Por lo tanto stateful:

Si permites una conexión de entrada, la respuesta de salida está automáticamente permitida.

No necesitas escribir reglas duplicadas como en un firewall clásico.

5.- Se gestionan con permisos explícitos

Por defecto, todo el tráfico está bloqueado (excepto el outbound libre).

Tú defines qué se abre.