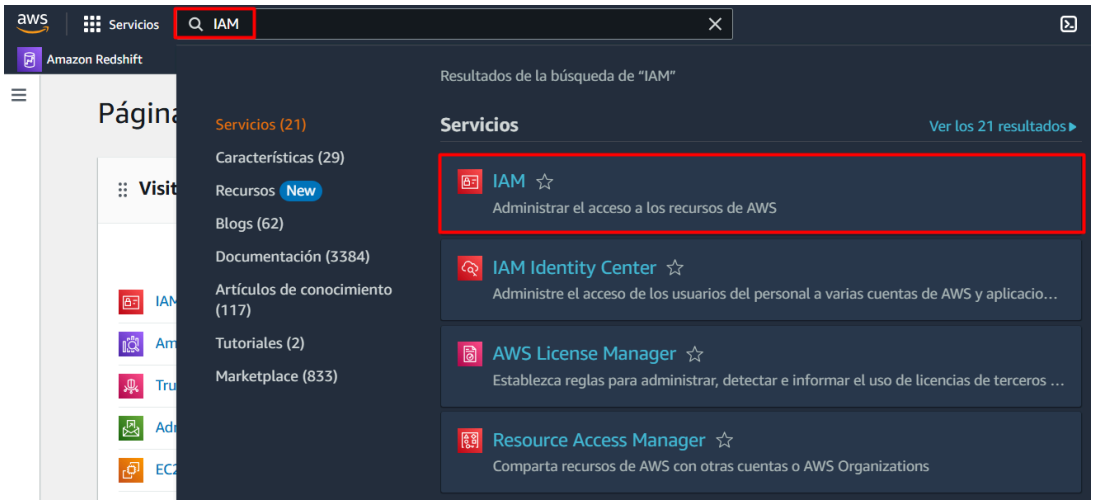
LABORATORIO DE AWS IAM, KMS Y SECRETS MANAGER

En este laboratorio vamos a crear usuarios y grupos, a agregar usuarios a los grupos y a asignar políticas a los usuarios y a los grupos.
Además, vamos a crear una key con KMS y a proteger secretos con Secrets Manager.

USUARIOS

1. Iniciamos sesión en la consola de administración de AWS e ingresamos a la consola de IAM:

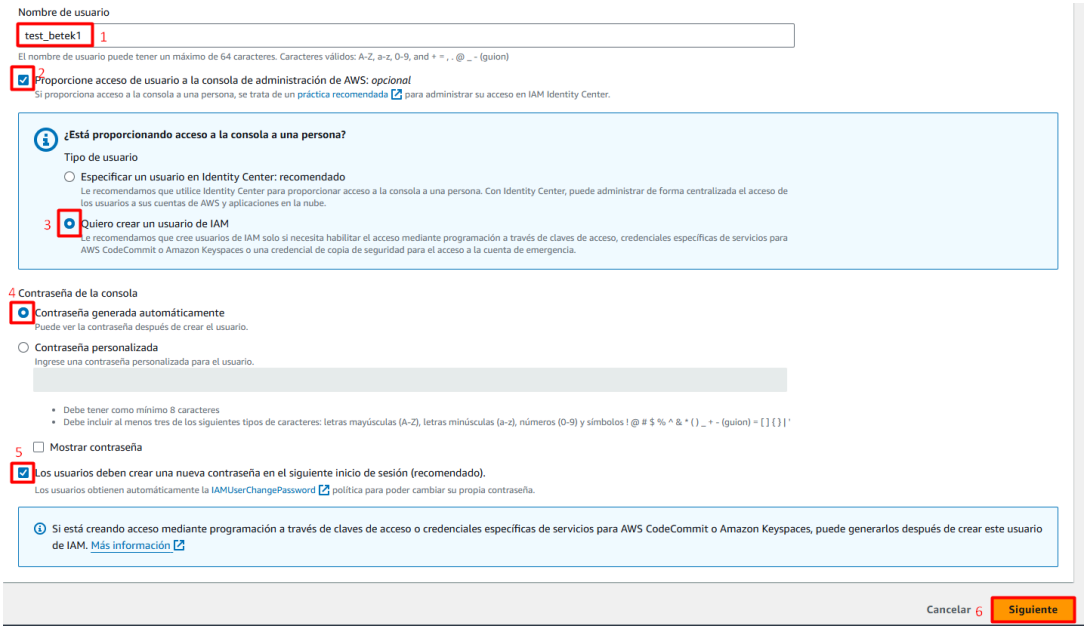


2. En el menú lateral, seleccionamos **Usuarios**:

3. Damos clic en **Crear usuario**:

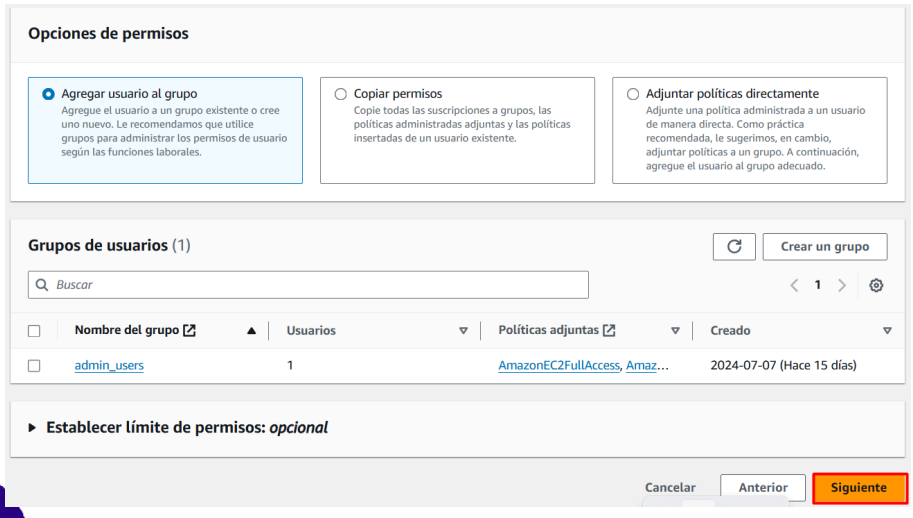
4. Especificar los detalles del usuario:

- Nombre de usuario: El nombre de usuario puede tener un máximo de 64 caracteres. Caracteres válidos: A-Z, a-z, 0-9, and + = , . @ _ - (guion).
  En este ejemplo usamos ***test_betek1***, puedes usar tu nombre si deseas.
- Proporcione acceso de usuario a la consola de administración de AWS: **Check (marcar)**.
  La consola de administración es la interfaz gráfica de AWS, por la que accedemos a los servicios a través del navegador del web.
- Seleccionar **Quiero crear un usuario de IAM**.
- Cuando vamos a crear la contraseña de la consola, tenemos dos opciones, Contraseña generada automáticamente y Contraseña personalizada. Para este laboratorio vamos seleccionar **Contraseña generada automáticamente**.
- Podemos determinar que los usuarios deban crear una nueva contraseña en el siguiente inicio de sesión, esta es una buena práctica recomendada de seguridad: **Marcar (check)**.
- Clic en **Siguiente**.

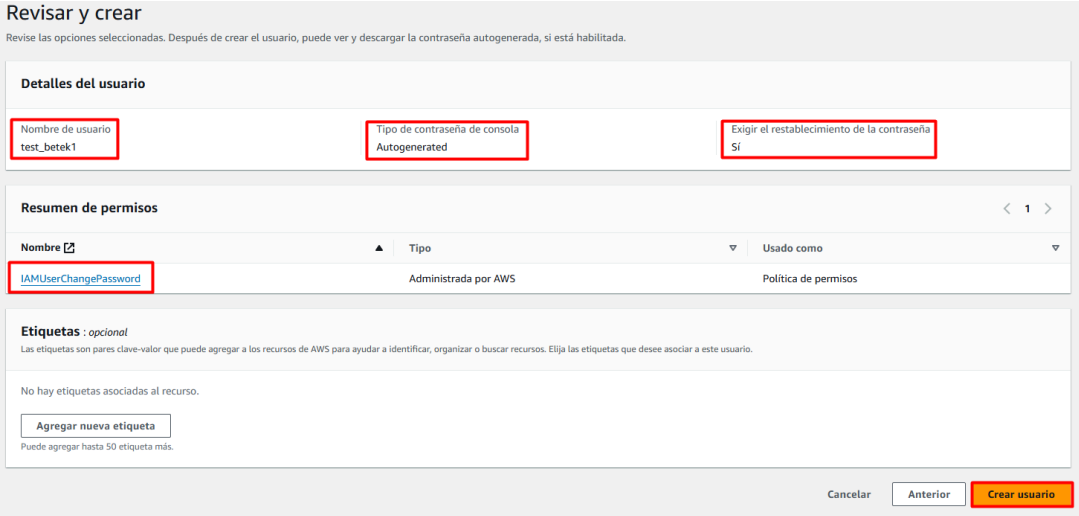


5. Establecer permisos: En este paso vamos a indicar qué acciones podrá realizar este usuario sobre los recursos de AWS:
   - Podemos agregar el usuario a un grupo, de esta manera el usuario heredará los permisos del grupo. Se recomienda utilizar grupos para administrar los permisos de usuario según las funciones laborales.
   - También podemos copiar los permisos de otro usuario ya existente.
   - O adjuntar una política administrada a un usuario de manera directa. Como práctica recomendada, se sugiere adjuntar políticas a un grupo y luego agregar el usuario al grupo adecuado.

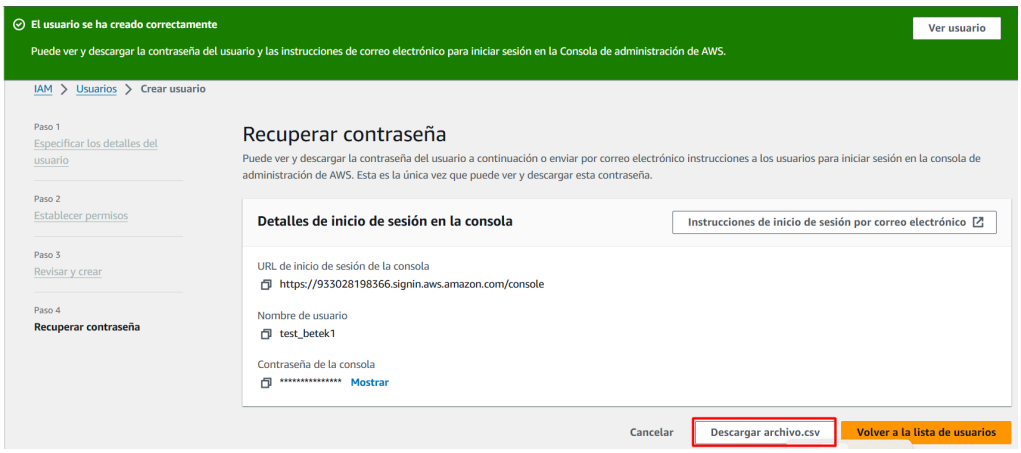
   Por ahora no le vamos a dar ningún tipo de permiso a este usuario y damos clic en **Siguiente**:



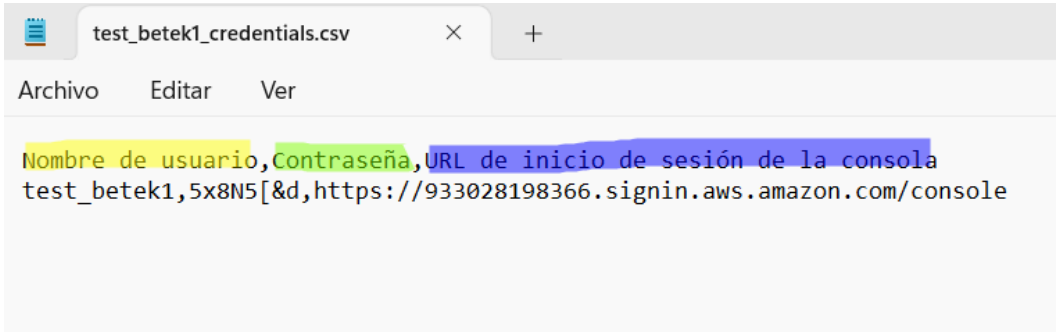
6. Revisar y crear: En este paso revisamos la configuración del usuario que vamos a crear, el nombre es ***test_betek1***, el tipo de contraseña es autogenerada, sí requiere un restablecimiento de la contraseña y, el único permiso que tiene es el de cambiar su propia contraseña, esto es para que él pueda cambiar su contraseña la primera vez que ingrese a AWS.
   Después de revisar, damos clic en **Crear usuario**:



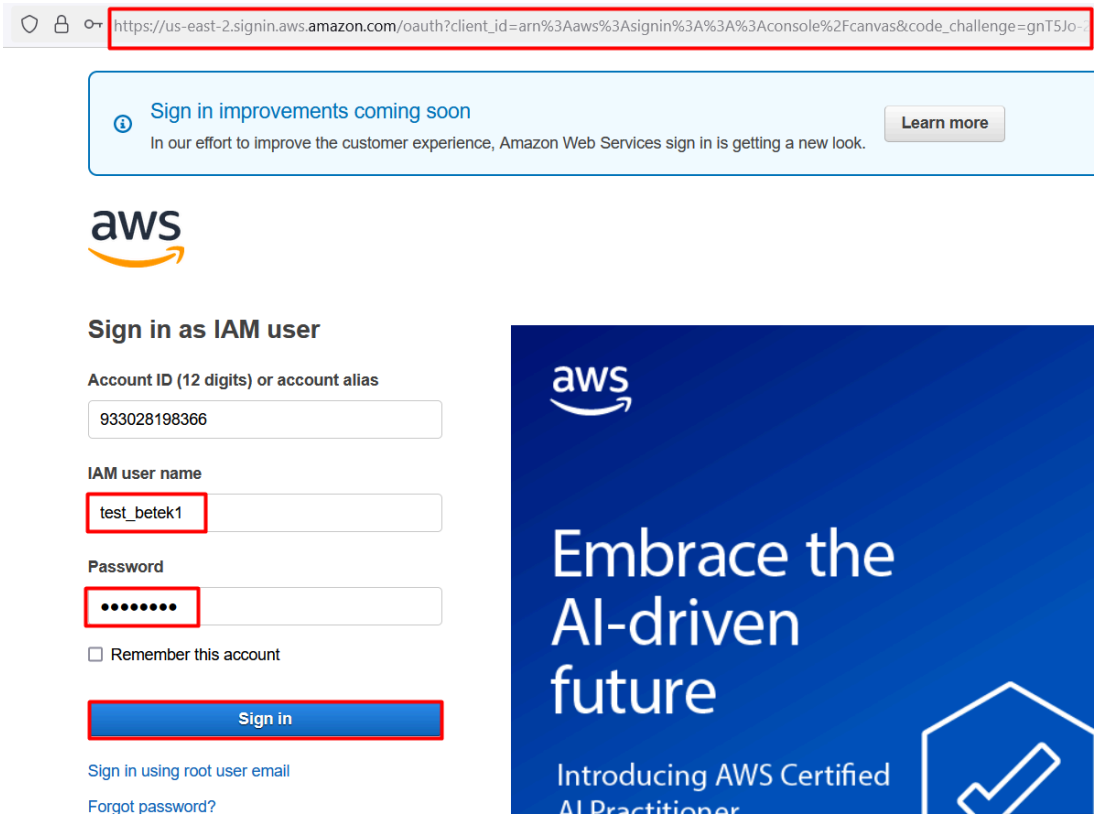
7. Una vez tengamos el mensaje de que el usuario se creó correctamente, descargamos las credenciales de este nuevo usuario:



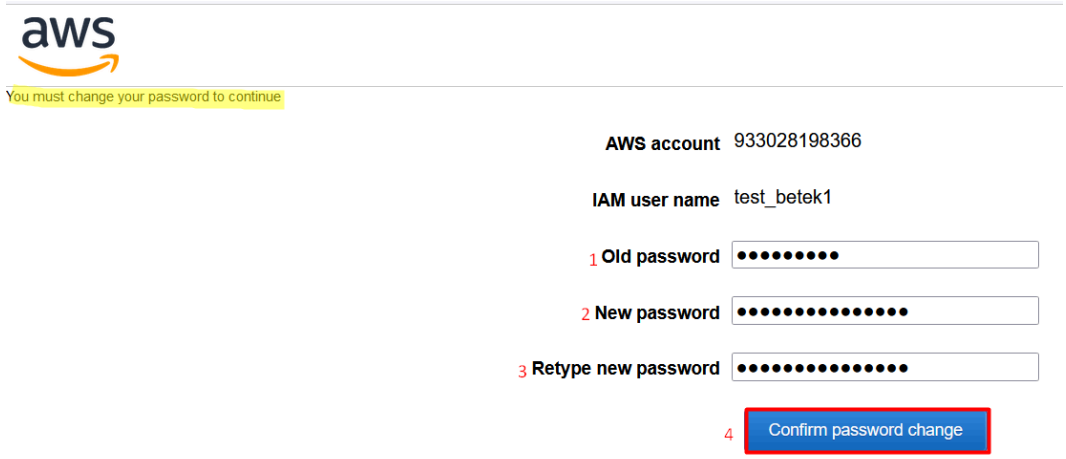
8. El archivo que descargamos, en formato CSV, tiene tres parámetros separados por coma: Nombre de usuario, contraseña y URL del inicio de sesión, con estos tres datos nuestro usuario ya puede ingresar a la consola de AWS:



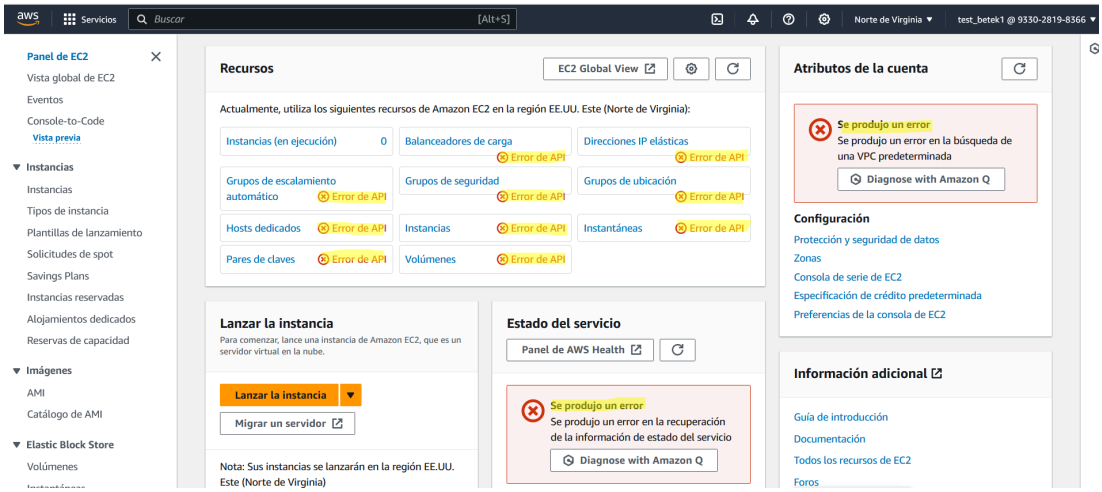
Para evitar conflictos entre las sesiones, pegamos la URL en otro navegador e ingresamos con los datos del CSV descargado:



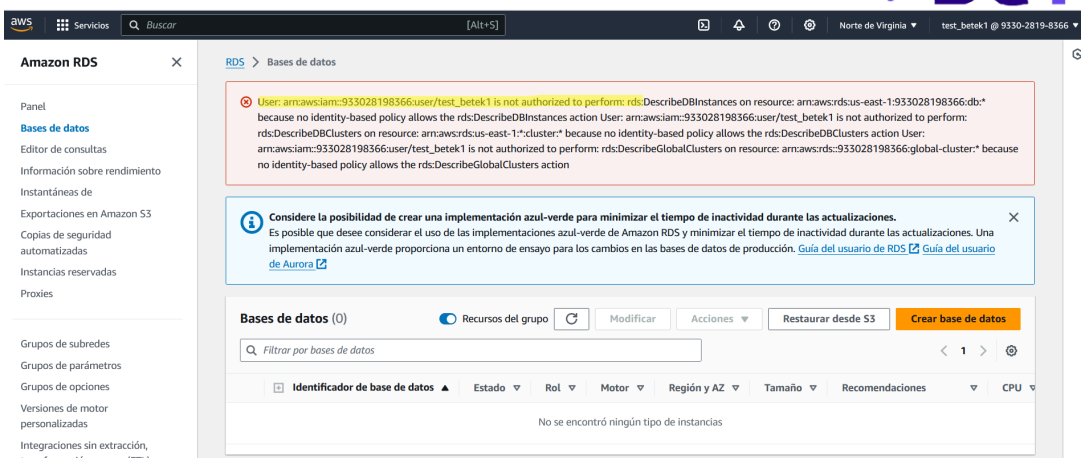
Como se lo indicamos en la creación del usuario, nos va a pedir cambio de contraseña, ingresamos la actual (del CSV) y establecemos la nueva:



Si navegamos a la consola de EC2, nos va a salir error, no podemos ver las instancias, los balanceadores de carga, nada:

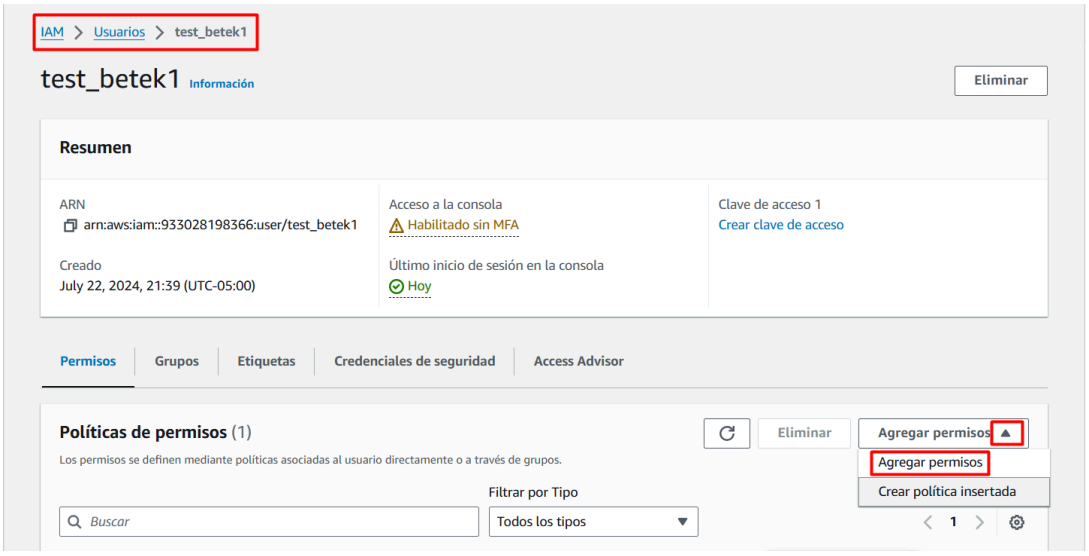


Lo mismo sucede si navegamos en la consola de RDS, o en cualquier otro servicio, no podemos ver nada:

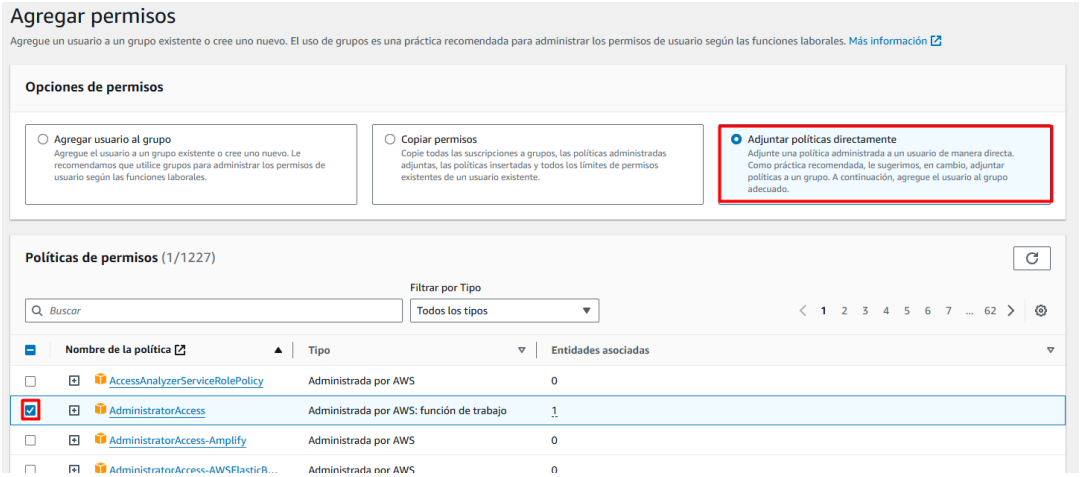


Esto se debe a que el usuario no tiene permiso en EC2, ni en RDS, ni en ningún recurso de AWS, el único permiso que tiene el usuario es el de cambiar su propia contraseña.
Para solucionar esto, debemos ir a la cuenta administradora y agregar los permisos al usuario para poder realizar estas acciones.

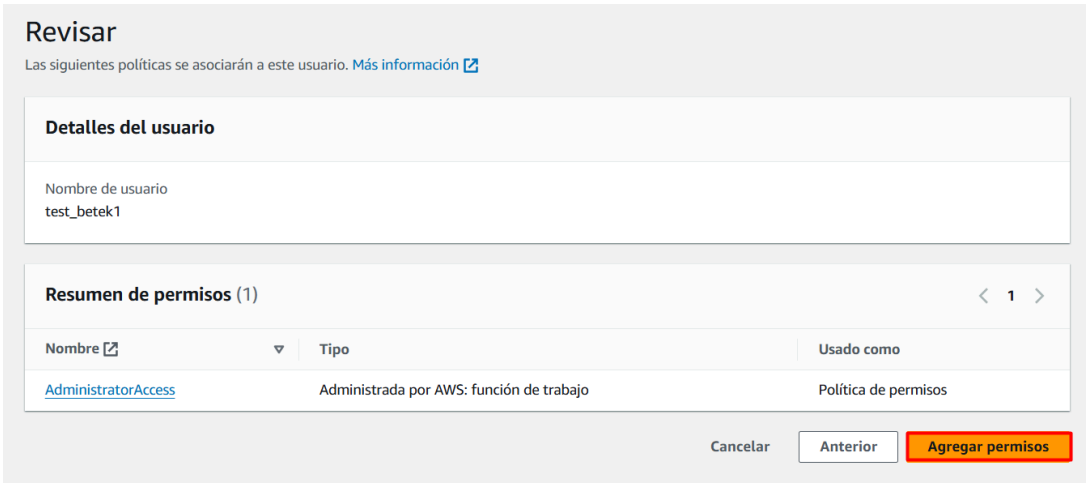
9. En la consola del usuario root, vamos al servicio **IAM**, ingresamos a la opción **Usuarios**, buscamos el usuario que creamos (en este ejemplo ***test_betek1***), desplegamos el menú Agregar permisos y seleccionamos **Agregar permisos**:



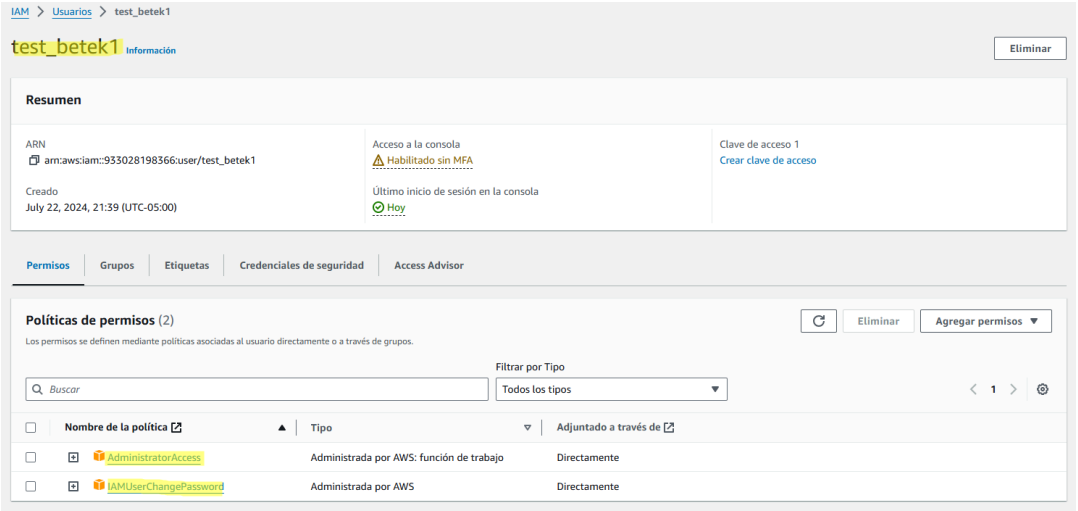
Vamos a adjuntarle una política que ya existe con los permisos requeridos: Seleccionamos **Adjuntar políticas directamente**, seleccionamos **AdministratorAccess**, esta política le va a dar acceso al usuario a todos los servicios de AWS y va a poder realizar cualquier acción:



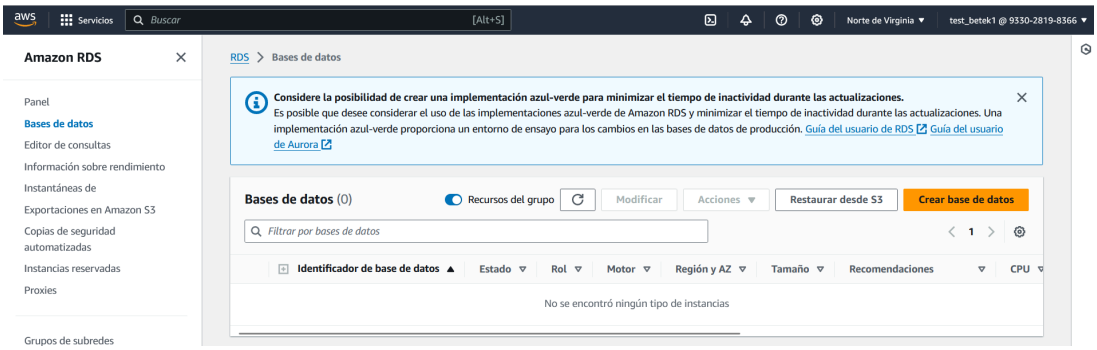
Al final de la página, damos clic en **Siguiente** y luego en **Agregar permisos:**

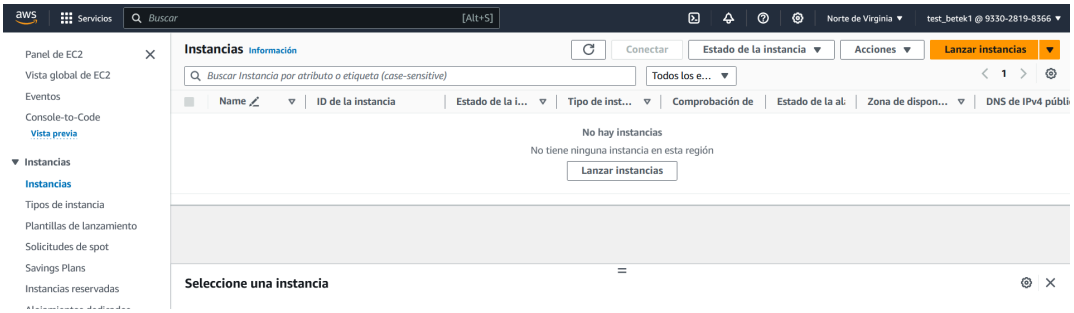


Ahora vemos que el usuario tiene el permiso FullAccess y el permiso de cambiar su contraseña:



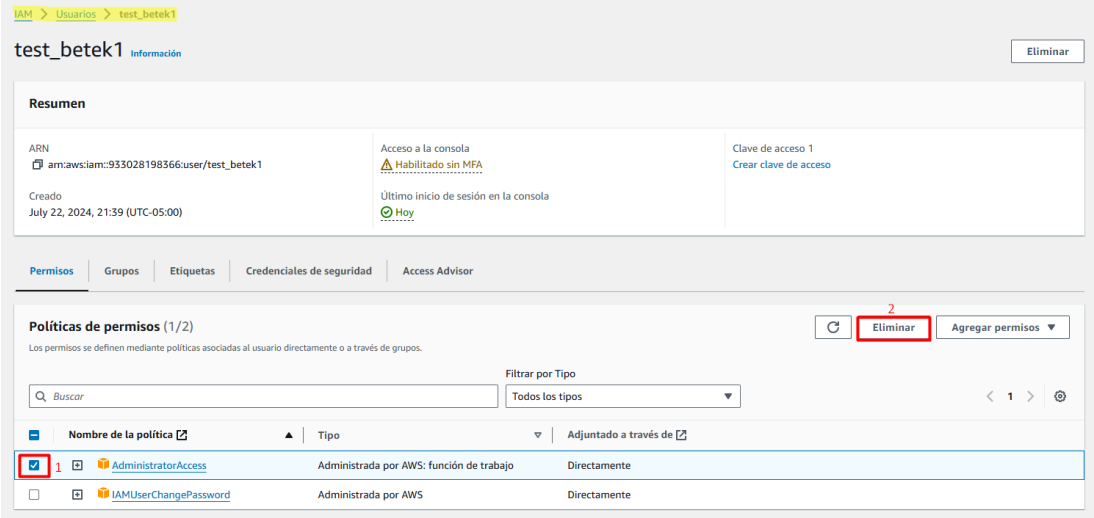
10. Regresamos a la sesión del usuario ***test_betek1***, damos F5 o refrescamos la página y ahora no tenemos los mensajes de error que teníamos antes de asignar los permisos:



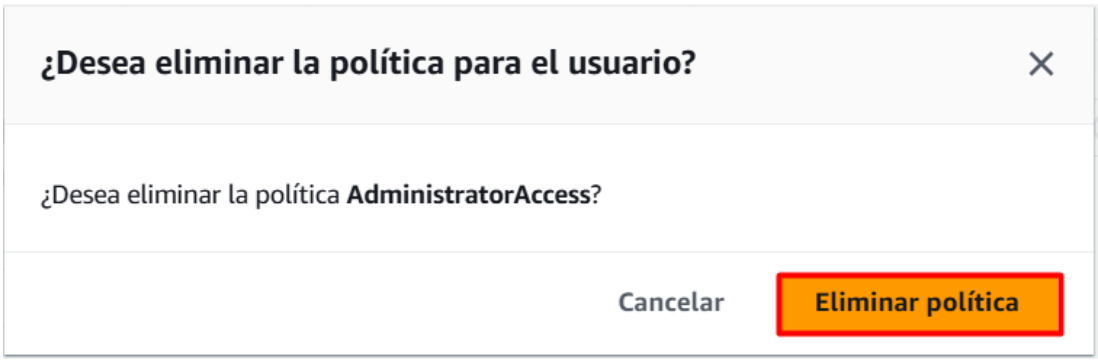


El usuario no ve bases de datos RDS ni instancias EC2 porque aún no ha creado ninguna, pero ya no tiene mensajes de error por falta de permisos, como antes.

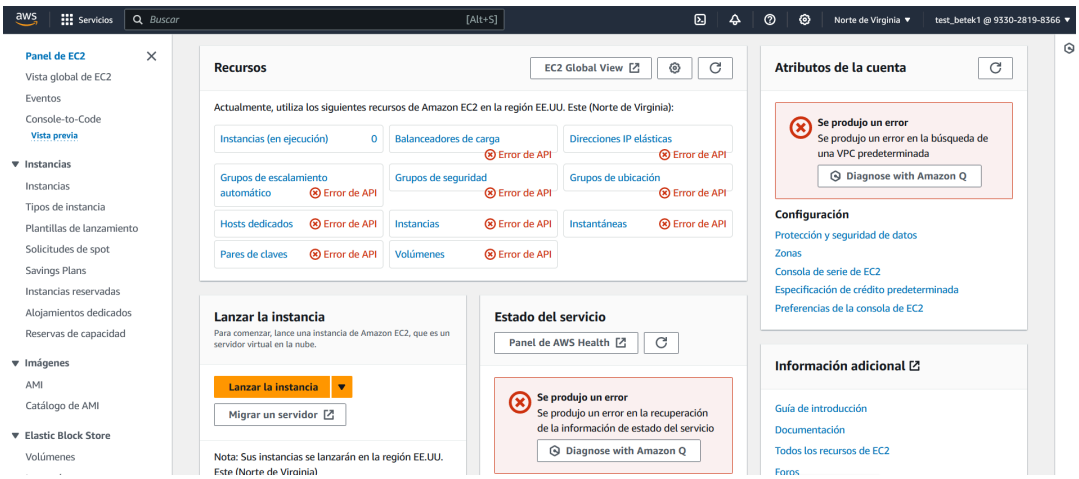
11. Ahora vamos a regresar a la cuenta del usuario root y le vamos a quitar los permisos de administrador al usuario ***test_betek1***:

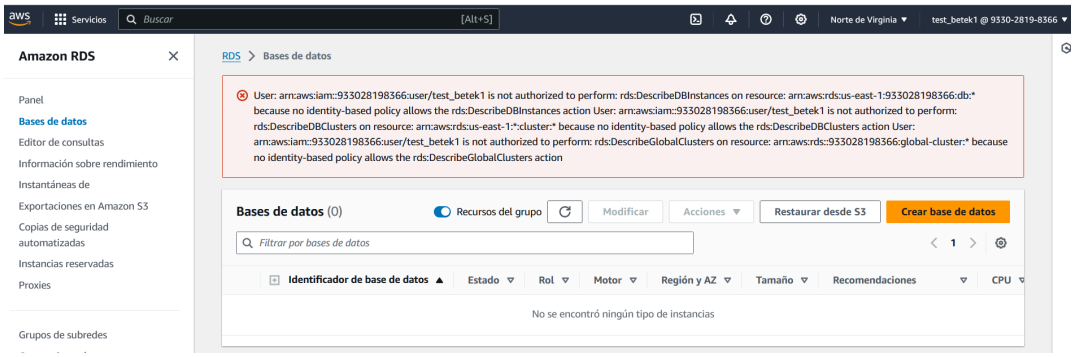


Confirmamos que queremos eliminar la política:



12. Si regresamos a la sesión del usuario ***test_betek1***, damos F5 o refrescamos la página ya no debería tener acceso a ver los recursos de EC2, ni de RDS, ni ningún otro recurso:

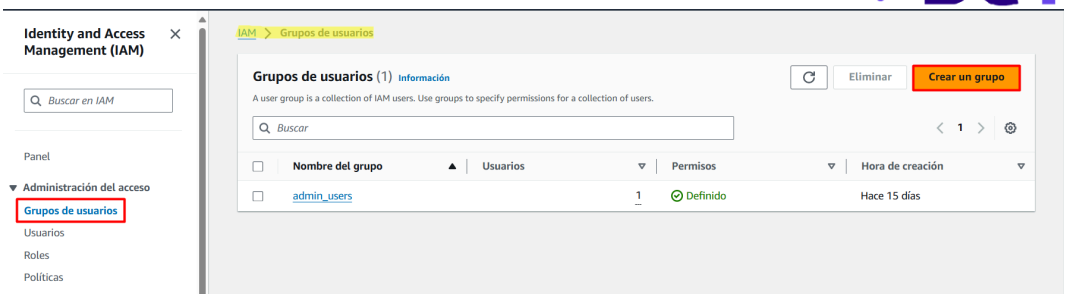




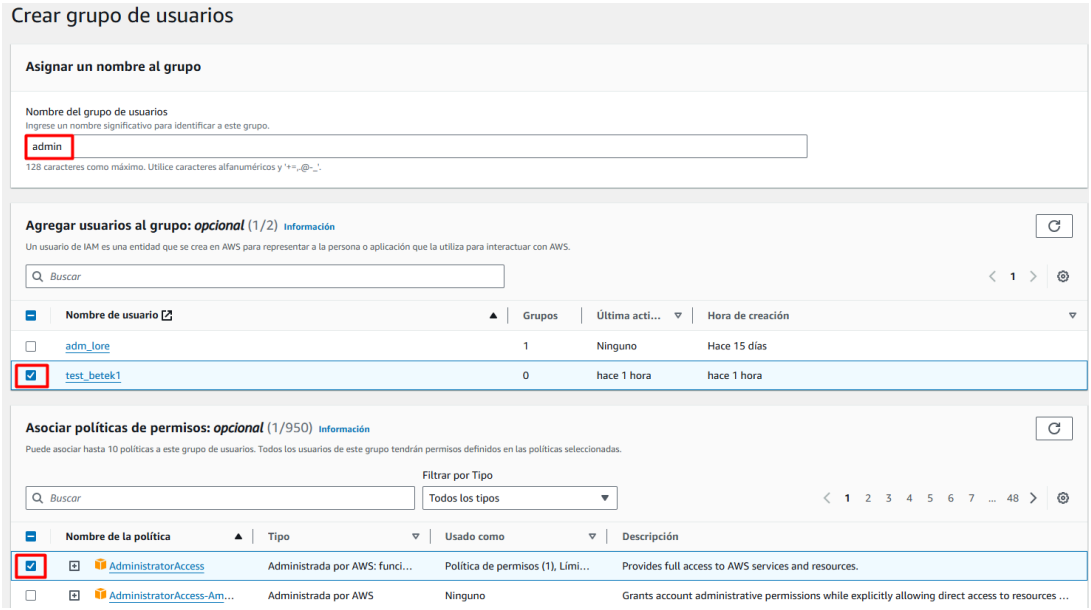
GRUPOS

Para evitar esto de estar asignando y quitando permisos a los usuarios, lo más recomendable es, primero crear grupos de usuarios y luego agregar los usuarios al grupo que le corresponda, según los permisos que requiera.
En este caso, creamos un grupo de administradores y a todos los usuarios que sean administradores dentro de nuestra empresa, los agregamos a este grupo.

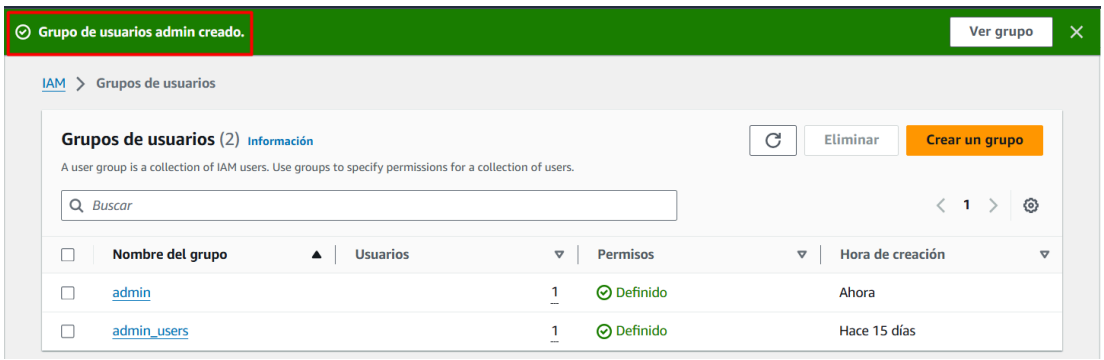
13. En la sesión del usuario root, vamos a la consola de **IAM**, seleccionamos **Grupos de usuarios** y damos clic en **Crear un grupo**:



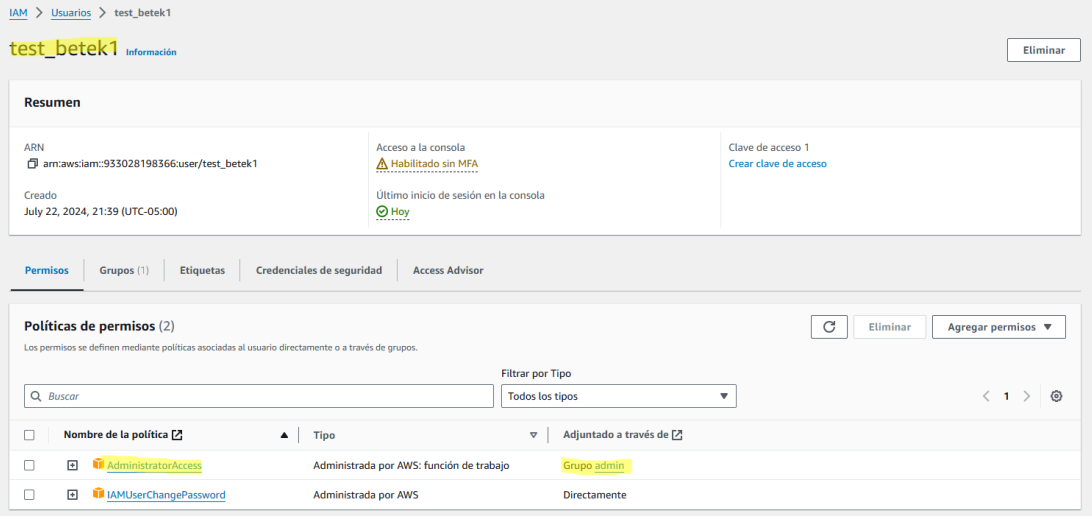
Asignamos un nombre al grupo de usuarios, en este caso será ***admin***, si ya tenemos usuarios creados para agregar al grupo, los seleccionamos, asignamos la política **AdministratorAccesss** y al final de la página damos clic en **Crear grupo de usuarios**:



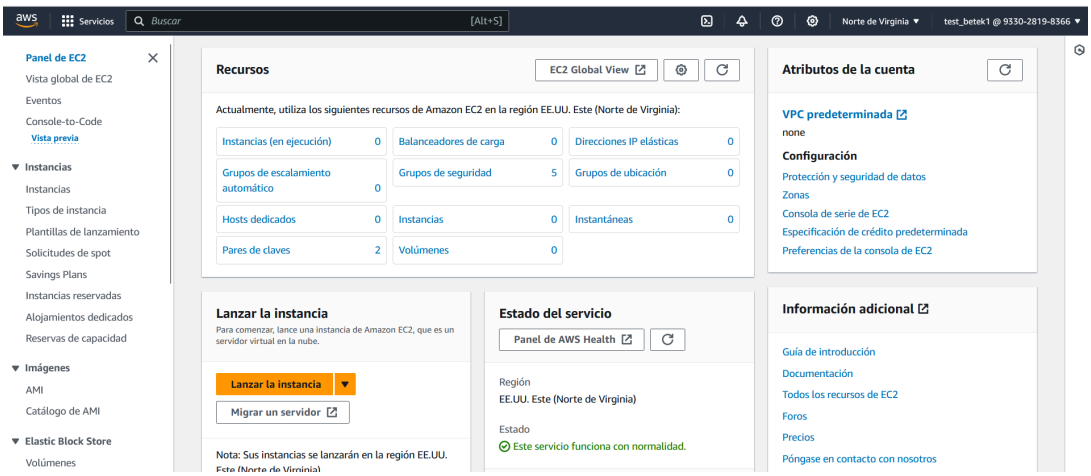
Y así de fácil ya tenemos creado el grupo de administradores, donde tenemos el usuario ***test_betek1***:

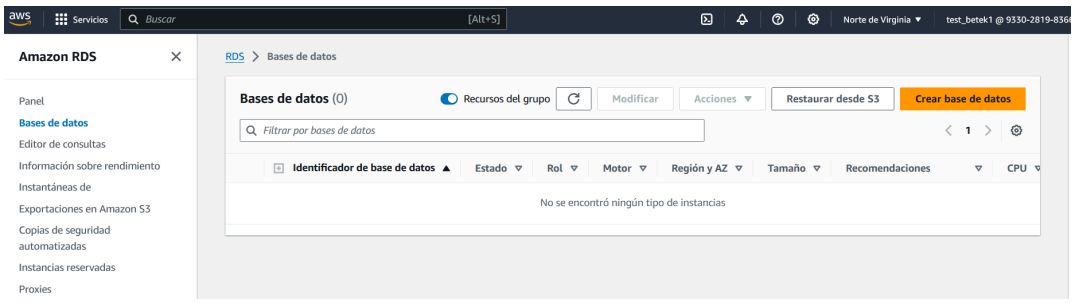


14. Regresamos al usuario ***test_betek1*** y encontramos que nuevamente el usuario tiene la política de administrador, pero esta vez no se le agregó directamente sino a través de un grupo:

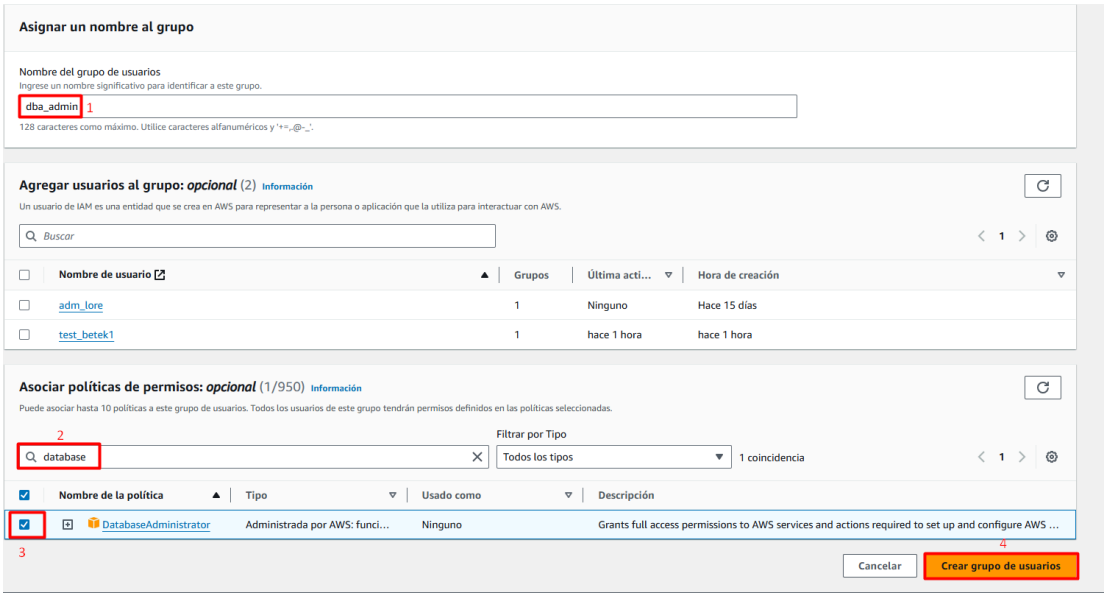


Si regresamos a la sesión del usuario ***test_betek1***, damos F5 y refrescamos la página, vamos a ver cómo este usuario otra vez puede ver todos los recursos (EC2, RDS y demás servicios de AWS) porque nuevamente tiene los permisos para ello:





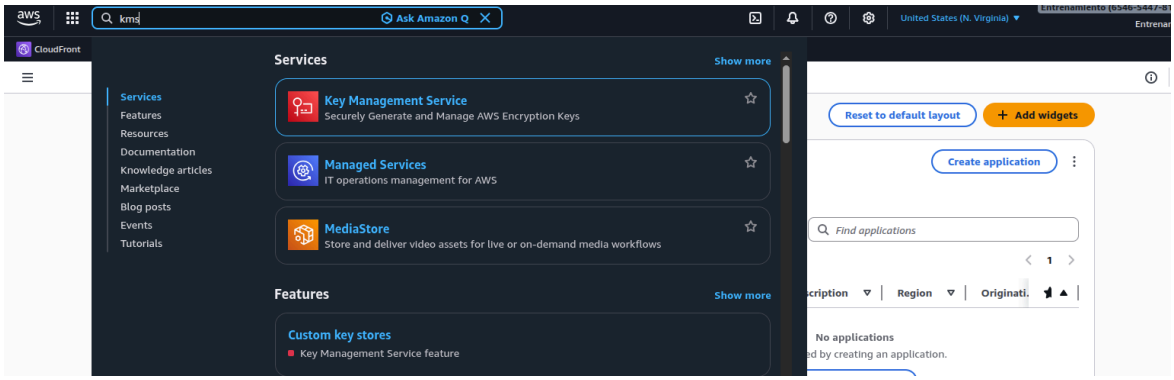
15. Regresemos a la consola de IAM y vamos a crear otro grupo.
    Supongamos que tenemos algunos administradores de bases de datos y queremos crear un grupo para ellos.
    El grupo se va a llamar ***dba_admin***, agregamos los usuarios (en este caso no tenemos usuarios para agregar porque el usuario test_betek1 ya tiene full access y no tiene sentido agregarlo a este grupo), buscamos la política **DatabaseAdministrator**, la seleccionamos y automáticamente se agregan los permisos necesarios para administrar bases de datos en la cuenta de AWS.
    Damos clic en **Crear grupo de usuarios**:



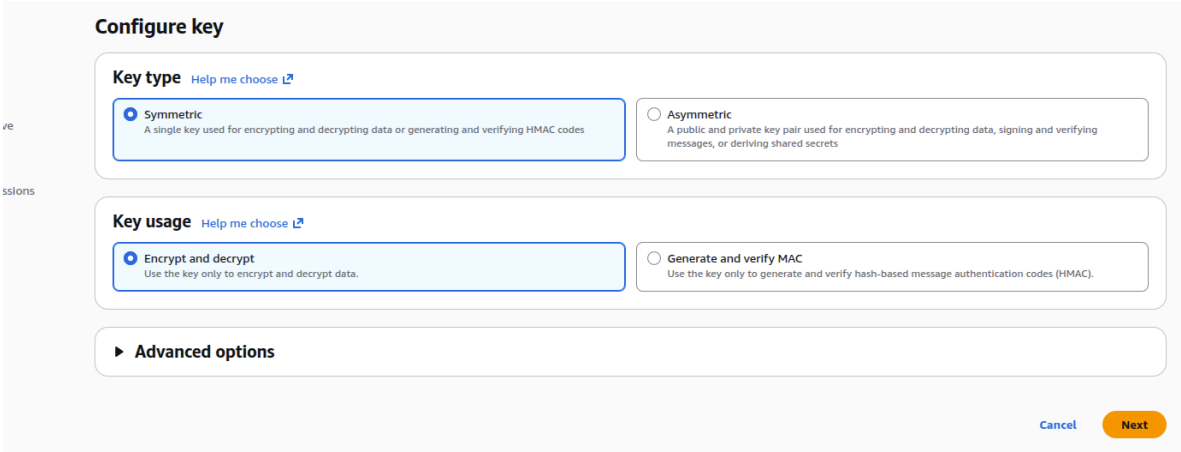
CREACIÓN DE KMS KEY

1. Ir la a consola

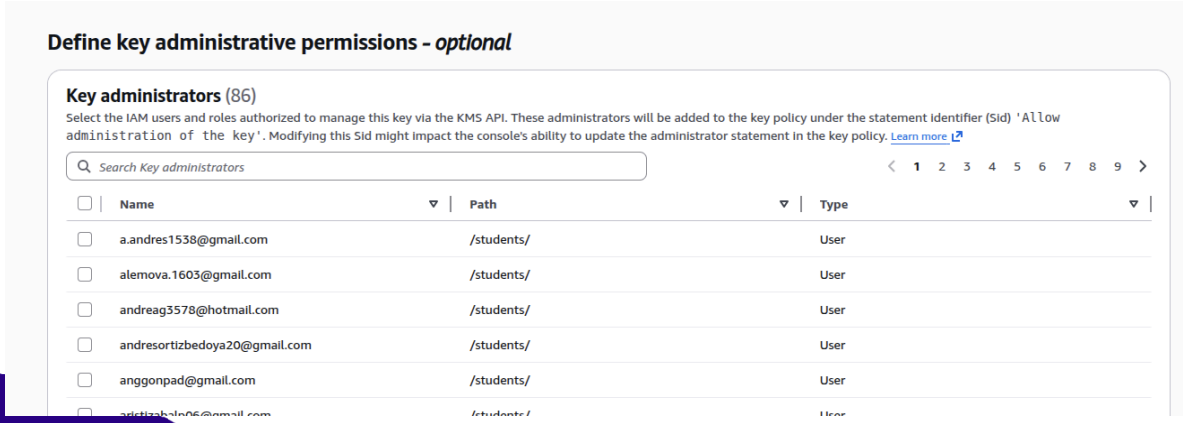
2. Buscamos KMS en el explorador de la consola:



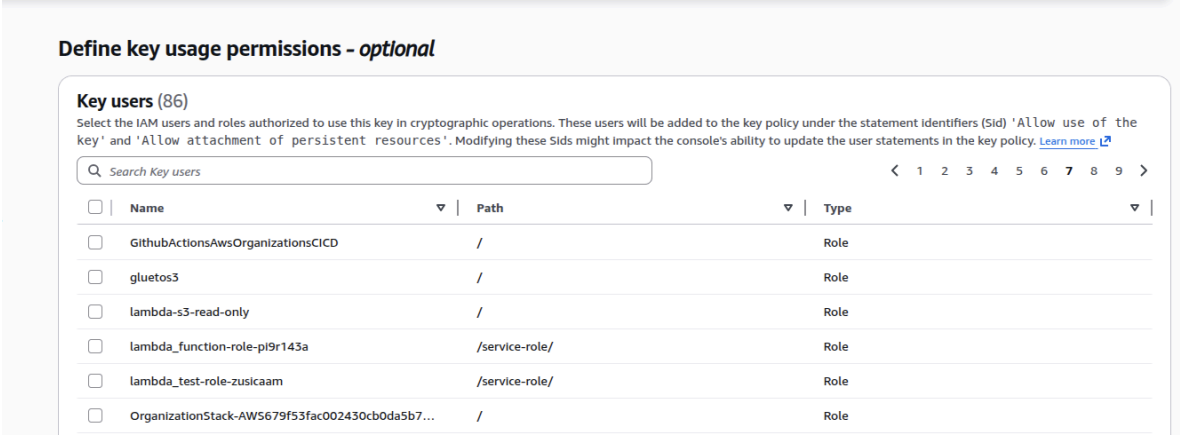
3. Creamos nuestra primera key



4. Nos asignamos como administradores de la key



5. Definimos quién puede usar esa key:



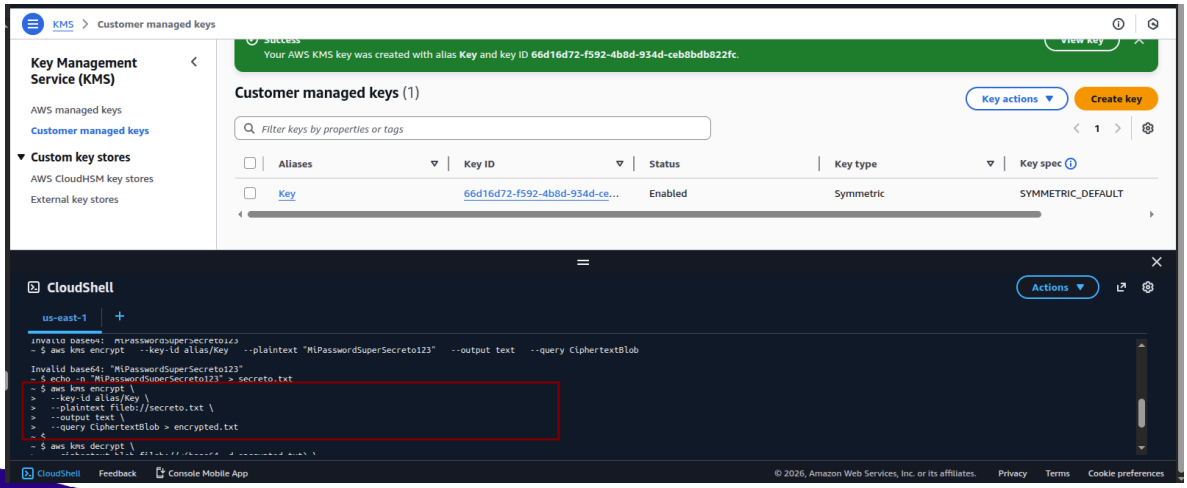
Desde la consola desde la cloudshell, podemos simular el cifrado.

- Con este comando, solo crea un archivo con un texto:

  echo -n "PasswordProduccion2026!" > secret.txt

- Para cifrar corremos este comando, ajustando el valor al de nuestra llave:

  **aws kms encrypt **
  **--key-id alias/app-prod-key **
  **--plaintext fileb://secret.txt **
  **--query CiphertextBlob **
  --output text > secret.encrypted



Ahora, tenemos un secreto cifrado en base64.

¿Qué ocurrió?

- ☐ CLI envió binario
- ☐ KMS generó una data key
- ☐ Cifró el texto
- ☐ Devolvió el blob cifrado

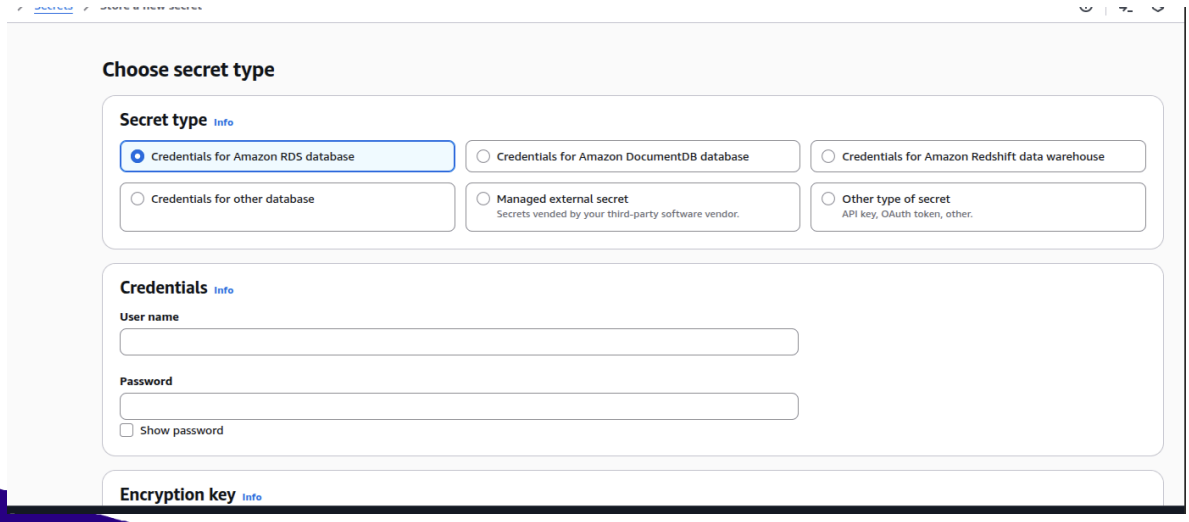
- Para desencriptar

**aws kms decrypt **
**--ciphertext-blob fileb://<(base64 -d secret.encrypted) **
**--query Plaintext **
--output text | base64 --decode

Solo quien tenga permiso `kms:Decrypt` puede hacerlo.

USUARIOS SECRETS MANAGER

1. En la consola, vamos al servicio Secrets Manager y luego a Store new secret



2. Ponemos en texto plano:

```
{
  "username": "app_user",
  "password": "PasswordFuerte2026!"
}
```

3. Necesitamos una política de IAM que permita usar el secreto

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "secretsmanager:GetSecretValue",
                                                                                "Resource":
"arn:aws:secretsmanager:us-east-1:654654478122:secret:prod/app/db-credentials-*"
    }
  ]
}
```

4. Para ejemplo práctico crearemos un rol, el cual podrá usar dicho secreto desde una instancia:

- Creamos una instancia
- Asociamos el rol que permite el secreto
- Nos conectamos via ssh a la ec2
- Ejecutamos:

```
aws secretsmanager get-secret-value \
  --secret-id prod/app/db-credentials \
  --query SecretString \
  --output text
```

- Nos permitirá y devolverá el secreto en formato json

RECUERDA: Eliminar todos los recursos desplegados en esta práctica