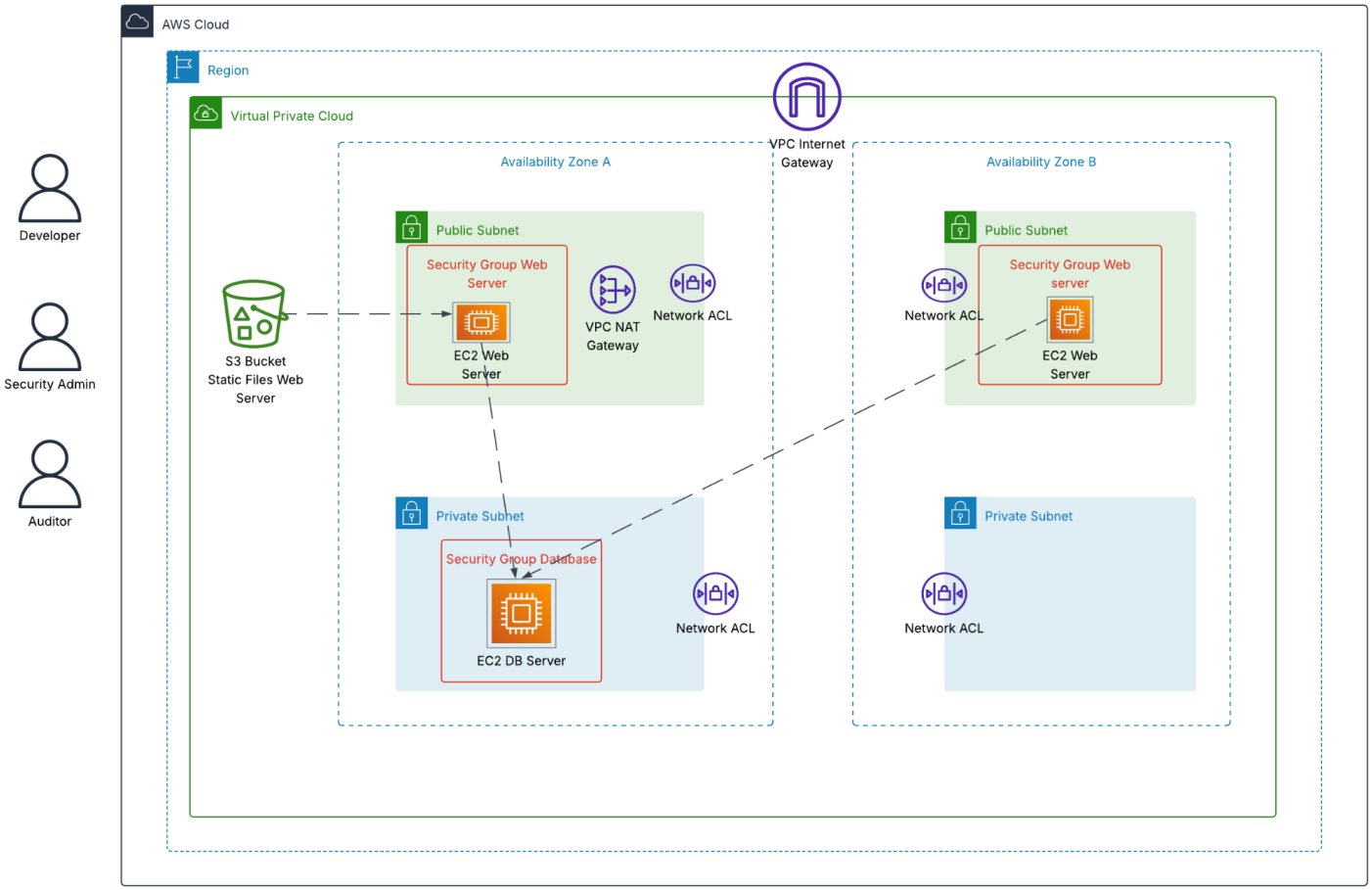
Ciberseguridad en la Nube

AWS Academy Learnerlab
Aplicación Web de Comercio Electrónico

Autor: Adrián Felipe Ibarra López

Objetivo: Diseñar, implementar y documentar una arquitectura segura en AWS para una aplicación web de comercio electrónico, aplicando los principios y prácticas aprendidas en el curso Cloud Security Foundations.



Fase 1. Arquitectura de Red Segura (VPC y Subredes)

Diseño y creación de la VPC

Para garantizar una arquitectura segura y escalable, se creó una VPC personalizada denominada SecureVPC, utilizando el rango CIDR 10.0.0.0/16. Este rango proporciona más de 65.000 direcciones IP privadas, lo que permite una amplia capacidad de expansión, en concordancia con las recomendaciones de la RFC 1918. Ver figura 1.

Para lograr alta disponibilidad y tolerancia a fallos, se distribuyeron subredes entre dos Zonas de Disponibilidad (us-east-1a y us-east-1b). Se crearon un total de cuatro subredes. Ver la siguiente tabla:

NOMBRE	*CIDR*	*ZONA DE DISPONIBILIDAD*	*TIPO*	*USO PRINCIPAL*
Public-Subnet-1	10.0.0.0/24	(us-east-1a)	Pública	Instancia EC2 Web 1
Public-Subnet-2	10.0.1.0/24	(us-east-1b)	Pública	Instancia EC2 Web 2
Private-Subnet-1	10.0.10.0/24	(us-east-1a)	Privada	Base de datos 1
Private-Subnet-2	10.0.11.0/24	(us-east-1b)	Privada	Base de datos 2 / alta disponibilidad

Tabla 1. Subredes.

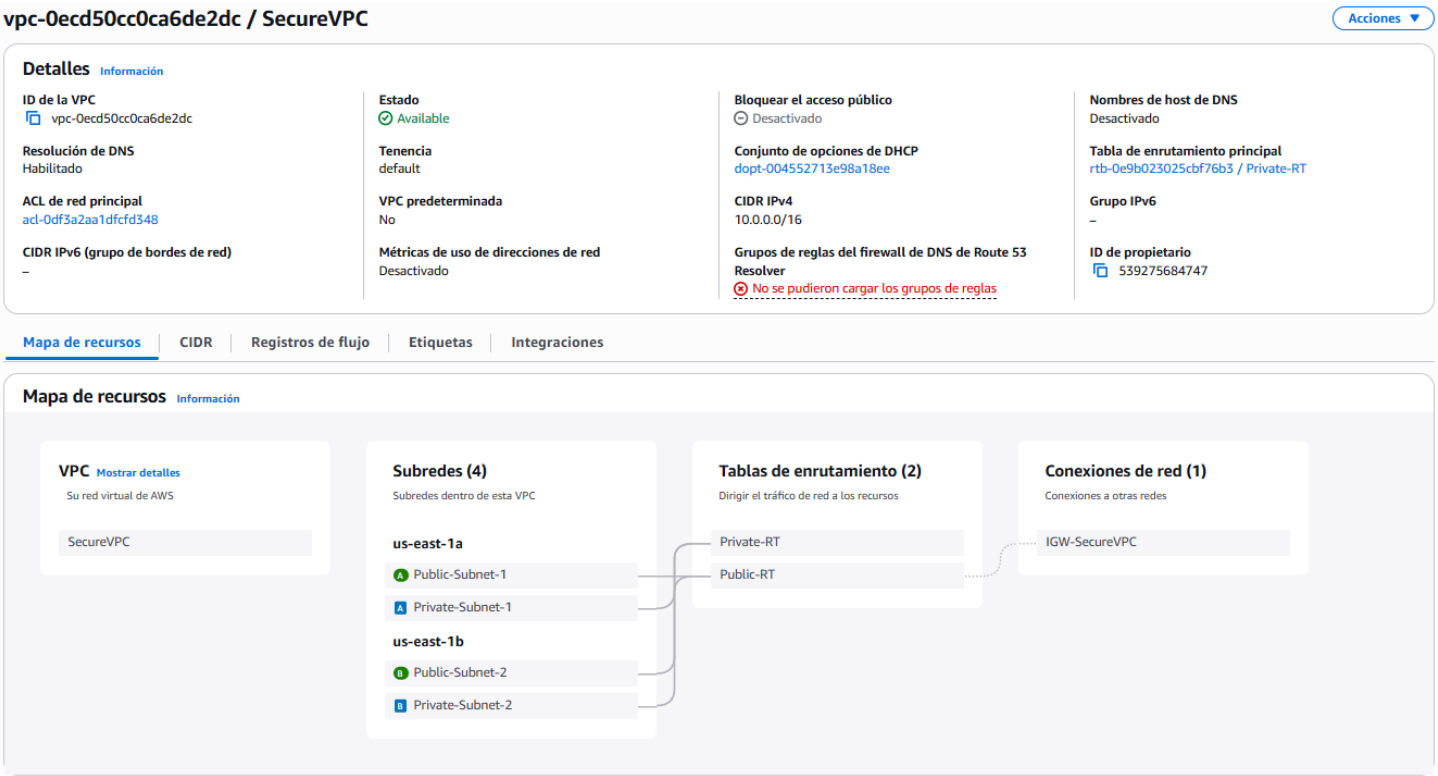


Figura 1. VPC - SecureVPC.

Este diseño garantiza un entorno organizado, seguro y preparado para el despliegue de una arquitectura web de alta disponibilidad y cumplimiento de las mejores prácticas de ciberseguridad en la nube.

Configuración del acceso a Internet

Se creó un Internet Gateway denominado IGW-SecureVPC, el cual fue adjuntado a la VPC. Para permitir que las subredes públicas tengan acceso a Internet:

1. Se creó una tabla de ruteo personalizada llamada Public-RT.
2. Se añadió una ruta que direcciona todo el tráfico (0.0.0.0/0) hacia el Internet Gateway.
3. Esta tabla fue asociada explícitamente a las subredes públicas (Public-Subnet-1 y Public-Subnet-2).

Figura 2. Internet Gateway - IGW-SecureVPC.

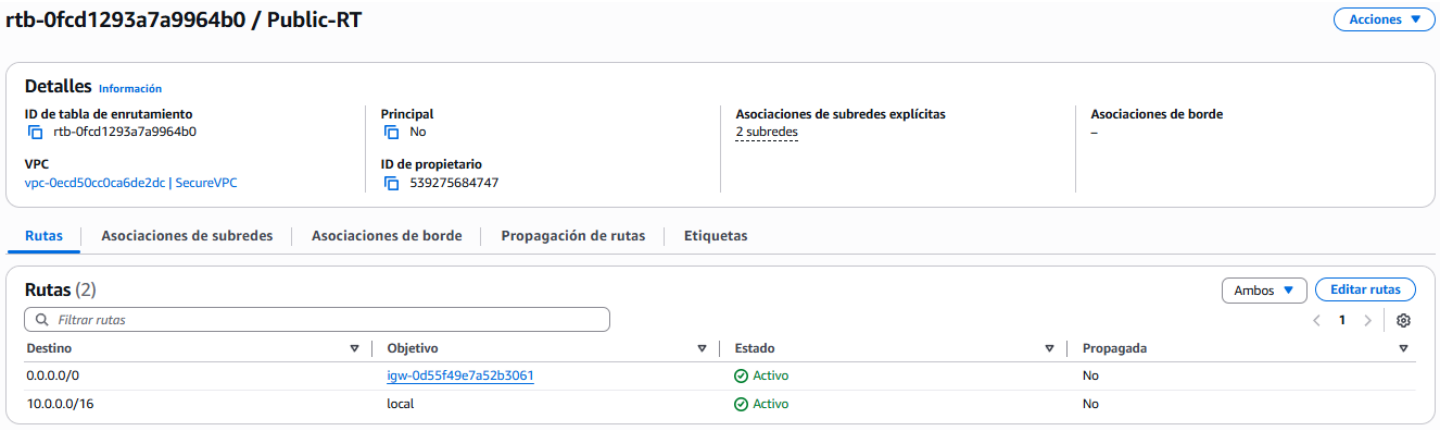


Figura 3. Tabla de enrutamiento – Public-RT.

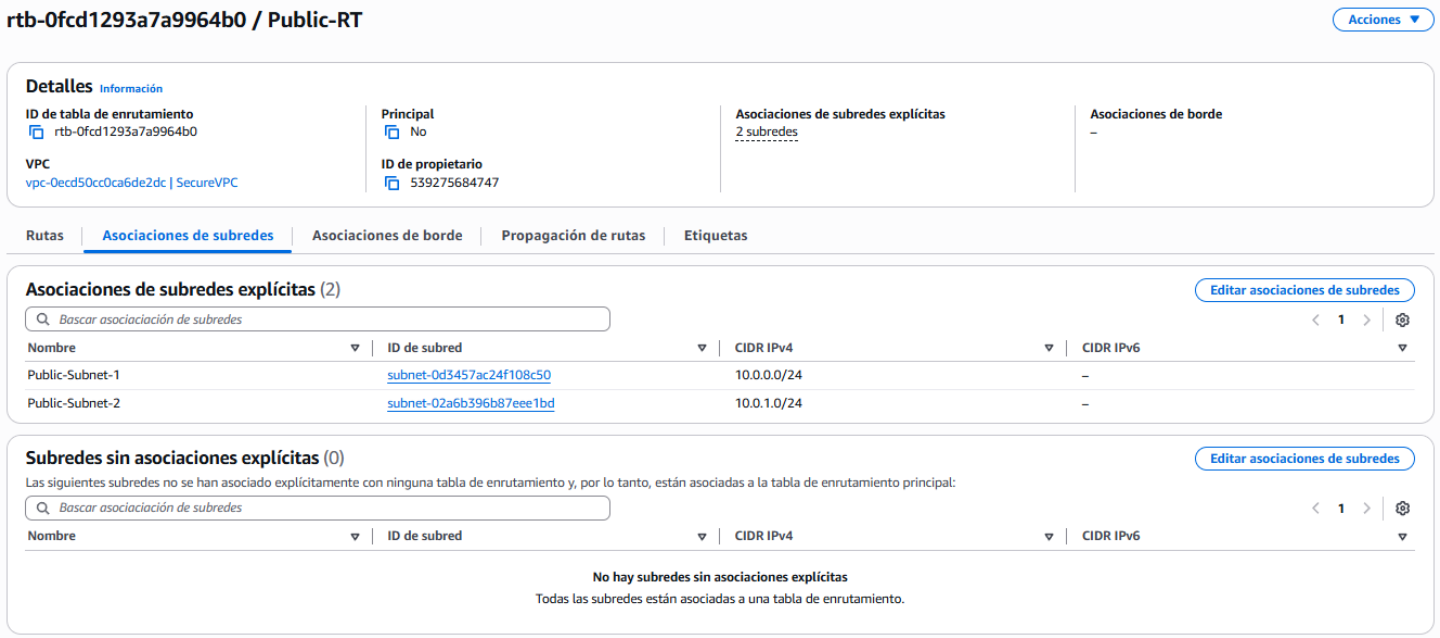


Figura 4. Tabla de enrutamiento – Public-RT.

Las subredes privadas no se asociaron a esta tabla, utilizando en su lugar la tabla de ruteo principal que no contiene rutas hacia Internet, garantizando así que los recursos en dichas subredes no sean accesibles desde el exterior, cumpliendo con los principios de segmentación y defensa en profundidad.

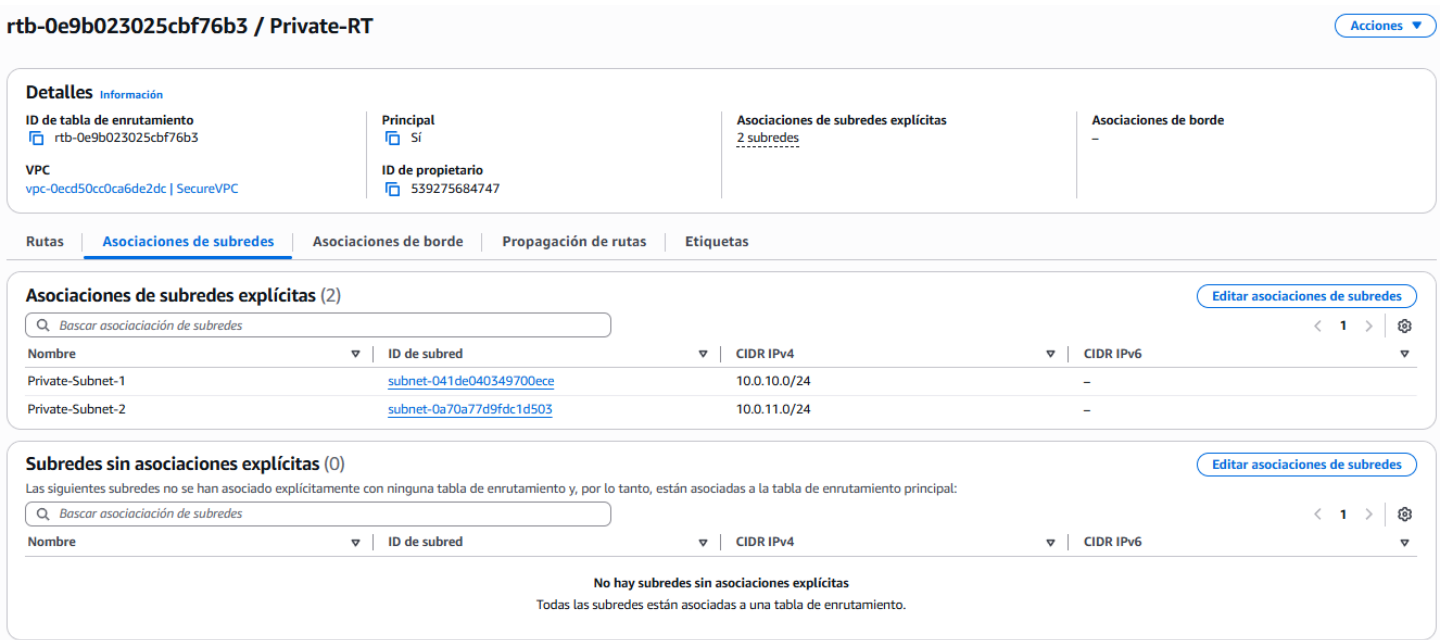


Figura 5. Tabla de enrutamiento – Private-RT.

Fase 2: Recursos de Cómputo (EC2)

Despliegue de instancias públicas para la aplicación web

Se implementaron dos instancias EC2 en las subredes públicas de la VPC SecureVPC, una en cada zona de disponibilidad (us-east-1a y us-east-1b), con el objetivo de alojar una aplicación web simple usando Apache. Ambas instancias fueron configuradas con el siguiente esquema:

- AMI: Amazon Linux 2023
- Tipo: t2.micro
- Nombre: Web-Server-1 y Web-Server-2
- Acceso público: habilitado con IP pública.
- Grupo de seguridad (SG-Web):
  - Permitir tráfico HTTP (puerto 80) desde cualquier IP (0.0.0.0/0)
  - Permitir tráfico SSH (puerto 22) desde IP específica para administración segura.

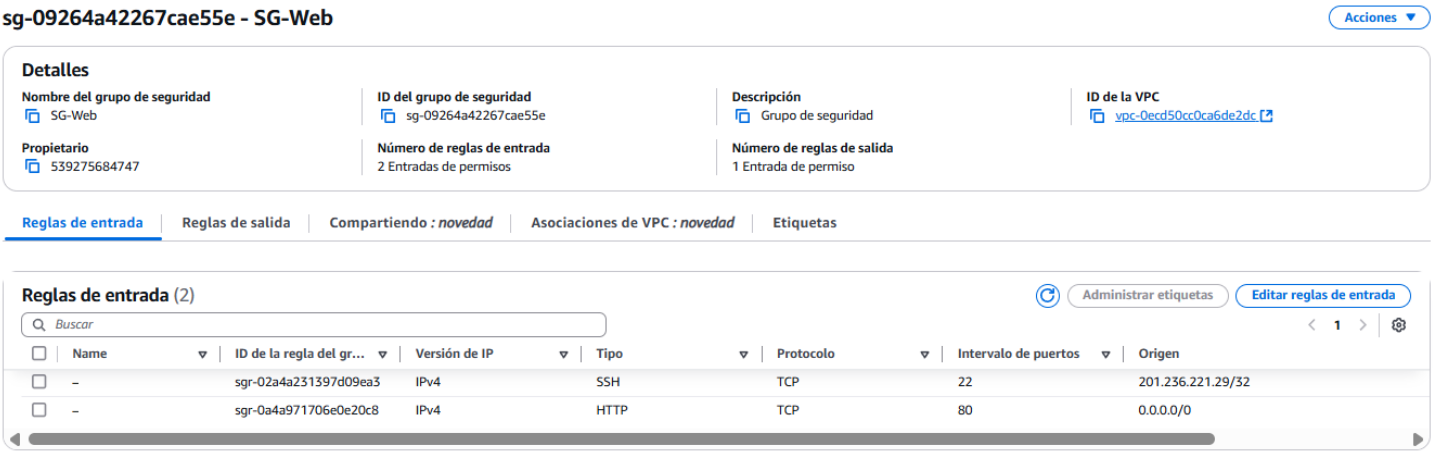


Figura 6. Grupo de seguridad SG-Web

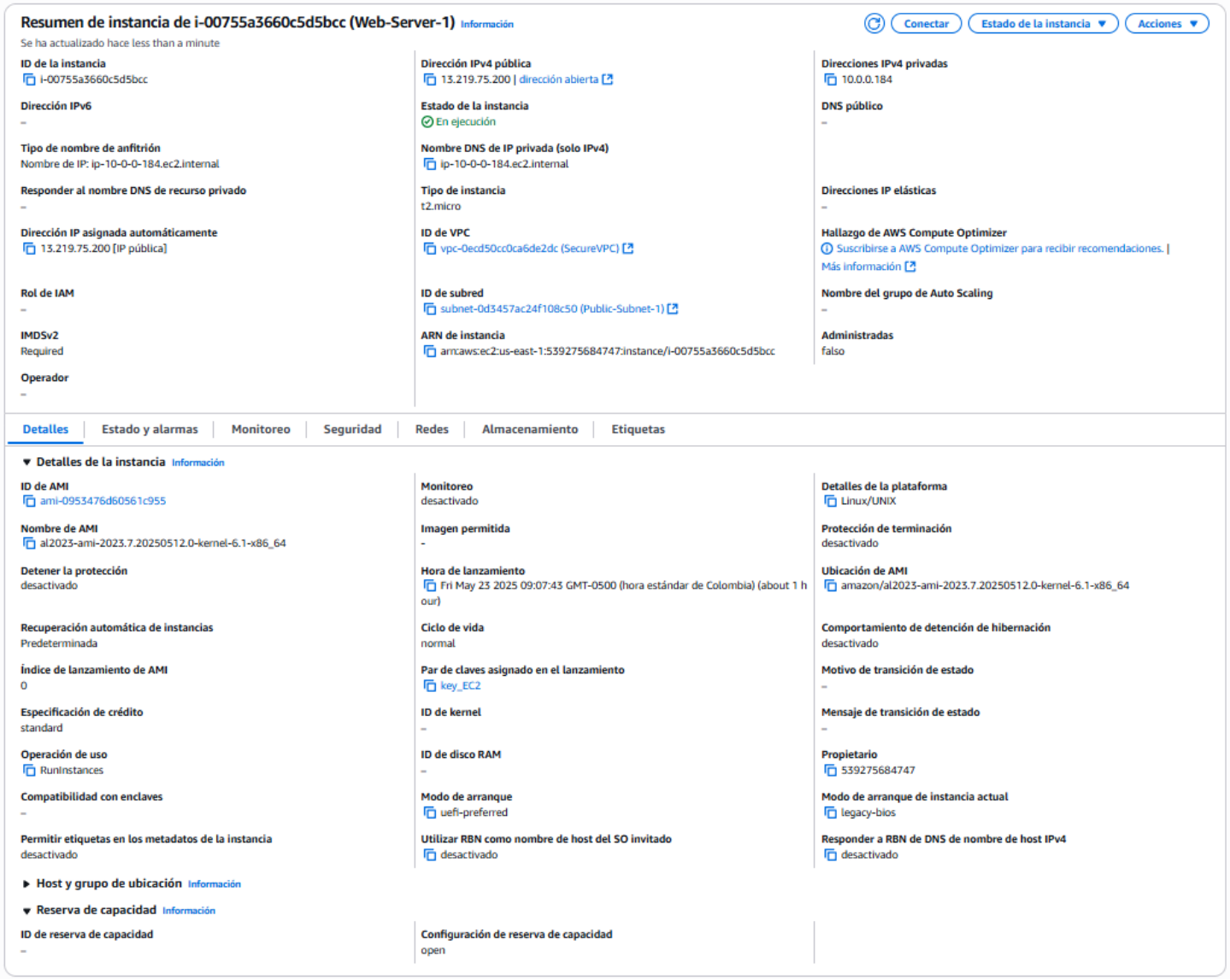


Figura 7. EC2 Web-Server-1

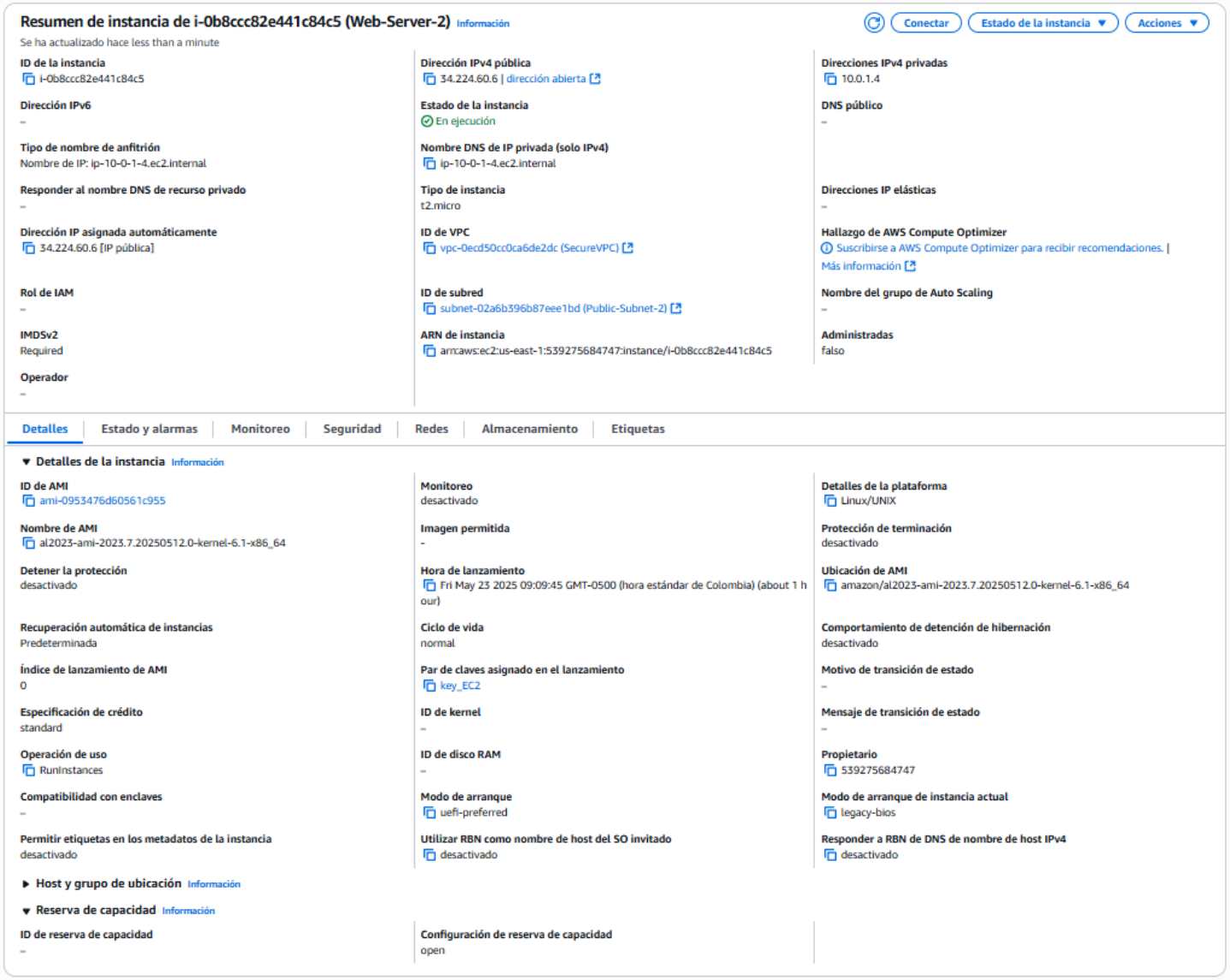


Figura 8. EC2 Web-Server-2

Una vez creadas, se instalaron los paquetes Apache y se configuraron páginas HTML de prueba para validar la conectividad externa.

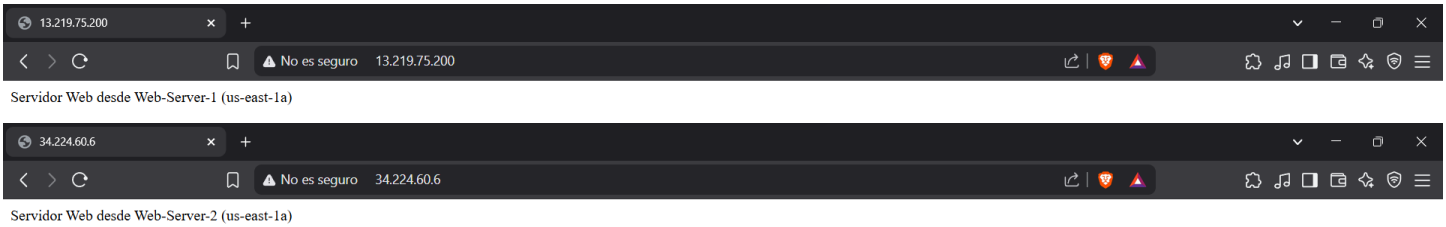


Figura 9. Servidor web de las instancias Web-Server-1 y Web-Server-2

Despliegue de instancia privada para base de datos

Se creó una tercera instancia EC2 en la subred privada Private-Subnet-1, sin dirección IP pública, destinada exclusivamente a alojar el servicio de base de datos MariaDB.

- Nombre: DB-Server
- AMI: Amazon Linux 2023
- Grupo de seguridad (SG-DB):

- Permitir conexiones al puerto 3306 únicamente desde el grupo de seguridad SG-Web (tráfico interno desde las instancias web).
- Permitir SSH únicamente desde las instancias públicas (para administración desde Web-Server-1 como jump host).

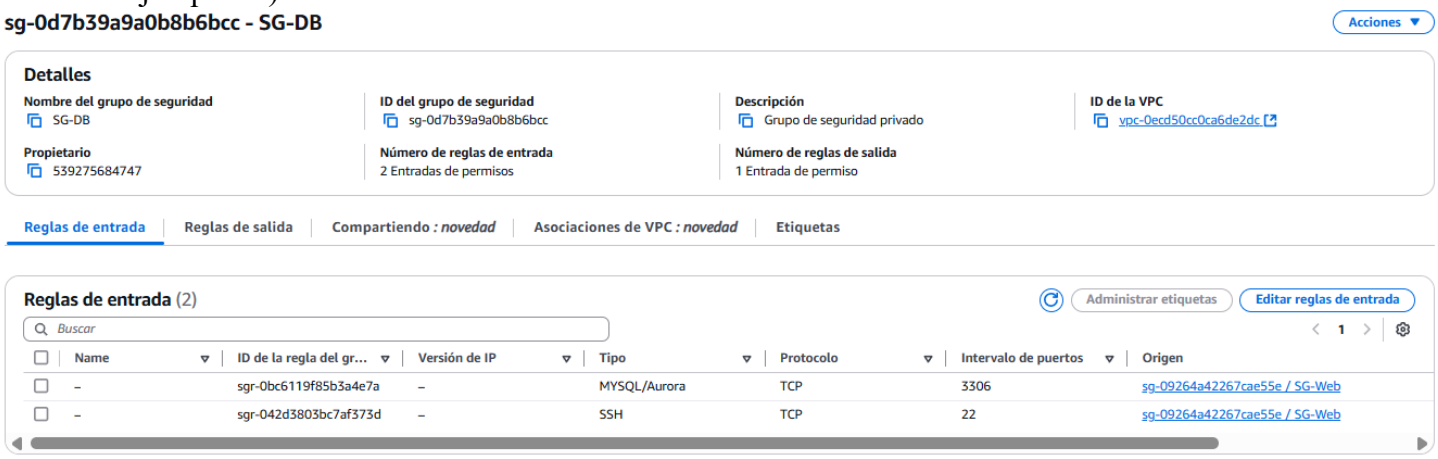


Figura 10. Grupo de seguridad SG-DB

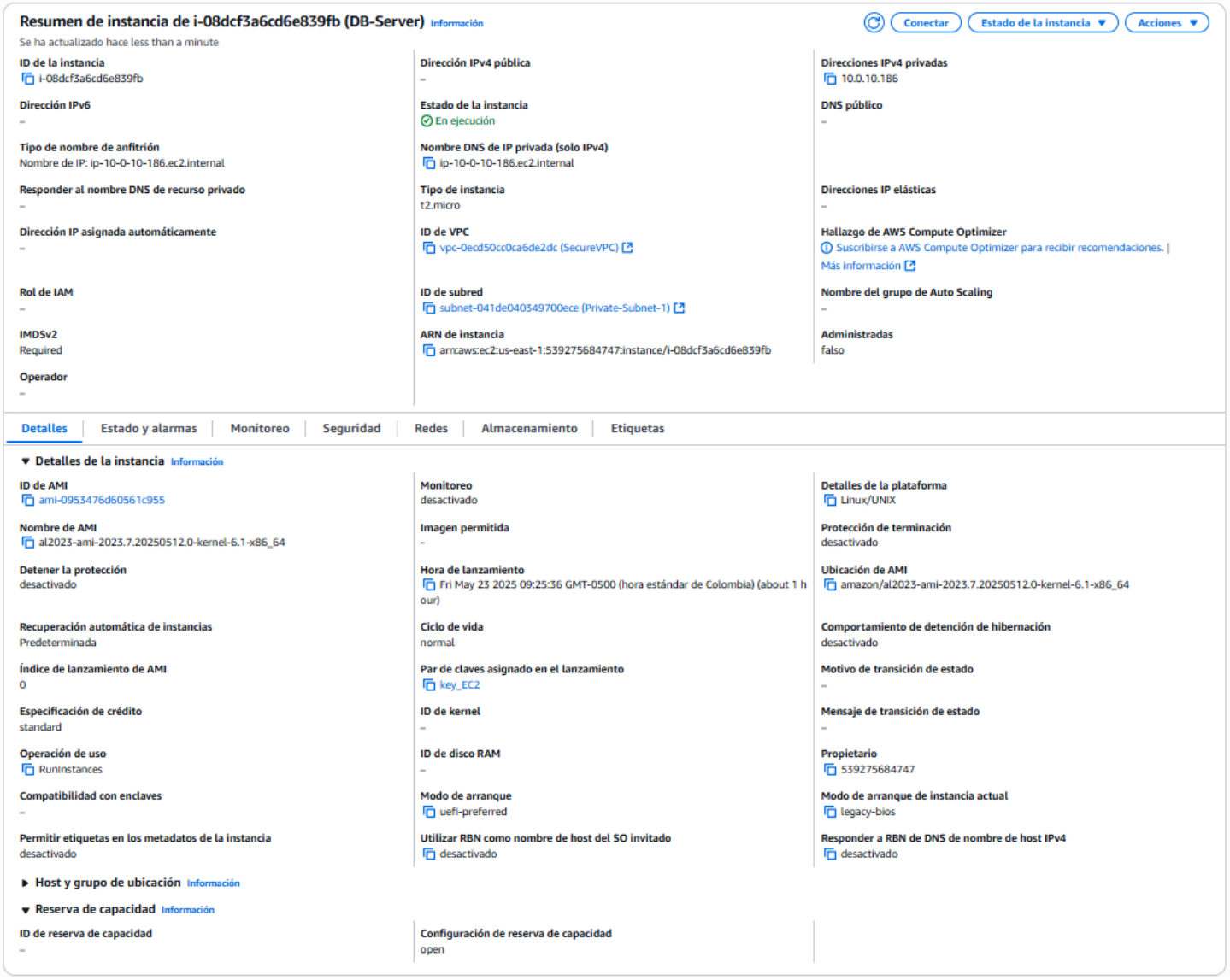


Figura 11. EC2 DB-Server

Dado que la instancia se encuentra en una subred privada, se configuró un NAT Gateway en la subred pública Public-Subnet-1, asociado a una IP elástica, y se editó la tabla de ruteo privada para permitir tráfico de salida hacia Internet exclusivamente mediante dicho NAT.

Figura 12. IP elástica

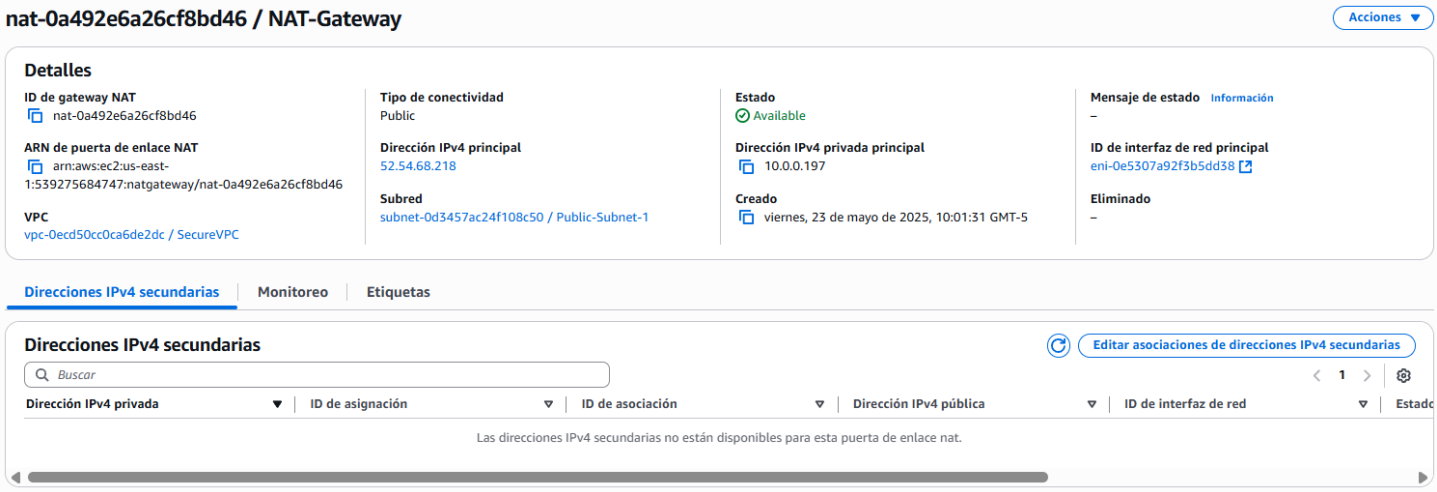


Figura 13. NAT-Gateway

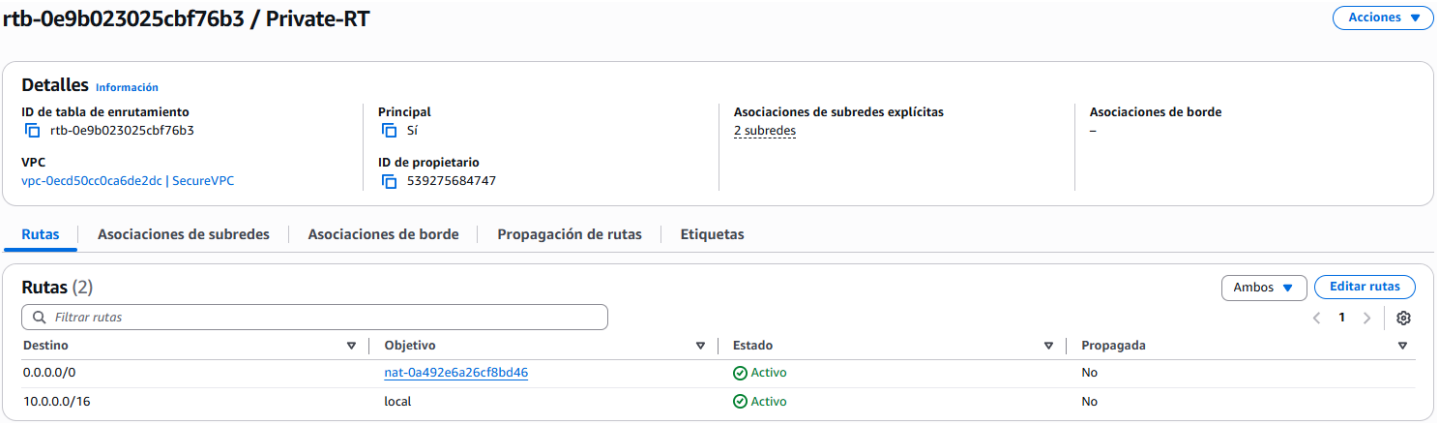


Figura 14. Tabla de enrutamiento Private-RT

Esto permitió instalar correctamente MariaDB 10.5 mediante dnf y habilitar su servicio con los siguientes comandos:

- sudo systemctl start mariadb
- sudo systemctl enable mariadb

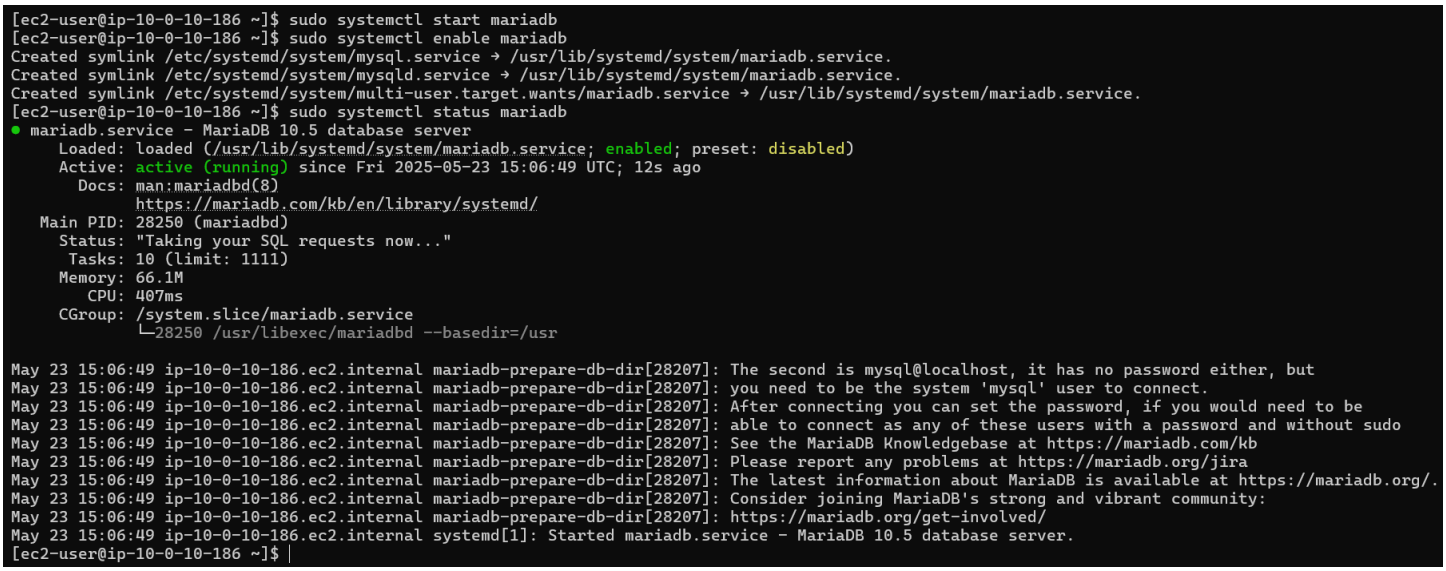


Figura 15. Instalación de MariaDB en la base de datos

Luego, se procedió a crear una base de datos, con un usuario remoto y se configuró el archivo /etc/my.cnf para permitir conexiones remotas desde las instancias web:

- Usuario: user
- Contraseña: student12

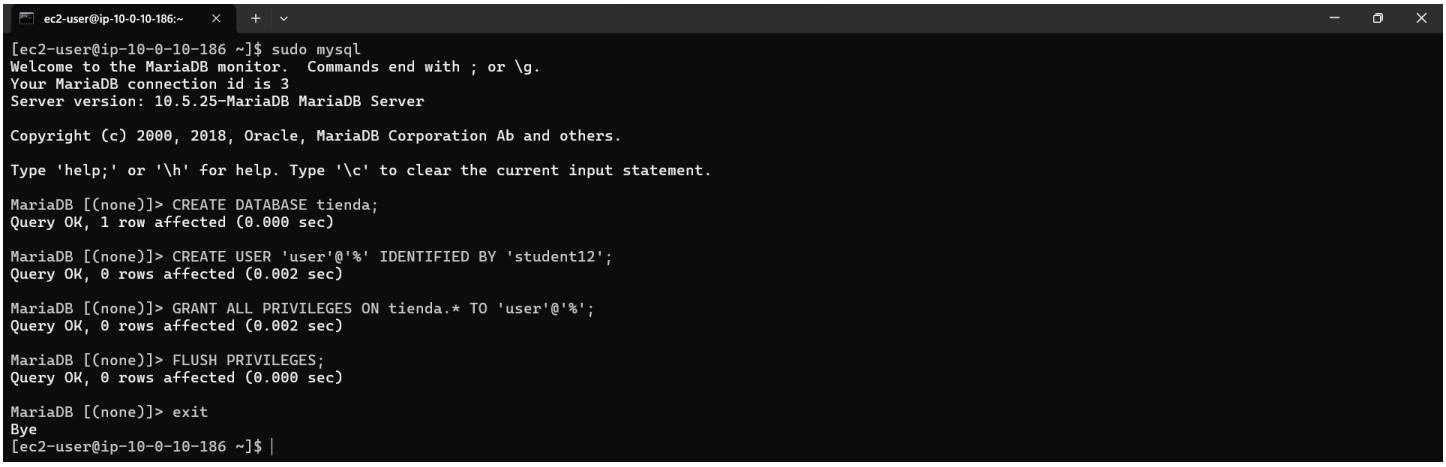


Figura 16. Base de datos.

Con estas acciones, se completó exitosamente la arquitectura de cómputo segura en AWS, con separación lógica entre los servidores web públicos y el servidor de base de datos privado, respetando principios de mínima exposición y segmentación de red.

Para probar la conexión a la base de datos, se ingresa a instancia Web-Server-1 y mediante mysql se conecta a la instancia de la base de datos con la IP privada, de la siguiente forma:

```
[ec2-user@ip-10-0-0-184 ~]$ mysql -h 10.0.10.186 -u user -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 4
Server version: 10.5.25-MariaDB MariaDB Server

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| tienda             |
+--------------------+
2 rows in set (0.001 sec)
```

Figura 17. Conexión a la base de datos desde Web-Server-1

Fase 3. Seguridad de Red

En esta fase se aplicaron mecanismos de control de tráfico a nivel de red para proteger los recursos desplegados en la VPC SecureVPC, cumpliendo con principios de defensa en profundidad, mínimo privilegio y segmentación lógica. Se utilizaron grupos de seguridad (SG) y listas de control de acceso a la red (NACLs), de forma complementaria.

Grupos de seguridad

Los grupos de seguridad fueron diseñados para restringir el acceso de manera específica a cada tipo de instancia:

- SG-Web (Instancias públicas):
  - Permitir acceso HTTP (puerto 80) desde cualquier origen (0.0.0.0/0).
  - Permitir acceso SSH (puerto 22) exclusivamente desde la IP pública del administrador.
  - No permite acceso a MySQL ni a otros puertos.

- SG-DB (Instancia privada):
  - Permitir acceso MySQL (puerto 3306) solo desde instancias que usen SG-Web.
  - Permitir SSH (puerto 22) solo desde SG-Web, para administración por salto.
  - No permite accesos desde el exterior.

Esta segmentación garantiza que la base de datos solo es accesible desde la capa de aplicación, y nunca directamente desde Internet.

Listas de control de acceso ACL

Se crearon NACLs personalizadas para las subredes públicas y privadas, reforzando el control de tráfico.

- ACL Pública (Public-ACL):
  Aplicada a Public-Subnet-1 y Public-Subnet-2. Las reglas permiten:
  - Entrada:
    1. HTTP (80) desde 0.0.0.0/0
    2. SSH (22) solo desde la IP pública del administrador
    3. Rango efímero (1024–65535) para retorno de tráfico
  - Salida:
    1. Permitir puertos efímeros, HTTP y SSH hacia 0.0.0.0/0
  - Denegación por defecto para todo tráfico no autorizado.

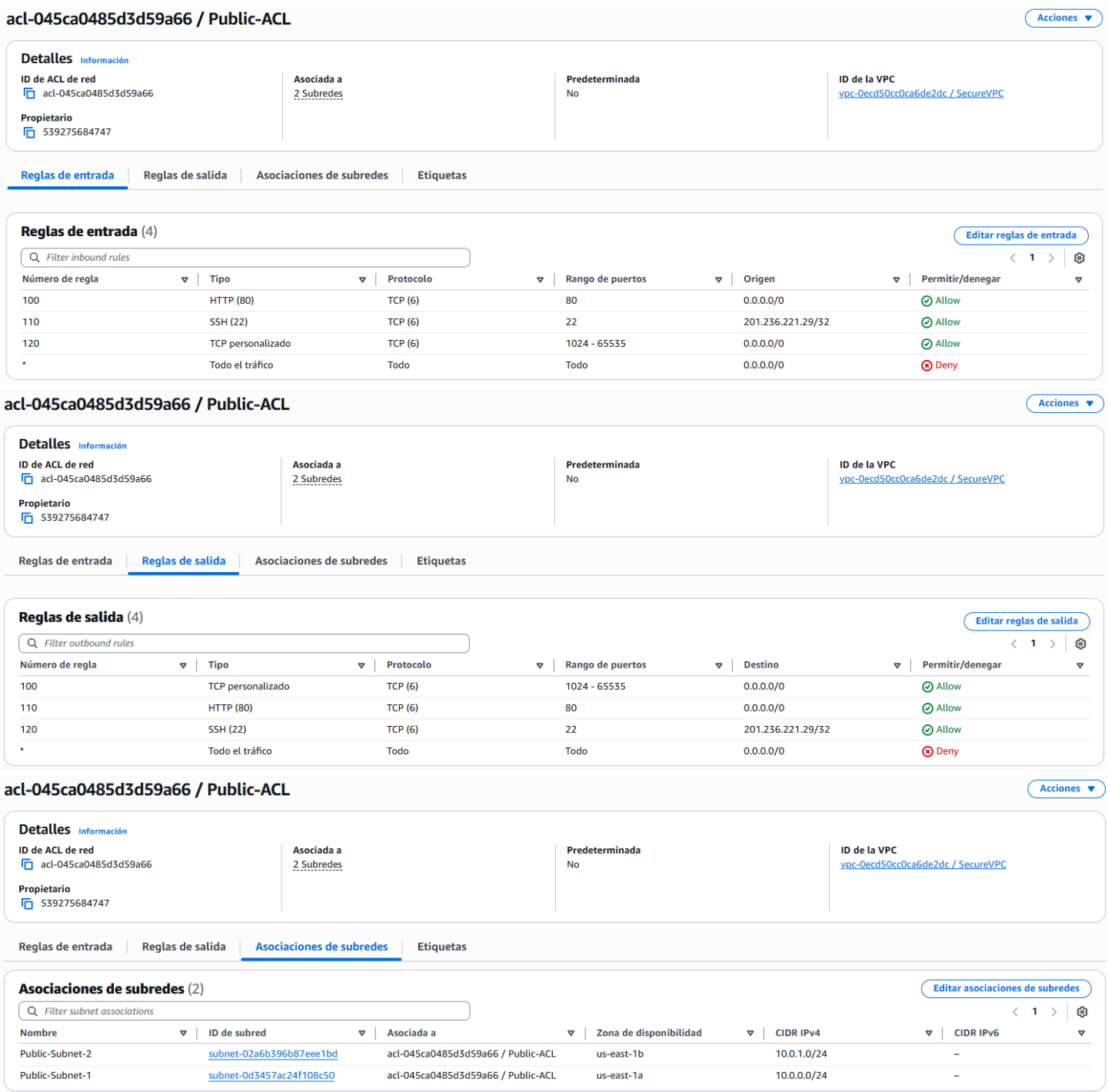


Figura 18. Configuración de ACL Public-ACL

- ACL Privada (Private-ACL):
  Aplicada a Private-Subnet-1 y Private-Subnet-2. Las reglas permiten:
  - Entrada: MySQL (3306), SSH (22) y puertos efímeros desde la red interna 10.0.0.0/16
  - Salida: Puertos efímeros, MySQL y SSH hacia la red interna y a través del NAT Gateway
  - Denegación explícita para todo lo no definido.

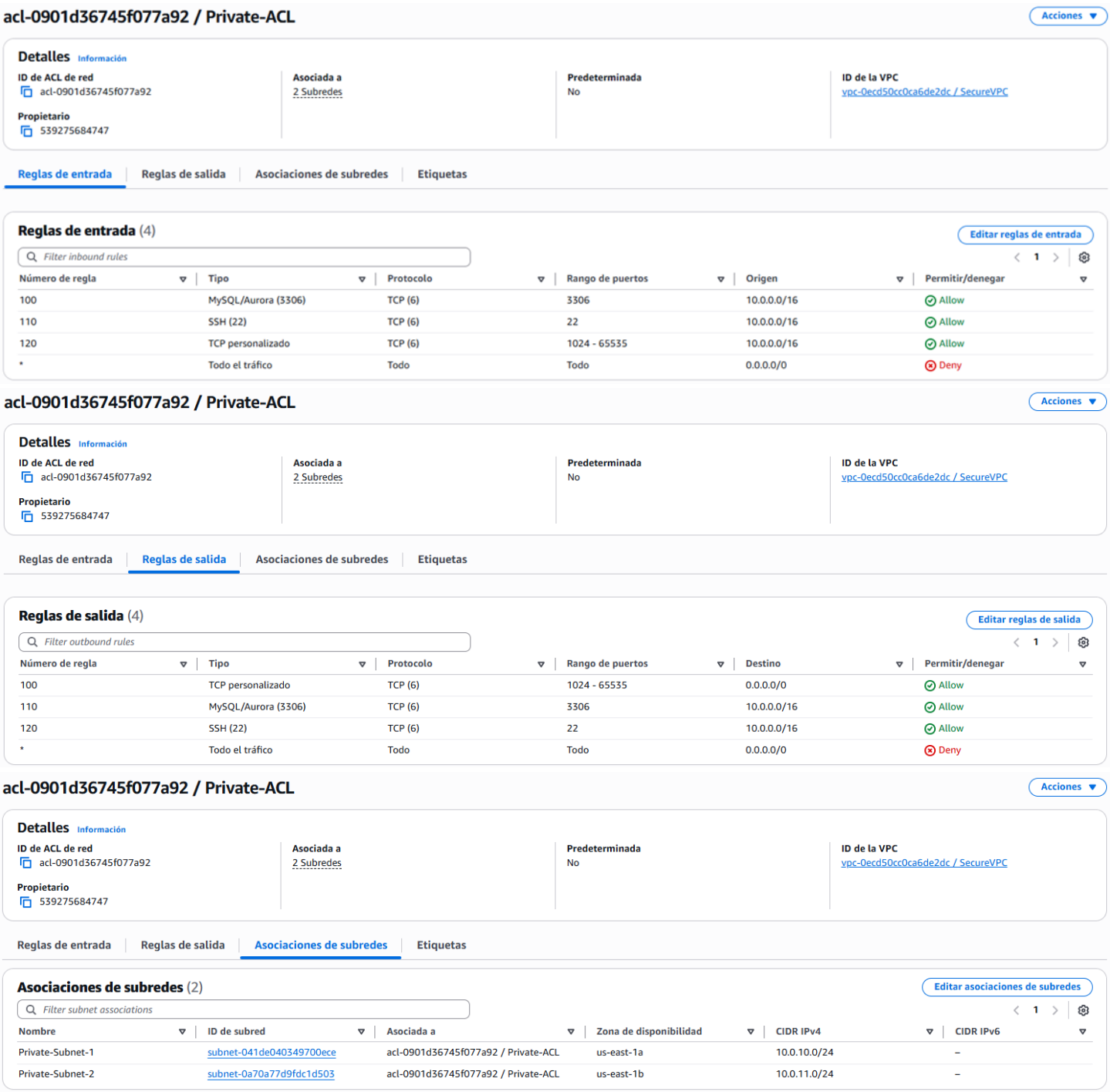


Figura 19. Configuración de ACL Private-ACL

Esta configuración asegura que el tráfico entre subredes está restringido y auditado, mientras que el acceso público a las instancias está controlado con reglas precisas.

Fase 4: Gestión de Identidades (IAM)

Durante el desarrollo de esta fase, se intentó crear usuarios IAM con diferentes niveles de acceso para aplicar control de identidades siguiendo las buenas prácticas de seguridad. Sin embargo, el entorno proporcionado por AWS Academy Learner Lab restringe ciertas operaciones administrativas, como iam:CreateUser, impidiendo completar esta etapa.

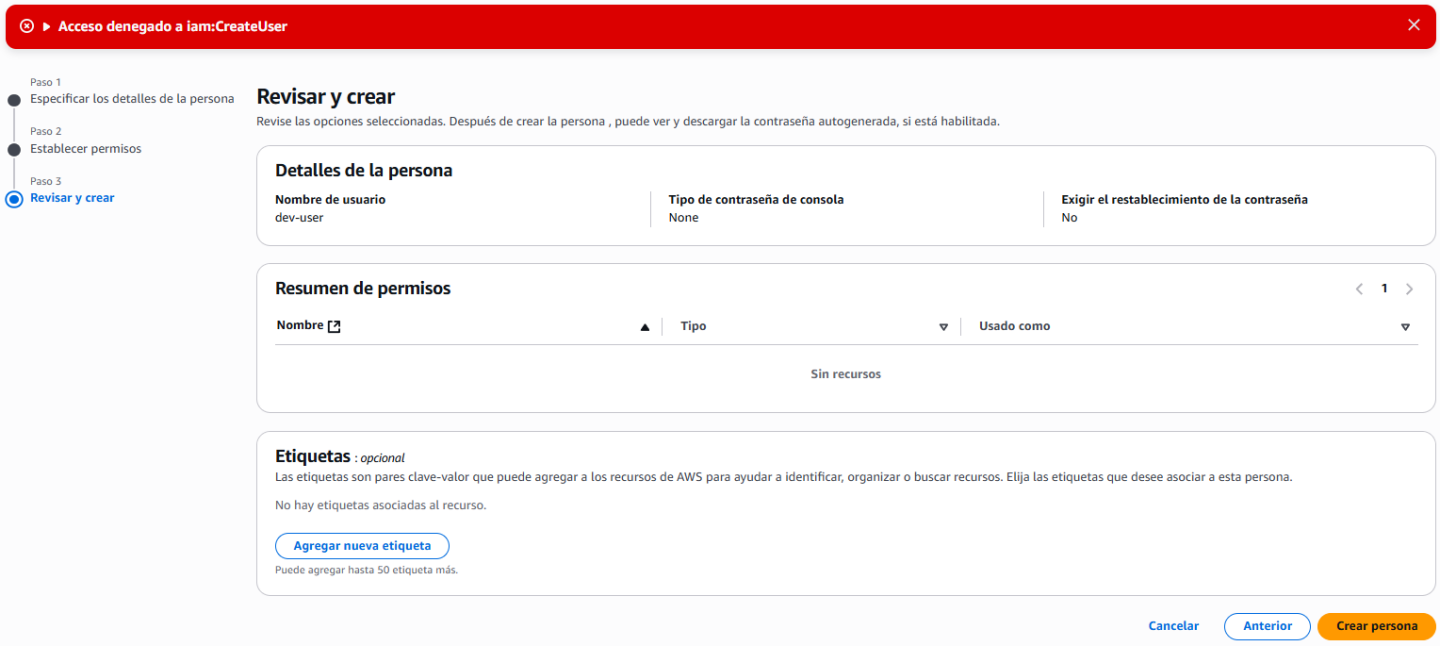


Figura 20. Error al crear usuarios en IAM

Fase 5. Almacenamiento Seguro (S3)

Con el objetivo de aplicar buenas prácticas de seguridad y almacenamiento en la nube, se configuraron dos buckets en Amazon S3:

1. webapp-static-files-v1: destinado a almacenar archivos estáticos de la aplicación web, como el archivo index.html.

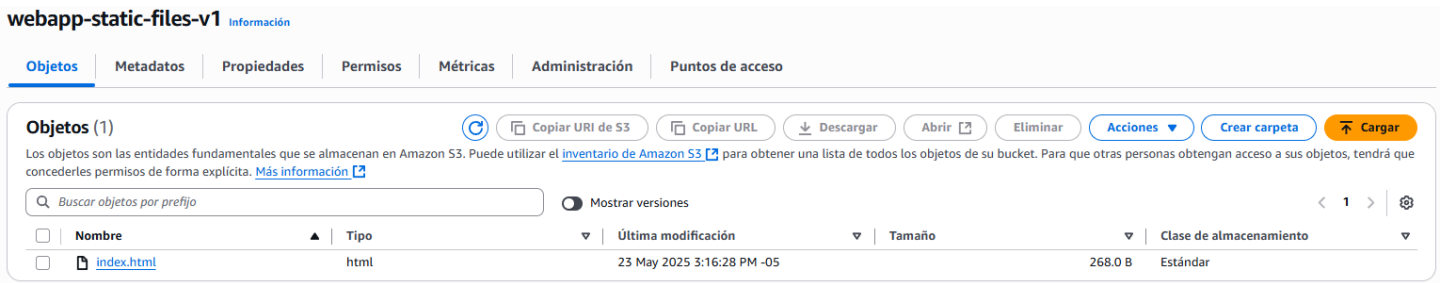


Figura 21. Bucket S3 webapp-static-files-v1

2. webapp-audit-logs-v2: utilizado como destino para almacenar los logs de acceso generados por el bucket anterior, activando el registro del servidor desde sus propiedades.

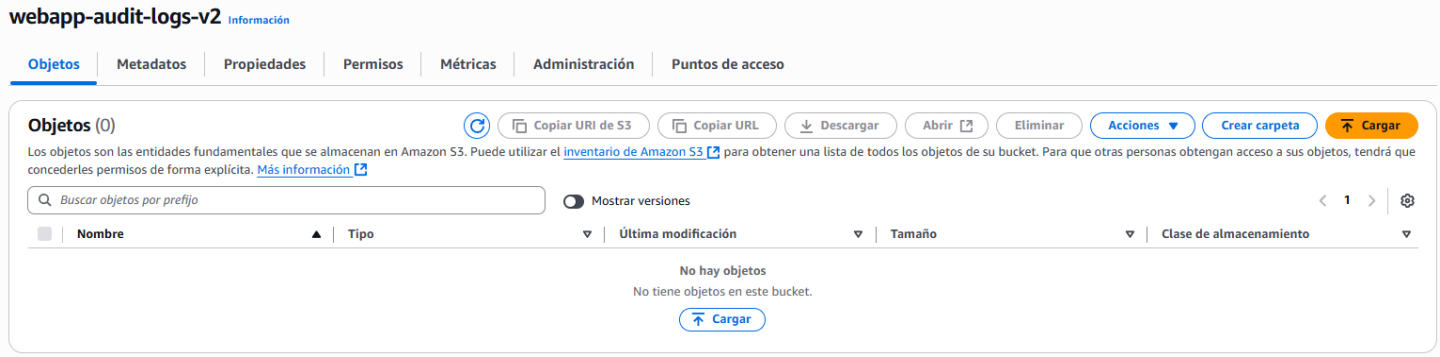


Figura 21. Bucket S3 webapp-audit-logs-v2

Ambos buckets fueron configurados con:

- Versionado habilitado, permitiendo mantener un historial de cambios de los objetos.
- Cifrado del lado del servidor (SSE-S3), activado por defecto para proteger los datos en reposo.
- Bloqueo de acceso público desactivado únicamente en el bucket de archivos estáticos, con intención de permitir el acceso público al archivo index.html.

Se intentó aplicar una política de bucket pública para habilitar el acceso anónimo a los archivos del bucket webapp-static-files-v1, con la siguiente política:

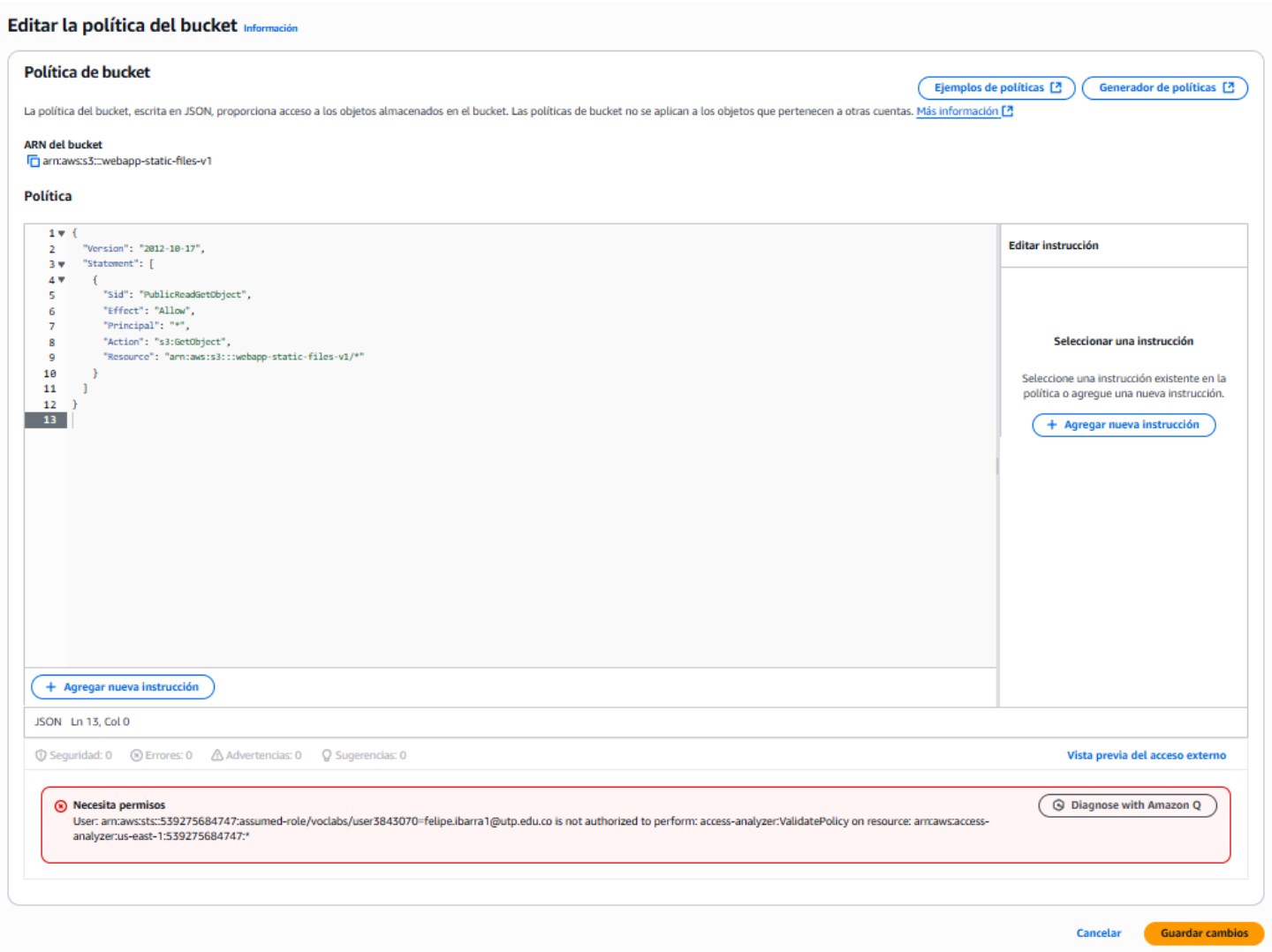


Figura 22. Error access-analyzer:ValidatePolicy AccessDenied

Sin embargo, el entorno AWS Academy Learner Lab restringe ciertas acciones administrativas, y se recibió un error al intentar aplicar esta política: access-analyzer:ValidatePolicy AccessDenied, lo cual impidió que los objetos fueran accesibles públicamente como parte de un sitio web estático.

A pesar de esto, se logró:

- Subir correctamente el archivo index.html.
- Confirmar que el archivo fue versionado y cifrado automáticamente.
- Redirigir correctamente los logs del bucket estático hacia el bucket de auditoría (webapp-audit-logs-v2), cumpliendo con los principios de trazabilidad y supervisión.

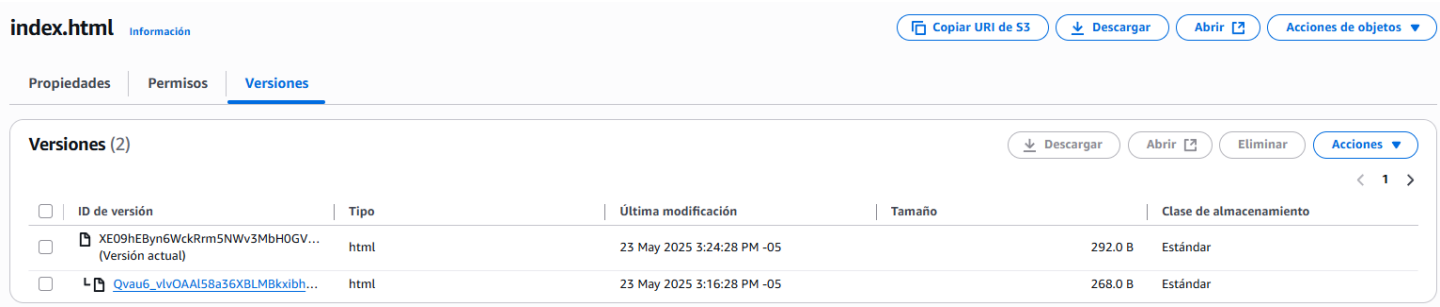


Figura 23. Control de versiones de webapp-static-files-v1

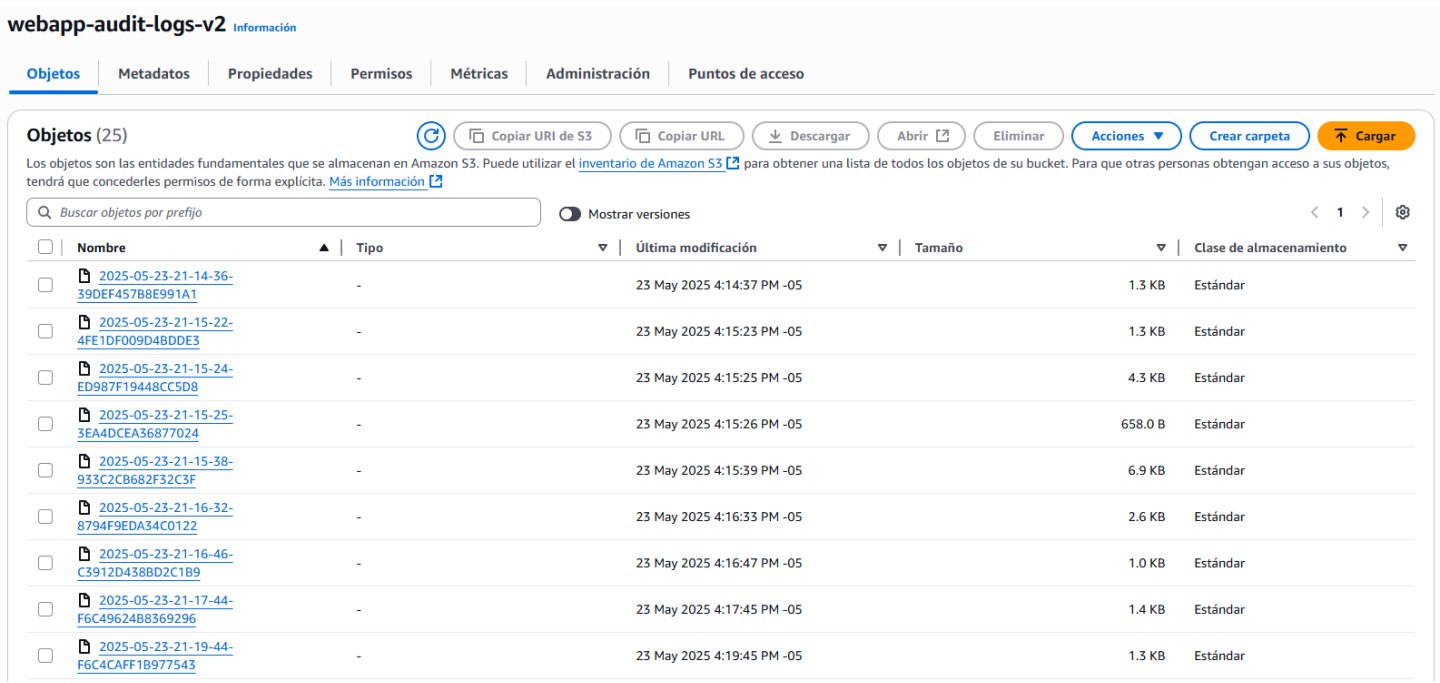


Figura 24. Registro de logs

Fase 6. Diagrama de Arquitectura

La siguiente figura representa la arquitectura implementada en AWS durante el desarrollo del laboratorio. Esta solución fue diseñada siguiendo las buenas prácticas de seguridad, alta disponibilidad y segmentación por capas.

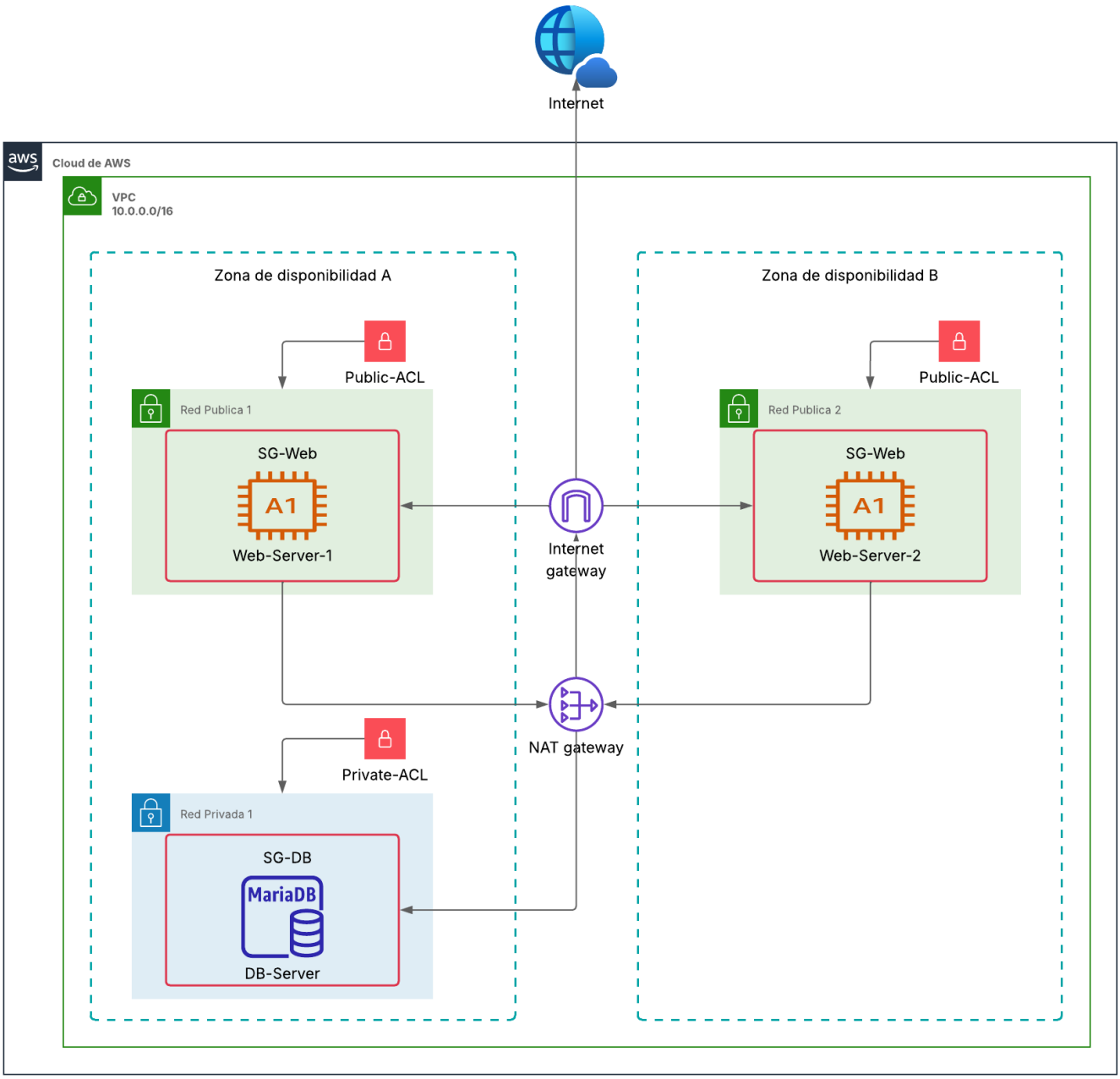


Figura 25. Diagrama de arquitectura

Pruebas de conectividad y acceso

Conexión a la Base de Datos Privada (DB-Server)

Dado que la instancia DB-Server se encuentra en una subred privada, sin dirección IP pública, no es posible conectarse directamente a ella desde Internet. Por seguridad y siguiendo buenas prácticas de arquitectura en AWS, se utilizó un modelo de salto (jump host) para acceder a la base de datos desde una instancia pública (Web-Server-1).

Desde la máquina local (o mediante EC2 Instance Connect), se conecta primero a la instancia Web-Server-1 que está en la subred pública:

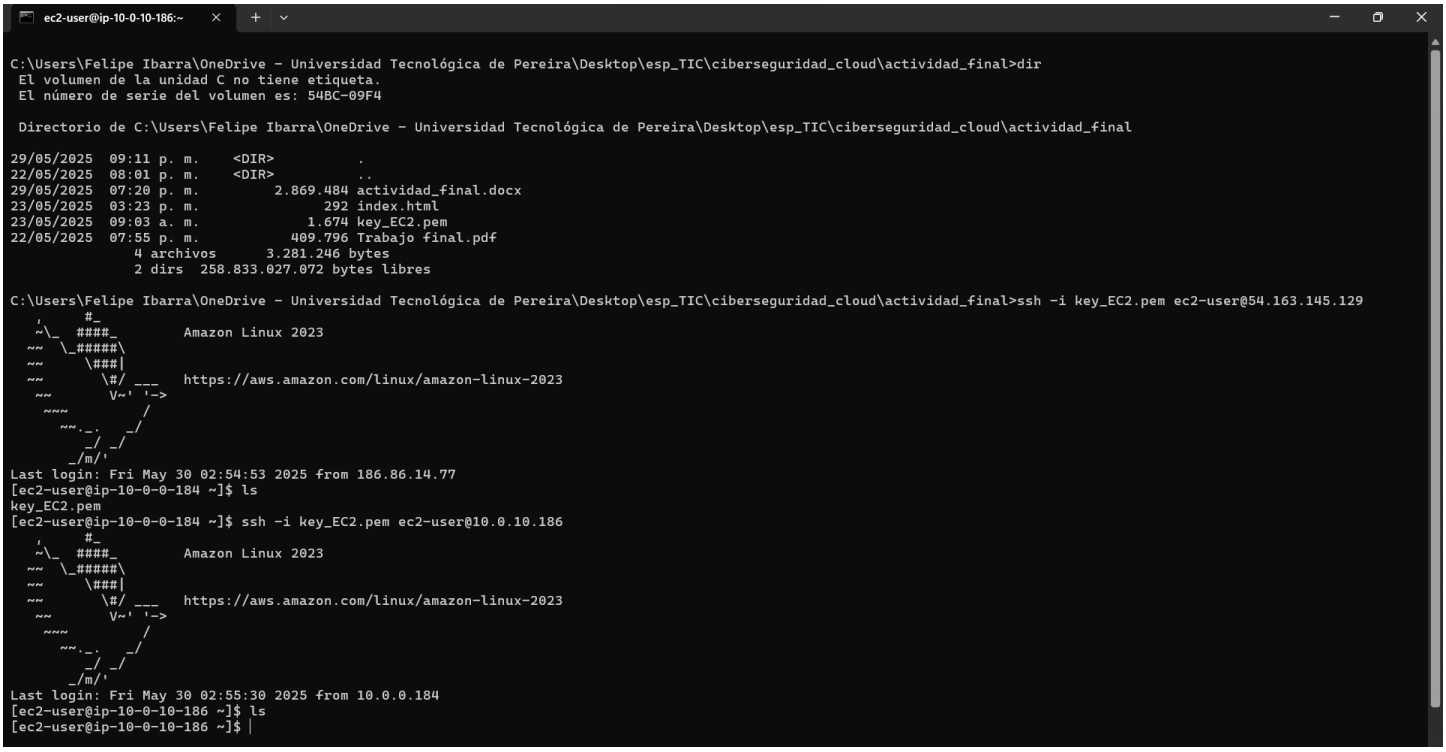


Figura 26. Conexión a DB-Server desde Web-Server-1 mediante SSH

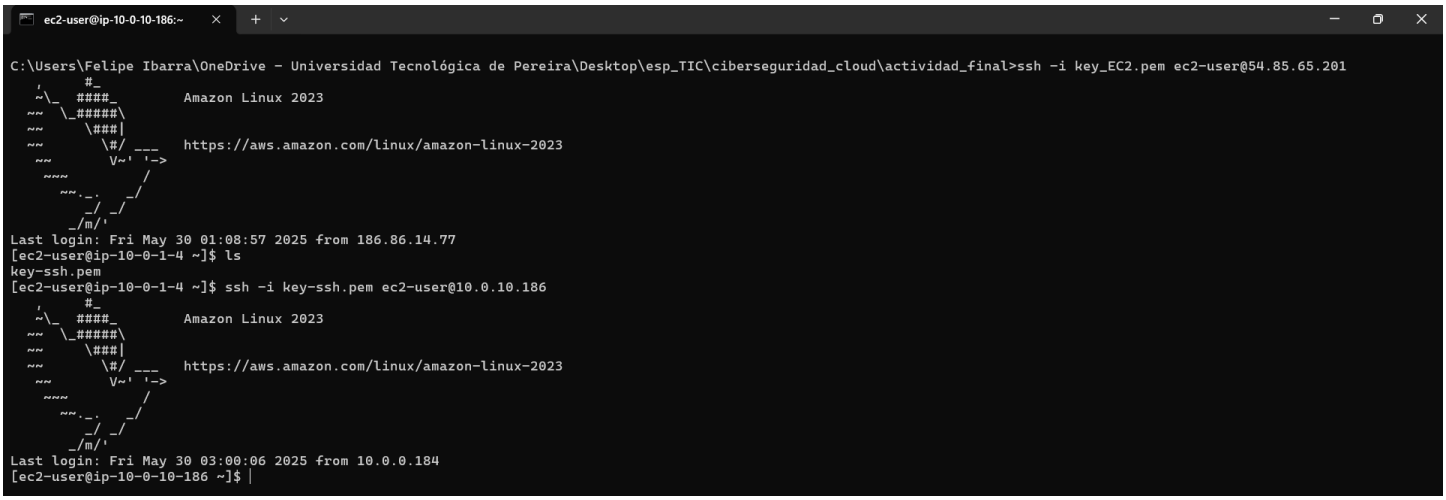


Figura 27. Conexión a DB-Server desde Web-Server-2 mediante SSH

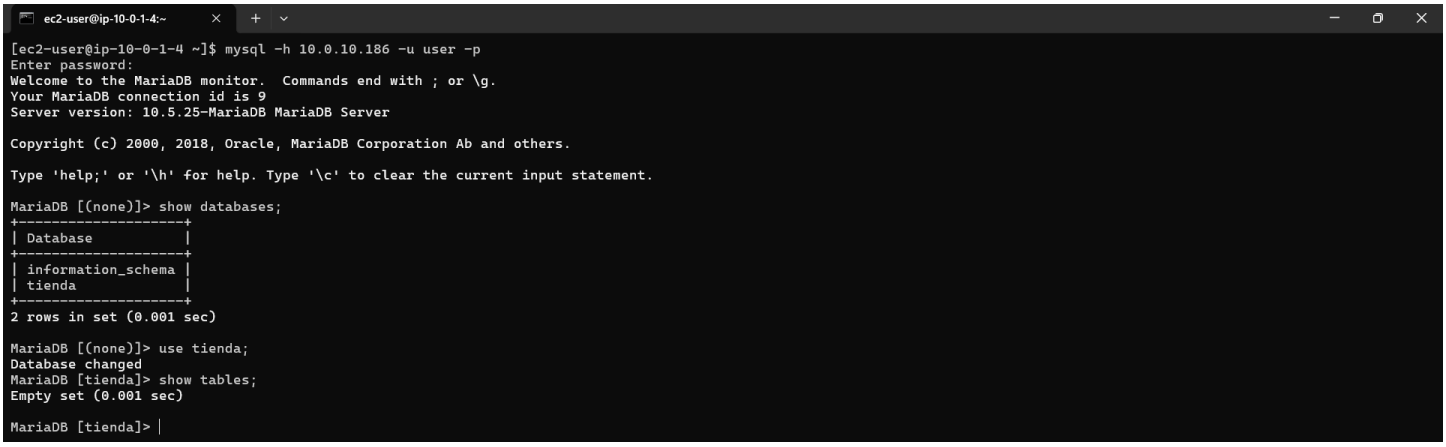


Figura 28. Conexión a DB-Server desde Web-Server-2 mediante MYSQL

Conclusiones

A lo largo del desarrollo de este laboratorio se implementó una arquitectura completa y segura en AWS, utilizando servicios fundamentales como VPC, subredes públicas y privadas, instancias EC2, buckets S3, grupos de seguridad, ACLs y NAT Gateway. La solución fue diseñada siguiendo buenas prácticas de segmentación, aislamiento de capas y control de accesos, permitiendo el despliegue de una aplicación distribuida con separación entre la capa web y la capa de datos, junto con almacenamiento externo en S3 para archivos estáticos y logs de auditoría. A pesar de las restricciones del entorno AWS Academy Learner Lab, que limitaron el uso de roles y usuarios IAM o la aplicación de ciertas políticas públicas, se documentó adecuadamente cómo se habría resuelto cada caso en un entorno productivo real.

Durante el proceso, se comprobó la importancia de un diseño planificado en la configuración de reglas de red, flujos de tráfico y acceso a recursos. La creación de buckets con cifrado, versionado y trazabilidad, el uso de Security Groups precisos, y la implementación de ACLs para reforzar el control a nivel de subred, permitieron aplicar los principios de seguridad desde la arquitectura. La validación de cada fase, incluyendo pruebas de conectividad, acceso restringido, y la interacción controlada entre servicios, reforzó la comprensión de cómo funciona la infraestructura en la nube y la responsabilidad compartida en la gestión de la misma.
En general, este laboratorio no solo permitió aplicar conocimientos técnicos sobre los servicios de AWS, sino que fortaleció habilidades clave como el razonamiento lógico, la documentación técnica, y la capacidad de solucionar problemas en entornos reales con limitaciones. El trabajo realizado evidencia la capacidad para diseñar soluciones sólidas, seguras y escalables en la nube, y marca una base sólida para enfrentar desafíos más complejos en entornos empresariales o profesionales.