APUNTES AWS

Manuel Vergara
2022

Índice

Este documento contiene los apuntes tomados en el curso «Amazon AWS al completo» impartido por Apasoft Training en agosto y septiembre de 2022. El curso udemy consta de 46 horas aproximadamente de vídeo-tutoriales. Las prácticas aquí contenidas tuvieron una duración de alrededor de unas 220 horas.

Los apuntes no fueron pensados para compartirlos, por ello pueden tener lagunas de información o contenido adicional respecto al curso, ya que se redactaron para recordar procedimientos y conceptos que el autor creyó relevantes. Teniendo un documento, a mi parecer, tan completo y entendiendo que el conocimiento debe ser libre se decidió compartirlo.

Si te parece útil este documento puedes agradecerlo a través de las vías de contacto de la web https://vergaracarmona.es

Recuerda,

> *"Quien se corta su propia leña se calienta dos veces"*

TEMA 1 - Introducción a Cloud Computing

Cloud Computing (nube) entrega de recursos tecnológicos bajo demanda.

NIST dice que es un modelo que permite un acceso a través de la red a un conjunto de recursos informáticos que son configurables, con aprovisamiento rápido y con la posibilidad de liberar con un mínimo esfuerzo.

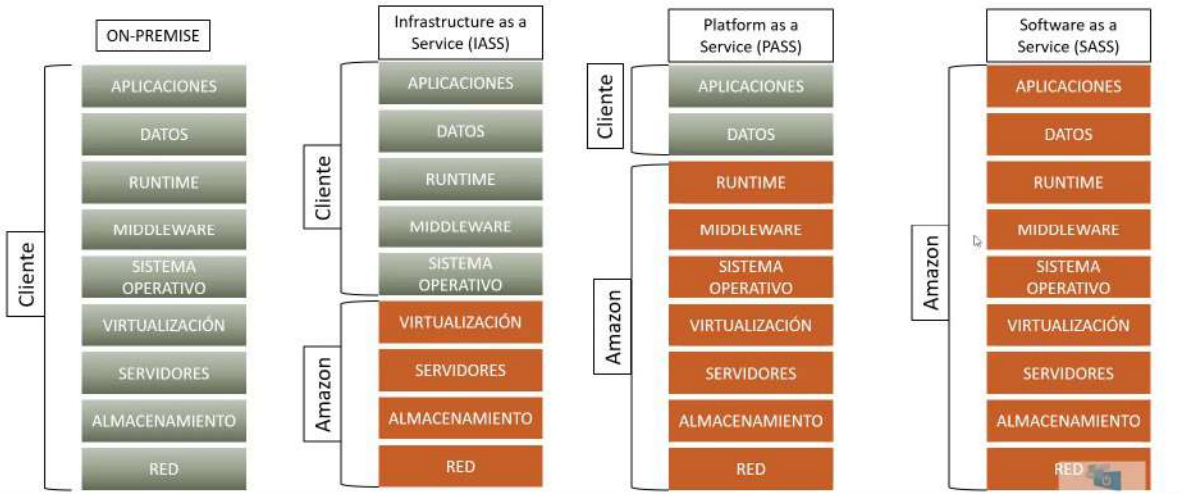
Recursos: Networks, Servidores, Almacenamiento, Aplicaciones, Servicios, Seguridad, Redes....

Características:

- On-demand Self Service. Se puede solicitar los servicios bajo demanda.
- Acceso a través de internet.
- Recursos compartidos
  - Modelo multi inquilino.
  - Se asignan y reasignan dinámicamente
  - Los clientes no se preocupan por la ubicación física
- Gran escabilidad.
- Control total sobre los gastos en servicios. Facturación al milímetro, si no se usa un servicio no se paga.

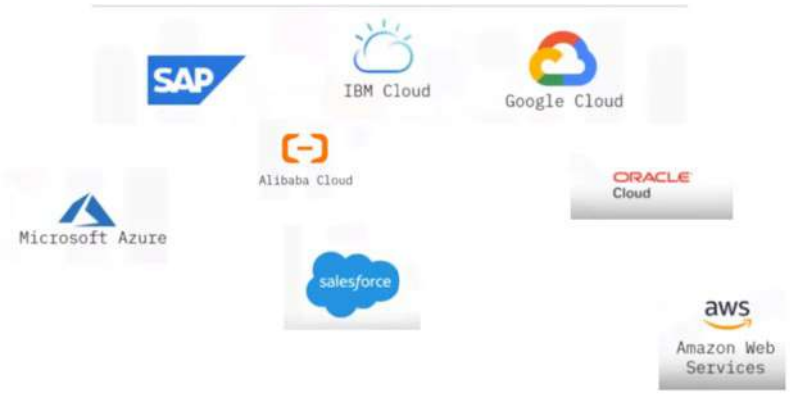
CCaS – Cloud Computing as a Service.

Modos de despliegue en los Cloud. Redes públicas, redes privadas (pueden ser on-premise) y las híbridas.



Incremento en gasto en Cloud: El mercado mundial de servicios en la nube pública crecerá.

Líderes en Cloud Computing:



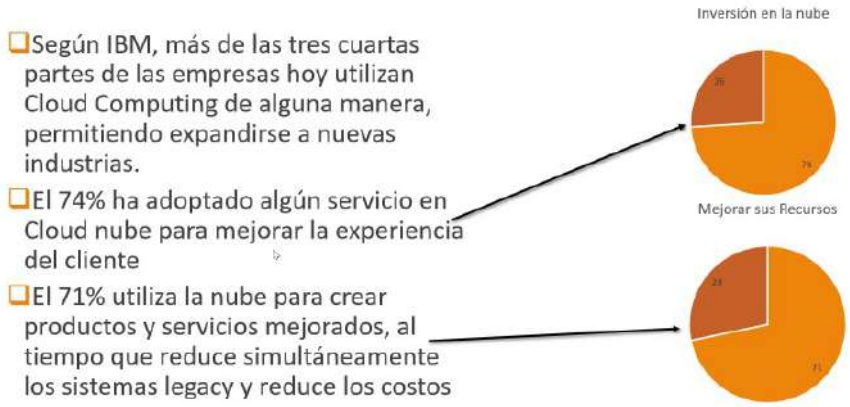
Salesforce y SAP – Ofrece productos de RRHH, Gestión de ventas, gestión empresarial...

Alibaba se conoce también por Aliyun, tiene su propio nicho de mercado.

Pasar a la nube (Se han democratizado los recursos):

- Pequeñas empresas tienen a su disposición recursos antes inviables.
- Ya no es necesario una gran inversión inicial.
- Se puede mantener la competividad
- Permite incluir tecnologías de vanguardia: Workflows, IA, Automatización, IoT, Blockchain, infraestructura híbridas/seguras, aplicaciones nuevas o heredadas, etc

- Es una estrategia, no es la solución para todo.



TEMA 2 - Introducción a AWS

La capa gratuita de Amazon: Ofrecen al menos para un año algunos productos gratis para probar. https://aws.amazon.com/es/free

Existen pruebas temporales, de 12 meses gratuitos y gratis siempre.

Regiones amazon:

Regiones google:

Una región de AWS está compuesta de **AZ** (Availability Zones. Zonas de disponibilidad) que son CPD. Aun estando separados no estarán a más de 100 millas.

También están las **localizaciones de borde** que contienen una infraestructura menos compleja que una AZ y acercan servicios a los usuarios. Cloud Front es una aplicaciones para llegar a los usuarios locales.

Las **zonas locales** están solo en EEUyU. Para la sensibilidad en latencia con servicios de BBDD, aplicaciones, etc

Wavelength - permite a los desarrolladores crear aplicaciones que ofrecen latencias de milisegundos de un solo dígito a dispositivos móviles y usuarios finales.

Outposts – Para tener HW amazon en un CPD propio y usarlo de manera híbrida, utilizando CPD propio y la nube de AWS

Motivos para seleccionar una región:

- Legales
- Precio
- Latencia. Distancia con el usuario final
- Disponibilidad de servicios.

AWS Billing Dashboard es la consola para controlar la facturación.

Con CloudWatch puedes crear alarmas de facturación.

Trucos para no gastar dinero en AWS:

- Utilizar la capa gratuita
- Borrar o parar los componentes una vez se haya terminado la prueba
- Utilizar instancias de tipo Spot (No se pueden utilizar en producción)

Personal AWS Health se utiliza para ver el estado de los servicios.

Se supone que hay una visión general de amazon pero el enlace no funciona: https://status.aws.amazon.com/

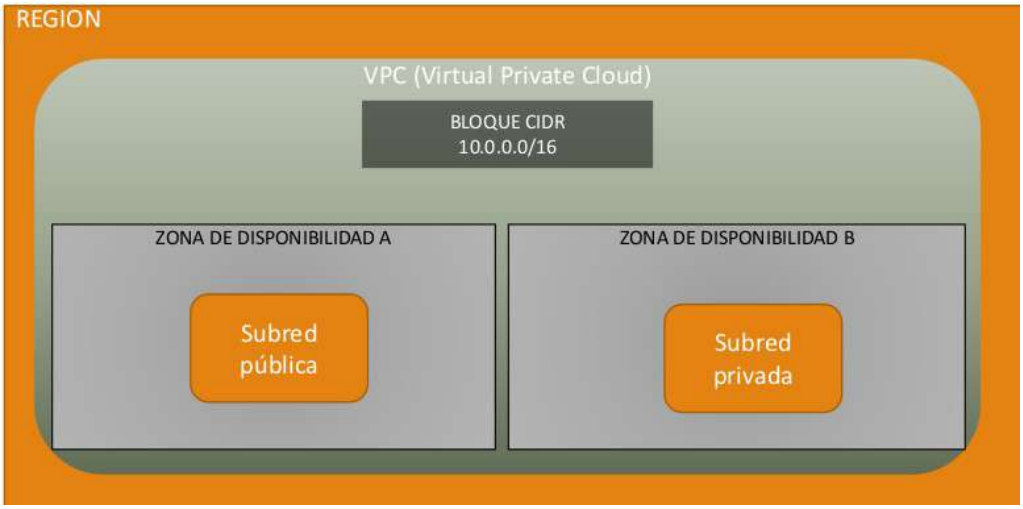
Trusted Advisor da recomendaciones para posibles problemas de la infraestructura. Tiene una parte gratuita y otra de pago (El más barato unos 30USD).

Tipos de soportes: Básico, desarrollador, Empresa, Empresas con sistemas críticos y entornos críticos. https://aws.amazon.com/es/premiumsupport/plans/

TEMA 3 - Networking. VPC.

Una **VPC (Virtual Private Cloud)** es un concepto virtual aislado para desplegar componentes. Es una zona de la red.

Cada VPC tiene un bloque CIDR (IPv4 o IPv6 o de ambas. Classles Inter-Domain Routing (Enrutamiento Entre Dominios Sin Clases, en castellano)). Son el conjunto de direcciones IP que se asignan a la VPC. Son direcciones privadas. Ejemplo: 10.0.0.0/16



Cada Región pueden tener distintas zonas de disponibilidad (AZ), de 2 a n zonas. Dentro de las zonas se crean las subredes que pueden ser públicas o privadas. En las subredes se distribuyen las direcciones IP de la CIDR.

3.1. - Objetos de una VPC

Tablas de rutas, ACLs, Subred y Gateway

3.1.1. - Tablas de rutas

Nos permite indicar como será el tráfico de entrada y salida con unas reglas. La *Main route table* es la tabla por defecto, será la tabla de rutas que heredarán las subredes (Asociación implícita. Si se asgina manualmente es explícita). Las propiedades son:

- Routes.
  - Destination. Es el ámbito donde se podrá acceder.
  - Target. Es el tipo de acceso.
    - Core Network
    - Egress Only Internet Gateway
    - Instance

        - Internet Gateway
        - Network Interface
        - Outpost Local Gateway
        - Peering Connection
        - Trnasit Gateway
        - Virtual Private Gateway
        - Gateway Load Balancer Endpoint
        - Local
    - Propagated.
- Subnet associations
- Edge association
- Route propagation
- Tags

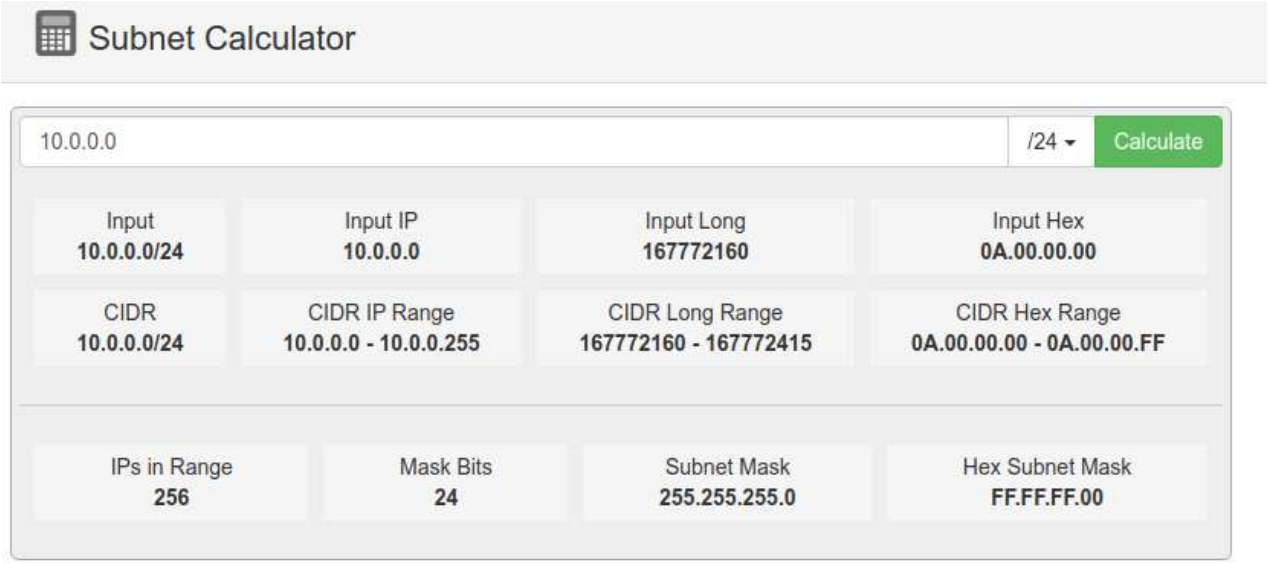
3.1.2. - ACLs

Access Control List. Reglas de entrada y salida para una VPC. La *Main Network ACL* se crea implícita con el VPC.

- Details
- Inbound rules. Permisos del tráfico que entra
    - Rule number. De inferior a superior. La primera regla es la que se aplica. * significa cualquier entrada.
    - Type
    - Protocol
    - Port range
    - Source
    - Allow/deny
- Outbound rules. Permisos del tráfico que sale.
- Subnet associations
- Tags

3.1.3. - Subred

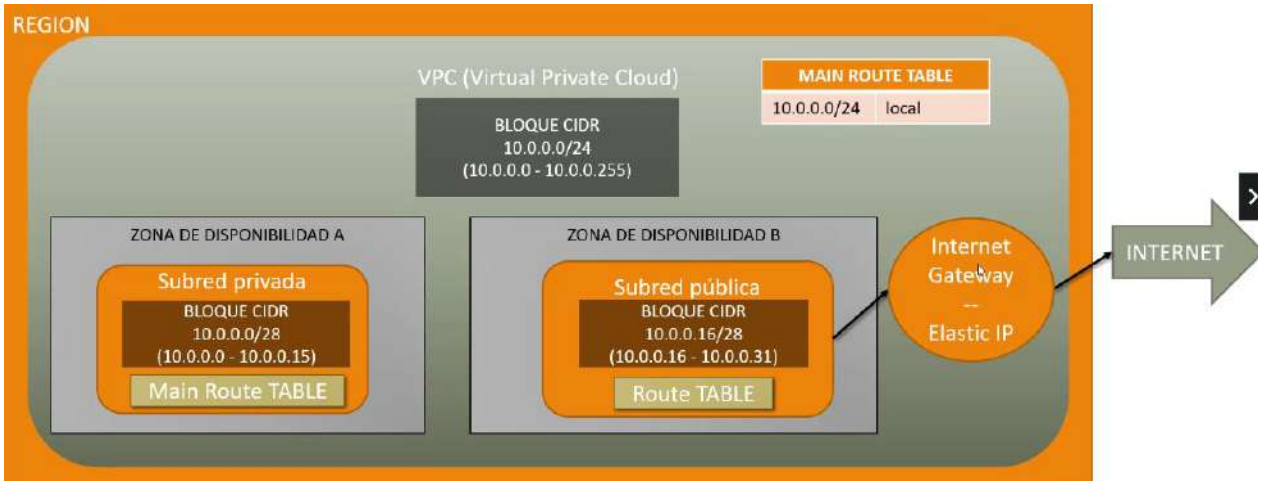
Las subredes se necesitan para trabajar dentro de una VPC y con una ACL. Cuando se crea un objeto en AWS siempre suele pedir una subnet. El VPC es el envoltorio donde se construyen las subredes y es dentro de las subredes donde se construyen las MV, BBDD, etc Con lo que no tiene sentido tener una VPC sin subredes.
Al crearla se asociará al CIDR de la VPC seleccionada. Se debe seleccionar una AZ de la región. Se debe dividir cada subnet con un rango adecuado.
Se puede calcular con las calculadoras online como https://mxtoolbox.com/SubnetCalculator.aspx



Propiedades de una subred:

- Details
- Flow logs. Registros
- Route table. Asigna la main por defecto, de manera implícita
- Network ACL. Asigna la main por defecto, de manera implícita
- CIDR reservations. Reserva de IPs
- Sharing
- Tags

Las subredes **por defecto son privadas**, sin acceso a internet, tan solo pueden acceder a las subredes del mismo VPC.
Para que tenga acceso (sea **pública**) se deben hacer distintas cosas, como añadir el componente IG (Internet Gateway) – Elastic IP.

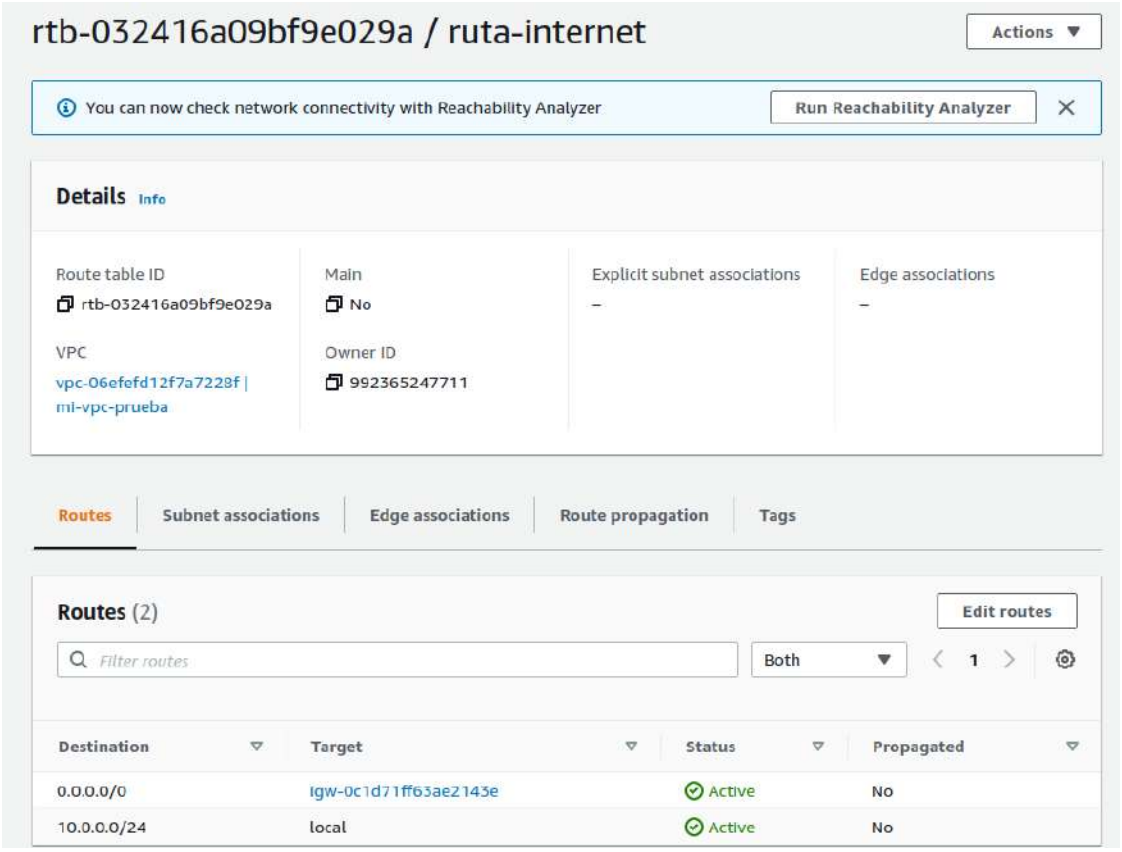


Además, se debe configurar la tabla de rutas (En el diagrama la AZ B). Con esto estará habilitado el tráfico tanto de entrada como de salida.

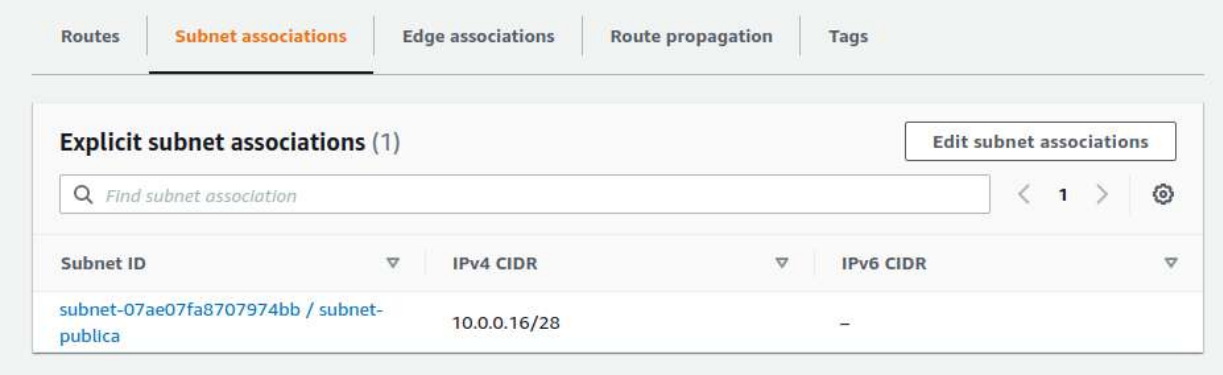
La **Elastic IP** es una IP estática pública para que nunca cambie la IP de entrada de tráfico.

3.1.4. - Gateway

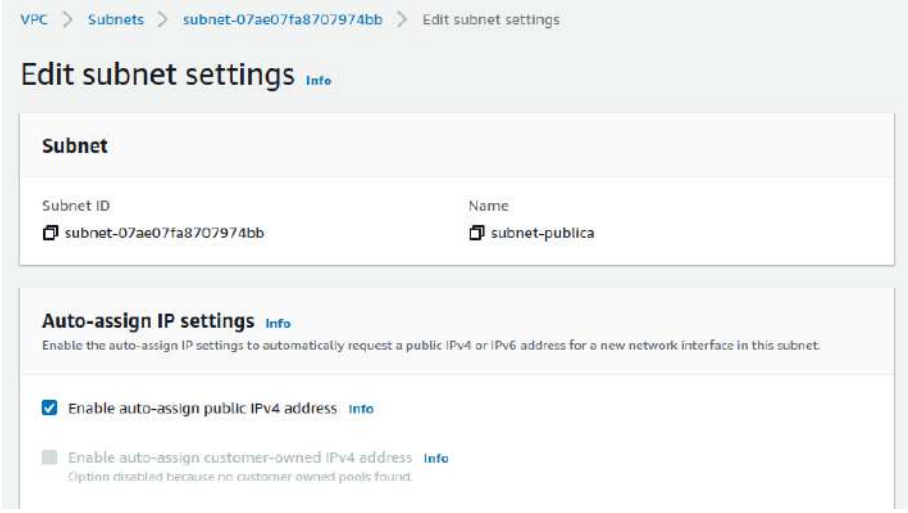
Será la puerta de entrada con la subred que seleccionemos como pública.
Las gateway se asocian a posteriori de su creación. (Attach to VPC) Entonces se podrá asociar a cualquiere subred de ese VPC, esto se hará creando una tabla de rutas.



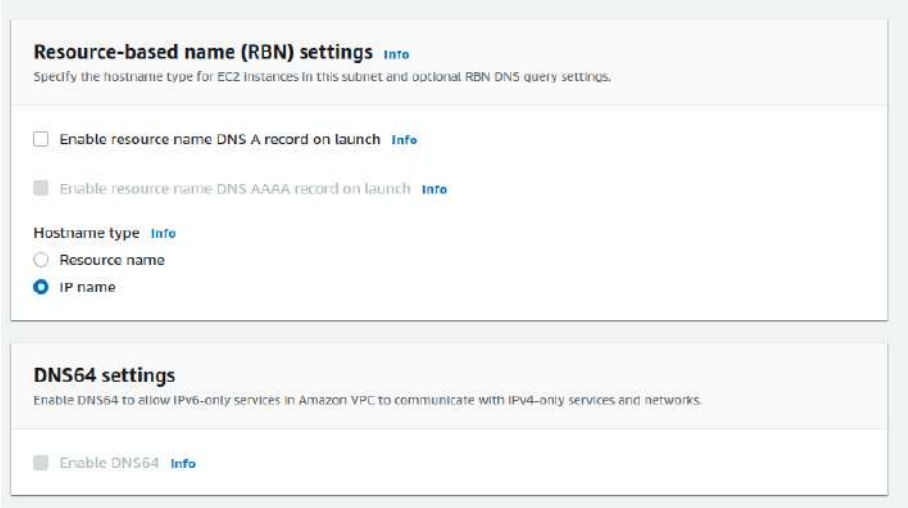
Luego se asigna la tabla de rutas a la subred pública



Además, editando las propiedades de la subnet pública podemos hacer que todas las instancias que se creen dentro de la subnet se le va a asociar una IP pública



Y se puede editar el tipo de hostname:



3.2. - Propiedades de VPC

CIDR, DNS hostname (debería estar activo), DNS resolution (debería estar activo), Middlebox route y tags

3.3. - Asistente para VPC

Se puede clicar «VPC and more» cuando creas una VPC donde tendremos una preview de la creación. Antes, con «Launch VPC Wizard», daba opciones predeterminadas de VPCs.

3.4. - Borrar VPC

En principio, siempre pide la confirmación del borrado con el nombre del objeto que se vaya a borrar.

Cuando borras una VPC intentará borrar todos los objetos asociados, a no ser que estén asociados a otros recursos. En algunas ocasiones, no permite borrar por alguna asociación.

TEMA 4 - Máquinas virtuales y entornos de procesamiento

EC2 es el servicio de MV. Es una de los servicios más conocidos de AWS porque dispone de las instancias y MV necesarias para afrontar cualquier tipo de trabajo en la infraesctructura de AWS.

Se ofrecen bajo distintos tipos de alquiler y de instancias.

EC2 está por debajo de muchos servicios de AWS como los de BBDD, IA, etc.

4.1. - Tipos de instancias

Hay muchos tipos de instancias, con lo que al principio cuesta elegir una. Soporte AWS: https://aws.amazon.com/es/ec2/instance-types/ En la ayuda divide las instancias en grupos y estos en familias de instancias:

- De uso general, Optimizadas para informática, Optimizadas para memoria, Informática acelerada, Optimizadas para almacenamiento

También se puede ver las Características de las instancias y la Medición del rendimiento de las instancias.

El explorador va muy bien para filtrar instancias: https://aws.amazon.com/es/ec2/instance-explorer/

En «EC2 Dashboard» también tenemos una manera de filtrar las instancias. Em esta tabla informátiva se puede ver el precio aproximado. Si marcamos una instancia veremos más detalles:

- Details, Compute, Networking, Storage, Accelators, Pricing

4.2. - Nitro System

Es la infraestructura con una combinación de hardware dedicado y software para que funcionen correctamente los servidores. Se usa como el motor para hacer funcionar las virtualizaciones con distintos componentes (Tarjetas, chip de seguridad, hipervisor, enclaves, TPM, ...). La plataforma subyacente para la última generación de instancias de EC2 que permite a AWS innovar con mayor rapidez, reducir aún más los costos para nuestros clientes y ofrecer más beneficios, como nuevos tipos de instancias y un nivel de seguridad superior.

Recursos del soporte de AWS (Vídeos):

- Video: Beneficios de seguridad de la arquitectura Nitro EC2 (pantalla de inicio)
- Video: Beneficios de seguridad de la arquitectura Nitro EC2 (presentación)
- Video: Análisis profundo de Nitro (presentación)

- Video: Evolución de Nitro System (presentación)
- Blog de Jeff Barr
- Perspectives - AWS Nitro System Support for Previous Generation Instances - James Hamilton

4.3. - Consola EC2

Recursos:

- Instancias corriendo
- Servidores dedicados
- IPs elásticas
- Instancias generales
- Pares de llaves
- Balanceadores de carga
- Placement groups
- Grupos de seguridad
- Capturas de instancias
- Volumenes

Atributos de la cuenta:

- Documentación
- Consola de la VPC
- Encriptación EBS
- Zonas
- Numero de serie de la consola EC2
- Especificaciones por defecto
- Experimientos en consola.

Se puede lanzar una instancia o efectuar una migración.

Aparece la información de eventos programados.

Aparece un resumen del servicio en Service health

También existe una consola global de todas las regiones dentro de la cuenta: «EC2 Global view»

4.4. - Limites en EC2 y VPC

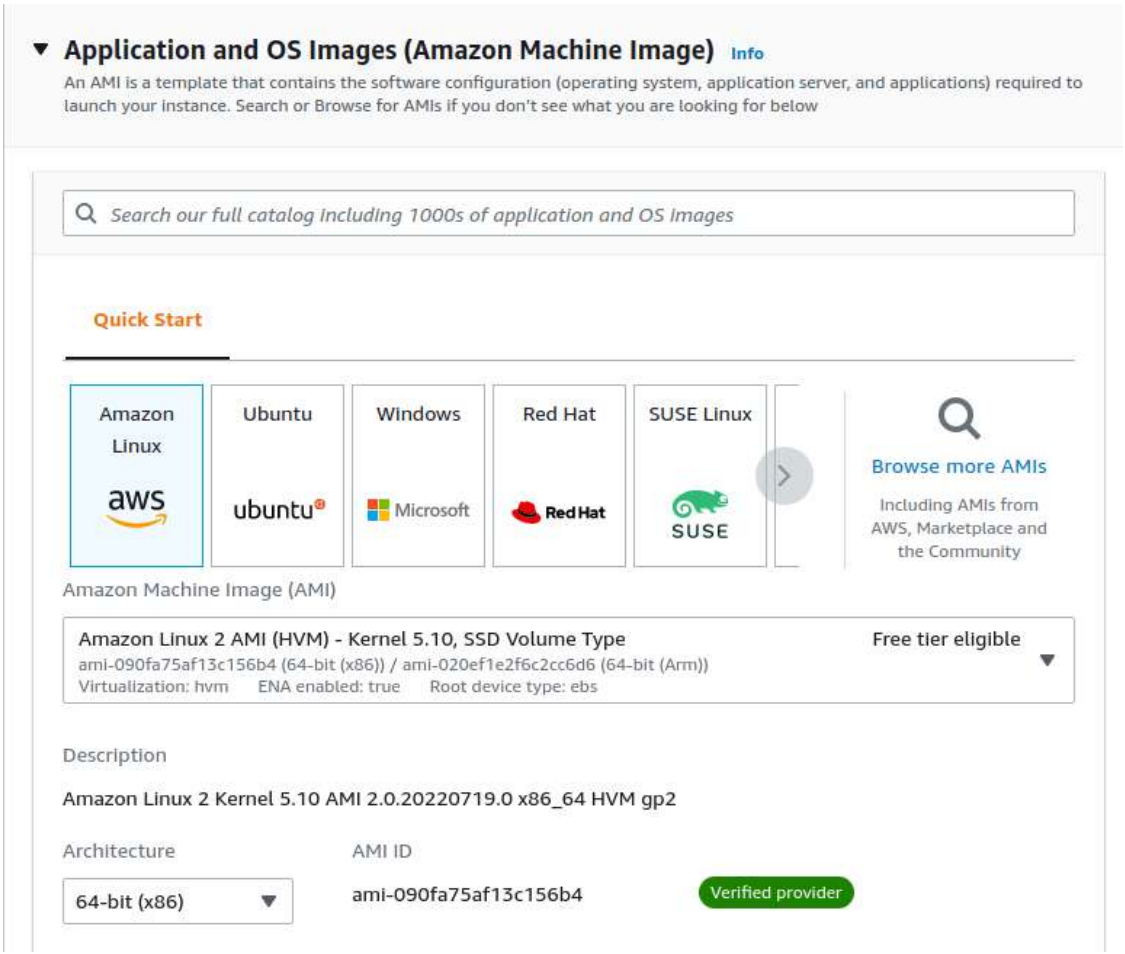
En limits podemos ver los límites actuales del EC2 y el VPC. Si se necesita aumentar algún límite se debe solicitar a AWS con «Request limit increase», que es un servicio de tickets. AES deberá aprobar el nuevo límite.

Se puede calcular los posibles límites con Calculate vCPU limit.

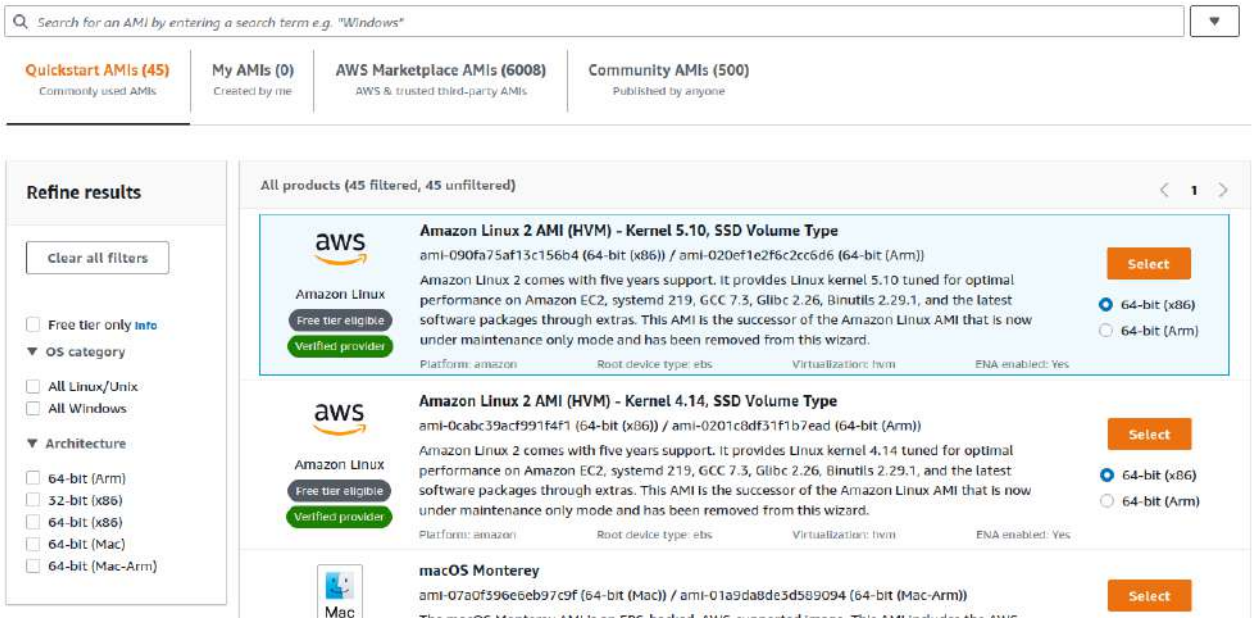
4.5. - Lanzar una instancia

Se puede **etiquetar** cada instancia. Con name será el nombre del objeto. Luego podemos darle un key relacionado con un value. Cada etiqueta se puede extender a otros tipos de recursos.

Una **AMI** es como una imagen ISO que se utilizan para crear las MV.

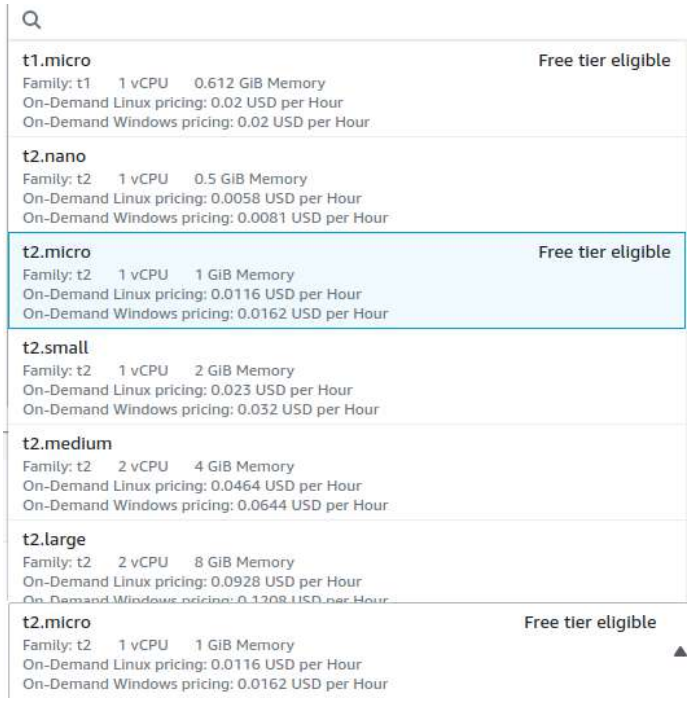


En principio ofrece las más habituales, pero en Browse more AMIs podemos encontrar más, con filtros de búsqueda.



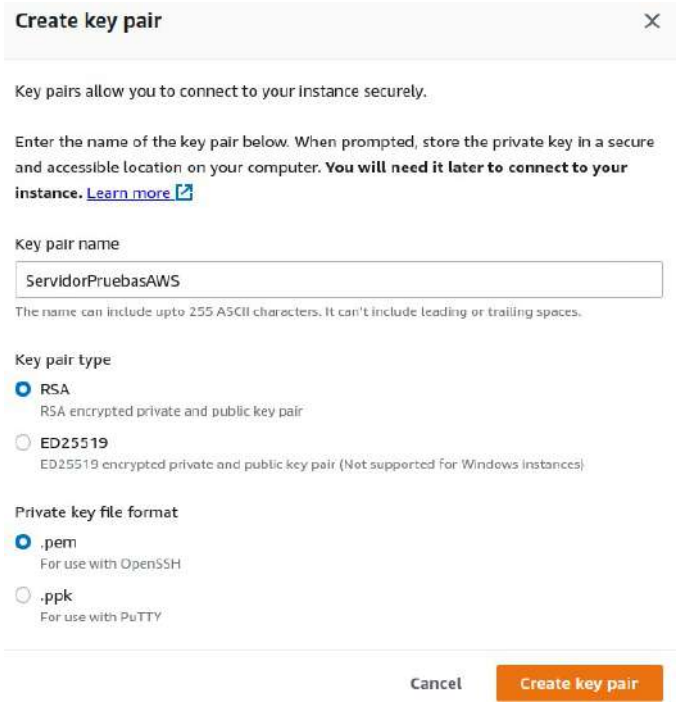
Podemos escoger entre las de inicio rápido, AMIs propias personalizadas, la del mercado de AWS y las de la comunidad.

Después se selecciona el **tipo de MV**:



Se pueden comparar el tipo de MV.

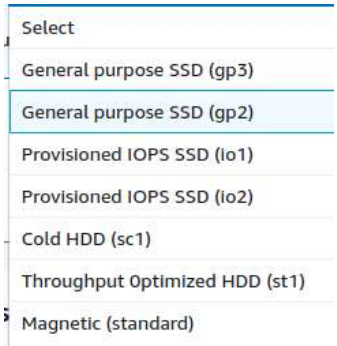
Después deberemos crear unos **pares de llaves** para poder conectarnos mediante ssh.



Después parametramos la **red**. Elegimos el VPC, la subnet, si asigna IP pública y grupos de seguridad.

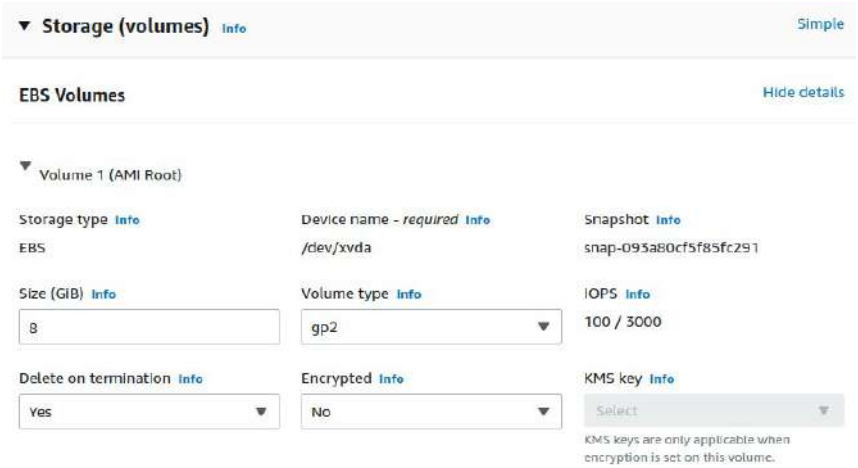
Los **grupos de seguridad** sirven para parametrar el firefox. Lo normal es utilizar el mismo grupo a servidores con el mismo rol.

Ahora el **almacenamiento**, donde podemos escoger la capacidad y el tipo de disco.

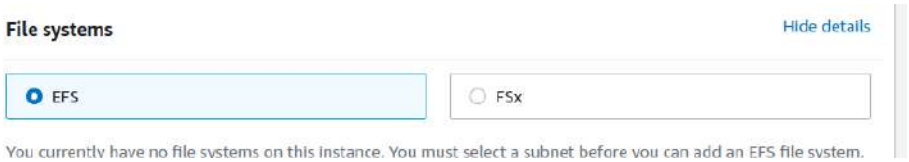


gp es gama media con un precio razonable.
io se usa cuando necesitas un procesamiento de entrada y salida rápido.
sc es el viejo HDD
st es un HDD optimizado
Magnetic es un tipo de disco antiguo.

En avanzado vemos más datos:



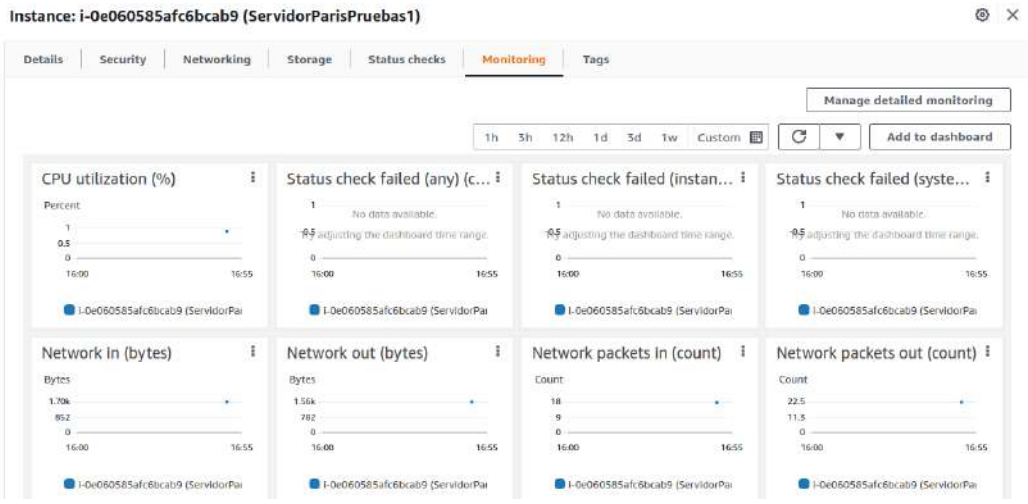
También se puede escoger el **sistema de ficheros**



Por último, en detalles avanzados se puede escoger algunos elementos y parámetros más.

En los **atributos** de la instancia creada encontraremos:

- Detalles, Seguridad, la red, el almacenamiento, el estado, monitorización y sus etiquetas.



Las **monitorizaciones** se pueden añadir en un panel general que se crea en CloudWatch.

Los **Estados** de las instancias pueden ser: Parado, encendida, reiniciando, hibernación y terminada.

Hay una diferencia entre parar/arrancar y reboot. Si paras y arrancas se cambia la IP y el nombre por defecto de la máquina.

Para una **IP estática** se necesita pagar una Elastic IP.

Para conectarse a las instancias desde internet necesitamos la IP pública. Podemos ir en la parte superior a Conect para ver las opciones que tenemos. Las opciones en MV Linux son:

- EC2 Instance Connect – A través de un navegador. No todas las instancias lo permiten.
- Session Manager – Con un agente de conexión
- SSH client – La tradicional
- EC2 serial console -

En las MV de Amazon Linux el usuario suele ser ec2-user. En Debian root. En Centos Centos.

Las opciones en MV Windows son:

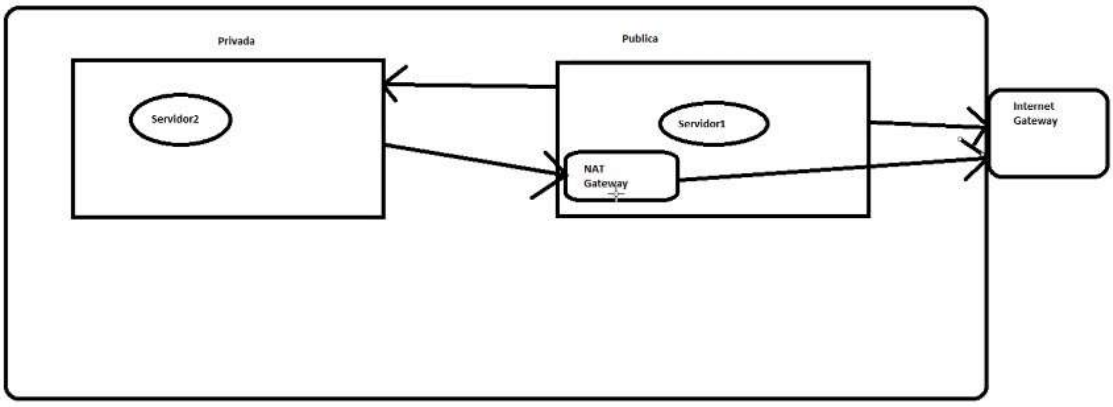
- Session Manager – Con un agente de conexión
- RDP client – La tradicional. Se deberá descargar el archivo rdp.
- EC2 serial console -

Ejemplo de instalación de Apache en Amazon Linux 2:

```
sudo yum update
sudo yum install httpd
sudo systemctl start httpd
sudo systemctl enable httpd
sudo systemctl status httpd
```

4.6. - NAT Gateway. Conectar una red privada a Internet

NAT Gateway es un componente para poder conseguir que una red privada llegue a Internet conectándose de manera segura a Internet Gateway. Hace de pasarela habilitando o deshabilitando el tráfico. De esta manera no se exponen todos los componentes de la red privada.

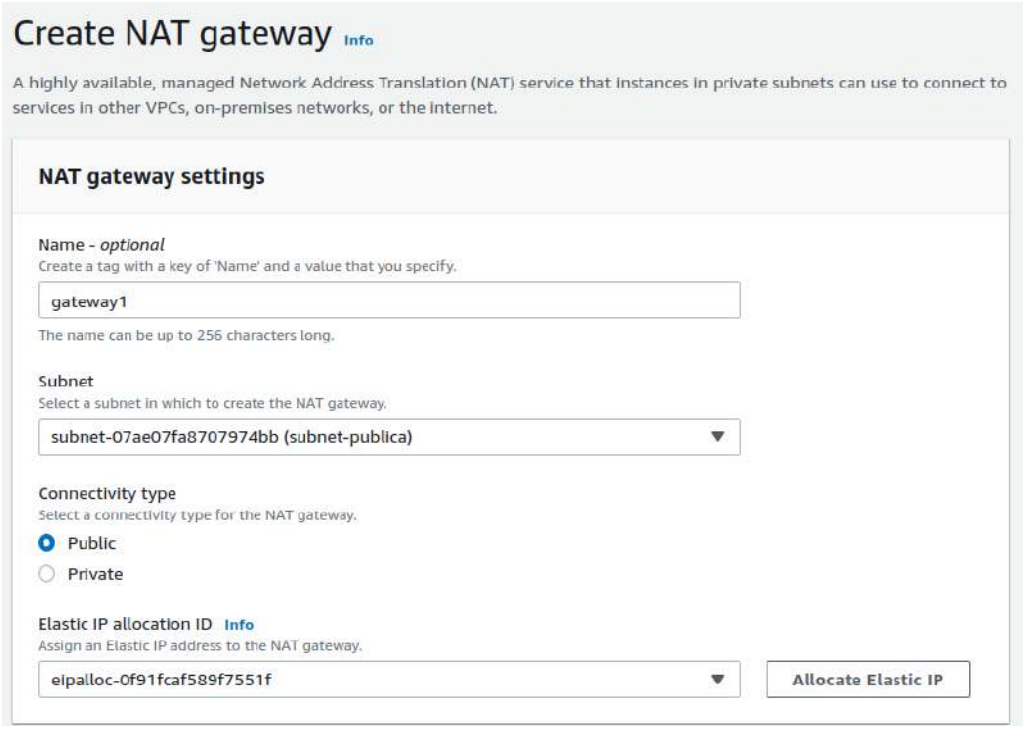


Deberemos generar una Elastic IP. Cuidado que son de pago.

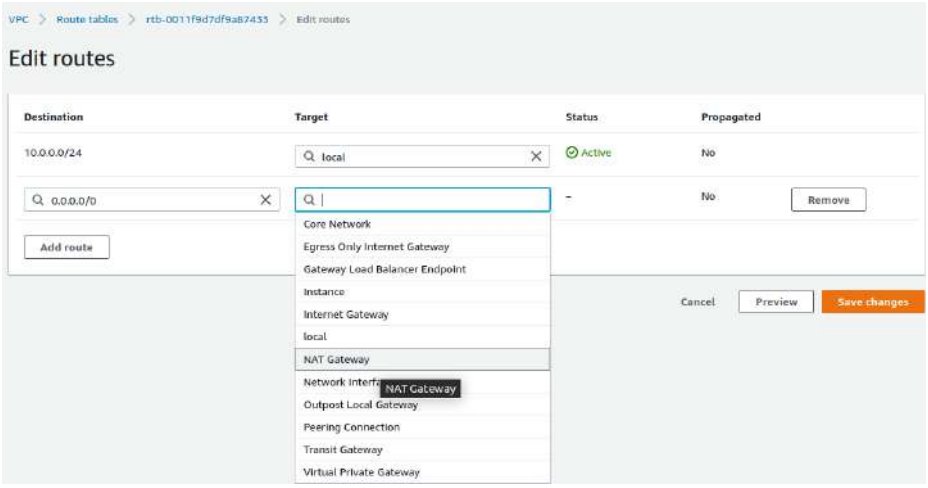
Tendremos una instancia en la subnet pública y otra en la privada. Podremos conectarnos desde la instancia de la subnet pública a la instancia de la subnet privada.

Ahora activamos el NAT Gateway en la instancia de la subnet pública. Para crearla debemos estar en la VPC. La NAT deberá estar en la red pública. Podrá ser pública o privada. Las privadas pueden ser para conectarse entre las VPC.

La crearemos en la red pública.



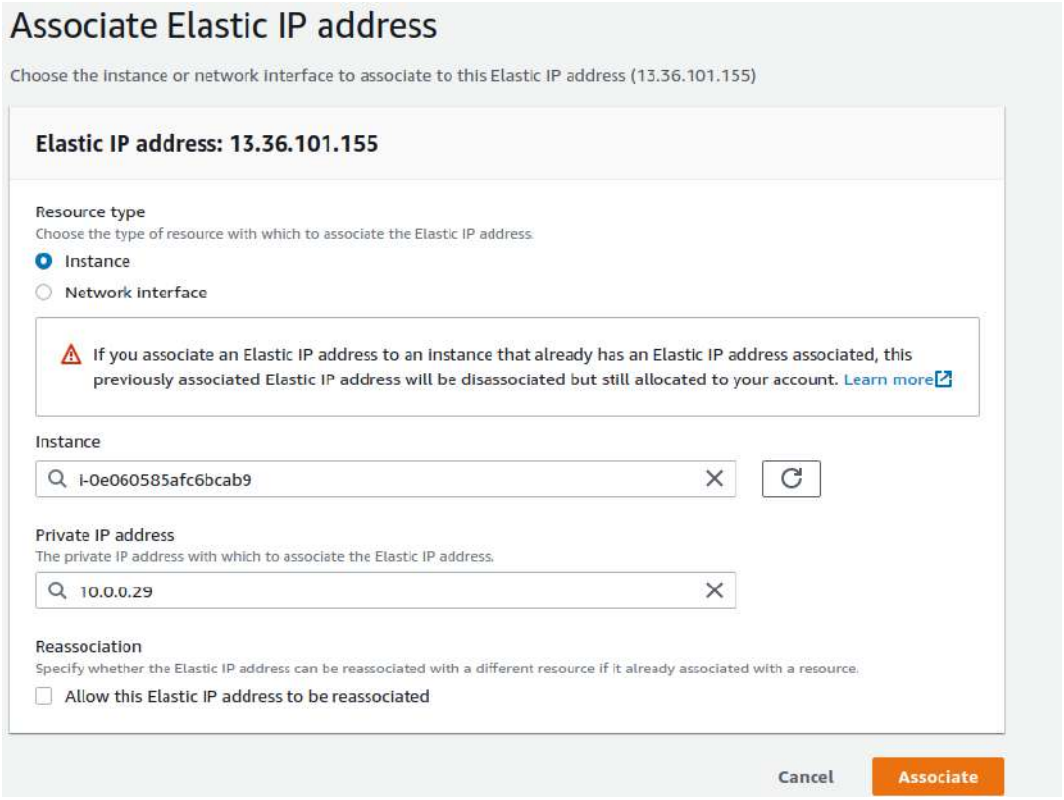
Para asociarla se debe hacer en la tabla de rutas de la subnet privada.



Ahora ya podremos conectarnos como antes y actualizar la instancia de la subnet privada porque ya tendremos internet.

4.7. - Elastic IP

Creamos una en “Allocate Elastic IP address”. Una vez creada la debemos asociar, lo podemos hacer a una Instancia en concreto o a una Network Interfaces. En ambos casos hay que decir la IP privada a la que estará asociada.



Cuando accederemos a la Elastic IP estará directamente conectada con la IP privada.

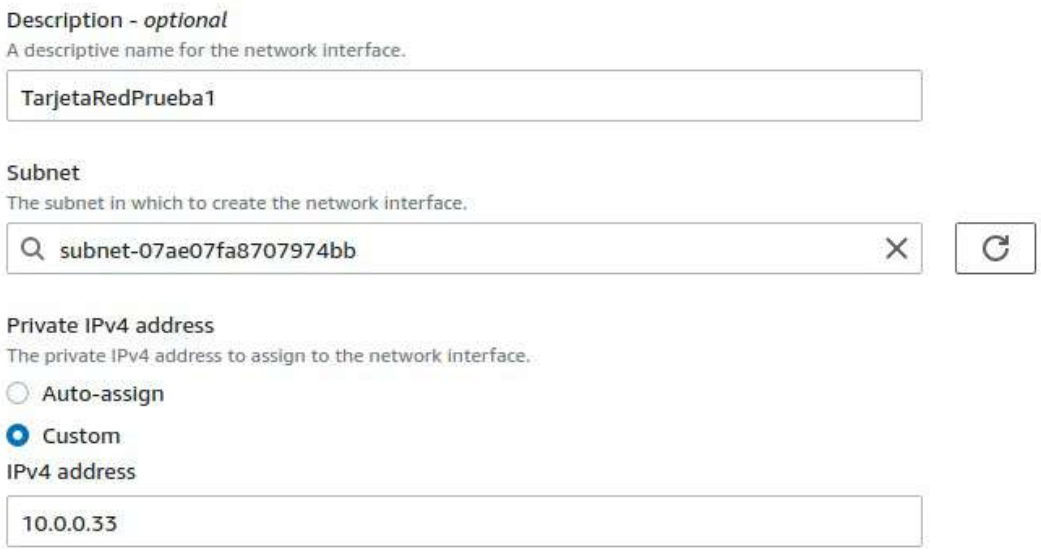
La Elastic IP se puede desasociar. Después se le puede dar un release.

4.8. - Network interface

Suelen empezar por eni (Elastic Network Interfaces). Son tarjetas de red. Se pueden tener varias tarjetas de red, pero según la AMI se pueden tener más o menos. También hay un límite global de las tarjetas que se pueden tener.

Instance type	Maximum network interfaces	Private IPv4 addresses per interface	IPv6 addresses per interface
t2.nano	2	2	2
t2.micro	2	2	2
t2.small	3	4	4
t2.medium	3	6	6
t2.large	3	12	12
t2.xlarge	3	15	15
t2.2xlarge	3	15	15

Para crear una tarjeta de red se crean en Nerwork Interfaces. Crear una nueva, se pondrá el nombre, se escogerá la subnet y se puede escoger la IP o que la asigne automáticamente.



También escogeremos el grupo de seguridad.

Después la seleccionaremos y asociamos con Attach a cualquier instancia de la subnet.

Para desasociar la Tarjeta de red es con Detach.

Una misma interfaces puede asignar más de una IP.

Las Elastic IP se pueden asociar con la interfaz que creemos.

TEMA 5 - EC2 Plantillas

Se pueden crear plantillas para lanzar las mismas instancias.

Se pueden anidar las plantillas, basarse en una plantilla para crear una segunda.

El tipo de instancia se puede seleccionar dando unos mínimos y máximos de CPU y memoria, siempre y cuando se use Spot Fleet, Auto Scale o EC2 Fleet.

Para lanzar una instancia con la plantilla tan solo hay que seleccionarlo.

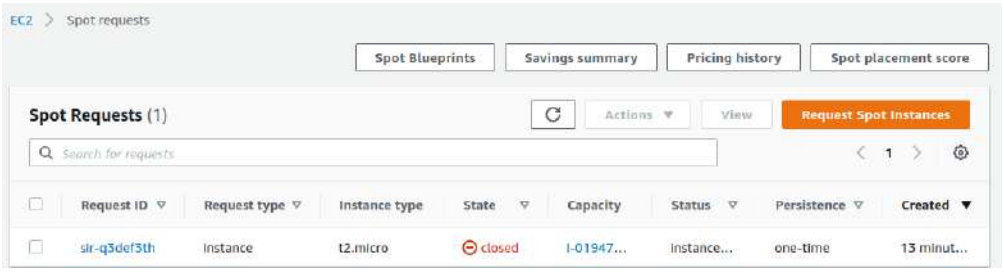
Las plantillas pueden tener versiones modificando la plantilla. Se puede seleccionar una versión como predeterminada. Estas versiones también se pueden eliminar.

Se puede crear un Spot Fleet y un Auto Scaling group

TEMA 6 - EC2 Instancias de tipo SPOT

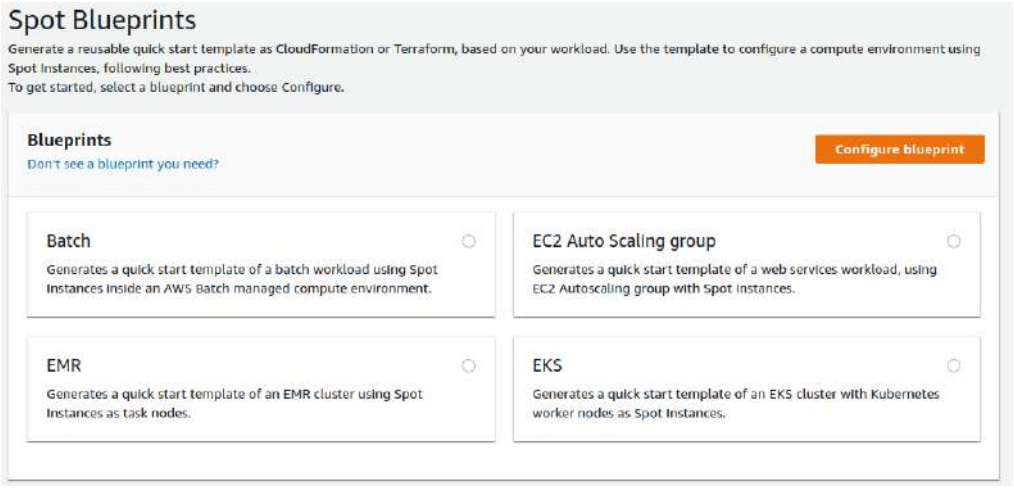
Hasta ahora se ha visto instancias bajo demanda. Las de tipos SPOT son MV que no se pueden parar, se pueden directamente eliminar pero no detener. Son MV más baratas que las normales.

Tiene una sección de Spot Request donde se pueden pedir una Spot. Puede que no la concedan si no hay recursos para ellos.



Si Amazon en un momento dado necesita estos recursos te puede tumbar la máquina.

Con Spot Blueprints son plantilla CloudFormation o Terraform:



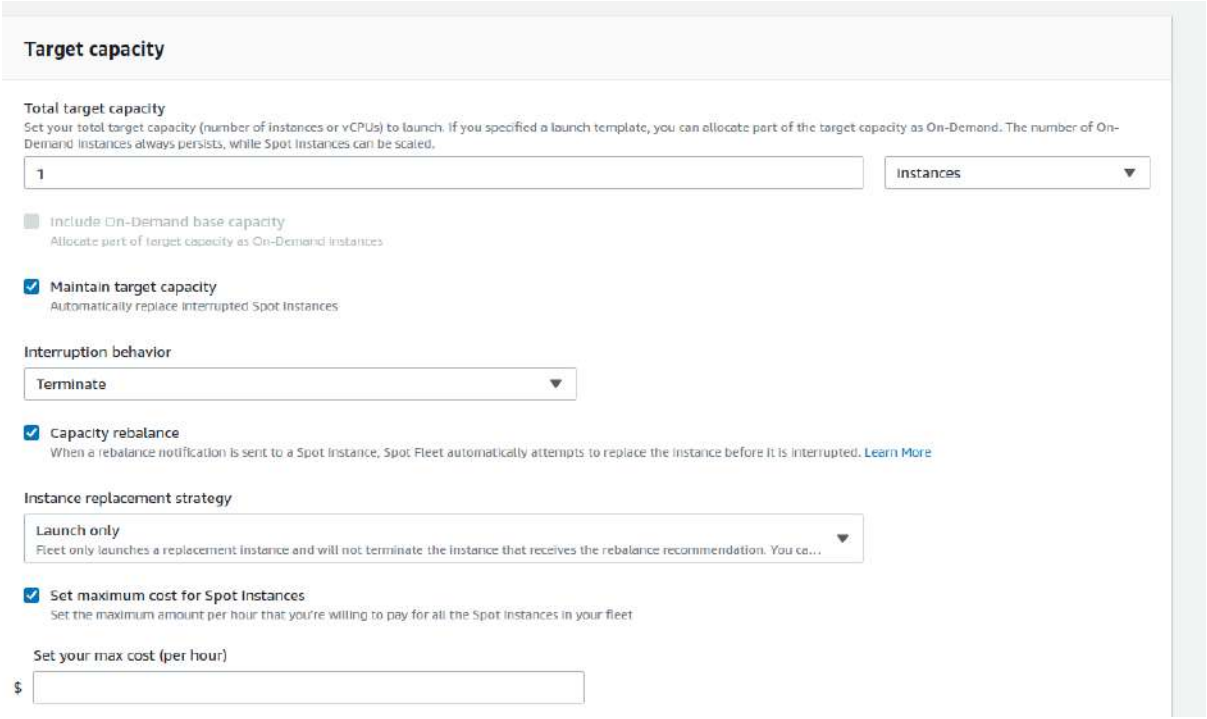
Cuando se solicita un Spot se pueden pedir EBS optimizado para mejorar el rendimiento de los discos, se puede añadir la monitorización de CloudWatch.

El Tenancy es para escoger si la queremos en una máquina compartida o dedicada.

Se puede especificar comandos Linux para que se ejecuten en el momento que se arranque

Un Fleet es una flota de instancias, se pueden crear varias instancias al mismo tiempo.

En la capacidad podemos parametrar algunas caracteristicas



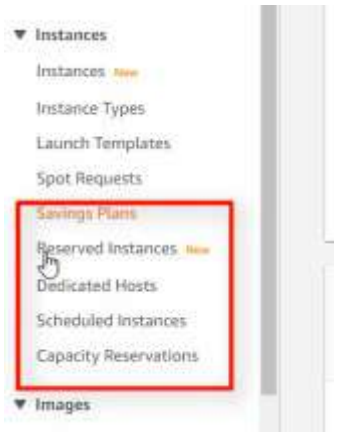
Se puede interrumpit un SPOT

- si los recursos de esas instancias AWS lo necesita para otro servicio que no sea SPOT o que considere prioritario.
- si supera el precio que se ha estimado.

Además de terminate, se puede parar o hibernar. Pero será amazon quien la reinicie cuando tenga la capacidad de recursos o del precio que hemos dicho. Se conservará el disco y demás pero todo lo temporal se pierde.

También existe la opción para la capacidad de balanceo, con la que AWS buscará un sustituto para el disco que se termine. Dos opciones: Launch only o Launch before terminate.

TEMA 7 - EC2 Otros tipos de compras de instancias



7.1. - Saving Plans

Se contrata un plan para ahorrar respecto a las máquinas on-demand. Es un servicio que se llama AWS Cost Management. En la primera visita al servicio no te permite ver las opciones hasta que no pasen 24 horas. Ofrece tres tipos:

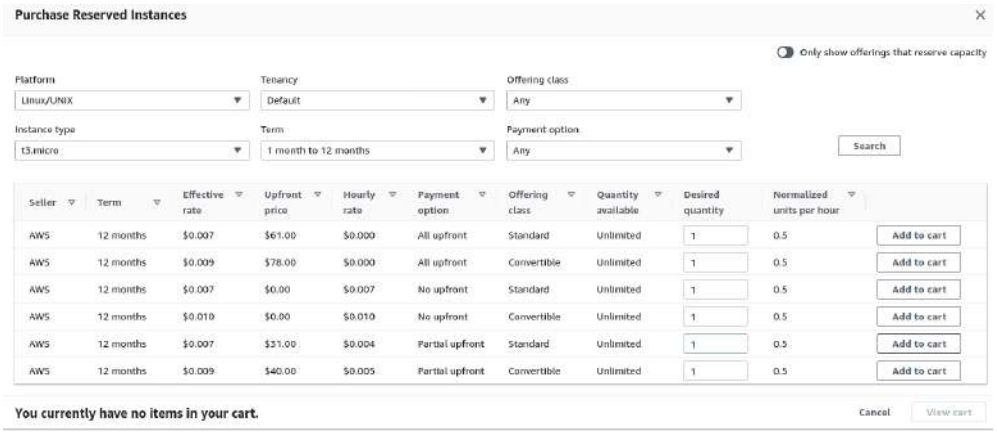
- **Compute**: Los que tienen que ver con el procesamiento. Mayor flexibilidad en EC2, Aws Fargate y AWS Lambda. Este plan es independiente de la família de la instancia, la región, la AZ o tenancy...
- **EC2 Instance**: Solo con EC2 e indicando la máquina y la región.
- **SageMaker**: Son los productos que se utilizan para machine learning.

Los pagos pueden ser por adelantado, parciales o sin adelantar, cada opción tiene sus ventajas.

Se puede hacer estimaciones con «recomendaciones». Estos planos son para entornos estáticos, monolíticos, donde la estructura a penas cambie. Así se hacen cálculos aproximados.

7.2. - Instancias Reservadas

Van bien cuando se necesita un servidor 24 horas los 7 días de la semana.



Se pueden hacer convertible para poder cambiar la máquina, pero solo aumentando. Reservando la instancia AWS te cobra igual si la tienes encendida o apagada.

7.3. - Hosts dedicados

Es lo mismo que las instancias pero en este caso es un host especifico en donde solo estarán las instancias de la misma familia que se creen, sin espacio para otros usuarios.

Se pueden marcar autoemplazamiento. Sirve para alojar aquí cada vez que se lance una instancia de la familia escogida en el host dedicado.

Precios: https://aws.amazon.com/es/ec2/dedicated-hosts/pricing/ Se pueden calcular los precios por hora de cada tipo de host dedicada.

La cuota de los limites en las cuentas, por defecto, es 0. Con lo cual, para crear un host dedicado se debe pedir a AWS que cambie los límites. Se hace en Limits.

Con “Config recording” se guarda un registro del uso de las máquinas y todos sus cambios de configuración. Así se puede tener un registro de las licencias.

7.4. - Programar instancias

Ya no están activas. Es una opción que ha quitado AWS. Eran similares a las instancias reservadas pero se podía configurar un horario concreto de uso.

7.5. - Capacity Reservations

Es una manera de reservar las capacidad de tener un cierto tipo de máquinas dedicadas para contratar. Te cobran como si tuvieras una máquina encendida.

Precios: https://docs.aws.amazon.com/es_es/AWSEC2/latest/UserGuide/capacity-reservations-pricing-billing.html

Esta opción es lógico cuando se planea tener una máquina muy grande, ya que así te aseguras tener esa máquina disponible en el momento que se necesite. Sin necesidad de esperar que AWS la pueda ofrecer.

Las reservas se pueden programar para que terminen en un momento concreto.

Una vez se tiene una reserva de capacidad, se debe indicar al crear la instancia que se va a utilizar dicha reserva, empezando por indicar la subred donde se encuentra y el tipo de máquina. La reserva debe tener configurado autoemplazar cualquier instancia.
O bien, crear la instancia desde la misma reserva.

TEMA 8 - AWS CLI Cliente en modo comando de Amazon

Ayuda: https://aws.amazon.com/es/cli/

Documentación: https://docs.aws.amazon.com/cli/index.html

Referencia de comandos: https://awscli.amazonaws.com/v2/documentation/api/latest/reference/index.html

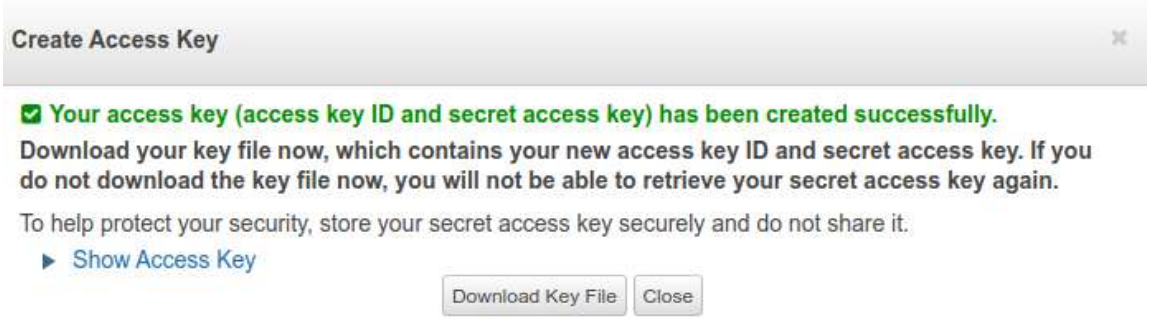
8.1. - Instalación Linux

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

Para ver la versión: aws –version

8.2. - Autenticación

Para autenticarse, en la barra superior, clicando en el nombre de usuario aparece una opción de credenciales: “Security Credentials”. Hay varios tipos de credenciales. Para CLI podemos usar Access Keys que contienen un Access Key ID y la clave. Esta clave es personal e intransferible, se pueden tener 2.



Access Key ID: AKIA6ODMT7TPSMXRMNYK
Secret Access Key: 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

Ahora, en la linea de comandos:

```
aws configure
```

Pedirá 4 datos: El ID de la clave, la clave, la region por defecto y el formato de salido por defecto (text, table, json, etc Por defecto es texto.)

```
administrador@ubuntudocker:~$ aws configure
AWS Access Key ID [None]: AKIA6ODMT7TPSMXRMNYK
AWS Secret Access Key [None]: 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
Default region name [None]: eu-west-3
Default output format [None]:
administrador@ubuntudocker:~$
```

Para probar que funciona podemos ver las instancias:

aws ec2 describe-instances

En la carpeta de usuario se crea la carpeta .aws con dos documentos: config y credentials

```
administrador@ubuntudocker:~$ cat .aws/c
config       credentials
administrador@ubuntudocker:~$ cat .aws/config
[default]
region = eu-west-3
administrador@ubuntudocker:~$ cat .aws/credentials
[default]
aws_access_key_id = AKIA6ODMT7TPSMXRMNYK
aws_secret_access_key = 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
administrador@ubuntudocker:~$
```

8.3. - Ejemplo de comandos concretos

Poniendo help detrás de la opción deseada nos aparecerá ayuda concreta del comando o el subcomando.

Ver la vpc:

aws ec2 describe-vpcs

Formatos con –output:

yaml	Los guiones significan listas.
text	Texto plano
table	En una tabla (En Windows no saca los colores y se debe añadir --color off)

Ver las subnets:

aws describe-subnets

8.4. - Filtros de las filas

Filtrar según los datos que queremos mostrar.

Ejemplo de opciones del subcomando describe-subnets:

[--filters <value>] Filtrar elementos (ver opciones: https://awscli.amazonaws.com/v2/documentation/api/latest/reference/ec2/describe-subnets.html#options) Se debe seguir las opciones de la doc. Ejemplo sintaxis general:

aws ec2 describe-subnets --filters "Name=state, Values=available" "Name=vpc-id, Values=vpc-06efefd12f7a7228f"

[--subnet-ids <value>] Mostrar una subnet o varias por Ids

[--dry-run | --no-dry-run]

[--cli-input-json | --cli-input-yaml]

[--starting-token <value>] A partir de qué elemento se empieza a listar

[--page-size <value>] Determinar el tamaño de la página

[--max-items <value>] Máximo de elementos a visualizar

[--generate-cli-skeleton <value>]

8.5. - Filtros de columnas. Opción Query

Filtrar según los metadatos que queremos mostrar.

El comando es –query

Ejemplos para filtrar subnets:

Filtrar subnets por array (Empieza en 0)

```
aws ec2 describe-subnets --query 'Subnets[1]'
```

Filtrar por rango del array

```
aws ec2 describe-subnets --query 'Subnets[1:3]'
```

Filtrar por key del array. El punto concatena el query

```
aws ec2 describe-subnets --query 'Subnets[*].AvailabilityZone'
```

Filtrar dos campos de dentro del array

```
aws ec2 describe-subnets --query 'Subnets[].[SubnetId, CidrBlock]'
```

En VPC. Filtrar con tres columnas

```
aws ec2 describe-vpcs --query 'Vpcs[].[VpcId,State,IsDefault]' --output table
```

Poner labels para identificar los campos

```
aws ec2 describe-vpcs --query 'Vpcs[].[{"Id de VPC":VpcId},{Estado:State},IsDefault]'
```

Si se pone labels en todos los campos se puede mostrar como tabla:

```
aws ec2 describe-vpcs --query 'Vpcs[].[{"Id de VPC":VpcId},{Estado:State},{"VPC por defecto":IsDefault}]' --output table
```

8.6. - Creando objetos

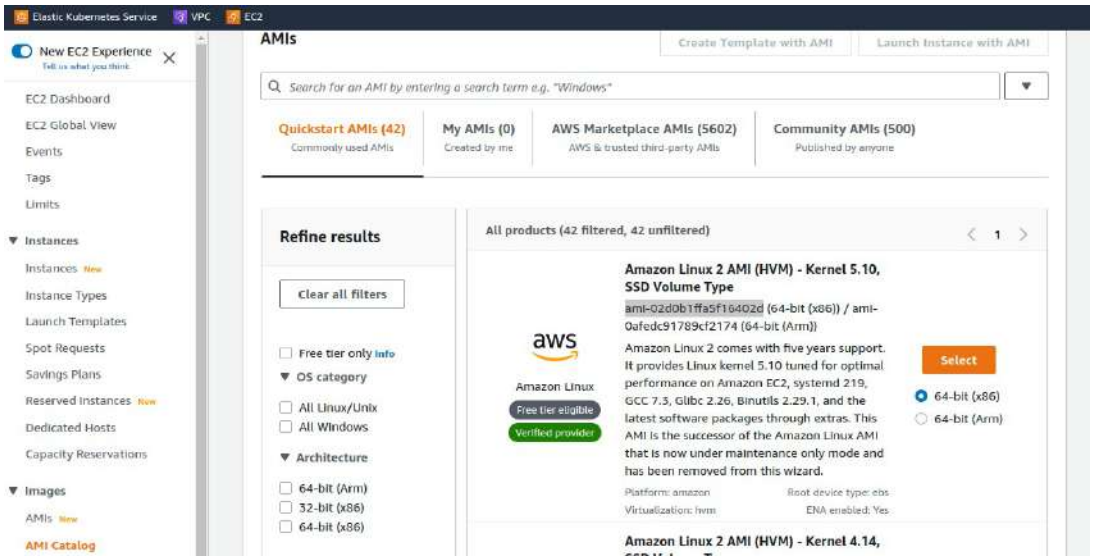
Ayuda: https://awscli.amazonaws.com/v2/documentation/api/latest/reference/ec2/run-instances.html

Para crear una instancia se utiliza el subcomando run-instances que tiene las siguientes opciones:

[--block-device-mappings <value>]
[--image-id <value>]

[--instance-type <value>] Tipo de instancia, familia, que se va a utilizar
[--ipv6-address-count <value>]
[--ipv6-addresses <value>]
[--kernel-id <value>]
[--key-name <value>] Este es el Key Pair que se desea
[--monitoring <value>]
[--placement <value>]
[--ramdisk-id <value>]
[--security-group-ids <value>]
[--security-groups <value>] Grupo de seguridad
[--subnet-id <value>]
[--user-data <value>]
[--additional-info <value>]
[--client-token <value>]
[--disable-api-termination | --enable-api-termination]
[--dry-run | --no-dry-run]
[--ebs-optimized | --no-ebs-optimized]
[--iam-instance-profile <value>]
[--instance-initiated-shutdown-behavior <value>]
[--network-interfaces <value>]
[--private-ip-address <value>]
[--elastic-gpu-specification <value>]
[--elastic-inference-accelerators <value>]
[--tag-specifications <value>]
[--launch-template <value>]
[--instance-market-options <value>]
[--credit-specification <value>]
[--cpu-options <value>]
[--capacity-reservation-specification <value>]
[--hibernation-options <value>]
[--license-specifications <value>]
[--metadata-options <value>]
[--enclave-options <value>]
[--private-dns-name-options <value>]
[--maintenance-options <value>]
[--disable-api-stop | --no-disable-api-stop]
[--count <value>] Se indica el número de instancia que se desea
[--secondary-private-ip-addresses <value>]
[--secondary-private-ip-address-count <value>]
[--associate-public-ip-address | --no-associate-public-ip-address]
[--cli-input-json | --cli-input-yaml]
[--generate-cli-skeleton <value>]

Para crear una instancia se necesita la ID de un AMI (Imágenes de máquina de Amazon).



Copio la ID del AMI que quiera usar: ami-02d0b1ffa5f16402d

Luego se debe indicar el tipo de instancia, la Key Pair y el grupo de seguridad. Además indicamos cuantas instancias queremos. Ejemplo:

aws ec2 run-instances --image-id ami-02d0b1ffa5f16402d --instance-type t3.micro --key-name ServidorPruebasAWS2 --security-groups ServidoresWEB --count 1

Se comprueba en la consola web que se ha creado la instancia. En la consola se puede comprobar con el subcomando describe-instances. Por ejemplo, puede ver el Id, el estado y la fecha de lanzamiento.

aws ec2 describe-instances --query 'Reservations[].Instances[].[{"Id de la instancia":InstanceId},{Estado:State.Name},{"Fecha de lanzamiento":LaunchTime}]' --output table

```
-------------------------------------------------------------------
|                        DescribeInstances                        |
+---------------+-----------------------------+-------------------+
|    Estado     |    Fecha de lanzamiento     |  Id de la instancia|
+---------------+-----------------------------+-------------------+
|               |                             |  i-0e060585afc6bcab9|
|  stopped      |                             |                   |
|               |  2022-08-16T23:49:22+00:00  |                   |
|               |                             |  i-00a3750814c5340e9|
|  shutting-down|                             |                   |
|               |  2022-08-17T14:43:31+00:00  |                   |
|               |                             |  i-0cee5531268123695|
|  stopped      |                             |                   |
|               |  2022-08-12T18:38:37+00:00  |                   |
+---------------+-----------------------------+-------------------+
```

Para saber el nombre de una instancia:

aws ec2 describe-instances --query 'Reservations[].Instances[].[{Nombre:Tags[?Key==`Name`]|[0].Value},{ID:InstanceId},{Estado:State.Name}]' --output table

8.7. - Parar, arrancar y terminar una instancia

Ejemplo de Stop

aws ec2 stop-instances --instance-ids i-00a3750814c5340e9

Ejemplo de start con query

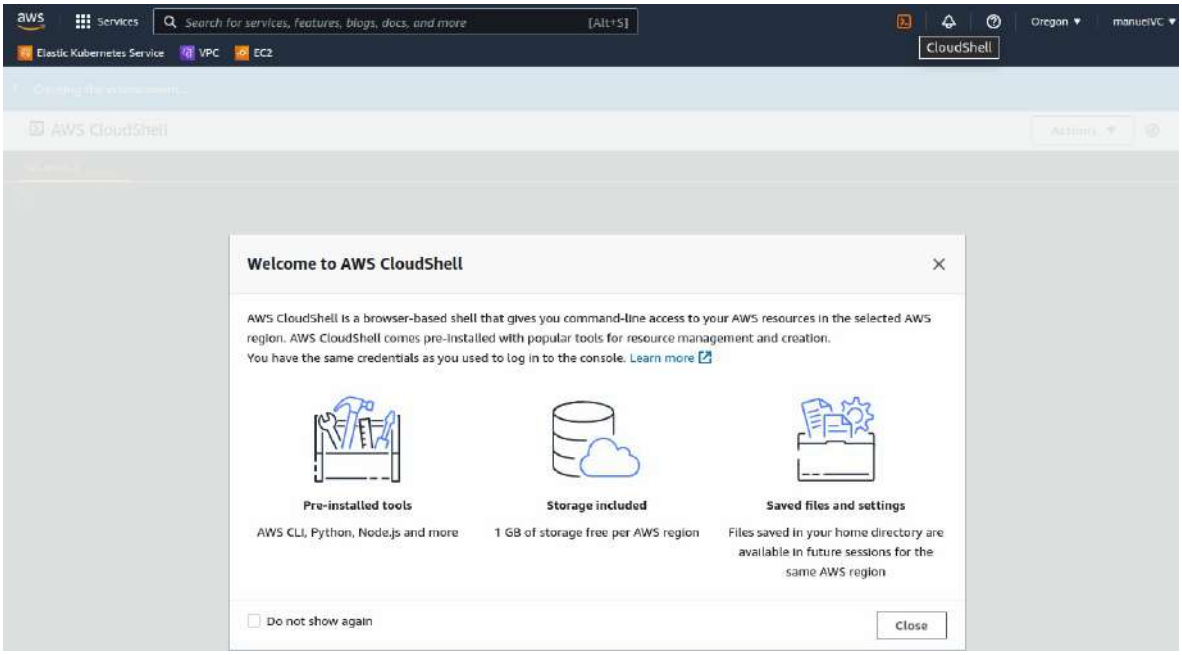
aws ec2 start-instances --instance-ids i-00a3750814c5340e9

Ejemplo de Terminar

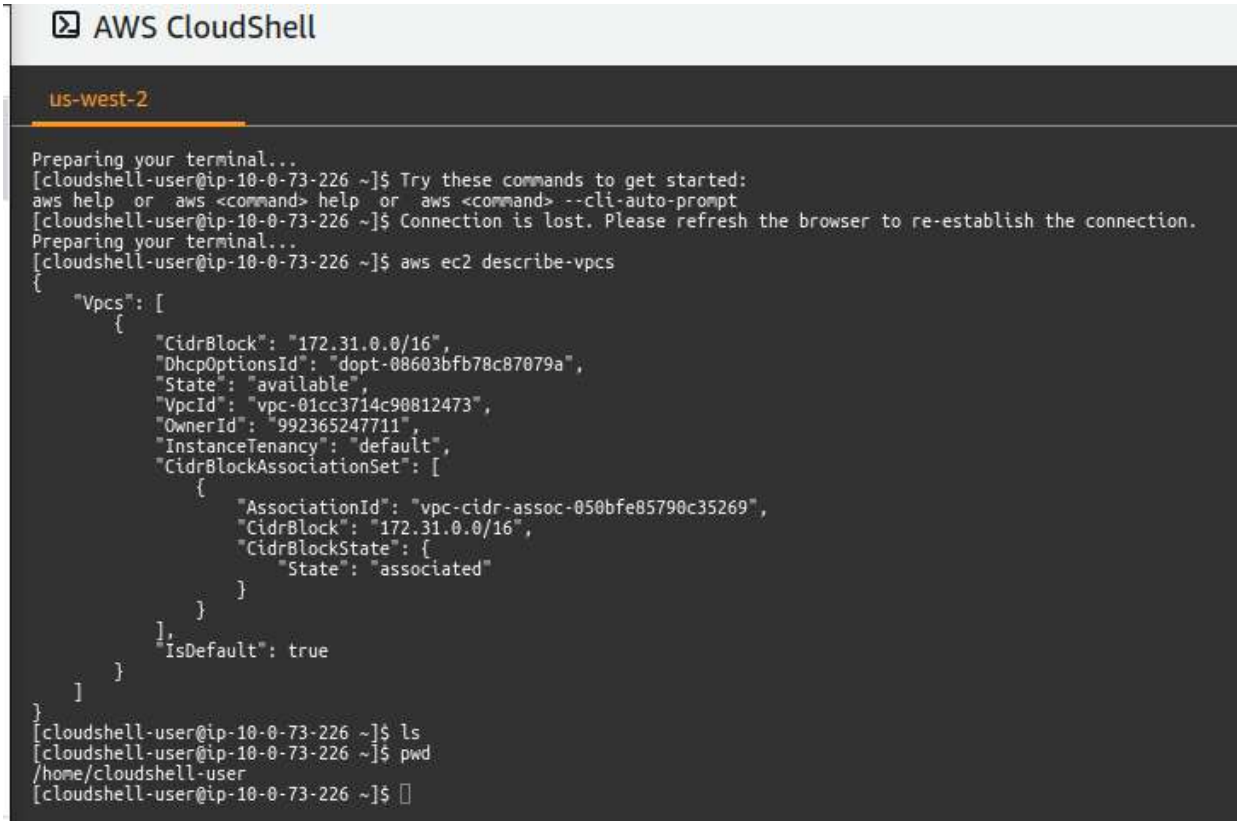
aws ec2 terminate-instances --instance-ids i-00a3750814c5340e9

8.8. - CloudShell

Es un entorno preparado para lanzar comandos, una shel en entorno navegador. No todas las AZ tienen esta opción. Aparece en la barra superior, al lado de las notificaciones



Es útil porque te guarda hasta un 1 GB de información por usuario, con lo cuál se puede abrir en cualquier navegador



TEMA 9 - EC2 Amazon Machine Image AMIs

En el catalogo salen divididas por4 grupos: Quickstart, my AMIs, Marketplace y Community.

Los filtros son muy extensos.

Una vez seleccionada se puede crear una instancia o una plantilla.

9.1. - Crear una AMI personalizada

Se puede crear una AMI como si fuera una ISO, con unos programas concretos que se necesiten desplegar en todas las instancias.

Una vez instaladas las necesidades se selecciona la instancia y en Action/image and templates/create image podemos crear nuestra imagen. Lo que hará es crear un snapshot de la imagen.

Entonces estará en nuestras AMIs. En esta sección se pueden encontrar las imágenes propias, las que se han compartido conmigo y las públicas.

9.2. - Línea de comandos

Para ver las AMIs es con el siguiente comando

aws ec2 describe-images --owner self

La opción --owner self se debe poner para que tan solo salga las AMIs propias y no todas a las que se tiene acceso, que son todas las públicas.

Ejemplo con una query compleja

aws ec2 describe-images --owner self --query 'Images[].[{"ID imagen":ImageId},{"Nombre":Name},{"ID SnapShot":BlockDeviceMappings[].Ebs.SnapshotId}]'

```
administrador@ubuntudocker:~$ aws ec2 describe-images --owner self --query 'Images[].[{"ID
 imagen":ImageId},{"Nombre":Name},{"ID SnapShot":BlockDeviceMappings[].Ebs.SnapshotId}]'
[
    [
        {
            "ID imagen": "ami-06a2ad98d6849aa2f"
        },
        {
            "Nombre": "EntornoLinuxDesarrolloWeb"
        },
        {
            "ID SnapShot": [
                "snap-0789fa5d444a32b64"
            ]
        }
    ]
]
```

Crear una AMI desde una instancia que este corriendo. Primero hay que examinar la ID de la instancia de la que se quiere crear el AMI.

aws ec2 describe-instances

Se podrá filtrar con --query o --filter para localizar la instancia concreta. Luego hacemos el comando:

aws ec2 create-image --instance-id i-0e060585afc6bcab9 --name AMIpersonalizadaConWeb

Para comprobar si es correcto con query

aws ec2 describe-images --owner self --query "Images[].[Name,ImageId,BlockDeviceMappings[].Ebs.SnapshotId,State]"

```
administrador@ubuntudocker:~$ aws ec2 describe-images --owner self --query "Images[].[Name,ImageId
,BlockDeviceMappings[].Ebs.SnapshotId,State]"
[
    [
        "AMIpersonalizadaConWeb",
        "ami-05c1cbc05794e8801",
        [
            "snap-022a0c4ecbac3a1eb"
        ],
        "available"
    ],
    [
        "EntornoLinuxDesarrolloWeb",
        "ami-06a2ad98d6849aa2f",
        [
            "snap-0789fa5d444a32b64"
        ],
        "available"
    ]
]
```

9.3. - MarketPlace

En el buscador de servicios podemos encontrar “AWS Marketplace Subscriptions” para gestionar las suscripciones a las AMIs en la que nos hemos suscrito, encontrar AMIs,

Cada producto tiene su página de suscripción con los detalles del producto. Hay curiosidades como bitname que ofrece Wordpress de manera gratuita pero en la suscripción te avisa que AWS te va a cobrar por la infraestructura, la instancia.

TEMA 10 - EC2 Trabajar con volumenes EBS (Elastic Bloc Store)

Un tipo de almacenamiento, un disco virtual que se asocia a una instancia para poder trabajar con él. Para crear un disco en una MV tan solo hay que añadir un volumen EBS. Hay que tener en cuenta cuando se crea una instancia si queremos que el volumen se llame igual tenemos que extender el tag al volumen.

En los volumenes se pueden ver los detalles, el Status Check, la monitorización de algunos parámetros y los tags.

Para añadir un volumen a una instancia primero iremos a “Volumes” y lo crearemos.

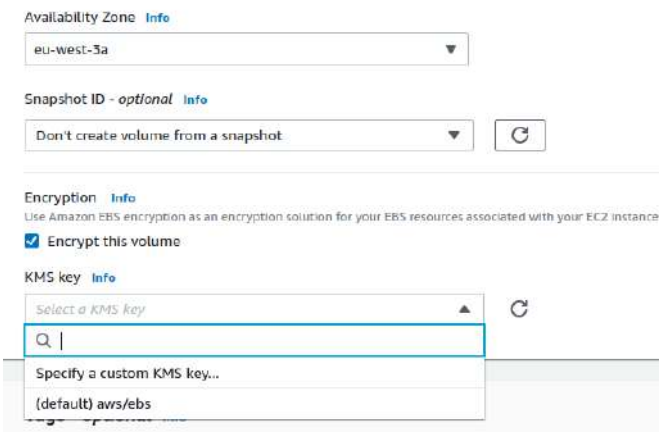
Tipos de discos:

- General purpose SSD. (gp) Los más habituales.
- Provisioned IOPS SSD (io) – Con latencia baja, es para trabajos de mucho tráfico de entrada y salida.
- Throughput Optimized HDD (st) – Disco magnetico de bajo coste. Para procesos secuenciales.
- Cold HDD (sc) – Disco magnetico de bajo coste. Para datos cold, que no tenga consultas. Como para backups o datos que se quieran guardar a largo plazo.
- Magnetic - Disco magnetico de bajo coste

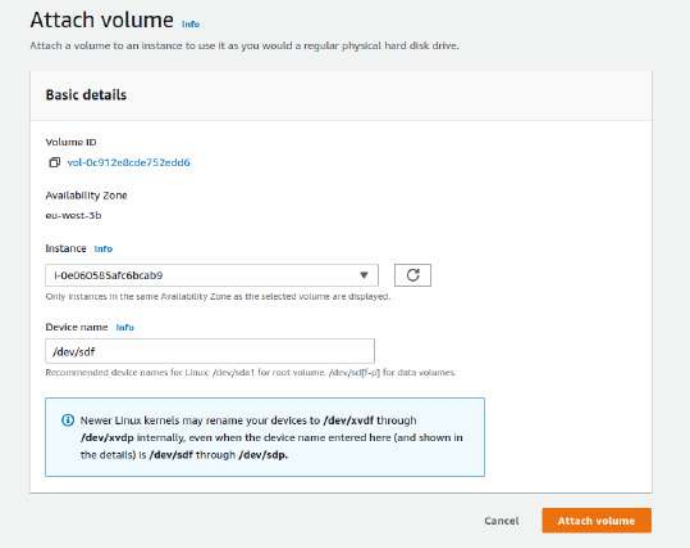
AWS da el dato IOPS, que es el número solicitado de operaciones de E/S por segundo que el volumen puede soportar.

Hay que asegurarse que el volumen está en la misma AZ que la instancia a la que le queremos asociar.

Se puede crear un volumen haciendo una copia de un snapshot. Además, también se puede encriptar.



Para asociar el disco se debe hacer un “Attach volume” en el menú de Actions con el disco seleccionado. Seleccionamos la instancia y nos dará un nombre por defecto.



Una vez asociado el disco habrá que particionarlo, formatearlo y montarlo en el SO.

10.1. - Particionar, formatear y montar volumen en Linux

Entramos por ssh en la terminal y vemos los disco montados con

```
df -h
```

Y podemos ver los discos SSD disponibles con

```
ls -l /dev/sd*
```

Primero debemos particionarlo usando fdisk

```
sudo fdisk /dev/sdf
```

Entrará en un prompt de fdisk del disco y podemos ver los posibles comandos con la ayuda con m:

```
DOS (MBR)
  a   toggle a bootable flag
  b   edit nested BSD disklabel
  c   toggle the dos compatibility flag

Generic
  d   delete a partition
  F   list free unpartitioned space
  l   list known partition types
  n   add a new partition
  p   print the partition table
  t   change a partition type
```

v verify the partition table
i print information about a partition

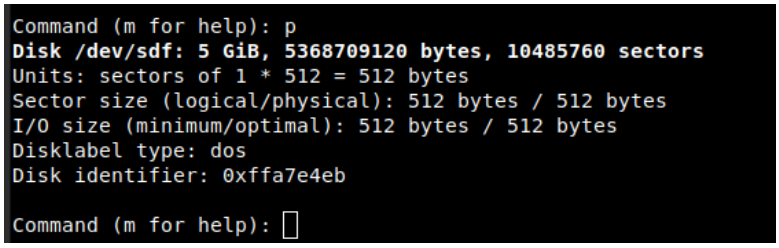
Misc
m print this menu
u change display/entry units
x extra functionality (experts only)

Script
I load disk layout from sfdisk script file
O dump disk layout to sfdisk script file

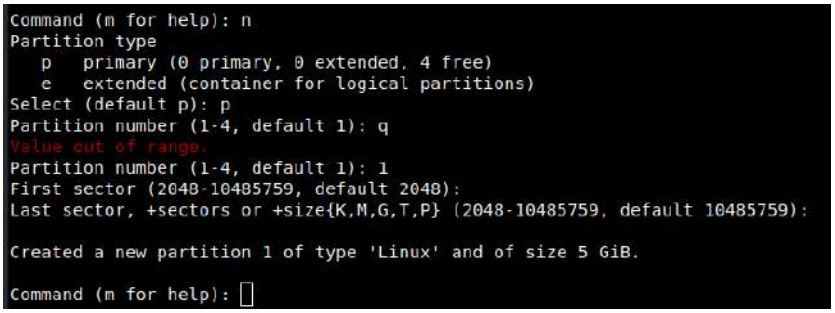
Save & Exit
w write table to disk and exit
q quit without saving changes

Create a new label
g create a new empty GPT partition table
G create a new empty SGI (IRIX) partition table
o create a new empty DOS partition table
s create a new empty Sun partition table

Con p veremos particiones pero en este caso no tiene



Con n crearemos una nueva partición y le decimos que es Primary, la partición 1, que empiece al principio y acabe al final (Los parámetros que vienen por defecto)



Ahora, dándole a la p para que enseñe las particiones se verá la creada

```
Command (m for help): p
Disk /dev/sdf: 5 GiB, 5368709120 bytes, 10485760 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0xffa7e4eb

Device     Boot Start      End  Sectors Size Id Type
/dev/sdf1        2048 10485759 10483712   5G 83 Linux

Command (m for help):
```

Con w guardaremos la tabla de particiones y sale del prompt. Podemos ver la partición con

ls -l /dev/sd*

```
Command (m for help): w
The partition table has been altered.
Calling ioctl() to re-read partition table.
Syncing disks.

[ec2-user@ip-10-0-0-29 ~]$ ls -l /dev/sd*
lrwxrwxrwx 1 root root 4 Aug 19 16:26 /dev/sda -> xvda
lrwxrwxrwx 1 root root 5 Aug 19 16:26 /dev/sda1 -> xvda1
lrwxrwxrwx 1 root root 4 Aug 19 16:55 /dev/sdf -> xvdf
lrwxrwxrwx 1 root root 5 Aug 19 16:55 /dev/sdf1 -> xvdf1
```

Ahora hay que formatearlo con mkfs.

```
[ec2-user@ip-10-0-0-29 ~]$ mkfs
mkfs         mkfs.ext2    mkfs.ext4    mkfs.minix   mkfs.vfat
mkfs.cramfs  mkfs.ext3    mkfs.fat     mkfs.msdos   mkfs.xfs
```

Usaremos el formato xfs, que dice que es el más moderno ¿? Es el sistema de ficheros que suele usar RHEL https://es.wikipedia.org/wiki/XFS No parece tener mucha diferencia con Ext.

mkfs.xfs /dev/sdf1

```
[ec2-user@ip-10-0-0-29 ~]$ sudo mkfs.xfs /dev/sdf1
meta-data=/dev/sdf1              isize=512    agcount=4, agsize=327616 blks
         =                       sectsz=512   attr=2, projid32bit=1
         =                       crc=1        finobt=1, sparse=0
data     =                       bsize=4096   blocks=1310464, imaxpct=25
         =                       sunit=0      swidth=0 blks
naming   =version 2              bsize=4096   ascii-ci=0 ftype=1
log      =internal log           bsize=4096   blocks=2560, version=2
         =                       sectsz=512   sunit=0 blks, lazy-count=1
realtime =none                   extsz=4096   blocks=0, rtextents=0
[ec2-user@ip-10-0-0-29 ~]$
```

Ahora tenemos que montar el disco con

mkdir /mnt/disco2

mount /dev/sdf1 /mnt/disco2

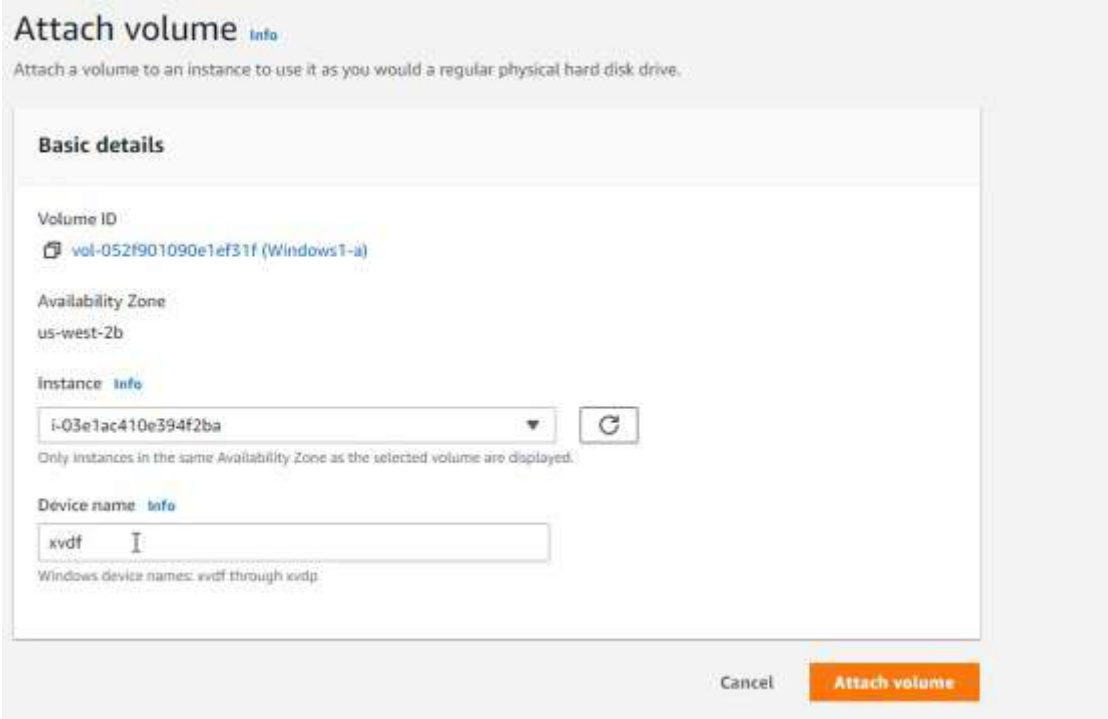
Y con df veremos el disco montado

```
[ec2-user@ip-10-0-0-29 ~]$ sudo mkdir /mnt/disco2
[ec2-user@ip-10-0-0-29 ~]$ sudo mount /dev/sdf1 /mnt/disco2
[ec2-user@ip-10-0-0-29 ~]$ df -h
Filesystem      Size  Used Avail Use% Mounted on
devtmpfs        474M     0  474M   0% /dev
tmpfs           483M     0  483M   0% /dev/shm
tmpfs           483M  476K  483M   1% /run
tmpfs           483M     0  483M   0% /sys/fs/cgroup
/dev/xvda1      8.0G  1.9G  6.2G  24% /
tmpfs            97M     0   97M   0% /run/user/1000
tmpfs            97M     0   97M   0% /run/user/0
/dev/xvdf1      5.0G   38M  5.0G   1% /mnt/disco2
```

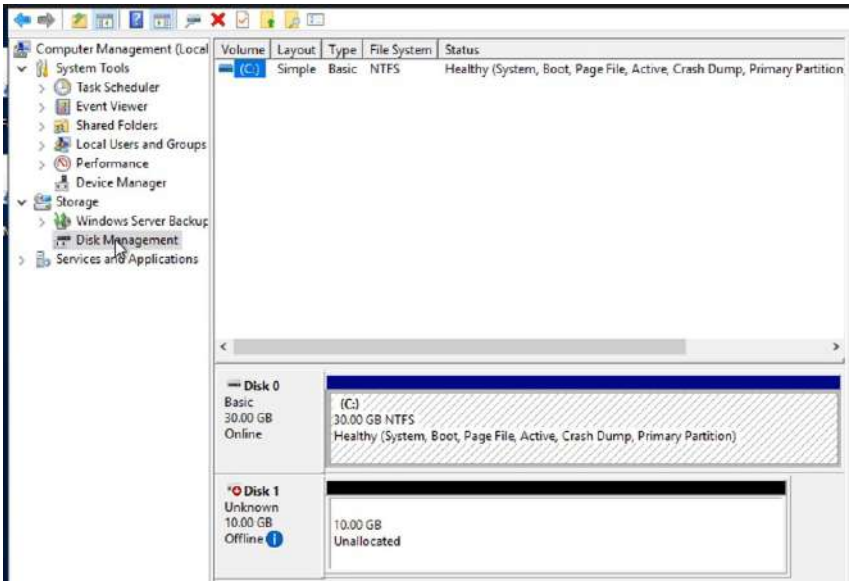
Ahora ya podremos trabajar con el disco.

10.2. - Particionar, formatear y montar volumen en Windows

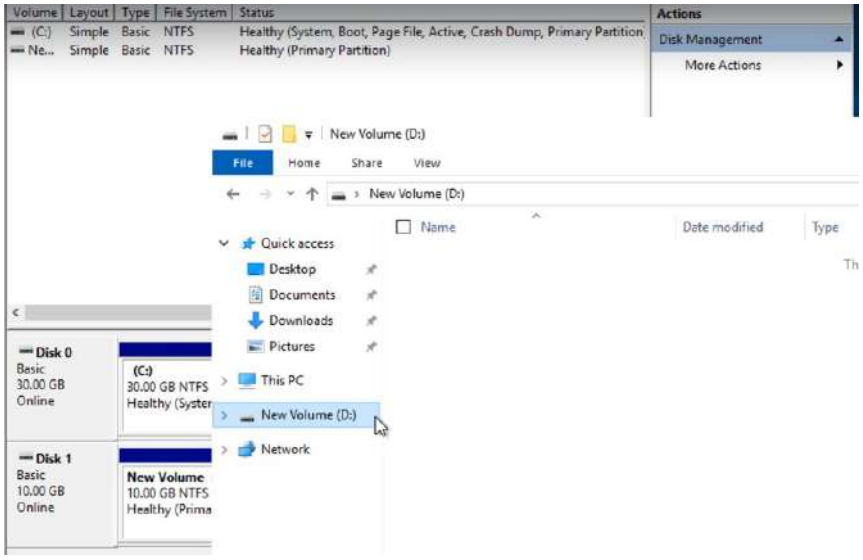
Cuando se asocia se puede poner la etiqueta en Device name, que es diferente a la de Linux



Entramos en Computer Management (Administrador del servidor), en Disk Managenemt (Administrador de disco) y veremos el disco como desconocido.

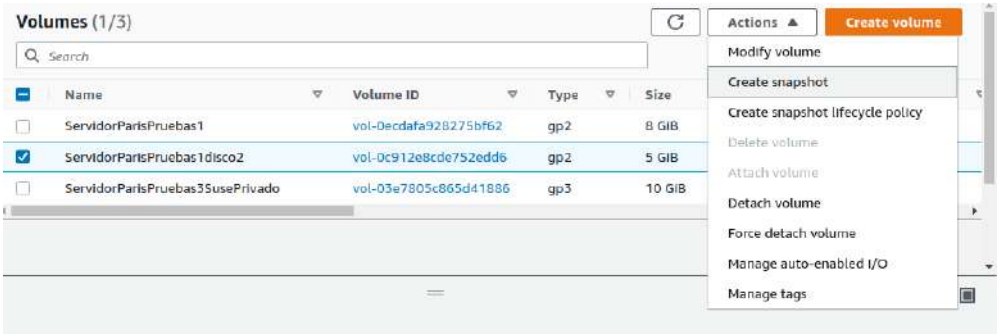


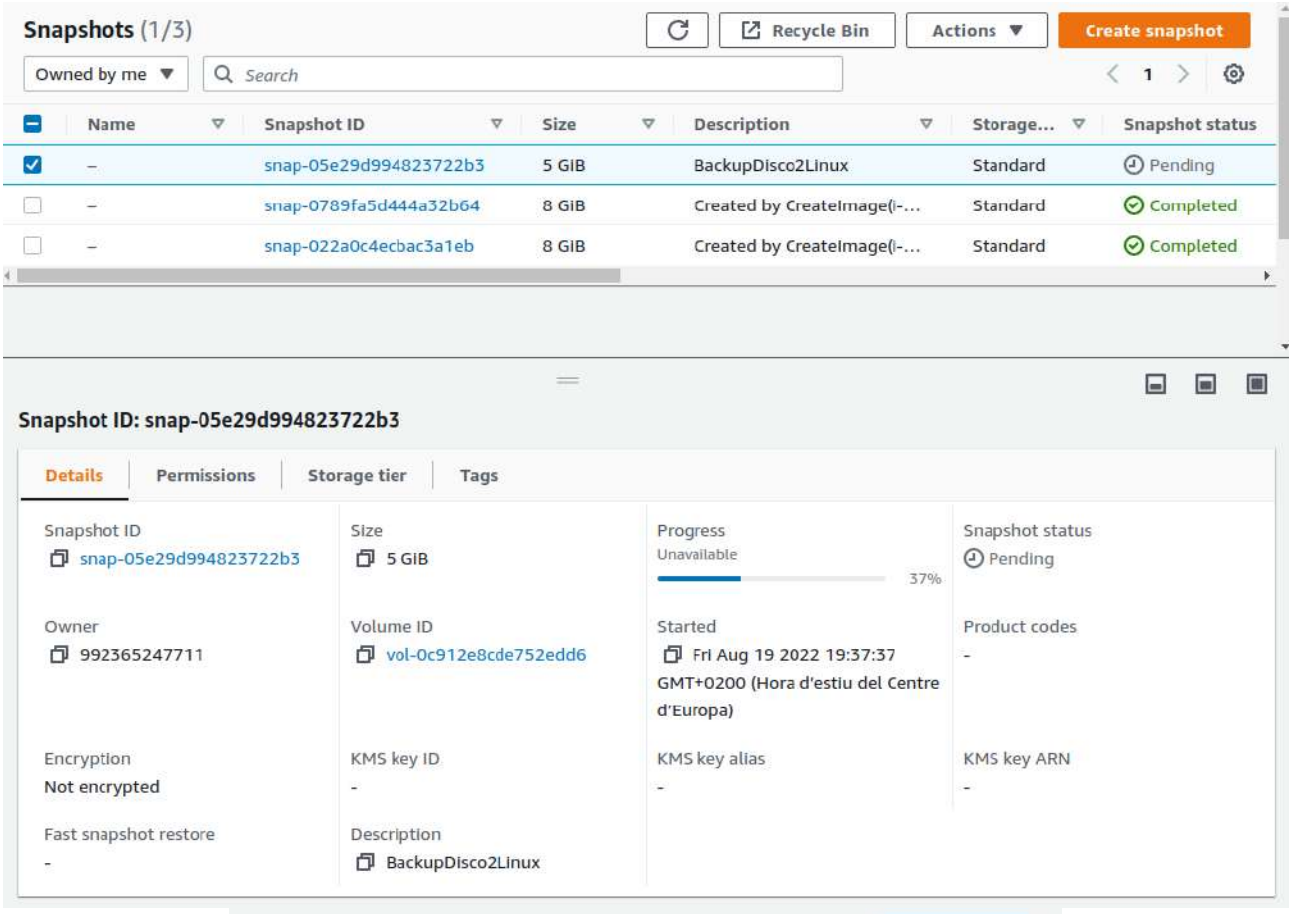
Clicamos con el secundario, lo ponemos online. Secundario de nuevo y se inicializa con MBR. Y luego creamos un volumen simple con el tamaño por defecto completo, con la letra que queramos y formato NTFS y la etiqueta que queramos.



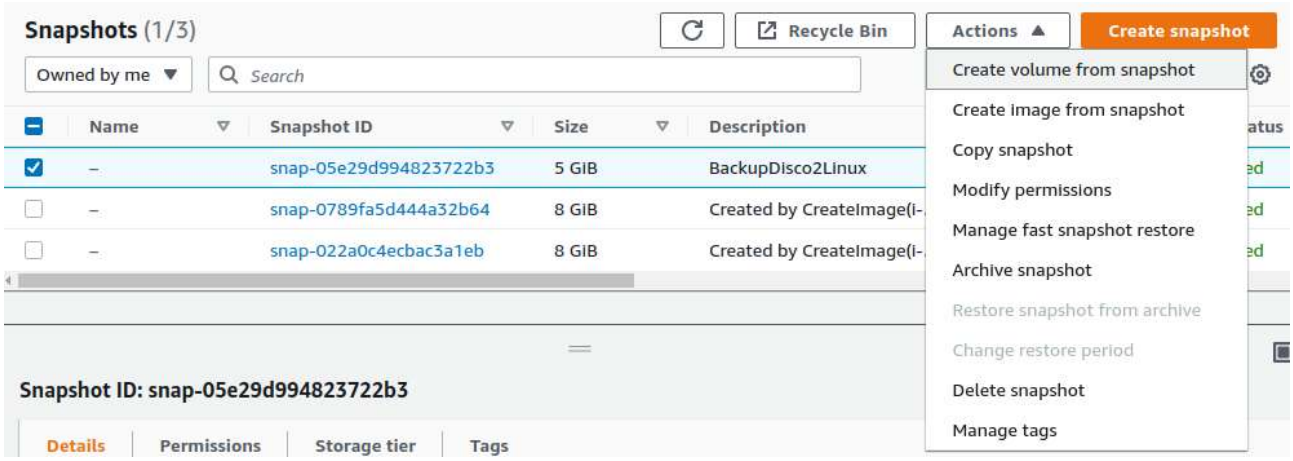
10.3. - Snapshot de volumen

Guardará el disco tal y como lo tengamos en ese momento, con toda la información. Se crea así:

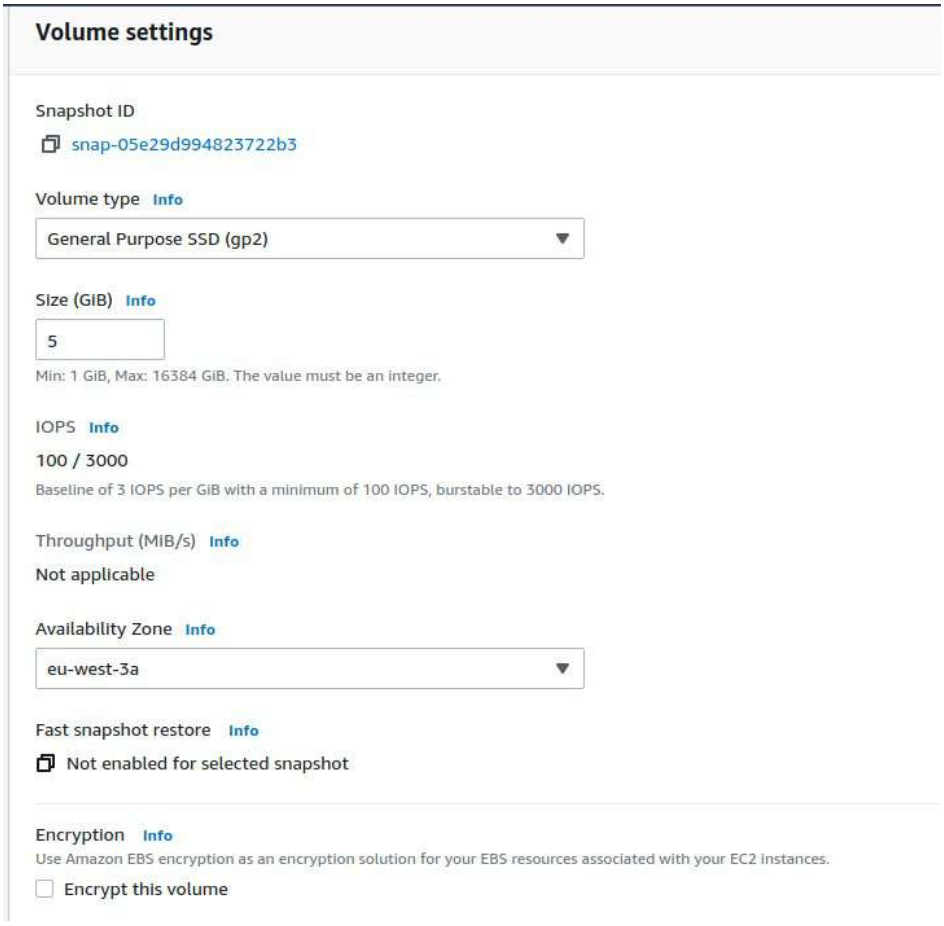




Ahora se puede crear un volumen a través de este snapshot.



Se podrá crear un disco más grande. Hay que4 seleccionar la zona correcta donde este la instancia a donde lo queremos asociar.



Cuando lo asociemos el SO detectará el formato del volumen.

Después, para borrar el volumen hay que borrar la instancia o desasociar y borrar el volumen.

Las snapshot se pueden borrar con facilidad. Si tenemos AMIs propias habrán Snapshot asociadas a esas AMIs, porque AWS las guarda así. Estas no se pueden borrar desde snapshot

10.4. - Linea de comandos con volumenes

Podemos ver los volumenes con

aws ec2 describe-volumes

Para ver solo los ids de los volumenes

aws ec2 describe-volumes --query 'Volumes[].VolumeId'

Y para ver a donde están asociadas

aws ec2 describe-volumes --query 'Volumes[].[{"Id del Volumen":VolumeId},{"ID instancia asociada":Attachments[].InstanceId}]'

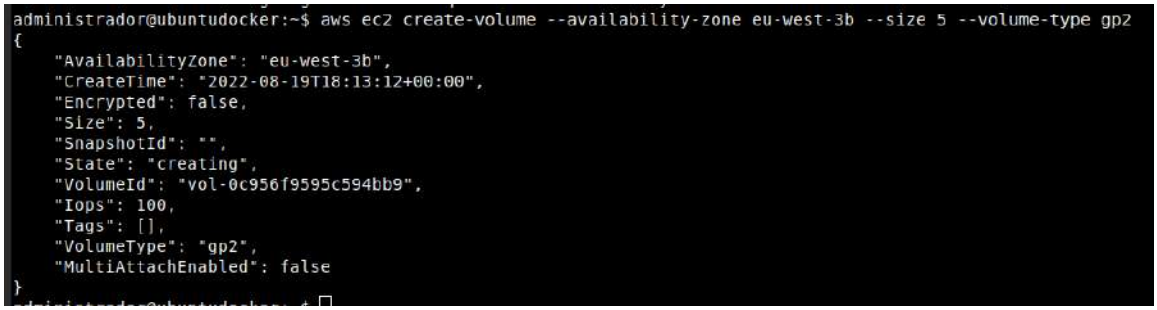
Para crear un volumen es con el sub-comando create-volume que tiene estas opciones:

--availability-zone <value>
[--encrypted | --no-encrypted]
[--iops <value>]
[--kms-key-id <value>]
[--outpost-arn <value>]
[--size <value>]
[--snapshot-id <value>]
[--volume-type <value>]
[--dry-run | --no-dry-run]
[--tag-specifications <value>]
[--multi-attach-enabled | --no-multi-attach-enabled]
[--throughput <value>]

[--client-token <value>]
[--cli-input-json | --cli-input-yaml]
[--generate-cli-skeleton <value>]

Tendremos que poner la AZ que debe ser la misma de la instancia a la que queremos asociarla. Ejemplo:

aws ec2 create-volume --availability-zone eu-west-3b --size 5 --volume-type gp2



Ahora ya aparece en el listado de volumenes, pero sin instancia asociada

aws ec2 describe-volumes --query 'Volumes[].[{"Id del Volumen":VolumeId},{"Estado asociación a Instancia":Attachments[].State},{"ID instancia asociada":Attachments[].InstanceId}]' --output yaml

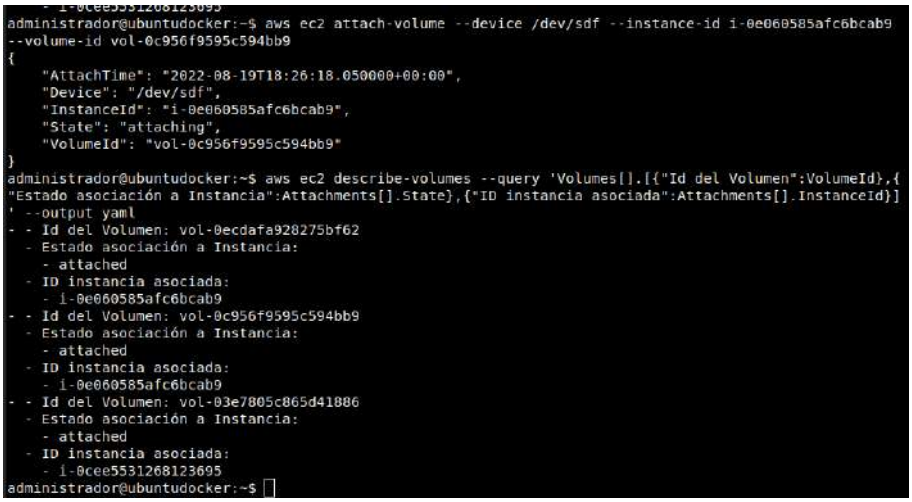
Para asociarlo es con el sub-comando attach-volume que tiene estas opciones:

--device <value>
--instance-id <value>
--volume-id <value>
[--dry-run | --no-dry-run]
[--cli-input-json | --cli-input-yaml]
[--generate-cli-skeleton <value>]

El comando queda así:

aws ec2 attach-volume --device /dev/sdf --instance-id i-0e060585afc6bcab9 --volume-id vol-0c956f9595c594bb9

Y ahora ya podemos ver la asociación



Para dettach

[--device <value>]
[--force | --no-force]
[--instance-id <value>]
--volume-id <value>
[--dry-run | --no-dry-run]
[--cli-input-json | --cli-input-yaml]
[--generate-cli-skeleton <value>]

aws ec2 detach-volume --volume-id vol-0c956f9595c594bb9

```
administrador@ubuntudocker:~$ aws ec2 detach-volume --volume-id vol-0c956f9595c594bb9
{
    "AttachTime": "2022-08-19T18:26:17+00:00",
    "Device": "/dev/sdf",
    "InstanceId": "i-0e060585afc6bcab9",
    "State": "detaching",
    "VolumeId": "vol-0c956f9595c594bb9"
}
administrador@ubuntudocker:~$ aws ec2 describe-volumes --query 'Volumes[].[{"Id del Volumen":VolumeId},{
"Estado asociación a Instancia":Attachments[].State},{"ID instancia asociada":Attachments[].InstanceId}]
' --output yaml
- - Id del Volumen: vol-0ecdafa928275bf62
  - Estado asociación a Instancia:
    - attached
  - ID instancia asociada:
    - i-0e060585afc6bcab9
- - Id del Volumen: vol-0c956f9595c594bb9
  - Estado asociación a Instancia:
    - detaching
  - ID instancia asociada:
    - i-0e060585afc6bcab9
- - Id del Volumen: vol-03e7805c865d41886
  - Estado asociación a Instancia:
    - attached
  - ID instancia asociada:
    - i-0cee5531268123695
administrador@ubuntudocker:~$
```

Para borrar es delete-volume

--volume-id <value>
[--dry-run | --no-dry-run]
[--cli-input-json | --cli-input-yaml]
[--generate-cli-skeleton <value>]

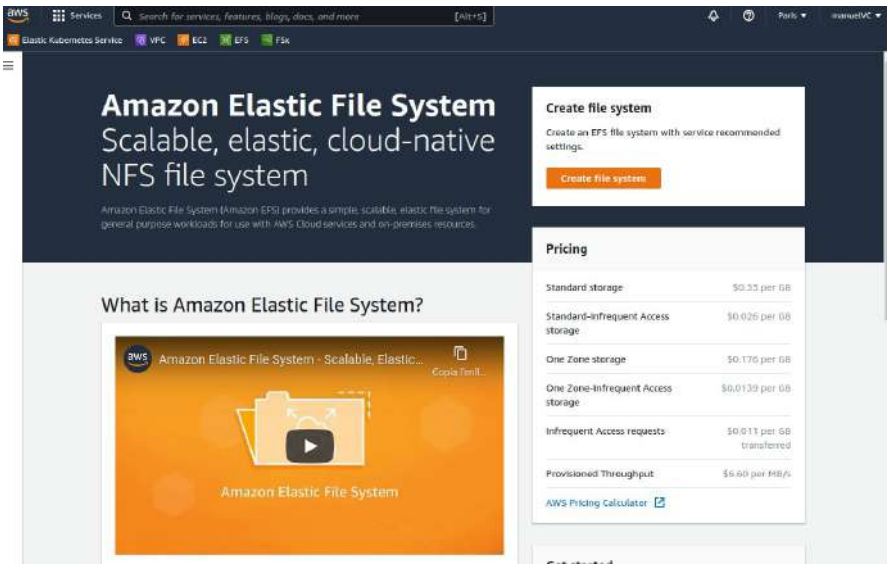
aws ec2 delete-volume --volume-id vol-0c956f9595c594bb9

```
administrador@ubuntudocker:~$ aws ec2 delete-volume --volume-id vol-0c956f9595c594bb9
administrador@ubuntudocker:~$ aws ec2 describe-volumes --query 'Volumes[].[{"Id del Volumen":VolumeId},{
"Estado asociación a Instancia":Attachments[].State},{"ID instancia asociada":Attachments[].InstanceId}]
' --output yaml
- - Id del Volumen: vol-0ecdafa928275bf62
  - Estado asociación a Instancia:
    - attached
  - ID instancia asociada:
    - i-0e060585afc6bcab9
- - Id del Volumen: vol-03e7805c865d41886
  - Estado asociación a Instancia:
    - attached
  - ID instancia asociada:
    - i-0cee5531268123695
```

TEMA 11 - EC2 Trabajar con EFS y FSX. Sistemas de ficheros compartidos

11.1. - EFS (Linux)

EFS (Amazon Elastic File System) es un Sistema de ficheros parecido a NFS (Network File System). Se utiliza para tener acceso en red entre los Linux.



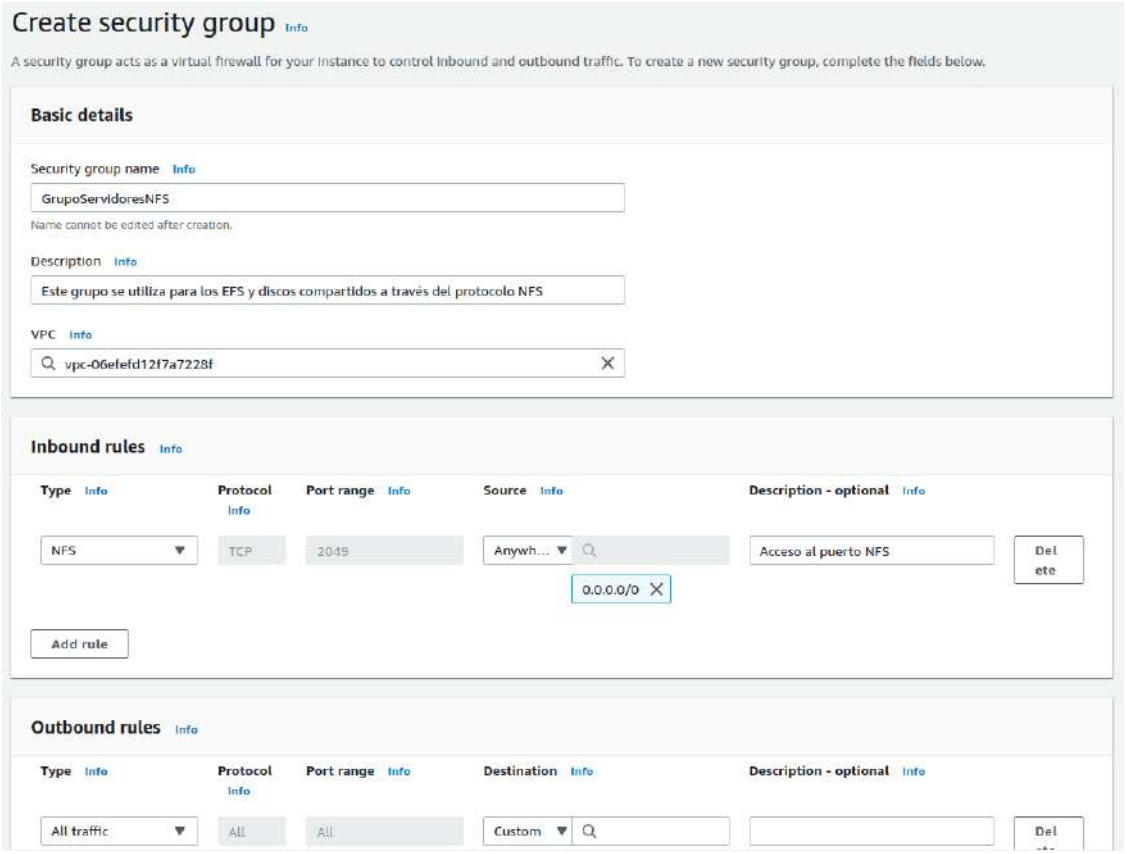
Standar almacena el fichero en dos AZ para tener backup.

Standard-infrequent es lo mismo que estandar pero para documentos que no se consulten mucho. Con esta opción cobran por acceder a los datos.

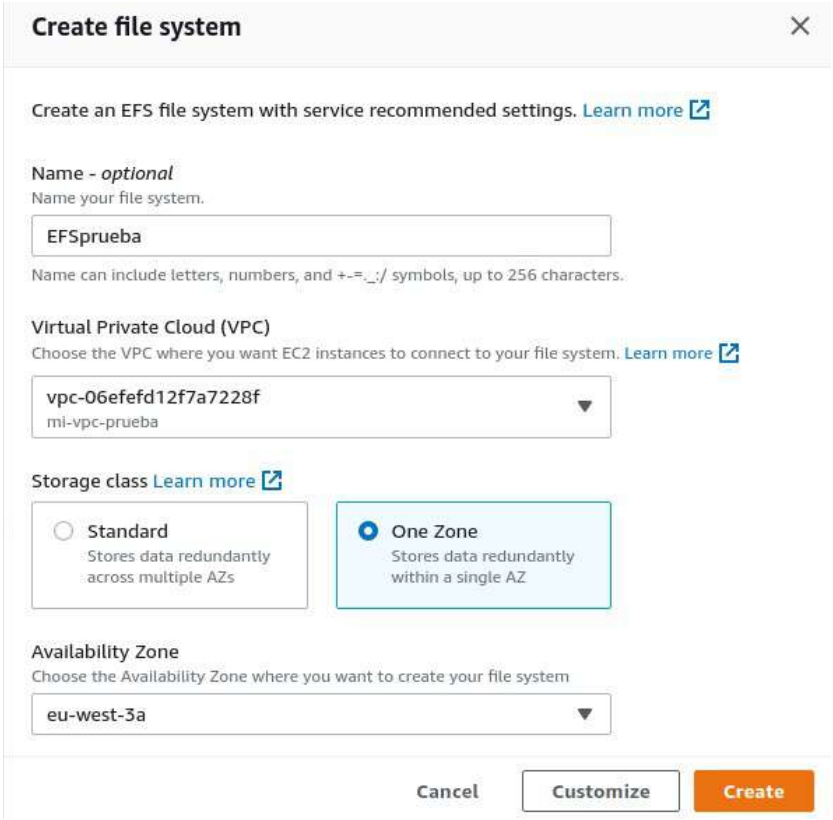
One Zone es que se almacenan los datos en una AZ. También tiene su opción por infrequent.

Se puede utilizar para hacer backup, para sincronizar datos y para transferir datos con un servicio ftp/sftp.

Antes de continuar tenemos que tener un grupo de seguridad que admita el acceso a través del puerto NFS (2049). El acceso al EFS es a través de un acces point al servidor, por eso lo necesita.

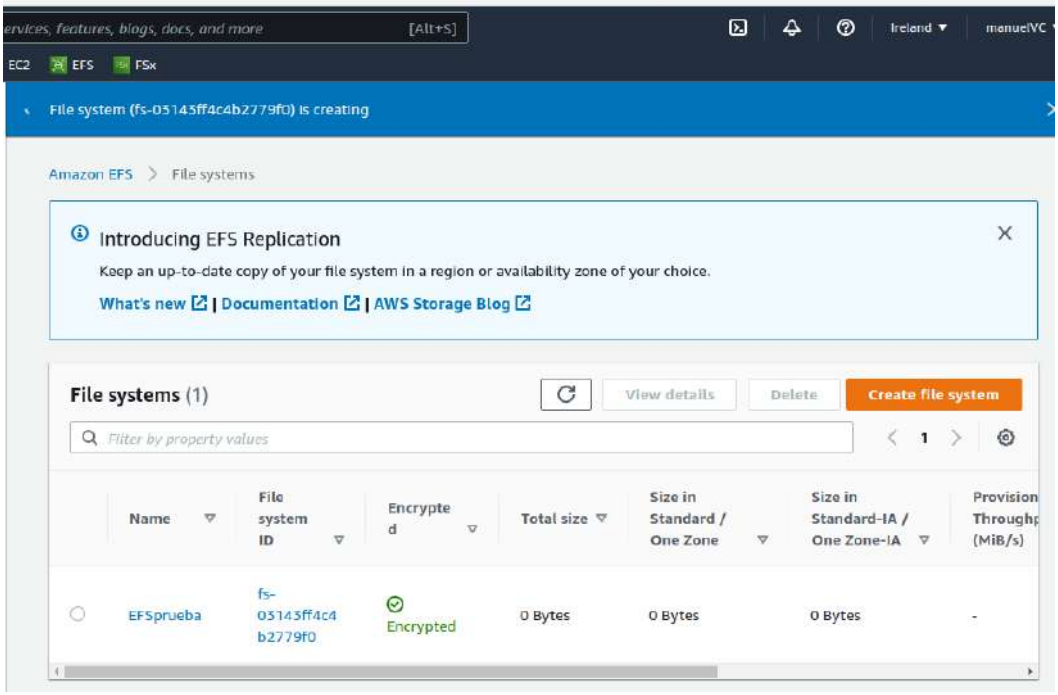


Para crearlo se puede hacer en modo sencillo o con más detalles en el modo avanzado clicando en Customize. Hay que asegurarse tanto el VPC correcto como la AZ.

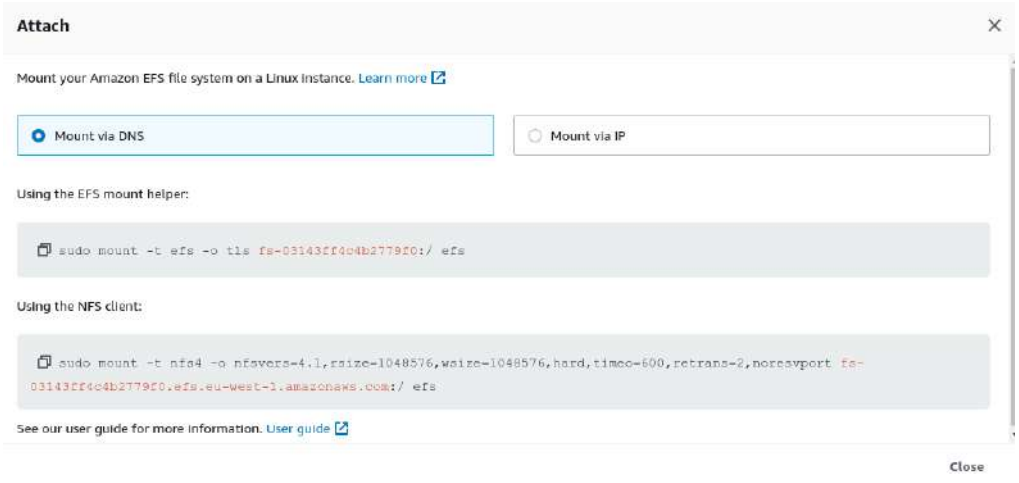


En la AZ Paris no permite crear EFS

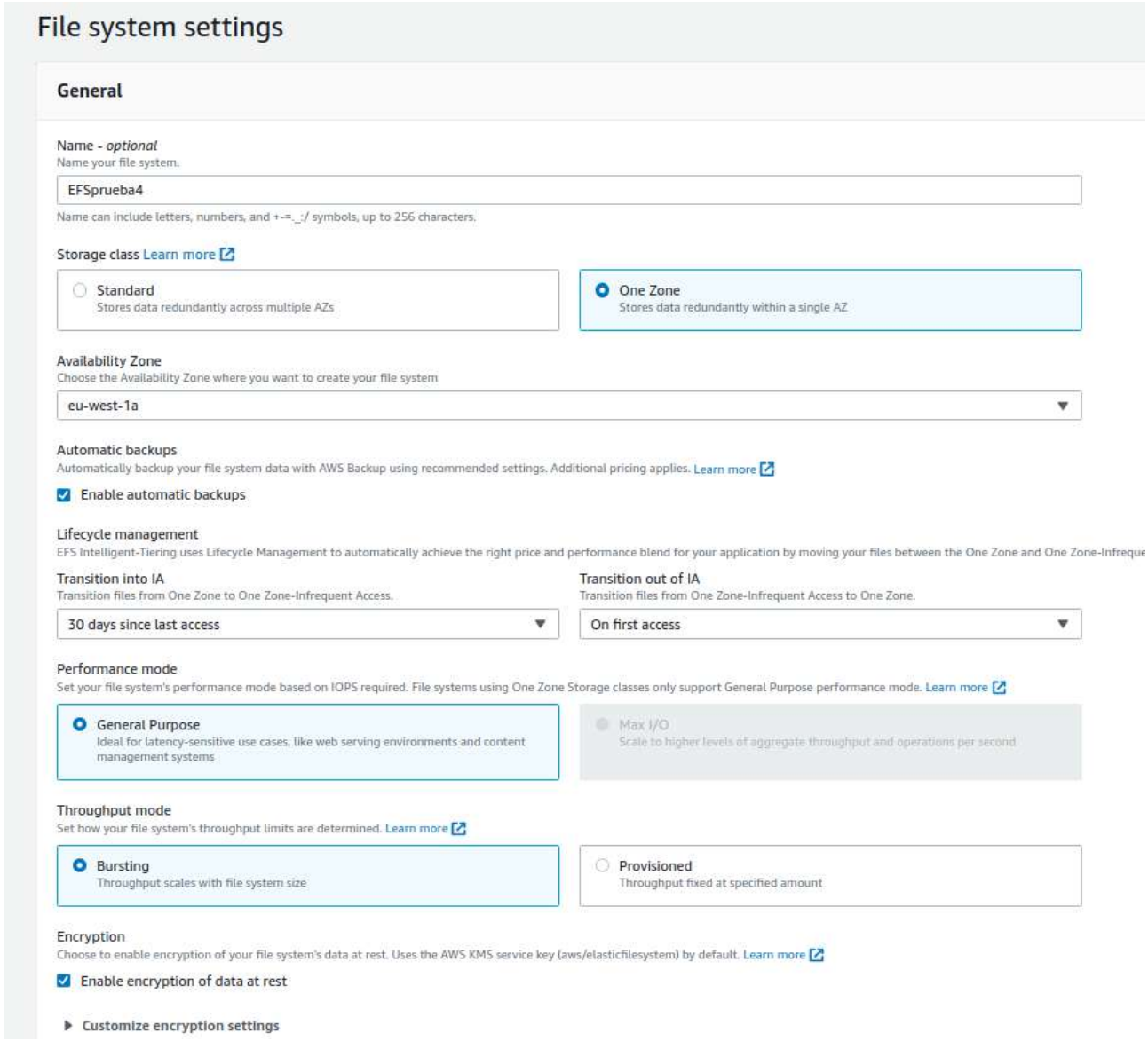
En Irlanda sí que lo permite



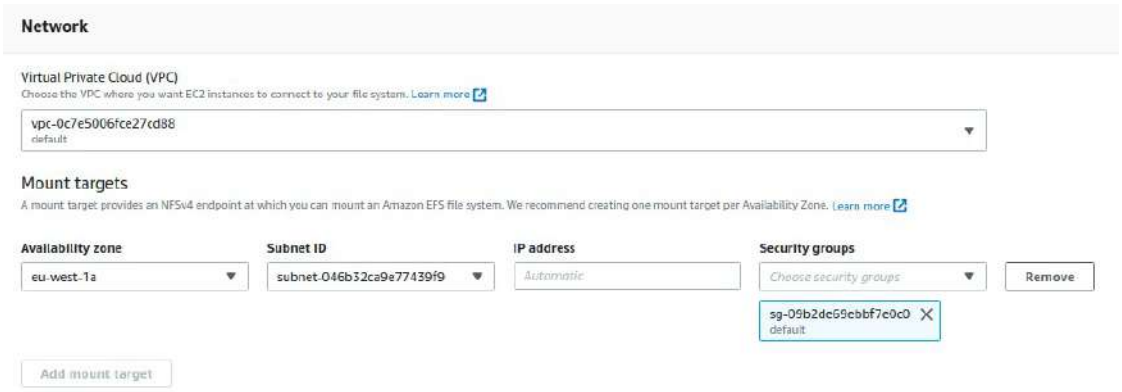
Para asociarlo existen varias formas



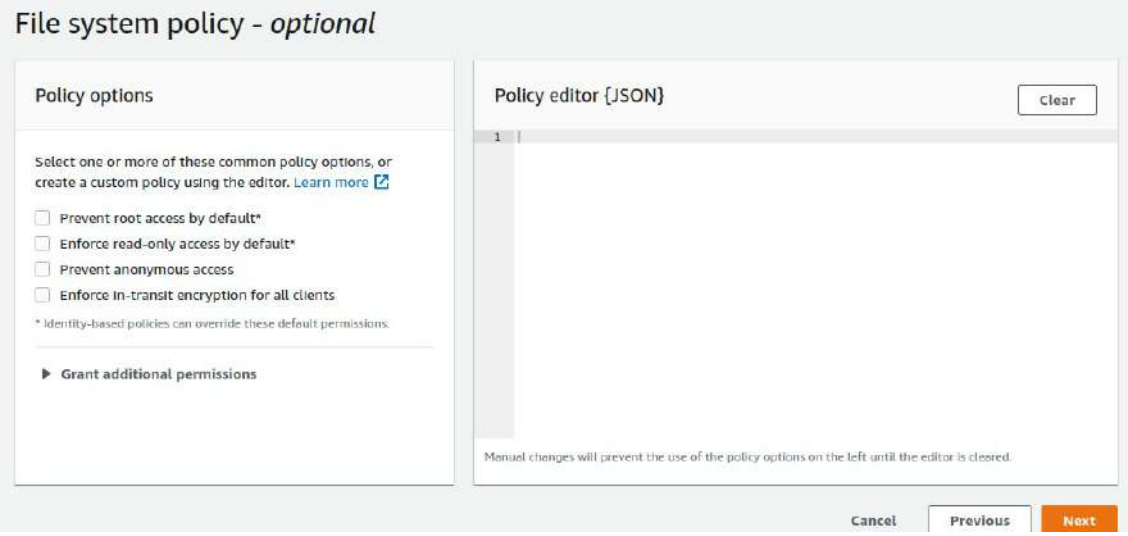
Los ajustes avanzados permiten hacer backup, gestión del ciclo de vida



Podemos especificar opciones de red. Importante escogerlo correctamente respecto las instancias que queremos que se conecten. Y el grupo de seguridad debe permitir el acceso NFS

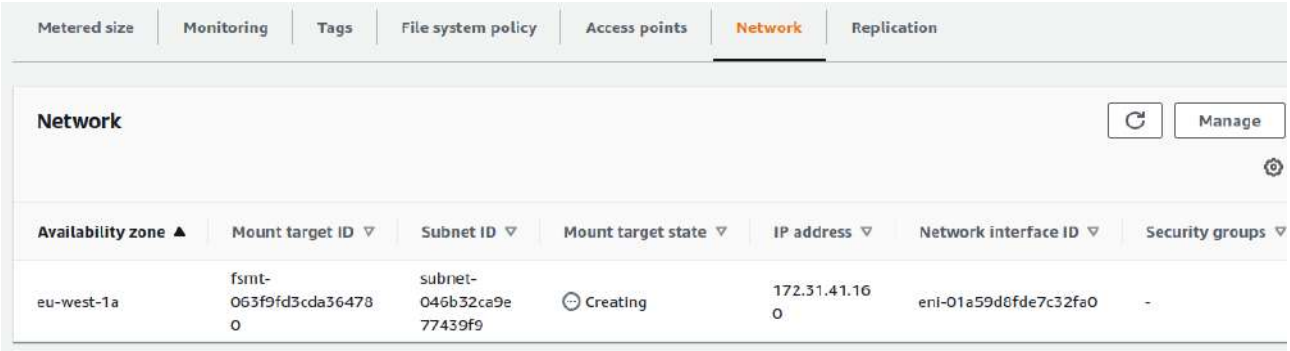


Podemos especificar con JSON una política de sistema de ficheros. Se puede editar en Policy editor. Tenemos las opciones: Impedir el acceso de root por defecto, Imponer el acceso de sólo lectura por defecto, Impedir el acceso anónimo y Imponer el cifrado en tránsito para todos los clientes.



Y ya tendríamos creado el Fyle System.

ATENCIÓN que la red del EFS tarda en crearse

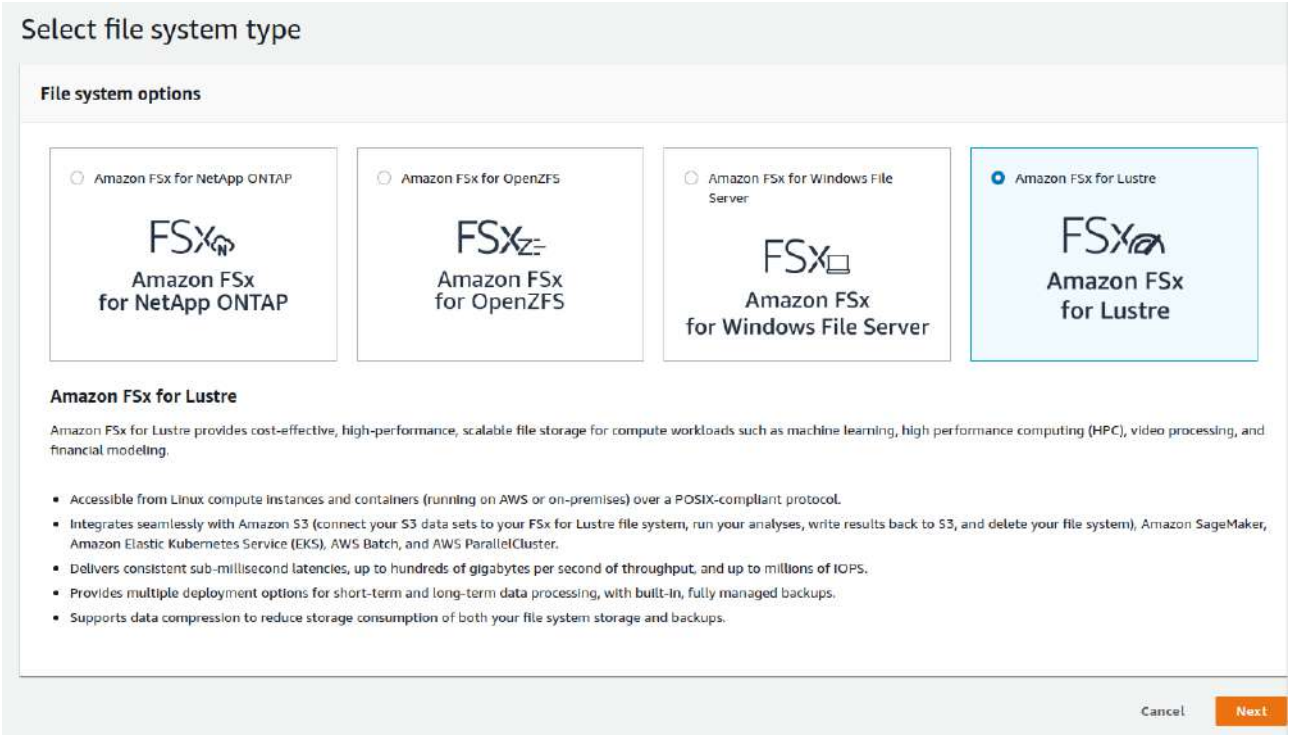


11.2. - FSx (Windows y otros SO)

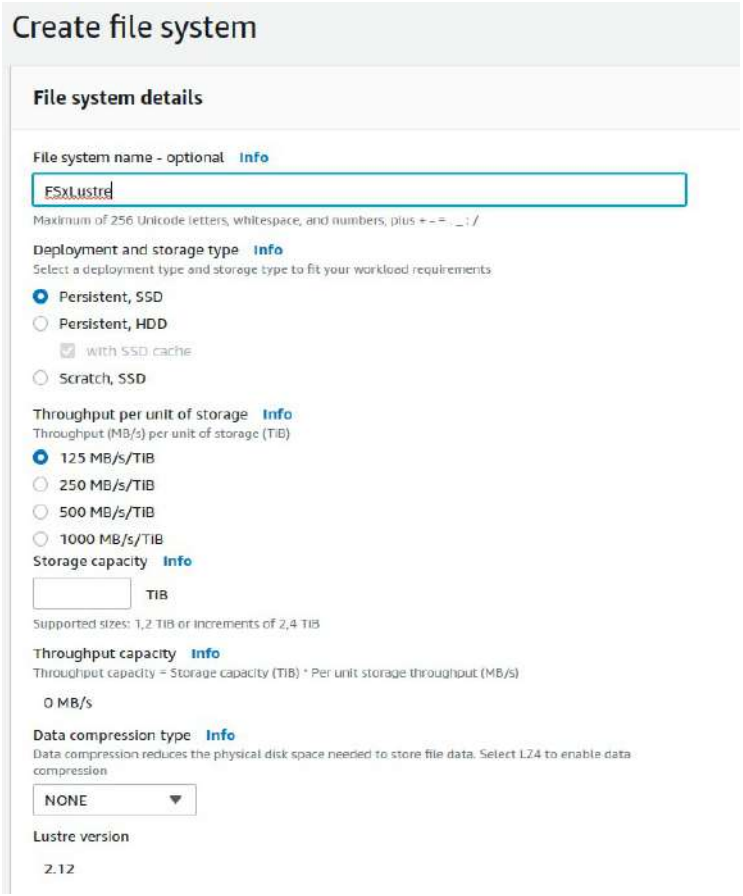
Es el remplazo de NFS para Windows. Se puede utilizar con 4 sistemas de ficheros:

- NetApp
- OpenZFS
- Windows File Server
- Lustre – Es open source https://www.lustre.org/

11.2.1. - Prueba con Lustre para Linux



Tenemos diversas opciones como clase de disco, velocidad de I/O, capacidad, etc



Los tipos de almacenamiento Scratch es para datos que no importan si se pierden.

Si queremos crear el grupo de seguridad a mano necesitaremos varios rangos de puiertos: 988 y 1021-1023

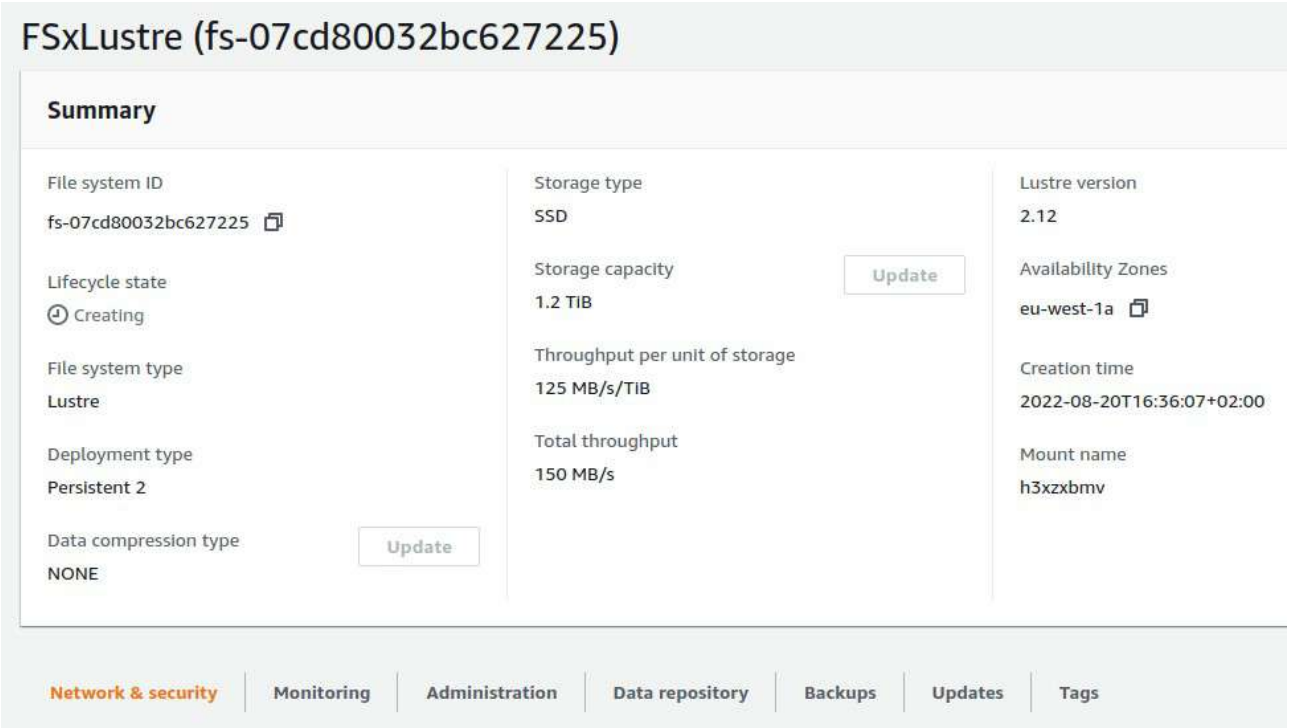
Type	Protocol	Port Range	Source	Description
Custom TCP rule	TCP	988	Choose **Custom** and enter the security group ID of the security group that you just created	Allows Lustre traffic between FSx for Lustre file servers
Custom TCP rule	TCP	988	Choose **Custom** and enter the security group IDs of the security groups associated with your Lustre clients	Allows Lustre traffic between FSx for Lustre file servers and Lustre clients
Custom TCP rule	TCP	1021-1023	Choose **Custom** and enter the security group ID of the security group that you just created	Allows Lustre traffic between FSx for Lustre file servers
Custom TCP rule	TCP	1021-1023	Choose **Custom** and enter the security group IDs of the security groups associated with your Lustre clients	Allows Lustre traffic between FSx for Lustre file servers and Lustre clients

Por defecto ya crear un grupo de seguridad correcto.

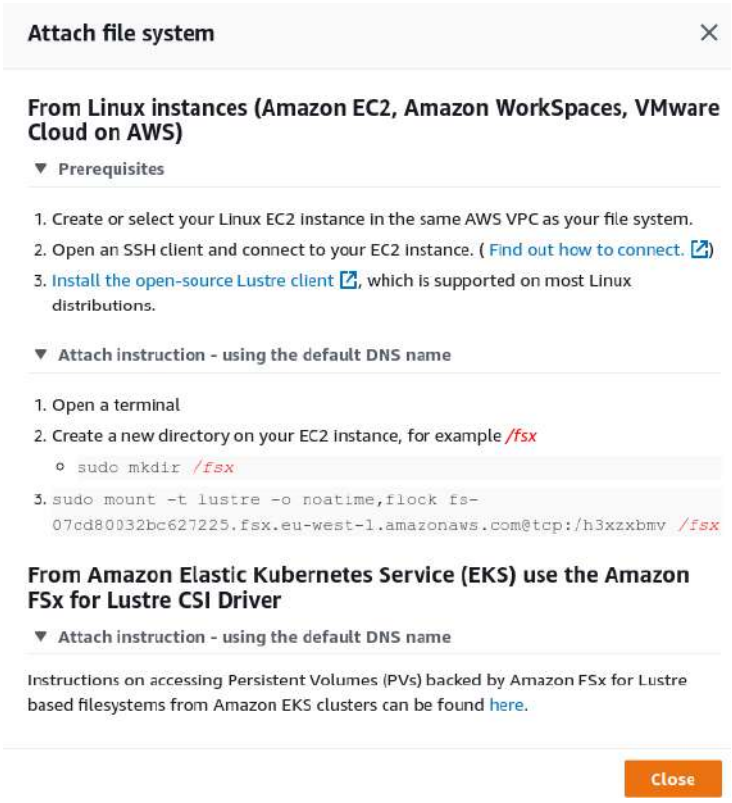
También aparece la encripctación, si queremos exportar datos a S3, configurar el logging (Los logs se podrán leer en CloudWatch), Backup (Se harán periodicos pero también se pueden hacer manuales) y Tags

Tardar un rato en crearse. Cuidado que es un producto caro, si es una prueba no hay que dejarlo creado.

Podremos ver detalles del componente



Para asociarlo es parecido a NFS, hay que clicar en Attach y dará los comandos para hacerlo

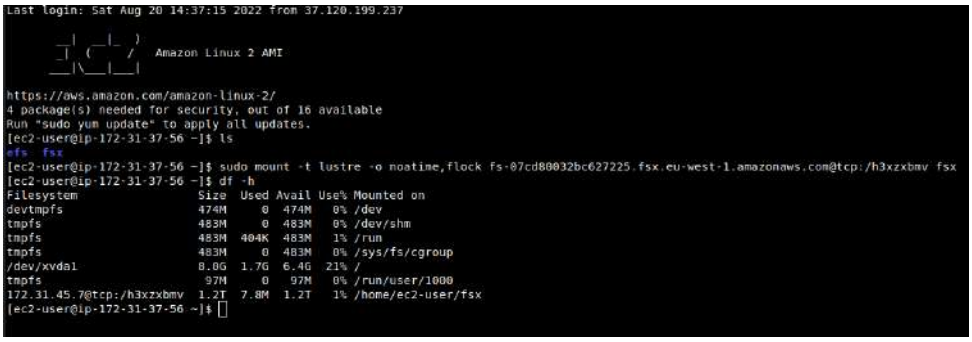


En este caso debemos instalar el cliente de Lustre. Para hacerlo en Amazon Linux 2 es:

sudo amazon-linux-extras install -y lustre2.10

sudo yum -y update kernel && sudo reboot

Luego ya podemos montar el volumen



11.3. - Linea de comandos

Para acceder a efs o fsx hay que entrar en sus subcomandos, sus subcomandos son:

aws efs

- create-access-point
- create-file-system
- create-mount-target
- create-replication-configuration
- delete-access-point
- delete-file-system
- delete-file-system-policy
- delete-mount-target
- delete-replication-configuration
- describe-access-points
- describe-account-preferences
- describe-backup-policy
- describe-file-system-policy
- describe-file-systems
- describe-lifecycle-configuration
- describe-access-points
- describe-account-preferences
- describe-backup-policy
- describe-file-system-policy
- describe-file-systems — Para ver los sistemas de ficheros creados
- describe-lifecycle-configuration
- describe-mount-target-security-groups
- describe-mount-targets
- describe-replication-configurations

list-tags-for-resource
modify-mount-target-security-groups
put-account-preferences
put-backup-policy
put-file-system-policy
put-lifecycle-configuration
tag-resource
untag-resource
update-file-system

aws fsx
associate-file-system-aliases
cancel-data-repository-task
copy-backup
create-backup
create-data-repository-association
create-data-repository-task
create-file-system
create-file-system-from-backup
create-snapshot
create-storage-virtual-machine
create-volume
create-volume-from-backup
delete-backup
delete-data-repository-association
delete-file-system
delete-snapshot
delete-storage-virtual-machine
delete-volume
describe-backups
describe-data-repository-associations
describe-data-repository-tasks
describe-file-system-aliases
describe-file-systems Para ver los sistemas de ficheros creados
describe-snapshots
describe-storage-virtual-machines
describe-volumes
disassociate-file-system-aliases
help
list-tags-for-resource
release-file-system-nfs-v3-locks
restore-volume-from-snapshot
tag-resource

untag-resource
update-data-repository-association
update-file-system
update-snapshot
update-storage-virtual-machine
update-volume

```
[root@curso ~]# aws fsx describe-file-systems
{
    "FileSystems": [
        {
            "OwnerId": "264297788131",
            "CreationTime": "2022-01-12T09:32:51.075000+01:00",
            "FileSystemId": "fs-09c7b3bd2f8bb1fac",
            "FileSystemType": "LUSTRE",
            "Lifecycle": "AVAILABLE",
            "StorageCapacity": 1200,
            "StorageType": "SSD",
            "VpcId": "vpc-0bb143acbd251ad96",
            "SubnetIds": [
                "subnet-07e47c4661a3bec50"
            ],
            "NetworkInterfaceIds": [
                "eni-034b16c3456471681",
                "eni-0cd8f284d5b68ba85"
            ],
            "DNSName": "fs-09c7b3bd2f8bb1fac.fsx.us-west-2.amazonaws.com",
            "KmsKeyId": "arn:aws:kms:us-west-2:264297788131:key/fff684a2-2573-4970-b37a-3378787a52d1",
            "ResourceARN": "arn:aws:fsx:us-west-2:264297788131:file-system/fs-09c7b3bd2f8bb1fac",
            "Tags": [
                {
                    "Key": "Name",
                    "Value": "fsx1"
                }
            ],
            "LustreConfiguration": {
                "WeeklyMaintenanceStartTime": "5:06:30",
                "DeploymentType": "PERSISTENT_2",
                "PerUnitStorageThroughput": 125,
                "MountName": "zeqzzbmv",
                "DailyAutomaticBackupStartTime": "06:30",
                "AutomaticBackupRetentionDays": 7,
                "CopyTagsToBackups": false,
                "DataCompressionType": "NONE",
                "LogConfiguration": {
                    "Level": "DISABLED"
                }
            },
            "FileSystemTypeVersion": "2.12"
        }
    ]
}
```

Ejemplo backup:

aws fsx create-backup –file-system-id xxxxxxxxxxxxx

```
[root@curso ~]# aws fsx create-backup --file-system-id fs-09c7b3bd2f8bb1fac
{
    "Backup": {
        "BackupId": "backup-019487af822374b4c",
        "Lifecycle": "PENDING",
        "Type": "USER_INITIATED",
        "CreationTime": "2022-01-12T10:51:09.081000+01:00",
        "KmsKeyId": "arn:aws:kms:us-west-2:264297788131:key/fff684a2-2573-4970-b37a-3378787a52d1",
        "ResourceARN": "arn:aws:fsx:us-west-2:264297788131:backup/backup-019487af822374b4c",
        "Tags": [],
        "FileSystem": {
            "FileSystemId": "fs-09c7b3bd2f8bb1fac",
            "FileSystemType": "LUSTRE",
            "StorageCapacity": 1200,
            "StorageType": "SSD",
            "ResourceARN": "arn:aws:fsx:us-west-2:264297788131:file-system/fs-09c7b3bd2f8bb1fac",
            "LustreConfiguration": {
                "WeeklyMaintenanceStartTime": "5:06:30",
                "DeploymentType": "PERSISTENT_2",
                "PerUnitStorageThroughput": 125,
                "DailyAutomaticBackupStartTime": "06:30",
                "AutomaticBackupRetentionDays": 7,
                "DataCompressionType": "NONE"
            },
            "FileSystemTypeVersion": "2.12"
        },
        "OwnerId": "264297788131"
    }
}
```

Se pueden ver los backups con:

aws fsx describe-backups

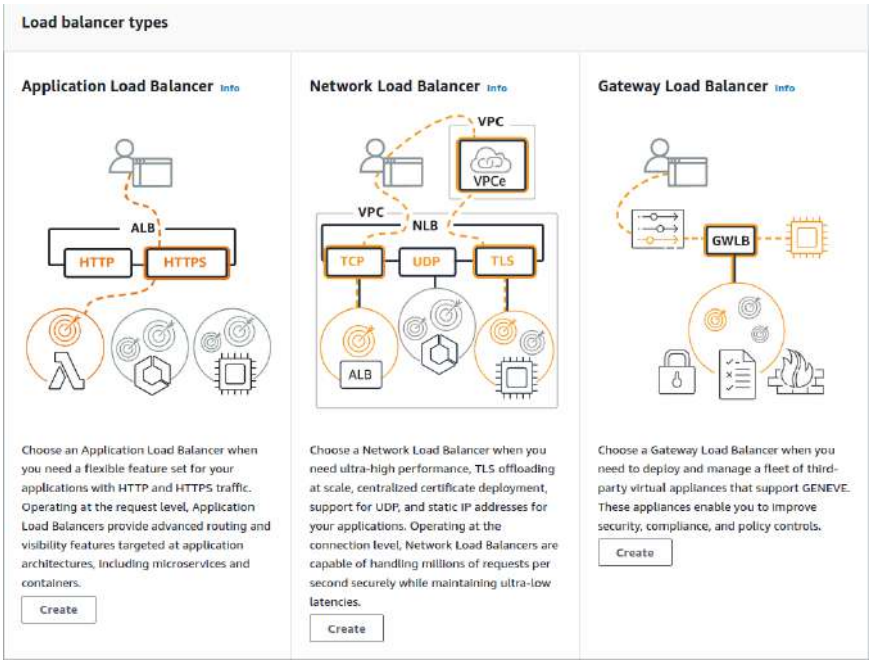
```
[root@curso ~]# aws fsx describe-backups
{
    "Backups": [
        {
            "BackupId": "backup-019487af822374b4c",
            "Lifecycle": "CREATING",
            "Type": "USER_INITIATED",
            "CreationTime": "2022-01-12T10:51:09.081000+01:00",
            "KmsKeyId": "arn:aws:kms:us-west-2:264297788131:key/fff684a2-2573-4970-b37a-3378787a52d1",
            "ResourceARN": "arn:aws:fsx:us-west-2:264297788131:backup/backup-019487af822374b4c",
            "Tags": [],
            "FileSystem": {
                "FileSystemId": "fs-09c7b3bd2f8bb1fac",
                "FileSystemType": "LUSTRE",
                "StorageCapacity": 1200,
                "StorageType": "SSD",
                "ResourceARN": "arn:aws:fsx:us-west-2:264297788131:file-system/fs-09c7b3bd2f8bb1fac",
                "LustreConfiguration": {
                    "WeeklyMaintenanceStartTime": "5:06:30",
                    "DeploymentType": "PERSISTENT_2",
                    "PerUnitStorageThroughput": 125,
                    "DailyAutomaticBackupStartTime": "06:30",
                    "AutomaticBackupRetentionDays": 7,
                    "DataCompressionType": "NONE"
                },
                "FileSystemTypeVersion": "2.12"
            },
            "OwnerId": "264297788131"
        }
    ]
}
```

TEMA 12 - EC2 Load Balancers. Balanceadores de carga

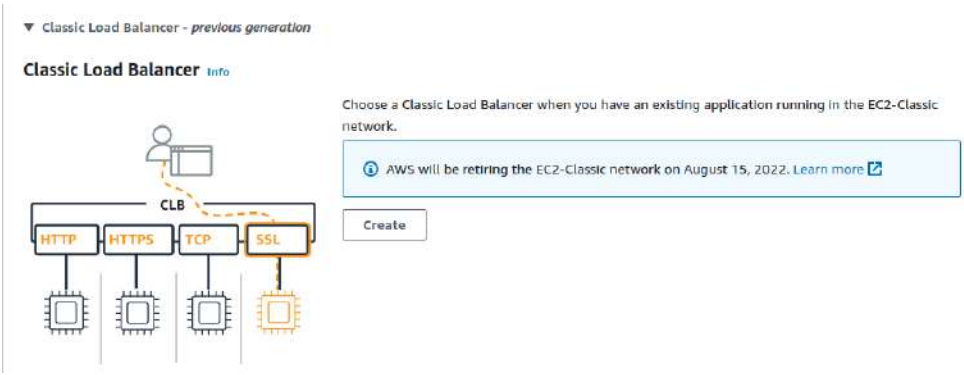
Mediante un failover se repartirá el tráfico entre las instancias que hayamos preparado con la misma app.

Se pueden crear 3 tipos de balanceadores de carga según en la capa OSI que trabajen:

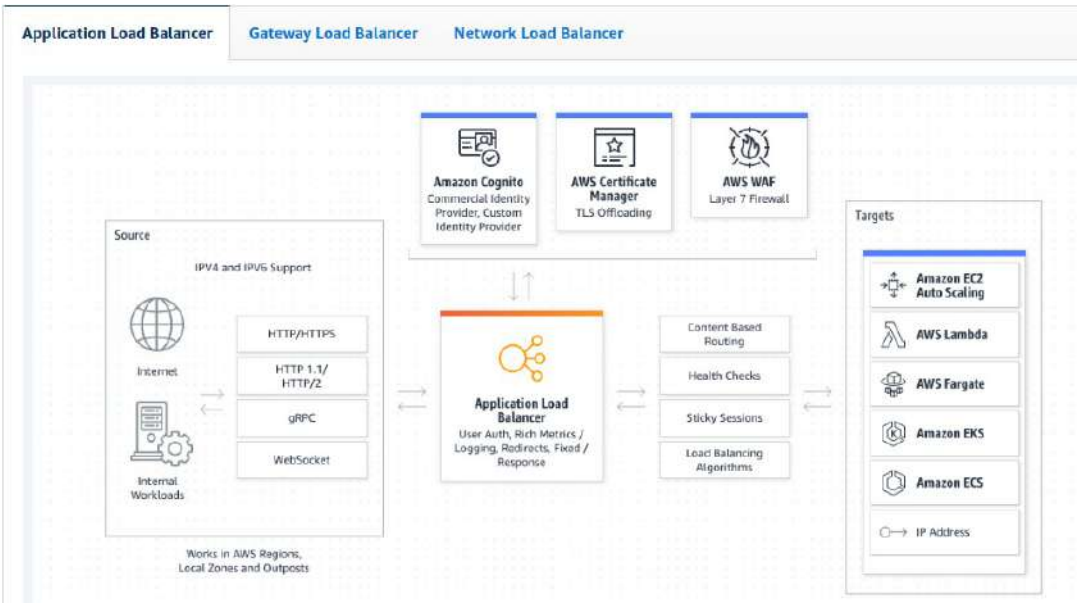
- **Balanceador de aplicación**. Trabajan en la capa de aplicación, en entornos web (http o https) pueden ir a instancias, a firegates o a Lamdba.
- **Balanceador de red**. Trabajan en la capa 4 de transporte. TCP, UDP y TLS. Para instancias y otros productos de Amazon. También a los ALB (Application Load Balancer)
- **Balanceador de Gateway**. Trabajan con adplayers¿? de terceros que soportan GENEVE.

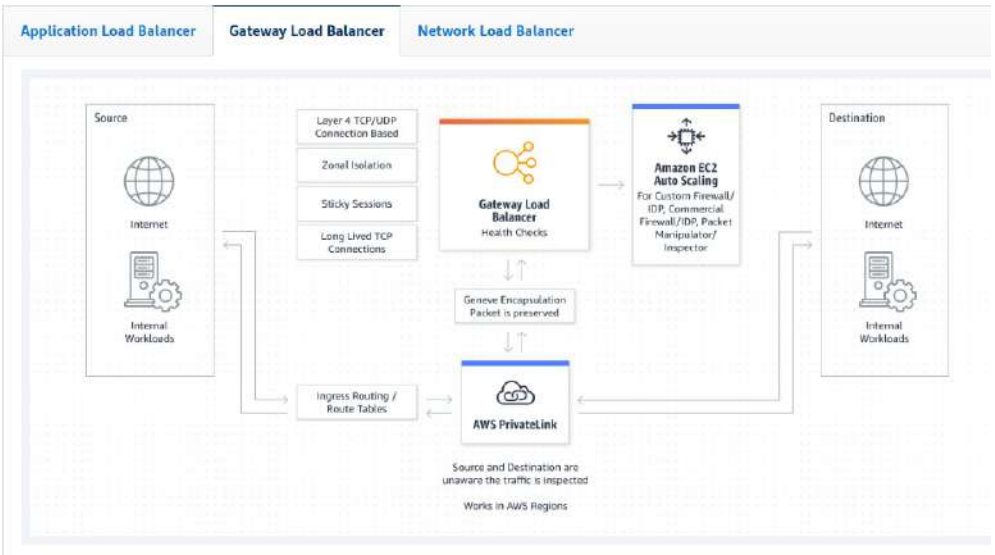


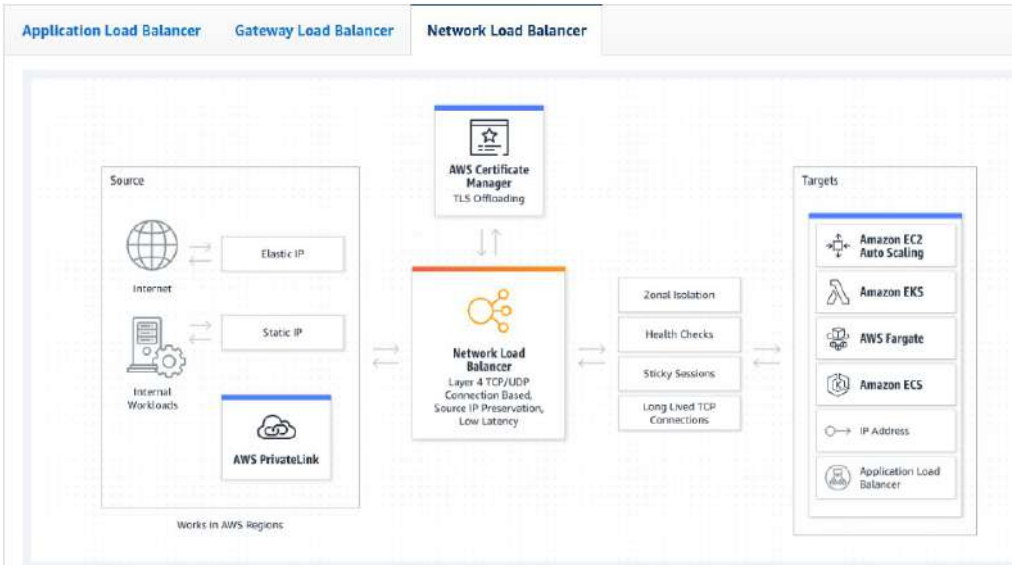
Por otra parte se puede acceder al balanceador de carga antiguo de Amazon, pero de manera temporal. Se supone que el 15 de agosto pasado dejaba de funcionar. Estaba pensado para las grandes plataformas que usaban el antiguo balanceador.



Documentación de los balanceadores: https://aws.amazon.com/elasticloadbalancing/?nc=sn&loc=1

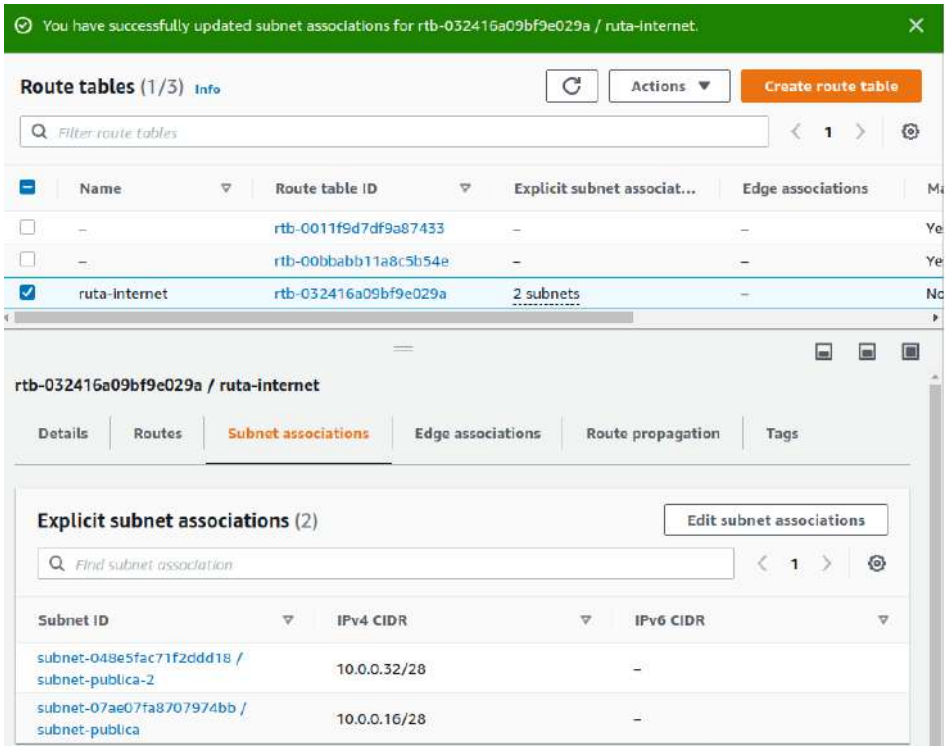




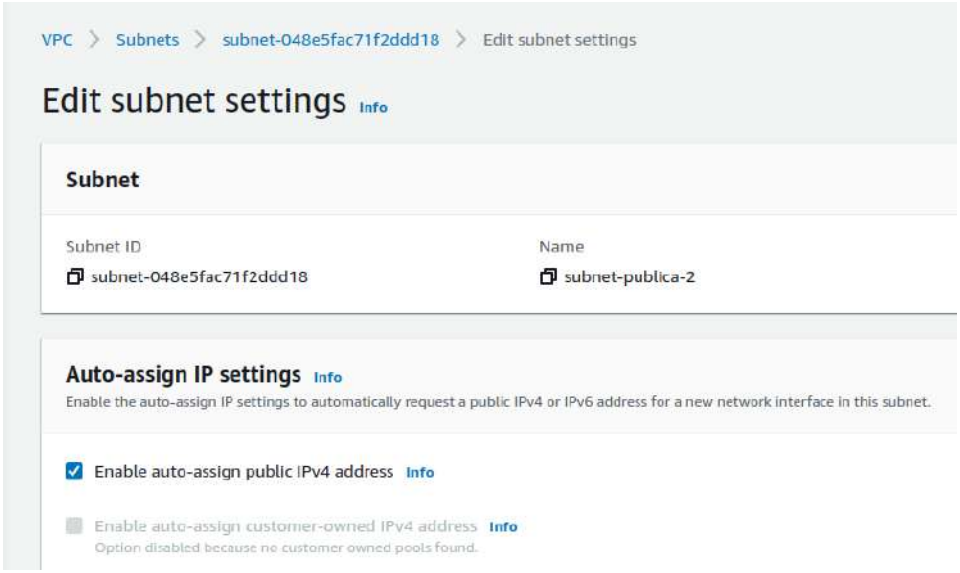


Necesitamos al menos dos subredes donde balancear, cada una debe estar en un AZ distinta.

Para hacerla pública se debe tener el gateway y en la tabla de rutas tener asociada la subred.



En la subred se debe habilitar la asignación de Ipv4 pública



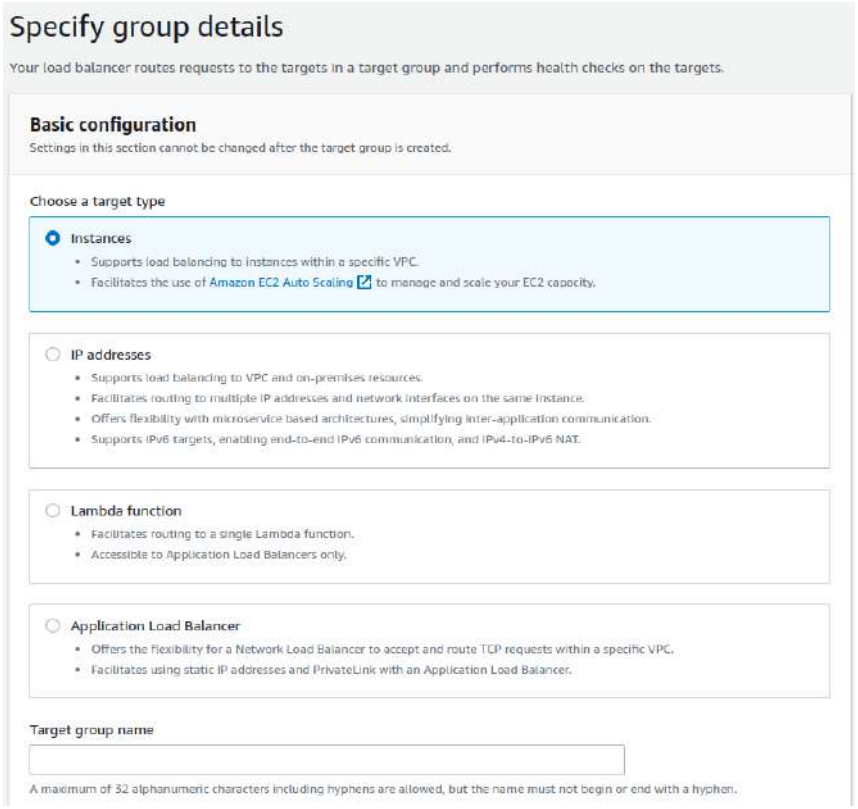
La VPC tendrá que tener las DNS hostnames y las DNS resolution habilitadas.

Las instancias nuevas tienen que tener habilitada la Ipv$ pública.

Ahora tenrdremos que crear el **Target Group**. Son los componentes que se balancearán, en este caso añadimos las 3 instancias.

Para el Gateway Load Balancer se necesitan más recursos para poder hacerlo, por eso se hará la práctica. Se hará del ALB y del NLB.

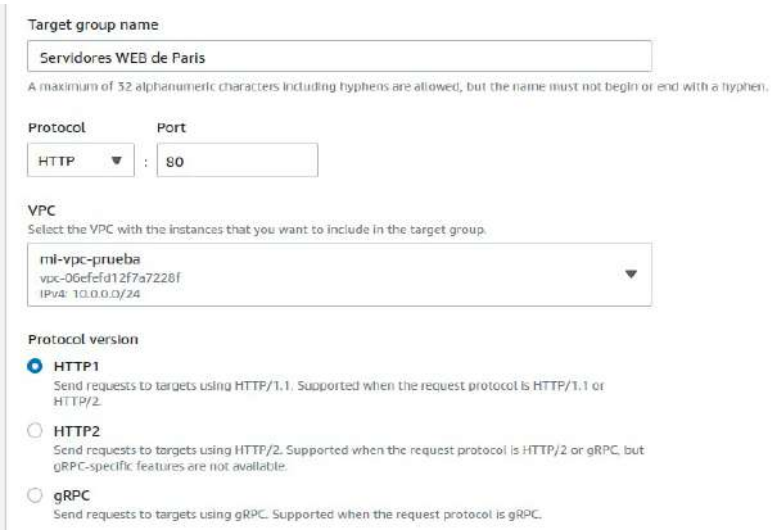
12.1. - Target groups



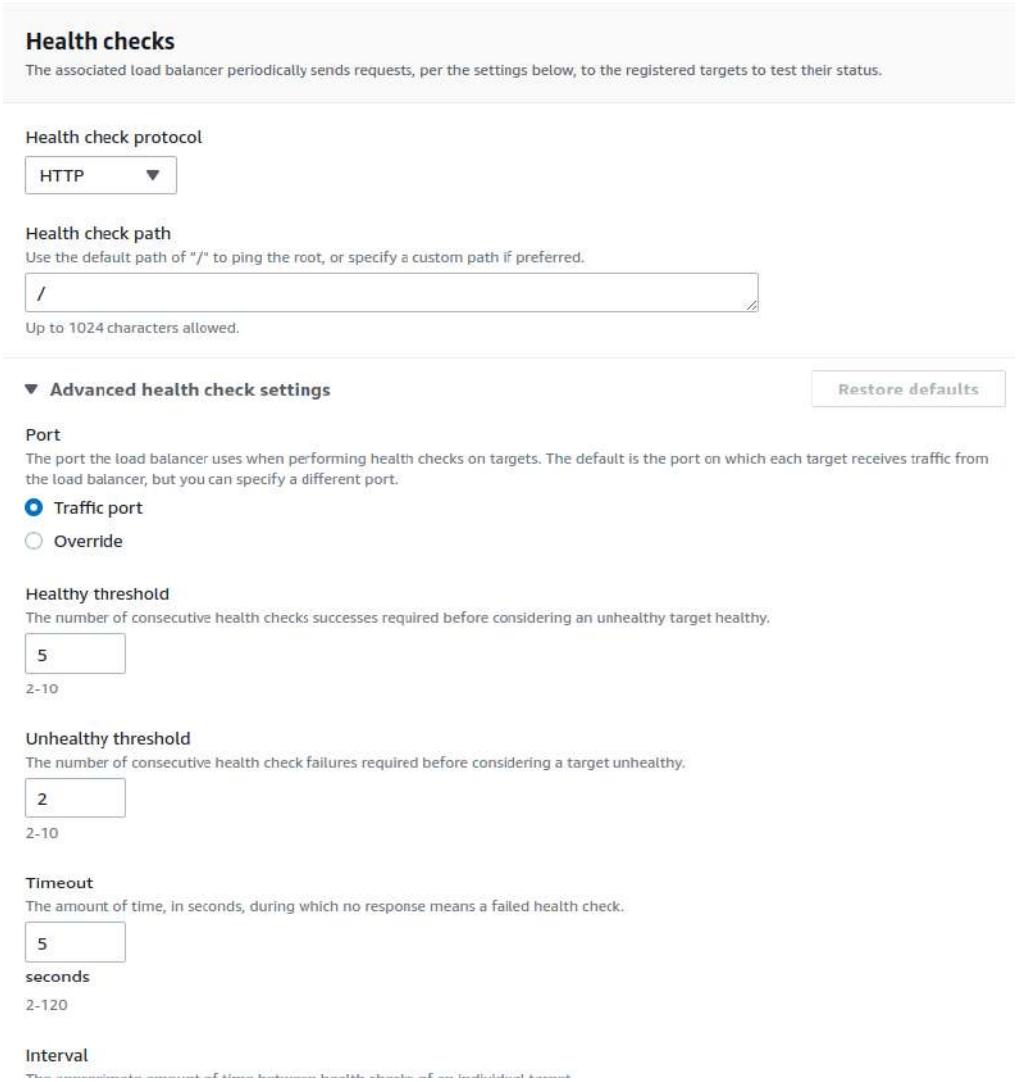
Se puede balancear:

- Instancias
- Direcciones IPs. Una instancias podría tener varias IPs
- Funciones Lambda. Es para ejecutar procedimientos y funciones en entornos serverless.
- Aplicaciones

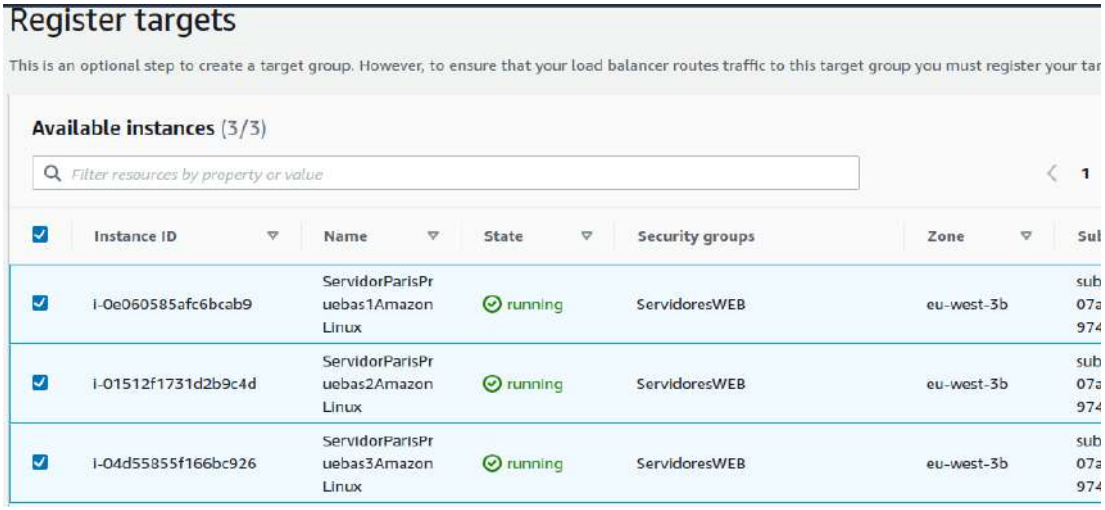
Se puede indicar el nombre, el protocolo, ls VPC, la versión de protocolo,



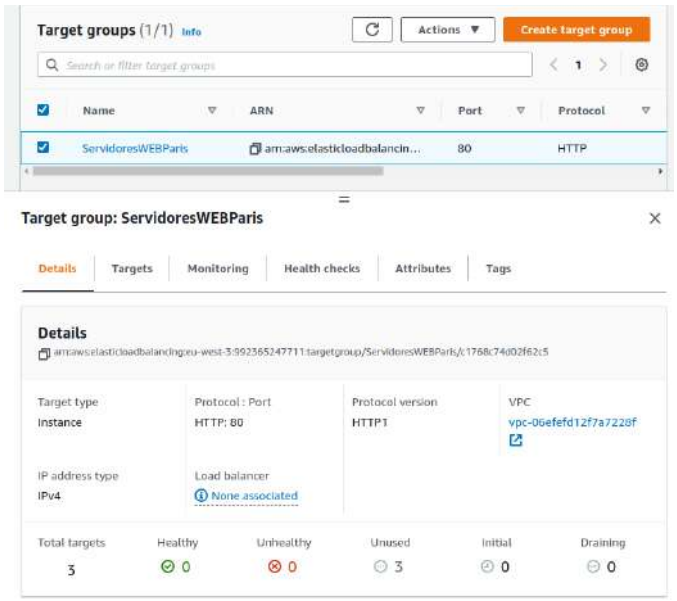
Puede hacer un checkeo para comprobar si funciona el protocolo. Lo que hará es lanzar ping y esperar su respuesta. Se pueden configurar detalles como poner una página concreta a comprobar.



Luego indica las instancias (o los elementos) que queremos seleccionar para balancear.



Aquí también se puede indicar el puerto. Después de incluir los elementos ya se crea el target group.

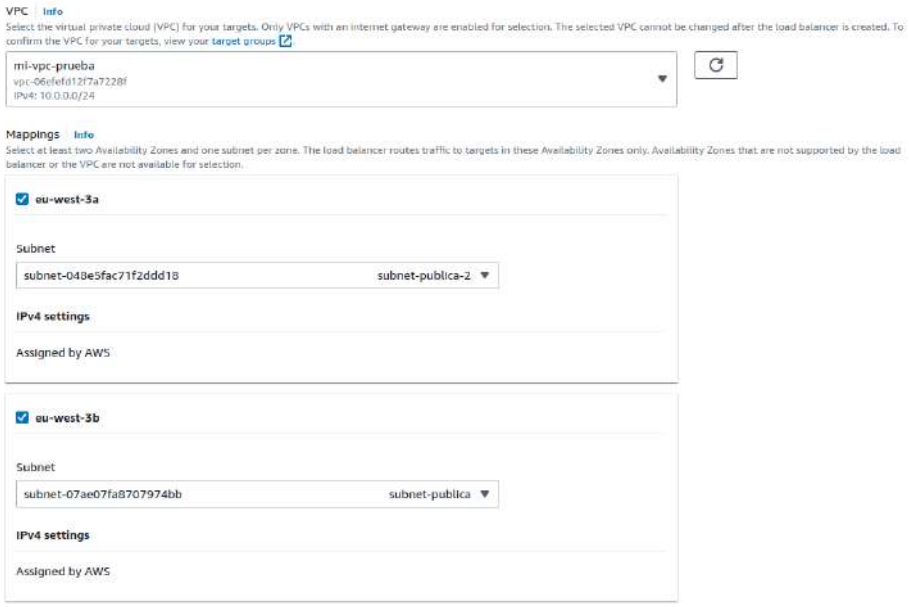


12.2. - Crear el balanceador de carga ALB

Para las 3 instancias anteriores usaremos el ALB para balancear en capa de aplicación.

En el esquema puede ser Internet-facing o Internal. La interna puede ser para SQL u otros servidores que estén dentro de la red privada.

El Network Mapping se debe configurar con cuidado para tener el acceso a los targets.

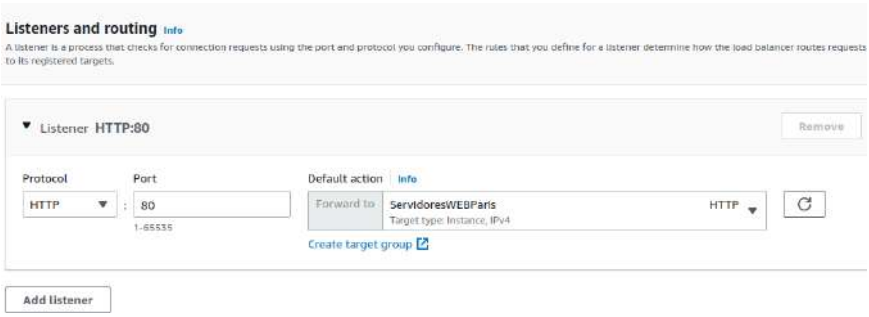


Aquí se ve que es obligatorio tener al menos dos AZ.

Ahora mismo no ofrece otros AZ porque solo ofrece los que tiene subnet.

En el grupo de seguridad debe tener habilitado el puerto 80.

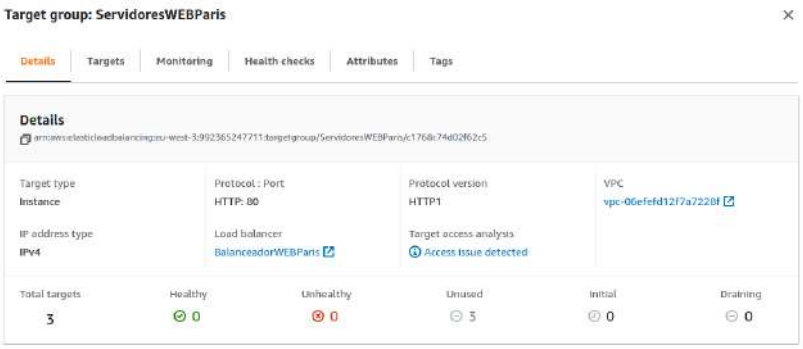
En el Listener es importante que este el puerto que necesite la aplicación. En este momento también se puede crear el Target Group.



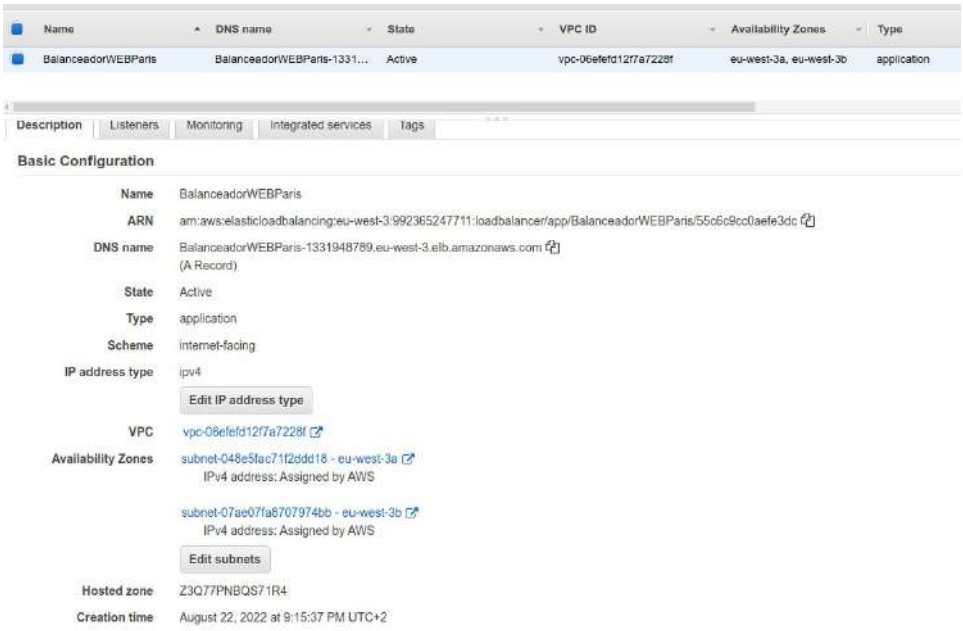
Como se pueden poner varios puertos se pueden destinar para aplicaciones distintas.

El AWS Global Accelerator sirve para utilizar direcciones IP fijas usando la red global de Amazon para optimizar el tráfico.

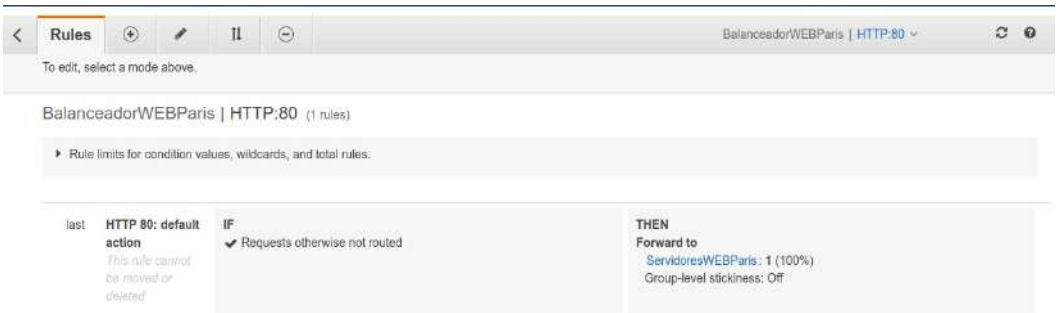
Una vez creado podemos comprobar en el target group que tenemos con Healtthy todas las instancias añadidas:



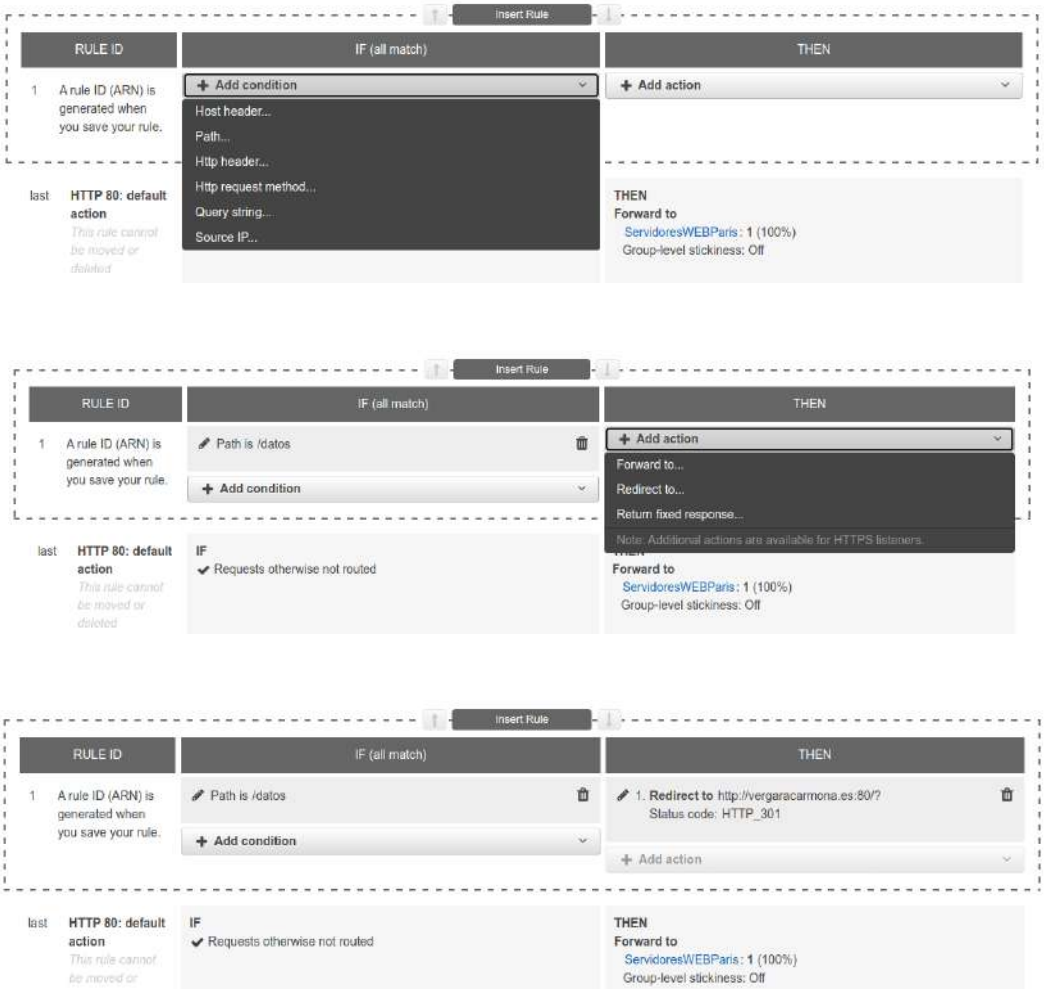
Para acceder al balanceaor nos da un nombre DNS. Como el algoritmo es Round Robin, cada vez que entremos a la dirección nos aparecerá el siguiente servidor web, por orden.



Podemos definir en Listeners distintos puertos. En cada puerto podemos editar las reglas



Por ejemplo, puede especificar que si se intenta entrar en /datos se redirecciones a otra página.

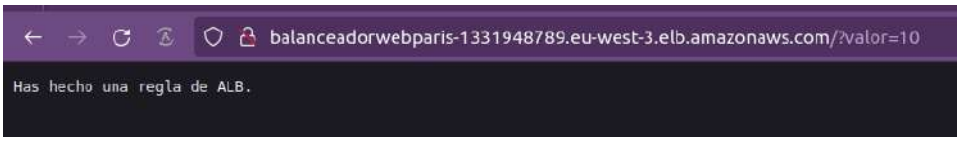


Con lo que si intento entrar en la url con /datos se redirecciona a la web indicada.

Las reglas se van consultando de arriba a abajo, si la primera se cumple no revisará las siguientes. Por eso aparece Insert Rule cuando le damos a añadir entre cada regla:

Por ejemplo, también se pueden insertar query. En este caso la respuesta es un error con un texto

Y si cumple la regla lo redirige al error



12.3. - Linea de comandos ALBs

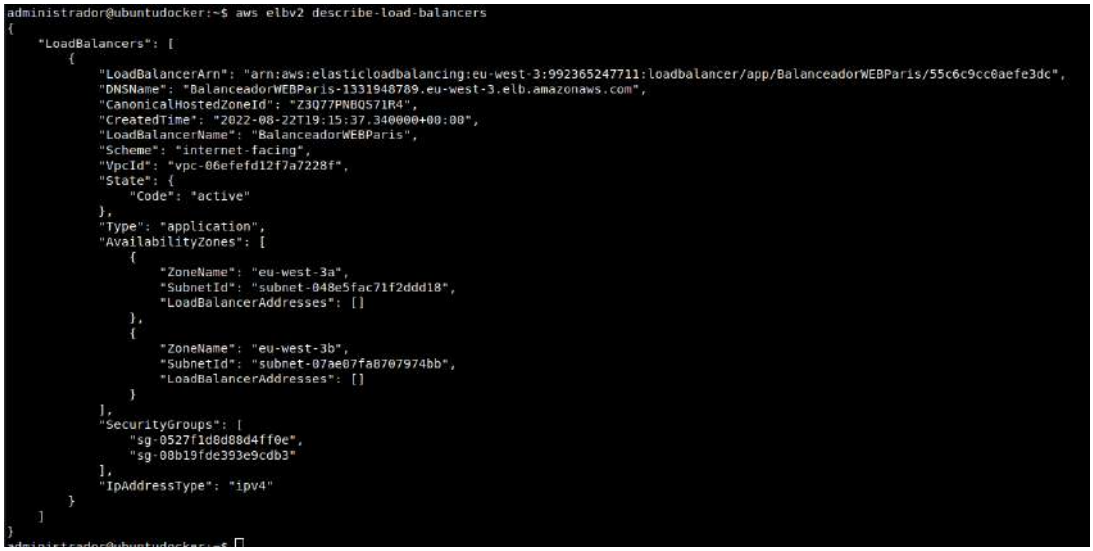
Opciones de aws elbv2

add-listener-certificates
- add-tags
- create-listener
- create-load-balancer
- create-rule
- create-target-group
- delete-listener
- delete-load-balancer
- delete-rule
- delete-target-group
- deregister-targets
- describe-account-limits
- describe-listener-certificates
- describe-listeners
- describe-load-balancer-attributes
- describe-load-balancers
- describe-rules
- describe-ssl-policies
- describe-tags
- describe-target-group-attributes
- describe-target-groups
- describe-target-health
- help
- modify-listener
- modify-load-balancer-attributes
- modify-rule
- modify-target-group
- modify-target-group-attributes
- register-targets
- remove-listener-certificates
- remove-tags
- set-ip-address-type

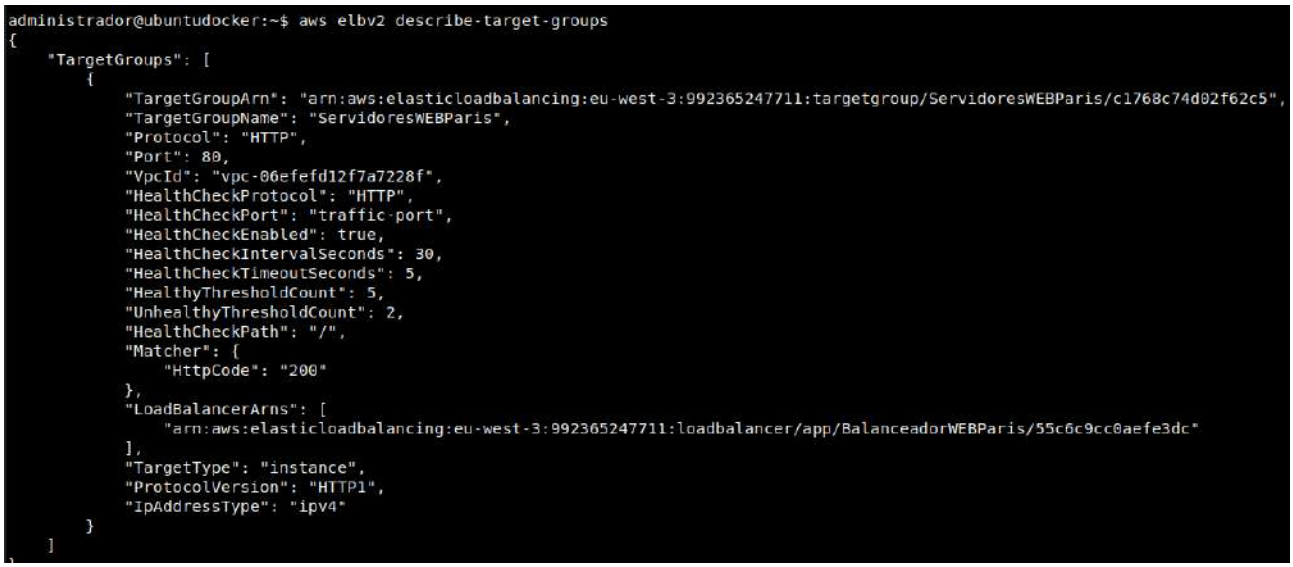
set-rule-priorities
set-security-groups
set-subnets
wait

Por ejemplo:

aws elbv2 describe-load-balancers

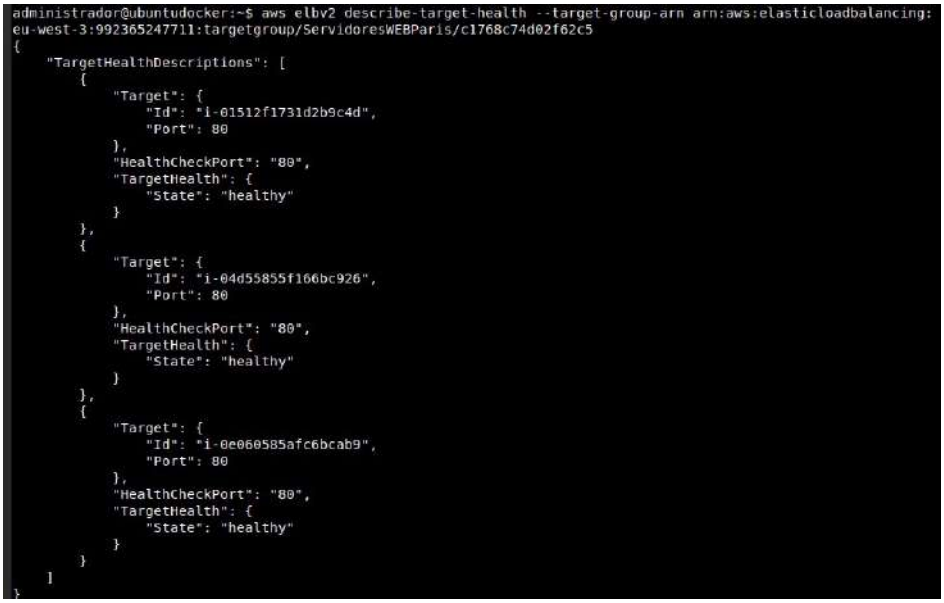


aws elbv2 describe-target-groups



O también se puede preguntar como está un target group. Este comando necesitará que se indique la url que utiliza Amazon para identificar objetos, que es la arn:

aws elbv2 describe-target-health --target-group-arn arn:aws:elasticloadbalancing:eu-west-3:992365247711:targetgroup/ServidoresWEBParis/c1768c74d02f62c5



aws elbv2 describe-listeners --load-balancer-arn arn:aws:elasticloadbalancing:eu-west-3:992365247711:loadbalancer/app/BalanceadorWEBParis/55c6c9cc0aefe3dc

aws elbv2 describe-rules --listener-arn arn:aws:elasticloadbalancing:eu-west-3:992365247711:listener/app/BalanceadorWEBParis/55c6c9cc0aefe3dc/12aa82e9221fed60

Cuando tengo identificada una regla concreta de listener puedo borrarla

aws elbv2 delete-rule --rule-arn arn:aws:elasticloadbalancing:eu-west-3:992365247711:listener-rule/app/BalanceadorWEBParis/55c6c9cc0aefe3dc/12aa82e9221fed60/76519d748f2250a1

Y lo podemos comprobar

12.4. - Crear el balanceador de carga NLB

El Network Loader Balancer va por la capa 4 de transporte.

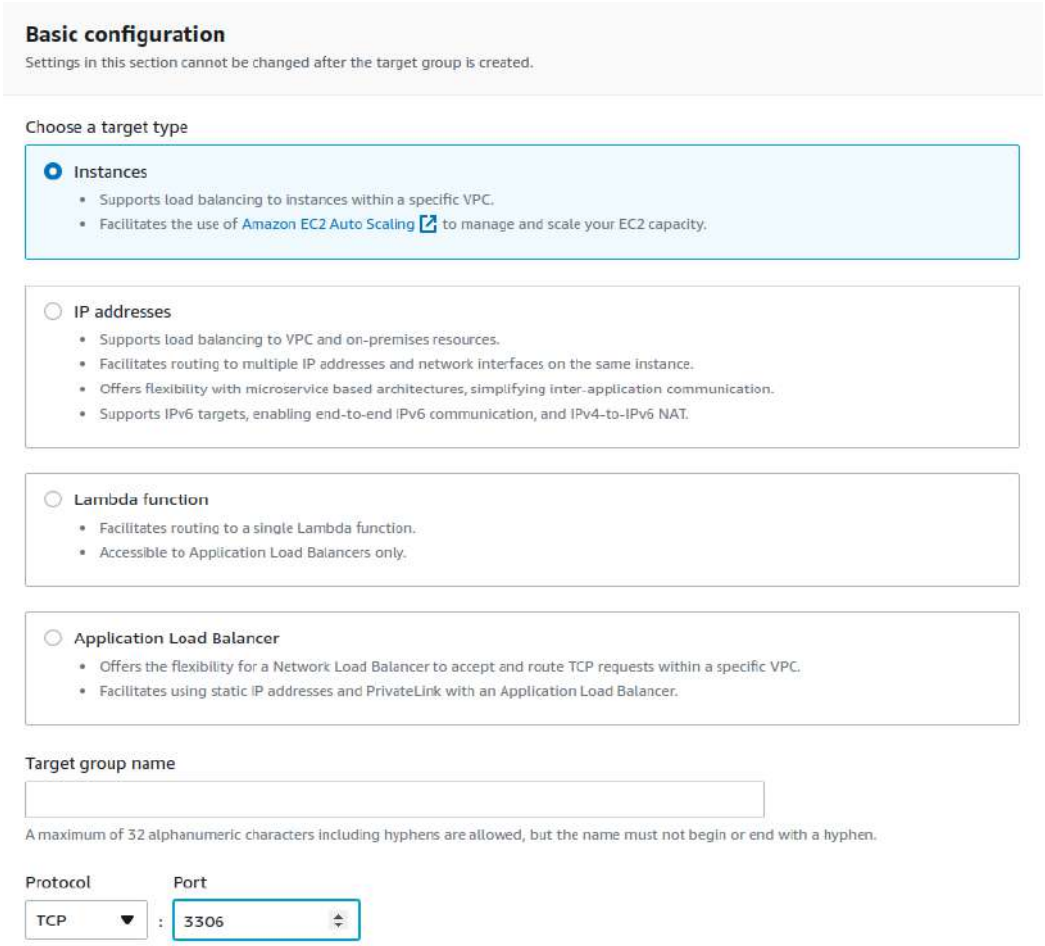
Para el ejemplo creamos dos instancias en dos subredes distintas

ServidorParisPruebas1AmazonLinux	i-0e060585afc6bcab9	Running		t2.micro	2/2 checks passed	No alarms	eu-west-3b
ServidorParisPruebas2AmazonLinux	i-0465e6b3a4a57e53d	Stopped		t2.micro	–	No alarms	eu-west-3b
ServidorParisPruebas3AmazonLinux	i-06b3474f683bbc51e	Stopped		t2.micro	–	No alarms	eu-west-3b
ServidorParisPruebas4AmazonLinux	i-0779d13421d6908af	Running		t2.micro	2/2 checks passed	No alarms	eu-west-3a

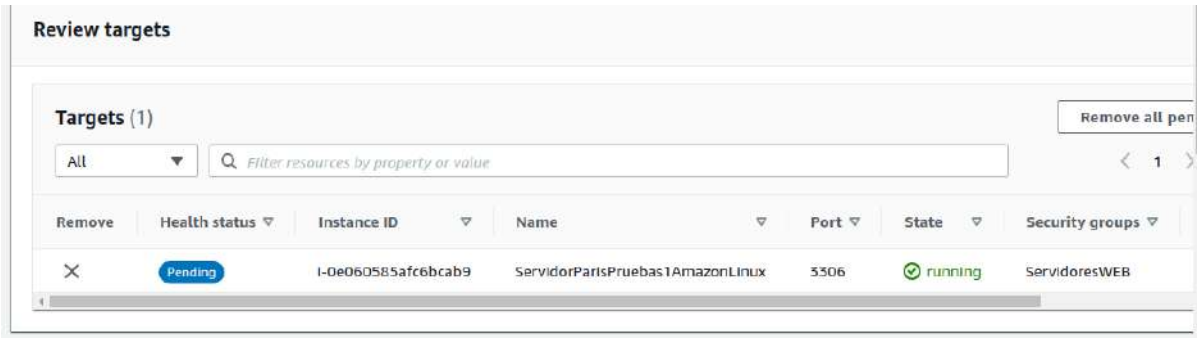
Y en ambas se instala mariadb y mariadb-server y se abre el puerto 3306 en el grupo de seguridad.

Se crea un usuario a cada BBDD y se le da privilegios.

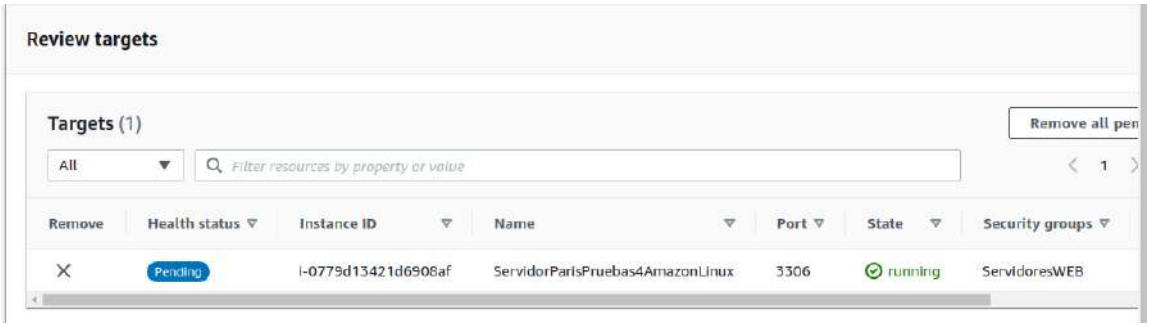
Ahora ya creamos el target group, basándonos en Instancias y con el puerto 3306



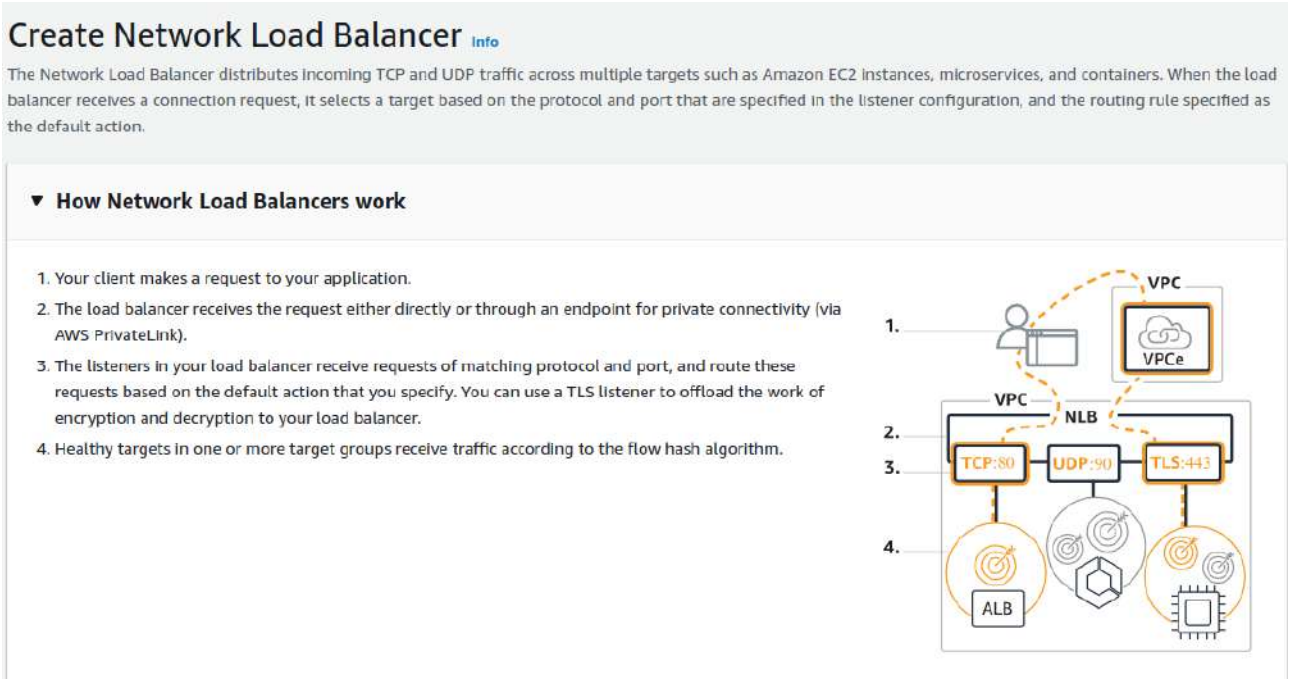
En esta ocasión solo pondremos una instancia por target group. El primero:



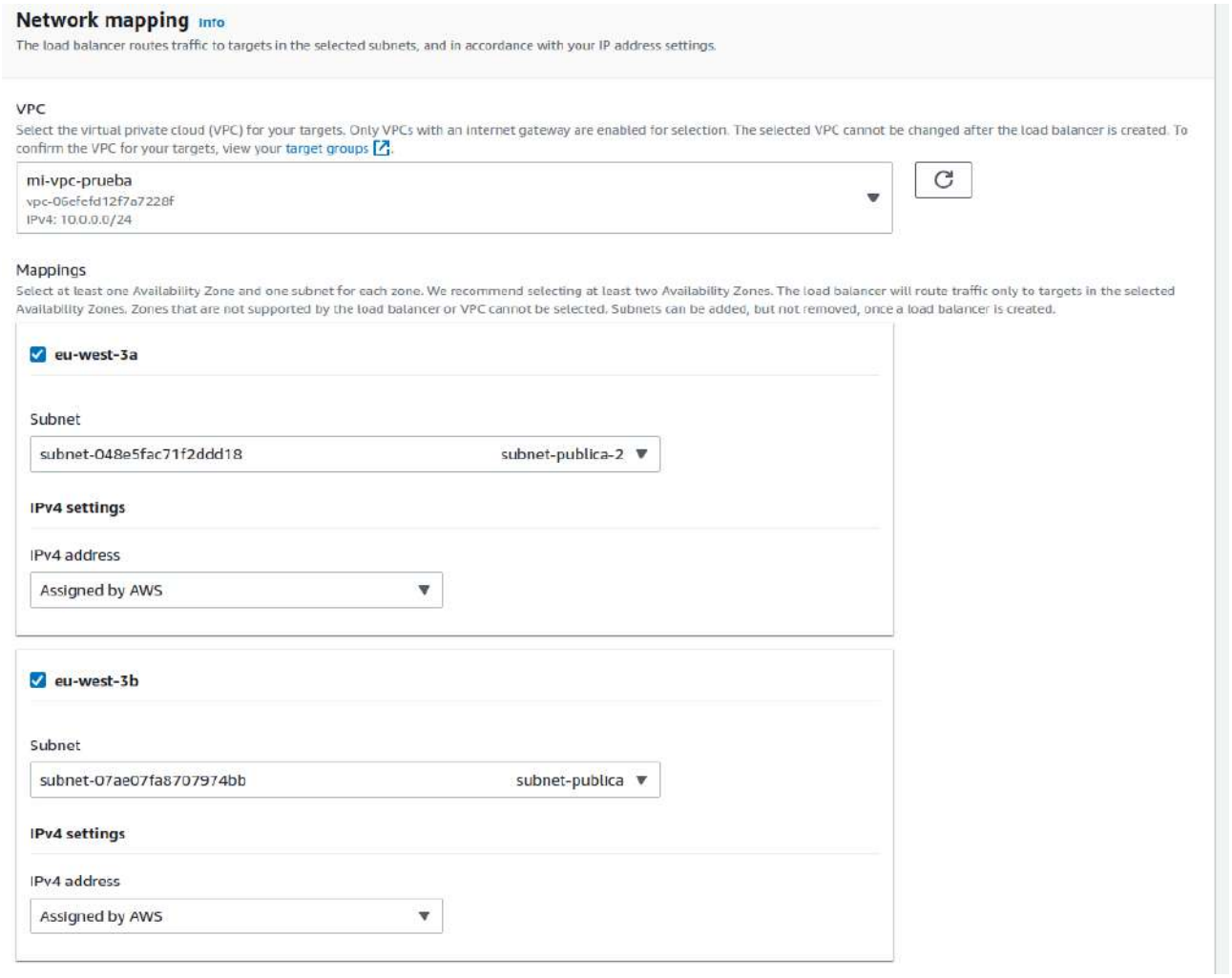
Y el segundo Target Group:



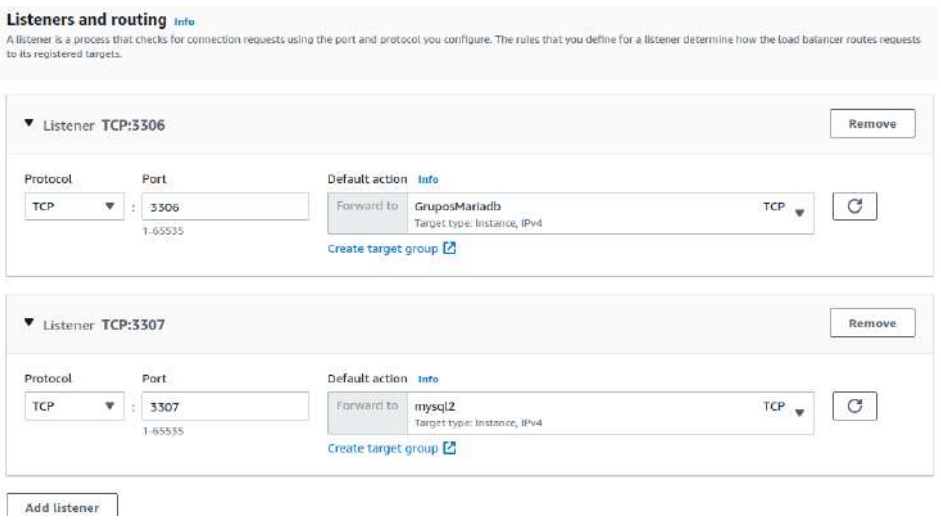
Ahora creamos el NLB



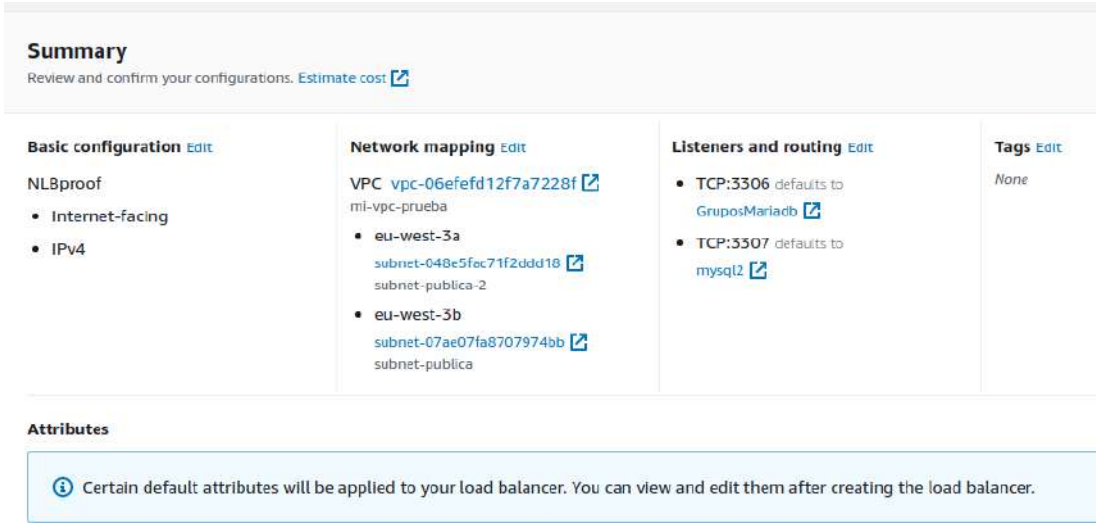
Tengo que escoger la VPC y las subnets involucradas



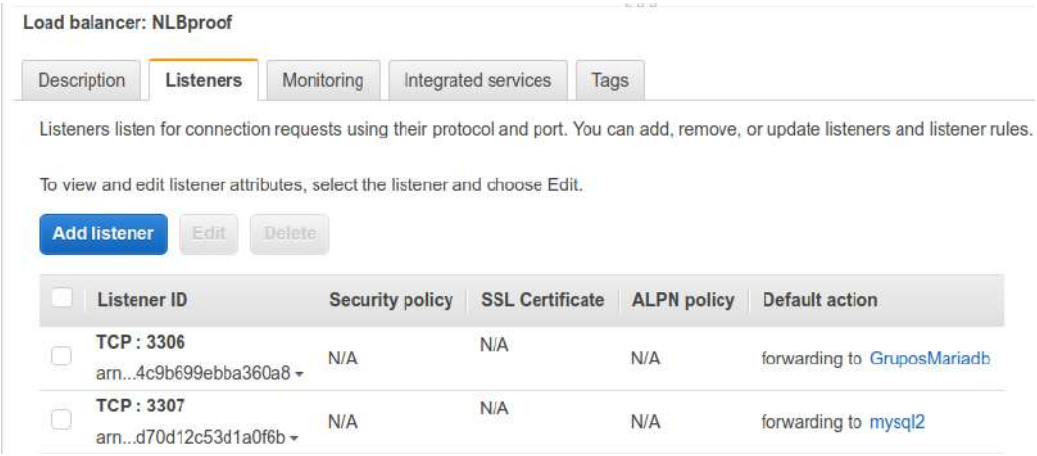
En el listener ponemos 2 puertos, 3306 y 3307. Los puertos de entrada es indiferente, pero iremos a los puertos 3306 de cada uno de los targets, que tienen una instancia distinta.



El resumen:

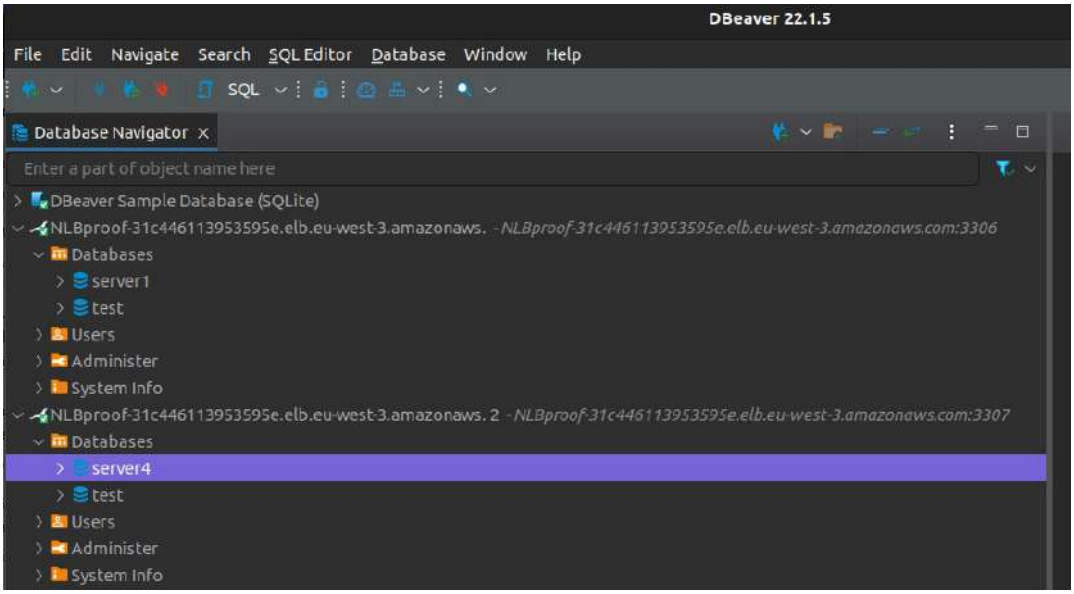


En estos listener no hay reglas como en el ALB



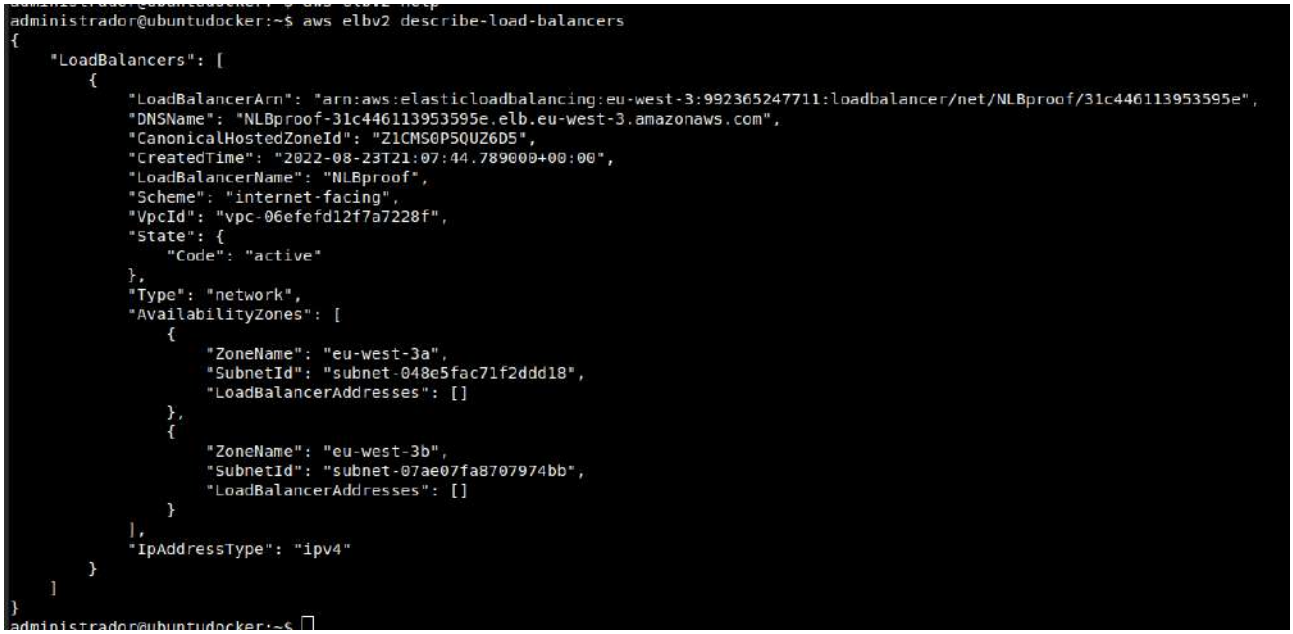
En los servicios integrados, además del Global Accelerator también tenemos el VPC Endpoint Services y el Traffic Mirroring..

Ahora para probar el balanceador tendremos que entrar en los servidores por los dos puertos en el mismo DNS del balanceador

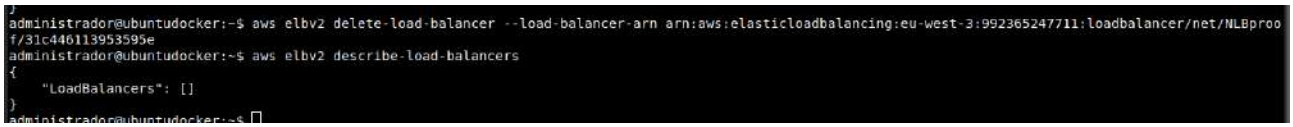

DBeaver 22.1.5
File Edit Navigate Search SQL Editor Database Window Help
SQL
Database Navigator
Enter a part of object name here
DBeaver Sample Database (SQLite)
NLBproof-31c446113953595e.elb.eu-west-3.amazonaws. - NLBproof-31c446113953595e.elb.eu-west-3.amazonaws.com:3306
Databases
server1
test
Users
Administer
System Info
NLBproof-31c446113953595e.elb.eu-west-3.amazonaws. 2 - NLBproof-31c446113953595e.elb.eu-west-3.amazonaws.com:3307
Databases
server4
test
Users
Administer
System Info

12.5. - Linea de comandos NLB

También es con aws elbv2. Por ejemplo, describe-load-balancers



Para borrar es con delete-load-balancer con la opción para indicar el arn –load-balancer-arn



Ahora pido información de los target groups

```
administrador@ubuntudocker:~$ aws elbv2 describe-target-groups
{
    "TargetGroups": [
        {
            "TargetGroupArn": "arn:aws:elasticloadbalancing:eu-west-3:992365247711:targetgroup/GruposMariadb/7061712c9cc52469",
            "TargetGroupName": "GruposMariadb",
            "Protocol": "TCP",
            "Port": 3306,
            "VpcId": "vpc-06efefd12f7a7228f",
            "HealthCheckProtocol": "TCP",
            "HealthCheckPort": "traffic-port",
            "HealthCheckEnabled": true,
            "HealthCheckIntervalSeconds": 30,
            "HealthCheckTimeoutSeconds": 10,
            "HealthyThresholdCount": 3,
            "UnhealthyThresholdCount": 3,
            "LoadBalancerArns": [],
            "TargetType": "instance",
            "IpAddressType": "ipv4"
        },
        {
            "TargetGroupArn": "arn:aws:elasticloadbalancing:eu-west-3:992365247711:targetgroup/mysql2/ee24752c9523dfeb",
            "TargetGroupName": "mysql2",
            "Protocol": "TCP",
            "Port": 3306,
            "VpcId": "vpc-06efefd12f7a7228f",
            "HealthCheckProtocol": "TCP",
            "HealthCheckPort": "traffic-port",
            "HealthCheckEnabled": true,
            "HealthCheckIntervalSeconds": 30,
            "HealthCheckTimeoutSeconds": 10,
            "HealthyThresholdCount": 3,
            "UnhealthyThresholdCount": 3,
            "LoadBalancerArns": [],
            "TargetType": "instance",
            "IpAddressType": "ipv4"
        }
    ]
}
administrador@ubuntudocker:~$
```

Y borro primero uno

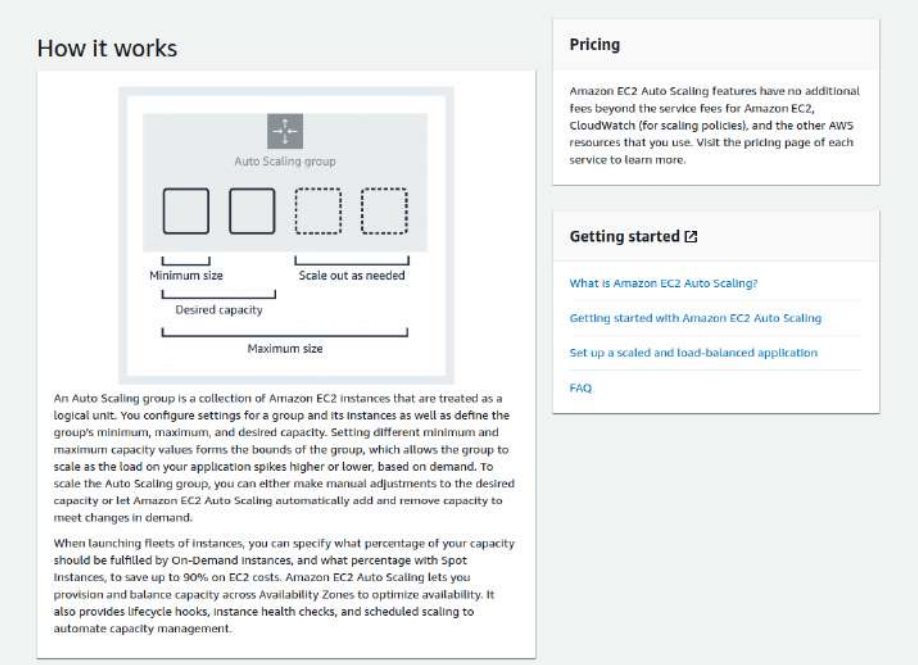
```
administrador@ubuntudocker:~$ aws elbv2 delete-target-group --target-group-arn arn:aws:elasticloadbalancing:eu-west-3:992365247711:targetgroup/GruposMariadb/
7061712c9cc52469
administrador@ubuntudocker:~$ aws elbv2 describe-target-groups
{
    "TargetGroups": [
        {
            "TargetGroupArn": "arn:aws:elasticloadbalancing:eu-west-3:992365247711:targetgroup/mysql2/ee24752c9523dfeb",
            "TargetGroupName": "mysql2",
            "Protocol": "TCP",
            "Port": 3306,
            "VpcId": "vpc-06efefd12f7a7228f",
            "HealthCheckProtocol": "TCP",
            "HealthCheckPort": "traffic-port",
            "HealthCheckEnabled": true,
            "HealthCheckIntervalSeconds": 30,
            "HealthCheckTimeoutSeconds": 10,
            "HealthyThresholdCount": 3,
            "UnhealthyThresholdCount": 3,
            "LoadBalancerArns": [],
            "TargetType": "instance",
            "IpAddressType": "ipv4"
        }
    ]
}
administrador@ubuntudocker:~$
```

Y luego el otro

```
administrador@ubuntudocker:~$ aws elbv2 delete-target-group --target-group-arn arn:aws:elasticloadbalancing:eu-west-3:992365247711:targetgroup/mysql2/ee24752
c9523dfeb
administrador@ubuntudocker:~$ aws elbv2 describe-target-groups
{
    "TargetGroups": []
}
administrador@ubuntudocker:~$
```

TEMA 13 - EC2 Grupos de Autoescalada. AutoScalling Groups

Los grupos de autoescalada son un conjunto de instancias EC2 que van a poder escalar hacia arriba o hacia abajo dependiendo de ciertas caracterísitcas. Por ejemplo, cuando se dispare el consumo de CPU.



Un grupo autoescalada tendrá un mínimo de instancias con un tamaño mínimo, un tamaño deseado y escalará cuando se necesite hasta el tamaño máximo que indiquemos.

Es muy útil cuando se tienen picos de tráfico temporal. Por ejemplo, una e-commerce que triplica las visitas los fines de semana, con el autoescalado se adaptaría al tráfico.

Las reglas del escalado se pueden crear basándose en la **CPU**, según las métricas del Amazon **CloudWatch** y/o usando un **Elastic Load Balancing**.

13.1. - Diferencias entre Launch Templates y Launch Configuration

Las configuraciones de lanzamientos sirven para indicar las características de las instancias que se usarán para el grupo de autoescalada.

Amazon recomienda que en vez de utilizar Launch Configuration se utilice las plantillas de Launch Templates.

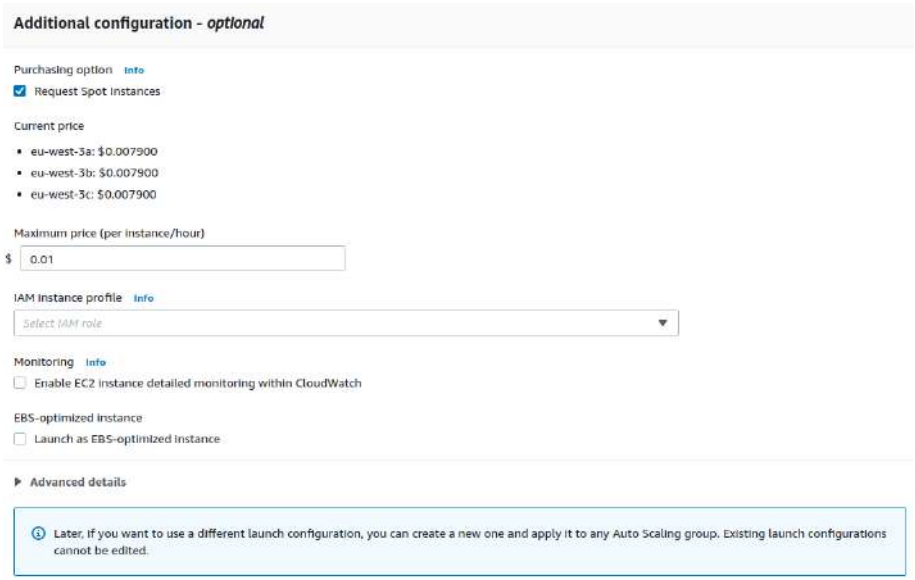
Launch Configuration no tiene tantas características, es más simple. Además solo se pueden utilizar con grupos de autoescalada mientras Launch Templates se puede utilizar también para crear instancias.

Es muy posible que AWS acabe eliminando las Launch Configuration.

13.2. - Crear una Launch Configuration

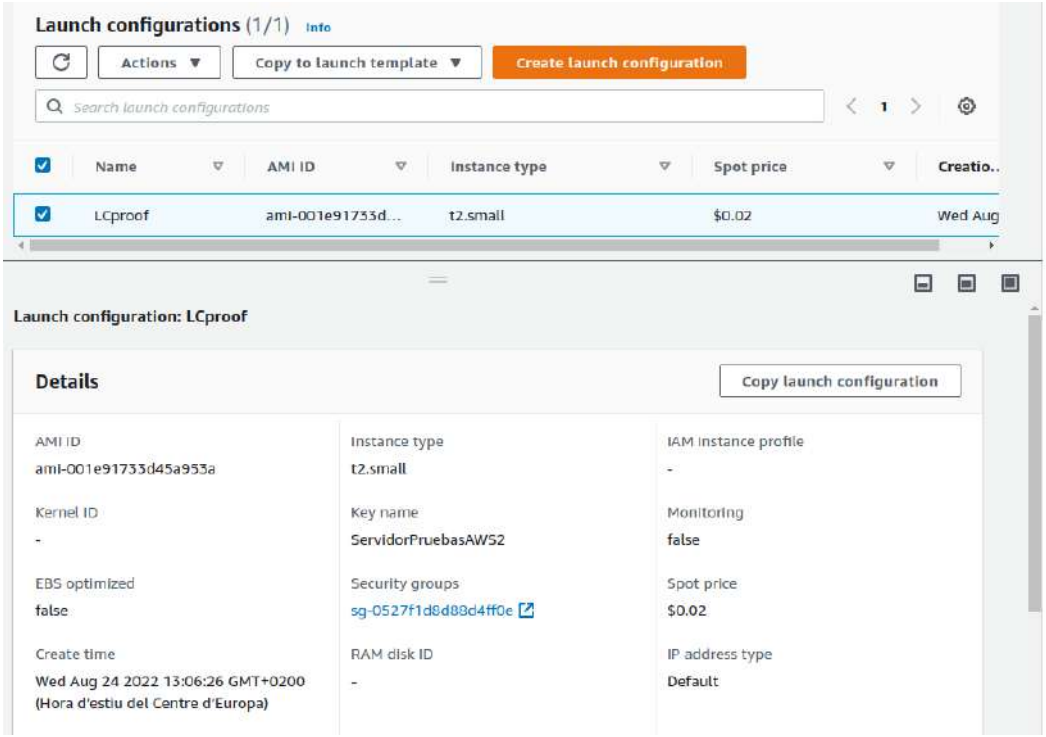
Tendremos que indicar el nombre, escoger el AMI (Cojo la personalizada con Apache) y el Tipo de instancia (escojo t2.small que es gratuita).

Luego, da la opción de que sean Spot Instances para ser más barato, el precio máximo a gastar por hora, el Perfil IAM (Identity Access Management), si queremos que la monitorización en CloudWatch sea detallada y la optimización EBS.



Deberemos indicar los volumenes, el grupo de seguridad y el archivo KeyPair para loguearse (Esto nos pide confirmación en un check). Y ya creamos,

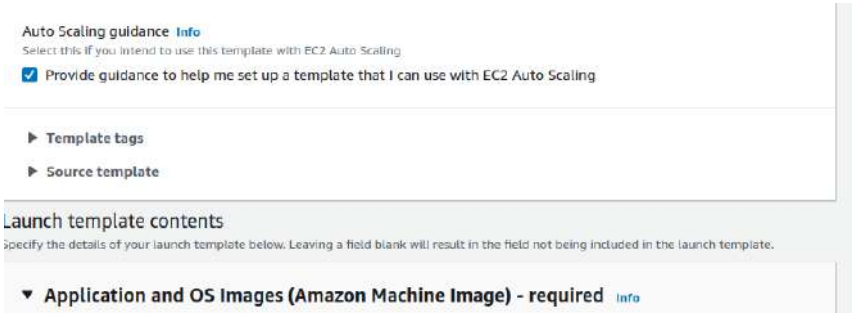
Todas las instancias tendrán estos parámetros comunes.



13.3. - Crear una Launch Template

Ahora crearemos una plantilla basada en el AMIs para poder utilizarla también en los grupos de autoescalado.

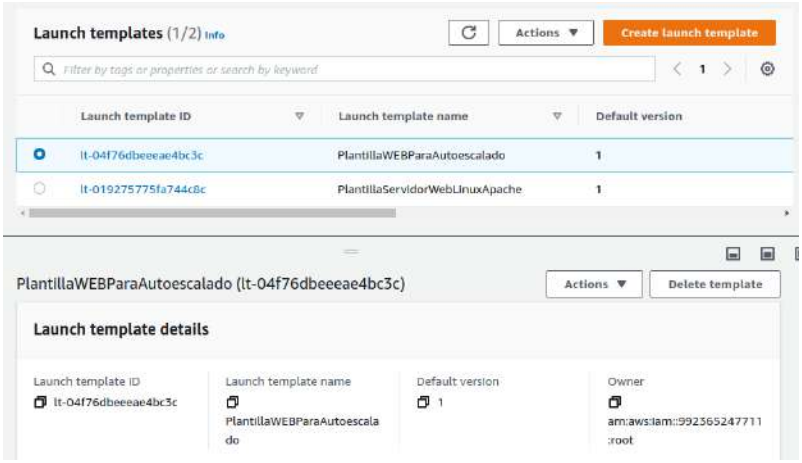
Al inicio hay un check para que la opciones se ajusten a una guía para el autoescalado



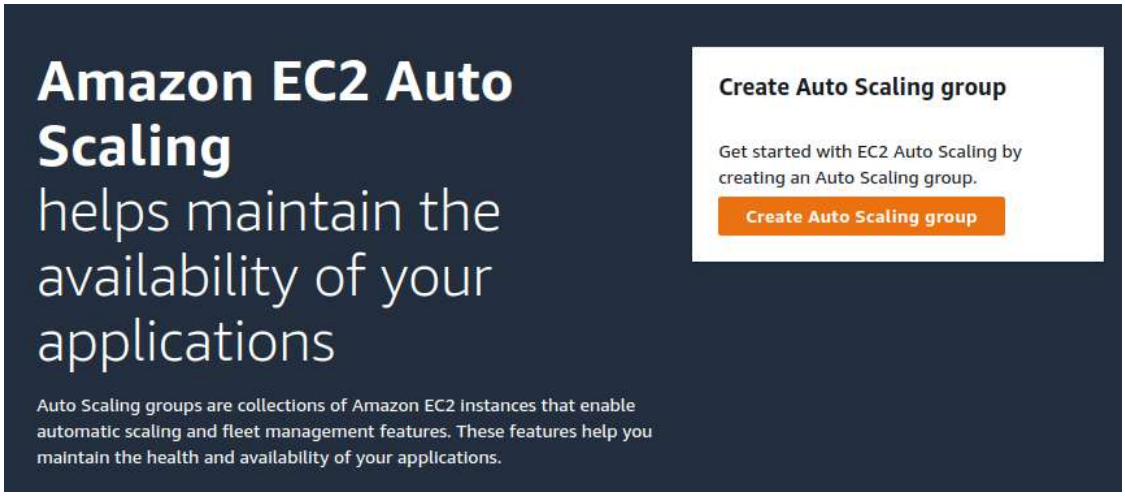
Selecciono la AMI que quiero para la plantilla, la AMIs con Linux y Apache.

Escogemos el tipo de instancia (t2.small), el key pair, la subred, el grupo de seguridad, el almacenamiento,

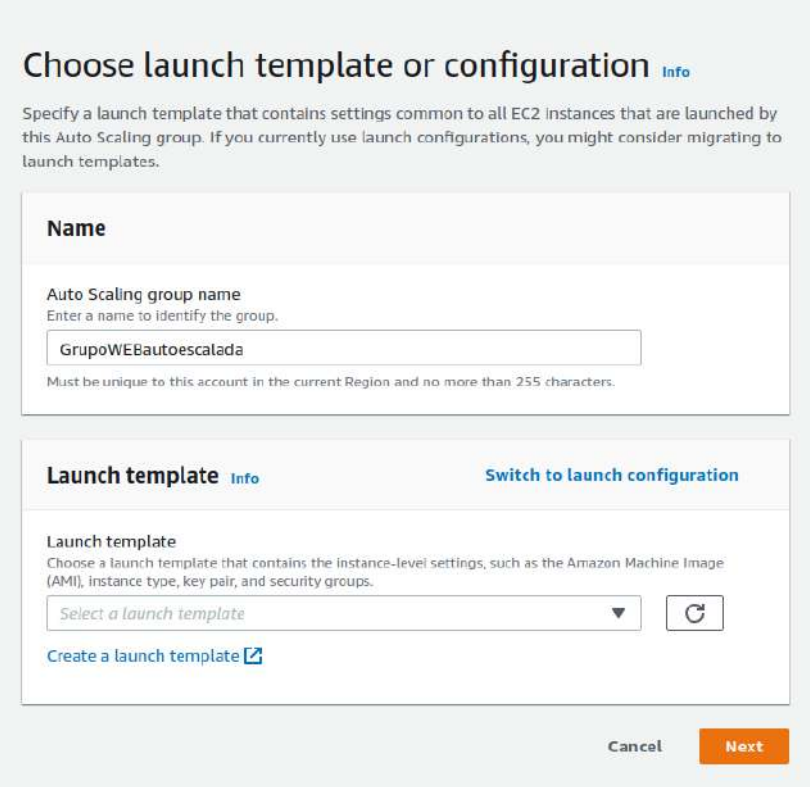
Si dejamos opciones vacías nos lo pedirá cuando lancemos el autoescalado.



13.4. - Crear un grupo de autoescalada



En Launch Template se puede cambiar a Launch configuration

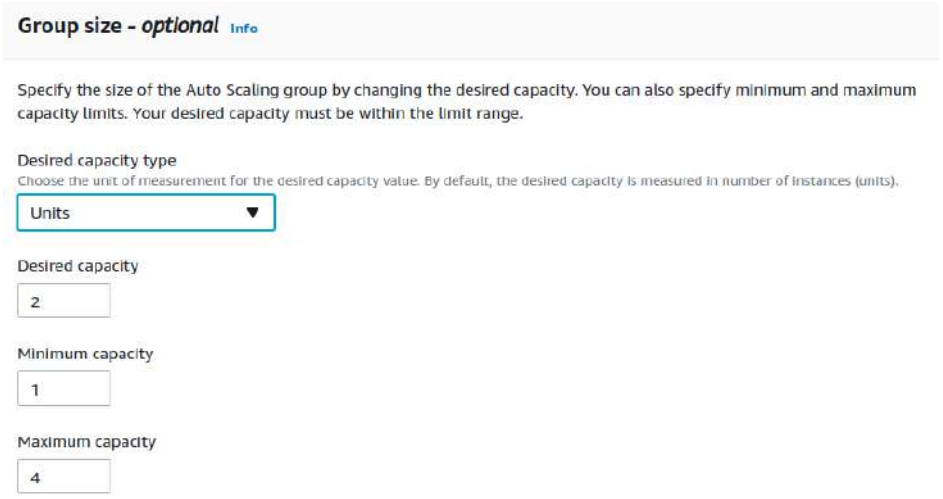


Por defecto aparece Launch template. En el **paso 1** seleccionamos la creada y hace un resumen de las características básicas.

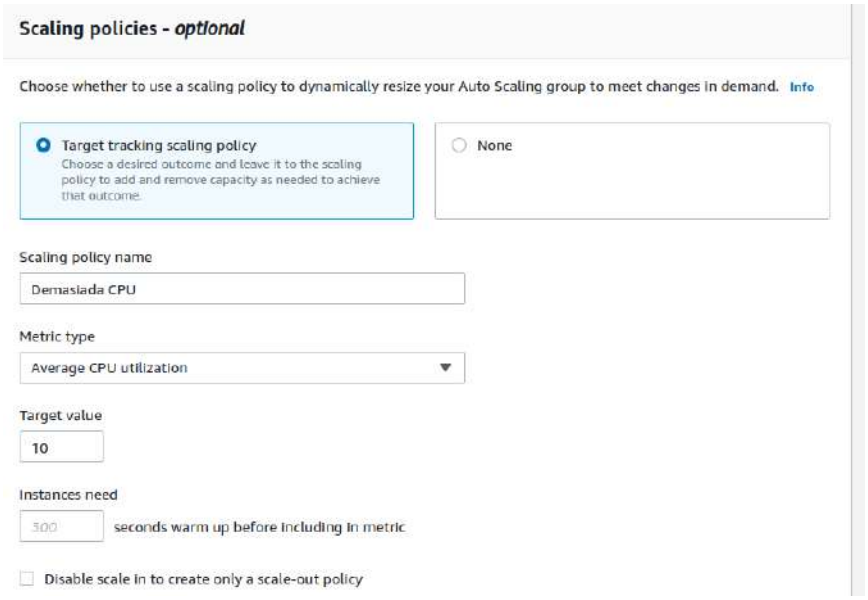
Luego, en el **paso 2,** se pueden cambiar algunas opciones como la VPC y las Azs (Se pueden indicar varias, escojo las públicas). También podemos indicar los requísitos de la instancia.

En el **paso 3** es el balanceador de carga, podemos indicar sin, con uno que exista o con uno que creemos. En Health check se creará un grupo para tan solo comprobar si las instancias se autoescalan estresandoles cada x tiempo. Por último se podrán añadir métricas adicionales en CloudWatch.

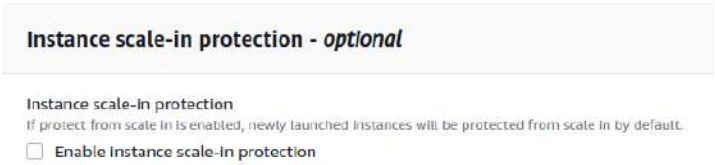
El **paso 4** es el más importante, es la configuración del tamaño del grupo y las reglas de escalada.



En las políticas se puede indicar el nombre, el tipo de métrica, el valor, el tiempo que espera para contabilizar el valor y si queremos no permitir que se autoescale hacia abajo.



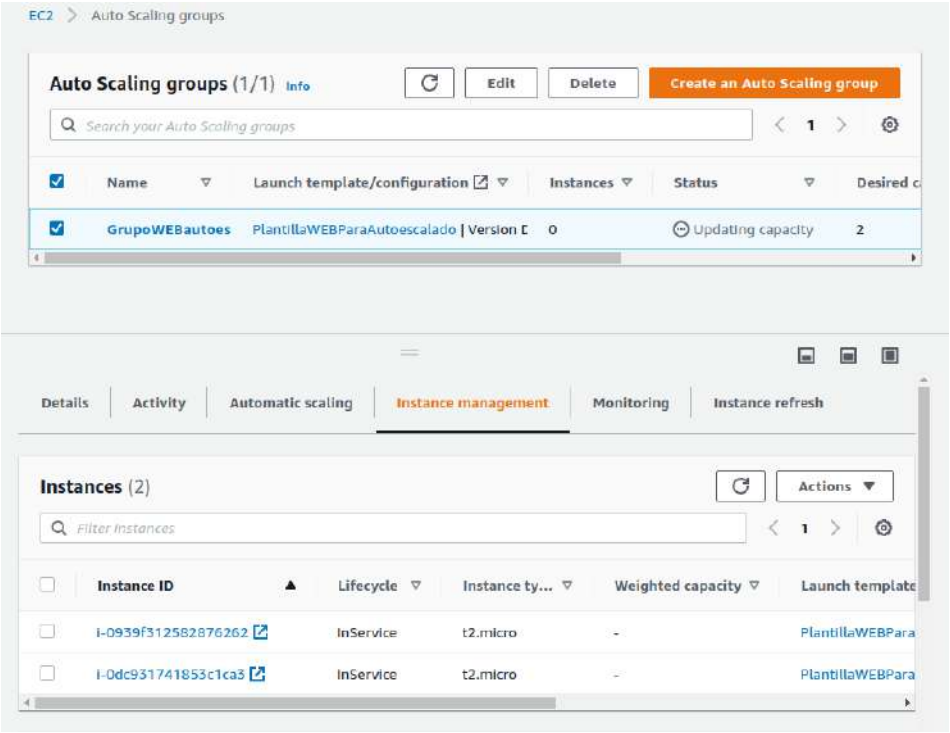
Además, podemos decir que no se tengan en cuenta los cierres cuando hagamos una política de scale-in, es decir, que no se elimine una instancia en el desescalado.



En el **paso 5** es que envíe notificaciones cuando efectúe autoescalado. Se envían por SNS (Simple Notification Service), se necesitan crear Topics.

Por último, **paso 6**, se pueden crear Tags específicas, como en todos los objetos y componentes de AWS. El **paso 7** es tan solo una revisión de las opciones seleccionadas.

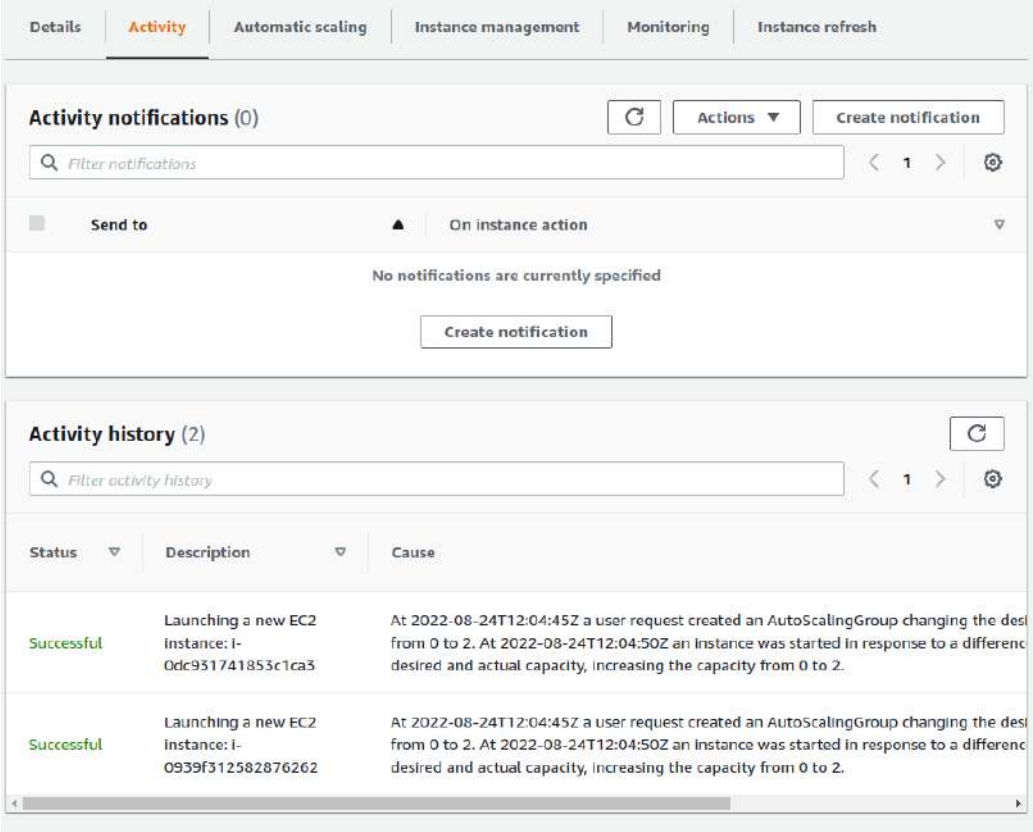
Una vez creado el grupo de autoescalada empezará a crear las instancias. CUIDADO. Cuando inicie la creación de grupo de autoescalada tenía recien creada la AMIs y el Launch Template seleccionado. Me ha dado problemas para acabar de crear el grupo de autoescalada y he tenido que volver al paso 1 para sincronizar los datos.



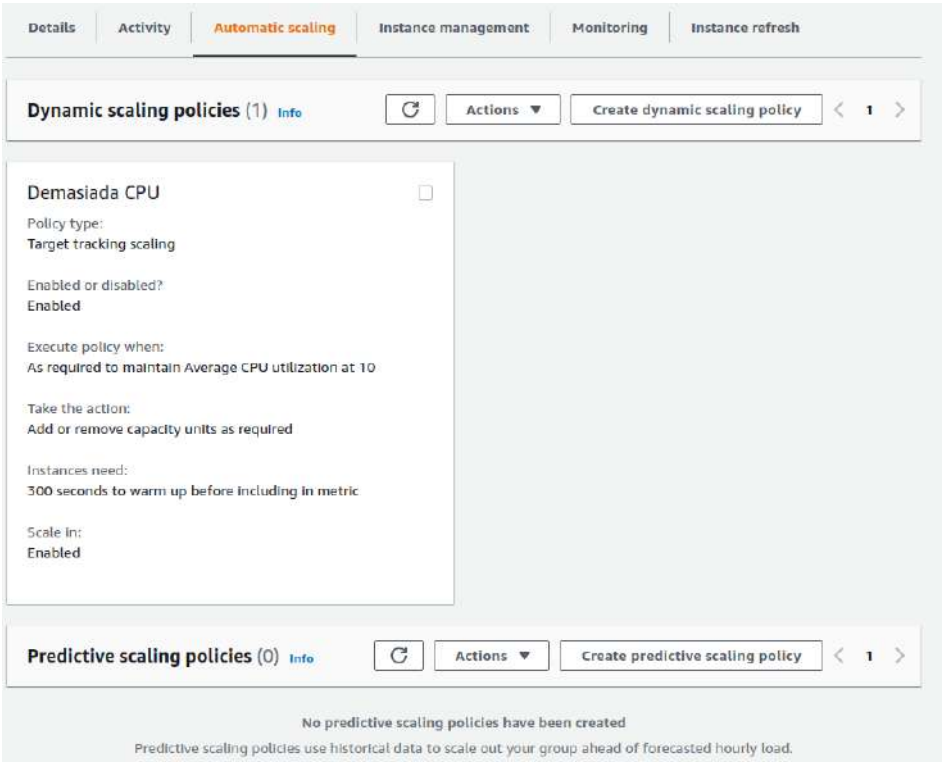
13.5. - Propiedades grupo de autoescalada

Detalles – Con un resumen de las características del grupo. Se pueden editar en cada uno de los grupos.

Activity – Aparece un hisptorial de los sucesos. Aquí se pueden activar notificaciones.



Automatic scaling – Se puede indicar el autoescalado.



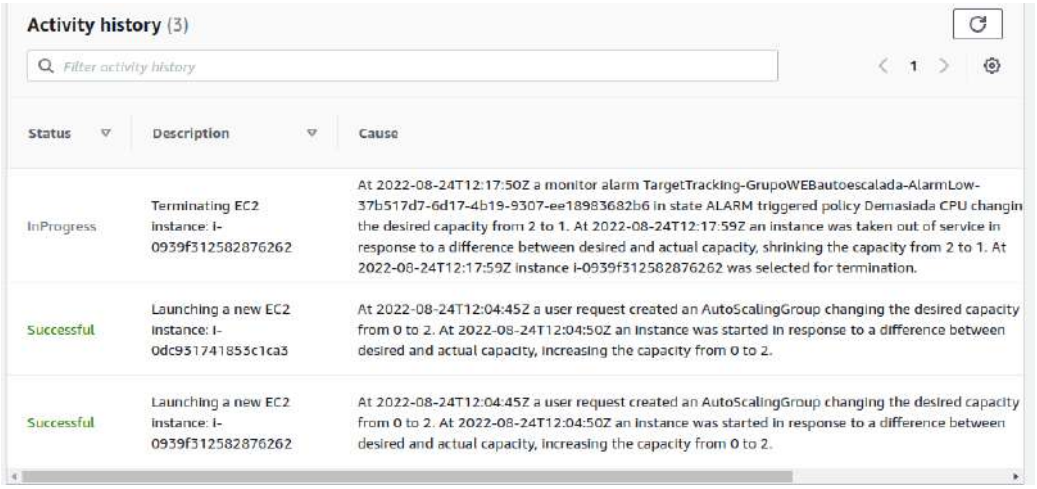
Instance management – Gestión de las instancias que conforman el grupo. Se puede añadir un Lifecycle Hooks, lanzando acciones concretas cuando se cambia el estado del objeto.

Monitoring – Monitorización de autoescalada y de las instancias EC2

Instance refresh – Apareceran las actualizaciones de las instancias. Por ejemplo, para poner actualizaciones de la aplicación que tengamos en las instancias.

13.6. - Comportamiento del autoescalado. Prueba

Al pasar un tiempo (300s), en activity se puede ver como ha terminado con una instancia



Es debido a qué para mantener menos del 10% de CPU indicado en la política «Demasiada CPU» se puede con tan solo una instancia.

Vamos a la instancia que está corriendo y le lanzamos un proceso pesado. Nos conectamos a través de SSH y utilizaremos un programa para estresar la CPU. Stress de Epel (Es de Red Hat, allí se puede instalar directamente con yum install). En ubuntu, debian y demás es otra forma para instalarlo.

Como tenemos Amazon Linux 2 tendremos que instalar un programa que nos permitirá instalar repositorios externos, como EPEL:

```
sudo yum install -y amazon-linux-extras
```

Ya está instalado. Ahora instalamos el respositorio EPEL

```
sudo amazon-linux-extras install epel
```

Y ahora ya podemos instalar el programa en concreto, que se llama stress.

```
Sudo yum install -y stress
```

Opciones de stress

`stress' imposes certain types of compute stress on your system

```
Usage: stress [OPTION [ARG]] ...
-?, --help          show this help statement
--version           show version statement
-v, --verbose       be verbose
-q, --quiet         be quiet
-n, --dry-run       show what would have been done
-t, --timeout N     timeout after N seconds
--backoff N         wait factor of N microseconds before work starts
-c, --cpu N         spawn N workers spinning on sqrt()
-i, --io N          spawn N workers spinning on sync()
-m, --vm N          spawn N workers spinning on malloc()/free()
--vm-bytes B        malloc B bytes per vm worker (default is 256MB)
--vm-stride B       touch a byte every B bytes (default is 4096)
--vm-hang N         sleep N secs before free (default none, 0 is inf)
--vm-keep           redirty memory instead of freeing and reallocating
-d, --hdd N         spawn N workers spinning on write()/unlink()
--hdd-bytes B       write B bytes per hdd worker (default is 1GB)
```

Example: stress --cpu 8 --io 4 --vm 2 --vm-bytes 128M --timeout 10s

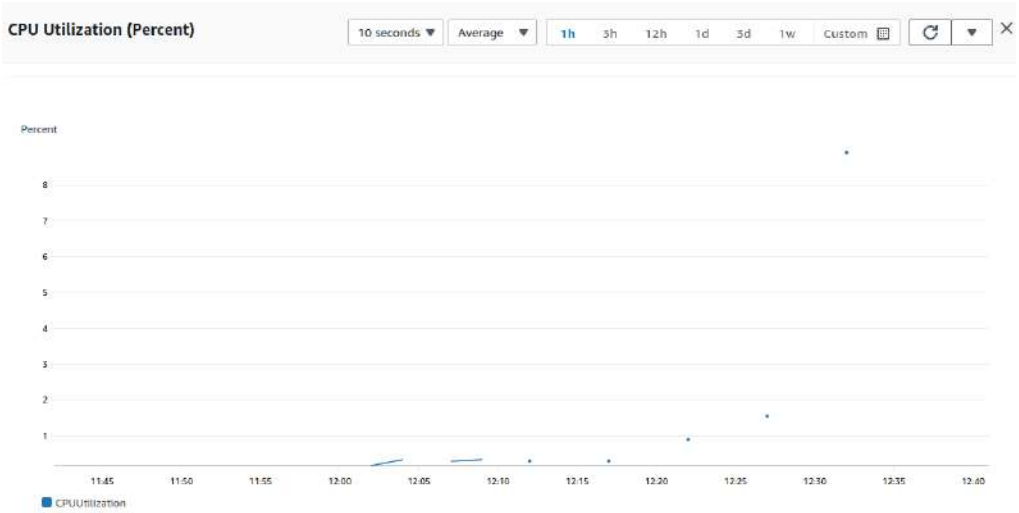
Note: Numbers may be suffixed with s,m,h,d,y (time) or B,K,M,G (size).

Vamos a estressar con más de 10 hooks de CPU

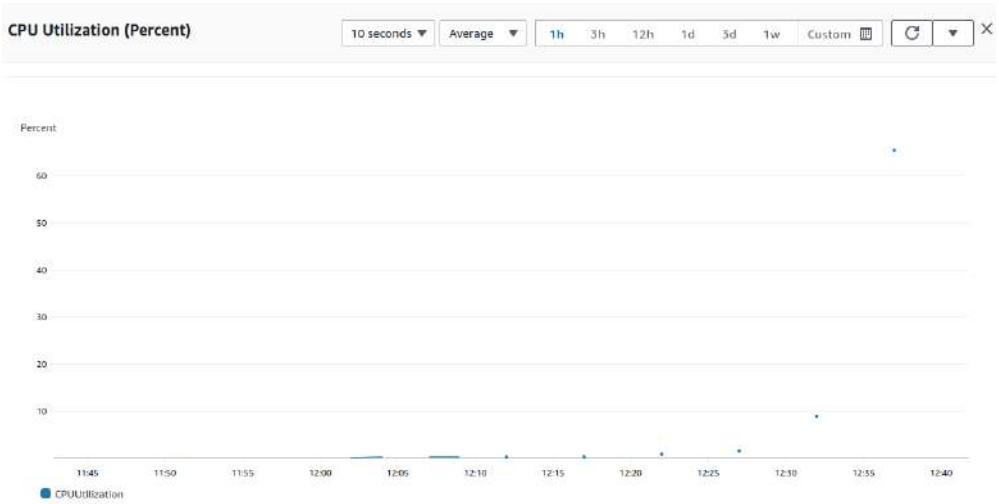
```
stress -c 10
```

```
[root@ip-10-0-0-30 ~]# stress -c 10
stress: info: [4173] dispatching hogs: 10 cpu, 0 io, 0 vm, 0 hdd
```

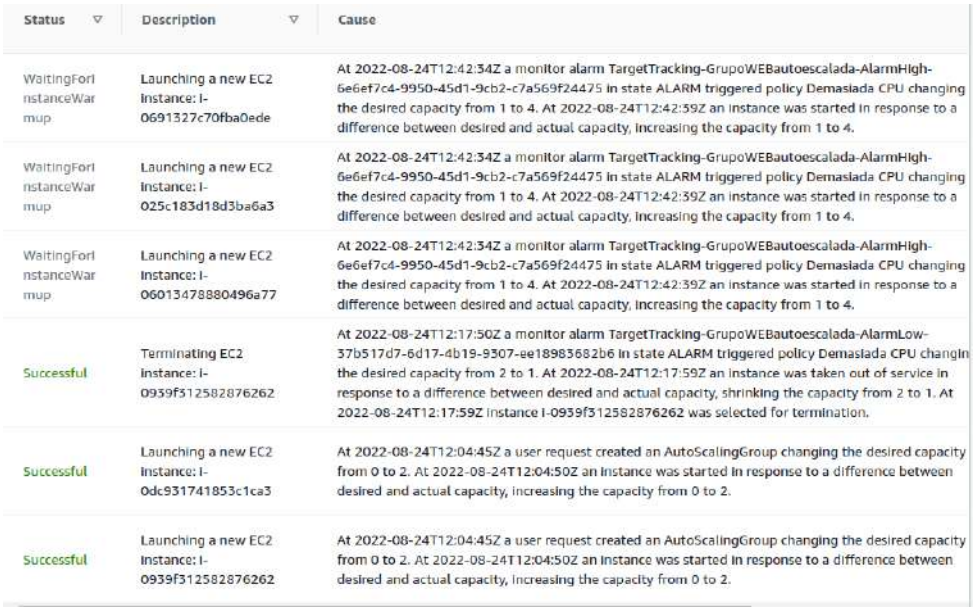
En el grupo de autoescalada, podemos monitorizar el CPU de las instancias EC2



Cuando este 300 segundos usando más del 10% de CPU creará las instancias que considere necesario para adaptarse a los recursos.



En Activity se puede ver como crea 3 instancias más que considera necesarias, según los parámetros dados.



Status	Description	Cause
WaitingForInstanceWarmup	Launching a new EC2 instance: i-0691327c70fba0ede	At 2022-08-24T12:42:34Z a monitor alarm TargetTracking-GrupoWEBautoescalada-AlarmHigh-6e6ef7c4-9950-45d1-9cb2-c7a569f24475 in state ALARM triggered policy Demasiada CPU changing the desired capacity from 1 to 4. At 2022-08-24T12:42:59Z an instance was started in response to a difference between desired and actual capacity, increasing the capacity from 1 to 4.
WaitingForInstanceWarmup	Launching a new EC2 instance: i-025c183d18d3ba6a3	At 2022-08-24T12:42:34Z a monitor alarm TargetTracking-GrupoWEBautoescalada-AlarmHigh-6e6ef7c4-9950-45d1-9cb2-c7a569f24475 in state ALARM triggered policy Demasiada CPU changing the desired capacity from 1 to 4. At 2022-08-24T12:42:59Z an instance was started in response to a difference between desired and actual capacity, increasing the capacity from 1 to 4.
WaitingForInstanceWarmup	Launching a new EC2 instance: i-0601347888049 6a77	At 2022-08-24T12:42:34Z a monitor alarm TargetTracking-GrupoWEBautoescalada-AlarmHigh-6e6ef7c4-9950-45d1-9cb2-c7a569f24475 in state ALARM triggered policy Demasiada CPU changing the desired capacity from 1 to 4. At 2022-08-24T12:42:59Z an instance was started in response to a difference between desired and actual capacity, increasing the capacity from 1 to 4.
Successful	Terminating EC2 instance: i-0939f312582876262	At 2022-08-24T12:17:50Z a monitor alarm TargetTracking-GrupoWEBautoescalada-AlarmLow-37b517d7-6d17-4b19-9307-ee18983682b6 in state ALARM triggered policy Demasiada CPU changin the desired capacity from 2 to 1. At 2022-08-24T12:17:59Z an instance was taken out of service in response to a difference between desired and actual capacity, shrinking the capacity from 2 to 1. At 2022-08-24T12:17:59Z instance i-0939f312582876262 was selected for termination.
Successful	Launching a new EC2 instance: i-0dc931741853c1ca3	At 2022-08-24T12:04:45Z a user request created an AutoScalingGroup changing the desired capacity from 0 to 2. At 2022-08-24T12:04:50Z an instance was started in response to a difference between desired and actual capacity, increasing the capacity from 0 to 2.
Successful	Launching a new EC2 instance: i-0939f312582876262	At 2022-08-24T12:04:45Z a user request created an AutoScalingGroup changing the desired capacity from 0 to 2. At 2022-08-24T12:04:50Z an instance was started in response to a difference between desired and actual capacity, increasing the capacity from 0 to 2.

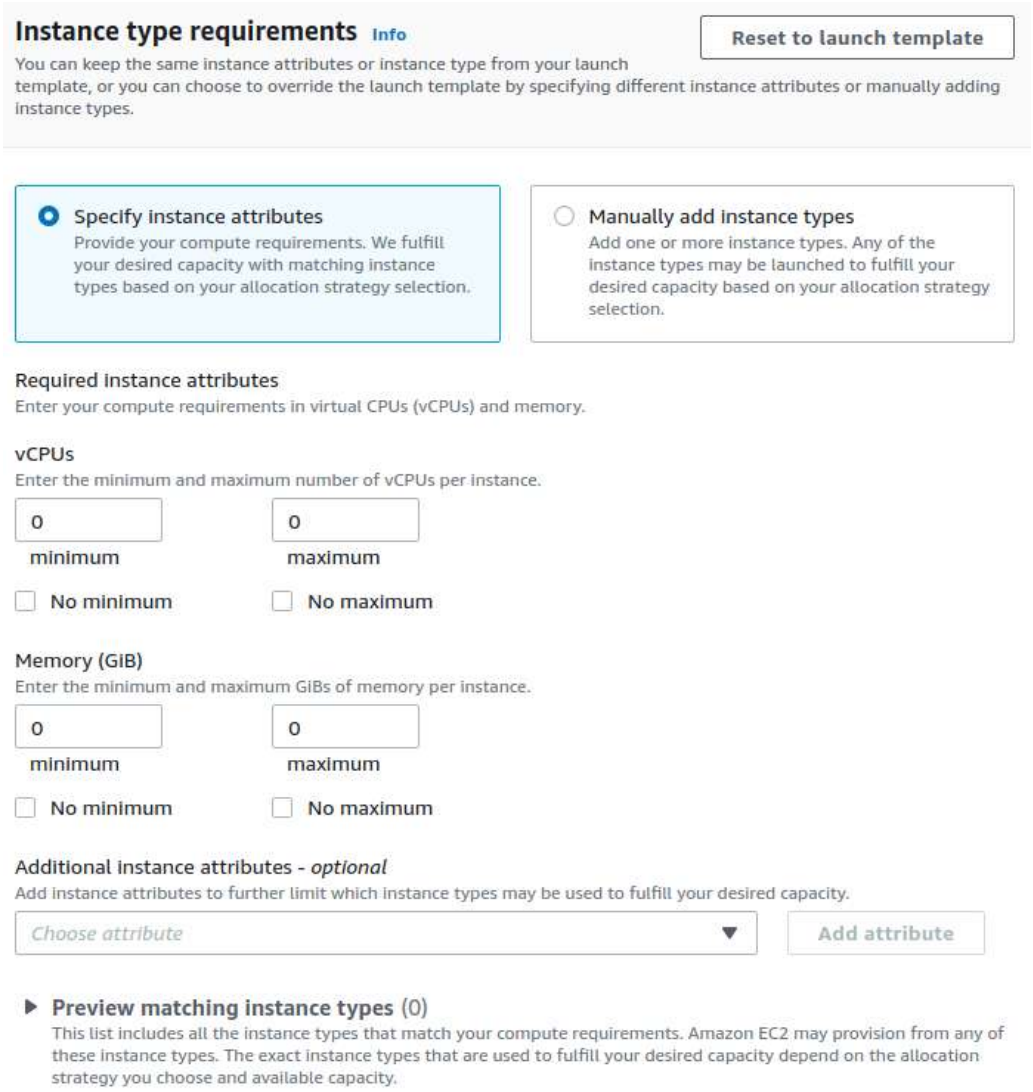
Ahora desactivo el stress con CTRL C y entonces volverá a la normalidad, terminando las 3 instancias. Ha tardado unos 30 minutos

13.7. - Opciones adicionales

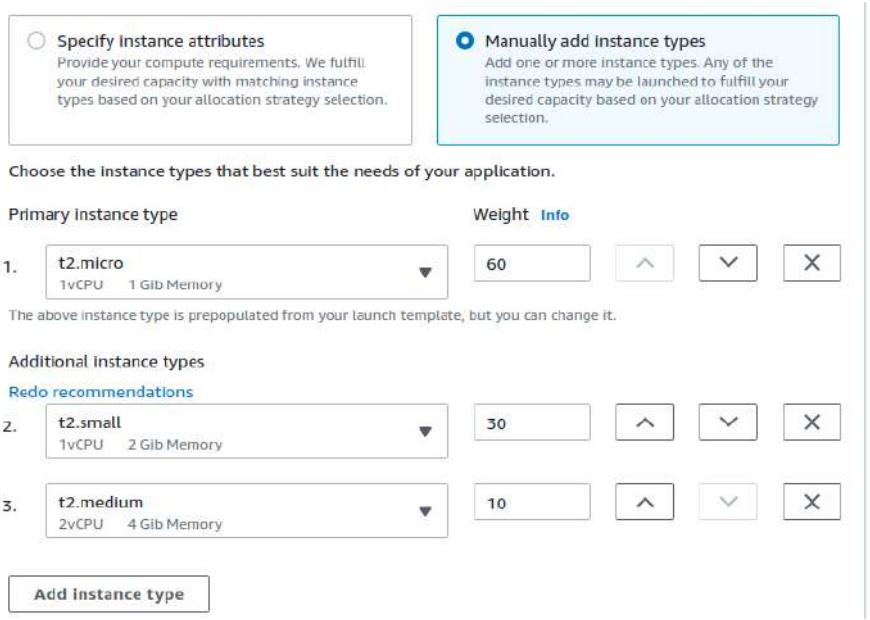
Creamos un nuevo grupo para darle opciones.

Utilizaremos una plantilla sin spot para poder modificarlo.

En los tipos de requerimientos de las instancias seleccionaremos «Override launch template» para ver más opciones. Se puede seleccionar parámetros concretos de CPUs y memoria para que seleccione tipos concretos de instancias.

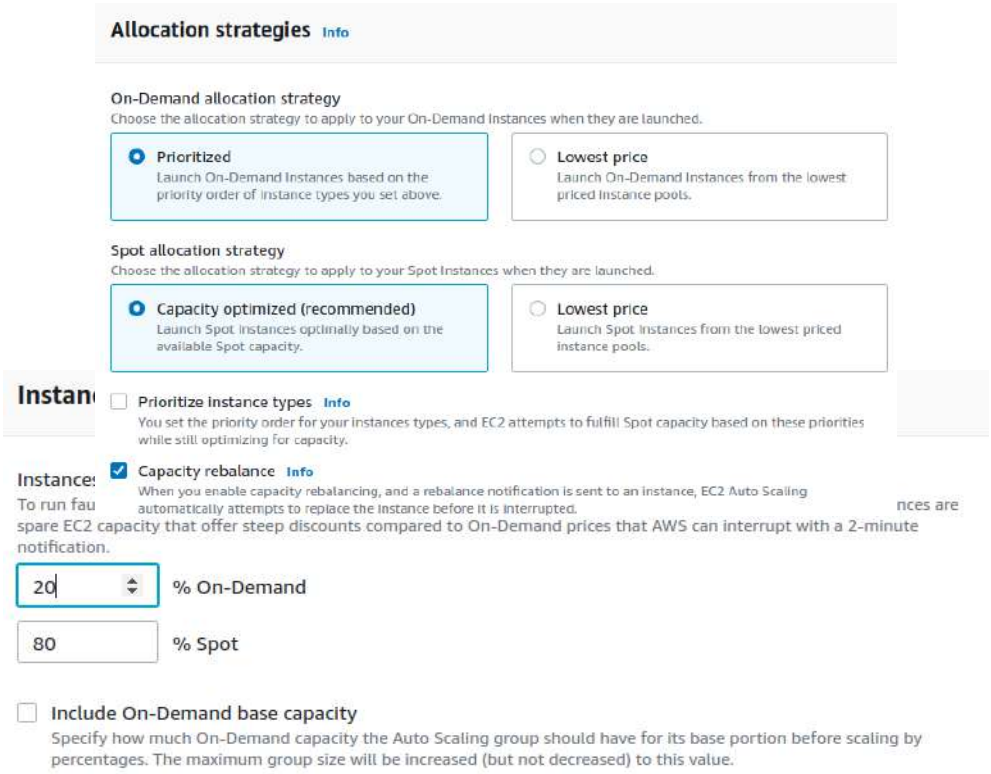


Y se podrían añadir los tipos concretos. También se puede seleccionar manualmente, parametrando el porcentaje de uso del ancho total tendrá cada tipo de instancia



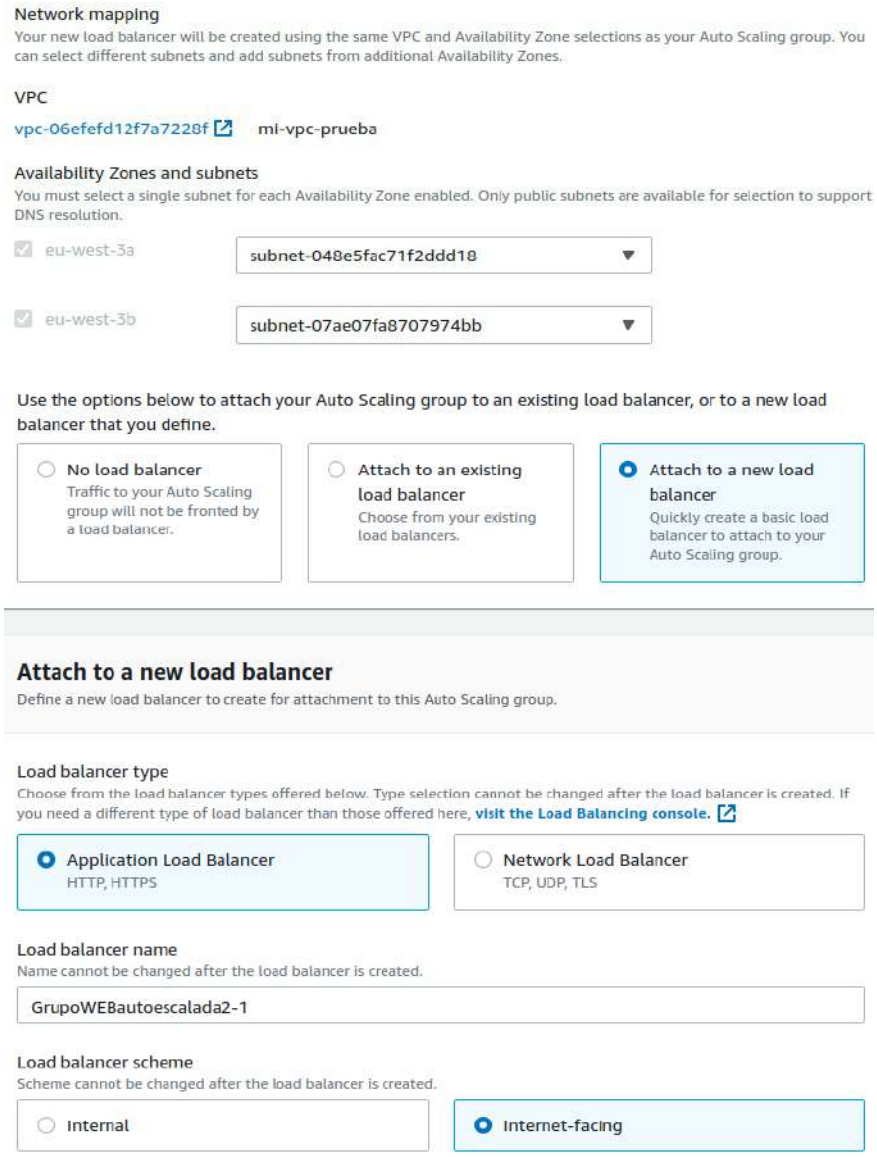
También se puede escoger un porcentaje On-Demand y Spot

También se puede escoger si se tendrá en cuenta para In-Demand la prioridad anterior o el precio más bajo. Y en las Spot la capacidad optimizada o el precio más bajo.

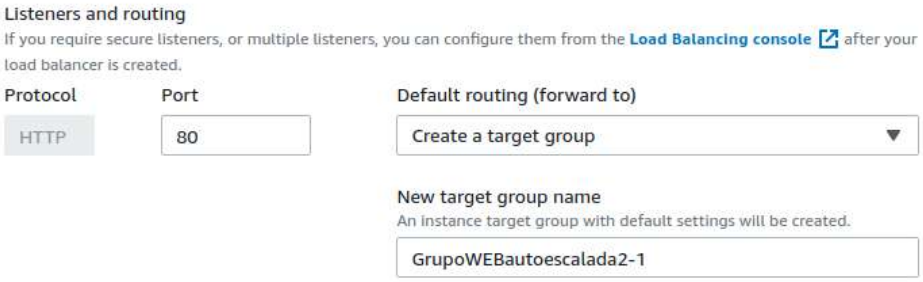


En el paso 3 podemos elegir un balanceador. Como ahora mismo no tenemos ninguno vamos a crearlo.

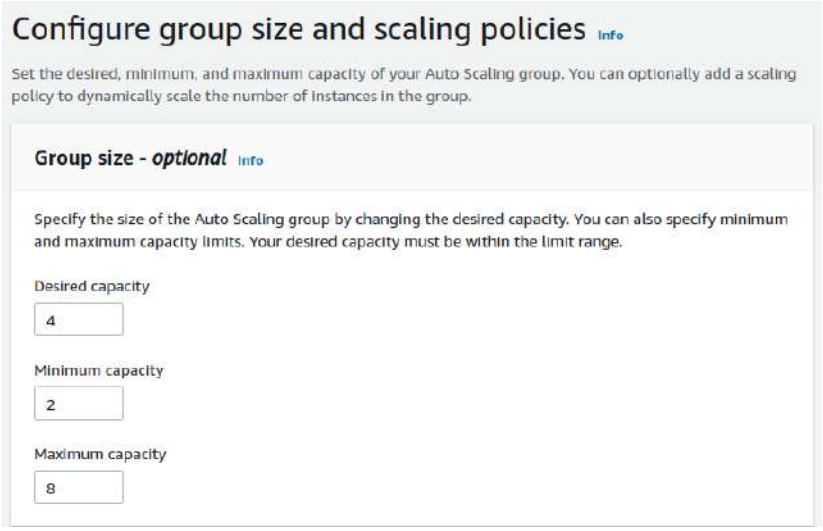
Como hemos escogido el esquema de internet-facing, necesitamos que las dos subredes para balancear sean públicas.



En Target Group tampoco tenemos ninguno así que lo creamos, pero en este caso las instancias serán añadidas tal y como las cree el grupo de autoescalado.



Esta vez le damos más capacidad

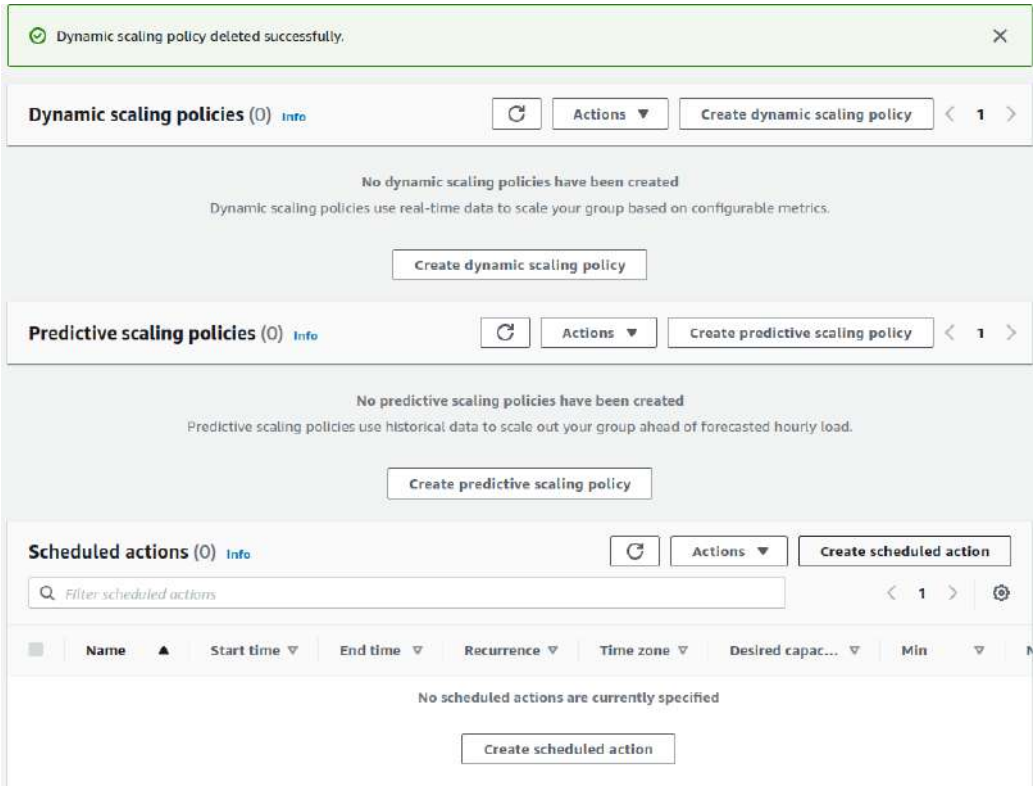


Ahora creará el conjunto de instancias que crea necesarias. Si nos vamos a Load Balancer podremos ver el Grupo creado y en Target Groups el grupo con las instancias que se van creando en el autoescalado.

Cuando eliminamos un grupo de autoescalada se elimina también las instancias, pero no el balanceador de carga y ni el target group del balanceador. CUIDADO QUE COBRAN

13.8. - Política de escalado. Simple y Step

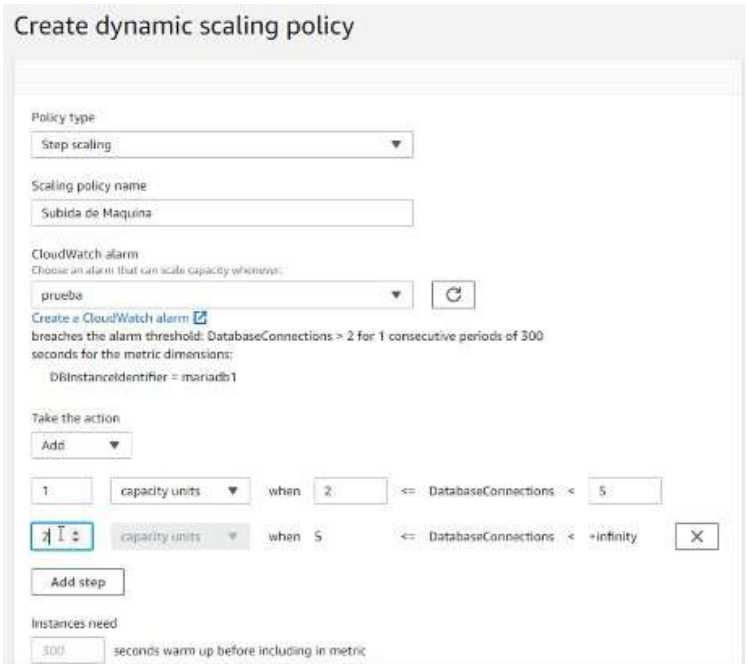
En Automatic scaling del grupo 1 de autoescalada borramos la política de «Demasiada CPU» creada anteriormente.



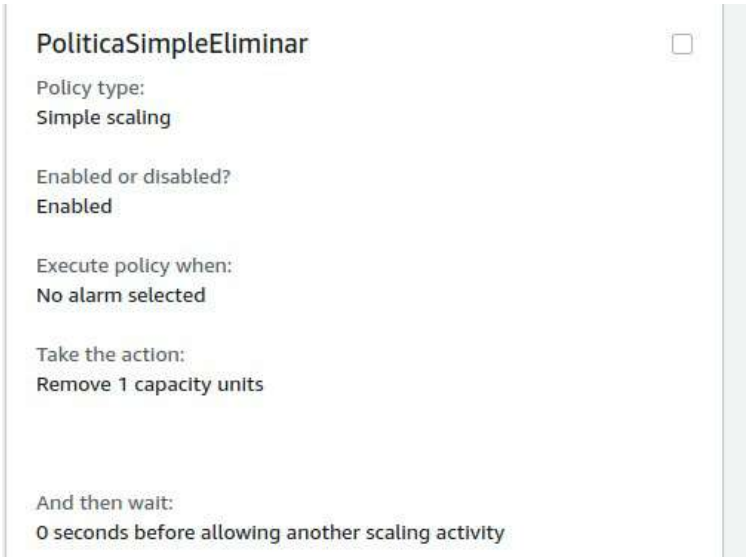
Creamos una nueva política de escalada donde podemos seguir estos parámetros según el **tipo de política** que se escoge.

- **Target tracking scaling –** Por defecto
  - Nombre
  - Tipo de métrica
    - CPU
    - Tráfico entrante (bytes)
    - Tráfico saliente (bytes)
    - Load Balancer
  - Valor de la métrica
  - Tiempo que espera contabilizando el valor para realizar acción.
  - Si queremos no permitir que se autoescale hacia abajo.
- **Step scaling –** Permite poner rangos
  - Nombre
  - Alarma CloudWatcht – Es obligatoria ponerla, no deja editar la acción si no la pones.
  - Acción

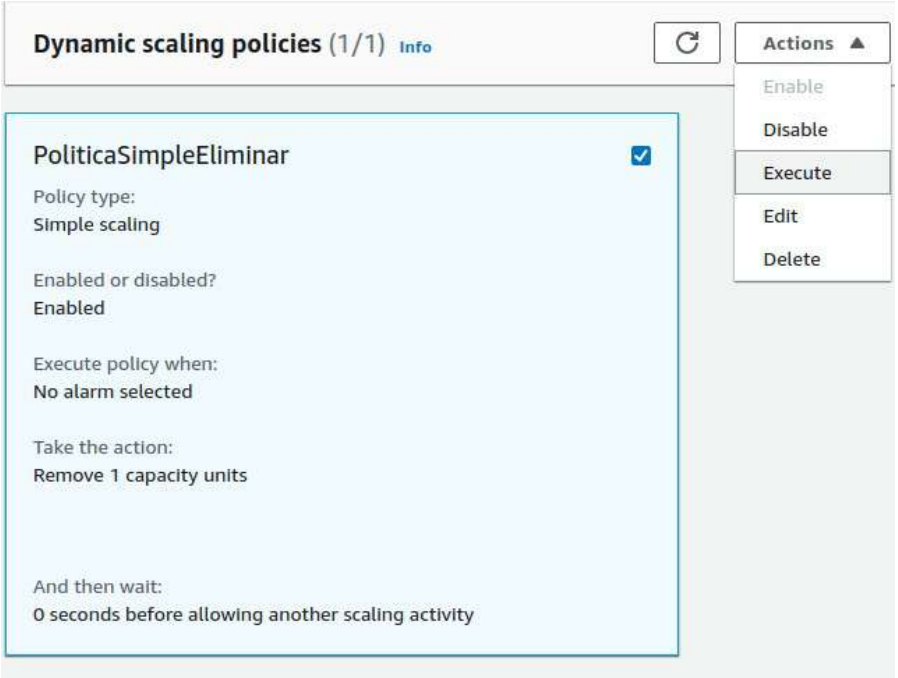
        - Añadir, borrar o cargar
        - Unidad
        - Tipo de unidad (Unidad o porcentaje)
        - RANGO - when (Cuando el valor está entre n y n)
    - Tiempo que espera contabilizando el valor para realizar acción



- **Simple scaling** – El simple. Lo normal es que sea la alarma quien dispare la acción.
    - Nombre
    - Alarma CloudWatcht – Puede estar asociado además a una alarma.
    - Acción
        - Añadir, borrar o cargar
        - Unidad
        - Tipo de unidad (Unidad o porcentaje)
    - Tiempo que espera contabilizando el valor para realizar acción



Se pueden ejecutar las políticas manualmente



TEMA 14 - S3 Almacenamiento escalable en la nube

S3 (Simple Storage Service.) es un servicio de almacenamiento del tipo objeto, guarda objetos. Backups, entornos web, guardar resultados de AWS, etc. Realmente se puede guardar cualquier cosa.

Funciona con un componente que se llama bucket, que es una zona de almacenamiento donde podemos guardar objetos.

- Nos permite un sistema de acceso muy granulado para decidir quien accede a nuestros datos.
- Se puede optimizar el coste con distintos tipos de almacenamiento.
- Se puede replicar en cualquier region.
- Acceso On-premises (Desde mi propio CPD) o VPC.
- Practicamente indestructible.

Se puede usar para IA, analñitica avanzada, Machine Learning, etc...

Productos con conectores con S3: Tablo, Oracle, Pentahoo, etc

14.1. - Buckets

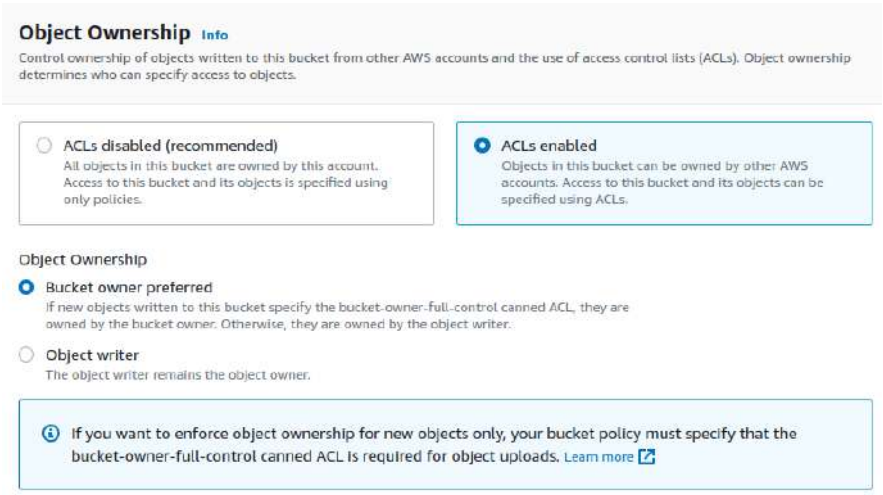
Un bucket es un contenedor que almacena datos. S3 es global, aunque cuando los creas eliges una region concreta.

Los **nombres** no se pueden repetir en todo AWS, así que tiene que ser un nombre personalizado.

Se pueden crear buckets basados en otros que tengamos con **choose bucket**.

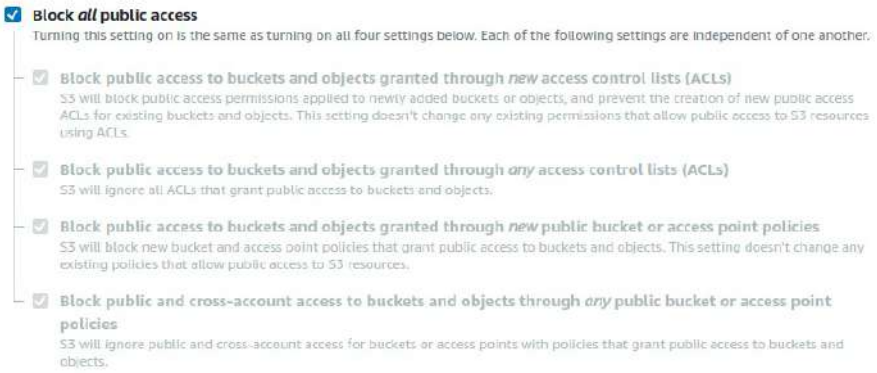
El número máximo de bucket que se pueden tener son 100. Cada bucket tiene espacio ilimitado, por ello no es necesario tener muchos.

Se puede habilitar **ACLs** en Object Ownership. Aquí se controlará quien será el propietario del bucket.



AWS recomienda que seas propiatario la cuenta actual y ya luego puedes dar otra propiedad.

Luego se puede bloquear el acceso, es donde se dan los permisos. Por defecto está todo bloqueado

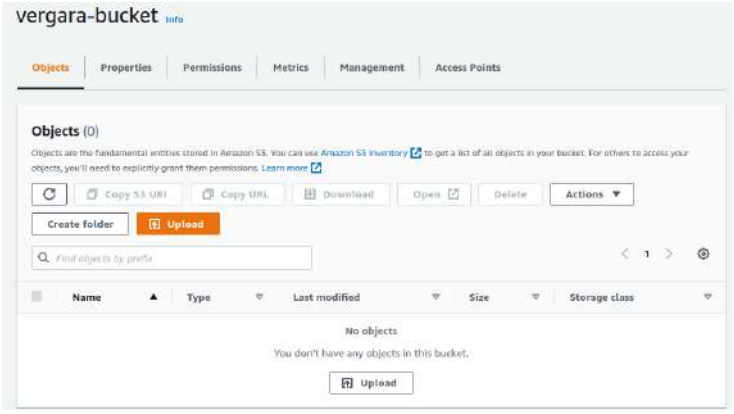


Se puede habilitar el **versionado** del bucket.

Podemos **encriptar** los objetos del contenedor, el contenedor no se encripta.

También se pueden **bloquear** los objetos para que no se puedan cambiar.

Una vez creado podemos añadir los objetos que queramos.



El resto de pestañas son para ver/editar las propiedades, los permisos, la gestión y puentos de acceso. Además de una pestaña de métricas que se pueden personalizar.

Se puede crear una estructura de carpetas para ordenar los objetos. Dentro de cada carpeta puede tener propiedades diferentes.

Los niveles de almacenamiento son

Storage class

Amazon S3 offers a range of storage classes designed for different use cases. Learn more or see Amazon S3 pricing

Storage class	Designed for	Availability Zones	Min storage duration	Min billable object size
Standard	Frequently accessed data (more than once a month) with milliseconds access	≥ 3	-	-
Intelligent-Tiering	Data with changing or unknown access patterns	≥ 3	-	-
Standard-IA	Infrequently accessed data (once a month) with milliseconds access	≥ 3	30 days	128 KB
One Zone-IA	Recreatable, infrequently accessed data (once a month) stored in a single Availability Zone with milliseconds access	1	30 days	128 KB
Glacier Instant Retrieval	Long-lived archive data accessed once a quarter with instant retrieval in milliseconds	≥ 3	90 days	128 KB
Glacier Flexible Retrieval (formerly Glacier)	Long-lived archive data accessed once a year with retrieval of minutes to hours	≥ 3	90 days	-
Glacier Deep Archive	Long-lived archive data accessed less than once a year with retrieval of hours	≥ 3	180 days	-
Reduced redundancy	Noncritical, frequently accessed data with milliseconds access (not recommended as S3 Standard is more cost effective)	≥ 3	-	-

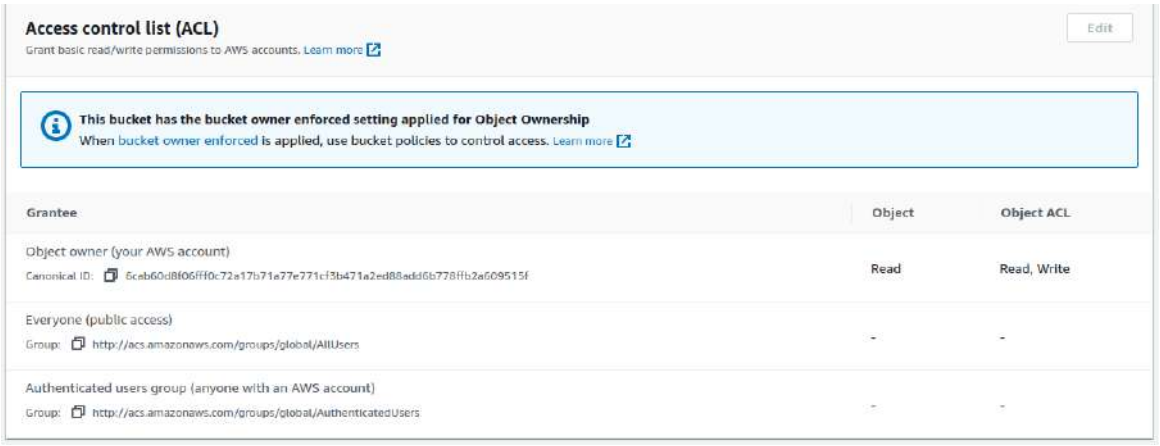
14.2. - Objetos

Un **objecto** tiene:

- **Propiedades**
  - **Key** – Un identificador único de cada objeto, que suele corresponder con el nombre y las estructura jerárquica. Una URI

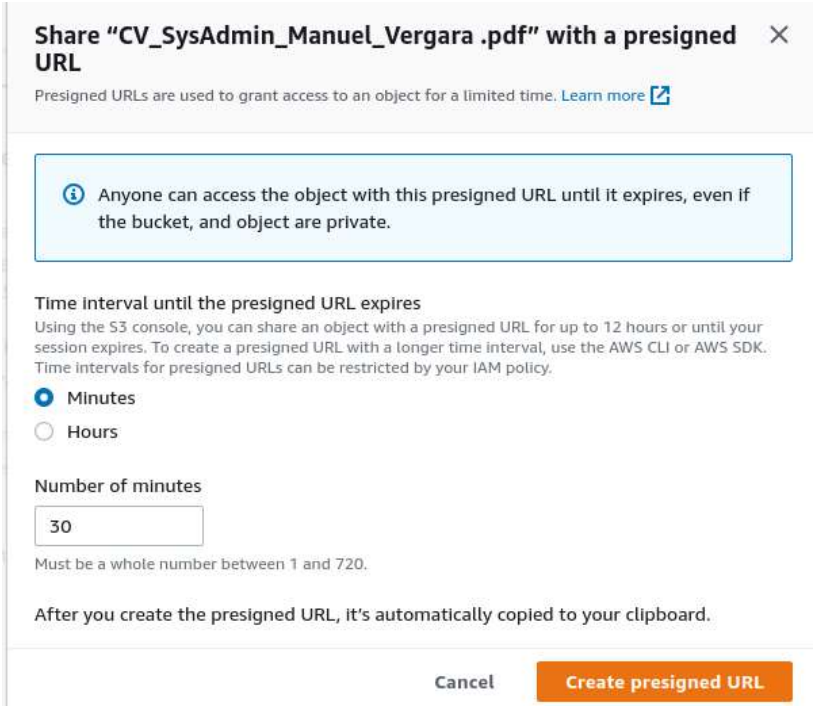
- **Metadata** – Son las propiedades, definen y describen el objeto. Se pueden añadir más metadatos.
- **Uri S3 -**
- **ARN** (Amazon Resource Name)
- **Etiqueta identitaria**
- **URL del objeto** - Sin permisos para acceder desde https aparece un error cuando entramos en el enlace

- **Permisos –** Si hemos dejado la propiedad a la cuenta actual hay que utilizar bucket policies para cambiar permisos. Nos indica quien es el propietario y qué permisos tiene. Por ejemplo, la cuenta propia, el acceso público y grupos de usuarios autenticados.



- **Versiones** – En versiones aparecerían los cambios si tuviesemos activado el versionado y con algún cambio efectuado.

En Objects actions existen varias opciones. Una de ella es «Share with a presigned URL» donde se puede compartir el objeto con un tiempo de expiración que determinamos.



Además, se puede descargar, se puede calcular el tamaño total (Práctico para las carpetas), copiar, mover, renombrar, editar metadatos en forma valor-key (definidos por sistema o por propietario), etc

14.3. - Clases de almacenamiento

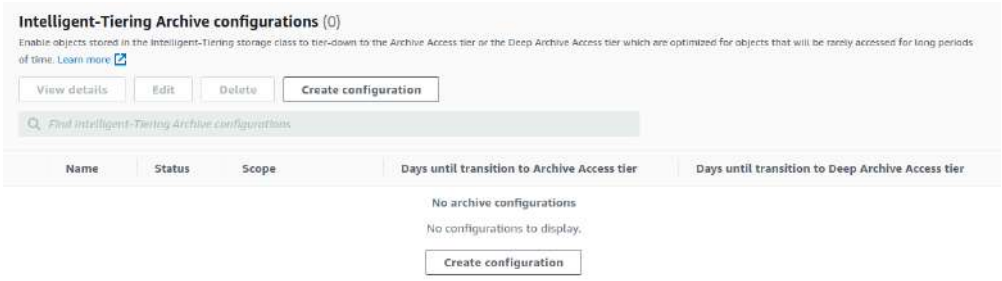
Dependiendo del uso que le demos al contenido tenemos varias clases de almacenamiento para economizar los costes. Cuando se sube objetos, AWS lo replica para tener acceso desde las distintas AZ.

Los niveles son:

- **Standard** – Documentos habituales. Una vez al mes. Es el más optimizado.
- **Intelligent-Tiering** – Cuando no se tiene claro el patrón de cambios, con esta opción es S3 quien determinado la clase más adecuada. Lo determina a través de la monitorización de como se utiliza el objeto. Tiene un coste adicional pero si no se tiene claro como se va a usar el objeto es la mejor opción. Las capas por las que mueve el contenido son:
  - **Frequent Access tie** – Automatica.
  - **Infrequent Access tier** – Automática. Si no es accedido en 30 días pasa aquí.

- **Archive Instant Access tier** – Automática. Si no es accedido en 90 días. (Se pueden personalizar)
- **Archive Access tier** – Opcional. Si no es accedido en 90 días. (Se pueden personalizar). Se tarda entre 3 y 5 horas en recuperar.
- **Deep Archive Access tier** - Opcional. Si no es accedido en 180 días. (Se pueden personalizar). Se tarda unas 12 horas en recuperar.

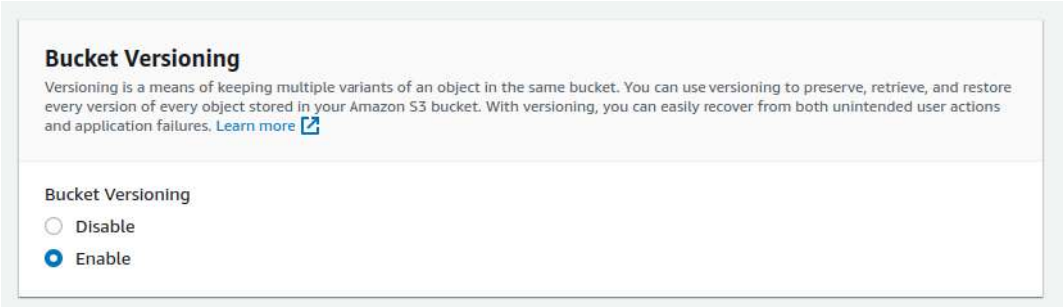
Las configuraciones Intelligent Tiering se configura en las propiedades del bucket



- **Standar-IA** – Esta pensado para datos que no se accede demasiado pero se necesita una acceso rápido.
- **One Zone-IA** – Este no tiene replica, solo se guarda en una AZ y en caso de perderla no hay forrma de recuperarla.
- **Glacier Instant Retrieval** – Cuando no se va a usar el objeto. Pero es el más rápido de obtener cuando se requiere.
- **Glacier Flexible Retrieval –** No se utiliza el objeto y tarda minutos o horas en recuperarse.
- **Glacier Deep Archive –** Este es el archivado profundo que es costoso en recuperarse.
- **Reduced redundancy –** Es el más inmediato para acceder, pero no es recomendable, es para información no crítica.

Cada tipo de almacenamiento tiene un precio, al igual que la recuperación u operar con los objetos.

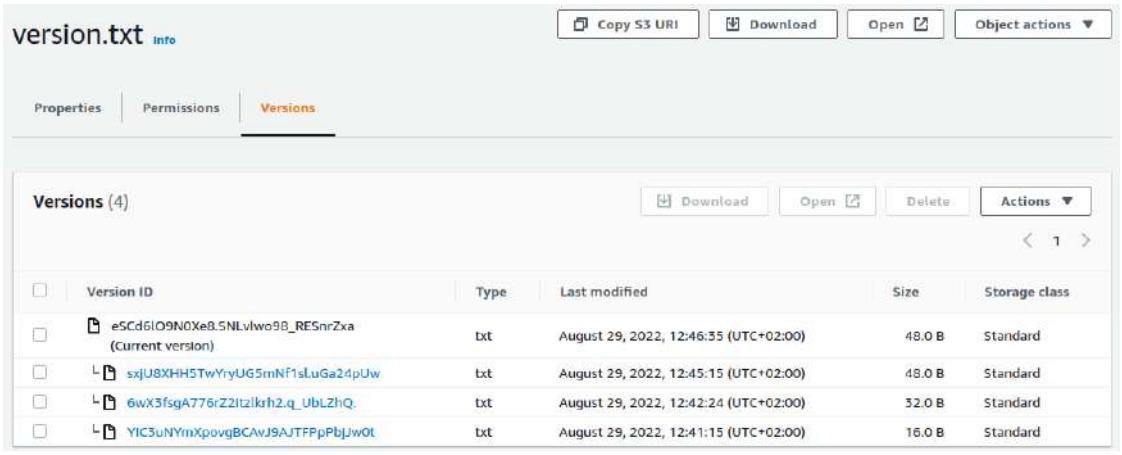
14.4. - Versionado



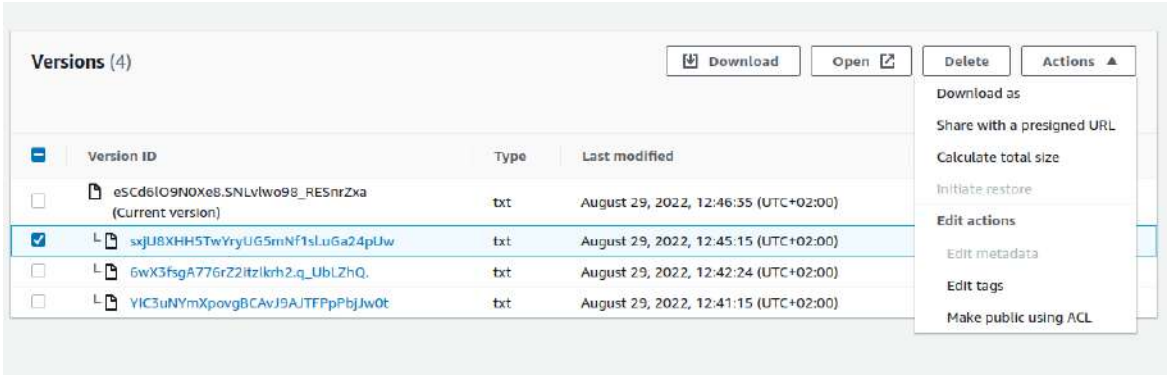
Con el versionado habilitado se pueden recuperar versiones anteriores de los objetos, tanto para guardarla en el mismo lugar como para recuperarla en otra parte como un backup.

Los versionados aparecen en cada uno de los objetos. Se crea la nueva versión cuando:

- Se sube un documento que resulta tener el mismo Key, con el mismo nombre.
- Se canvian los metadatos

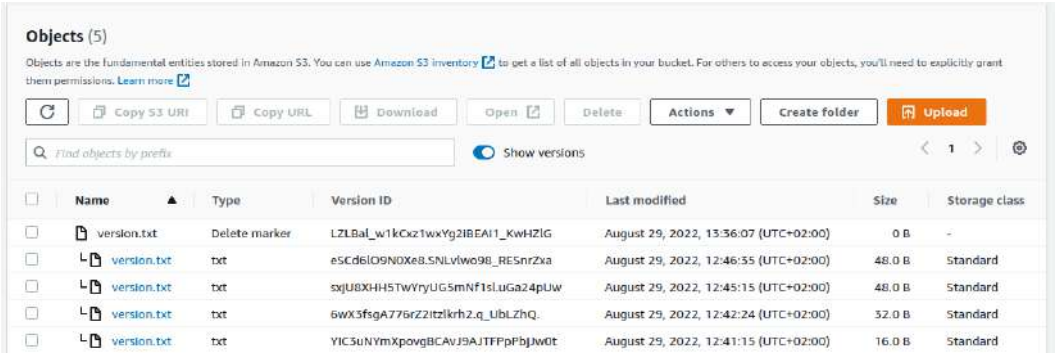


Para recuperar los objetos puede parecer que es en Initiate restore pero esto es para recuperar objetos archivados (Deep glacier)



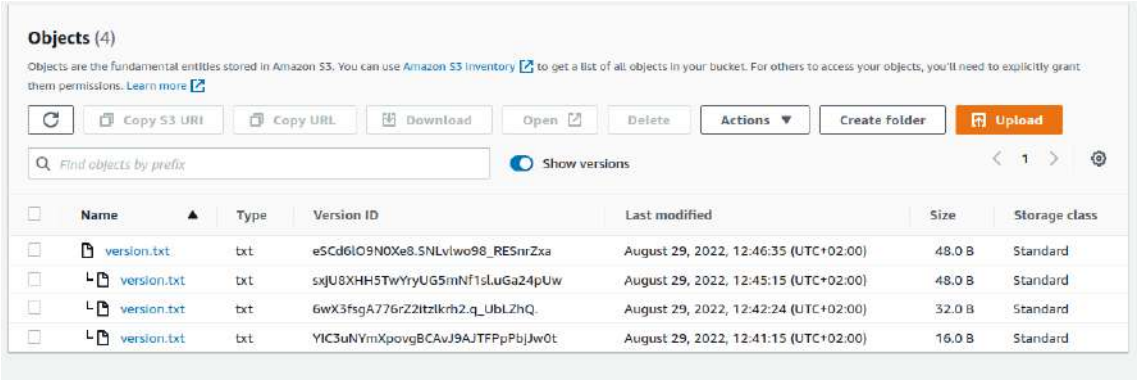
Para recuperar una versión antigua se debe descargar la versión concreta y volverla a subir.

También se puede borrar el objeto y ver en la carpeta concreta las versiones con «Show version»:



Cuando se borra lo que hace es borrar la versión actual y dejar el resto en una especie de marcador. Estos también ocupan espacio de almacenamiento por cada versión.

Cpn lo cual, debemos borrar permanentemente solo el Delete marker. Entonces, la última versión se convierte en la versión actual.



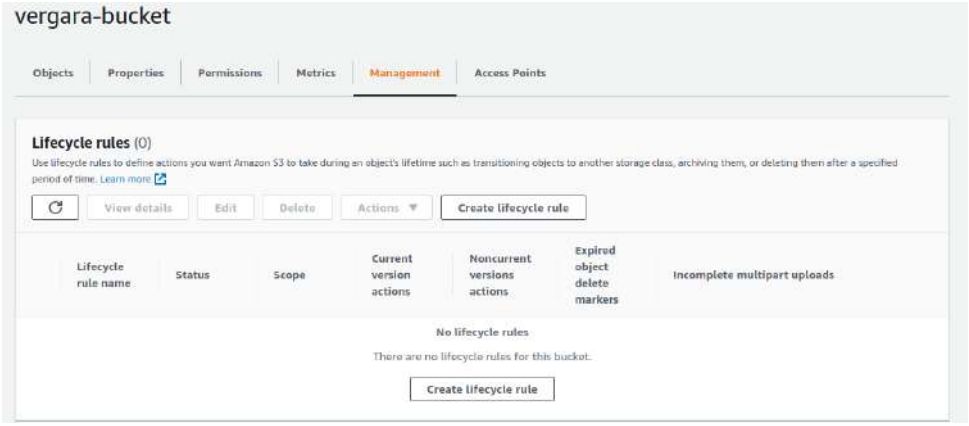
Tambien, si se sube un objeto con el mismo nombre (misma key) se recuperan todas las versiones anteriores.

Seleccionando todo y seleccionando borrado permanente es la manera para que no quede rastro.

14.5. - Management – Gestión

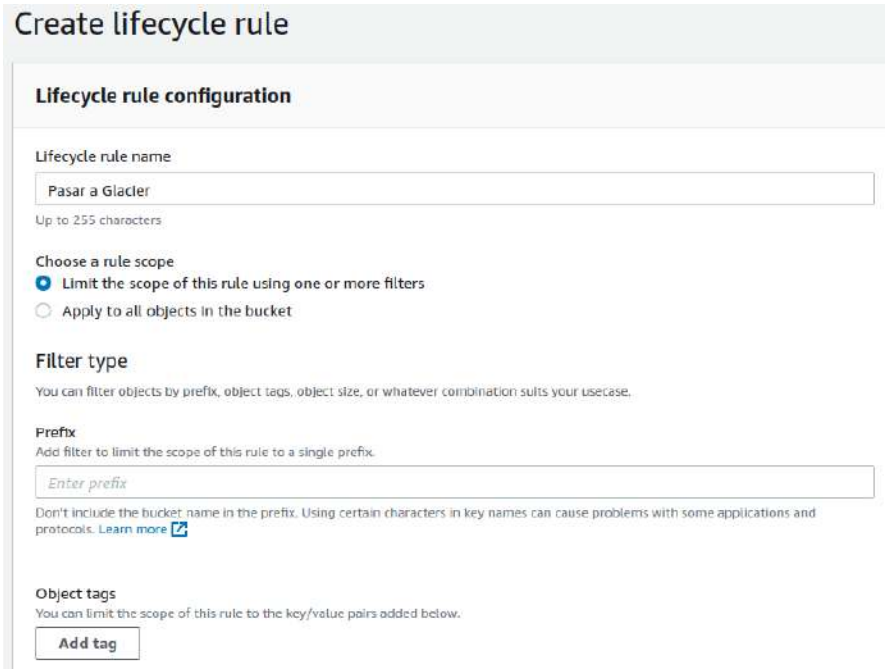
14.5.1. - Ciclo de vida

El ciclo de vida son las acciones que se toman dentro del contenido.

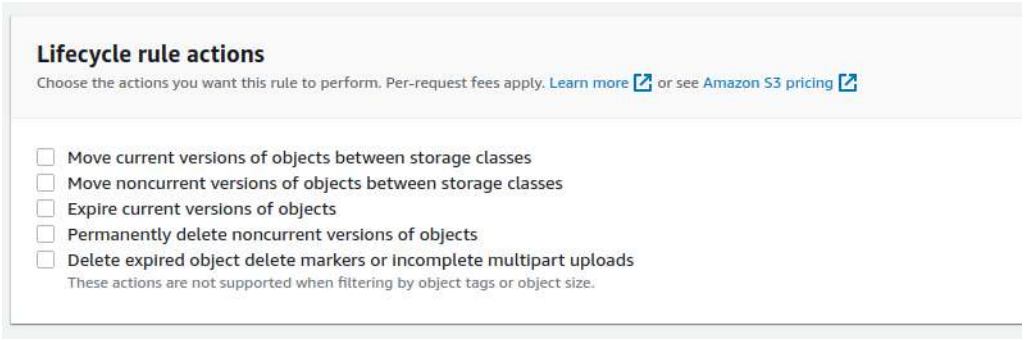


Se pueden hacer distintas acciones con el contenido. Se debe crear las reglas del ciclo de vida.

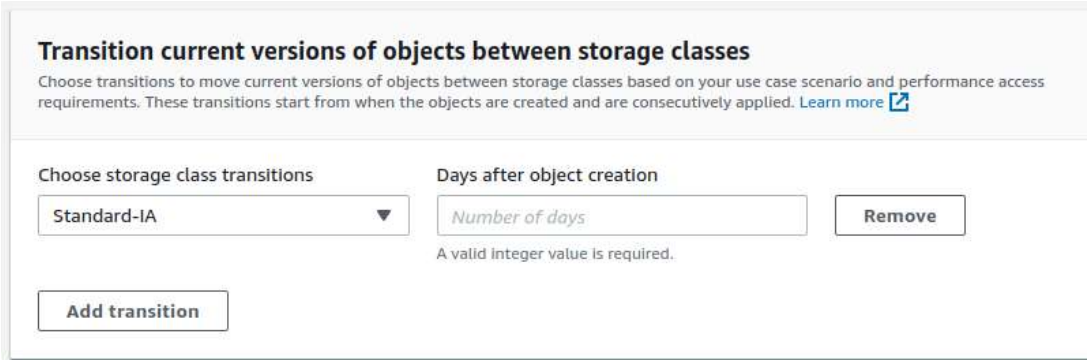
Se pueden limitar a uno o varios elementos o se aplicará a todos los objetos. Se puede filtrar por prefijos o a traves de tags.



También se puede mover las versiones actuales entre distintas clases de almacenamientos, las no actuales. Se puede decir que expiren las versiones. Se puede borrar permanente los objetos no actuales. Y se puede eliminar versiones expiradas.



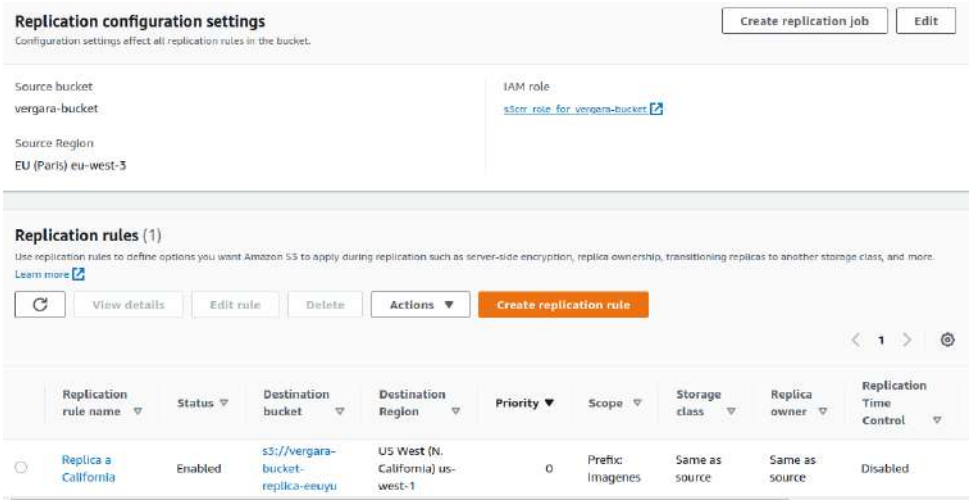
Por ejemplo, en una regla se puede manejar la transición del objeto filtrado.



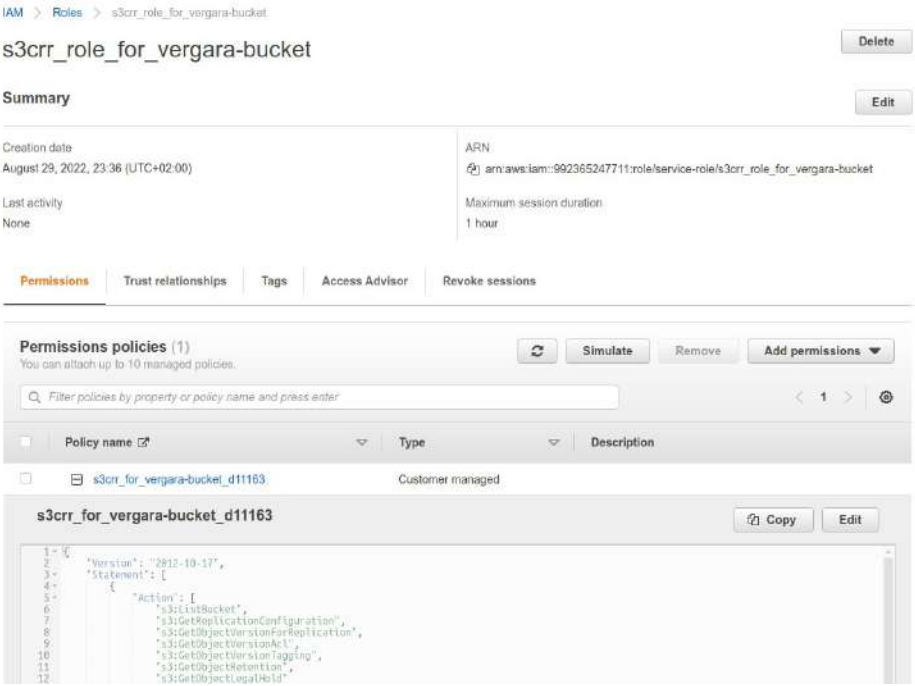
14.5.2. - Replicación

Sirve para llevar contenido de un bucket a otro o de una carpeta a otra, aunque tenga otra AZ. Puede servir para llevarlo a un bucket compartido o a una carpeta de backups.

- Se puede tener habilitado o no.
- Se selecciona el Source bucket y se utilizan filtros y tags para seleccionar los objetos a replicar.
- Se seleccionar el destino tanto de la cuenta propia como ajena.
- Se debe activar el versionado, indica para activarlo
- Se puede seleccionar el IAM o también se puede crear uno nuevo
- Se puede encriptar
- Se puede escoger la clase de almacenamiento en el destino.
- Con el RTC se puede obligar a que haga la replicación en menos de 15 minutos.
- Se pueden extraer métricas.
- Se puede eliminar las operaciones del objeto original.
- Se puede eliminar los metadatos añadidos.



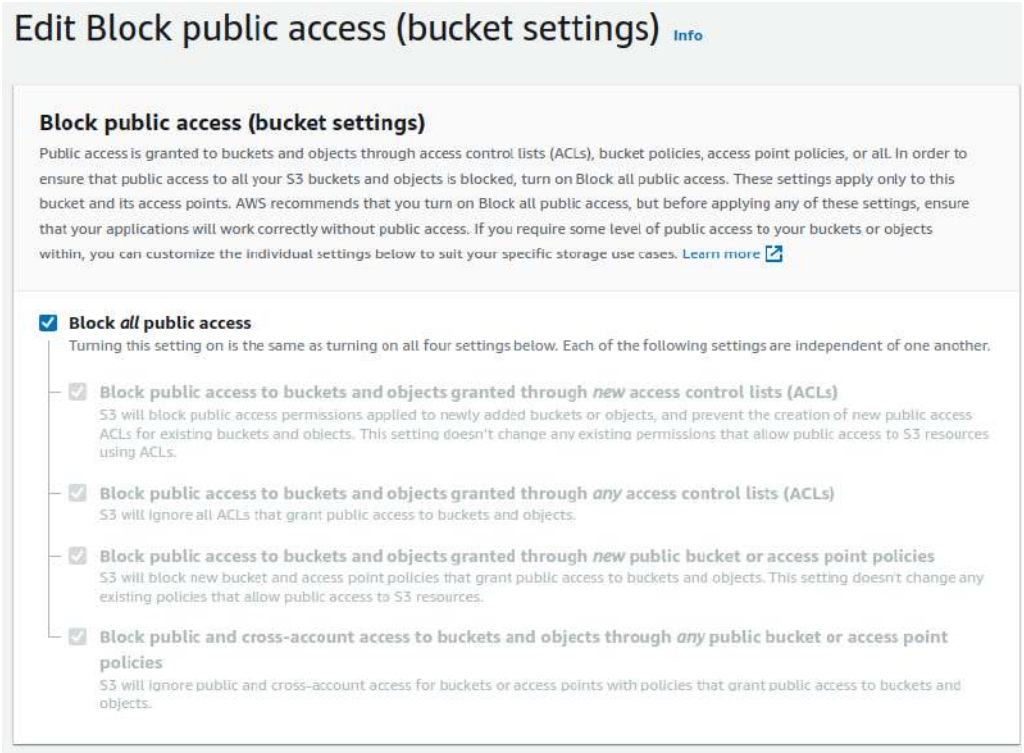
Cuando lo creamos aparece un enlace al rol de replicado creado en el que se puede ver las acciones que puede hacer este rol



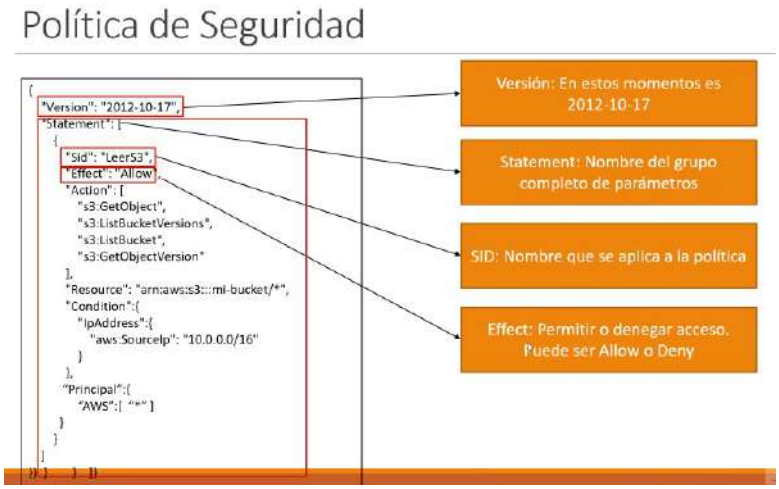
14.6. - Acceso público

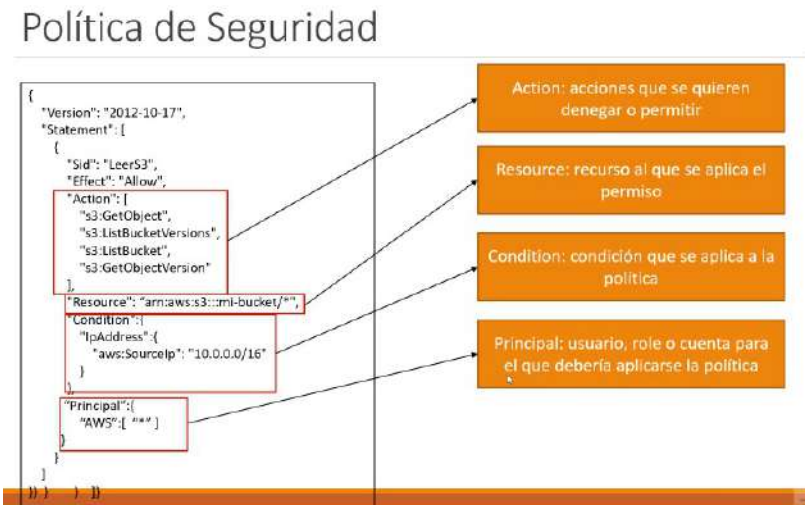
En Permissions podemos cambiar varios parámetros:

- El bloqueo de acceso público – Editando aparece las mismas opciones que creando el bucket
  - Dos bloqueos por ACLs y otros dos por políticas

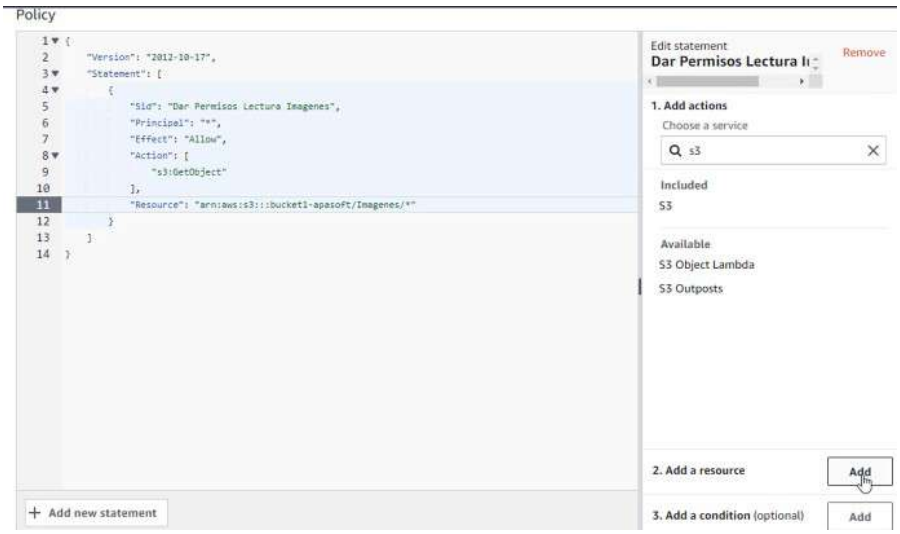


- La política del bucket es del tipo recurso. Se puede utilizar para dar o quitar permisos a usuarios concretos. La parte de políticas de seguridad en AWS se suele manejar con las IAM. Se crea a través de JSON.
    - Ejemplo de permisos:

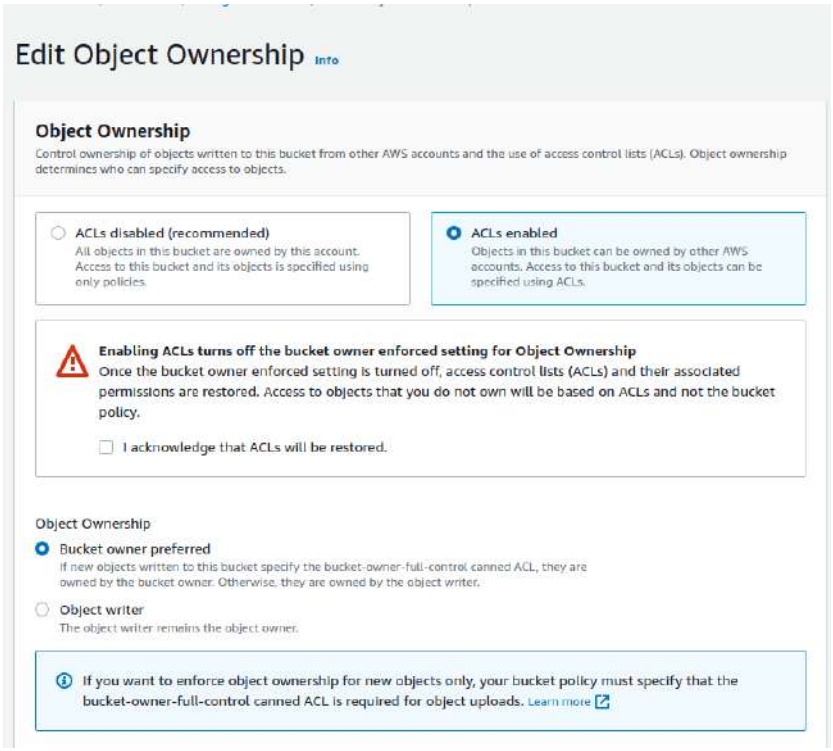




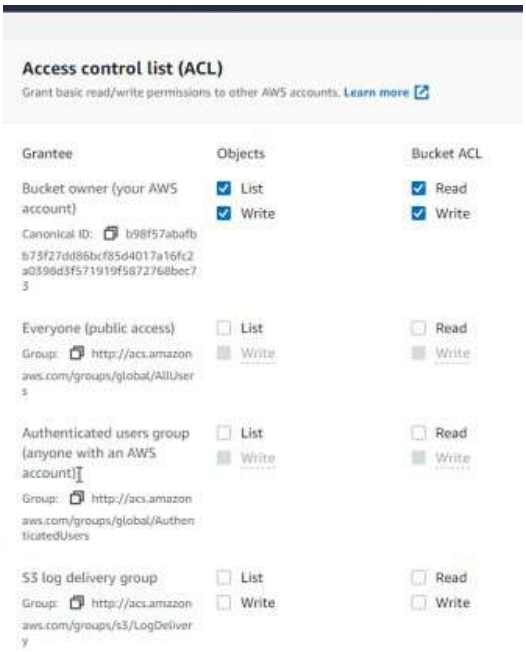
Editor de políticas JSON



- La propiedad del bucket – Lo preferible es la recomendada, tenerlo deshabilitado. De esta manera son las Politicas del bucket quienes dan los permisos. Si se habilita son las ACLs quienes manejan permisos.



- Las listas de control de acceso (ACL) – Para que se active el botón de editar debemos tener habilitadas las ACLs en la propiedad.

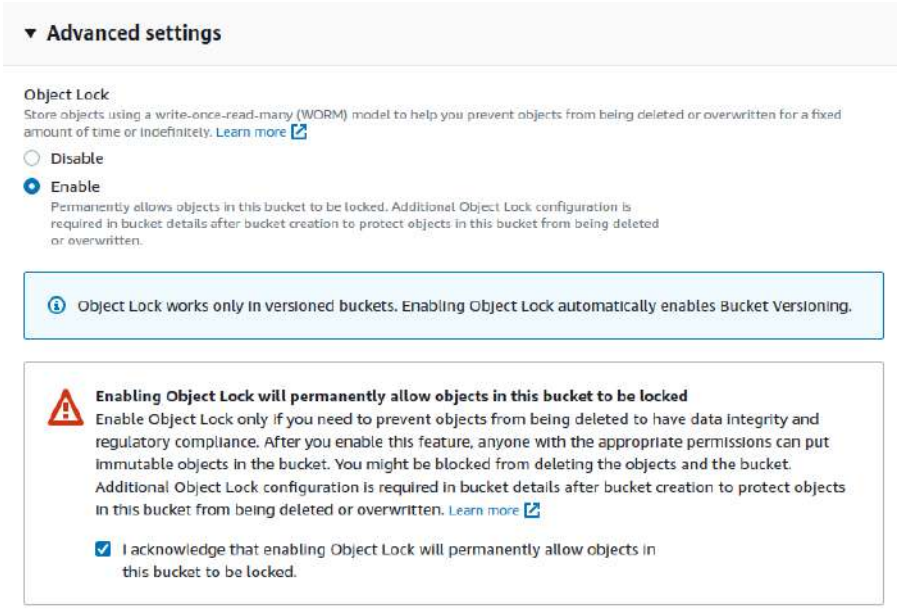


Lo interesante de esta opción es que se pueden editar las ACLs por objetos concretos sin afectar al resto. Pero si se hace a nivel de bucket se hará a todos los archivos.

- Cross-origin resource sharing (CORS)

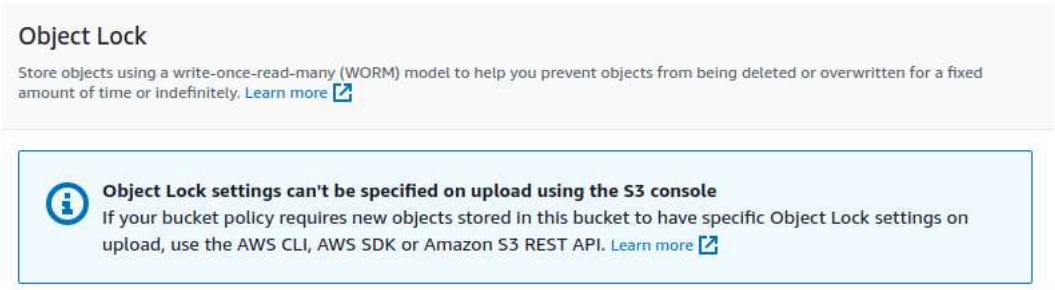
14.7. - Bloqueo objetos

El bloqueo de objetos, si no lo hacemos cuando creamos el bucket no lo podremos hacer después. Se tendría que abrir una petición a AWS para que lo hicieran manualmente. Cuando se activa automáticamente también se activa el versionado.



También avisa que si se crea un bloqueo a cierto objecto puede quedar de manera permanente hasta la fecha de expiración del bloqueo o con el parámetro indicado. Con lo cuál, no se podrá eliminar el bucket y la única forma de eliminarlo es eliminando la cuenta.

Mientras se sube un objeto no se puede bloquear, a no ser que se use AWS CLI, AWS SDK o S3 REST API.



Una vez subido el objeto, dentro del mismo en propiedades, hay dos opciones de bloqueo:

- **Object Lock legal hold** – No tiene temporalidad, solo se habilita o deshabilita.
- **Object Lock retention** – CUIDAO! Puede bloquear completamente sin que se pueda desbloquear. El bloqueo tiene una fecha de expiración. Hay dos modos:
  - **Governance mode** – Se puede volver atrás y deshabilitar el bloqueo
  - **Compliance mode** – Nadie puede quitar el bloqueo antes de la fecha indicada

Si se intentan borrar, en principio parece que se puede pero quedará versionado. Esta versión no se puede borrar con el bloqueo habilitado.

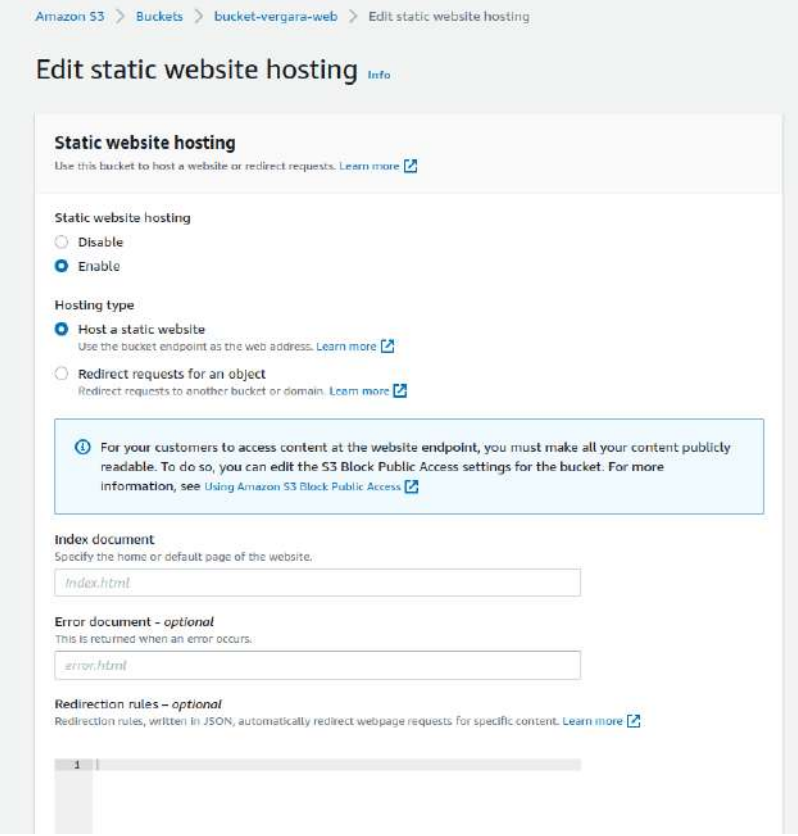
El bloqueo puede servir legalmente para indicar que un objeto no ha sido modificado.

14.8. - S3Browser

Es una herramienta de un tercero para poder navegar por los buckets como si fuera un fichero en nuestro SO. https://s3browser.com/ Como esta herramienta existen otras, AWS no tiene ninguna herramienta interna para esto, en cambio Azure si que la tiene.

14.9. - Web estática en S3

Con las ACLs activadas (O haciendo políticas expresas para hacer accesible la carpeta donde este la web) y haciendo todo público. Vamos al final de las propiedades del bucket y está la opción «Static website hosting».



Lo habilitamos e indicamos el host o la redirección. En el caso del host debemos indicar el documento que será el índice y el de error. Además, podemos añadir una reglas de redireccionamiento.

Luego tenemos que subir los documentos de la web dándole permiso de lectura pública. Si volvemos de nuevo a la propiedades a la parte de Static website hosting, veremos que nos ha generado un enlace.

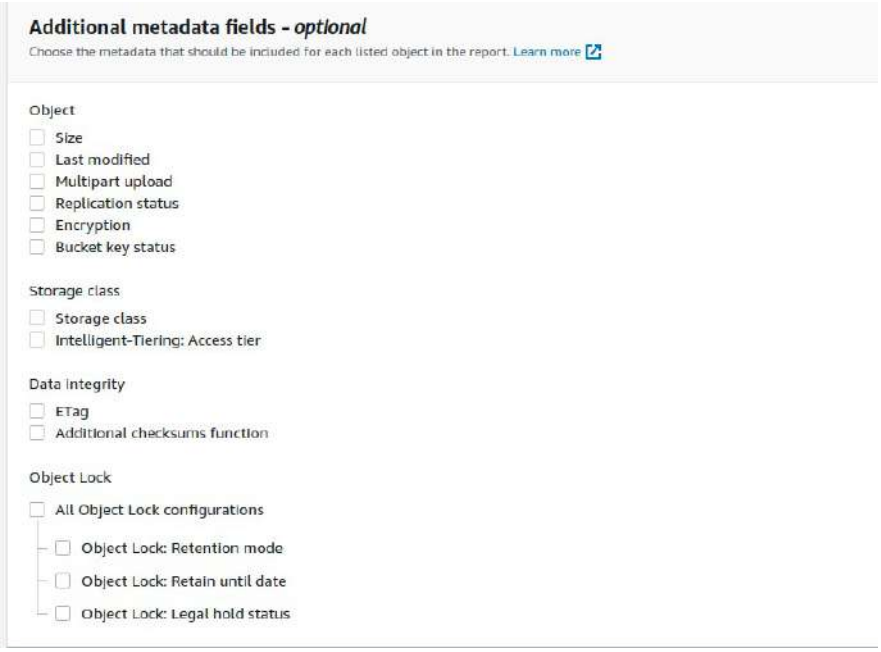
14.10. - Inventario de los objetos

Recopilación de inventario se puede hacer en la pestaá management en el apartado configuración de inventarios.

Cuando creas un inventario puede tardar días en recopilar la información de los objetos y sus metadatos.

Creando una configuración de inventario se parámetra:

- el nombre
- Opción a filtrar por prefijos
- Listar por versión actual o por todas las versiones anteriores
- Destino del resultado. Path de la ubicación.
- Política para permisos en el bucket destino. Permite el servicio S3 en el bucket correspondiente indicando el origen y la cuenta para el control.
- Frecuencia del inventario. Diaria o semanal
- Formato de salida. CSV, Apache ORC o Apache Parquet
- Si la configuración queda activa o desactiva para publicar los inventarios
- Si se quiere encriptar
- Se le puede añadir campos extrar en los metadatos



Aparecen varias carpetas. Una con el report diario, otro con los datos y otro con hive (visualizar los datos en BBDD de entorno BigData https://hive.apache.org/). Dentro del report diario hay dos archivos: un manifest.json con un resumen de los datos y un checksum.

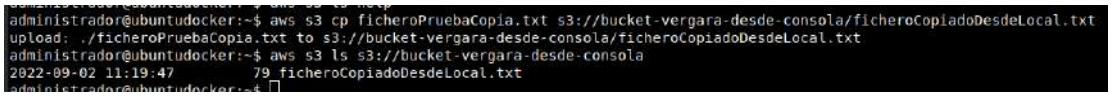
14.11. - Línea de comandos para S3

14.11.1. - Subcomando "aws s3"

Documentación: https://awscli.amazonaws.com/v2/documentation/api/latest/reference/s3/index.html

Opciones:

cp – Copiar. Tanto desde o hacia un bucket como desde o hacia local: <LocalPath> <S3Uri> or <S3Uri> <LocalPath> or <S3Uri> <S3Uri>

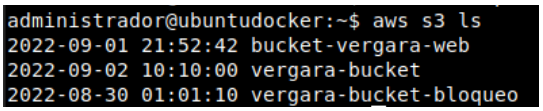


Tiene opciones parecidas a cp de bash. Por ejemplo --recursive Para copiar directorios y su contenido.
Se puede mezclar con un script de Linux. Por ejemplo, para subir todos los ficheros del directorio local:

for i in *
do

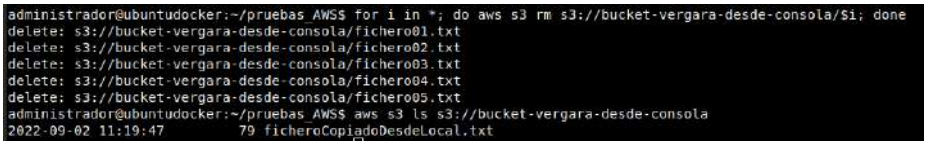
aws s3 cp $i s3://vergara-bucket/Docs/
done

ls – Listar

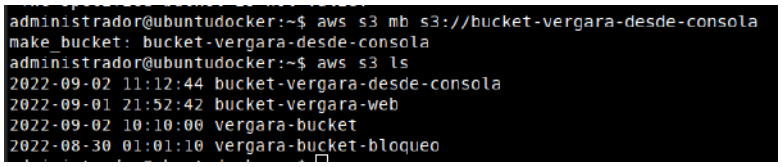


mv - Mover

rm – Borrar objetos y buckets. Por ejemplo, usando un for de bash



mb – Crear un bucket. Sintaxis: aws s3 mb s3://mybucket



presign - Genera una URL pre-firmada para un objeto de Amazon S3. Esto permite que cualquiera que reciba la URL pre-firmada pueda recuperar el objeto de S3 con una solicitud HTTP GET. Todas las URL pre-firmadas utilizan ahora sigv4, por lo que la región debe configurarse explícitamente.

rb – Borrar un bucket

sync - Sincroniza directorios y prefijos de S3. Copia recursivamente los archivos nuevos y actualizados del directorio de origen al de destino. Solo crea carpetas en el destino si contienen uno o más archivos.

website - Establece la configuración del sitio web para un bucket.

14.11.2. - Subcomando "aws s3api"

Es más parecido a lo visto hasta ahora como CLI. Documentación:
https://awscli.amazonaws.com/v2/documentation/api/latest/reference/s3api/index.html

Por ejemplo, para listar los buckets:

aws s3api list-buckets

```
administrador@ubuntudocker:~/pruebas_AWS$ aws s3api list-buckets
{
    "Buckets": [
        {
            "Name": "bucket-vergara-desde-consola",
            "CreationDate": "2022-09-02T09:12:44+00:00"
        },
        {
            "Name": "bucket-vergara-web",
            "CreationDate": "2022-09-01T19:52:42+00:00"
        },
        {
            "Name": "vergara-bucket",
            "CreationDate": "2022-09-02T08:10:00+00:00"
        },
        {
            "Name": "vergara-bucket-bloqueo",
            "CreationDate": "2022-08-29T23:01:10+00:00"
        }
    ],
    "Owner": {
        "ID": "6cab60d8f06fff0c72a17b71a77e771cf3b471a2ed88add6b778ffb2a609515f"
    }
}
```

Para crear un bucket:

aws s3api create-bucket --bucket nombre-del-nuevo-bucket --region eu-west-3 --create-bucket-configuration LocationConstraint=eu-west-3

CUIDADO: Si no ponemos la region lo creará en us-east-1. Y si no ponemos la configuración del bucket no permite crearlo.

```
administrador@ubuntudocker:~/pruebas_AWS$ aws s3api create-bucket --bucket bucket-vergara-desde-consola2 --region eu-west-3

An error occurred (IllegalLocationConstraintException) when calling the CreateBucket operation: The unspecified location constraint
 is incompatible for the region specific endpoint this request was sent to.
administrador@ubuntudocker:~/pruebas_AWS$ aws s3api create-bucket --bucket bucket-vergara-desde-consola2 --region eu-west-3 --creat
e-bucket-configuration LocationConstraint=eu-west-3
{
    "Location": "http://bucket-vergara-desde-consola2.s3.amazonaws.com/"
}
administrador@ubuntudocker:~/pruebas_AWS$
```

Para subir un objeto:

aws s3api put-object --bucket nombre-bucket --key nombre-fichero-en-destino –body nombre-fichero-actual

Para listar objetos:

aws s3api list-objects --bucket nombre-bucket

```
administrador@ubuntudocker:~/pruebas_AWS$ aws s3api put-object --bucket bucket-vergara-desde-consola2 --
key ficheroPutObject.txt --body fichero03.txt
{
    "ETag": "\"d41d8cd98f00b204e9800998ecf8427e\""
}
administrador@ubuntudocker:~/pruebas_AWS$ aws s3api list-objects --bucket bucket-vergara-desde-consola2
{
    "Contents": [
        {
            "Key": "ficheroPutObject.txt",
            "LastModified": "2022-09-02T10:28:54+00:00",
            "ETag": "\"d41d8cd98f00b204e9800998ecf8427e\"",
            "Size": 0,
            "StorageClass": "STANDARD",
            "Owner": {
                "ID": "6cab60d8f06fff0c72a17b71a77e771cf3b471a2ed88add6b778ffb2a609515f"
            }
        }
    ]
}
```

Opciones del subcomando s3api:

abort-multipart-upload
complete-multipart-upload

copy-object
create-bucket
create-multipart-upload
delete-bucket
delete-bucket-analytics-configuration
delete-bucket-cors
delete-bucket-encryption
delete-bucket-intelligent-tiering-configuration
delete-bucket-inventory-configuration
delete-bucket-lifecycle
delete-bucket-metrics-configuration
delete-bucket-ownership-controls
delete-bucket-policy
delete-bucket-replication
delete-bucket-tagging
delete-bucket-website
delete-object
delete-object-tagging
delete-objects
delete-public-access-block
get-bucket-accelerate-configuration
get-bucket-acl
get-bucket-analytics-configuration
get-bucket-cors
get-bucket-encryption
get-bucket-intelligent-tiering-configuration
get-bucket-inventory-configuration
get-bucket-lifecycle-configuration
get-bucket-location
get-bucket-logging
get-bucket-metrics-configuration
get-bucket-notification-configuration
get-bucket-ownership-controls
get-bucket-policy
get-bucket-policy-status
get-bucket-replication
get-bucket-request-payment
get-bucket-tagging
get-bucket-versioning
get-bucket-website
get-object
get-object-acl

get-object-attributes
get-object-legal-hold
get-object-lock-configuration
get-object-retention
get-object-tagging
get-object-torrent
get-public-access-block
head-bucket
head-object
list-bucket-analytics-configurations
list-bucket-intelligent-tiering-configurations
list-bucket-inventory-configurations
list-bucket-metrics-configurations
list-buckets
list-multipart-uploads
list-object-versions
list-objects
list-objects-v2
list-parts
put-bucket-accelerate-configuration
put-bucket-acl
put-bucket-analytics-configuration
put-bucket-cors
put-bucket-encryption
put-bucket-intelligent-tiering-configuration
put-bucket-inventory-configuration
put-bucket-lifecycle-configuration
put-bucket-logging
put-bucket-metrics-configuration
put-bucket-notification-configuration
put-bucket-ownership-controls
put-bucket-policy
put-bucket-replication
put-bucket-request-payment
put-bucket-tagging
put-bucket-versioning
put-bucket-website
put-object
put-object-acl
put-object-legal-hold
put-object-lock-configuration
put-object-retention

put-object-tagging
put-public-access-block
restore-object
select-object-content
upload-part
upload-part-copy
wait
write-get-object-response

14.11.3. - Subcomando "aws s3control"

Para el plano de control.

Documentación: https://awscli.amazonaws.com/v2/documentation/api/latest/reference/s3control/index.html

Opciones:

create-access-point
create-access-point-for-object-lambda
create-bucket
create-job
create-multi-region-access-point
delete-access-point
delete-access-point-for-object-lambda
delete-access-point-policy
delete-access-point-policy-for-object-lambda
delete-bucket
delete-bucket-lifecycle-configuration
delete-bucket-policy
delete-bucket-tagging
delete-job-tagging
delete-multi-region-access-point
delete-public-access-block
delete-storage-lens-configuration
delete-storage-lens-configuration-tagging
describe-job
describe-multi-region-access-point-operation
get-access-point
get-access-point-configuration-for-object-lambda
get-access-point-for-object-lambda
get-access-point-policy
get-access-point-policy-for-object-lambda

get-access-point-policy-status
get-access-point-policy-status-for-object-lambda
get-bucket
get-bucket-lifecycle-configuration
get-bucket-policy
get-bucket-tagging
get-job-tagging
get-multi-region-access-point
get-multi-region-access-point-policy
get-multi-region-access-point-policy-status
get-public-access-block
get-storage-lens-configuration
get-storage-lens-configuration-tagging
list-access-points
list-access-points-for-object-lambda
list-jobs
list-multi-region-access-points
list-regional-buckets
list-storage-lens-configurations
put-access-point-configuration-for-object-lambda
put-access-point-policy
put-access-point-policy-for-object-lambda
put-bucket-lifecycle-configuration
put-bucket-policy
put-bucket-tagging
put-job-tagging
put-multi-region-access-point-policy
put-public-access-block
put-storage-lens-configuration
put-storage-lens-configuration-tagging
update-job-priority
update-job-status

14.11.4. - Subcomando "s3outposts"

Para las operaciones de S3 on Outposts.

Documentación: https://awscli.amazonaws.com/v2/documentation/api/latest/reference/s3outposts/index.html

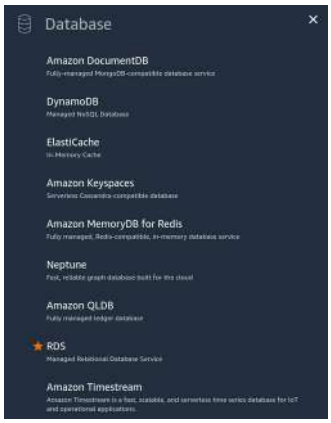
Opciones:

create-endpoint
delete-endpoint

list-endpoints
list-shared-endpoints

TEMA 15 - RDS Bases de Datos Relacionales

Hay distintos tipos de BBDD en AWS.



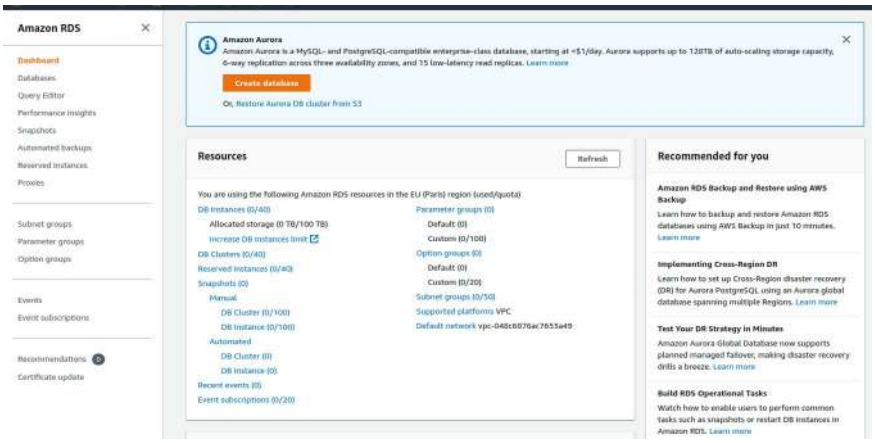
A rasgos generales:

- BBDD **relacionales** de terceros.
  - RDS – Oracle, SQL server, Mariadb, postgreSQL, mySQL..
    - Aurora es una BBDD enfocada para cloud. También forma parte de RDS pero es diferente al resto. Soporta serverless.
- **NoSQL** -
  - Key-Value
    - DynamoDB
  - In-Memory
    - Elastic Cache (Memcached y redis). Es un frontal para tener por detrás una BBDD tradicional.
    - MemoryDB for Redis. BBDD completa, con todas las características necesarias.
  - Document
    - DocumentDB compatibles con MongoDB
  - Graph
    - Neptune se parece a otras como NeoForJava
  - Columnar
    - Keyspaces como Apache Cassandra
  - Time series. Permite recorrer datos de tipo temporal masivos.
    - Timestream

    - Ledger – Contable. Certifica de manera inequívoca los asientos que se hacen. Incluso, tienen validez judicial.
        - QLDB

- **Data warehouse**
    - Redshift – BBDD más popular

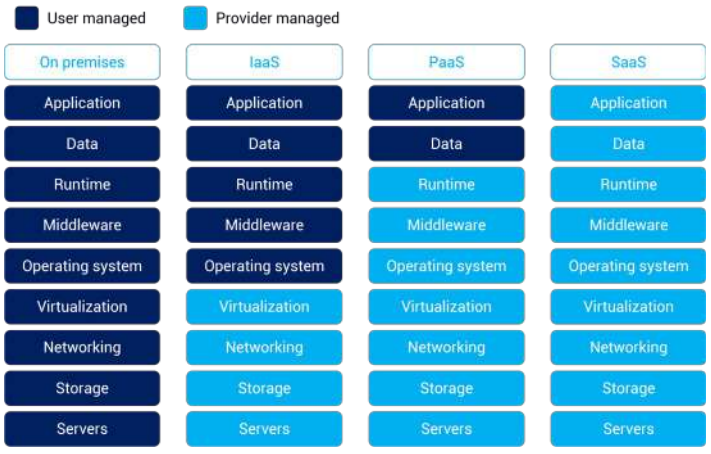
RDS



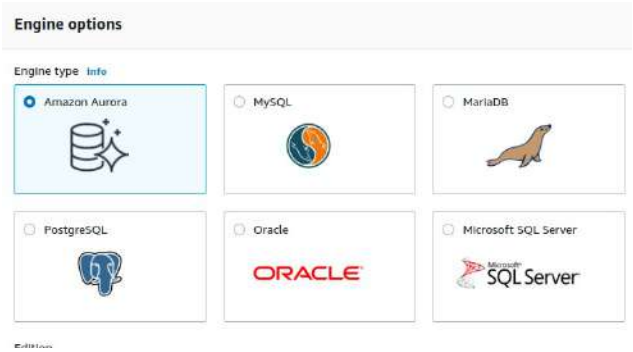
RDS es un PaSS (Plataforma como servicio). AWS se encargará de la gestión de la máquina y el cliente puede usar el software. Tampoco se puede configurar la BBDD o administrar su instalación, las actualizaciones…

Para controlarlo todo se debe crear una instancia y montar dentro la BBDD.

Recordatorio:



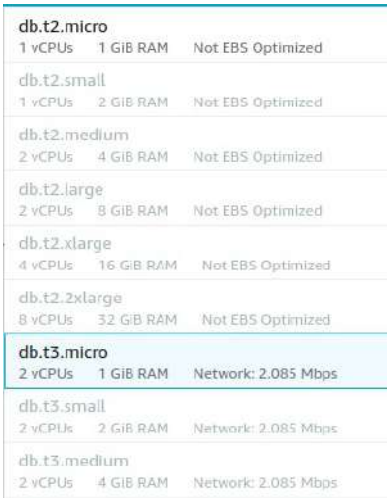
Si le damos a crear BBDD podemos ver las BBDD relacionales que podemos utilizar.



Aurora son BBDD parecidas a SQL pero no son lo mismo, es otra implementación.

15.1. - Crear BBDD en RDS Standard

- Tenemos dos **tipos de creación**:
  - **Standard** – Te pide algunas variables para su configuración.
  - **Easy** – Ofrece algunas configuraciones predefinidas por defecto
- **Tipo de BBDD**. Aurora, mysql, mariadb, etc
  - Podemos escoger la versión
- **Plantillas** – Según que escogemos nos dará unas opciones u otras.
  - Producción
  - Desarrollo
  - Free tier – Pruebas en una capa gratuita.
- **Opciones**
  - Nombre de la BBDD
  - Credenciales. Usuario admin y password
- **Configuración de la instancia**. Según el tipo de plantilla podremos seleccionar unas máquinas u otras, según la optimización que se requiera. Cada BBDD necesita unos recursos y según como sea combiene escoger una instancia u otra (ver documentación: https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/Concepts.DBInstanceClass.html). El precio que cuesta la BBDD está basado en la instancia que se elija. Las instancias empiezan por db. porque están diseñadas orientadas a las BBDD.



- **Almacenamiento.** No tenemos todos los tipos de disco, con las instancias varían. Normalmente se activa la autoescalada para que puede crecer hasta cierto tamaño. Indica un mínimo de espacio.
- **Disponibilidad y durabilidad**. Para despliegue en distintas AZ. Lo que hace es crear Instancias Standby, de réplica. No es posible usarlo con la plantilla Free tier.
- **Conexión**. Aquí decidimos
  - si queremos conectarla con una instancia EC2 (Tiene que estar arrancada)
  - La VPC
  - El grupo de subred
  - Si queremos acceso público.
  - El grupo de seguridad. Lo seguro es un grupo que solo permita el puerto de la BBDD
  - AZ
  - También podemos cambiar el puerto de conexión en la configuración adicional.
- **Tipo de autenticación**.
- **Monitorización**. Habilitar o no la monitorización.
- **Configuración adicional**.
  - Una BBDD inicial, nos pide el nombre-bucket
  - El grupo de parámetros es para especificar el comportamiento de la BBDD
  - En el grupo de opciones se especifican opciones.
  - El backup. Por defecto está habilitado. Debe estar habilitado si no es una prueba. Se guarda como un snapshot.

- Nos preguinta si queremos encriptación
- AWS KMS key
- Performance Insights – permite detectar posibles problemas de rendimiento o de datos. Es gratuita la versión de 7 días. La de 2 años es de pago.
- Exportación de Logs. (Cuidado, ocupan mucho espacio normalmente). Esto se exporta a CloudWatch
- IAM role
- Mantenimiento – Habilitar actualizaciones automáticas en pequeños cambios. Se pueden programar
- Se puede habilitar la protección de borrado.

- **Estimación costes mensuales**.

En caso de Oracle u otras BBDD con licencia, AWS no se responsabiliza y tenemos que adquirirla.

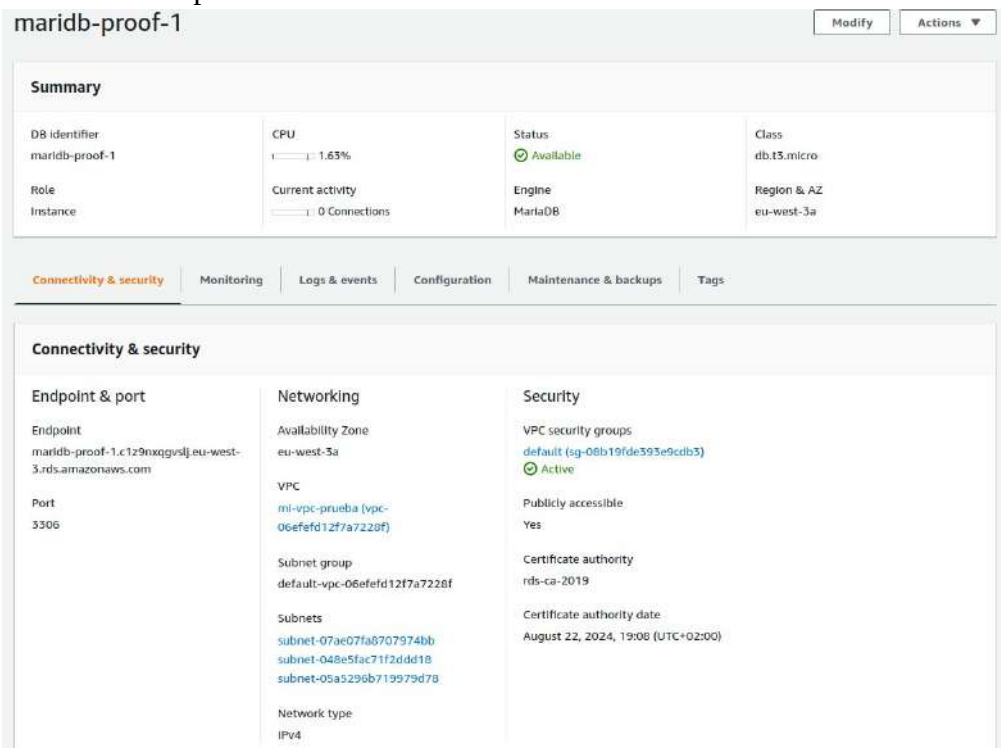
Es normal que tarde unos minutos en crearse.

15.2. - Crear BBDD en RDS Easy

Tan solo escoges el tipo de BBDD, el tipo de instancia (Desarrollo, producción o Free tier), el nombre de la bbdd y el usuario/pass. Luego un resumen y se crea la BBDD. easy

15.3. - Gestionar, modificar y borrar BBDD en RDS

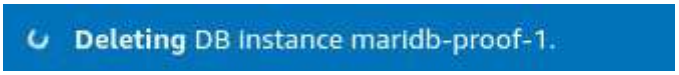
La vista de una BBDD es parecida a una instancia



- Connectivity & security
- Monitoring
- Logs & events - cabe la posibilidad de hacer un watch a un log en el mismo panel.
- Configuration
- Maintenance & backups – Aquí se pueden ver los snapshots que se van creando.
- Tags

Las modificaciones pueden tardar en aplicarse. En principio, cualquier modificación permite dejarlo para el momento del mantenimiento o realizarla inmediatamente.

Hay que tener en cuenta que el borrado también tardará.



Existen opciones diferentes para borrar según el tipo de BBDD.

- Se puede crear un snapshot final para posibles errores.
- Se pueden guardar los snapshot creados automáticamente, de los últimos 7 días.

Si no se guarda ningún snapshot deberemos confirmar nuestro consentimiento.

15.4. - Opciones subnet group, parameter groups y option groups

Subnet group son un conjunto de subnet que se pueden asociar en grupo a una BBDD. Es obligatorio tener un grupo de subnet en cada BBDD. Si no la creamos lo hace automáticamente y le llama default. Puede ser que no sea lo mejor, porque puede conectarse por una subnet que no sea deseada. Es importante crearla personalizada según las necesidades que tengamos.

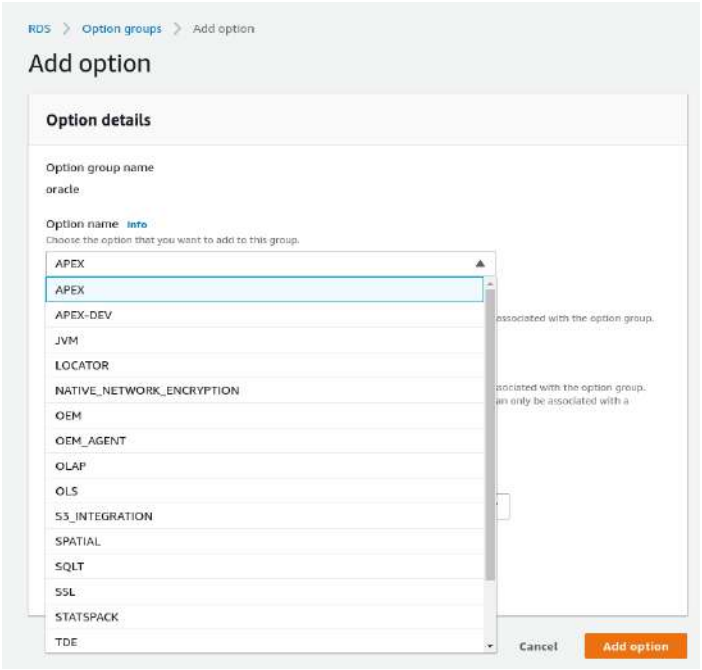
Tendremos que relacionar las subredes que tengamos publicas para poder conectarnos.

Parameter groups permiten crear una configuración para las BBDD. Como es un PaaS no se tiene acceso a los archivos de configuración del tipo de BBDD que escojamos. Entonces, esta opción permite cambiar algunos parámetros de la BBDD para poder cambiar esos archivos.

Se crean para tipo de BBDD concretas, para versiones concretas, y después de crearlo se puede entrar y modificar la mayoría de parámetros. Algunos parámetros no están permitido cambiar porque AWS considera que puede afectar al sistema.

Option groups es muy parecido al parameter groups pero está más orientado a las opciones. Creando un grupo deberemos escoger el motor, por ejemplo «oracle-ee».

Una vez creado podremos entrar dentro y comprobar las opciones (Conjunto de componentes) que nos permite añadir en el motor para cuando se arranque la BBDD.



Cuando cambiamos opciones de un grupo que se está usando desde una BBDD, nos da la opción de aplicar inmediatamente o esperar a la hora de mantenimiento.

15.5. - Propiedades y parámetros concretos según la BBDD

Según el tipo de BBDD tendremos unas propiedades distintas. Para verlas todas las posibilidades, tendremos que escoger la plantilla de producción.

15.5.1. - MariaDB

Licencia GPL. Gratuita.

- El tipo de almacenamiento permite
  - gp2 – Es la económica.
  - io1 - IOPS provisionadas puede subir bastante los costes. Estas BBDD tienen un rendimiento muy optimizado.
  - Magnetic – Esta pensado para entornos legacy o BBDD poco utilizadas.
- Se puede tener autoescalada
- Permite standby (Modo replica)
- En los grupos de parámetros y opciones, se tendría que tener previamente uno creado específico para Mariadb para que nos de la opción de escogerlo.

15.5.2. - MySQL

Licencia GPL. Gratuita. Desde que Oracle compró Sun Microsystem y adquirió mysql, entre la versión 5 y la 8 han habido muchos cambios, sobretodo de conectividad.

- En disponibilidad y durabilidad, además del multi-AZ y el mono-AZ, también se soporta el multi-AZ cluster. Se activo a partir de la versión 8.0.26. Es como crear una primaria con dos réplicas. Está en modo beta (preview). Aparece en el tutorial pero no en la creación actual de la BBDD.
- En la autenticación se puede autenticar también por kerberos.

El resto de propiedades son iguales que mariadb.

15.5.3. - PostgreSQL

Licencia GPL. Gratuita. Es exactamente igual que MySQL.

15.5.4. - Oracle

Es una versión comercial.

- En el vídeo-tutorial dice que existe dos posibilidades de gestión:
  - Amazon RDS normal – Será AWS quien gestione la BBDD. Un PaaS en toda regla.
  - Amazon RDS custom – Permite una personalización de BBDD, con cierto acceso.
- Se puede utilizar una arquitectura multitenant. Contiene una BBDD primaria (Contenedora. CDB) con otras secundarias (PDB)
- Permite dos clases de versiones (AWS no gestiona licencias, se debe comprar a parte):
  - Oracle Enterprise Edition. Hay que tener licencia
  - Oracle Standard Edition Two. Permite que AWS te ofrezca la licencia y te la cobra.
- No hay plantilla Free tier. Directamente es en producción o en desarrollo.
- En las instancias, lo mínimo que permite Oracle son máquinas de 2 GB de Ram. En Memory optimized y Standard Classes permite incluir los core e hilos. Además, se puede incluir memoria adiccional.

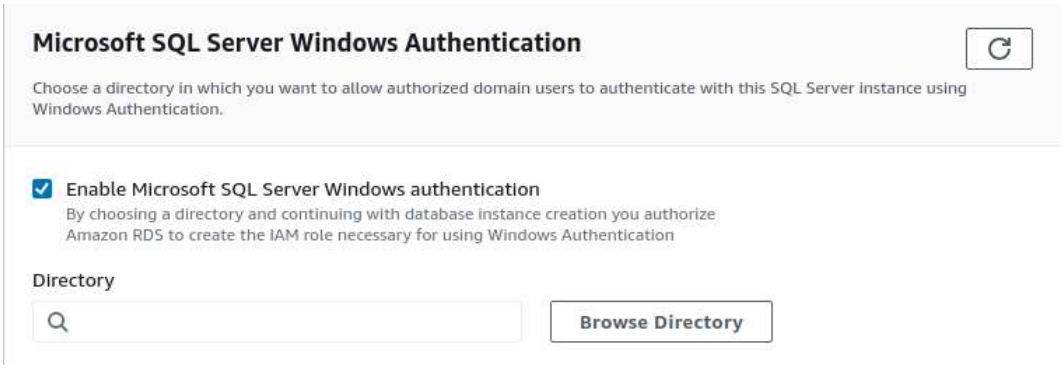
15.5.5. - Microsoft SQL Server

Es una versión comercial.

- En el vídeo-tutorial dice que existe dos posibilidades de gestión:
  - Amazon RDS normal – Será AWS quien gestione la BBDD. Un PaaS en toda regla.
  - Amazon RDS custom – Permite una personalización de BBDD, con cierto acceso.
- Permite escoger entre 4 edicciones

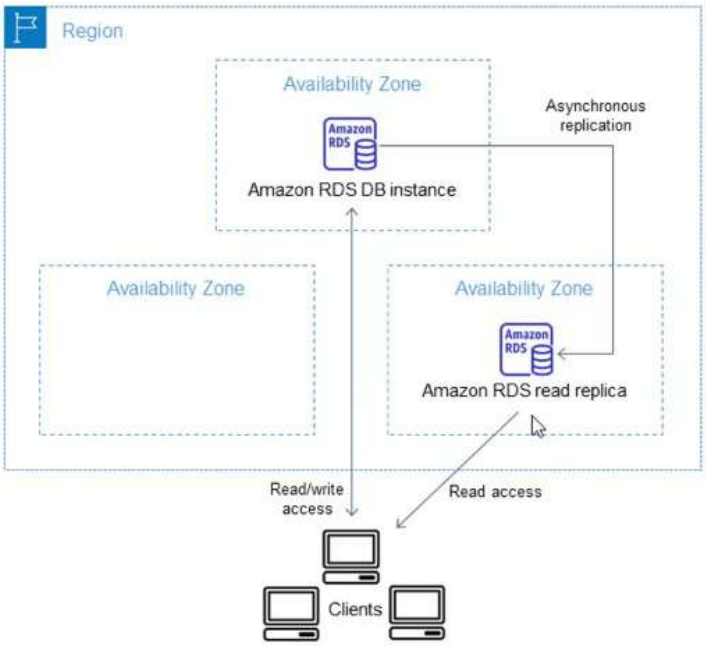
- SQL Server Express Edition. Version gratuita hasta 10 GB. Solo se puede utilizar en la plantilla Free Tier. Exige 2 cpu y 2 GB de ram como mínimo, solo puedes escoger instancias pequeñas de t3.
- SQL Server Web Edition. Es una versión limitada para web. Solo se puede utilizar en la plantilla de producción . Exige 2 cpu y 2 GB de ram como mínimo
- SQL Server Standard Edition. Se puede escoger entre producción y desarrollo. Exige 2 cpu y 8 GB de ram como mínimo
- SQL Server Enterprise Edition. Es la versión completa. Se puede escoger entre producción y desarrollo. Exige 4 cpu y 16 GB de ram como mínimo

- Tiene una sección especial de Microsoft SQL donde se puede incluir la autenticación de un Directorio Activo



15.6. - Read replica

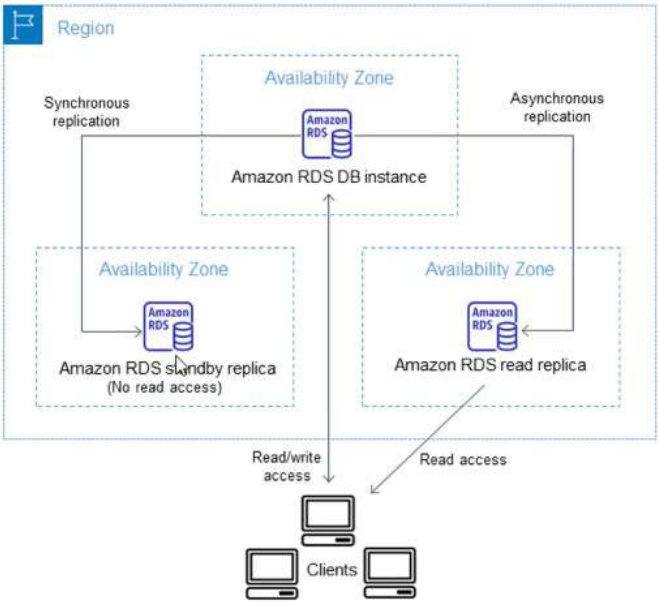
Se puede crear una Read Replica para que en una region, los clientes puedan acceder a la BBDD. Se replican de manera asincrona, con lo cual, se puede crear un acceso de escritura directo a la BBDD y otro acceso de solo lectura a la Read Replica.



De pendiendo del motor de BBDD se puede crear más de una Read Replica.

Utiliza la tecnología de las propias BBDD (mysql, oracle, etc) para crear estas replicas.

Si tenemos una Multi-AZ, también se puede crear una Read Replica.



La replicación en standby (multi-AZ) es sincrono. Está más pensada para backup. Aunque la Read Replica también se puede usar para backup.

Se tiene que tener en cuenta que puede existir un desajuste de los datos con la Read Replica por ser asincrona la replicación.

Podemos promover una replica de solo lectura a una instancia normal y corriente, para utilizarla como una BBDD normal. Para promover se debe parar la BBDD original.

Para mantener la arquitectura se debe crear otra Read Replica.

15.6.1. - Crear Replica de Lectura

Para la creación de una Replica de lectura es muy parecida en todas los tipos de BBDD menos en oracle y SQL Server, las Enterprise. El concepto es el mismo pero porque tiene componentes distintos.

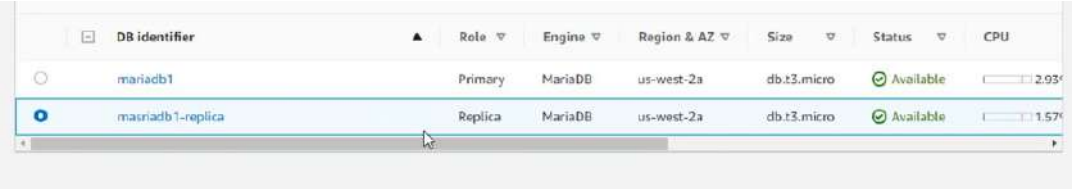
Hay un límite de 5 replicas por BBDD. Tiene un coste por los recursos que utiliza. No podemos crear una replica de una BBDD si no tenemos activados los backups.

En «Actions» seleccionamos «Create read replica» y nos lleva a la configuración con los mismos parámetros que una BBDD original:

- Decimos la clase de instancia.
- Si queremos la replica en multi-AZ o no
- El tipo de instancia
- La region AWS – Podemos escoger una region distinta para la replica.
- Tipo de almacenamiento
- Availability & durability
- Conectividad
- Autenticación
- Configuración adicional

- Backups
- Encriptación
- Monitorización
- Logs
- Mantenimiento

Hereda muchos campos de la BBDD original. Realmente, la única diferencia es que no se puede escribir, es de solo lectura.



15.6.2. - Promover Replica de Lectura

Marcando la replica, en «Actions» seleccionamos «Promote» y nos da la opción de Habilitar Backup y su periodo. Entonces pasará a ser una BBDD primaria.

Esto significa que dejará de replicarse con la primera, y la primera estará igual.

15.7. - RDS Reservadas

Es igual que las instancias reservadas. Consiste en alquilar una RDS para un largo periodo y así ahorrar costes respecto al on-demand. De la misma manera se puede pagar el total, parcial, etc.

Igual que pasa con las instancias, cuando se reserva aunque no se use te siguen cobrando.

15.8. - CLI RDS

Crear una base de datos RDS de tipo Mysql

aws rds create-db-instance –db-instance-identifier db-20 --db-instance-class db.t3.micro --engine mysql –master-username admin1 --master-user-password lepanto1 --allocated-storage 20

Comprobar que se ha creado, tanto en la consola como en modo comando

aws rds describe-db-instances --query DBInstances[].DBInstanceIdentifier

Crear un snapshot de la Base de datos

aws rds create-db-snapshot --db-instance-identifier db-20 –db-snapshot-identifier snp-20

Comprobar en la consola y en línea de comandos que se ha creado

aws rds describe-db-snapshots

Visualizar solo el nombre del snapshot

aws rds describe-db-snapshots --query

También se puede poner sin nombre

aws rds describe-db-snapshots --query "DBSnapshots[*].DBSnapshotIdentifier"

Borrar el snapshot

aws rds delete-db-snapshot --db-snapshot-identifier snp-11

Borrar la Base de datos

aws rds delete-db-instance --db-instance-identifier db-02 --skip-final-snapshot

También se puede pedir el nombre de la snapshot, si sabemos concretarla, meterla en una variable bash y utilizarla para borrar:

snapshot=$(aws rds describe-db-snapshots --snapshot-type manual --query DBSnapshots[*].DBSnapshotIdentifier --output text)

aws rds delete-db-snapshot --db-snapshot-identifier $snapshot

Borrar la BBDD

aws rds delete-db-instance --db-instance-identifier db-02 –skip-final-snapshot

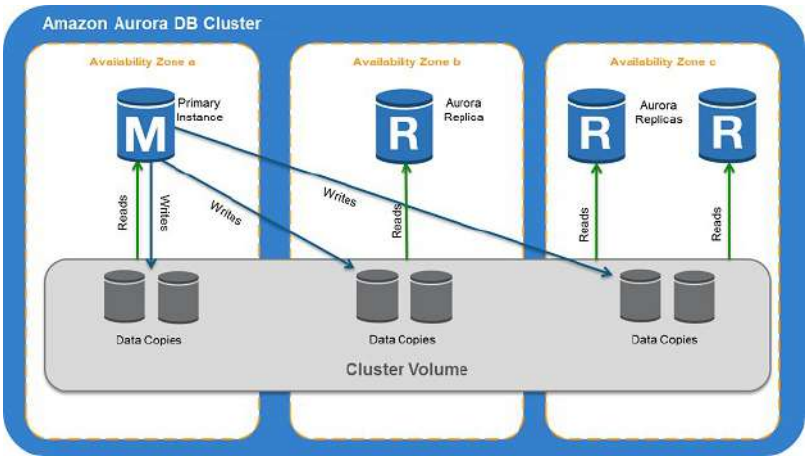
TEMA 16 - RDS Aurora

Aurora DB clusters es una versión compatible con mysql y postgres.

Pueden trabajar como un servidor normal y en serverless.

Los **clusters** pueden ser:

- **single-master** (un solo master y réplicas) – Una instancia primaria y hasta 15 réplicas (solo lectura). A diferencia de las réplicas de las otras BBDD, estas comparten un cluster de almacenamiento. AWS recomienda que la arquitectura se reparta en distintas AZ.
- **multi-master** (todas las instancias de aurora serían de escritura y lectura) - Solo puede tener hasta 4 instancias



16.1. - Crear BBDD tipo Aurora

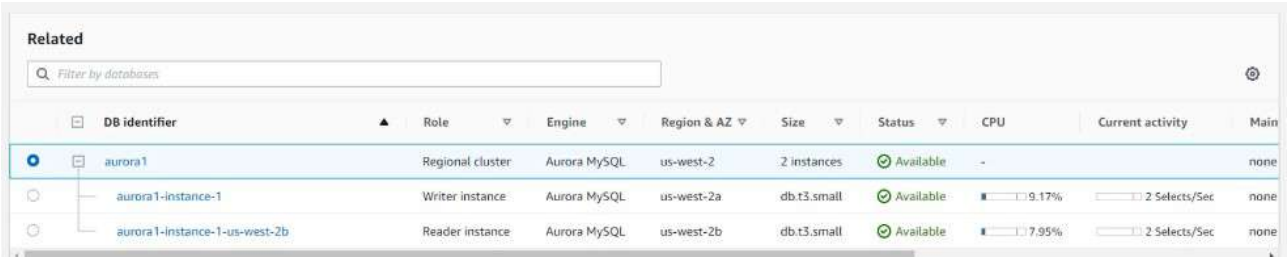
- Edición
  - mysql
  - postgreSQL
- (No aparece single-master y multi-master)
- Permite mostrar
  - Versiones que soportan BBDD global
  - Versiones que soportan consultas paralelas. Traslada el procesamiento a la capa de almacenamiento. Habría que crear un «parameter group» con el parámetro aurora_parallel_query activado.

- Versiones severless,

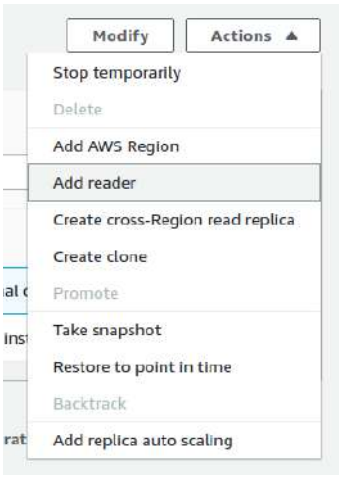
- Plantillas- No hay Free Tier, solo producción y desarrollo.
- Configuración Instancias según la versión que hemos escogido.
- En la configuración adicional hay una casilla de prioridad failover para escoger entre las 15 posibles replicas la preferente. En caso necesario, de forma automática, pasará la master a la escogida.

- El backup es obligatorio
- BackTrack – Permite volver a un tiempo de la BBDD si necesidad de recrear la instancia.

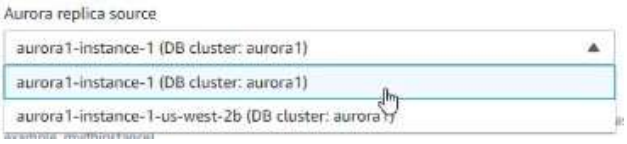
El resto de posibles configuraciones son similares al resto de BBDD.



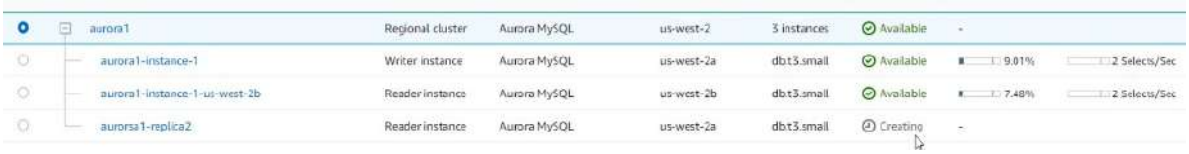
Se puede añadir regiones, o réplicas de lectura en “actions”



Cuando se añade una réplica nueva se puede escoger de fuente la original u otra replica

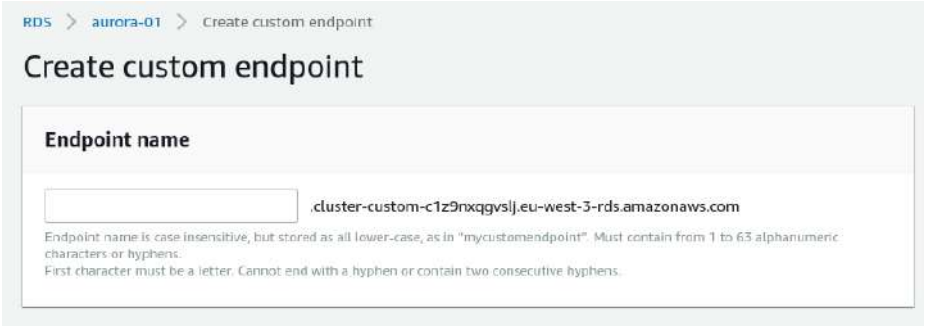


Vuelve a dar la opción de escoger la prioridad del failover.



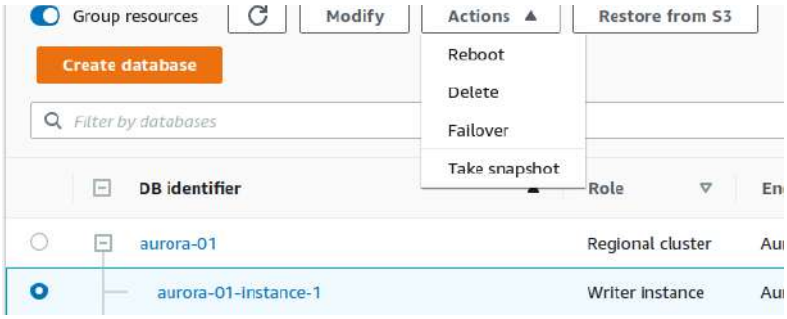
Los endpoints son globales para todas las réplicas, uno de escritura y otro de lectura (Con prefijo -ro-).

Se puede **personalizar los endpoints**

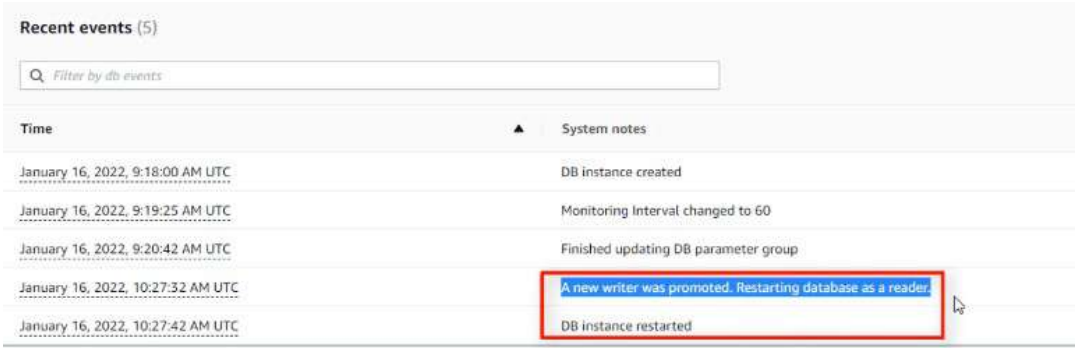


Pueden servir para entrar a réplicas concretas configuradas con instancias más potentes por requerimientos de algún servicio o todo lo contrario, un endpoint a réplicas más ligeras para servicios con pocos requerimientos.

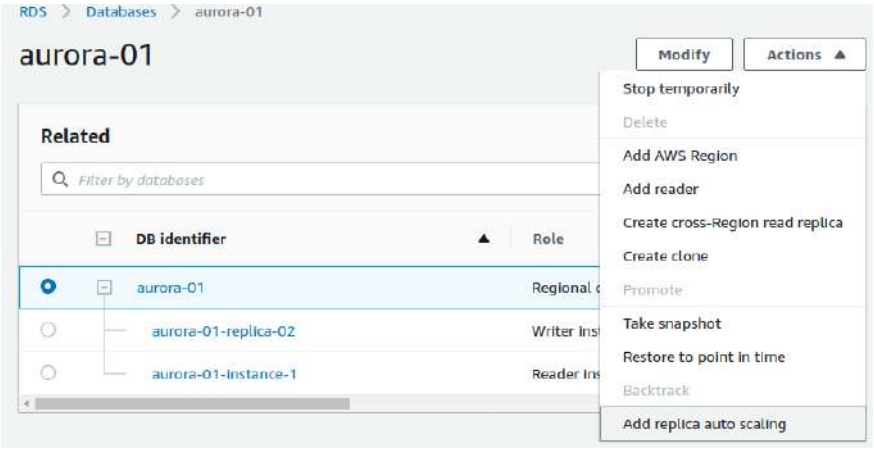
Se puede crear un **failover de una réplica** concreta de escritura con el botón “Actions”.



Para comprobarlo se puede ver en los logs del cluster

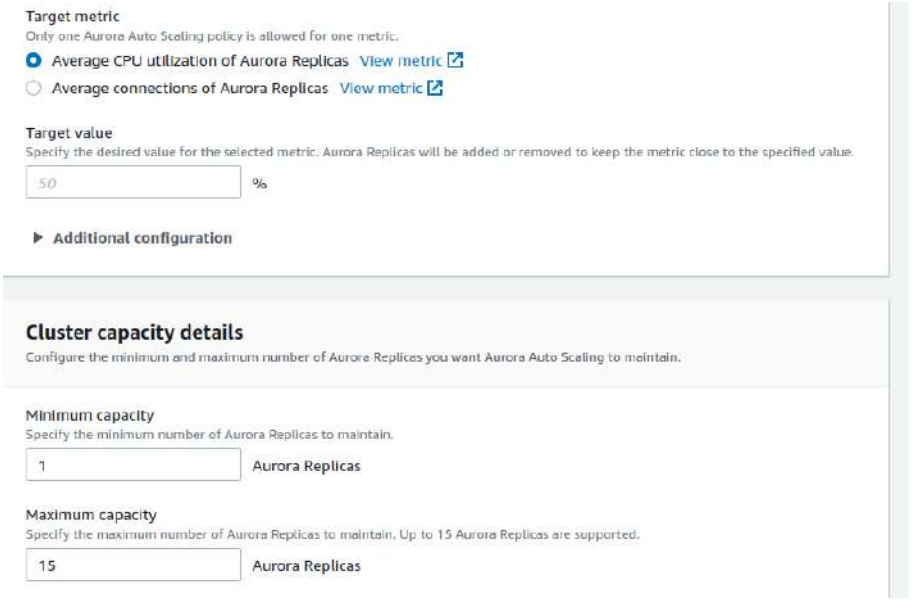


Se puede crear una **réplica de autoescalada** que pueden venir bien si se desconoce el ciclo de vida que pueden tener las réplicas. Se selecciona el clúster y en “actions” está la opción de “Add replica auto scaling”.

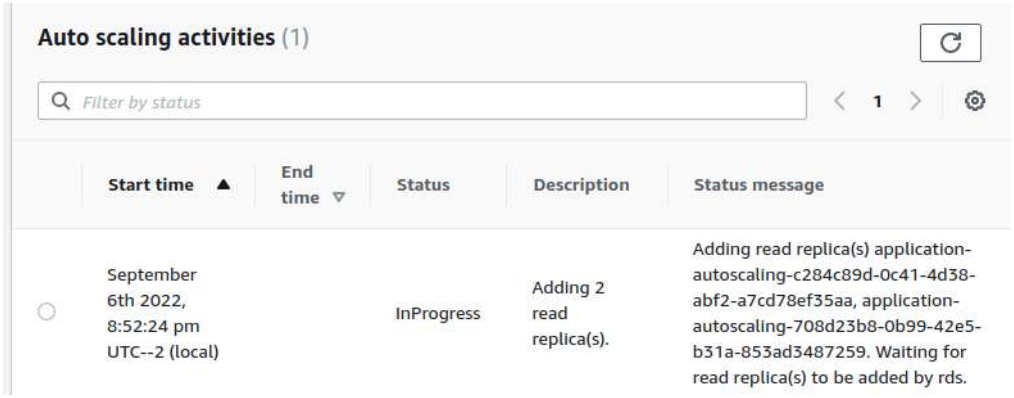


Se le nombra a la política. Se puede

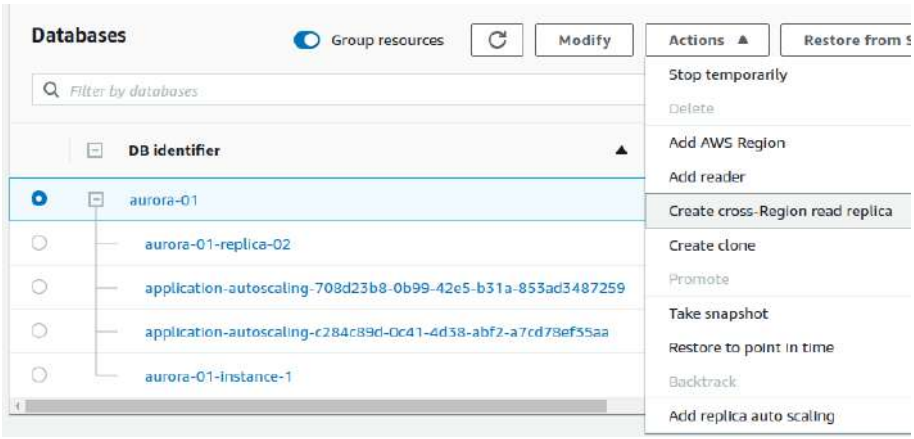
- basar en métricas de CPU o en las conexiones. También se puede determinar los segundos de espera para crear o reducir.
- Indicar el mínimo y el máximo de réplicas.



En el momento que necesita las replicas se pueden ver en los logs de autoescalada.

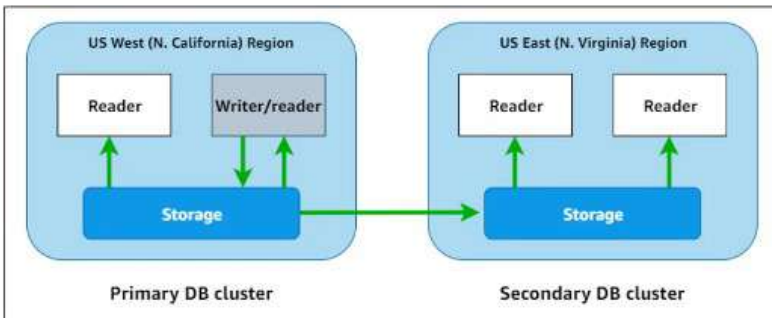


También se puede **replicar el cluster en otra region**. Se presiona “Actions” y “Create cross-Region Read replica”.



Es conveniente seleccionar multi-AZ porque si la replicación se interrumpe se deberá volver a empezar. Es importante antes crear en la region destino el grupo de subnets necesario.

También se puede **extender el cluster a otra región** para hacerlo más grande. Es en “Actions” y “Add AWS Region”. Es la manera de hacerlo global.



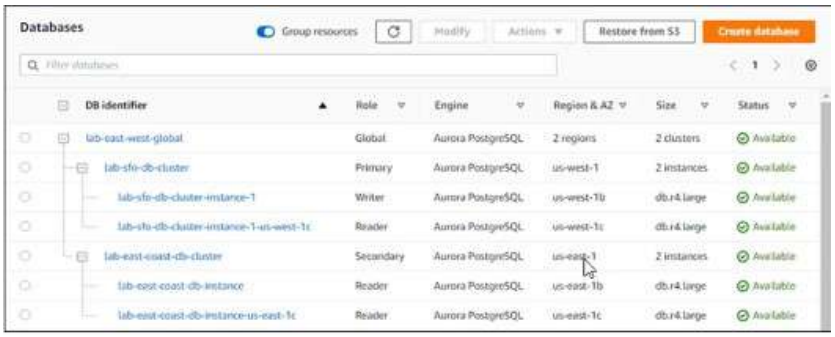
Aurora global databases tiene algunas limitaciones:

- No está en todas las regiones.
- Requiere algunos parámetros de configuración muy concretos.
- Hay unas versiones de mysql y de postgreSQL que lo soporta.
- No soporta Multimaster (No está actualmente en RDS), Serverless, Backtrackin ni RDS Proxy.
- Necesitan instancias memory-intensive.

Resumen:

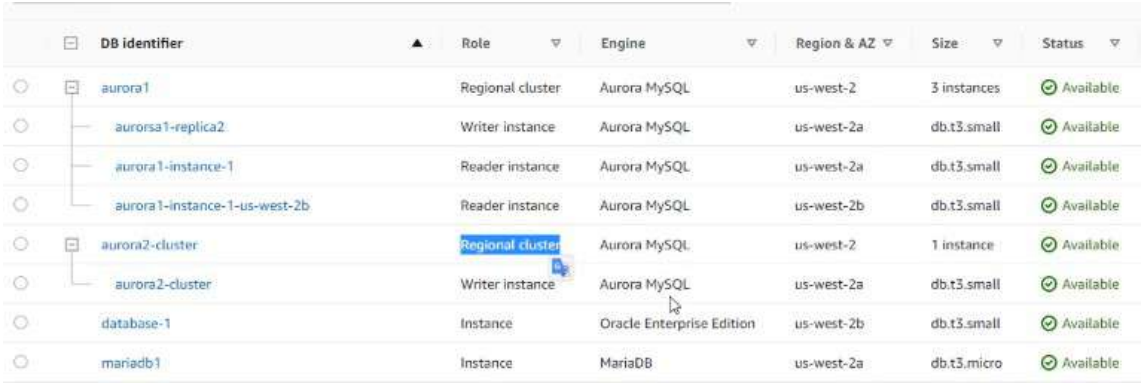
Description	Primary AWS Region	Secondary AWS Regions
Aurora DB clusters	1	5 (maximum)
Writer instances	1	0
Read-only instances (Aurora replicas), per Aurora DB cluster	15 (max)	16 (total)
Read-only instances (max allowed, given actual number of secondary Regions)	15 - s	s = total number of secondary AWS Regions

Aparecería así en las BBDD:



También se puede **clonar el cluster**. En “Action” y “Create clone”. Es la manera más rápida de crear una copia de las BBDD de Aurora.

Hereda el motor, la instancia y la conectividad entre otras cosas. De todas formas algunas propiedades se pueden modificar para que lo haga en la copia. Queda como un clúster independiente del primero:



DB identifier	Role	Engine	Region & AZ	Size	Status
aurora1	Regional cluster	Aurora MySQL	us-west-2	3 instances	Available
aurorsa1-replica2	Writer instance	Aurora MySQL	us-west-2a	db.t3.small	Available
aurora1-instance-1	Reader instance	Aurora MySQL	us-west-2a	db.t3.small	Available
aurora1-instance-1-us-west-2b	Reader instance	Aurora MySQL	us-west-2b	db.t3.small	Available
aurora2-cluster	Regional cluster	Aurora MySQL	us-west-2	1 instance	Available
aurora2-cluster	Writer instance	Aurora MySQL	us-west-2a	db.t3.small	Available
database-1	Instance	Oracle Enterprise Edition	us-west-2b	db.t3.small	Available
mariadb1	Instance	MariaDB	us-west-2a	db.t3.micro	Available

Para borrar el cluster completo hay que ir borrando instancias hasta que no quede ninguna. Lo ideal es hacerlo desde CLI con un bucle que se ocupe de borrarlas todas. Las instancias en modo gráfico se borran en “Action” y “Delete”. En la última instancia nos pregunta si queremos guardar un snapshot.

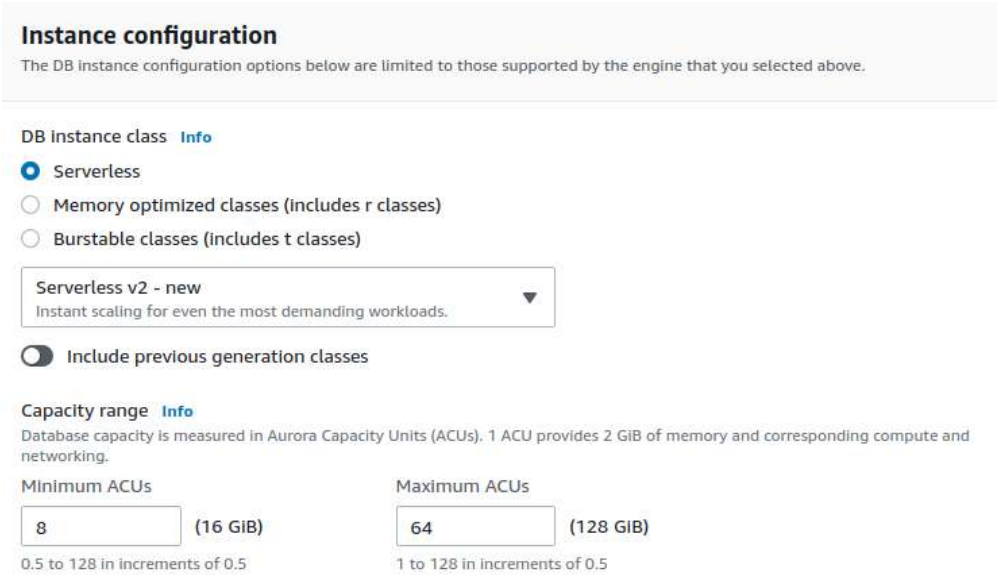
Mientras borramos podemos comprobar que si borramos primero la réplica de escritura, igual que el failover, hará que cualquiera de las otras réplicas se hará de escritura para sustituirla.

No se puede parar réplicas individualmente, se debe parar el clúster completo.

Para **crear una multi-master** (No lo veo, lo explica en el curso, creo que AWS ya no permite multi-master) solo hay unas versiones concretas y en mysql-aurora. El resto de propiedades son identicas. Pero necesita unas características concretas.

El modo **serverless** en Aurora tiene dos versiones de mysql y 3 de postgreSQL. En este modo tendremos la opción de escoger sin servidor (serverless) en la configuración de la instancia.

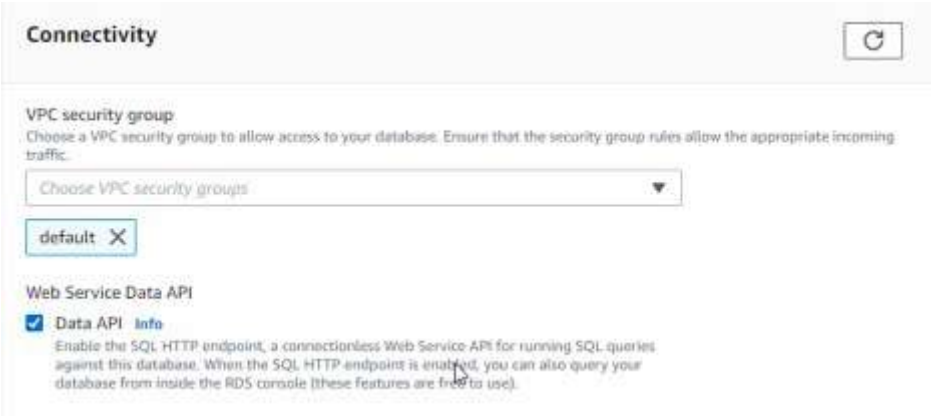
Nos permite escoger un rango de ACUs (Aurora Capacity Units. Más o menos un ACU equivale a 2 GB y se le asocia una cantidad de CPU y de red). Con serverless se hace un escalado de recursos para según las necesidades que se van teniendo.



Los cambios entre el rango de ACUs, AWS lo hace cuando menos queries tiene. Se recomienda no hacer queries muy largas para darle espacios a los posibles cambios.

En serveless no hay réplicas, es directamente la escala de capacities que hemos indicado.

El **Query editor** es una forma de conectarse a las BBDD serverless, pero necesita el Data API habilitado. Para habilitarlo se debe modificar la BBDD serverless y en conectividad se puede activar la opción.



Entonces ya nos podremos conectar a Query editor y ejecutar las queries online.

Es muy sencilla, para trabajar siempre es mejor un editor externo. Pero se pueden hacer pruebas.

TEMA 17 - RDS Backup y Mantenimiento

Los backups indican:

- Tiempo que entre las copias
- El tiempo que puede retroceder entre los backups existentes
- El momento en que se hacen las copias.

Un backup es la creación de snapshot de un momento concreto. En los automáticos le da el nombre snapshot-nombrebbdd-fecha. Modificando la BBDD se puede modificar algunos aspectos del Backup como el periodo de retención como el horario aproximado para realizarse.

En los Automated backups constan los backups definidos con un resumen de sus propiedades y las snapshots.

En Snapshots aparecen por tipos:

- Manuales – Se hacen eligiendo la BBDD y “Take snapshot”. También en snapshot, en “Take snapshot”
- Automáticas
- Compartidas por mi
- Públicas
- Del servicio de Backups – Es un servicio a parte para hacer backups, y aquí aparecerá los backups que se hagan desde este servicio y que estén relacionados con las BBDD.
- Exportadas en S3

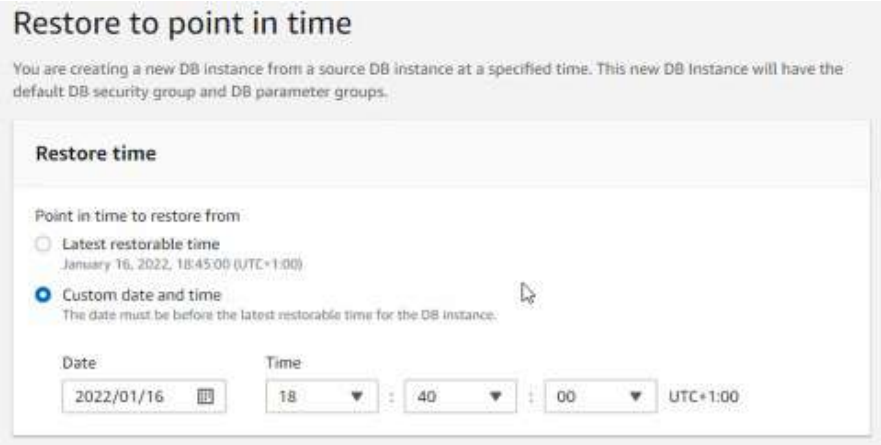
En “Actions” de los snapshots se puede **restaurar, copiar, compartir, migrar y borrar.**

Copiar es interesante porque te lo puedes llevar a otra región. Sería como un backup del backup.

Cuando se **comparte** puedes indicar la ID de la cuenta AWS o lo puedes hacer público. El incoveniente que tiene es que se debe compartir la clave por defecto del encriptado. Con lo cuál, tendrá que tener una clave de encriptación (KMS) concreta para esto, no puede ser la clave por defecto que indica AWS.

Para **restaurar**, tan solo hay que darle al botón en “Actions” y lo que hará es generar una nueva BBDD, a la cuál se le puede poner un nombre nuevo y modificar algunos de los parámetros. Es mejor no cambiar el tipo de almacenamiento porque puede tener problemas en la migración.

Una recuperación de un punto en el tiempo de un backup (En Actions/Restore to point in time), en la que vas una semana atrás, por ejemplo, hace perder todas las líneas que se hayan escrito en ese tiempo.



El tiempo que tarde en restaurar será equivalente al que se tardó en crear más el tiempo que le lleve recuperar todos los datos de la BBDD.

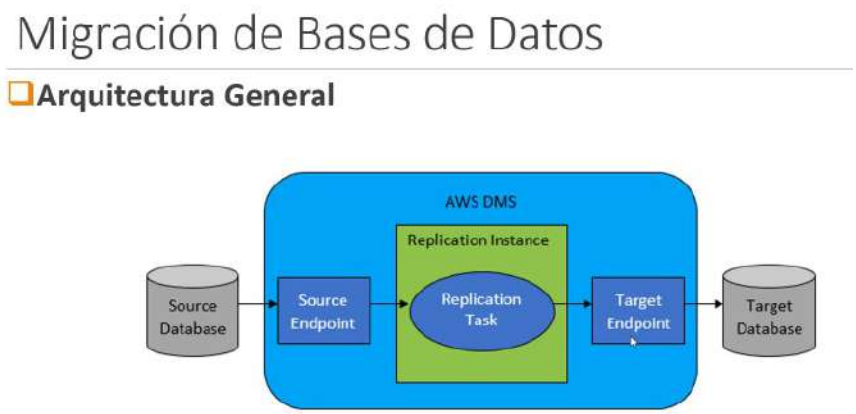
Si se escoge una versión de BBDD antigua, se puede simular como AWS te recomienda actualizar la versión a la actual. Cuando se deben hacer esta clase de actualizaciones, siempre es recomendable efectuar un backup antes de empezar. El **mantenimiento** aparecerá en la pestaña “Maintenance & backups” de la BBDD, en este lugar se pueden aplicar los mantenimientos/actualizaciones.

TEMA 18 - RDS Migración de Bases de Datos

Hay un servicio en concreto para esto con su propio dashboard: Database migration service.

- Permite migrar muchos tipos de bases de datos, tanto en modo cloud como en modo on-premise.
- Se pueden realizar migraciones únicas y también los cambios que se puedan ir produciendo.
- Se pueden migrar entre distintos motores de BBDD
- Se pueden perfilar y mejorar la migraciones con la herramienta SCT (Schema Conversion Tool). Se descarga en local y permite definir los esquemas de entrada y salida.
- Se pueden hacer transformaciones y conversiones en el proceso.

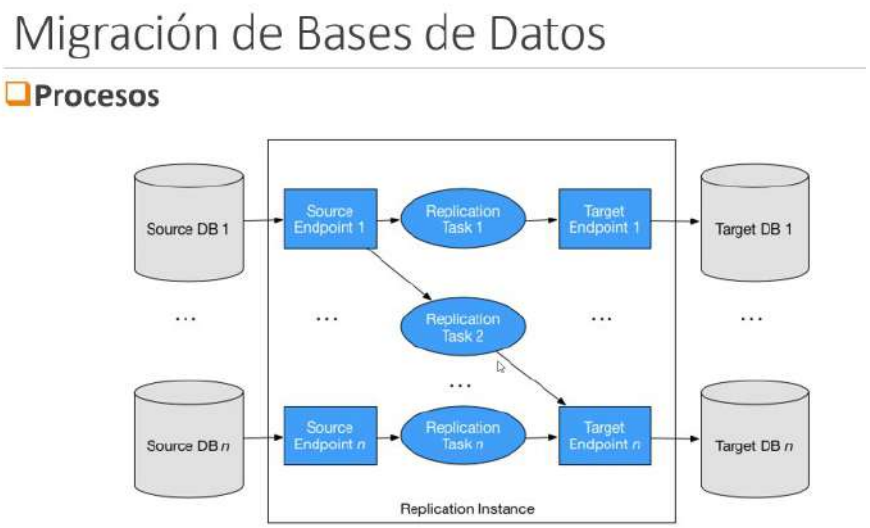
Se necesita una instancia para efectuar la migración. En esta máquina se hará todo el trabajo pesado de replicación. Hay que tener en cuenta el costo de la instancia.



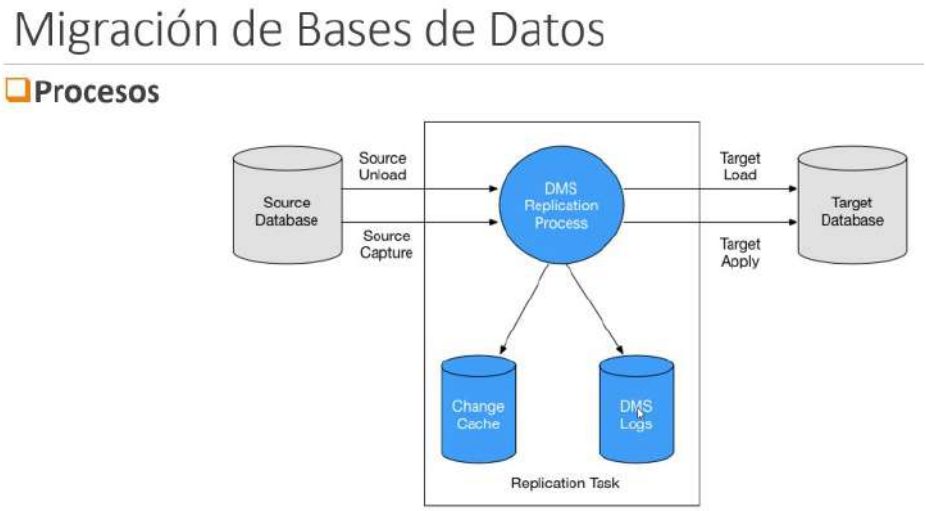
Tendremos un origen apuntando a un endpoint de entrada y un endpoint de salida apuntando al destino de la migración.

En la documentación podemos encontrar las BBDD origen y destino que se soportan en AWS.

Como trabaja es creando replication task que son independientes.



Creará una cache de cambios y logs de cada proceso de replicación



18.1. - Hacer migración

La migración se hace en AWS DMS (Database Migration Service).

Lo primero que debemos hacer es una replication instances. Hay instancias especializadas para esta cuestión. Dependiendo de las necesidades de la migración necesitaremos una instancia más grande o más pequeña, existen de todos los tamaños.

Las propiedades que le podemos dar son:

- Nombre
- Nombre descriptivo para el sufijo del ARN

- Descripción
- Clases de instancias
- Versión de la ingeniería. Se debe asegurar en la documentación que acepta el tipo y versión de BBDD. Se puede escoger una versión Beta
- Tamaño del almacenamiento. No es solo el tamaño de la BBDD, hay que contar que habrá archivos temporales, procesos, etc además del tamaño de la BBDD.
- VPC
- Opción Multi-AZ. Si se escoge tendremos una replicación primaria y el resto serán standby. Puede venir bien si creemos que puede dar errores un proceso de migración.
- Acceso público
- Grupos de subnet. Son igual que las de BBDD, pero no son las mismas. Hay que crearla antes de empezar a crear la instancia
- Zona habilitada
- Grupo de seguridad
- KMS key
- El modo mantenimiento
- Tags. Importante en desarrollo y producción tenerlos para poder filtrar búsquedas

Estas máquinas aparecerán en el servicio DMS, no en EC2 ni en otro lugar. La manera de gestionarla es en la consola correcta.

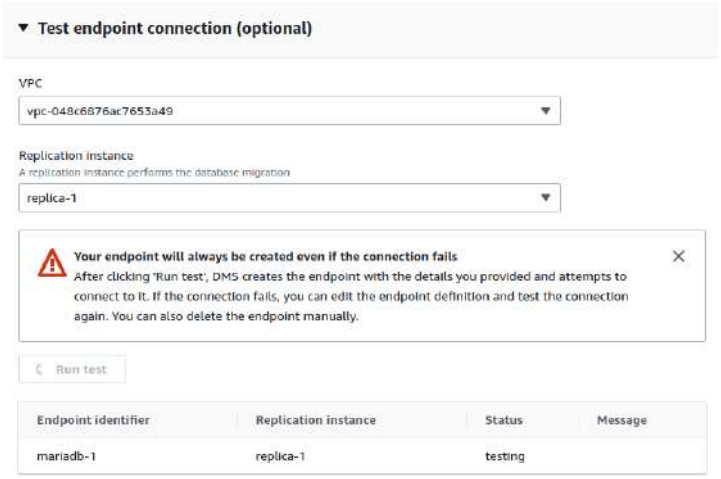
Ahora debemos **preparar el entorno** de la migración. **DMS tiene que poder conectarse a las BBDD** tanto de origen como de destino. O bien con acceso público o por VPN o con conexión interna….

Tenemos que **tener un usuario en la BBDD para la migración** con los permisos adecuados para los datos a migrar.

Ahora tendremos que **crear los endpoint** de tipo fuente y de tipo destino. Cuando escogemos una BBDD de RDS tendremos que marcarlo para que nos aparezca. Si la BBDD está en otra ubicación deberemos dejarlo desmarcado. Las propiedades de una BBDD en RDS son:

- Deberemos marcar el tipo de acceso, si es “AWS Secrets Manager” (Se verá más adelante, permite guardar contraseñas) o “Provide access information manually” (Introduccimos la contraseña de manera manual)
- KMS key
- Tags

- Test conexión endpoint – para probar si se puede conectar.



Si todo va bien aparecerá successful

Y creamos endpoint.

Las propiedades de una BBDD fuera de RDS son las mismas pero cambian algunas:

- En la conexión deberemos especificar:
  - Nombre de servidor
  - Puerto
  - Nombre usuario
  - Contraseña

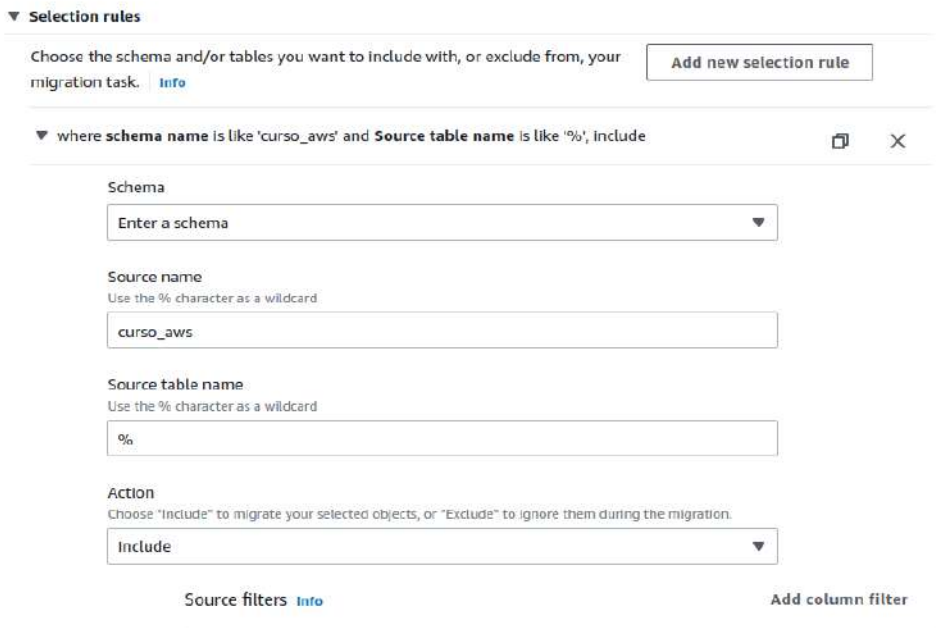
Es IMPORTANTE efectuar los test de conexión para evitar problemas o confusiones.

Ahora tendremos que **crear una tarea de migración**. Se pueden definir los siguientes parámetros:

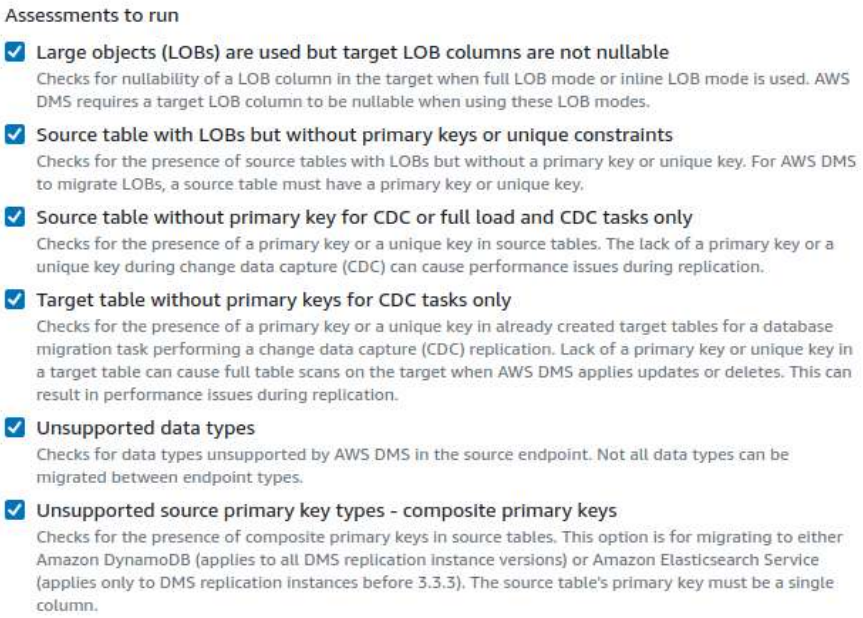
- identificado de tarea
- Prefijo ARN
- Indicar la instancia replica
- indicar las BBDD de origen y destino
- Tipo de migración. Puede ser:

  - Migrar datos existentes
  - Migrar datos existentes y replicar los cambios que vayan sucediendo.
  - Replicar los cambios que vayan sucediendo

- En las opciones se pueden poner a través de un asistente Wizard o de un archivo JSON. El JSON tiene más posibles opciones pero normalmente no es necesario. Ejemplos de opciones:
  - Las tablas en destino se pueden borrar, truncar o no hacer nada.
  - Que se hace con las LOB (Large Objects. Que contienen objectos grandes. Sueler ser binarios con archivos adjuntos.). Se pueden no incluir, incluir todo o un modo limitado.
  - Podemos habilitar una validación al final que tardará equivalentemente a lo que tarde en migrar.
  - Podemos añadir logs a CloudWatch
  - Crear una control table.
- Mapeos de tabla que también tiene opción Wizard y por archivo JSON.
  - Añadir nueva regla schema
  - Decir el schema concreto
  - Las tablas
  - Y la acción.

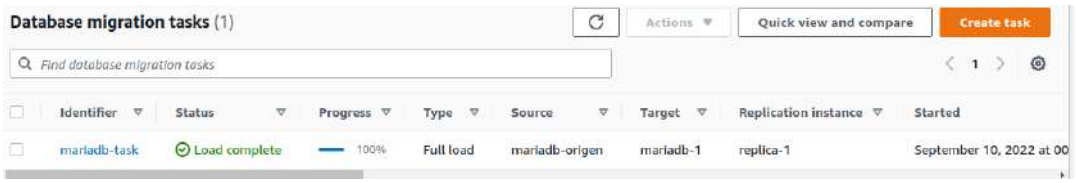


- Reglas de transformación. No es una ETL (Extraer, transformar y cargar) que en AWS es el servicio GLUE, mucho más completo.
  - Escoger un origen de schema, tabla o Columna
  - Nombre del schema
  - Nombre de tabla
  - Y escoger la acción. (Mover a, renombrar, hacer mayúscula o minúsculas, añadir prefijo, etc)
- Premigration assessment. Sirve para hacer una serie de pruebas antes de empezar la migración para buscar posibles errores. El informe lo deja en un bucket S3 Se le da un nombre identificativo y se puede comprobar



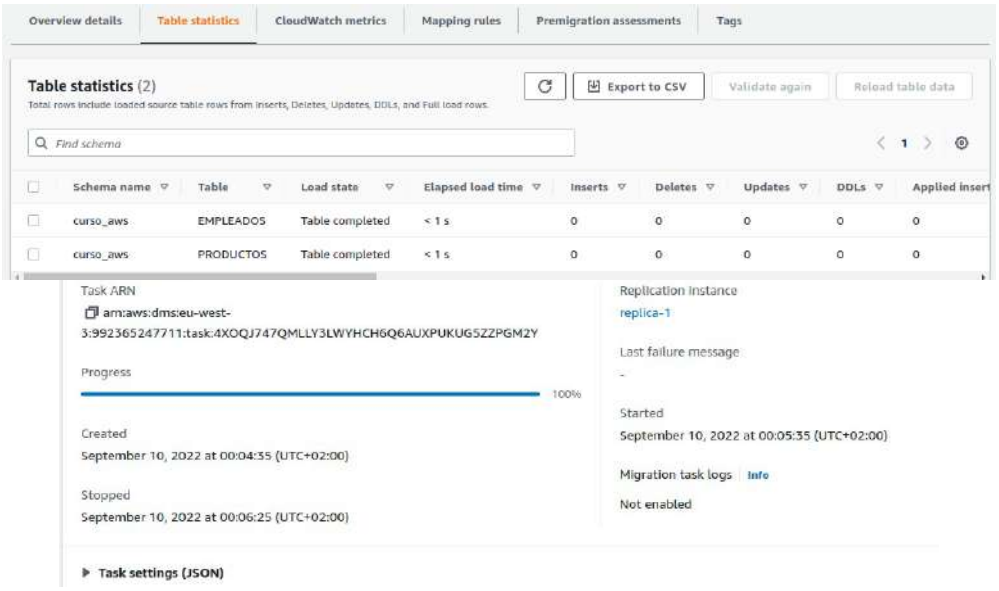
- Pregunta si comienza la migración solo crear la tarea o posteriormente.

Una vez se completa la tarea nos da bastante información:



- Detalles generales

- Estadísticas de tabla. Se pueden comprobar las filas y muchos otros datos



- Las métricas CloudWatch
- Mapping rules (Reglas de mapeo)
- La premigración.
- Etiquetas

Hay que prestar especial atención cuando la migración es en BBDD de origen y destino de diferentes tipos.

18.2. - SCT (Schema Conversion Tool)

La herramienta **SCT (Schema Conversion Tool)** nos puede ayudar a la conversión de datos concretos que den problemas. Nos permite algunas conversiones:

Source Database	Target Database on Amazon RDS
Oracle Database	Amazon Aurora MySQL-Compatible Edition (Aurora MySQL), Amazon Aurora PostgreSQL-Compatible Edition (Aurora PostgreSQL), MariaDB 10.5, MySQL, PostgreSQL
Oracle Data Warehouse	Amazon Redshift
Microsoft Azure SQL Database	Aurora MySQL, Aurora PostgreSQL, MySQL, PostgreSQL
Microsoft SQL Server	Amazon Redshift, Aurora MySQL, Aurora PostgreSQL, Babelfish for Aurora PostgreSQL (only for assessment reports), MariaDB, Microsoft SQL Server, MySQL, PostgreSQL
Teradata	Amazon Redshift
IBM Netezza	Amazon Redshift
Greenplum	Amazon Redshift
HPE Vertica	Amazon Redshift
MySQL	Aurora PostgreSQL, MySQL, PostgreSQL
PostgreSQL	Aurora MySQL, Aurora PostgreSQL, MySQL, PostgreSQL
IBM DB2 LUW	Aurora MySQL, Aurora PostgreSQL, MariaDB, MySQL, PostgreSQL
IBM Db2 for z/OS	Aurora MySQL, Aurora PostgreSQL, MySQL, PostgreSQL
Apache Cassandra	Amazon DynamoDB
SAP ASE	Aurora MySQL, Aurora PostgreSQL, MariaDB, MySQL, PostgreSQL
Amazon Redshift	Amazon Redshift
Azure Synapse Analytics	Amazon Redshift
Snowflake	Amazon Redshift
BigQuery	Amazon Redshift

Se debe descargar e instalar en Windows (exe), Fedora (rpm) y Ubuntu (deb).

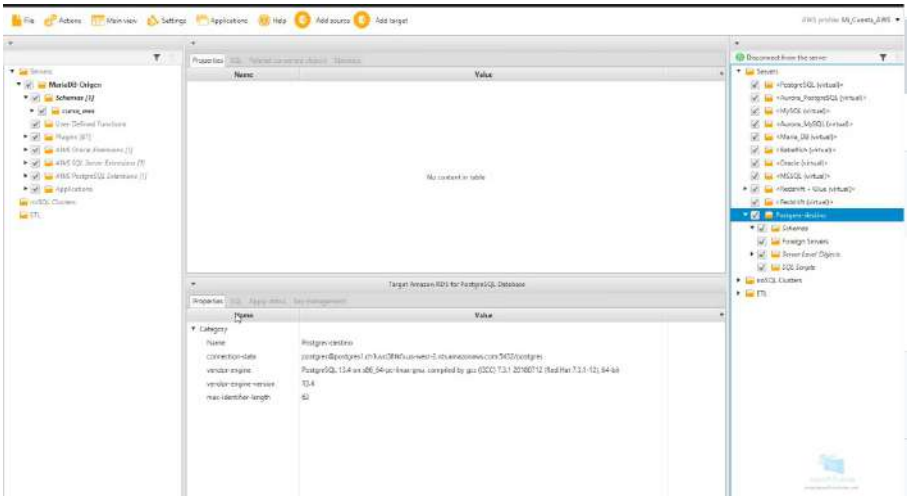
Dentro de la aplicación, antes de hacer nada tenemos que:

- Global Setting/AWS service profile/Añadir uno nuevo con las credenciales correctas. Se convierte en el perfil por defecto.
- Global Settings/Drivers/Descargar los drivers que se necesiten para el mapeo. Por ejemplo, el de mysql es el connecto/J, el de postgres se llama igual… Los dos son .jar.

Podremos comprobar en Data Migration view que podremos comprobar los proyectos de migración que tengamos en la cuenta conectada.

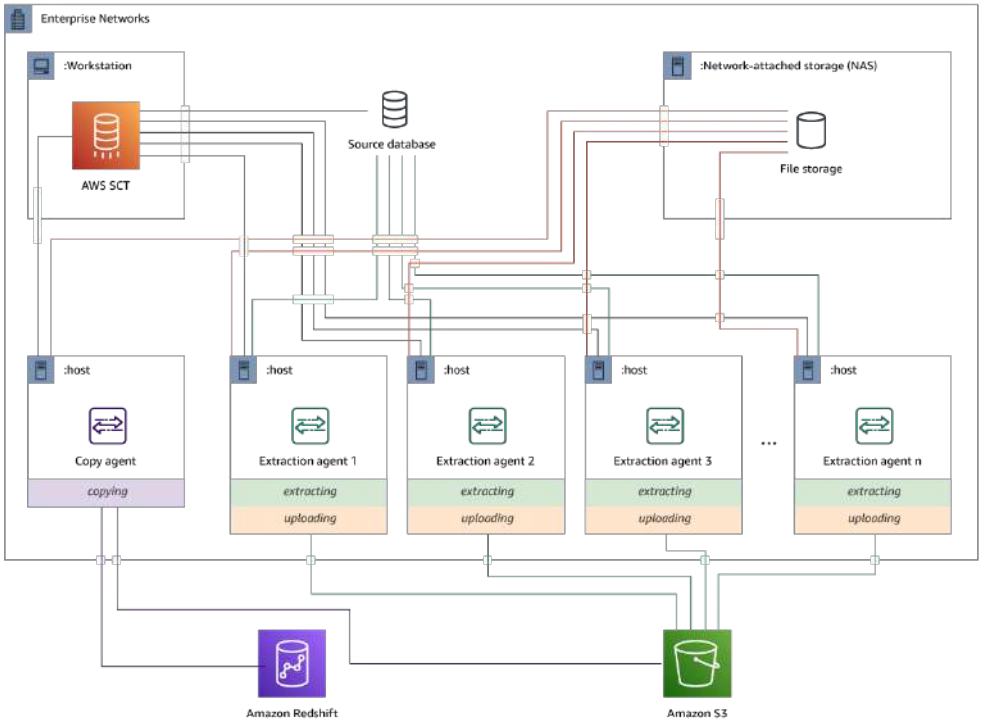
En el **mapping view se crea el mapeo**.

- Se añade la fuente con todos los datos y se efectua un test connection
- Se añade el destino con todos los datos y se efectua un test connection
- Ahora ya se puede crear el mapping
  - Nueva regla de migración y sus datos.
  - Nueva regla de transformación y sus datos.
- Después podemos revisar en main view los schemas de las BBDD



- Después creamos la tarea de migración

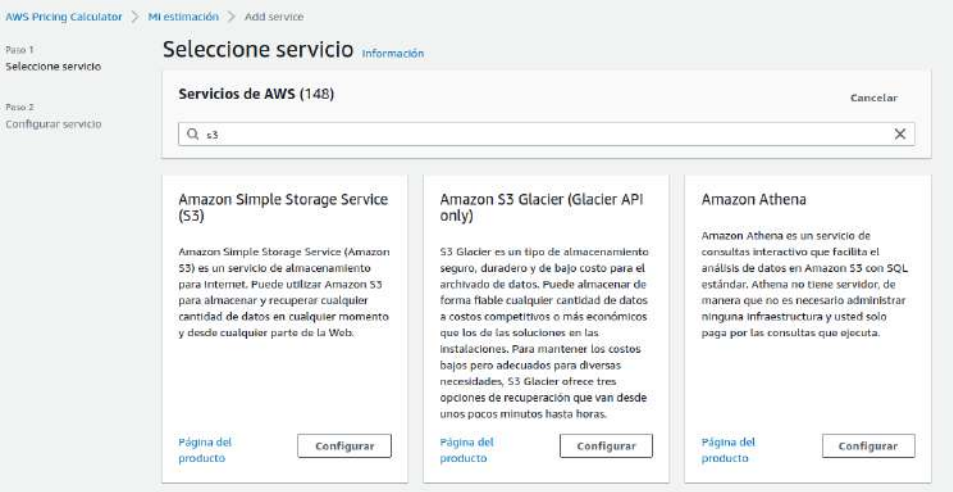
Ejemplo de mapeo:



TEMA 19 - Calculadora de Precios

Con esta herramientas se pueden hacer estimaciones. Si se tiene claro los productos que se necesitan y el tiempo de uso, está herramienta es muy certera: https://calculator.aws/#/

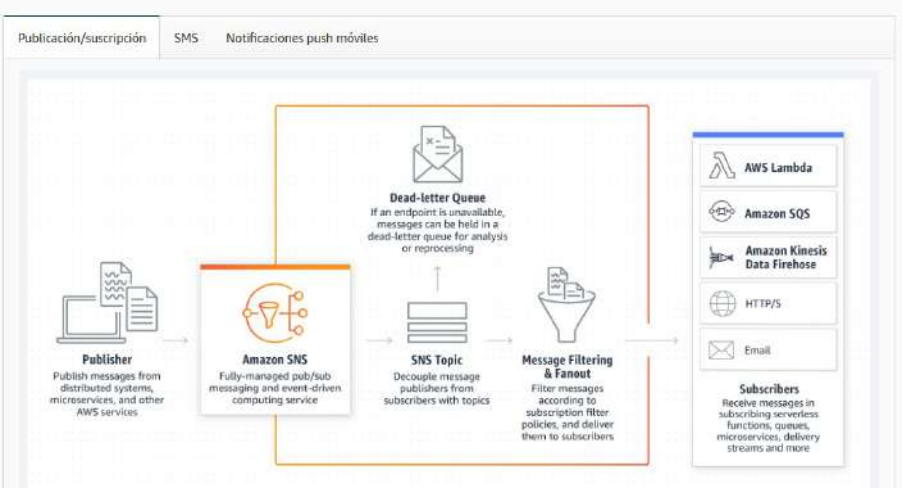
Se crea una estimación, se añaden servicios por tipo configurando al detalle los recursos.

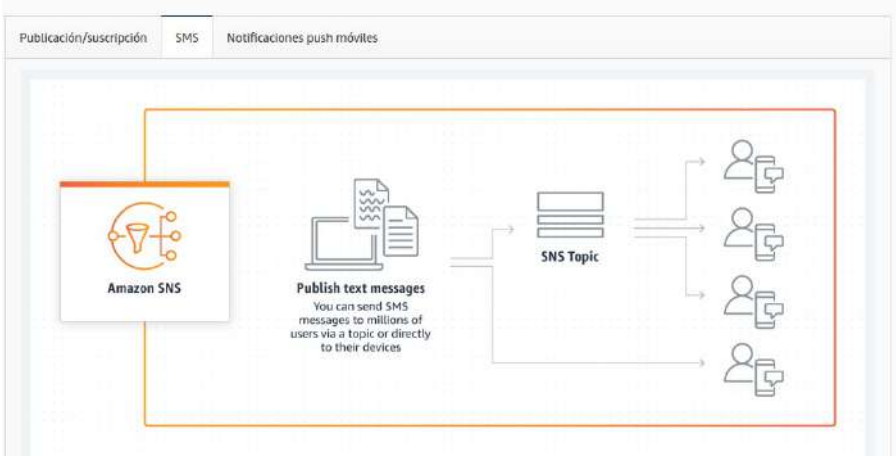


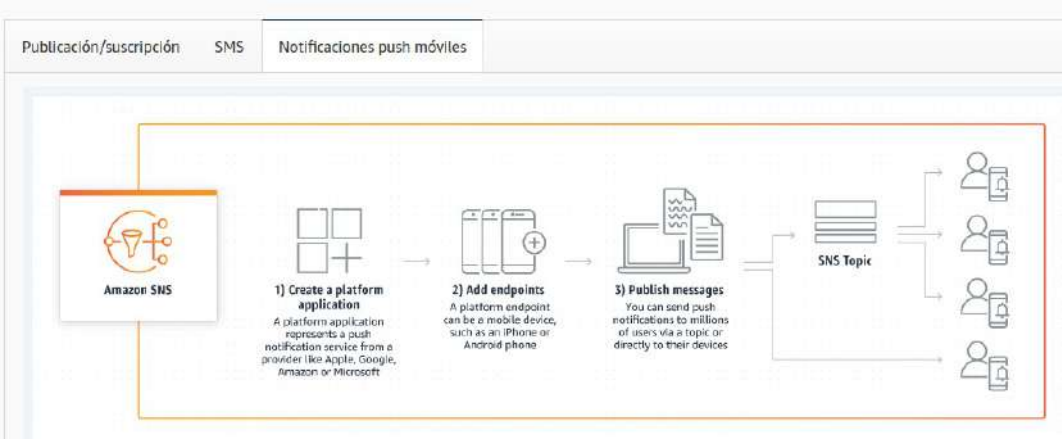
TEMA 20 - SNS Simple Notification Service

Es un servicio de notificaciones push móviles, correo electrónico, SMS y mensajería de suscripción que nos puede servir para cualquier notificación de AWS.

AWS SNS se ocupa de guardar los mensajes en un SNS Topic. Con lo cuál, las aplicaciones conectadas lo que hacen es consultar SNS Topic como si fuese un buzón







Con este servicio se pueden coger mensajes de cualquier servicio y redirigirlo a cualquier otro.

Los Topics sirven para conectar dos infraestructuras diferentes, es un servidor de mensajes como IBM-MQ de IBM y JMS de Java. Lo que hacen es preparar un mensaje origen para que lo pueda interpretar en destino.

A nivel de servidores de mensajes tenemos varios tipos:

- Point to point – Los que tienen un mensaje específico para un destinatorio concreto.
- Topic – Son las suscripciones. Son mensajes que se recopilan para leer por 1 a n destinos.

Otro servicio de AWS es SQS que es un servidor de mensajes que tiene Point to point y Topics.

20.1. - Crear un servicio SNS

Lo primero es **crear un Topic**. Los parámetros configurables son:

- Tipo
  - FIFO Primero en llegar es el primero en salir
  - Standard
- Nombre
- Descripción
- Se puede definir una Política de protección de datos.
- Encriptación
- Definir la política de Acceso
- La política de devolución de mensajes
- Logging

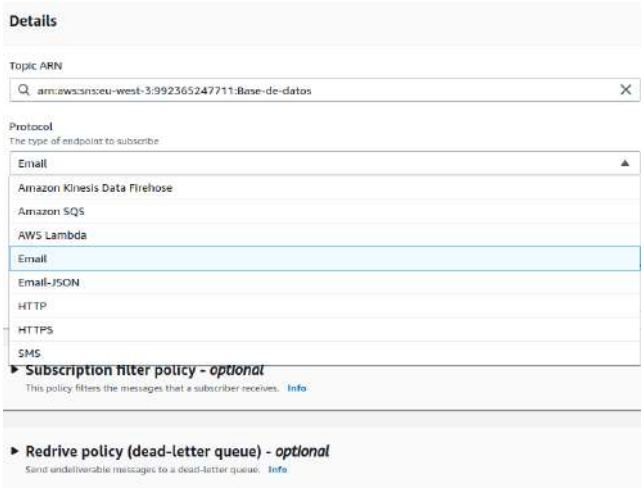
- Etiquetas

Se puede hacer una prueba de mensajes, indicando:

- El mensaje
- TTL
- body - la estructura del mensaje. Se puede escoger una diferente según el destino
- Los atributos – estarán asociados al mensaje. Será el destino quien tenga que descifrarlos.

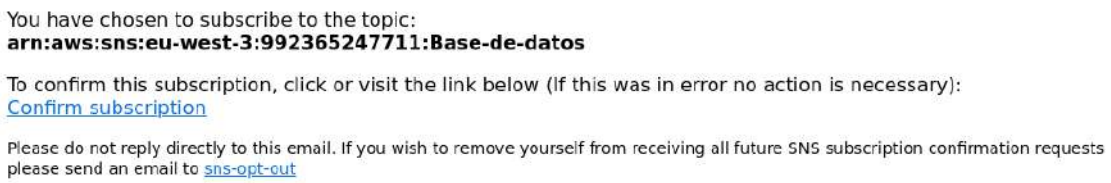
El siguiente paso es **crear una suscripción**. Que es indicar el tipo de destino a donde queremos enviar los mensajes. Tendremos que indicar:

- El topic que queremos utilizar
- El protocolo que queremos utilizar.

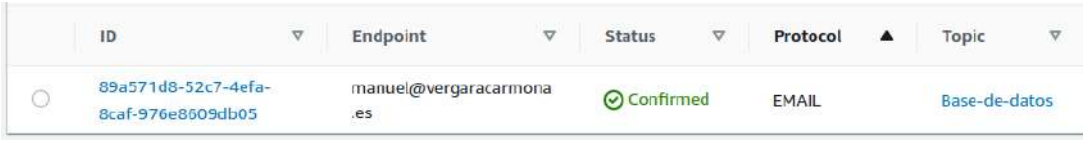


- Indicaremos el endpoint, por ejemplo, en email será la cuenta de correo eletróncio en concreto.
- Se pueden filtrar suscripciones. Por ejemplo, algún tipo como error.
- dead-letter-queue – Es la cola de mensajes muertos, que no se han podido crear. Puede venir bien para manipular esos mensajes en un futuro.

Una vez se crea la suscripción, el mail destino tendrá que confirmar que quiere esta suscripción.



Deberemos confirmar en el mensaje y después podremos ver en suscripción que ya está confirmado.



ID	Endpoint	Status	Protocol	Topic
89a571d8-52c7-4efa-8caf-976e8609db05	manuel@vergaracarmona.es	Confirmed	EMAIL	Base-de-datos

Ahora ya tenemos el topic y el destino de la suscripción, deberemos **crear el origen**.

En RDS hay una opción en el sidebar que son los eventos. Vamos a «Event subscirptions» y creamos uno nuevo. Podemos escoger los siguientes parámetros:

- Nombre
- Destino. Aquí podemos escoger el topic o crear uno nuevo.
- Origen. En el caso de RDS se pueden indicar snapshots, Cluster snapshots, Instances, Parameter groups, Security Groups, Clusters y RDS proxies.
    - Se podrá especificar todos los objetos, varios o uno en concreto.
    - Se podrá especificar todos los eventos, varios o uno en concreto.

Una vez creado, tendremos un evento activo que producirá mensajes hacia el topic, y este, los remitirá al suscriptor.

En S3 podemos ir a las propiedades de un bucket podremos ver “AWS CloudTrail data events”, que es un servicio parecido, y “Events notifications” donde podremos enviar una notificación a SNS con los eventos. Ahora bien, en el caso de S3 debemos dar permisos correctos porque el usuario es el de S3:

- Seleccionamos la ARN del topic SNS.
- Seleccionamos el ARN del bucket.
- Volvemos a SNS y creamos una política de seguridad en el topic que queramos usar:

```
{
  "Version": "2012-10-17",
  "Id": "example-ID",
  "Statement": [
    {
      "Sid": "Ejemplo SNS",
      "Effect": "Allow",
      "Principal": {
        "Service": "s3.amazonaws.com"
      },
      "Action": [
        "SNS:Publish"
      ],
      "Resource": "SNS-topic-ARN",
```

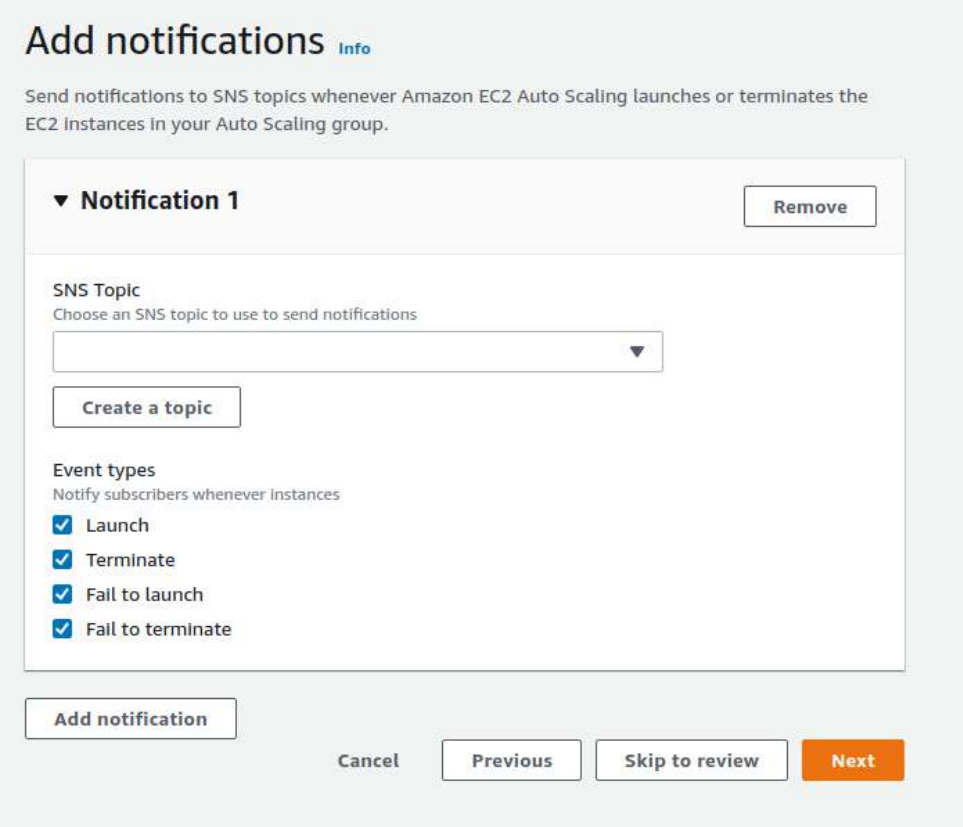
```
        "Condition": {
          "ArnLike": {
            "aws:SourceArn": "arn:aws:s3:*:*:bucket-name"
          },
          "StringEquals": {
            "aws:SourceAccount": "bucket-owner-account-id"
          }
        }
      }
   ]
}
```

Las políticas son identicas a las de S3. En este ejemplo habilitamos que el servicio S3 públique en SNS utilizando el topic SNS desde el origen del bucket concreto y con el usuario indicado.

Ahora ya podemos volver a S3 y en las propiedades del bucket “Creamos un event notification” con los siguientes parámetros:

- Nombre
- Prefijo y/o Sufijo – para filtrar la ubicación y los objetos en sí.
- Tipos de eventos a notificar. Se pueden especificar en concreto o seleccionar todos.
- Destino. Podemos escoger entre varios. En todos ellos tendremos que escogerlos de una lista o especificar el ARN.
    - Función Lambda – Es una función que se ejecuta en un entorno Severless.
    - SNS Topic
    - SQS queue

En EC2, en concreto en un grupo de autoescalada. Cuando se crea un grupo de autoescalada, en el paso de “Add notifications” donde elegimos el SNS Topic y los tipos de eventos que notifica.



Tal y como se vayan produciendo los eventos seleccionados los notificará al topic.

20.2. - Ejemplo CLI RDS

Documentación: https://awscli.amazonaws.com/v2/documentation/api/latest/reference/sns/index.html

aws sns

Subcomandos:

add-permission
check-if-phone-number-is-opted-out
confirm-subscription
create-platform-application
create-platform-endpoint
create-sms-sandbox-phone-number
create-topic
delete-endpoint
delete-platform-application
delete-sms-sandbox-phone-number
delete-topic
get-data-protection-policy
get-endpoint-attributes
get-platform-application-attributes
get-sms-attributes
get-sms-sandbox-account-status
get-subscription-attributes
get-topic-attributes
list-endpoints-by-platform-application
list-origination-numbers
list-phone-numbers-opted-out
list-platform-applications
list-sms-sandbox-phone-numbers
list-subscriptions
list-subscriptions-by-topic
list-tags-for-resource
list-topics
opt-in-phone-number
publish
publish-batch
put-data-protection-policy
remove-permission
set-endpoint-attributes
set-platform-application-attributes
set-sms-attributes

set-subscription-attributes
set-topic-attributes
subscribe
tag-resource
unsubscribe
untag-resource
verify-sms-sandbox-phone-number

Ver los topics

aws sns list-topics

Ver las suscripciones

aws sns list-subscriptions

Crear un topic

aws sns create-topic --name Nombre-del-topic

Asociar suscripción con un topic

aws sns subscribe --topic-arn ARN-del-topic --protocol [http | https | email | email-json | sms | sqs | application | lambda | firehose] --notification-endpoint Correo-electronico-o-telefono-o…

Desasociar suscripción de un topic

aws sns unsubscribe -subscription-arn ARN-de-suscripción

Borrar Topic

aws sns delete-topic --topic-arn Nombre-ARN-del-topic

Borrar endpoint

aws sns delete-endpoint

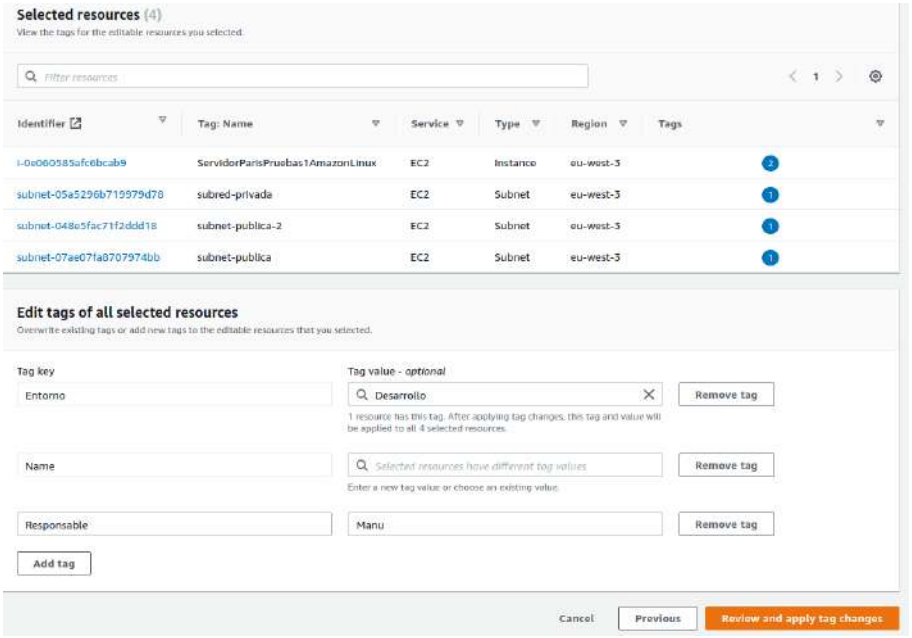
TEMA 21 - Grupo de Recursos y editor de Tags

Una buena política de etiquetas nos puede servir para poder tener categorizados con una jerarquía los objetos desplegados. De esta manera, desde la monitorización en CloudWatch o desde el control de costes en Billing, tendremos facilidad para informes, para aplicar métricas, para gestión y control de multitud de aspectos. Es IMPORTANTE la decisión de una política de etiquetado en la infraestructura entre los miembros de la organización.

Con “Resource Groups & Tag Editor” podemos tener un control de la gestión de las etiquetas. Además, podemos creas plantillas de infraestructura con CloudFormation.

En el **Editor de Tags** podemos etiquetar masivamente recursos concretos de forma que lo podamos agrupar en un grupo de recursos. Los parámetros que podemos usar para filtrar son por Regiones, por tipo de recursos y/o por etiquetas ya existentes.

Una vez le damos a buscar recursos nos aparece un listado desde donde podemos seleccionar para etiquetar con el key:valor que queramos clicando en “Manager tags”.



En “Tags Policies” es una opción que nos permite crear una política de etiquetado para añadirlas en los nuevos objetos.

En **crear un grupo de recursos** podemos basarlo en Tags o en CloudFormation stack. Basados en Tags tenemos las siguiente configuración:

- Se puede agrupar con unos criterios concretos, por ejemplo, buscando recursos directamente en un listado o añadiendo directamente las etiquetas key:valor. Hacemos una preview de los resultados de filtrado.
- Se puede agrupar otros grupos de recursos
- Nombre y descripción del grupo

- Etiquetas de grupo (Opcional)

Una vez se cree se pueden ver en “Saved Resource Groups”.

TEMA 22 - CloudWatch. DashBoards y Métrics

Es el servicio que ofrece AWS para monitorizar y controlar logs. Agrupa un conjunto de funcionalidades para métricas, logs, alarmas, etc.

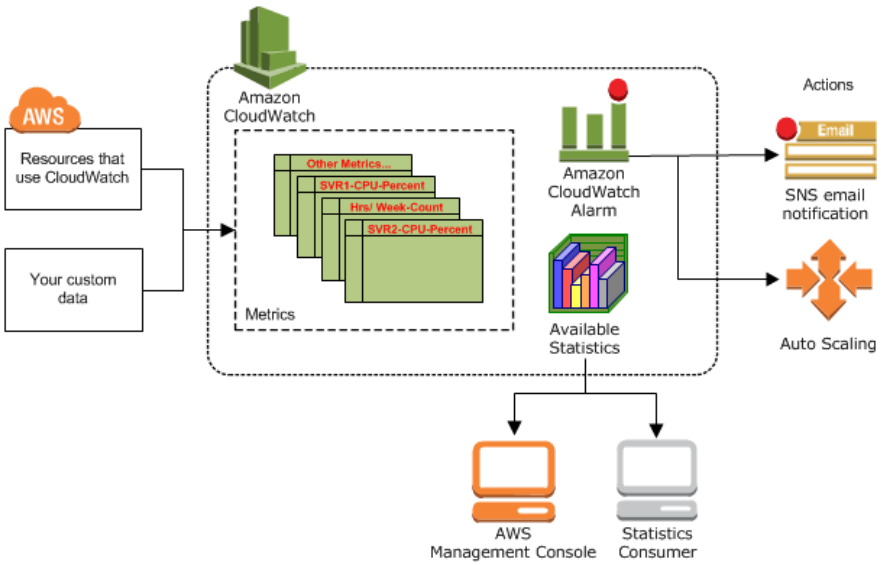
Existen servicios parecidos para este control, como “AWS EventBridge”.

CloudWatch recolecta información para la monitorización. A través de los eventos puede proceder a acciones concretas destinadas a ayudar al análisis de la información para encontrar patrones, detectar posibles probleas o incluso incluirlos en machine Learning.

Este ciclo permite llevarlas a aplicaciones de monitorización, tendremos una visión amplia del sistema, podremos optimizar los recursos y unificar las operaciones de mantenimiento.

22.1. - Conceptos

Arquitectura del producto:



Es el lugar donde se almacenan las métricas de los recursos configurados.

AWS envía sus métricas y podemos configurarlas. Con las alarmas de CloudWatch se pueden hacer dos acciones: Notificación SNS o Auto escalar.

Las estadísticas que va guardando CloudWatch se pueden explotar desde la gestión de AWS o por terceros.

Unos **conceptos básicos** son:

- **namespaces** – Un contenedor para métricas de CloudWatch. Las métricas en distintos espacios de nombres están aisladas entre sí, de forma que las métricas de distintas aplicaciones no estén acumuladas por error en las mismas estadísticas. Los espacios de nombres de AWS utilizan la nomenclatura: AWS/service. Ejemplos: AWS/S3, AWS/EC2, etc
- **Métricas** – Representan una serie de DataPoints ordenados temporalmente que se publican en CloudWatch. Una métrica es una variable que hay que monitorizar y los **DataPoints** son los valores de esa variable a lo largo del tiempo. Por ejemplo, el uso de la CPU de una determinada instancia EC2 es una métrica que Amazon EC2 proporciona y se almacena como un DataPoint.
- AWS dispone de un **conjunto de métricas** para sus servicios de forma gratuita:
  - Monitorización básica: frecuencia de 5 minutos.
  - Monitorización detallada: frecuencia de 1 minuto. Se cobra a parte.
  - Métricas personalizadas: pueden llegar a una frecuencia de 1 segundo.

  Las métricas son específicas de una región.
- **Retención de métricas**:
  - Los DataPoints con un período de menos de 60 segundos están disponibles durante 3 horas. Estos DataPoints son métricas personalizadas de alta resolución.
  - Los DataPoints con un período de 60 segundos están disponibles durante 15 días.
  - Los DataPoints con un período de 300 segundos (5 min.) están disponibles durante 63 días.
  - Los DataPoints con un período de 3600 segundos (1 hora) están disponibles durante 455 días. (15 meses).
  - Despúes de los 15 meses si no se usan se pierden.

  Los DataPoints con un periodo más corto se acumulan para almacenarlos a largo plazo. Por ejemplo, si recopila datos con un período de 1 minuto, los datos están disponibles durante 15 días con una resolución de 1 minuto. Después de 15 días estos datos siguen estando disponibles, pero se acumulan y solo se pueden recuperar con una resolución de 5 minutos. Después de 63 días, los datos siguen acumulándose y están disponibles con una resolución de 1 hora.

- **Time stamp** – Cada DataPoint debe asociarse a una marca temporal o timestamp. La marca temporal puede ser de hasta dos semanas en el pasado y de hasta dos horas en el futuro. CloudWatch crea una en función de la hora a la que se recibió el punto de datos.
- **Dimensiones** – Es un par de nombre-valor que forma parte de la identidad de una métrica... Cada métrica tiene características específicas que la describen y puede considerar las dimensiones como categorías para las características.
- **Units** – Cada estadística tiene una unidad de medida. Entre las unidades de ejemplo se incluyen Bytes, Seconds, Count y Percent.
- **Aggregation** – Amazon CloudWatch acumula estadístitcas de acuerdo con la duración del periodo que especifique al recuperar las estadísticas.
- **Percentiles** – Indica el peso relativo de un valor en un conjunto de datos. Por ejemplo, el percentil 95 significa que el 95% de los datos está por debajo de este valor y el 5 por ciento de los datos está por encima del mismo. Los percentiles ayudan a entender mejor la distribución de los datos de métricas. Los percentiles se suelen utilizar para aislar anomalías.
- **Alarms** – Vigila una única métrica durante el período especificado y realiza una o varias acciones especificadas según el valor de la métrica relativo a un determinados umbral durante un período de tiempo. La acción es una notificación que se envía a un tema de Amazon SNS o a una política de Auto Scaling.

22.2. - Consola de CloudWatch

- **Dashboards** – Son los cuadros de mando desde donde monitoriza. Se pueden personalizar con widgets o se pueden crear automáticos.
- **Alarms** – Gestión de las alarmas
- **Logs** – Se pueden agrupar o se pueden usar queries para buscar concretos con Logs Insights.
- **Metrics** – Es el componente básico donde se va recopilando la información para poder utilizarla en la monitorización y el rendimiento.
- **x-ray traces** - Ayuda a analizar y depurar las aplicaciones distribuidas, como las construidas con una arquitectura de microservicios. Se puede comprender el rendimiento de aplicaciones y servicios subyacentes, e identificar y solucionar la causa raíz de los problemas de rendimiento y los errores.
- **Events** – Los eventos se gestionan desde EventBridge. Antes estaba integrado aquí pero se separo el servicio.
- **Application monitoring** – Se puede construir aplicaciones para que envíen información a CloudWatch.
- **Insights** – Son descubrimientos que se hacen en algún entorno. Se pueden aplicar en Contenedores, Lambda, con colaboradores y en aplicaciones.

En las **Settings** se pueden configurar algunos comportamientos del producto.

22.2.1. - Dashboards

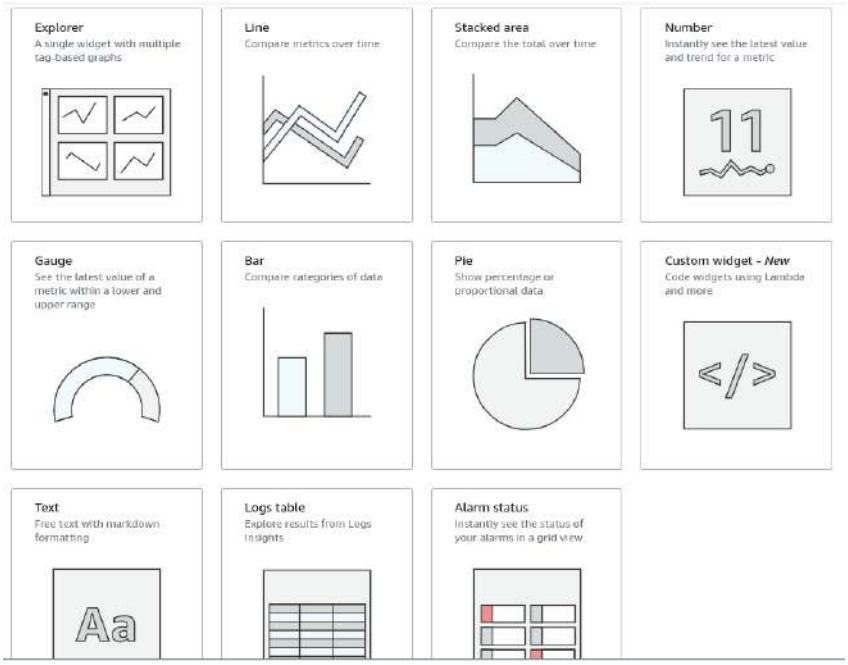
3 son gratuitos, con 50 widgets por panel. A partir del cuarto son 3 dolares mensuales, aunque solo se cree y se borre.

Los elementos que se pueden añadir en los dashboard se le llaman widgets.

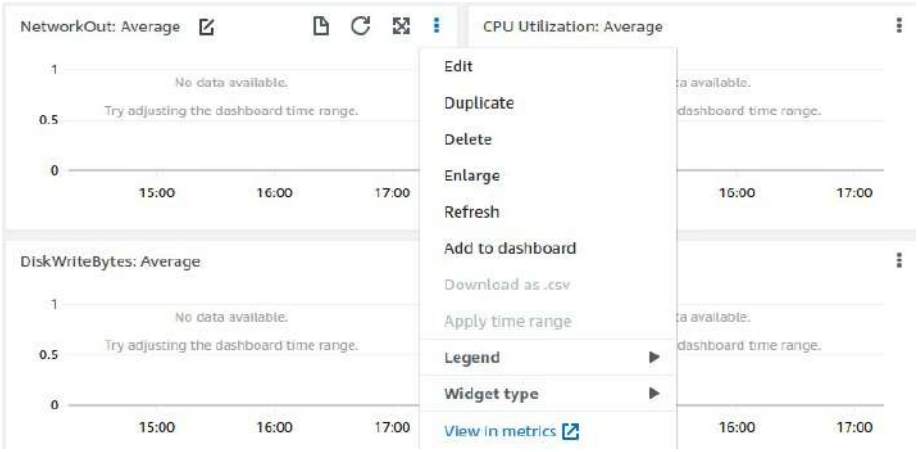
Cuando se crea uno nos pide el nombre y luego podemos añadir widgets de forma manual o de manera automática.

22.2.2. - Widgets y Métricas

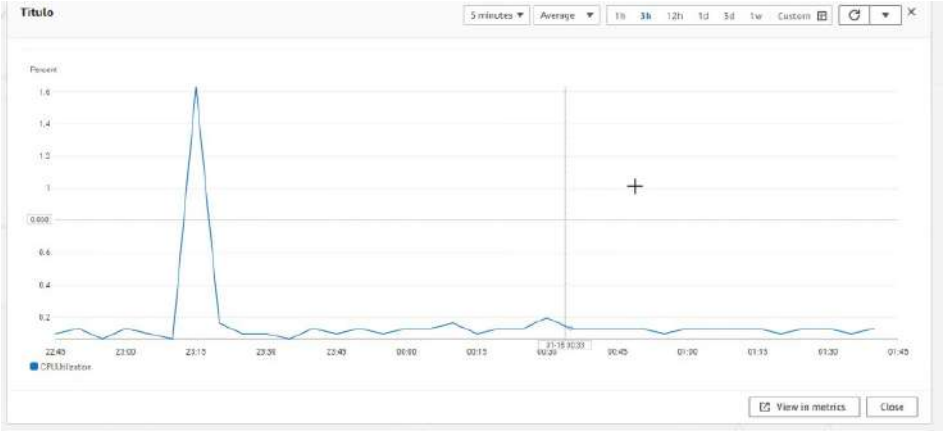
Los tipos de widgets son una forma de mostrar las métricas, podemos escoger entre distintos tipos.



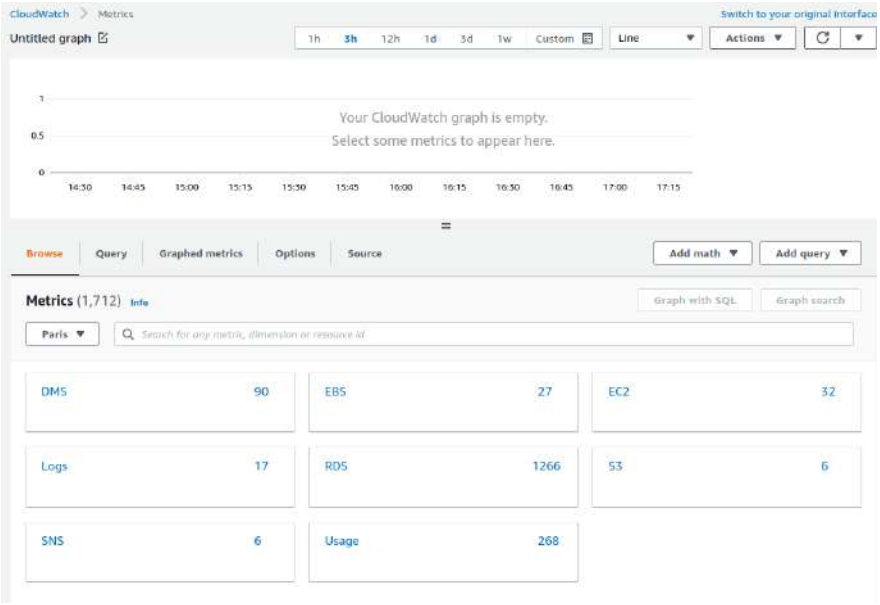
Podemos escoger la ubicación y el orden de los widgets en la consola. Tenemos opciones para cada uno de los widgets.



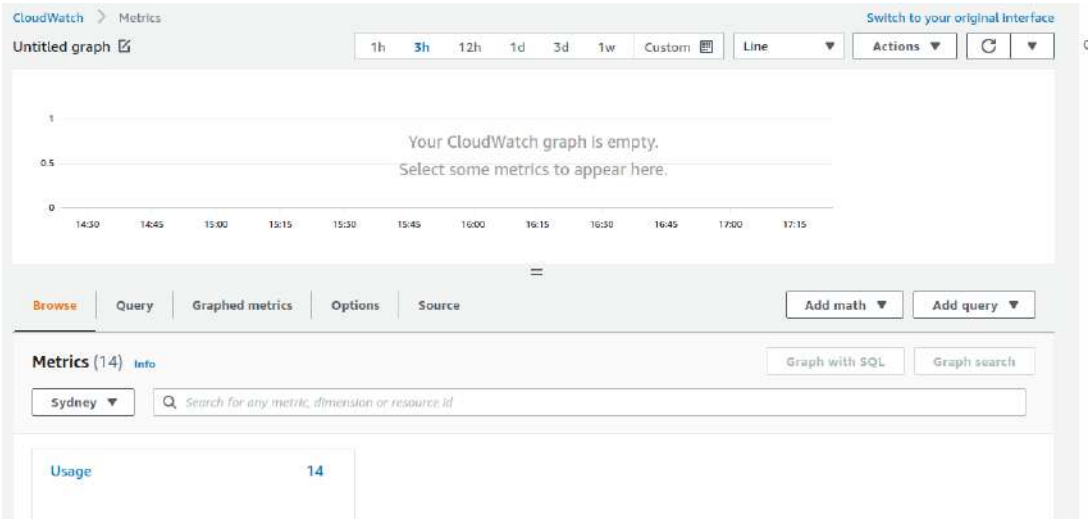
Podemos ampliarlo a pantalla completa para ver con más detalles y variar los parámetros de la gráfica.



Lo más interesante es configurar nuestras propias métricas para que sea lo más personalizado posible a nuestras necesidades. Dentro de las metrics aparecen solo las de los productos que hayamos utilizado.



Por ejemplo, si cambiamos a una region que no hayamos usado a penas aparecerá ninguna.



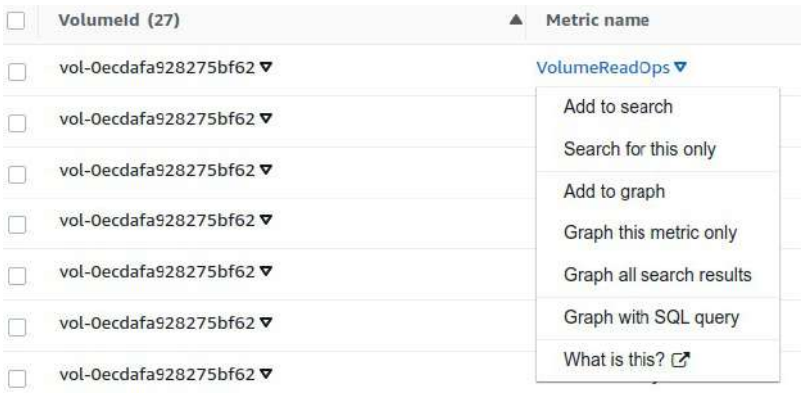
En este panel podemos escoger métricas para crear nuestro widget personalizado. Incluso podemos incluir las métricas de otras regiones.

Podemos filtrar por búsquedas que pueden incluir métricas de distintas clases. Por ejemplo, si buscamos EBS nos saldrán las métricas que tengan que ver con este servicio aunque no estén dentro de la clase EBS. Podrían estar en RDS o EC2

Si añadimos varias búsquedas irá filtrando

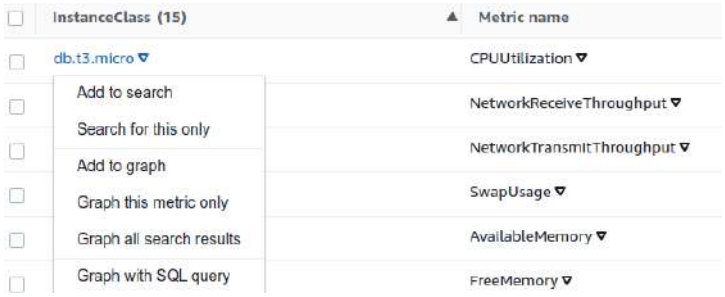
Cada métrica va a estar asociados a una ID de la clase de elemento y a un nombre de métrica.

Para entender que es cada nombre de métrica tendremos que acudir a la documentación. Podemos hacerlo directamente en el menú contextual de cada nombre. En «What is this?».



Añadir la búsqueda, buscar solo este, añadir al gráfico, gráfico de esta métrica solo, gráfico de toda la cerca del resultado, Gráfico con query SQL y What is this?.

Cada elemento también tiene una serie de opciones.



Añadir la búsqueda, buscar solo este, añadir al gráfico, gráfico de esta métrica solo, gráfico de toda la cerca del resultado y Gráfico con query SQL.

En algunos casos podemos añadir todos los elementos directamente

En un gráfico se pueden añadir varias métricas. Cuando las tenemos seleccionadas, en la pestaña «Graphed metrics», podemos cambiar el nombre de la métrica para la gráfica.

Se podrán seleccionar algunas acciones que queremos que recoja en la métrica. Por ejemplo, el color, statistic podemos seleccionar qué recoge, podemos decir el periodo, etc

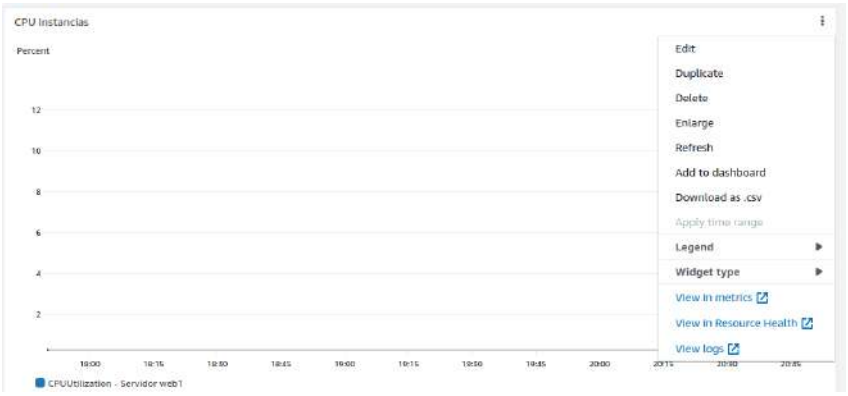
En Options podemos parametrizar algunos aspectos de cada tipo de gráfica.

En Source nos genera un fichero JSON que podría utilizar, por ejemplo, desde AWS CLI o AWS SDK.

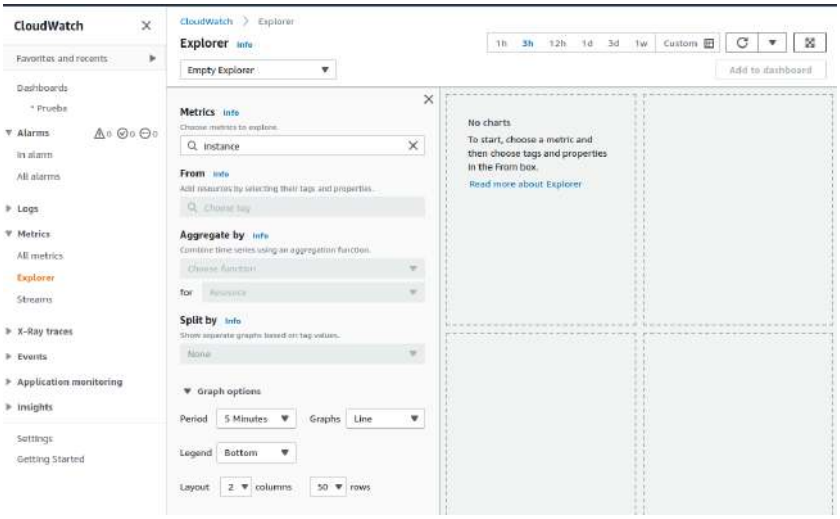
En la parte de arriba podemos cambiarle el nombre al gráfico.

En «Actions» podemos añadirlo a un dashboard, compartir o descargar en .csv.

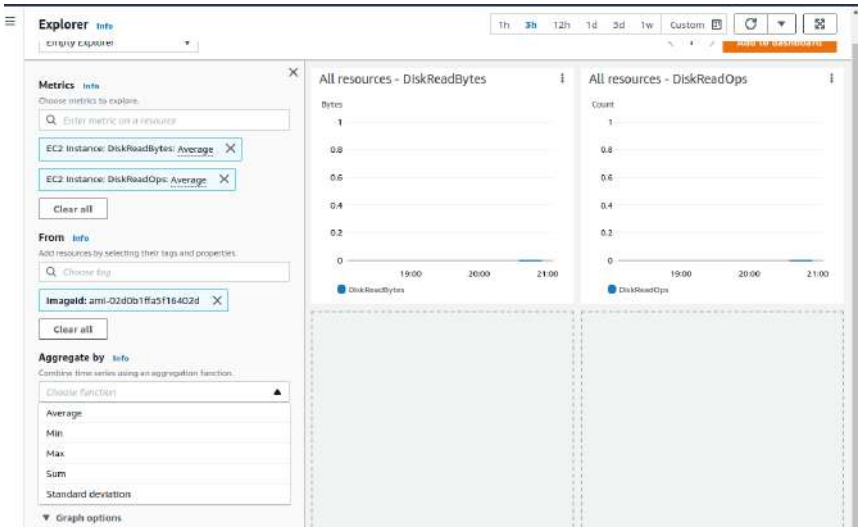
Una vez lo pasemos al dashboard tendremos también algunas acciones como editarlo, duplicarlo, borrarlo, etc



Dentro de las métricas, también tenemos el explorer. Sirve para agrupar métricas, dividirlas, etc Se pueden seleccionar algunas plantillas para hacerlo rápido y en masa por servicio.



Se selecciona el tipo de métrico y desde donde recoge los datos (el recurso o tags). Tal y como vayamos parámetros va generando los gráficos. Luego se le puede indicar la agregación. Con split se pueden separar los gráficos por valores.



Por último se pueden configurar opciones y agregar al dashboard

22.2.3. - Agente CloudWatch

Se pueden configurar para que una instancia mande métricas especiales además de las que se mandan por defecto. Se instala dentro de la instancia y puede mandar métricas y logs (Por ejemplo, de los apache o tomcat o cualquier servicio interno de la instancia.).

En principio, se puede instalar en todas las instancias soportadas por AWS (RedHat, Suse, etc). En **cada SO tendrá un modo de instanción.** Documentación: https://docs.aws.amazon.com/es_es/AmazonCloudWatch/latest/monitoring/install-CloudWatch-Agent-on-EC2-Instance.html

En el **caso de Amazon Linux 2** es con el siguiente comando:

```
sudo yum install amazon-cloudwatch-agent
```

Ahora tendremos que parametrizar el archivo de configuración del agente. Documentación: https://docs.aws.amazon.com/es_es/AmazonCloudWatch/latest/monitoring/CloudWatch-Agent-Configuration-File-Details.html

Los ficheros están en /opt/aws

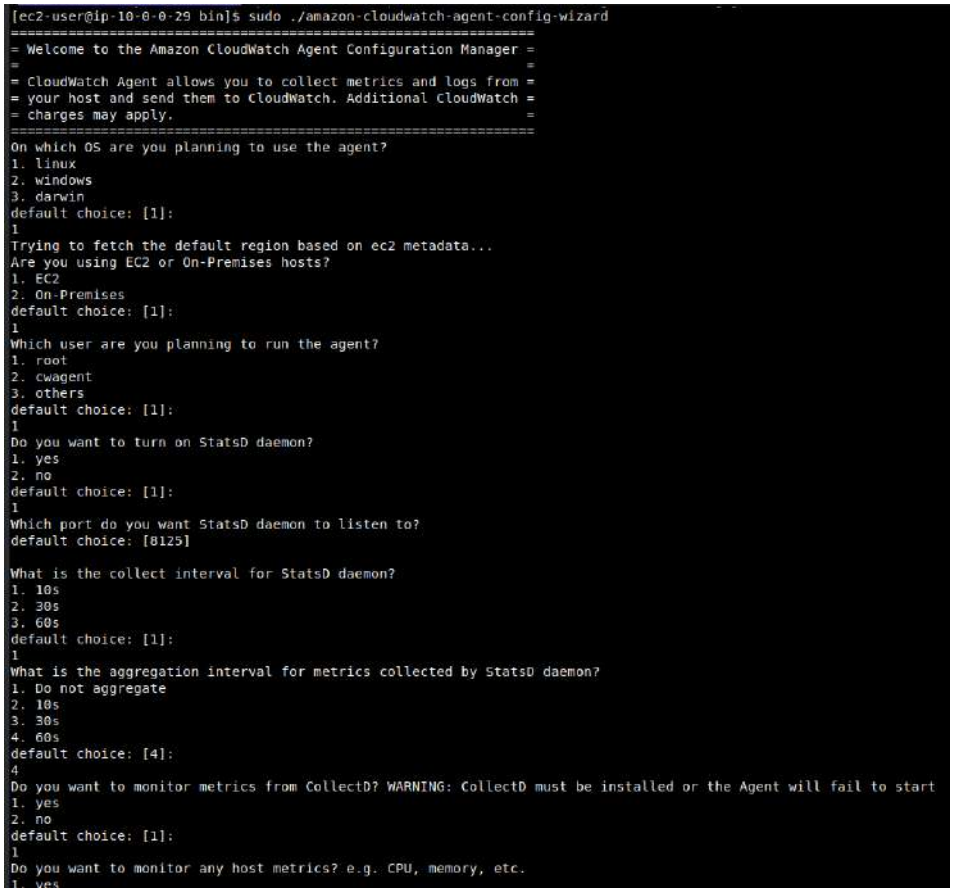
```
[ec2-user@ip-10-0-0-29 ~]$ cd /opt/aws/
[ec2-user@ip-10-0-0-29 aws]$ ls
amazon-cloudwatch-agent  apitools  bin
[ec2-user@ip-10-0-0-29 aws]$ tree amazon-cloudwatch-agent/
amazon-cloudwatch-agent/
├── bin
│   ├── amazon-cloudwatch-agent
│   ├── amazon-cloudwatch-agent-config-wizard
│   ├── amazon-cloudwatch-agent-ctl
│   ├── config-downloader
│   ├── config-translator
│   ├── cwagent-otel-collector
│   ├── CWAGENT_VERSION
│   └── start-amazon-cloudwatch-agent
├── cwagent-otel-collector
│   ├── etc
│   │   └── cwagent-otel-collector.d
│   ├── logs
│   └── var
├── doc
│   └── amazon-cloudwatch-agent-schema.json
├── etc
│   ├── amazon-cloudwatch-agent.d
│   └── common-config.toml
├── LICENSE
├── logs
├── NOTICE
├── RELEASE_NOTES
├── THIRD-PARTY-LICENSES
└── var

11 directories, 14 files
```

En el archivo bin tenemos unos scripts por defecto para lanzar el agente, para configurarlo entre otras cosas.

```
[ec2-user@ip-10-0-0-29 amazon-cloudwatch-agent]$  ll bin/
total 207944
-rwxr-xr-x 1 root root 66556280 Jun  9 19:14 amazon-cloudwatch-agent
-rwxr-xr-x 1 root root  9261336 Jun  9 19:14 amazon-cloudwatch-agent-config-wizard
-rwxr-xr-x 1 root root    19318 Jun  9 19:14 amazon-cloudwatch-agent-ctl
-rwxr-xr-x 1 root root  8255864 Jun  9 19:14 config-downloader
-rwxr-xr-x 1 root root 18741720 Jun  9 19:14 config-translator
-rwxr-xr-x 1 root root 91570232 Jun  9 19:14 cwagent-otel-collector
-rw-r--r-- 1 root root       11 Jun  9 19:14 CWAGENT_VERSION
-rwxr-xr-x 1 root root 18508024 Jun  9 19:14 start-amazon-cloudwatch-agent
```

También tiene un controlador del agente que se llama amazon-cloudwatch-agent-ctl

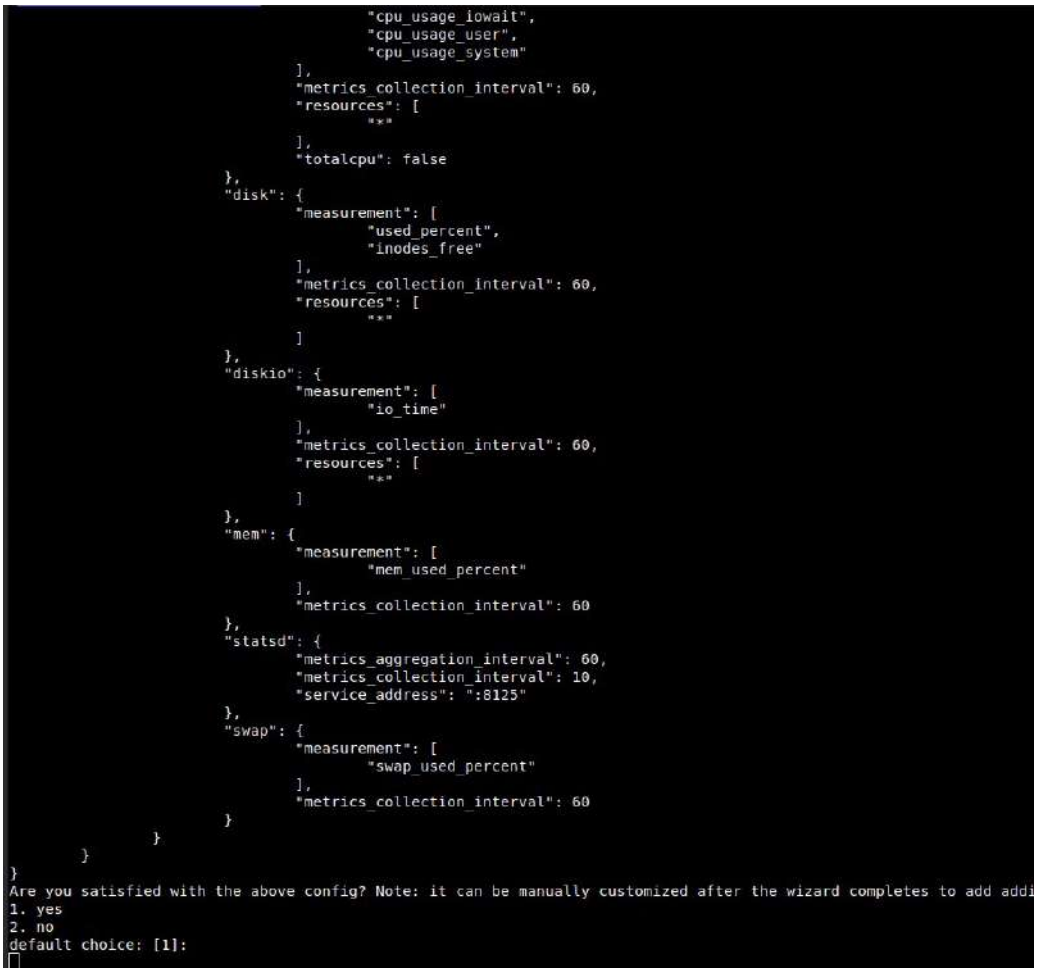
El **fichero de configuración** para definir las características de lo que queremos enviar a CloudWatch es un JSON que se puede generar desde el script amazon-cloudwatch-agent-config-wizard



Nos hará una serie de preguntas básicas:

- Que SO estamos usando
- Donde está alojado. EC2 o On-Premises
- Que usuario será quien gestione el agente.
- Si queremos ejecutar el demonio StatsD
- Nos pregunta el puerto que utilizará el demonio.
- Cada cuanto tiempo recopila información el demonio.
- Cada cuanto tiempo agrega información el demonio.
- Luego pregunta si queremos recolectar información con el demonio CollectD, que monitoriza las métricas.
- Si se quiere monitorizar cualquier tipo de métrica
- Si quiero monitorizar por la cpu por core.
- Si queremos añadir las dimensiones EC2 (ImageId, InstanceId, InstanceType, etc). Nos permite que luego podamos configurar la métrica.
- Si queremos agregar InstanceId

- Si queremos la posibilidad de añadir alta resolución en la recogida, más bajo del minuto.
- Que tipo de métricas por defectos queremos generar (Básica, Standard, Avanzada, ninguna) Cuanto más le añadimos, más grande será el fichero de recolección.
- Nos muestra una previsualización del JSON y nos pregunta si estamos conformes.

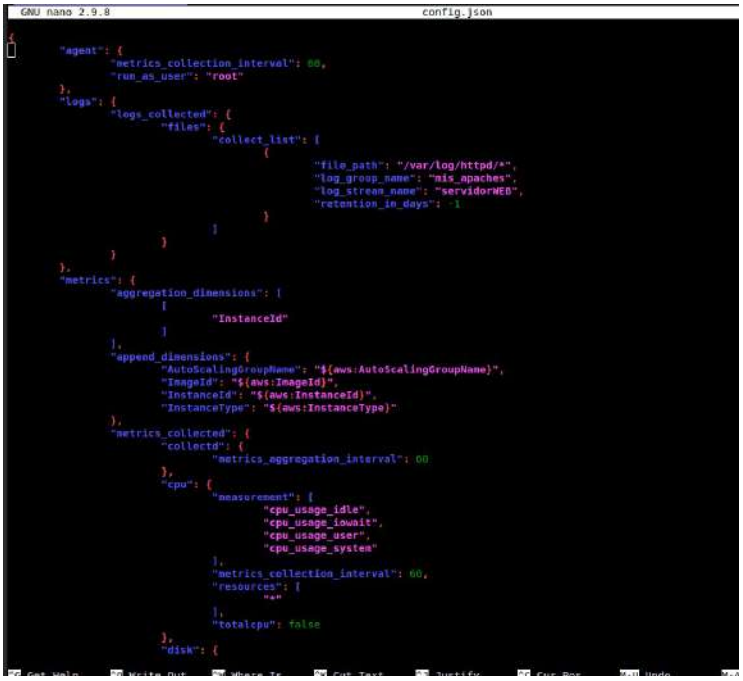


- Nos pregunta si teníamos otro agente instalado previamente.
- Si vamos a monitorizar ficheros de log. Si decimos que sí tenemos que especificar la ubicación de los ficheros logs
- Nos pregunta el nombre del grupo que le vamos a poner.
- El nombre de stream se puede dejar el nombre de la instancia.
- Pregunta por los días de retención
- Pregunta si queremos más ficheros de logs
- Pregunta si queremos guardarlo en un SSM parameter store. Esto requiere una serie de permisos dentro de Amazon. Si decimos que no lo guarda en local.

Ahora ya tenemos el documento generado como config.json. En la documentación se puede ver los parámetros que podemos definir: https://docs.aws.amazon.com/es_es/AmazonCloudWatch/latest/monitoring/metrics-collected-by-CloudWatch-agent.html

La **estructura de config.json** son los siguientes parámetros:

- agent - indica cada cuanto tiempo y el usuario.
- logs
- metrics
  - aggregation_dimensions
  - append_dimensions
  - metrics_collected



Para añadir por ejemplo la memoria libre, la memoria inactiva, la memoria total y la memoria usada, comprobamos como se llama cada componente en la documentación:

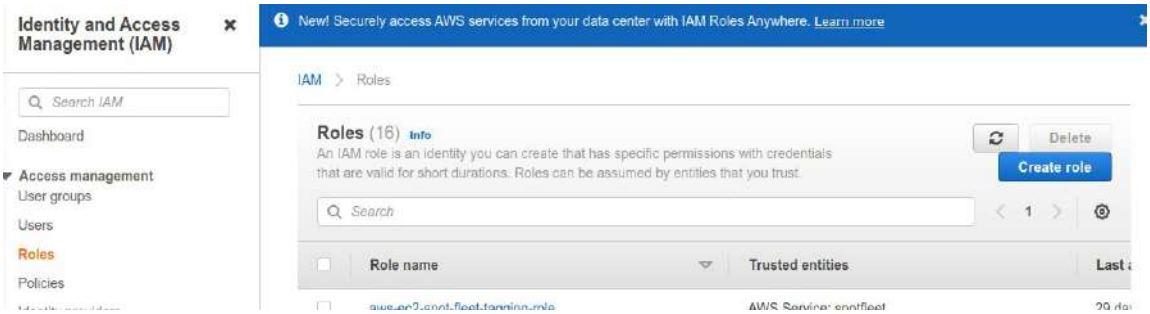
Métrica	Descripción
mem_free	La cantidad de memoria que no se está utilizando. Unidades: bytes
mem_inactive	La cantidad de memoria que no se ha utilizado de alguna manera durante el último período de muestreo. Unidades: bytes
mem_total	La cantidad total de memoria. Unidades: bytes
mem_used	La cantidad de memoria en uso actualmente. Unidades: bytes

Y lo añadimos al espacio de mem de metrics_collected.

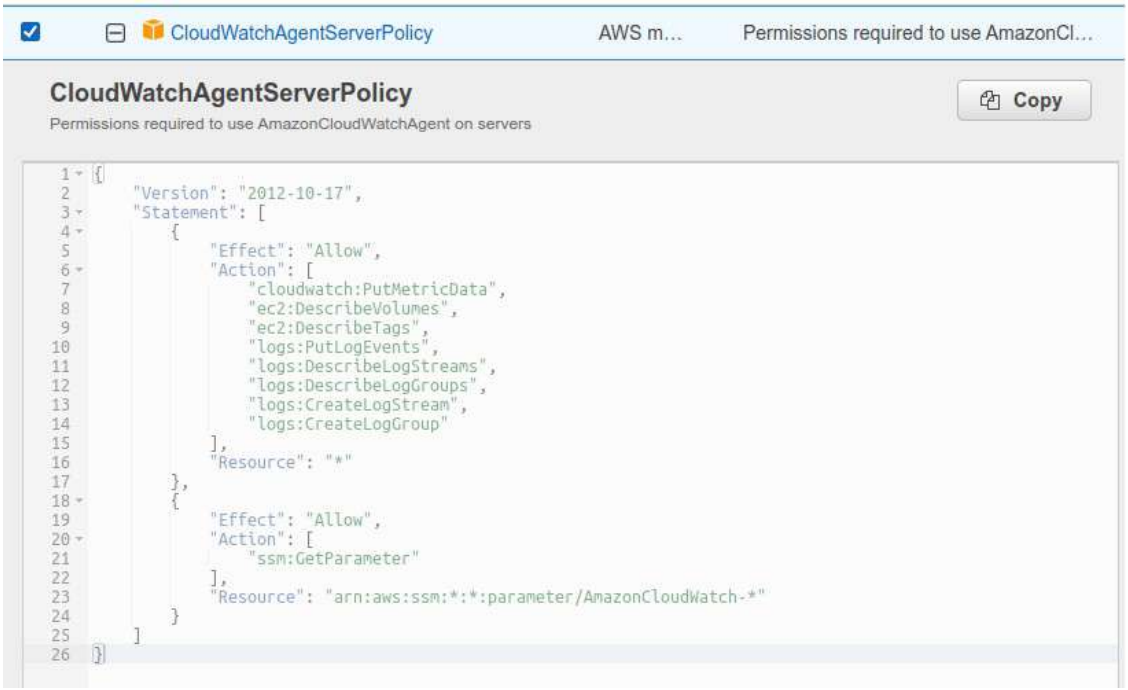
Antes:

Después:

Antes de arrancar el agente **necesitamos un rol (IAM)** para darle los permisos necesarios, que son la de escribir en CloudWatch, y se lo asignaremos a la instancia. Así que buscamos el servicio IAM, el apartado roles y creamos uno nuevo. Esto será un grupo de permisos.

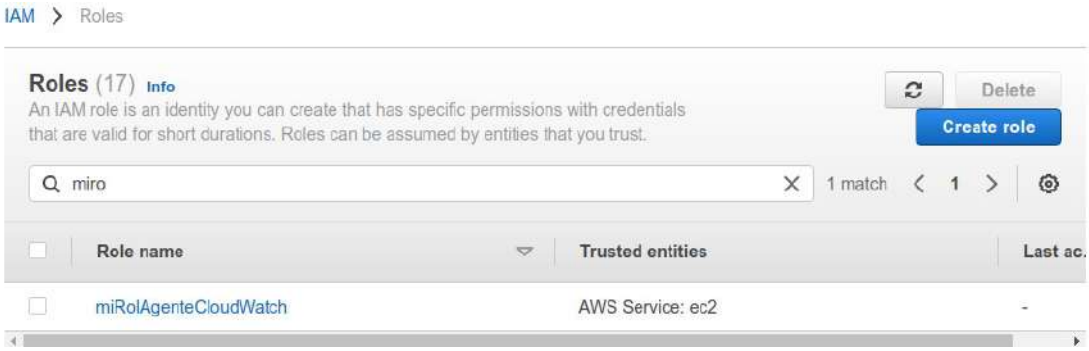


Creamos un AWS service, de tipo EC2 y le damos permisos. En este caso se llaman CloudWatchAgentServerPolicy, son políticas de permisos.

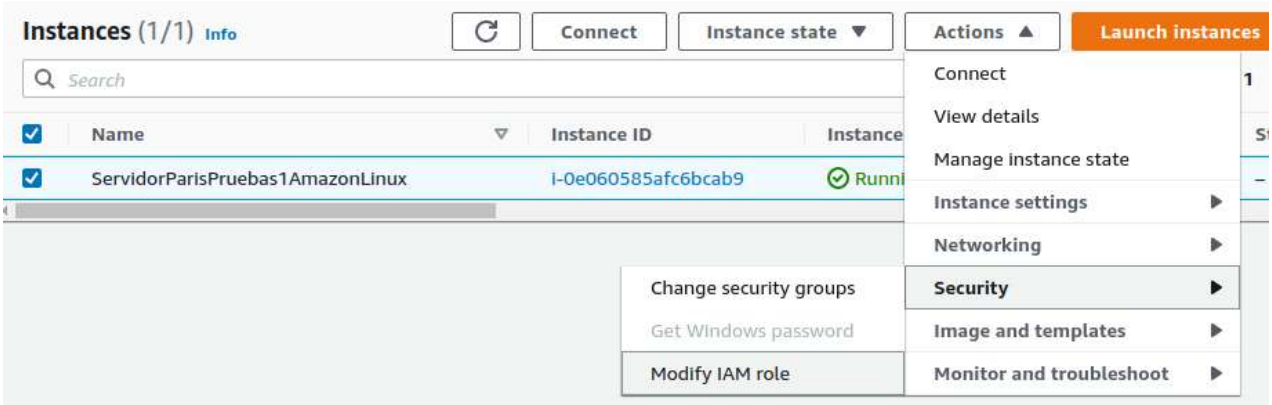


Después se le debe indicar el nombre del rol, la descripción y si queremos tags.

Una vez creado deberemos asignarle a la instancia este rol.



Entonces vamos a la instancia, la seleccionamos, botón «Actions», Security y Modify IAM role.



Seleccionamos el rol y ya lo tenemos. Lo podemos comprobar en la pestaña Security de la instancia.

Ahora tenemos que arrancar el agente. deberíamos ver en la documentación cuál es el mecanismo correcto según nuestro tipo de instancia y de SO: https://docs.aws.amazon.com/es_es/AmazonCloudWatch/latest/monitoring/install-CloudWatch-Agent-commandline-fleet.html Es IMPORTANTE comprobar la documentación porque lo van cambiando.

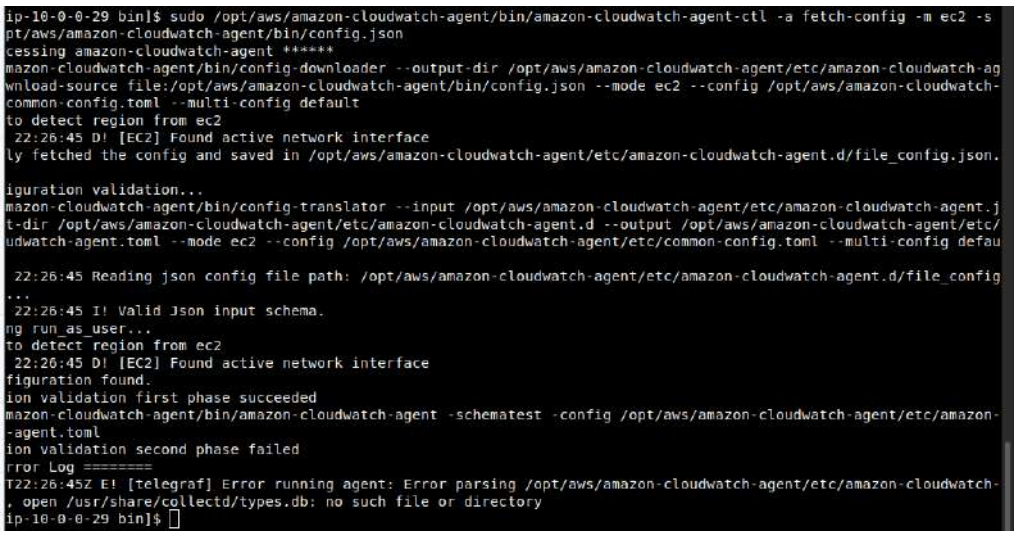
En el caso de EC2 con Amazon Linux 2 es este:

sudo /opt/aws/amazon-cloudwatch-agent/bin/amazon-cloudwatch-agent-ctl -a fetch-config -m ec2 -s -c file:configuration-file-path

Lo lanza el binario controlador del agente amazon-cloudwatch-agent-ctl que tiene la siguiente sintaxis:

amazon-cloudwatch-agent-ctl -a
stop|start|status|fetch-config|append-config|remove-config|set-log-level
[-m ec2|onPremise|auto]
[-c default|all|ssm:<parameter-store-name>|file:<file-path>]
[-o default|all|ssm:<parameter-store-name>|file:<file-path>]
[-s]
[-l INFO|DEBUG|WARN|ERROR|OFF]

Nos da error



No puede abrir /usr/share/collectd/types.db, que significa que no tiene el recolector CollectD, así que lo instalamos.

sudo yum install -y collectd

Si no funciona podemos utilizar el gestor de paquetes amazon-linux-extras.

Lanzamos de nuevo el arranque del agente, esta vez con éxito:

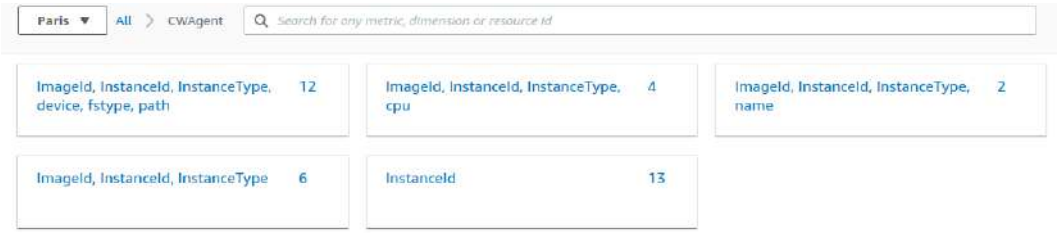
```
[ec2-user@ip-10-0-0-29 bin]$ sudo /opt/aws/amazon-cloudwatch-agent/bin/amazon-cloudwatch-agent-ctl -a fetch-config -m ec2 -s -c file:/opt/aws/
amazon-cloudwatch-agent/bin/config.json
****** processing amazon-cloudwatch-agent ******
/opt/aws/amazon-cloudwatch-agent/bin/config-downloader --output-dir /opt/aws/amazon-cloudwatch-agent/etc/amazon-cloudwatch-agent.d --download-
source file:/opt/aws/amazon-cloudwatch-agent/bin/config.json --mode ec2 --config /opt/aws/amazon-cloudwatch-agent/etc/common-config.toml --mul
ti-config default
I! Trying to detect region from ec2
2022/09/11 22:31:27 D! [EC2] Found active network interface
Successfully fetched the config and saved in /opt/aws/amazon-cloudwatch-agent/etc/amazon-cloudwatch-agent.d/file_config.json.tmp
Start configuration validation...
/opt/aws/amazon-cloudwatch-agent/bin/config-translator --input /opt/aws/amazon-cloudwatch-agent/etc/amazon-cloudwatch-agent.json --input-dir /
opt/aws/amazon-cloudwatch-agent/etc/amazon-cloudwatch-agent.d --output /opt/aws/amazon-cloudwatch-agent/etc/amazon-cloudwatch-agent.toml --mod
e ec2 --config /opt/aws/amazon-cloudwatch-agent/etc/common-config.toml --multi-config default
2022/09/11 22:31:27 Reading json config file path: /opt/aws/amazon-cloudwatch-agent/etc/amazon-cloudwatch-agent.d/file_config.json.tmp ...
2022/09/11 22:31:27 I! Valid Json input schema.
I! Detecting run_as_user...
I! Trying to detect region from ec2
2022/09/11 22:31:27 D! [EC2] Found active network interface
No csm configuration found.
Configuration validation first phase succeeded
/opt/aws/amazon-cloudwatch-agent/bin/amazon-cloudwatch-agent -schematest -config /opt/aws/amazon-cloudwatch-agent/etc/amazon-cloudwatch-agent.
toml
Configuration validation second phase succeeded
Configuration validation succeeded
amazon-cloudwatch-agent has already been stopped
Created symlink from /etc/systemd/system/multi-user.target.wants/amazon-cloudwatch-agent.service to /etc/systemd/system/amazon-cloudwatch-agen
t.service.
Redirecting to /bin/systemctl restart amazon-cloudwatch-agent.service
```

Si comprobamos el estado del demonio nos dice que está activo.

```
[ec2-user@ip-10-0-0-29 bin]$ systemctl status amazon-cloudwatch-agent.service
● amazon-cloudwatch-agent.service - Amazon CloudWatch Agent
   Loaded: loaded (/etc/systemd/system/amazon-cloudwatch-agent.service; enabled; vendor preset: disabled)
   Active: active (running) since Sun 2022-09-11 22:31:28 UTC; 1min 51s ago
 Main PID: 8255 (amazon-cloudwat)
   CGroup: /system.slice/amazon-cloudwatch-agent.service
           └─8255 /opt/aws/amazon-cloudwatch-agent/bin/amazon-cloudwatch-agent -config /opt/aws/amazon-cloudwatch-agent/etc/amazon-cloudwat...

Sep 11 22:31:28 ip-10-0-0-29.eu-west-3.compute.internal systemd[1]: Started Amazon CloudWatch Agent.
Sep 11 22:31:28 ip-10-0-0-29.eu-west-3.compute.internal start-amazon-cloudwatch-agent[8255]: /opt/aws/amazon-cloudwatch-agent/etc/amazon-...t.
Sep 11 22:31:28 ip-10-0-0-29.eu-west-3.compute.internal start-amazon-cloudwatch-agent[8255]: I! Detecting run_as_user...
Hint: Some lines were ellipsized, use -l to show in full.
```

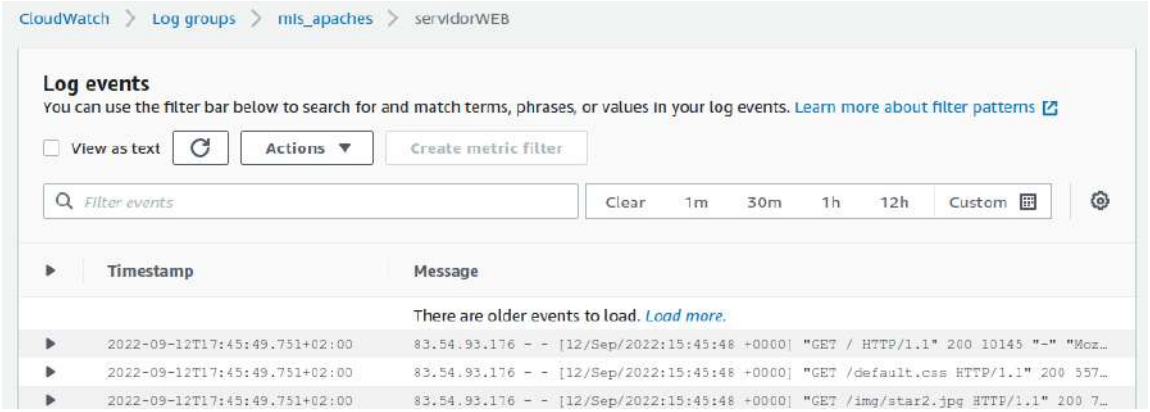
Ahora, si nos vamos a métricas de CloudWatch veremos un nuevo elemento, el CWAgent

Y dentro tendremos las métricas configuradas.



Por ejemplo podremos seleccionar los parámetros de memoria añadidos:

Además, en Log Groups que veremos luego, también guarda un fichero de logs de apache.



TEMA 23 - CloudWatch Alarmas

En “**All alarms**” tenemos todas las alarmas configuradas.

En “**In alarms**” tenemos las alarmas que se han disparado.

Los símbolos, de izquierda a derecha:

- **Alarmas activas**
- **Alarmas ok**
- **Alarmas sin datos suficientes**. Suele ser provocado por estar recien creada o no tiene suficientes DataPoint para saber si está correcta.

Los **tres estados** de las alarmas son “que ha saltado”, “que está ok” o “sin datos suficientes”.

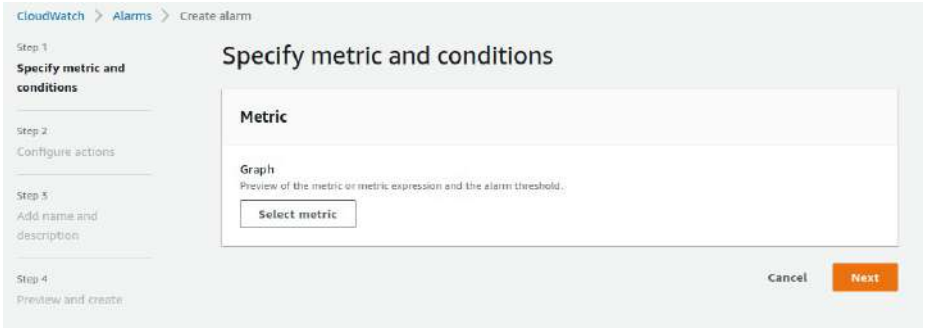
Los **tipos de alarmas** son “compuestas” o “alarmas métricas”.

Y las **acciones** que pueden hacer son “Habilitarlas”, “deshabilitarlas” o “sin acciones”.

Las alarmas se pueden basar en métricas y en condiciones. En base a esto se le indica las acciones.

23.1. - Crear una alarma

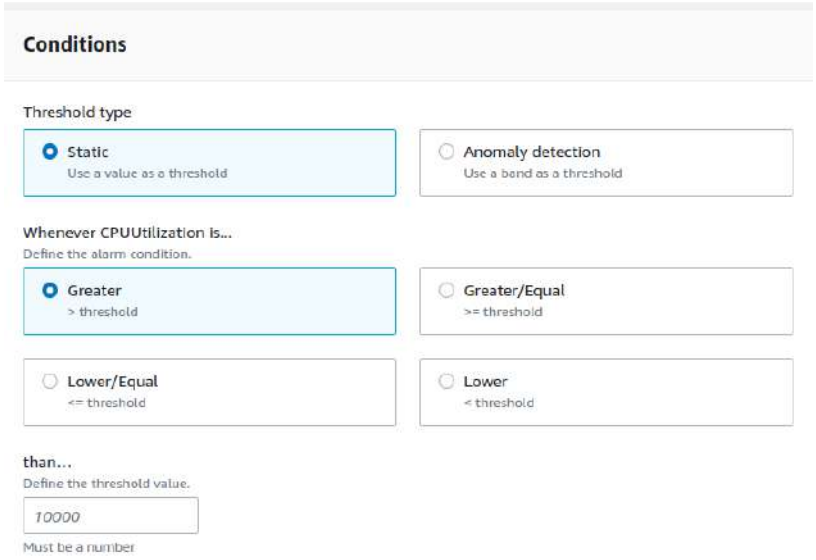
Se crea en 4 pasos.

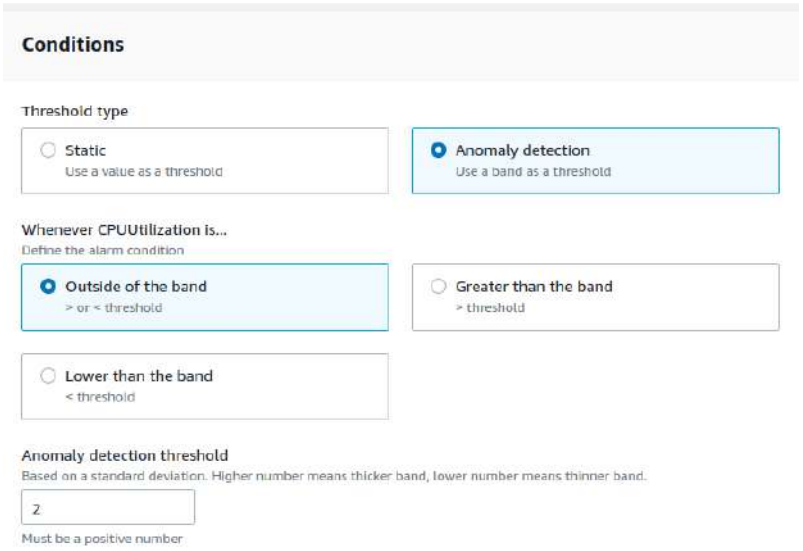


Como ejemplo vamos a crear una alarma de CPU que se conecte al SNS y envíe un correo.

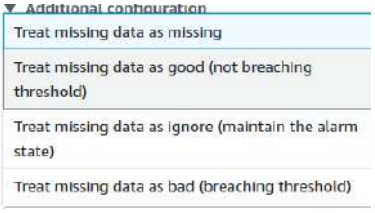
Primero seleccionamos la métrica. Es como el panel de metrics.

Ahora decidimos las condiciones con un valor estático o con una función de anomalía (un rango de comportamiento).





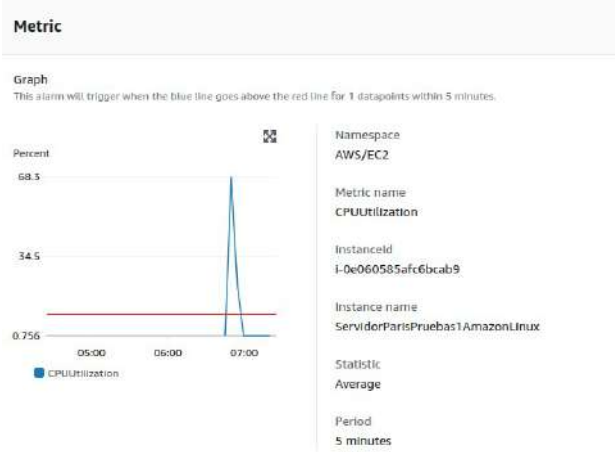
En las opciones adicionales podemos decidir que hacemos con la perdida de datos. Si se dan por perdidos, si se dan por datos correctos o si es algo malo que no haya datos.



Una vez tenemos la métrica y las condiciones ya podemos escoger la acción entre:

- Enviar una notificación. Se puede enviar una notificación si se dispara una alarma, si la alarma se quita y paso ok, o si no hay datos suficientes. Se selecciona o se crea el topic. Se puede enviar más de una notificación, se pueden añadir.
- Una acción de auto escalada – Se selecciona un grupo de autoescalada y se especifica la tarea.
- Una acción EC2 – Podemos hacer un recover (las instancias pequeñas no lo soportan), parar la instancia, Terminarla o reiniciarla.
- Una acción System Manager.

Una vez configurada la acción aparece un preview antes de crear.



Podemos revisar y volver a editar algún aspecto. Si todo es correcto se crea y podemos ver los detalles, las acciones, el histórico y alarmas parentales.

En este caso, después de estresar el CPU (stress -c 10 -t 600s) en el histórico vemos que se ha intentado la notificación pero SNS no tiene los sufientes permisos para enviar el correo

Date	Type	Description
2022-09-12 07:44:20	Configuration update	Alarm "CPU ServidorWEB" updated
2022-09-12 07:41:15	Action	Failed to execute action arn:aws:sns:eu-west-3:992365247711:Base-de-datos. Received error: "CloudWatch Alarms is not authorized to perform: SNS:Publish on resource:arn:aws:sns:eu-west-3:992365247711:Base-de-datos"
2022-09-12 07:41:15	State update	Alarm updated from In alarm to OK.
2022-09-12 07:36:15	Action	Failed to execute action arn:aws:sns:eu-west-3:992365247711:Base-de-datos. Received error: "CloudWatch Alarms is not authorized to perform: SNS:Publish on resource:arn:aws:sns:eu-west-3:992365247711:Base-de-datos"
2022-09-12 07:36:15	State update	Alarm updated from OK to In alarm.
2022-09-12 07:31:15	Action	Failed to execute action arn:aws:sns:eu-west-3:992365247711:Base-de-datos. Received error: "CloudWatch Alarms is not authorized to perform: SNS:Publish on resource:arn:aws:sns:eu-west-3:992365247711:Base-de-datos"
2022-09-12 07:31:15	State update	Alarm updated from Insufficient data to OK.
2022-09-12 07:29:44	Configuration update	Alarm "CPU ServidorWEB" created

Creamos la política de permisos en el topic.

```
{
  "Version": "2012-10-17",
  "Id": "example-ID",
  "Statement": [
    {
      "Sid": "Eventos de S3",
      "Effect": "Allow",
      "Principal": {
        "Service": "s3.amazonaws.com"
      },
      "Action": "SNS:Publish",
      "Resource": "arn:aws:sns:us-west-2:264297788131:Base-de-Datos",
      "Condition": {
        "StringEquals": {
          "aws:SourceAccount": "264297788131"
        },
        "ArnLike": {
          "aws:SourceArn": "arn:aws:s3:::bucket1-apasoft"
        }
      }
    },
    {
      "Sid": "Cloudwatch",
      "Effect": "Allow",
      "Principal": {
        "Service": "cloudwatch.amazonaws.com"
      },
      "Action": "SNS:Publish",
      "Resource": "arn:aws:sns:us-west-2:264297788131:Base-de-Datos",
      "Condition": {
        "ArnLike": {
          "aws:SourceArn": "arn:aws:cloudwatch:us-west-2:264297788131:alarm:Cpu Servidor2"
        }
      }
    }
  ]
}
```

TEMA 24 - CloudWatch Logs

CloudWatch Logs es el entorno que almacena archivos de logs.

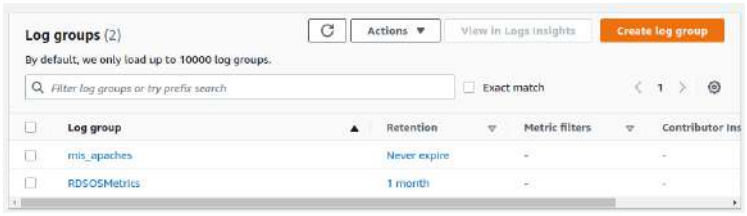
- Permite monitorizar, almacenar y acceder a los archivos de log generados por instancias de Amazon Elastic Compute Cloud (Amazon EC2), AWS CloudTrail, Route 53 y otras fuentes y servicios de AWS. Algunos entornos son automáticos y otros necesitan un agente que envíe los logs.
- Permite centralizar los logs de todos los sistemas, aplicaciones y servicios de AWS en un único punto central y de gran escalabilidad.
- Se puede buscar a través de patrones específicos, filtrarlos en función de campos específicos o archivarlos de forma segura para futuros análisis. CloudWatch inside.
- Se pueden implementar como un flujo único y consistente de eventos ordenados por tiempo.
- Se pueden consultar y ordenarlos según múltiples dimensiones, agruparlo por campos específicos, crear cálculos personalizados y visualizar datos en DashBoards y paneles personalizables.

Conceptos:

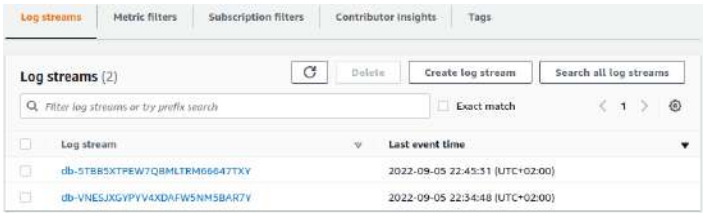
- **Log Events**: Son los eventos registrados en logs. Es un registro de alguina actividad que ha guardado una aplicación o el recurso que se está monitorizando. Contiene dos propiedades:
  - **Timestamp:** Cuando se produjo el evento.
  - **Mensaje** del evento sin procesar. Son otras herramientas las que procesan el mensaje.
- **Log Stream**: Es una secuencia de log events que comparten el mismo origen. Representar la secuencia de eventos procedente de una aplicación o recurso.
- **Log Groups**: Un grupo de Log Stream que comparten las mismas características. Cada Log Stream pertenece a un Log Group.
- **Metric Files**: Permiten estraer dobservaciones y datos.

En el panel de CloudWatch tenemos dos pestañas de Logs: Log Groups y Logs Insights.

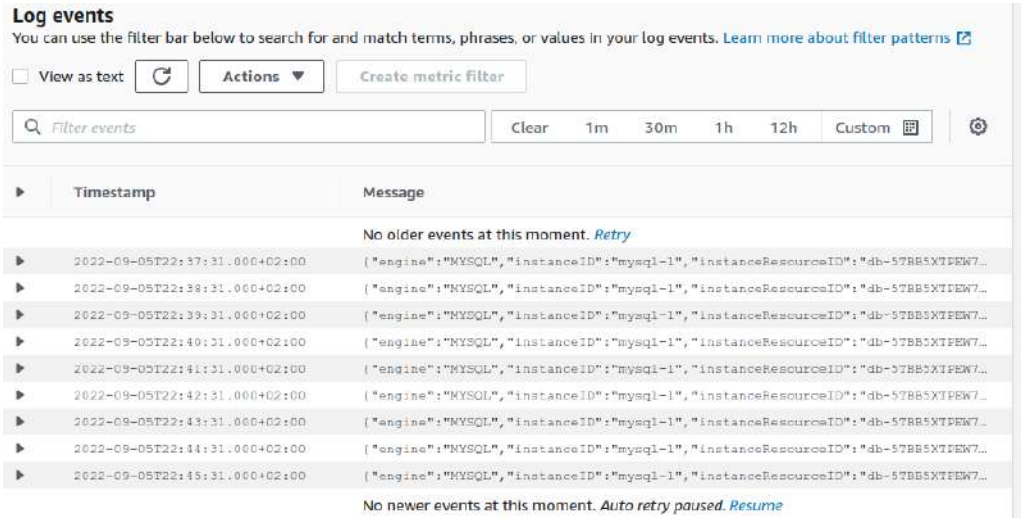
- **Log Groups** – Donde quedan los grupos configurados.



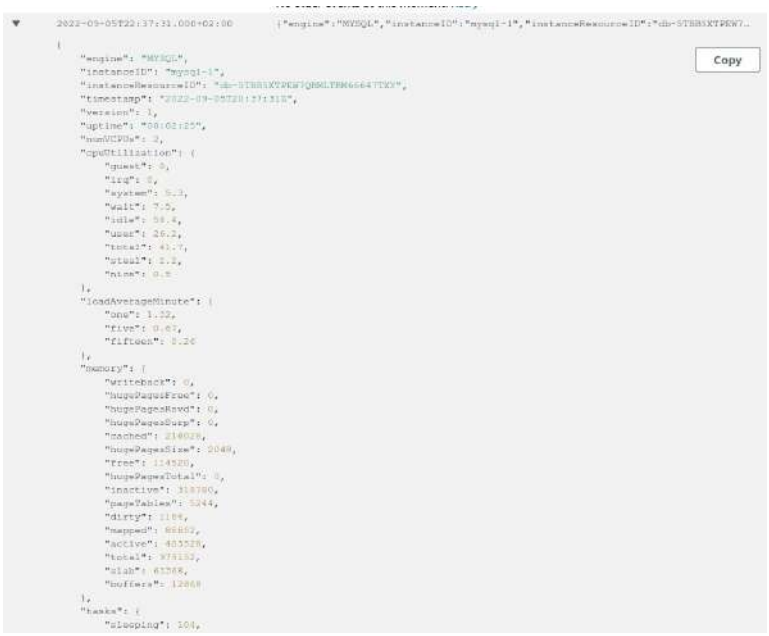
  - Dentro está compuesto de **Logs Stream**



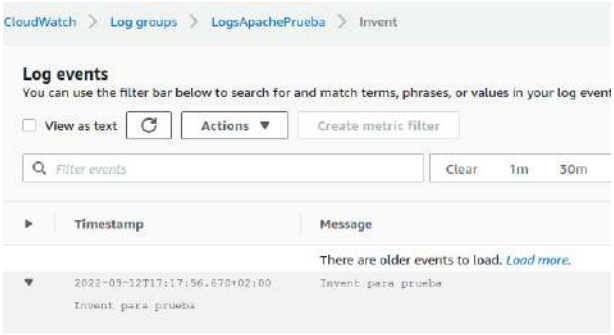
- Y dentro de cada log stream hay una serie de **eventos de logs**.



Cada **event log es un fichero sin procesar** con detalles del evento. En este caso un JSON.



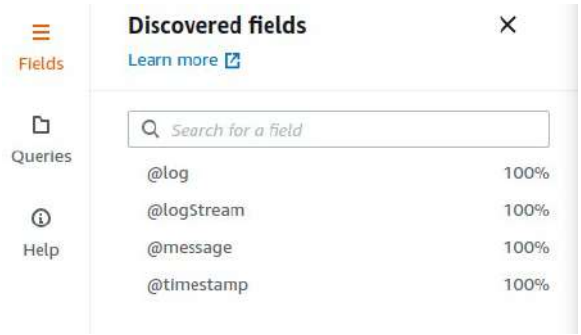
Se puede crear un log group. Nos pedirá el nombre, cuanto tiempo se retienen los logs y el KMS key ARN (Si estuviera encriptado). Además también se puede etiquetar. Dentro de log group se puede crear un log stream y dentro crear un log event.



- Logs Insights – Para efectuar búsquedas dentro de los grupos de logs. Se pueden buscar patrones, filtrar filas, información concreta, etc

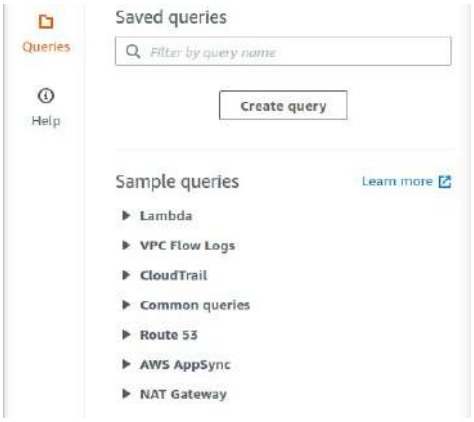
  Tenemos que seleccionar la query en fielfs, decir el orden en sort y el limite.

En el sidebar derecho hay facilidades para buscar. Por ejemplo, los fields (Campos de las columnas)

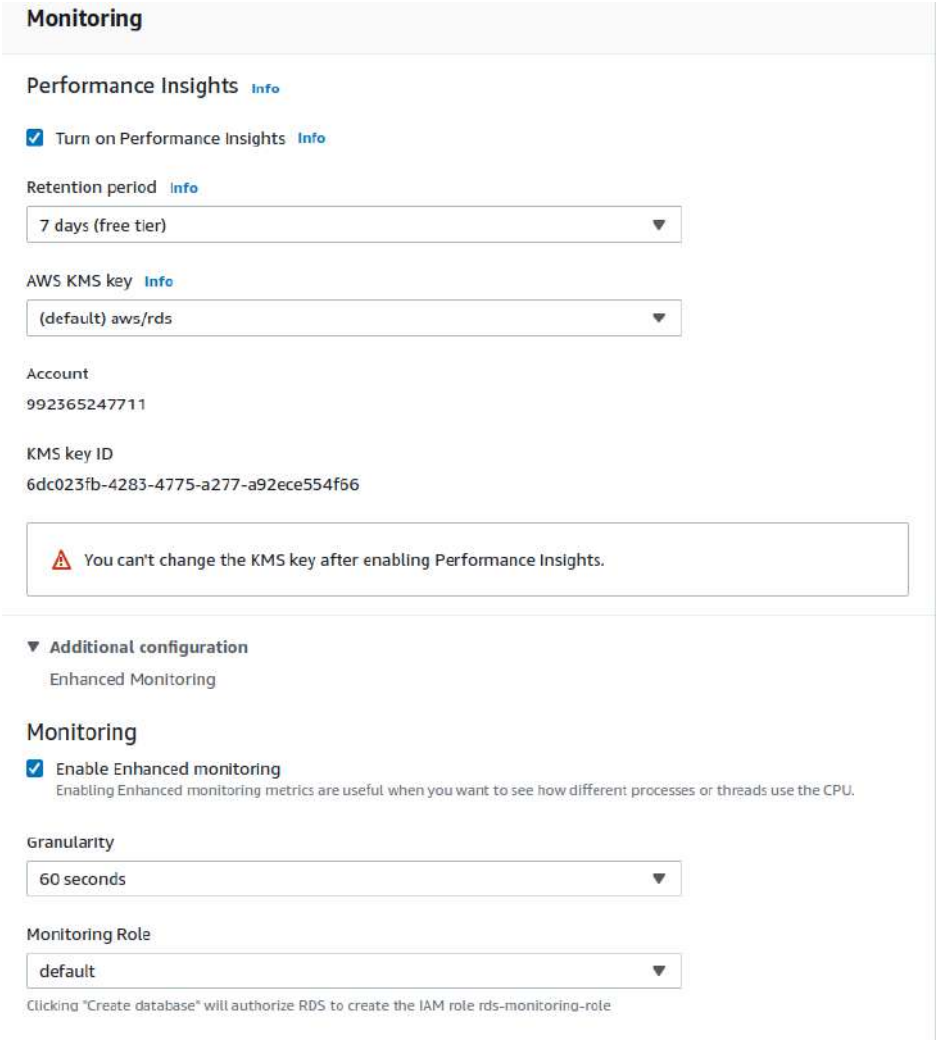


Cuando aplicamos con run query aparecen los datos de la búsqueda

En las queries del sidebar derecho tenemos ejemplos de búsqueda y las búsquedas que guardamos.



Ejemplo. **Generar Logs desde RDS**. Cuando creamos una BBDD, por una parte, desde monitoring podemos configurar eventos que se envían a CloudWatch

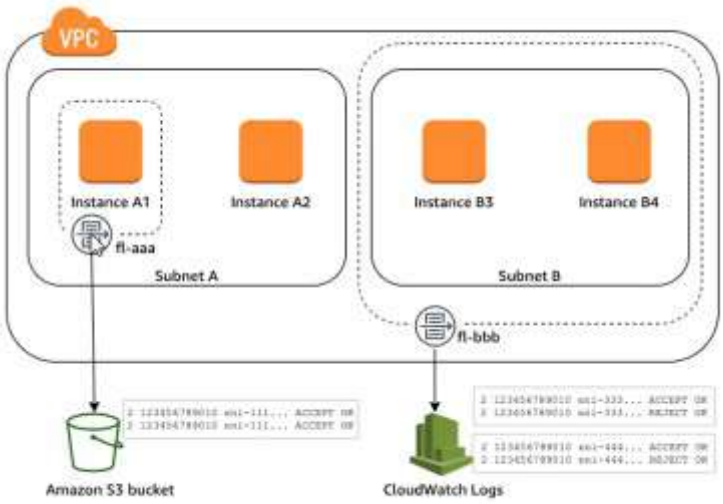


Además, en opciones avanzadas podemos indicar los logs concretos que queremos que se envíen.

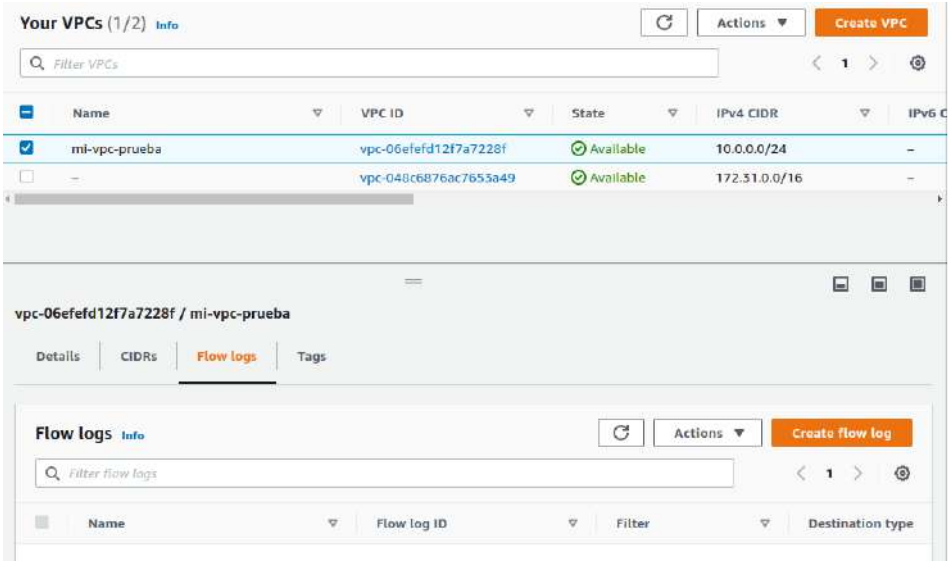
Los logs de error siempre se mandan, pero los otros tres (Audit, General y Slow Query) no se mandarán por si solos. Debemos configurar los parámetros con un parameter group que lo indique:

Ahora, en la BBDD creada con el paremeter group configurado y los logs activado, podemos ver los logs. Y en CloudWatch, en el grupo de logs “RDSOSMetrics” aparecerán los logs de SO de esta BBDD (CPU, RAM, etc) por tener activada la monitorización detallada.

Se puede capturar el tráfico de las VPC, de las subnet e incluso a nivel instancia con **VPC Flow Logs**.



Para crear un Flow log debemos estar en las VPCs y tiene una pestaña Flow Logs.



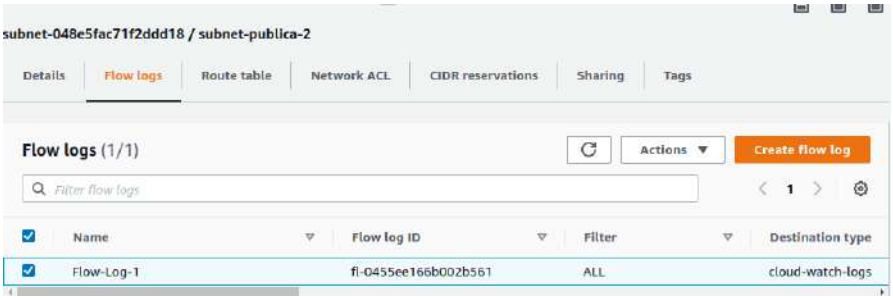
Podremos configurar algunos parámetros:

- Nombre
- Filtros. Accept, Reject o All
- Intervalo de las agragaciones
- Cumplimentando algunos datos lo podemos mandar a
  - CloudWatch Log – Tendremos que escoger un Log Group y un IAM role
  - Amazon S3 Bucket

    - Kinesis Firehose con la misma cuenta o con otra cuenta. (Es un servicio de transmisiones generadas en tiempo real para el análisis de datos.)

- Podemos escoger el formato en el que queremos la grabación de logs. Puede ser por defecto o personalizada escogiendo los datos.

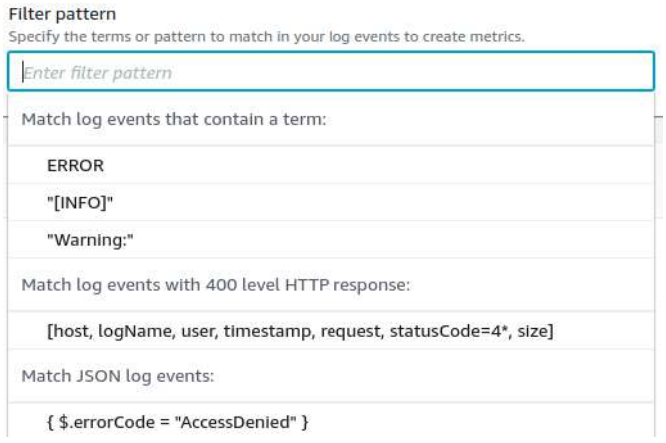
En una subnet también podemos crear un flowlog. Ahora que hemos creado un flow log a nivel de VPC las subnets han heredado este flowlog



En las Network interfaces (Tarjetas de red) de EC2 también podemos crear un Flow Log. Como pasa con las subnets, al tener la VPC un flow log las tarjetas de red lo heredan.

Con lo cual, las instancias también lo heredan.

Las **metric Filters** son filtros para limitar el resultado que queremos extraer. Se hace a través de patrones de filtros.

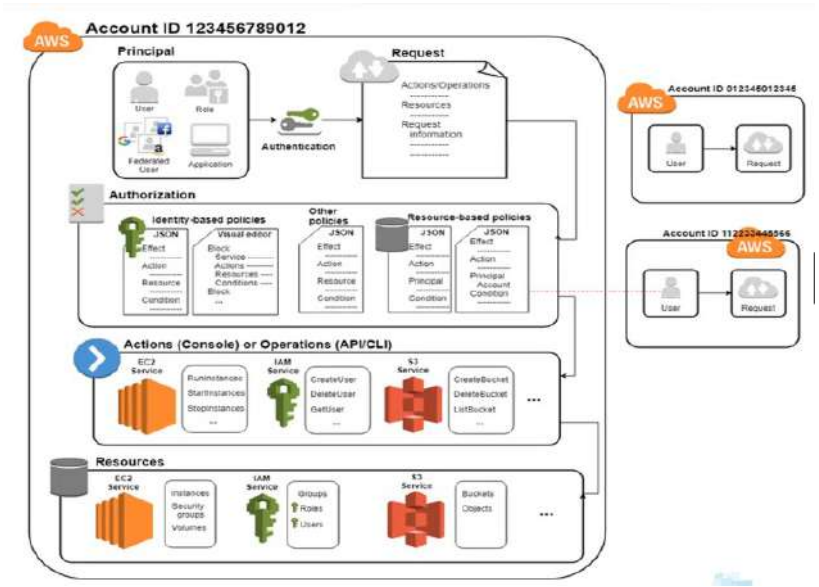


TEMA 25 - IAM Identify Access Management. Gestión de identidades en AWS

Se debe dejar de trabajar con el usuario root inicial.

IAM permite definir el acceso a los recursos de AWS.

- **Recurso IAM**: Usuarios, grupos, roles, políticas, providers.
- **Identities**: Identifican accesos(Usuarios, grupos y roles)
- **Entities**: Objetos que usan para autenticarse (Usuarios y roles)
- **Principal**: Persona o aplicación que asume un usuario o rol. (User, role, Federated User, application)



Lo primero que tenemos que hacer es **autenticarnos** para tener la **autorización** (permisos). Podemos tener políticas (permisos) basadas en identidad (usuarios) o basadas en recursos o en tags u otras. Luego tenemos **acciones** que se pueden efectuar en los recurso que queremos acceder.

Principal - Podemos entrar de tres formas: Usuario ROOT, Usuario IAM y AWS ROLE.

Usuario ROOT

- Usuario principal con el que se crea la cuenta al que se le denomina “root user”. Es similar al usuario root de Linux.
- Se puede usar para acceder por consola o a través de código.

- Puede hacer cualquier tipo de operación.
- No se debería usar para las tareas diarias

Usuario IAM

- Es el usuario que se debe usar en el día a día.
- Representan usuario o aplicaciones que se conectarán y usarán los recursos de AWS
- Pueden acceder desde la consola o desde un entorno CLI o SDK
- Se les puede asignar distintos tipo de permisos para su trabajo diario.

Grupos

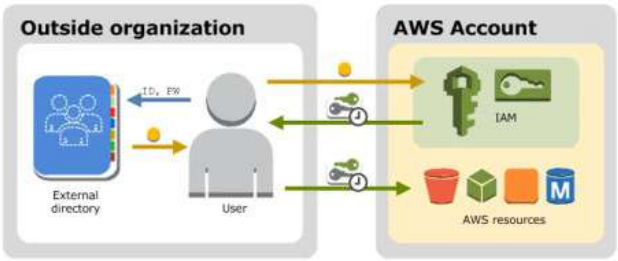
- Colección de usuario IAM que permite agrupar permisos para asignarlos a los usuarios.

AWS ROLE

- Se usan para dar permisos y privilegios durante un momento determinado.
- Cuando un determinado actor asume un rol, se le asigna un token temporal que le permite acceder a un determinado recurso para poder utilizarlo.
- Es similar a un usuario IAM pero en vez de estar asociado a una persona, es asumido por quien lo necesite para unas gestiones concretas.
- No tiene credenciales ni claves u por tanto no tienen que ser enviadas o integradas dentro de una aplicación
- El servicio que asigna estos Tokens se denomina “AWS Security Token Service”

Federación

- Se pueden “federar” usuarios ya creados dentro de nuestro propio entorno.
- De esta forma se puede asumir unas credenciales temporales para el acceso a AWS.



Políticas y permisos

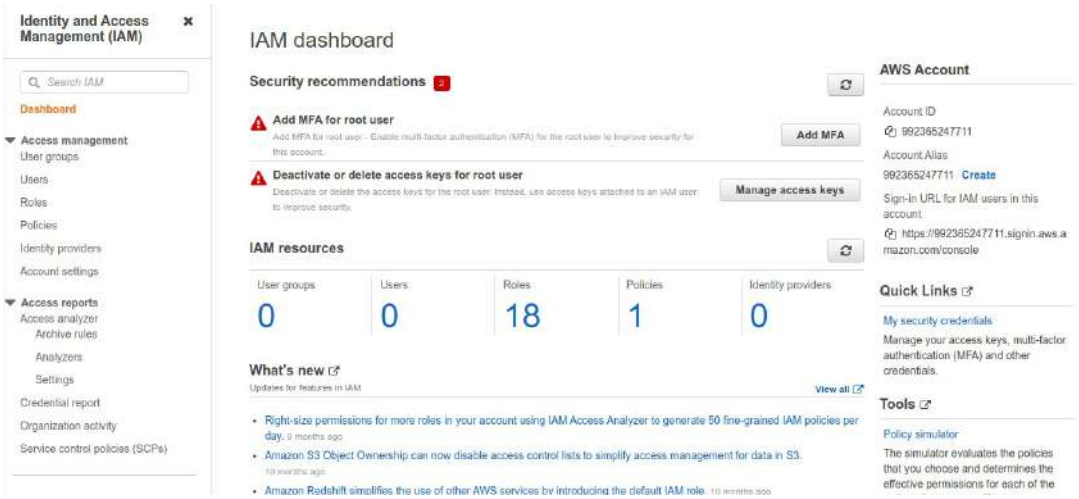
- Para gestionar el acceso a un usuario o rol a AWS se crean políticas para asignarles.
- Es un objeto AWS que al ser asignado a un usuario o recurso define sus permisos. Por lo tanto, identifican que se puede o no se puede hacer.
- Suelen estar hechas con un fichero JSON.

```
{
 "Version": "2012-10-17",
 "Statement": {
  "Effect": "Allow",
  "Action": "dynamodb:*",
  "Resource": "arn:aws:dynamodb:us-east-2:123456789012:table/Books"
 }
}
```

Tipos de políticas

- **identity policy** – Se asocian a una entidad IAM, como usuarios, grupos o roles. Permite controlar las acciones que se pueden realizar en un recurso y bajo que condiciones. Pueden ser:
  - **Manager** – Políticas independientes que pueden ser asignadas a multiples usuarios, grupos o roles. Pueden ser creadas por AWS o personalizadas por nosotros. Es lo habitual.
  - **Inline** – Se integran en un usuario, grupo o rol. Son las menos adecuadas.
- **Resource-Based Policies** – Se asocian a un determinado recurso AWS (por ejemplo S3) y determinan las acciones que se pueden hacer sobre ese recurso.
- Otras políticas: como las de tags.

25.1. - Consola IAM



Solo abrirla aparecen dos avisos:

- Añadir el MFA para el usuario root
- Desactivar o eliminar access keys para usuario root. Las hemos utilizado para CLI pero no se deberían usar.

Lo ideal es que root solo se pueda conectar por consola.

A la derecha tenemos:

- Account ID que es única, la comparten root y todos los usuarios que se creen.
- Account Alias. Es único para cada usuarios que se cree.
- URL de acceso para los usuarios IAM
- Más herramientas. Seguridad, simulador de políticas, federación de identidades…

En la izquierda tenemos a los grupos de usuarios, usuarios, roles, políticas, proveedores de identidad y opciones.

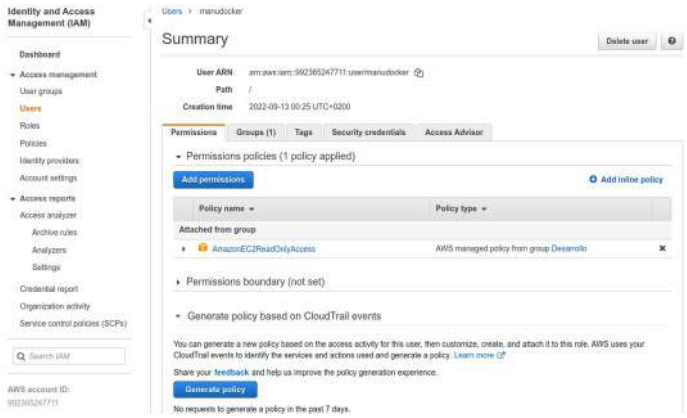
Para **crear un grupo de usuarios** podemos configurar los siguientes parámetros:

- Nombre
- Se pueden añadir usuarios y otros grupos.
- Indicar los permisos (Políticas) del grupo.

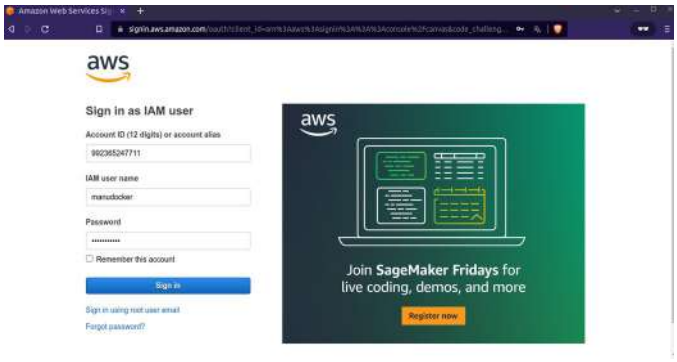
Para **crear un usuario** podemos configurar los siguientes parámetros:

- Nombre/s. Se puede añadir hasta 10 usuarios para crearlos de una vez.
- El tipo de acceso
  - A través de código AWS API, CLI, SDK… Herramientas de desarrollo
  - Con el management Console Access.
    - Pregunta contraseña y si la primera vez que se entra se debe cambiar.
- Políticas de permisos.
  - Añadir a un grupo
  - Copiar permisos de otro usuario
  - Asignar políticas
- Tags
- Muestra resumen y después la posible descarga de un CSV para enviar el usuario y contraseña.

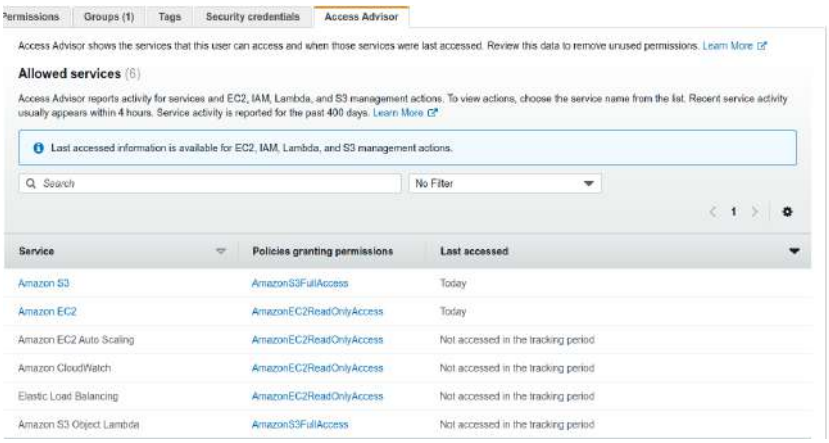
Se puede entrar en el usuario y en el grupo para poder visualizar las opciones, modificar y borrar.



Para acceder tendremos que entrar por la URL con la ID account



Access Advisor es una herramienta para poder comprobar que clases de políticas tiene un grupo o un usuario.



Para **crear un rol** (Un grupo de permisos) podemos configurar los siguientes parámetros:

- Tipo de entidad de confianza
  - Servicios AWS
    - Caso de uso
      - EC2

        - Lambda
        - Otro servicio
    - Cuentas AWS
    - Identidad web – AWS Cognito. Para crear usuarios para aplicaciones externas.
    - federación SAML 2.0
    - Políticas de confianza personalizadas.
- En los permisos podemos filtrar los que queremos. Clicando en el buscador tenemos más filtros.
- Nombre y descripción
- Aparece el json de las políticas seleccionadas y un resumen de las que son.
- Tags

Políticas de seguridad –

- **Identity** – que se asocian a un usuario, un rol o un grupo.
    - **Manager** – Políticas independientes que pueden ser asignadas a multiples usuarios, grupos o roles. Pueden ser creadas por AWS o personalizadas por nosotros. Es lo habitual.
    - **Inline** – Se integran en un usuario, grupo o rol. Son las menos adecuadas.
- **Resource –** que se asocian a un recurso.

Todas comparten el mismo **fichero JSON**. Esqueleto, empezando por los parámetros Identity y Resource:

- **Versión** – En estos momentos es 2012-10-17
- **Statement** – Nombre del grupo completo de parámetros.
- **SID** – Código / id de la política
- **Effect** – Permitir o denegar acceso. Puede ser Allow o Deny
- **Action** – Acciones que se quieren denegar o permitir
- **Resource** – El recurso donde se aplica el permisos
- **Condition** – Condiciones para restringir o limitar el acceso.

Ahora, este parámetro tan solo se añade a políticas de recursos (Resources)

- **Principal** – Usuario, rol o cuenta para el que debería aplicarse la política.

Para **crear una Managed Policy** podemos hacerlo directamente en un JSON o seguir un asistente visual configurando los siguientes parámetros:

- Servicio - para el que será la política.
- Acciones - que se suelen dividirse en
  - List – Listar, describir
  - Read – Lectura
  - Tagging – Etiquetado
  - Write – Escritura
  - Permisos de gestión
- Recursos. Por ejemplo, si hemos escogido S3 podremos decir uno o n buckets, uno o n object, uno o n accesspoint
- Condiciones. Requerir MFA, Recurso IP u Otras
- Luego podemos poner Tags
- Nombre y descripción. Nos enseña un resumen.

Dentro de la política vemos

- Permisos
- Uso – Quien o qué la está usando
- Tags
- Versiones de política – pueden haber hasta 5 versiones guardadas.
- Access Advisor

Si queremos modificarla tan solo tenemos que abrirla y “Edit policy”. Tendremos las mismas opciones que creando: Fichero JSON o Editor visual. Cuando guardamos tendremos dos versiones.

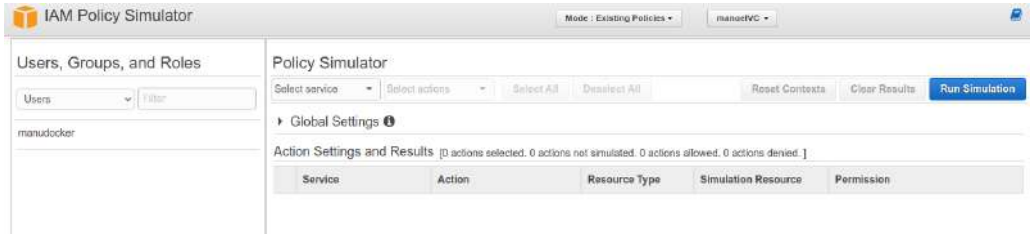
	Version	Creation time
▸	Version 2 (Default)	2022-09-13 10:19 UTC+0200
▸	Version 1	2022-09-13 10:03 UTC+0200

Para **asignar las politicas** a un grupo, a un usuario o a un rol tan solo tenemos que ir a la identidad concreta y, en permisos, añadirla con Attach policy.

Para **crear una política inline** (Es mejor no usarlas, se asocian directamente a la identidad) tan solo hay que ir al rol, usuario o grupo y a la hora de añadir se selección “Create inline policy”.

Para **crear una política de recurso** se hace en el mismo recurso dentro del servicio. Por ejemplo, dentro de un topic concreto de SNS.

Tenemos un **simulador de políticas** en el sidebar derecho, es un servicio a parte.
https://policysim.aws.amazon.com/



En la izquierda podemos escoger la identidad concreta para simular. A la derecha podemos comprobar las acciones que puede realizar la identidad seleccionada. En algunos casos, puede haber ACL o elementos que puedan dar resultados diferentes a los reales.

Credential Report – Se pueden descargar informes en CSV de IAM para poder tratar los datos de usuarios.

Actividad de la organización – Tenemos la estructura de la organización. Las organizaciones se utilizan para tener una división de los servicios y utilidades de AWS. Aquí podemos ver los últimos accesos. Se pueden gestionar las organizaciones en el servicio AWS organizations.

Service control policies (SCPs)– Son políticas a nivel de organización. Aquí se puede ver los usos de estas poíticas. También se gestionan desde AWS organizations. Un ejemplo de SCPs es que no se puedan arrancar instancias caras para que nadie pueda dar estos usos con los costes que supone.

Access analyzer – Podemos crear analisis personalizados para comprobar por partes los componentes de AWS. Nos pedirá el nombre y podremos lanzarlo para toda la organización o solo en la cuenta actual. A través de las tags podemos filtrar los recursos que queremos analizar. Con el

resultado se podrá ver los activos, los archivados, los resueltos y todos. Se pueden mover entre estos resultado. Tenemos 3 espacio: Reglas archivadas, análisis y opciones.

Storage Lens en S3 - Se puede crear un DashBoard personalizado con métricas y datos en relación a S3. Tendrá cuadros de mandos para una visualización rápida y poder gestionar los datos. No se puede acceder a un dashboard con usuario ROOT, tiene que ser un usuario IAM con los permisos:

- s3:ListStorageLensConfigurations
- s3:GetStorageLensConfiguration
- s3:GetStorageLensDashboard

En cambio, si que podemos crear un Dashboard con usuario ROOT. Podemos darle los siguientes parámetros:

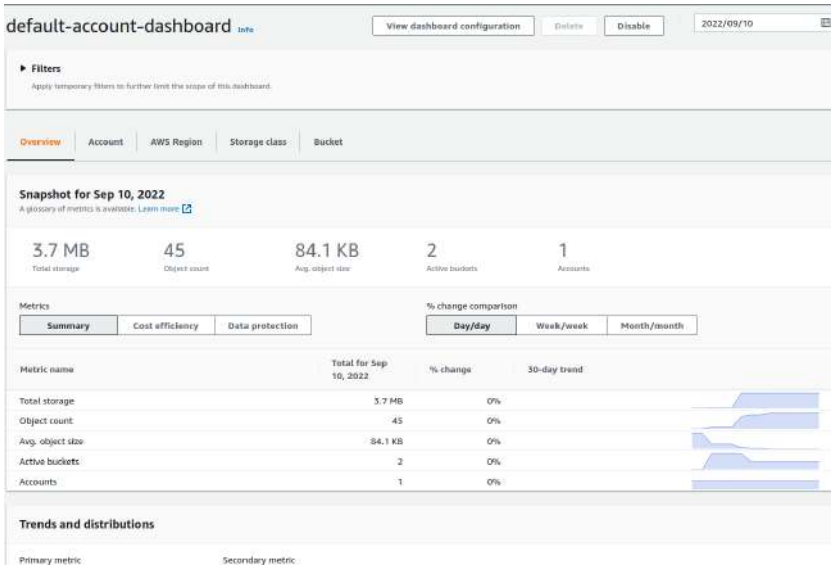
- Nombre
- Elegimos la region
- Si queremos actualizar las métricas diariamente.
- Tags
- Datos de la organización
- Que buckets queremos incluir
- Elegir entre métricas gratis o métricas avanzadas.
- Si queremos exportar las métricas cada 24 horas. Si habilitamos la exportación:
  - Podemos elegir el formato entre csv o Apache Parquet
  - Podemos elegir el bucket destino
  - Los permisos del bucket
  - Encriptación

Aun creando el Dashboard desde usuario ROOT no podremos entrar en él con esta cuenta, tendremos que entrar con un usuario IAM con los permisos adecuados.

Podemos crear una política de acceso a Dashboards de Storage Lens para un usuario que solo se encargue de esto.

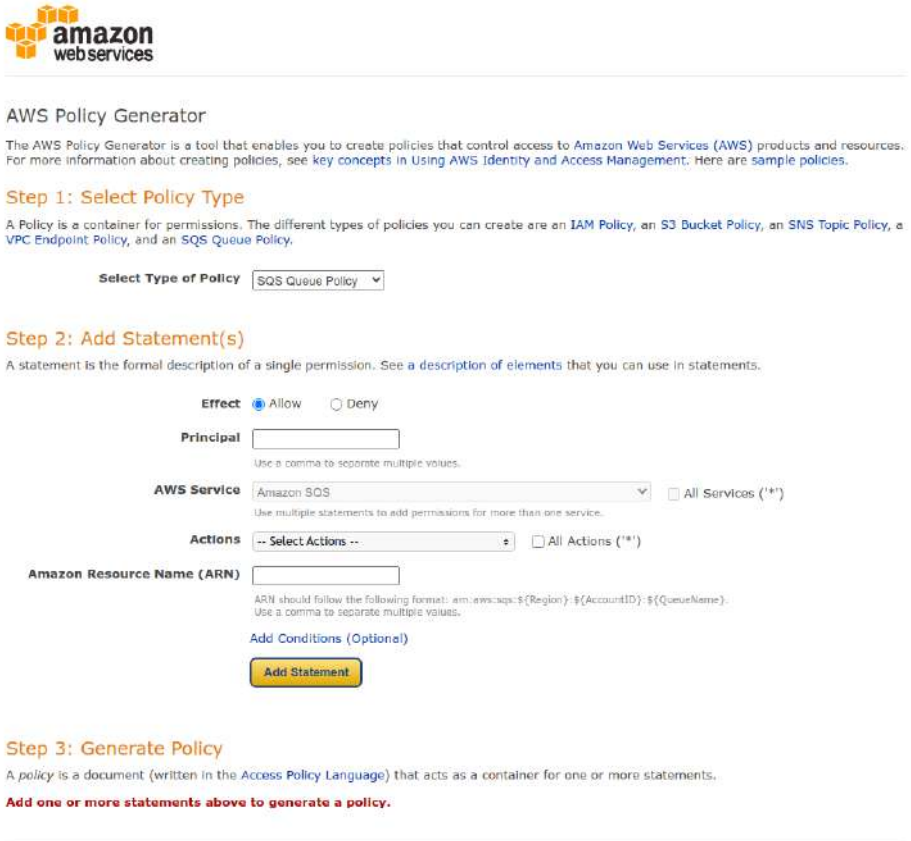
Cuando entramos en el DashBoard recien creado no tiene datos.

En el default-account-dashboard aparece de todas las regiones y todos los buckets.

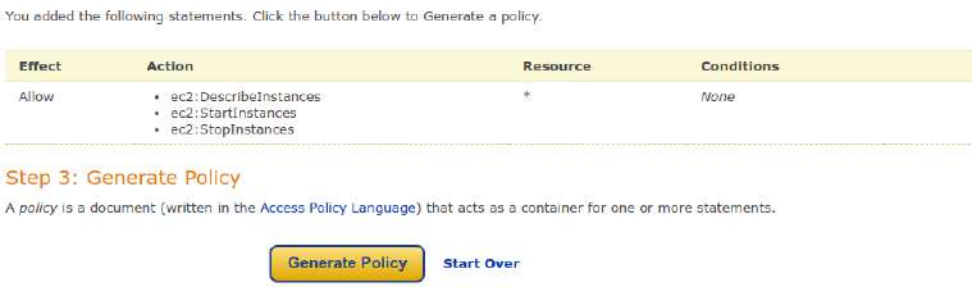


El panel aparece las métricas que ha ido recogiendo de los buckets.

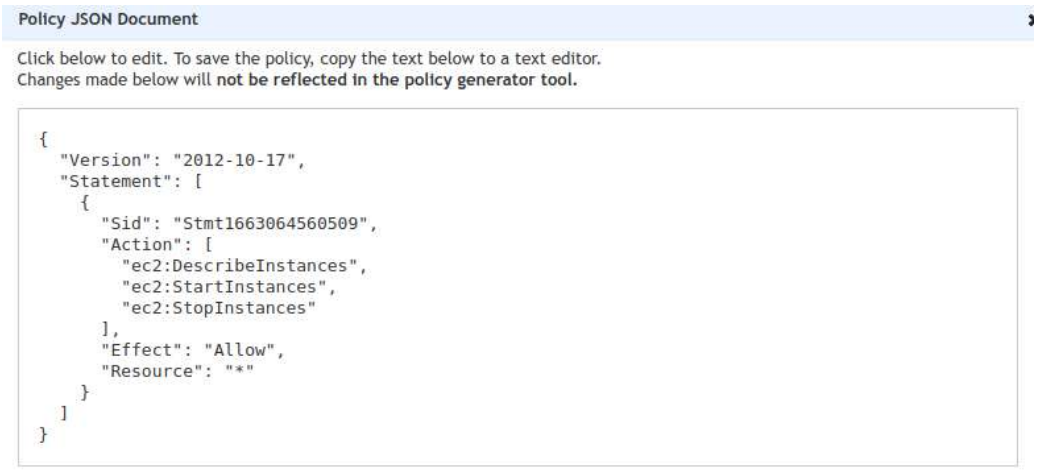
Volviendo a las políticas de IAM, existe un **generador de políticas** en este enlace: https://awspolicygen.s3.amazonaws.com/policygen.html



Es algo parecido al asistente que tenemos en la creación de las políticas. Antes de crearla nos da un resumen.



Nos dará un fichero JSON



Puede ser un mecanismo para descargar el archivo json. Hay que copiar y guardar en un documento plano con la extensión.

25.2. - CLI IAM

En **CloudShell** entramos en una shell virtual de la region. Aquí tenemos 1 GB gratuito por region. Tenemos preinstalado AWS CLI, Python, Node.js y otros.

El comando de IAM es

aws iam

Listar grupos

aws iam list-groups

Recordatorio de query y del formato de salida

aws iam list-groups --query “Groups[].GroupName” --output table

Listar las políticas (Saca todas las políticas, pueden ser muchas)

aws iam list-policies

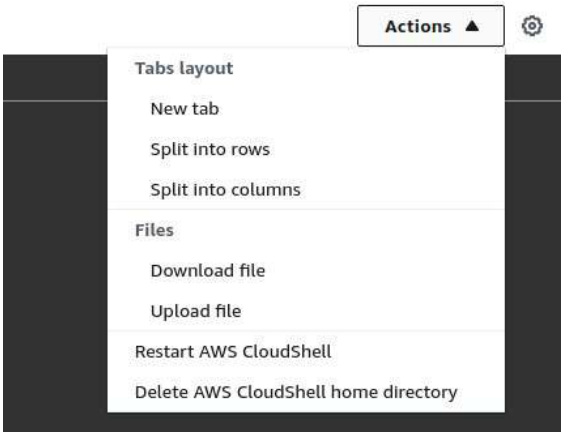
Nos ayuda a filtrar la opción scope con las posibilidades Local o AWS o All

aws iam list-policies --scope Local

Para listar los usuarios

aws iam list-users

Desde CloudShell podemos realizar algunas acciones en actions como “Nueva pestaña”, “Dividir filas”, “Dividir columnas”, “Descargar archivos” (Le daremos el path), “Subir archivos” (Lo dejará en la carpeta del usuario), “Reiniciar el servicio” o “Borrar el directorio home de AWS CloudShell”.

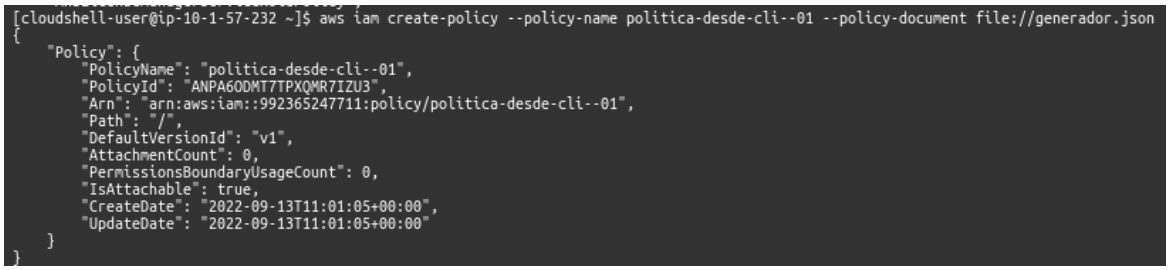


En la rueda dentada podemos cambiar algunas opciones del diseño.

Subimos el fichero de la política creada con el generador.

Para crear una políticas:

aws iam create-policy --policy-name politica-desde-cli--01 –policy-document [Podría subirlo desde un bucket con el arn y los permisos adecuado. O desde local con file://nombredocumento]

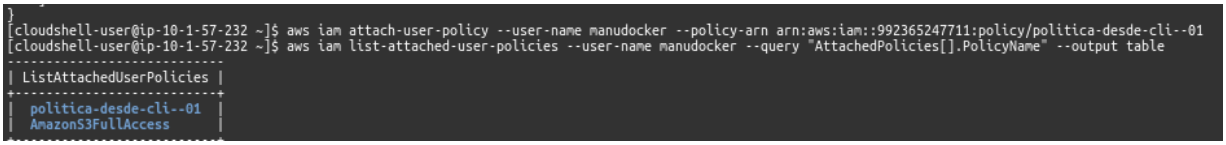


Ahora podemos darles estas políticas a una identidad o crear una identidad con él, etc

Añadir la política a un usuario y comprobar política del usuario en concreto:

aws iam attach-user-policy --user-name manudocker --policy-arn arn:aws:iam::992365247711:policy/politica-desde-cli--01

ws iam list-attached-user-policies --user-name manudocker



Todos los subcomandos de aws iam:

add-client-id-to-open-id-connect-provider
add-role-to-instance-profile
add-user-to-group
attach-group-policy
attach-role-policy
attach-user-policy
change-password
create-access-key
create-account-alias
create-group
create-instance-profile
create-login-profile
create-open-id-connect-provider
create-policy
create-policy-version
create-role
create-saml-provider
create-service-linked-role
create-service-specific-credential
create-user
create-virtual-mfa-device
deactivate-mfa-device
delete-access-key
delete-account-alias
delete-account-password-policy
delete-group
delete-group-policy
delete-instance-profile
delete-login-profile
delete-open-id-connect-provider
delete-policy
delete-policy-version
delete-role
delete-role-permissions-boundary
delete-role-policy
delete-saml-provider

delete-server-certificate
delete-service-linked-role
delete-service-specific-credential
delete-signing-certificate
delete-ssh-public-key
delete-user
delete-user-permissions-boundary
delete-user-policy
delete-virtual-mfa-device
detach-group-policy
detach-role-policy
detach-user-policy
enable-mfa-device
generate-credential-report
generate-organizations-access-report
generate-service-last-accessed-details
get-access-key-last-used
get-account-authorization-details
get-account-password-policy
get-account-summary
get-context-keys-for-custom-policy
get-context-keys-for-principal-policy
get-credential-report
get-group
get-group-policy
get-instance-profile
get-login-profile
get-open-id-connect-provider
get-organizations-access-report
get-policy
get-policy-version
get-role
get-role-policy
get-saml-provider
get-server-certificate
get-service-last-accessed-details
get-service-last-accessed-details-with-entities
get-service-linked-role-deletion-status
get-ssh-public-key
get-user
get-user-policy
list-access-keys

list-account-aliases
list-attached-group-policies
list-attached-role-policies
list-attached-user-policies
list-entities-for-policy
list-group-policies
list-groups
list-groups-for-user
list-instance-profile-tags
list-instance-profiles
list-instance-profiles-for-role
list-mfa-device-tags
list-mfa-devices
list-open-id-connect-provider-tags
list-open-id-connect-providers
list-policies
list-policies-granting-service-access
list-policy-tags
list-policy-versions
list-role-policies
list-role-tags
list-roles
list-saml-provider-tags
list-saml-providers
list-server-certificate-tags
list-server-certificates
list-service-specific-credentials
list-signing-certificates
list-ssh-public-keys
list-user-policies
list-user-tags
list-users
list-virtual-mfa-devices
put-group-policy
put-role-permissions-boundary
put-role-policy
put-user-permissions-boundary
put-user-policy
remove-client-id-from-open-id-connect-provider
remove-role-from-instance-profile
remove-user-from-group
reset-service-specific-credential

resync-mfa-device
set-default-policy-version
set-security-token-service-preferences
simulate-custom-policy
simulate-principal-policy
tag-instance-profile
tag-mfa-device
tag-open-id-connect-provider
tag-policy
tag-role
tag-saml-provider
tag-server-certificate
tag-user
untag-instance-profile
untag-mfa-device
untag-open-id-connect-provider
untag-policy
untag-role
untag-saml-provider
untag-server-certificate
untag-user
update-access-key
update-account-password-policy
update-assume-role-policy
update-group
update-login-profile
update-open-id-connect-provider-thumbprint
update-role
update-role-description
update-saml-provider
update-server-certificate
update-service-specific-credential
update-signing-certific
update-user
upload-server-certificate
upload-signing-certificate
upload-ssh-public-key
wait
wizard

TEMA 26 - CloudTrail: Monitorización accesos de usuarios

Es una herramienta de monitorización de la actividad de las cuentas de AWS. Se captura toda la actividad de AWS y se puede guardar en la consola de CloudTrail, a un bucket S3 o en CloudWatch Log.

Podemos activar alarmas en eventos.

Se puede encontrar información en la consola o en otros servicios como Amazon Athena (queries sobre ficheros de log con lenguaje light SQL).

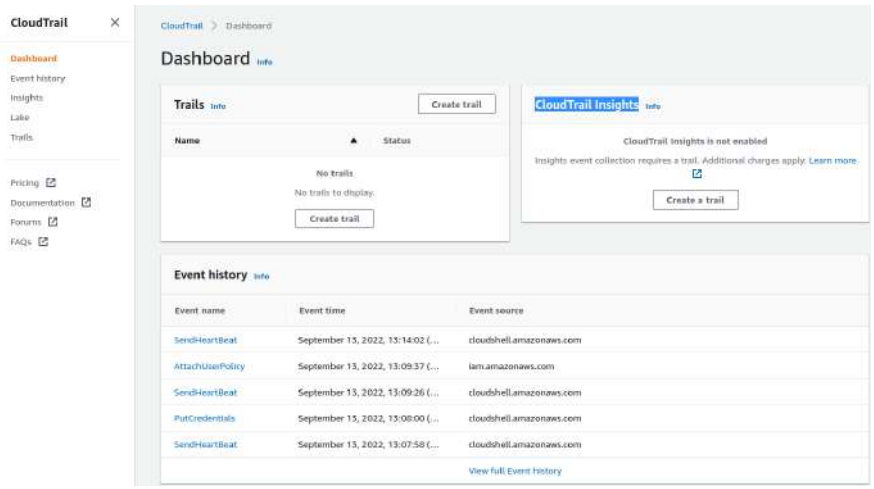
Otro servicio es CloudTrail lake donde almacenamos los datos como Data store y podemos hacer consultas SQL.

Por defecto, se capturan todos los datos de administración gratuitos, reteniendolos por 90 días.

Cloud Trail Lake nos permite guardar la información más tiempo, pero es caro.

En el dashboard vemos:

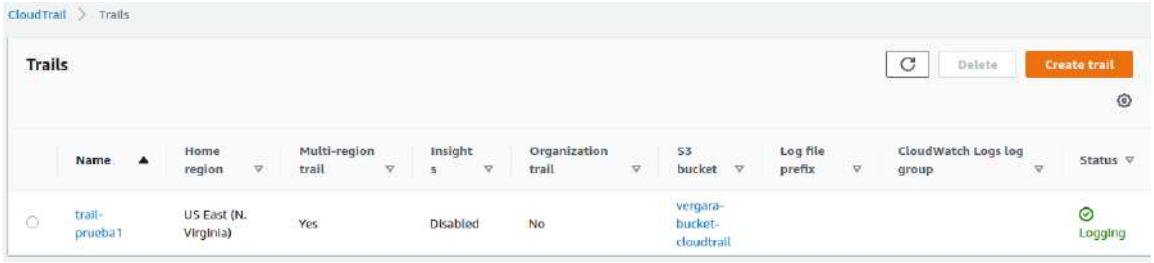
- Trails
- CloudTrails Insights
- Event history



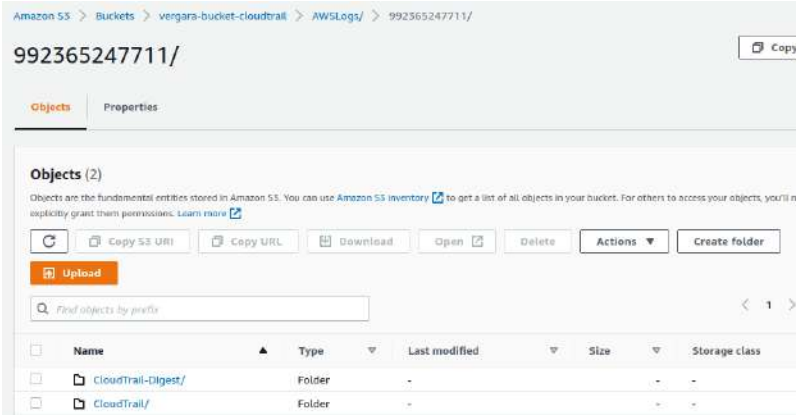
Un **trail** es un contenedor donde guardamos los eventos con los que trabajar. Cuando creamos uno podemos configurar:

- Nombre
- Almacenamiento. Podemos crear un bucket o usar uno existentes.
- Encriptación

    - Clave KMS. Podemos crear una o usar una existente.
- Validación de logs – Para controlar que no haya manipulaciones.
- Crear notificación de entrega por SNS.
- Activar CloudWatch Logs
- Eventos a guardar
    - Administración
        - Read
        - Write
        - Excluir AWS KMS
        - Excluri RDS
    - Data (gestión de recursos)
        - Podemos escoger los recursos
    - Insights (actividades inusuales). - Cuando activamos esta opción suele tardar en activarse porque busca patrones. Además, tiene una pantalla de eventos con detalles.
        - API call rate
        - API error rate



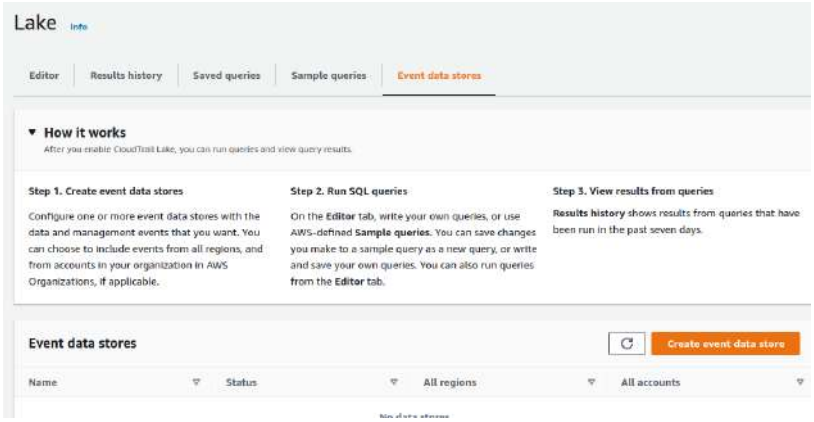
En el bucket crea un esqueleto de carpetas para guardar eventos.



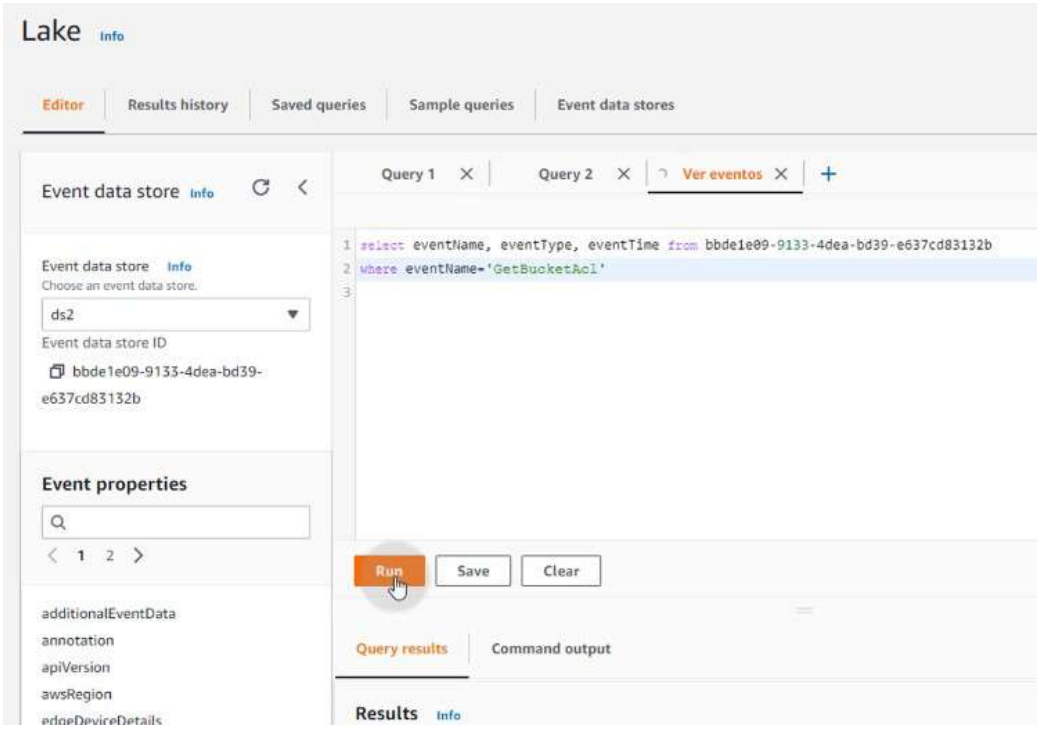
En CloudTrail-Digest se guardan ficheros de metadatos. En CloudTrail estarán lo eventos que se guardan con un esqueleto de region y fecha en json comprimidos

Lo más cómodo es enviar los logs a CloudWatch para poder visualizarlos.

Consola de **CloudTrail Lake** – Guarda los eventos en un formato que permite las SQL queries.



Consulta SQL:



Para eliminar un datastore tendremos que quitar la protección de borrado. Luego ya podremos borrar pero hasta que no pasen los días de retención que hayamos puesto no desaparecerá, eso sí, dejará de guardar datos.

26.1. - AWS CLI

El subcomando utilizado es

aws cloudtrail

Por ejemplo, listar la trails

aws cloudtrail list-trails

o para ver propiedades

aws cloudtrails describe-trails –trail-name-list nombre nombre-bucket

Crear un trail en un bucket existente. Necesitaremos una política y para conseguirla rápido, si ya tenemos el bucket asociado con CloudTrail podemos ir a sus permisos, copiar la política de resources y pegarla con una coma más abajo, indicando el nombre del trail nuevo. Entonces, ya podemos crearlo.

aws cloudtrail create-trail --name Nombre-Trail –s3-bucket-name nombre-bucket

Por defecto, guarda los datos de administración.

Para eliminar un trail

aws cloudtrail delete-trail --name nombre-del-trail

Las opciones del comando son:

add-tags
cancel-query
create-event-data-store
create-trail
delete-event-data-store
delete-trail
describe-query
describe-trails
get-channel
get-event-data-store
get-event-selectors
get-insight-selectors
get-query-results
get-trail
get-trail-status
list-channels
list-event-data-stores

list-public-keys
list-queries
list-tags
list-trails
lookup-events
put-event-selectors
put-insight-selectors
remove-tags
restore-event-data-store
start-logging
start-query
stop-logging
update-event-data-store
update-trail
validate-logs

TEMA 27 - Cloud9 y las SDK – Entorno de desarrollo y pruebas

Cloud9 es un entorno en cloud para desarrollar, testear y probar código. Tiene un IDE (Entorno de Desarrollo Integrado) incorporado con herramienta gráfica online que permite incorporar AWS CLI, Clientes SDK (Kit de desarrollo de software), entre otras herramientas. Además de almacenar el código en AWS y compartirlo entre otro usuarios. En definitiva, es una instancia con una plantilla instalada con todo el entorno necesario para trabajar, es muy pequeña y muy barato. Por ejemplo, una máquina para trabajar 40 horas semanales puede salir a 5 dolares al mes.

Cloud Shell es una línea de comandos que también nos permite trabajar online.

Se puede utilizar con DocumentDB (servicio de base de datos escalable, de larga duración y completamente administrado para operar cargas de trabajo de MongoDB esenciales) o con Red Shit (servicio de almacenamiento de datos en la nube).

No es recomendable utilizar usuario ROOT con cloud9

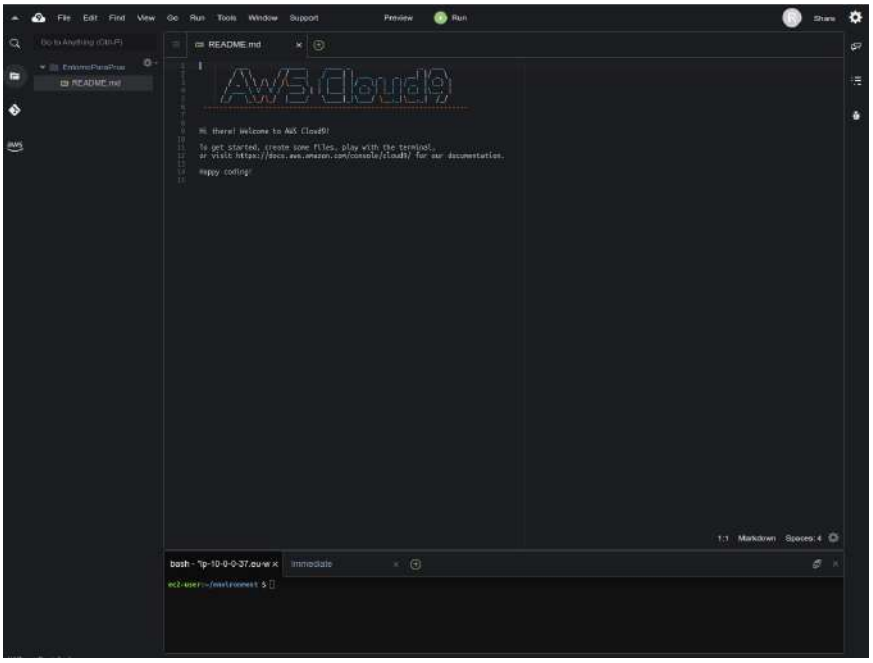
Para **crear un entorno**, entramos en el servicio y los parámetros configurables son:

- Nombre y descripción
- Tipo de entorno
  - Instancia nueva
  - De acceso con System Manager
  - En una instancia existente
- Tipo de instancia – Son instancias pequeñas.

- SO – Da 3 opciones: Amazon Linux 2, Amazon Linux AMI o Ubuntu Server 18,04 LTS
- Opciones de hibernación
- IAM role
- Network (VPC, subnet)
- Tags

Cuando lo abres con ROOT te da un aviso.

Es un IDE normal como eclipse, NetBeans o demás, pero está dentro de un entorno de AWS y con acceso directo a los servicios AWS.



En la parte izquierda tenemos varias pestañas. De archivos, de git y de navegación por AWS.

Como otros IDEs podemos abrir varias pestañas de ficheros que se controlan arriba.

En la parte de abajo hay un terminal linux y inmediate que nos permite trabajar con un entorno REPL de javascript.

En la parte de menu superior podemos acceder a todas las acciones posibles. Tenemos un preview y un run. Y más a la derecha esta nuestro usuario, el botón de compartir y una rueda dentada con las opciones del diseño de la interfaz.

En las pestañas se puede crear también terminales. El terminal está preparado como una máquina normal, pero además tiene instalado AWS CLI, Python, Java, Go, node, ruby, C, git, docker, perl, kotlin, .NET, Rust, Swift, php, sqlite, terraform, etc

SDK en AWS: https://aws.amazon.com/es/developer/tools/ Aquí podremos encontrar los SDK necesarios para trabajar en los distintos lenguajes.

En concreto, en Java, podemos utilizar una version del OpenJDK, podemos descargar Amazon Corretto.

Adaptación de Visual Studio Code para AWS: https://aws.amazon.com/es/visualstudiocode/

27.1. - Ejemplo de código python

El SDK que necesitamos es Boto3, que es una librería que nos permite acceder a todos los recursos de AWS.

Necesitamos utilizar el gestor de paquetes pip.

```
pip install boto3
```

Se puede hacer las pruebas en el modo comando de python

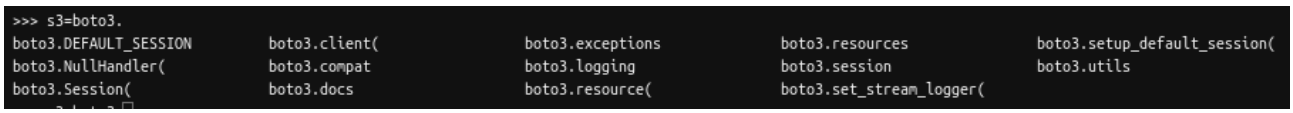
```
python
```

Importamos la librería boto3, si no hay mensaje de error la hemos descargado correctamente.

```
>>> import boto3
```

Para acceder a un recurso nos ayudaremos de una variable. Si empezamos a escribir boto3 y le damos a tabulador nos saldrán las opciones que podemos utilizar.

```
>>> s3=boto3.
```



Lo que queremos en preparar la conexión en la variable, así que elegimos el recurso

```
>>> s3=boto3.resource('s3')
```

Ahora la variable s3 es un recurso de tipo servicio y podremos lanzarle comandos.

>>> s3

```
>>> s3
s3.ServiceResource()
>>>
```

Hacemos un bucle para contabilizar los buckets (Es importante las identaciones en python, los espacios en la segunda linea del bucle.)

>>> for bucket in s3.buckets.all():

… print(bucket.name)

…

```
>>> for bucket in s3. buckets.all():
...     print(bucket.name)
...
vergara-bucket
vergara-bucket-02
>>>
```

De esta manera podemos acceder a muchas propiedades de AWS.

Si pasa un tiempo se desconecta python y tenemos que salir y volver a entrar para que todo funcione.

Ahora vamos a crear un bucket. Primero situamos en la variable, decimos la opción que queremos ejecutar y dentro del paréntesis indicamos propiedad. Es obligatorio decir el nombre del bucket e indicarle la region (se puede hacer con el formato json).

>>> s3.create_bucket(Bucket="vergara-bucket-03",CreateBucketConfiguration={'LocationConstraint':'eu-west-3'})

```
>>> s3.create_bucket(Bucket="vergara-bucket-03",CreateBucketConfiguration={'LocationConstraint':'eu-west-3'})
s3.Bucket(name='vergara-bucket-03')
>>> for bucket in s3. buckets.all():
...     print(bucket.name)
...
vergara-bucket
vergara-bucket-02
vergara-bucket-03
>>>
```

Ahora metemos en una variable el puntero al bucket

```
bucket=s3.Bucket('vergara-bucket-03')
```

Y subimos un objeto al bucket.

```
bucket.put_object(Key='fichero',Body='/home/ec2-user/environment/README.md')
```

Y ahora podemos listar los objetos que hay en el bucket

```
for objeto in s3.Bucket('vergara-bucket-03').objects.all():
...     print(objeto.key)
...
```

Podemos añadir a una variable el puntero al objeto concreto

```
v1=s3.Object('vergara-bucket-03','fichero')
```

y preguntar por su tamaño y su tipo

```
print(v1.content_length)
print(v1.content_type)
```

TEMA 28 - Bases de Datos: DocumentDB

Una versión de MongoDB en AWS. Está orientada a Cloud. Cualquier dato en MongoDB es fácil reapuntarla a DocumentDB.

Conceptos básicos:

- NOSQL (NotOnlySQL) es una evolución del sistema clásico de BBDD relacionales cuya principal característica es que no se requiere una definición inicial de las estructuras sobre las que se almacenarán los datos.
- Evolución ddel modelo entidad-relación para el soporte de estructuras variables en el tiempo.
- MongoDB es una BBDD NOSQL orientada a documento.
- Desarrollada por 10gen (Ahora llamados MongoDB)

Características

- Flexibilidad de los Modelos
- Alto rendimiento, alta disponibilidad
- Escalabilidad
- etc

El modelo de documentos se basa en el estándar JSON/BSON (BSON es una serialización binaria de JSON). Características de BSON

- Permite tener documentos dentro de documentos
- Podemos definir arrays dentro de documentos.

Comparación SQL vs MongoDB

SQL	MongoDB
Tabla	Colección
Fila	Documento
Columna	campo
Clave primaria	Id de objecto
Índice	Índice
Vista	Vista
Tabla u objeto anidado	Documento incrustado
Array	Array

Ejemplo de un documento con vista JSON

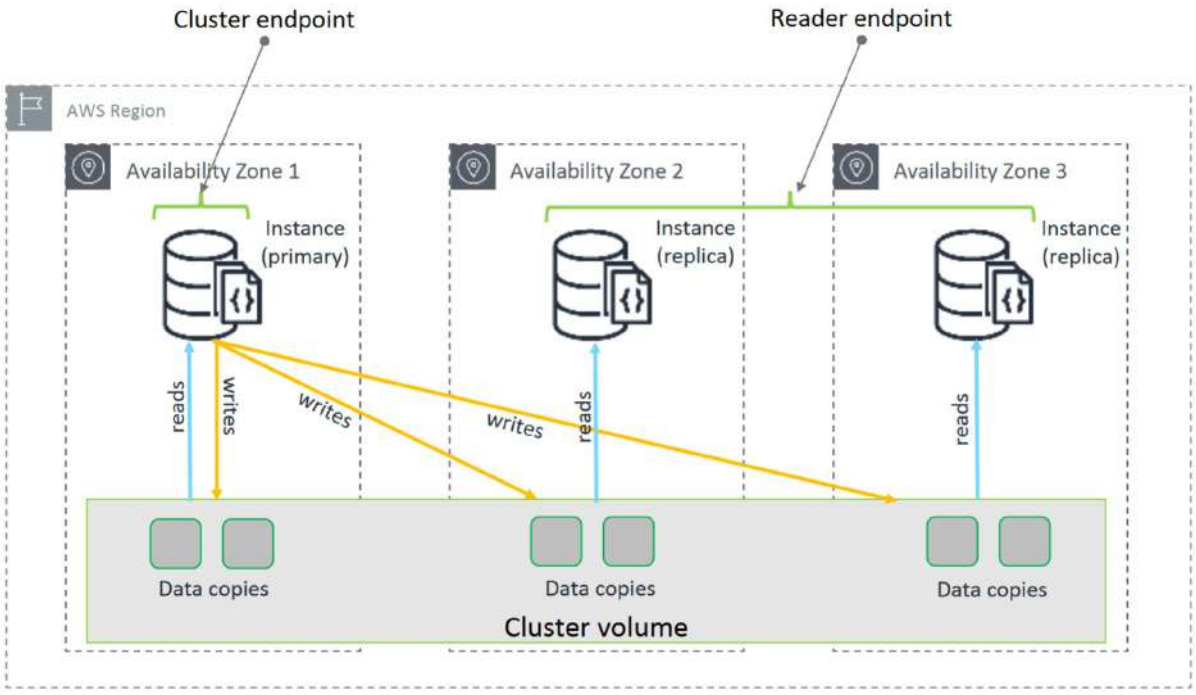
```
{
  "_id": 1,
  "name": { "first" : "John", "last" : "Backus" },
  "contribs": [ "Fortran", "ALGOL", "Backus-Naur Form", "FP" ],
  "awards": [
    {
      "award": "W.W. McDowell Award",
      "year": 1967,
      "by": "IEEE Computer Society"
    }, {
      "award": "Draper Prize",
      "year": 1993,
      "by": "National Academy of Engineering"
    }
  ]
}
```

Este documento sería una fila, se podría subir a una collection, que sería una tabla.

Cuando se crea una colection se respeta el tipo de documento que tienen dentro.

28.1. - Arquitectura

Un cluster consta de 0 a 16 instancias y un volumen de almacenamiento de cluster que gestiona los datos de esas instancias. Los datos del clúster se almacenan en el volumen del clúster con copias en tres zonas de disponibilidad diferentes.



El cluster volume es el almacenamiento global. Solo escriben datos la instancia primaria. Tanto la primaria como las réplicas leen datos.

28.2. - Consola DocumentDB

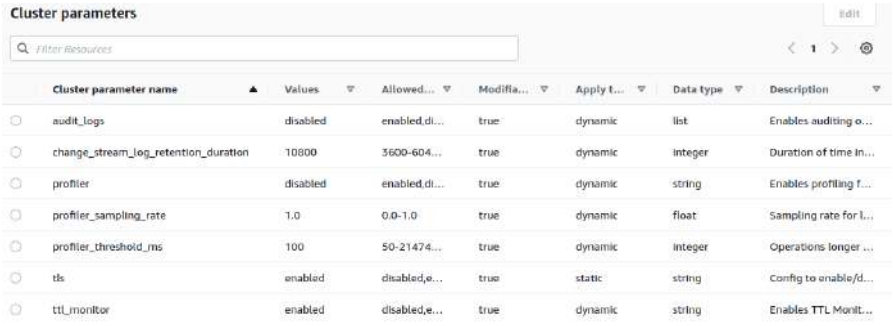
Opciones en la consola:

- Dashboard
- Clusters – Nos pedirá más o menos las mismas propiedades que en Aurora. Igual que pasa con las instancias de RDS, las instancias de DocumentDB son Plaform as a Service, es decir no se pueden administrar. Siempre van a cobrar 10 minutos como mínimo y no existe opción gratuita. Para **crear un cluster** podremos:
  - Nombre
  - Versión – Permite 2 versiones. 3.6.0 y 4.0.0
  - Instancia
  - Número de instancias – Lo mínimo ideal son 3 para tener un cluster en alta disponibilidad.
  - Autenticación – Usuario master (No puede ser una palabra reservada: admin, master, etc) y contraseña.
  - Opciones avanzadas
    - Network – VPC, grupo de subnets y Security groups

- Opciones Cluster – Puerto (27017) y Grupo de parámetros
- Encriptación – La master key por defecto es la de AWS/RDS
- Además: Backup, Performance Insights, Exportar logs, Mantenimiento, Etiquetado y Protección de borrado

Cluster identifier	Role	Engine version	Region & AZ	Status	CPU	Curre
documentdb-cluster-01	Regional c...	4.0.0	eu-west-3	creating	-	-
documentdb-cluster-01	Replica in...	4.0.0	-	creating	-	-
documentdb-cluster-012	Replica in...	4.0.0	-	creating	-	-
documentdb-cluster-013	Replica in...	4.0.0	-	creating	-	-

- Performance insights
- Snapshots
- Subnets groups - Comparte con RDS los grupos de subnets, pero siempre es mejor personalizar el grupo para cada cluster. Para **crear un grupo de subnets** podemos configurar:
  - Nombre, descripción y en añadir subnet se debe indicar la VPC, la AZ y la subnet concreta. Se debe añadir al listado cada subnet que se necesite.
- Parameter groups – Son los parámetros configurables de la BBDD. Son pocos los parámetros si lo comparamos a otros tipos de BBDD.



Cluster parameters

Edit

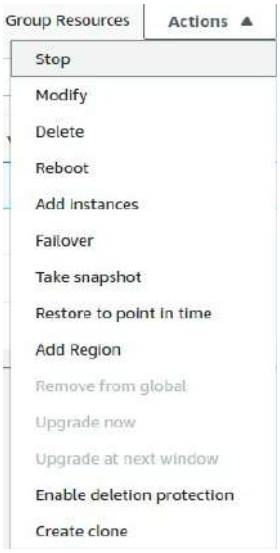
Filter Resources

1

Cluster parameter name	Values	Allowed...	Modifia...	Apply t...	Data type	Description
audit_logs	disabled	enabled,di...	true	dynamic	list	Enables auditing o...
change_stream_log_retention_duration	10800	3600-604...	true	dynamic	integer	Duration of time in...
profiler	disabled	enabled,di...	true	dynamic	string	Enables profiling f...
profiler_sampling_rate	1.0	0.0-1.0	true	dynamic	float	Sampling rate for l...
profiler_threshold_ms	100	50-21474...	true	dynamic	integer	Operations longer ...
tls	enabled	disabled,e...	true	static	string	Config to enable/d...
ttl_monitor	enabled	disabled,e...	true	dynamic	string	Enables TTL Monit...

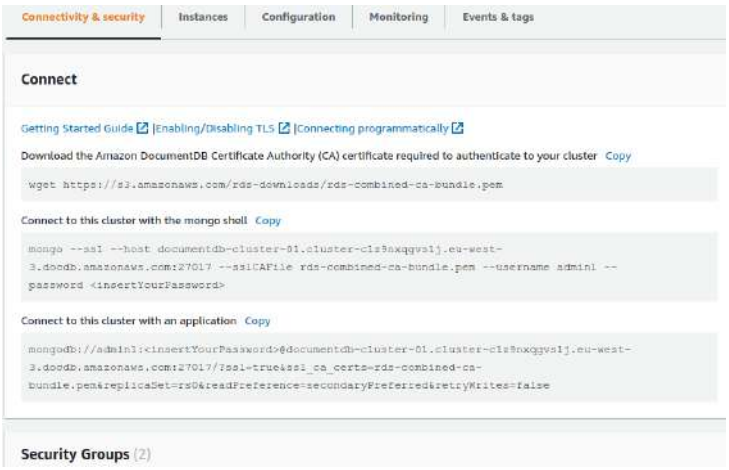
- Event Subscriptions – Nos sirve para conectar con SNS. También los comparte con RDS.
- Events – Nos dará información de arranque, parada y otros generales.

Las acciones en los cluster son muy parecidas a las de RDS



Componentes de un cluster

- Connectivity & security – Nos indica como podemos conectarnos a un cluster de este tipo
  - Mongo shell – Es una herramienta en modo comando de MongoDB. Documentación oficial: https://www.mongodb.com/docs/v4.4/mongo/
  - Aplicación -



  - Existen muchas otras herramientas para conectarse a un MongoDB: Robo 3T, Nosqlclient, Umongo, Mongotro
- Instances – Con la primaria y las réplicas que tengamos.
- Configuration – Acepta modificaciones. Indica todas las conexiones
- Monitoring
- Events & tags

En las **propiedades de las instancias** (primarias y réplicas) aparece los datos de la instancia en concreto.

28.3. - Conectar con la BBDD Mondo

Para conectarnos fácilmente podemos utilizar Cloud9 y seguir los siguientes pasos:

Crear el repositorio

```
echo -e "[mongodb-org-4.0] \nname=MongoDB Repository\nbaseurl=https://repo.mongodb.org/yum/amazon/2013.03/mongodb-org/4.0/x86_64/\ngpgcheck=1 \nenabled=1 \ngpgkey=https://www.mongodb.org/static/pgp/server-4.0.asc" | sudo tee /etc/yum.repos.d/mongodb-org-4.0.repo
```

Instalar mongo shell. Primero siempre es mejor una actualización (sudo yum update). Si pusieramos mongodb lo instalaría todo, pero podemos trabajar con solo la shell.

```
sudo yum install -y mongodb-org-shell
```

Ahora ya está descargado, pero no podemos conectarnos. Lo siguientes pasos ya están en la pestaña de connecivity del cluster.

Descargar el certificado

```
wget https://s3.amazonaws.com/rds-downloads/rds-combined-ca-bundle.pem
```

Conectar el cluster con mongo shell

```
mongo --ssl --host documentdb-cluster-01.cluster-c1z9nxqgvslj.eu-west-3.docdb.amazonaws.com:27017 --sslCAFile rds-combined-ca-bundle.pem --username admin1 --password <insertYourPassword>
```

Podemos probar **algunos comandos de MongoDB**

Comprobar las bases de datos existentes

```
show dbs
```

Creamos una nueva Base de datos denominada datos

```
use datos
```

Creamos un documento para comprobar que funciona. Hasta que no creamos un documento dentro de una base de datos, esta no se incializa

```
db.movie.insert({"name":"Ejemplo de documento"})
```

Comprobamos que está

```
show dbs
```

Comprobamos la colección que acabamos de crear

```
show collections
```

```
rs0:PRIMARY> use datos
switched to db datos
rs0:PRIMARY> show dbs
rs0:PRIMARY> db.pelicula.insert({"nombre":"Lo que el viento se llevo"})
WriteResult({ "nInserted" : 1 })
rs0:PRIMARY> show dbs
datos  0.000GB
rs0:PRIMARY> show collections
pelicula
rs0:PRIMARY>
```

Podemos comprobar que tiene la BBDD concentrador

db.pelicula.find()

```
rs0:PRIMARY> db.pelicula.find()
{ "_id" : ObjectId("6322fe05985b2824b53508a4"), "nombre" : "Lo que el viento se llevo" }
```

Podemos insertar más campos (columnas) en el documento (fila) en la misma colección (tabla)

```
rs0:PRIMARY> db.pelicula.insert({"nombre":"Terminator","Actor":"Arnold"})
WriteResult({ "nInserted" : 1 })
rs0:PRIMARY> db.pelicula.find()
{ "_id" : ObjectId("6322fe05985b2824b53508a4"), "nombre" : "Lo que el viento se llevo" }
{ "_id" : ObjectId("6322ff62985b2824b53508a5"), "nombre" : "Terminator", "Actor" : "Arnold" }
```

CUIDADO. Se pueden poner datos que no correspondan

```
rs0:PRIMARY> db.pelicula.insert({"codigo_factura":100,"Descripcion":"Ejemplo de descripcion"})
WriteResult({ "nInserted" : 1 })
rs0:PRIMARY> db.pelicula.find()
{ "_id" : ObjectId("6322fe05985b2824b53508a4"), "nombre" : "Lo que el viento se llevo" }
{ "_id" : ObjectId("6322ff62985b2824b53508a5"), "nombre" : "Terminator", "Actor" : "Arnold" }
{ "_id" : ObjectId("6322ffb0985b2824b53508a6"), "codigo_factura" : 100, "Descripcion" : "Ejemplo de descripcion" }
```

Creamos una colección manualmente, denominadas facturas

db.createCollection("facturas")

show collections

```
rs0:PRIMARY> db.createCollection('facturas')
{ "ok" : 1, "operationTime" : Timestamp(1663238143, 1) }
rs0:PRIMARY> show collections
facturas
pelicula
rs0:PRIMARY>
```

Podemos crear colecciones con límites

db.createCollection("foros", { capped : true, vi fsize : 6142800, max : 10000 })

Borramos la colección

db.facturas.drop()

show collections

```
rs0:PRIMARY> db.facturas.drop()
true
rs0:PRIMARY> show collections
pelicula
rs0:PRIMARY>
```

Podemos hacer un find con un formato JSON

db.pelicula.find().pretty()

```
rs0:PRIMARY> db.pelicula.find().pretty()
{
        "_id" : ObjectId("6322fe05985b2824b53508a4"),
        "nombre" : "Lo que el viento se llevo"
}
{
        "_id" : ObjectId("6322ff62985b2824b53508a5"),
        "nombre" : "Terminator",
        "Actor" : "Arnold"
}
{
        "_id" : ObjectId("6322ffb0985b2824b53508a6"),
        "codigo_factura" : 100,
        "Descripcion" : "Ejemplo de descripcion"
}
```

Podemos insertar varios documentos

```
db.facturas.insert([
{
        title: 'Servidores',
        description: 'Servidores para el CPD',
        by: 'Departamento de informatica',
        url: 'https://www.ibm.com',
        tags: ['servidores, 'informatica', 'ibm'],
        likes:100
},
{
        title: 'Boligrafos',
        description: 'Compra de bolis',
        by: 'Pepe',
        tags: ['servidores', 'informatica', 'ibm'],
        comments:[
        {
                precio: 5,
                descuento: 0,
                pagado: 'con visas'
        }
        ]
}
])
```

Insertar mediante script cargar.js

```
db.articulos.drop()
for(i = 1; i <= 10; i++) {
db.articulos.insertOne(
{
_id: i,
nombre: 'nombre'+i
}
)
}
```

Y lo insertamos mediante el comando load

```
load('cargar.js')
```

Comprobar que se ha cargado

```
db.articulos.find()
```

Actualizar la fila numero 1

```
db.articulos.update({"_id":1},{$set:{"nombre":"PEPITO"}})
```

Borrar fila con nombre3

```
db.articulos.remove({"nombre":"nombre3"})
```

Contar documentos de una colección

```
b.articulos.count()
```

Buscar documentos

```
db.articulos.find()
```

Sin formato json

```
db.articulos.find().pretty()
```

Buscar documentos con query clave/valor

```
db.facturas.find({"by":"Pepe"}).pretty()
```

o

```
b.facturas.find({"tags":"informatica"}).pretty()
```

Buscar documentos con precio menor de 6

```
db.facturas.find({"comments.precio":{$lt:6}}).pretty()
```

Añadir campo

```
db.facturas.update({"title":"Servidores"},{$set:{"precio":10}})
```

Borramos documento

```
db.facturas.remove({"title":"cables"})
```

Visualizar solo un campo de cada documento

```
db.facturas.find({},{"title":1})
```

Visualizar solo un campo de cada documento sin el id

```
db.facturas.find({},{"title":1,_id:0})
```

Visualizar solo un campo de cada documento sin el id y solo los primeros dos documentos

```
db.facturas.find({},{"title":1,_id:0}).limit(2)
```

Visualizar ordenados por nombre

```
db.facturas.find({"title":{$ne:""}},{"title":1,_id:0}).sort({"name":1})
```

Crear un indice sobre title

```
db.ejemplo.find({"title":{$ne:""}},{"title":1,_id:0}).sort({"title":1})
```

28.4. - Gestión de instancias y cluster

La gestión es identica a la de Aurora en RDS.

TEMA 29 - CloudFormation. Plantillas para nuestra infraestructura

No tiene nada que ver con la formación. Es una herramienta que nos permite modalizar y provisional las estructuras en AWS.

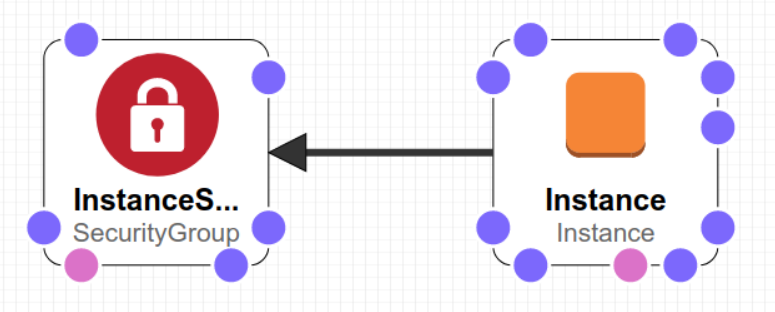
Se utiliza para repetir la misma estructura con una plantilla de la misma, que se llaman stack. Mientras se lanza un stack se puede modificar (nombre, componentes, etc)

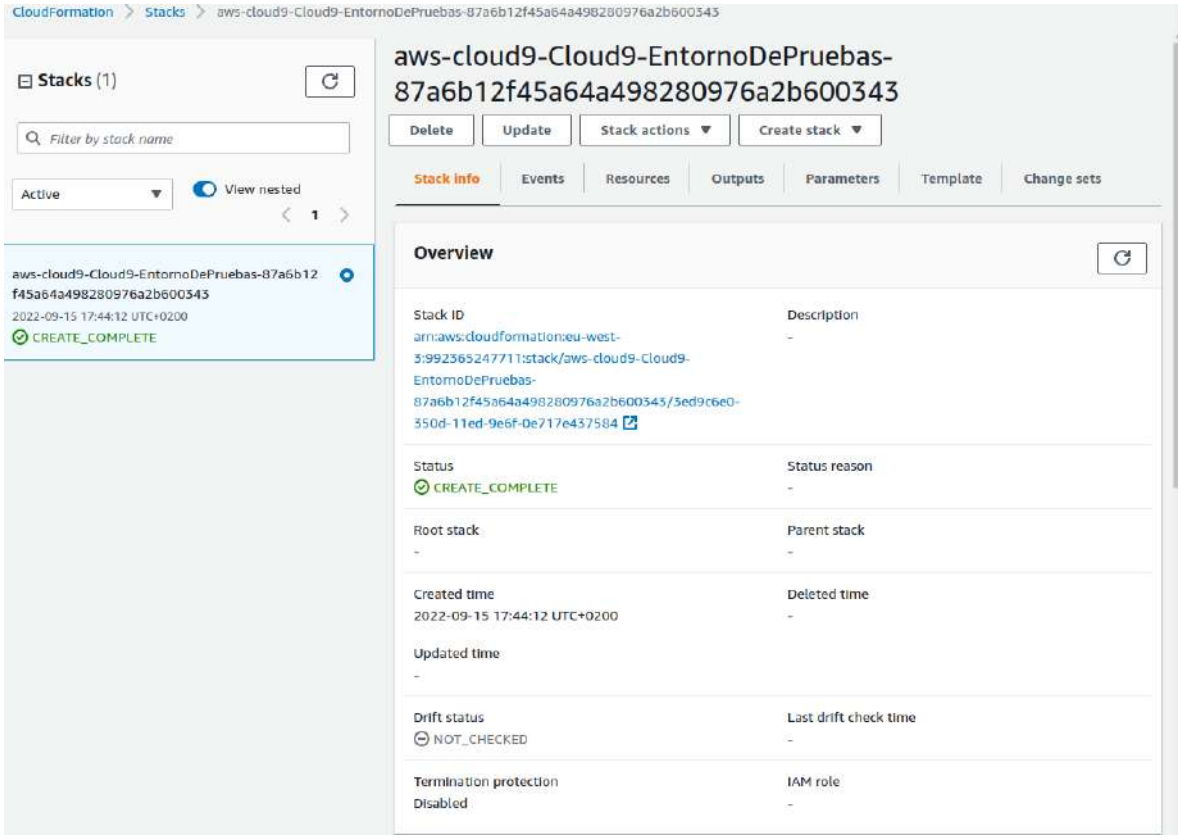
Posibilita crear infraestructuras complejas de manera muy fácil.

Hay un diseñador donde se puede dibujar la infraestructura para ahorrar ciertas conexiones.

Cuando se borra un stack se borran todos los componentes que se incluyen dentro.

Cuando se crea un entorno cloud9 automáticamente se crea el stack en CloudFormation. Lo único que hace es crear un grupo de seguridad y una instancia. Diseño del stack de cloud9**:**



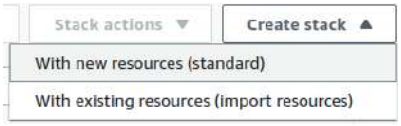


Los elementos que tiene son:

- Stack info
- Events – Los eventos por los que ha pasado el stack
- Resource – Lista de los recursos que tiene el stack
- Ouputs – Las salidas
- Parameters – Los parámetros que se piden al crear el stack
- Template – La plantilla que se ha utilizado. Es el core del stack. En un archivo JSON o YAML que indica las caracteristicas del stack.
- Change sets – Son los cambios que se hacen del stack.

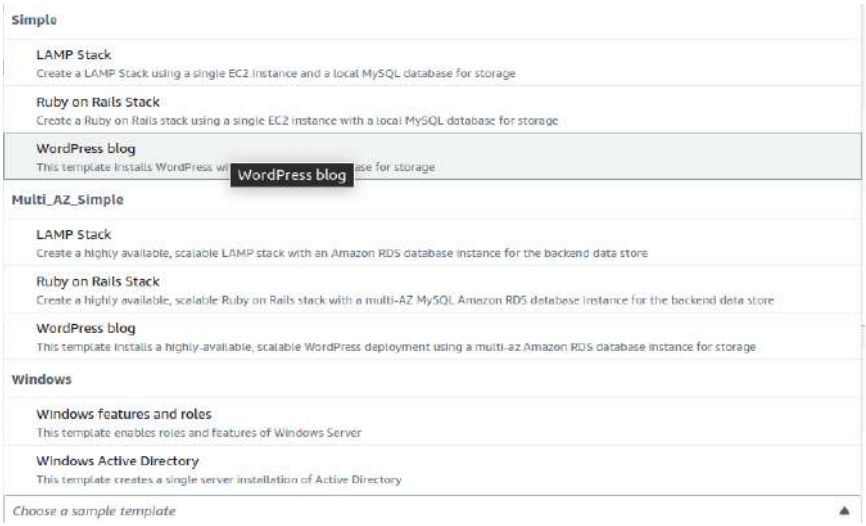
29.1. - Crear un stack

Para crear un stack se puede elegir con nuevos recursos o con recursos existentes



Para crear un stack indicamos:

- Una plantilla
  - Que hemos creado
    - Indicar Amazon S3 URL
    - Subir el archivo de plantilla
  - Las de ejemplo
    - Escoger tipo de plantilla

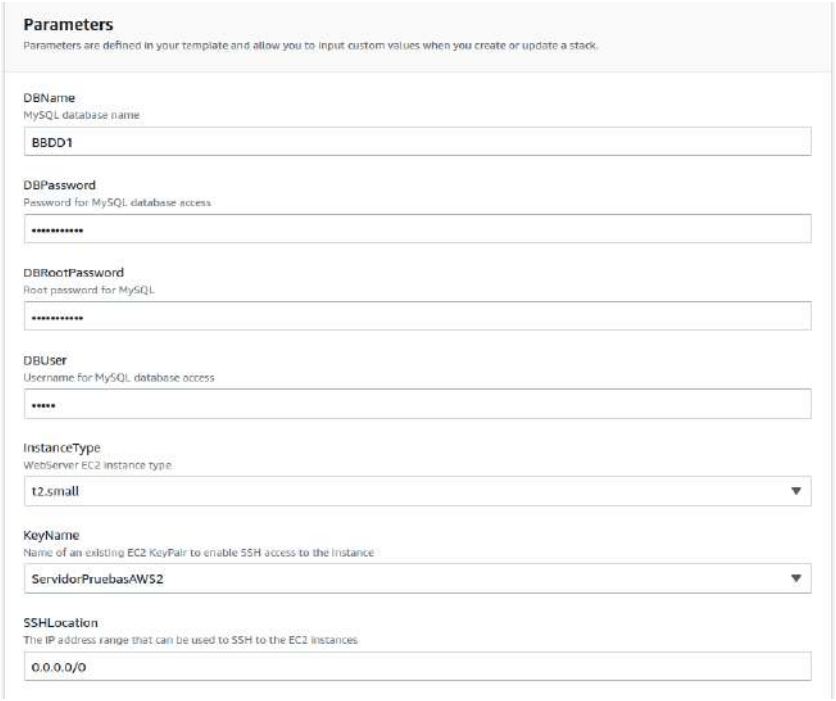


Cuando seleccionas puedes ver el diseño

  - Utilizando en la herramienta Designer.

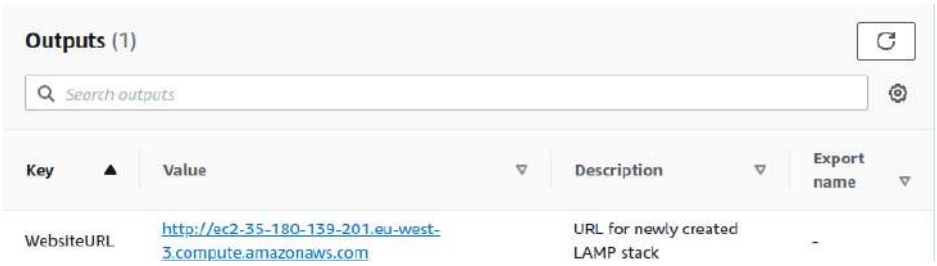
Una vez seleccionada la plantilla tendremos que dar algunos parámetros más. Por ejemplo, en el LAMP Stack simple:

- Nombre stack
- Parámetros (Depende de la plantilla pedirá unos parámetros u otros)

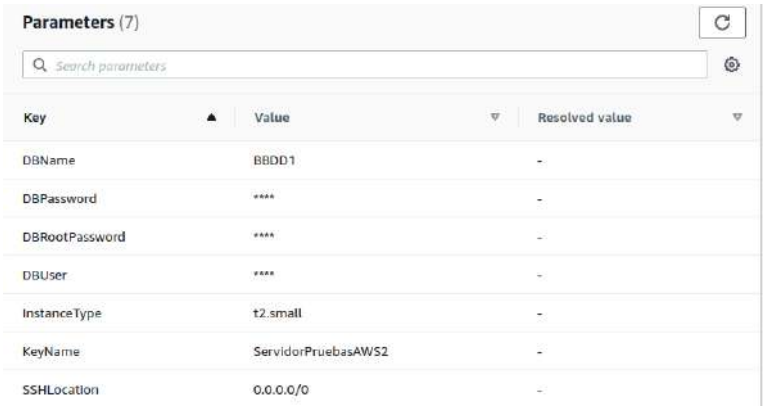


- Tags y Permisos
- Stack failure options – Comportamiento cuando falla. Puedes escoger entre rollback de todos los recursos o guarda los que se ha conseguido crear.
- Stack policy – Política de seguridad para configurar permisos.
- Rollback – A través de una alarma se efectua un rollback
- Opciones de notificación
- Opciones de creación del stack – Controlar el tiempo de creación. Protección de eliminación.

Después de crear, en los eventos se puede ver justo lo que está haciendo si vamos actualizando la lista. En este caso también ha creado un ouput, una URL.



Los parámetros que ha solicitado al crear los refleja en la pestaña de parámetros.



29.2. - Designer para crear una plantilla

Documentación tipo de propiedades y recursos: https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/aws-template-resource-type-ref.html

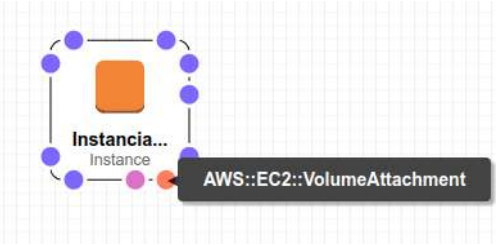
Cuando creamos un stack, sea plantilla ya creada, ejemplo de plantilla o designer. Siempre podemos ir al designer.

Componentes de designer:

- A la izquierda una lista de todos los recursos.
- En el panel central aparecen los recursos seleccionados. Si clicamos con el botón derecho salen opciones: Editar propiedades (Irá al documento inferior, al asiento del elemento), duplicar, borrar o documentación (A la ayuda de AWS).
- En la parte inferior está la plantilla que se puede ver en JSON o YAML. Además hay pestañas:
  - Plantilla completa
  - Componentes – Dentro de los componentes sale el que este seleccionado en el panel central. Según el que sea tendrá distintas pestañas superiores, por ejemplo:
    - Propiedades, Metadata, Política de borrado, Dependiente de …, Condiciones
- En la pestaña superior se puede:
  - Guardar en local o en Amazon S3.
  - Deshacer algún paso.
  - Crear el stack
  - Validar el stack – Que puede validarlo pero que tenga errores.
  - Cerrar

Ejemplo de creación:

Dentro de EC2 puedo arrastrar las instancias. En el diseño tiene una serie de bolas para añadir algunos recursos como:



- VolumeAttachment
- Grupo de seguridad
- Subnet
- Volumen
- Zona de disponibilidad
- Placement Group
- Host
- Network interface

Lo primero que hacer es configurar las propiedades de la instancia. Para ello tendremos que ir a la documentación para ver cuales son las obligatorias. En el caso de la instancia es SecurityGroupIds e ImageId

Instancia01

```
{
    "Resources": {
        "Instancia01": {
            "Type": "AWS::EC2::Instance",
            "Properties": {
                "InstanceType": "t2.nano",
                "SecurityGroupIds": "sg-0527f1d8d88d4ff0e",
                "ImageId": "ami-001e91733d45a953a"
            }
        }
    }
}
```

Para crear los parametros, se clica en un espacio vacío del centro y en componentes nos sale la lista.

Existen parámetros dinámicos para pedir valores del sistema. Se definen en la plantilla. Por ejemplo, podemos añadir en el json que pida tipos de capacidad de volumen y tipos de instancias así:

```
{
   "Parameters": {
      "DiskSize": {
         "Type": "Number",
         "Default" : 5,
         "AllowedValues":[
            5,
            10,
            15
         ],
         "Description" : "Introduce la capacidad del disco."
      },
      "maquina" : {
         "Type": "String",
         "Default" : "t2.micro",
         "Description" : "Introduce el tipo de maquina"
      }
   }
}
```

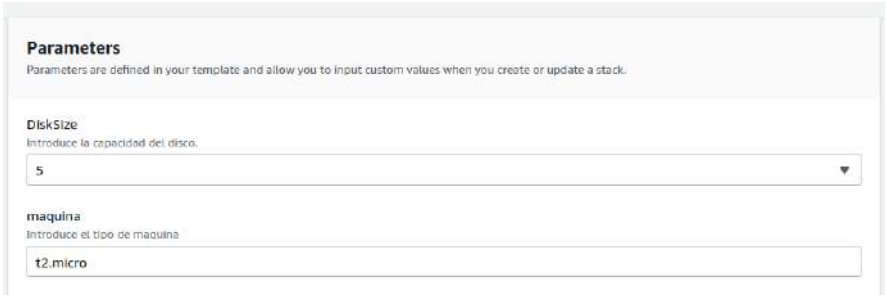
Para ponerlo en los objetos, en el lugar donde va el parámetro va “Ref” : “NombreParametro”. Por ejemplo:

```
            "Size": {
               "Ref" : "DiskSize"
            },
```

y

```
            "InstanceType": {
               "Red": "maquina"
         },
```

Cuando creamos el stack lo vemos así:



CloudFormation guarda las plantillas en buckets S3

Un Drifts pertenece a cada uno de los posibles stacks que tengamos. Cuando ejecutas un drift puede tardar un poco. Intenta detectar si se ha producido un cambio después de lanzar un stack.

TEMA 30 - Compute – Lightsail

Dentro de Compute AWS ofrece distintos servicos:

- **AWS app Runner** – Permite desplegar desde código fuente o desde un contenedor en un entorno escalable, orientado a contenedores.
- **Batch** – Procesa cargas de trabajo en modo batch dentro la nube
- **EC2** – Elastic Compute. Máquinas virtuales
- **EC2 Image Builder** – Constructor de AMIs
- **Elastic Beanstalk** – Implementar y administrar aplicaciones dentro de la nube de AWS sin tener que preocuparse por la infraestructura que las ejecuta.
- **Lambda** – Permite ejecutar código en un entorno severless.
- **Ligthsail** – Entorno destinado a usuarios con menos nivel técnico donde pueden crear instancias, contenedores, BBDD, redes, etc
- **Outpost** – Permite ejecutar servicios de AWS en entornos on-premise.

Ligthsail tiene 6 tipos de creaciones de componentes de manera rápida:

- **Instances**
  - Podemos seleccionar la zona.
  - Blueprint son entornos preparados con aplicaciones para poderlos crear una instancia rápida con ellos. (Wordpress, LAMP, node.js, Joomla, etc). El principal preparador de estas instancias es Bitnami, una empresa que se dedica a las librerías de instaladores o paquetes de software para aplicaciones web y stacks de software.
  - Se puede añadir un script de lanzamiento.
  - Se puede añadir un SSH key pair. Se puede usar por defecto o subir una clave.
  - Se pueden automatizar snapshots (Backups)
  - Se escoge el plan económico.
  - Se pueden añadir tags sin valor o con valor.
- **Containers –** Podemos crear nuestros contenedores Docker, como si fuese ECS (Elastic Containers Services) o EKS (Elastic Kubernetes Services) pero con una gestión simple.
  - Escogemos la region para trabajar.
  - Escogemos las características
  - Cuantos nodos queremos

    - Podemos escoger la aplicación a desplegar en los contenedores. Ofrece Hello World, Nginx y Redis, pero también podemos especificarla de docker hub. Aquí también se especifican variables, puertos, etc

- **Databases –** La creación es bastante parecida a la opción “easy” de RDS.
    - Podemos escoger entre MySQL y PostgreSQL
    - Definimos las credenciales de acceso y el nombre de la BBDD maestra.
    - Nos preguntar si queremos standard o alta disponibilidad.
    - Los distintos planes de precios.
    - Podemos identificar la BBDD y crear tags.
- **Networking** – Se puede crear
    - **IP estática**. Es una elastic IP.
    - **Balanceador de carga** de HTTP y HTTPS
    - **CDN** (Content Delivery Netowrk), que se puede crear con cloudFront. Lo que hace básicamente es acercar los datos y los componentes a local (Pequeños CPD que tiene amazon cerca de ciudades). Con esto se mejora la latencia. Se escoge el origen, la cache, la distribución y la identificación de la distribución.
    - **Zona DNS –** Con un dominio propio registrado se puede crear nuestra zona. En Amazon es Route 53.
- **Storage** – Podemos crear
    - Buckets
    - Discos
- **Snapshots**

Cuando creemos estos componentes tendremos unos paneles simples para poder gestionarlos, ver métricas, conectarnos a ellos, etc

En los contenedores se pueden subir imagenes personalizadas de docker desde local usando AWS CLI. Hay que tener en cuenta que debemos tener un contenedor creado en Lightsail, debemos tener la imagen construida en docker local (A través de Dockerfile o como se quiera) y necesitamos el plugin “Lightsail control”. El comando para subir la imagen:

aws lightsail push-container-image --region <nombre-region> --service-name <nombre-contenedor-en-lightsail --label <nombre-nuevo-contenedor --image <nombre-imagen-docker>

Esto hace un push en el entorno de contenedores de LightSail para tener la imagen en LigthSail.

TEMA 31 - Compute – Elastic Beanstalk

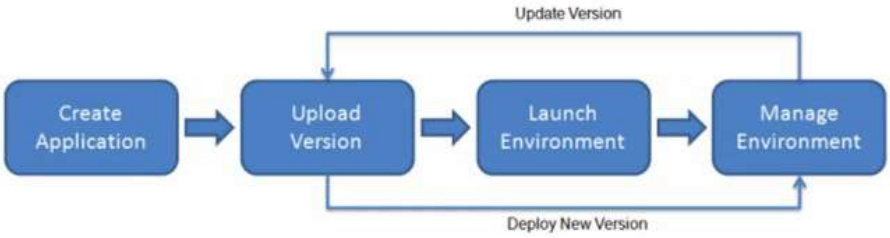
Permite implementar y administrar aplicaciones dentro de la nube de AWS sin tener que preocuparse por la infraestructura que las ejecuta. Se pueden desplegar en python, en node, en java, en go, en .NET, en php y ruby… sin indicarle si necesita un balanceador de carga, un grupo de seguridad… A través de un asistente creará un entorno apropiado a la aplicación.

Reduce la complejidad de la administración sin restringir la libertad de elección ni el control.

Solo tiene que cargar la aplicación y Beanstalk gestionará de manera automática los detalles de aprovisionamiento de capacidad, balanceo de carga, escalado y monitorización de estado de la aplicación.

Los componentes son accesibles para poder modificarlos.

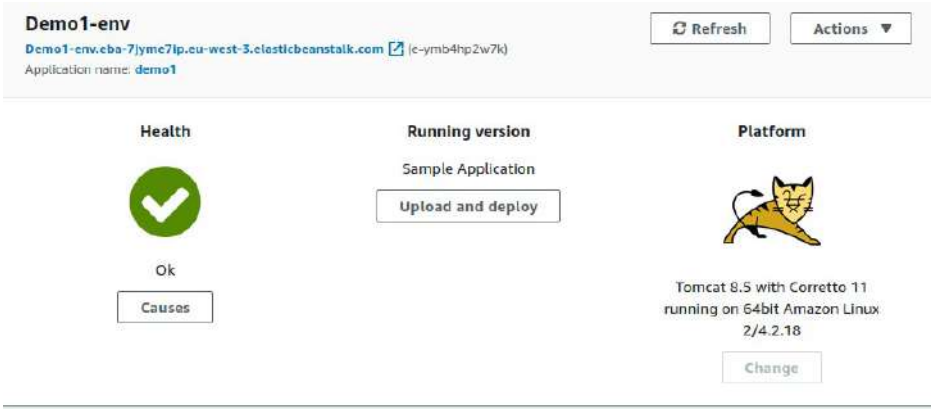
Cuando se implementa la aplicación se crea la versión de la plataforma compatible seleccionada y aprovisiona uno o varios recursos de AWS, como instancias EC2.



Amazon tiene una serie de demos para probar este servicio. Para crearlo podemos detallar las siguientes configuración:

- Nombre y tags
- Plataforma, rama y versión. (.NET, Docker, Go, Java, Node, PHP, Python, Ruby o Tomcat)
- El código de la aplicación. Se puede elegir una aplicación de ejemplo.

Crea un environments para la aplicación. Por ejemplo, en Tomcat crea un bucket, un grupo de seguridad, un target group, crea una instancia en EC2, un grupo de autoescalado, alarmas CloudWatch y un balanceador de carga.



En el sidebar izquierdo podemos ver:

- Ir al environment
- Configuración – Se pueden ver los componentes y editarlos.
- Logs
- Health
- Monitorización
- Alarmas
- Gestionar actualizaciones
- Eventos
- Tags

En la creación, si escogemos subir nuestro propio código, podremos subirla desde local o desde un enlace publico S3. Además podemos taguear la aplicación.

Las configuraciones avanzadas son para el entorno, donde podemos parametrar los siguientes componentes:

- Preajustes (Instancia simple, spot, alta disponibilidad, personalizada…)
- Plataforma Tomcat
- Software
- Instancias – Podemos activar/desactivar el IMDS, que permite acceder a los metadatos de la instancia.
- Capacidad
- Balanceador de carga
- Actualizaciones y despliegues continuos

- Políica de despliegue
  - All at once – Es el despliegue más rápido porque actualiza en todas las instancias al mismo tiempo, pero durante ese tiempo la aplicación estará parada.
  - Rolling – No habrá indisponibilidad. Irá actualizando una instancia trás otra. Se puede indicar porcentaje de máquinas en cada actualización o el número de máquinas que actualiza a la vez
  - Rolling with additional batch - Tarda un poco más en acabar la actualización que el anterior, porque creará nueva instancias para poder atender las peticiones de los usuarios. Se podrá mantener el ancho de banda.
  - Immutable – No actualiza instancias, crea nuevas y vuelve a desplegar la aplicación. Es el que más tarda.

Políticas de implementación admitidas

Política de implementación	Entornos con balanceo de carga	Entornos de una sola instancia
Todo a la vez	Sí	Sí
Continua	Sí	No
Continua con un lote adicional	Sí	No
Inmutable	Sí	Sí
División de tráfico	Sí (Balanceador de carga de aplicaciones).	No

- Seguridad
- Monitorización
- Gestión de actualizaciones
- Notificaciones
- Red
- BBDD
- Etiquetas

Una vez creado un entorno, podemos pasarlo de single-instance a multi-instance para tener alta disponibilidad. Se haría en capacidad, donde indicamos que queremos balanceo de carga.

Para actualización la aplicación a una nueva versión, entramos dentro de las versiones de la aplicación y clicamos en upload. Una vez lo tenemos subido lo seleccionamos y en “Actions” desplegamos clicando en “Deploy”. Nos dará a elegir el entorno.

El botón de Actions, nos permite en aplicaciones:

- Crear entorno
- Borrar aplicación.

- Ver las versiones
- Salvar la configuración

En los entornos nos permite:

- Salvar la configuración
- Cargar configuraciones guardadas
- Swap enviroment URLs
- Clonar el entorno
- Reconstruir el entorno
- Reiniciar el servidor de la aplicación.
- Terminar con el entorno

Cabe recordar que se puede entrar en la instancia EC2 de manera normal.

TEMA 32 - Compute AWS Lambda

Nos permite a trabajar en entorno Serverless.

Ejecuta el código en una infraestructura informática de alta disponibilidad y realiza todas las tareas de administración de los recursos informáticos, incluido el mantenimiento del servidor y del sistema operativo, el aprovisionamiento de capacidad y el escalado automático, así como la monitorización del código y las funciones de registro.

Se puede ejecutar código para prácticamente cualquier tipo de aplicación o servicio de backend.

Está basado en eventos.

El código se ejecuta basado en respuesta a eventos de otros servicios de Amazon como por ejemplo S3, DinamoDB, peticiones HTTP, etc

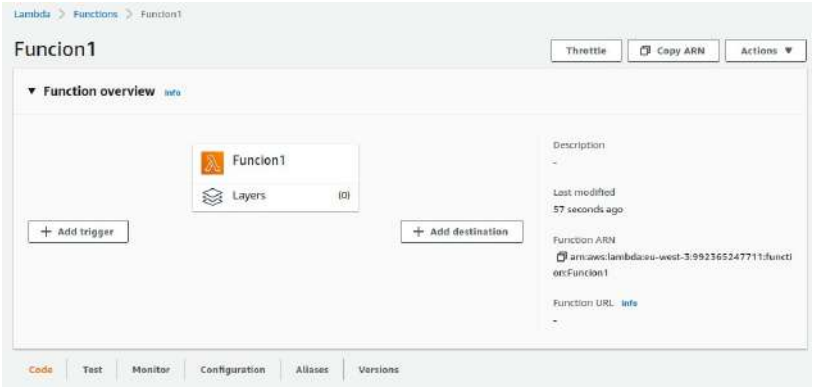
Solo tenemos que poner nuestro código dentro del servicio Lambda, configurar el trigger y esperar a que se reciban los eventos que disparan el trigger.

Los lenguajes que admite Lambda son: NodeJS, t Java, Python, C#, go , etc...

Documentación: https://docs.aws.amazon.com/es_es/lambda/latest/dg/welcome.html?icmpid=docs_lambda_help

Si creamos una función Lambda podemos parametrizar:

- 4 opciones
  - Desde scratch. Una prueba con un hola mundo
    - Nombre
    - Se selecciona el lenguaje
    - Roles de ejecución
    - Opciones avanzadas: Firma del código, URL, eituqetas y VPC
  - Con blueprint, que es una plantilla
  - Desde una imagen de contenedor
  - Desde un repositorio de app serverless



Una vez creada, tendremos un panel de control con el trigger, la función y el destino.

Con la función tenemos Layers que nos permite implementar librerías, dependencia, etc..

En la parte de abajo veremos las pestañas:

- Code – El código fuente. Aparece en un editor parecido a Cloud9.
- Test – Podemos enviar un evento para probar si funciona correctamente. Crear uno o utilizar alguno guardado.
- Monitor – Podemos ver Métricas, logs y trazas.
- Configuration – Aquí tenemos toda la configuración.
    - Configuración general
    - Triggers
    - Permisos
    - Destinos
    - URL
    - Variable de entorno
    - Etiquetas
    - VPC
    - Monitorización y herramientas de operaciones
    - Concurrencia
    - Invocación asíncrona
    - Firma de código
    - Proxy BBDD
    - Sistema de ficheros
    - Estado de las máquinas

- Aliases – Nombre alternativos que podemos darle a la Lambda y que pueden apuntar a determinadas versiones.
- Versiones

El código de prueba en node de hola mundo desde scratch:

```
exports.handler = async (event) => {
    // TODO implement
    const response = {
        statusCode: 200,
        body: JSON.stringify('Hello from Lambda!'),
    };
    return response;
};
```

Export.handler es la función que inicia todo el proceso del Lambda cuando recibe el evento.

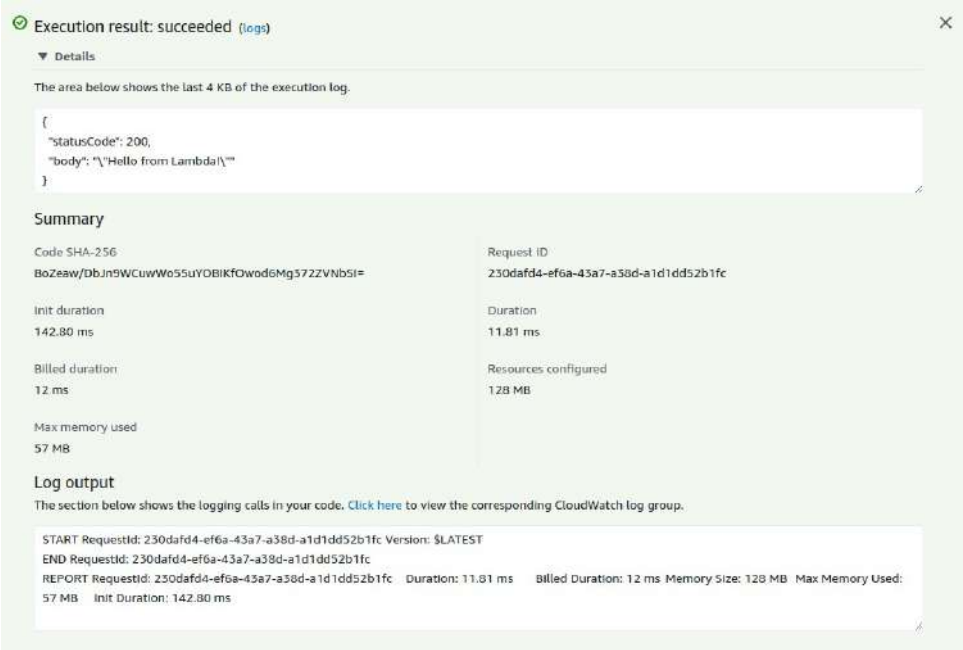
Se crea una constante llamada response con dos campos: StatusCode entero y body con un print de Hola mundo.

Por último devuleve el componente

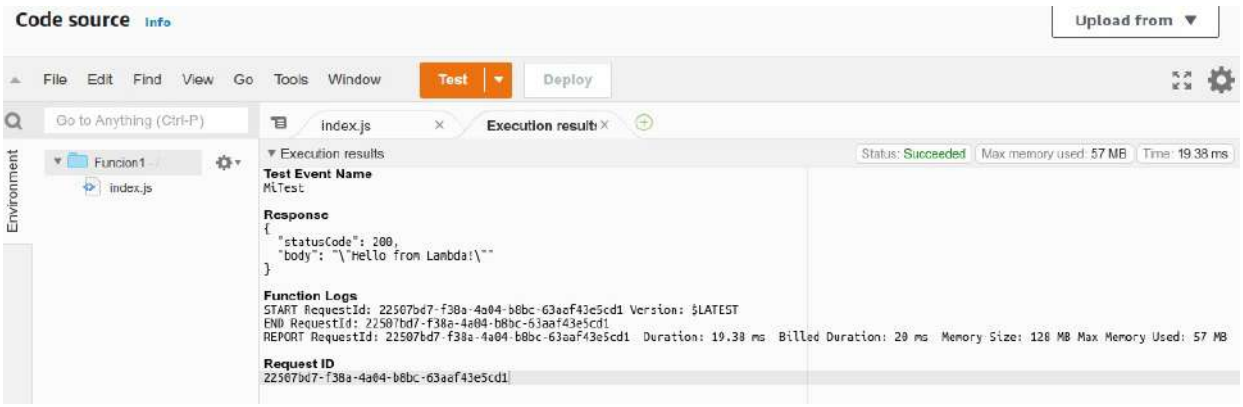
El código de test por defecto:

```
{
  "key1": "value1",
  "key2": "value2",
  "key3": "value3"
}
```

En este caso no testea nada de la aplicación porque no da ningún valor que interprete. En todo caso si aplicamos el test la aplicación retorna su constante. Pantalla del testeo desde pestaña “Test”



Resultado del testeo desde pestaña “Code”:



El código de prueba en python de hola mundo desde scratch:

```
import json
def lambda_handler(event, context):
    # TODO implement
    return {
        'statusCode': 200,
        'body': json.dumps('Hello from Lambda!')
    }
```

32.1. - Conceptos

- **Handler** – Es el fichero main, el que se ejecutará el primero cuando se reciba el evento. Desde este fichero se iniciarán el resto.

- **Evento** – Son los parámetros de la función, los que se tratarán.
- Las funciones Lambda pueden ser invocadas por eventos externos o por internos como SNS, S3, CloudFormation, etc. En el segundo caso los eventos son específicos, los parámetros que lanzan según el servicio se tendrán que consultar en la documentación: https://docs.aws.amazon.com/lambda/latest/dg/lambda-services.html
- **Contexto** – Es el entorno donde se lanza la versión Lambda. Por ejemplo, tenemos en la documentación el contexto de cuando se trabaja con Python: https://docs.aws.amazon.com/lambda/latest/dg/python-context.html

Tanto el evento como el contexto se pueden utilizar para realizar procesos en la función.

32.2. - Probando blueprint

Como ejemplo usaremos la función “Get S3 object”.:

- Le ponemos nombre, creamos rol y plantilla de políticas.
- Seleccionamos bucket (Tiene que estar en la misma region que la función), tipo de evento, prefijo, sufijo...

El código es el siguiente:

```
import json
import urllib.parse
import boto3

print('Loading function')

s3 = boto3.client('s3')

def lambda_handler(event, context):
    #print("Received event: " + json.dumps(event, indent=2))

    # Get the object from the event and show its content type
    bucket = event['Records'][0]['s3']['bucket']['name']
    key = urllib.parse.unquote_plus(event['Records'][0]['s3']['object']['key'], encoding='utf-8')
    try:
        response = s3.get_object(Bucket=bucket, Key=key)
        print("CONTENT TYPE: " + response['ContentType'])
        return response['ContentType']
    except Exception as e:
        print(e)
        print('Error getting object {} from bucket {}. Make sure they exist and your bucket is in the same region as this function.'.format(key, bucket))
        raise e
```

Los import son los móduilos que necesita para poder trabajar: Con json, con url y con amazon.

Prepara la conexión s3 en una variable.

Dentro de la función lambda prepara en la variable bucket los datos que quiere recoletar del event: Records, la posición 0, dentro de s3, dentro de bucket, dentro del nombre. Recorre el documento json hasta el dato que le interesa.

Prepara la variable key con la url, la llibrería parse y dentro el evento

Una vez se tiene preparadas las anteriores variables, prepara la variables response con el getobject, que será lo que retoranará con el print enviado al return.

Por último hay un except por si falla lo anterior imprime un mensaje de error.

32.3. - Ejemplo de trigger

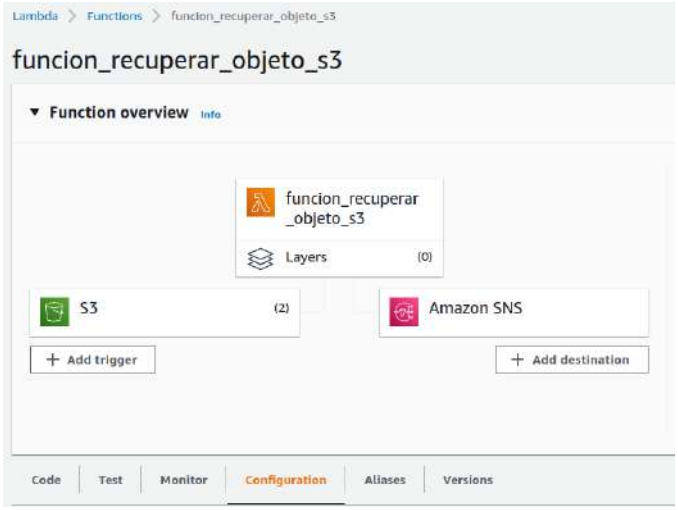
Podemos seleccionar un trigger ya preconfigurado. Si escogemos un bucket de S3 podremos ver la configuración en el bucket concreto, en el apartado de properties en la sección de “Event notifications”. Aquí también podemos crear el evento para lanzarlo contra Lambda, SNS o SQS. Los eventos contra Lambda aparecerán también en el panel de functions de Lambda.

32.4. - Ejemplo de destino

Nos permite mandar a otro recurso cierta información. Cuando creamos un destino podemos escoger entre:

- Asincrono (On failure(En caso de fallo) o On succes (Encaso de éxito)) o stream (DynamoDB stream)
- Tipo de destino entre SNS, SQS Lambda o EventBridge event bus (Gestor de eventos de AWS).
- El destino

Ahora, el panel de funciones queda así:



Tiene 2 triggers que recoge la creación y eliminación de objetos en un bucket S3. La función recupera información de los triggers para mostrarla y el destino es un SNS con un correo electrónico.

32.5. - Ejemplo con todos los pasos para crear una función, trigger y destino

- Creamos la función con “Author from scratch”, ponemos nombre, escogemos python y decidimos los permisos (Creamos un rol con permisos de lectura en S3).
- Vamos a la parte del código para sustituirlo con lo que queramos que haga la función y le damos a deploy. El código que utilizo es el siguiente:

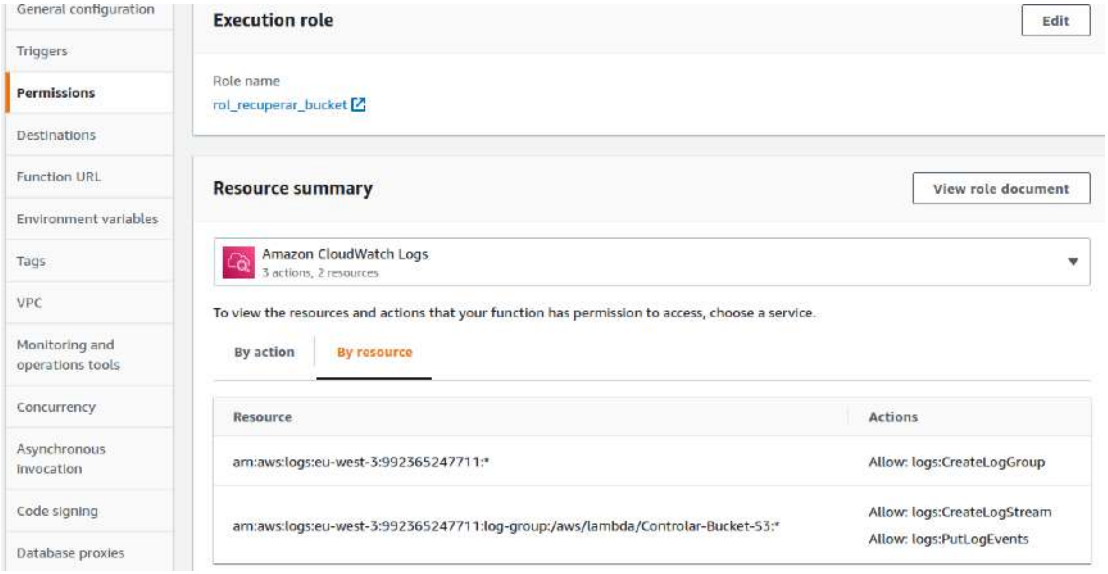
```
import json
import urllib.parse
import boto3

print('Loading function')

s3 = boto3.client('s3')

def lambda_handler(event, context):
    #print("Received event: " + json.dumps(event, indent=2))

    # Recuperar el objeto y ver su content type
    bucket = event['Records'][0]['s3']['bucket']['name']
    key = urllib.parse.unquote_plus(event['Records'][0]['s3']['object']['key'], encoding='utf-8')
    try:
        respuesta = s3.get_object(Bucket=bucket, Key=key)
        print("Nombre del objeto:"+key)
        print("Tamaño: " + str(respuesta['ContentLength']))
        return "El objeto "+key+ "tiene un tamaño de -->"+str(respuesta['ContentLength'])+" Bytes"
    except Exception as e:
        print(e)
```

print('Error al recuperar el objeto {} del bucket {}. Comprueba que existe y que se encuentra en la misma region.'.format(key, bucket))
raise e

- Añadimos un trigger, que se disparará cuando se cree un objeto en s3.
- Creamos el topic de las Lambda. Añadimos dos destinos para que mande la información en caso de fallo y otra en caso de éxito, en este caso al mismo SNS.

Si añadimos objetos a nuestro bucket enviará un SNS al correo configurado con el mensaje de la función.

Podemos revisar en “Monitor” las métricas, los logs y las trazas. En el caso de los logs, también podemos verlas en CloudWatch. Si nos vamos a Configuración y entramos en permisos podremos ver que CloudWatch tiene permisos para coger logs (Stream y events) para ponerlos en grupos de logs que va creando.



Podemos ver el role de permisos, que estará el que configuramos expresamente y los que va añadiendo Lambda por ser necesarios.

Si vamos a CloudWatch veremos en el grupo de logs los streams que se han generado por cada evento.

32.6. - AWS Step Functions

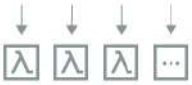
Es un servicio de orquestación que le permite conectar sus funciones de Lambda en flujos de trabajo sin servidor, denominados máquinas de estado. Cree flujos de trabajo de larga duración para la automatización de TI y los casos de uso de aprobación humana, o cree flujos de trabajo de gran volumen y corta duración para el procesamiento e ingestión de datos en streaming.

Chaining

Connect functions into a series of steps, with the output of one step providing the input to the next step.

Catch and retry

Handle errors using sophisticated catch-and-retry functionality.



Parallelism

Run functions in parallel, or use dynamic parallelism to invoke a function for every member of any array.

Branching

Design your workflow to choose different branches based on Lambda function output.

TEMA 33 - SQS – Servicio de mensajes de Amazon

Amazon SQS proporciona colas para la mensajería de alto rendimiento entre sistemas. Puede utilizar las colas para desacoplar procesos pesados y para almacenar en búferes y por lotes el trabajo. Amazon SQS almacena los mensajes hasta que los microservicios y las aplicaciones sin servidor los procesan.

33.1. - Diferencia SNS vs SQS

Servicio de mensajería como SNS pero más orientado a aplicaciones. La diferencia es que **SNS** es un sistema distribuido publicar-suscribirse. Los mensajes son empujado a los suscriptores cuando los editores los envíen a SNS. En cambio, **SQS** esta distribuido hacer cola sistema. Los mensajes son no empujado a los receptores. Los receptores tienen que sondear o tirar mensajes a SQS (pull). Varios receptores no pueden recibir mensajes al mismo tiempo. Cualquier receptor puede recibir un mensaje, procesarlo y eliminarlo. Otros receptores no reciben el mismo mensaje más tarde. El sondeo introduce inherentemente cierta latencia en la entrega de mensajes en SQS, a diferencia de SNS, donde los mensajes se envían inmediatamente a los suscriptores. SNS admite varios puntos finales como correo electrónico, SMS, punto final HTTP y SQS. Si desea que un número y tipo de suscriptores desconocidos reciban mensajes, necesita SNS.

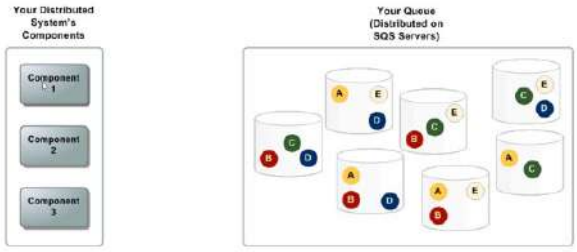
SQS es un sistema distribuido, un receptor recibe el mensaje, lo procesa y lo elimina. No puede haber más recpetores.

Es un concepto muy utilizado, los primeros sistemas de mensajes eran mainfraim de IBM con IBMMQ. En el mundo Java se llaman JNS. Permiten que dos aplicaciones que esten desacopladas puedan compartir información mediante la cola de mensajes del servidor.

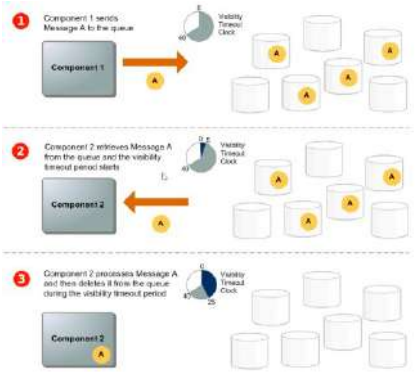
33.2. - Tipos de colas de mensajes

- Estandar
  - Tienen un nivel de procesamiento ilimitada.
  - Los mensajes se entregan al menos una vez.
  - El orden de entrega puede ser distinto al del envío. La aplicación tiene que ser consciente de este hecho y tratar el mensaje de forma coherente.
- FIFO – First In First Out
  - Se admiten hasta 300 mensajes por segundo y agrupando se pueden conseguir hasta 3000 mensajes por segundo

- Los mensajes solo se mandan una vez hasta que se procesan y eliminan. No se introducen duplicados
- El orden de entrega es igual al del envío.



El modo de tratar los mensajes SQS es de alta disponibilidad en las colas estándar. Hace duplicados en distintos servidores para asegurar su entrega.



El ciclo de vida de las colas estándar es:

1. Un componente envía un mensaje a la cola de SQS, se reparte entre servidores.
2. Otro componente pide el mensaje con un pull y empieza lo que se llama el VTC (Visibility Timeout Clock). Durante cierto tiempo el mensaje estará oculto mientras que el componente 2 no lo proceso.
3. Cuando el componente 2 procesa el mensaje se elimina de los servidores.

Esta es la manera de trabajar para que el mensaje se envíe solo una vez.

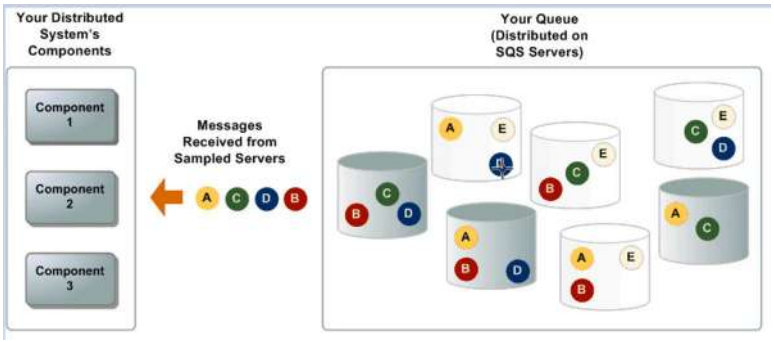
En las colas FIFO solo se envía un mensaje.

33.3. - Crear cola standard

Cuando creamos una cola standard podemos configurar:

- Visibility Timeout – El tiempo que está invisible mientras se procesa el mensaje.
- Periodo de retención del mensaje – Los mensajes que no se reclaman se eliminan después de este tiempo.
- Delevery delay – Cuanto espera un mensaje que llega a la cola en estar disponible.
- Capacidad máxima del mensaje.

- Receive message wait time – Determina si hacemos short o long polling (Sondeo corto o largo). Con Short polling se recoge un conjunto de servidores y se devuelve, no mira todos los servidores. Utiliza un proceso aleatorio, con lo que puede quedarse alguno fuera. Si ponemos más de 0 segundos será Long Polling se preguntará a todos los servidores de manera que se devolverán todos los mensajes.



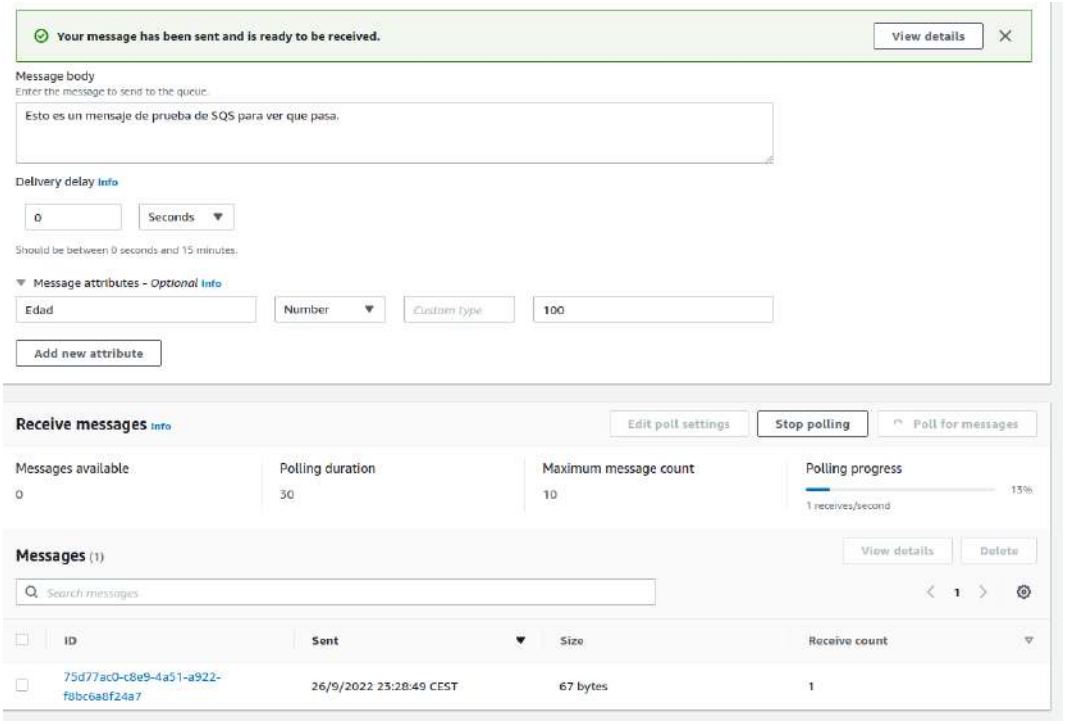
En este ejemplo de short polling vemos que no recoge todos los mensajes, le falta E que lo recogerá en un segundo polling. Short polling es razonable con menos de 1000 mensajes.

- La política de acceso – En basico nos da las opciones de quien puede enviar mensajes y quien puede enviarlos.

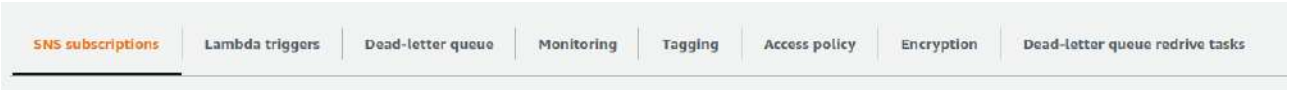
Una vez creada la cola podemos probarla en el botón “Send and recive messages”, indicando:

- Cuerpo del mensaje
- Delivery delay
- Atributos – Podrían ser utilizados para encontrar más información o para dar un valor que se necesite.

Se podrá testear si se envía y también si se puede recoger el mensaje. Una vez recibido el mensaje veremos el que se ha enviado exitosamente.



En las colas tendremos distintas pestañas con componentes



- Suscripciones SNS – Se puede seleccionar un topic para recibir los mensajes.
- Triggers Lambda – Tan solo funcionan con colas standard
- Cola de mensajes muertos – Es una cola donde se pueden enviar los mensajes que no se han procesado correctamente o se hayan rechazado.
- Monitorización
- Etiquetas
- Política de acceso
- Encriptación
- Tareas de colas de mensajes muertos

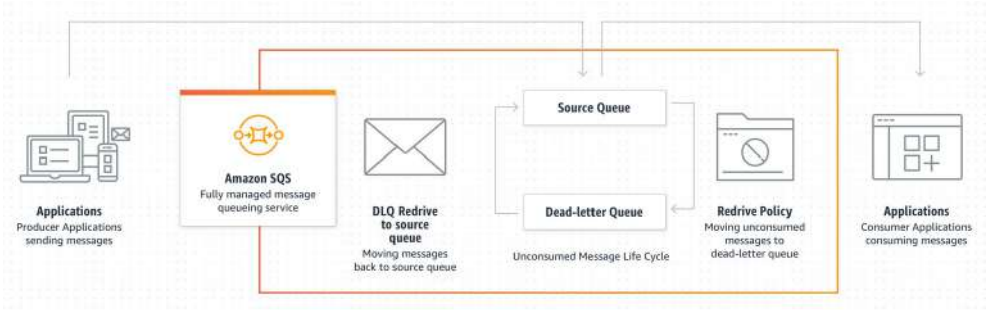
33.4. - Crear cola FIFO

Cuando creamos una cola FIFO además de las propiedades standard podemos configurar:

- Content-based deduplication – En cada mensaje le dará un hash para controlar que no hayan repetidos. Si no utilizamos esta opción, el productor del mensaje tendrá que dar a cada mensaje un messageDeduplicationId para poder identificar inequivocamente cada mensaje.
- Enable high throughput FIFO – Si lo habilitamos el scope y el limite será por grupo de mensajes.
    - Deduplication scope
        - Por cola
        - por Grupo de mensajes
    - FIFO throughput limit
        - Por cola
        - Por ID de grupo de mensajes.

La deduplicación tiene un tiempo de 5 minutos de control. Pasado ese tiempo deja de controlar si el mensaje ha llegado. A día de hoy este rango no se puede cambiar.

33.5. - Cola de mensajes muertos



Las DLQ nos permiten reconducir los mensajes a la cola fuente para volver a procesar.

Para esto necesitamos una cola del mismo tipo que la cola a la que queramos ponerle DLQ (Standard con standar y FIFO con FIFO). Una vez creada, nos vamos a Dead-letter queue de la cola origen y editamos para activar el Dead-letter queue. Indicamos la cola de mensajes incorrecto (La cola creada) y el número de mensajes que queremos que provoque que vaya a la cola de incorrectos.

Ahora, en la cola de mensajes incorrectos, marcamos DLQ redrive indicando que los mande a la cola original. Podríamos indicar otra cola de mensajes.

33.6. - Enviar y recibir desde python

Para enviar usamos el siguiente programa:

```
import boto3
```

```
# Crear un cliente SQS
sqs = boto3.client('sqs')

# Enviar una mensaje a una cola SQS
def mandar_mensaje(cola,cuerpo,nombre,apellidos):
 response = sqs.send_message(
    QueueUrl=cola,
    DelaySeconds=10,
    MessageAttributes={
      'nombre': {
         'DataType': 'String',
         'StringValue': nombre
      },
      'apellidos': {
         'DataType': 'String',
         'StringValue': apellidos
      }
    },
    MessageBody=cuerpo
 )
 print(response['MessageId'])
```

Entramos en python y importamos el programadores

```
import enviar
```

Luego utilizamos el método de dentro del programa añadiendo los argumentos que se deseen. El primero debe ser el enlace de la cola.

```
enviar.mandar_mensaje("https://sqs.eu-west-3.amazonaws.com/992365247711/Cola-standard","Esto es un mensaje de prueba desde python","Rosa","Rodriguez");
```

Y ya podremos comprobar que se ha enviado.

Para recibir usamos el siguiente programa:

```
import boto3

# Create SQS client
sqs = boto3.client('sqs')

def recibir_mensaje(cola):
# Recibir un mensaje de la cola
```

```
 response = sqs.receive_message(
    QueueUrl=cola,
    AttributeNames=[
        'SentTimestamp'
    ],
    MaxNumberOfMessages=1,
    MessageAttributeNames=[
        'All'
    ],
    VisibilityTimeout=0,
    WaitTimeSeconds=0
 )

 message = response['Messages'][0]
 receipt_handle = message['ReceiptHandle']

# Borrar el mensaje que hemos recibido
 sqs.delete_message(
    QueueUrl=cola,
    ReceiptHandle=receipt_handle
 )
 print('El mensaje ha sido recibido y borrado: %s' % message)
```

Este programa va a devolver tan solo un mensaje, en plano de pruebas nos sirve, pero en un entorno real no es adecuado.

Entonces, importamos en python el programa

```
import recibir
```

Utilizamos el método del programa

```
recibir.recibir_mensaje(“https://sqs.eu-west-3.amazonaws.com/992365247711/Cola-standard”);
```

TEMA 34 - EKS Elastic Kubernetes Service

Puede ser montado con instancias normales o con fairgate (Entonro sin servidor).

Como todo clúster de Kubernetes se creará un servidor maestro y el resto serán servidores workers.

Antes de crear un clúster tendremos que tener:

- Creada la red
- Roles IAM para el clúster:
  - Uno para el servidor maestro
  - Uno para los servidores workers

Necesitaremos una VPC para EKS con algunas peculiaridades deseables. Algunas opciones:

- Redes públicas y redes privadas. Normalmente 2 de cada. Deben estar en distintas AZ para mantener el alta disponibilidad.
- 3 redes públicas distribuidas en distintas AZ. Entornos de desarrollo y no críticas
- 3 redes privas distribuidas en distintas AZ. Entornos de desarrollo y no críticas

Lo interesante es despleagar plantillas de CloudFormation para estas opciones: https://docs.aws.amazon.com/eks/latest/userguide/creating-a-vpc.html

Con estas plantilla podemos crear la red fácilmente con CloudFormation. Por ejemplo, la plantilla de 3 redes públicas creará:

- La VPC
- Las subredes
- Tablas de rutas
- Internet Gateway
- Un grupo de seguridad

Para crear el **rol de IAM del maestro** iremos a roles (dentro de IAM). Seleccionamos en “AWS service”, entre los casos de uso “EKS” y EKS cluster. Por defecto, en los permisos pone “AmazonEKSClusterPolicy de tipo de gestión. Le ponemos un nombre que le identifique: rol-eks-master.

Para crear el **rol de IAM de los workers** le daremos permisos con EC2 y buscaremos las siguientes políticas:

- AmazonEC2ContainerRegistryReadOnly
- AmazonEKS_CNI_Policy
- AmazonEKSWorkerNodePolicy

Le ponemos un nombre que le identifique: rol-eks-nodos

Siempre conviene mirar la documentación porque puede cambiar las políticas: https://docs.aws.amazon.com/eks/latest/userguide/add-user-role.html

34.1. - Crear un cluster

Ahora que ya tenemos la red y el IAM podemos crear el clúster. En el primer paso crearemos el controlplay, el maestro. Para crear sin gestionar muchos parámetros podemos configurar:

- Nombre y versión de kubernetes
- Ahora nos pide el rol para el servicio, que debe ser el preparado para el master. Si lo tenemos preparado ya aparecerá.
- Luego podemos encriptar los secrets
- Ahora indicamos la red, que debe ser apropiada, como las plantillas que ofrece AWS. Podemos crear un grupo de seguridad o el que creo la plantilla.
- Indicamos si queremos un endpoint público o no
- Nos pregunta sobre la configuración de logs

Una vez creado podemos entrar para ver sus propiedades:

- **Overview** – Detalles del cluster
  - API server endpoint – Será la entrada por donde ejecutaremos las ordenes.
  - Certificate aythority – Es el que tendremos que usar para conectarnos
  - OpenID Connect provider URL
  - Cluster IAM role ARN
  - Cluster ARN
- **Resources** -
  - Workloads
  - Cluster
  - Service and networking
  - Config and secrets
  - Storage
  - Authentication
  - Authorization
  - Policy

    - Extensions
- **Compute** – Gestionaremos y veremos los workers
- **Networking** – Veremos la VPC con las subnets, el grupo de seguridad y otros detalles de red.
- **Add-ons** – Por defecto, en un cluster vienen coredns, kube-proxy y vpc-cni.
- **Authentication** -
- **Logging** -
- **Update history** -
- **Tags** -

34.2. - Crear workers

En la pestaña “Compute” podremos lanzar grupos de nodos. Podremos especificar:

- Nombre
- Node IAM role. Debe ser el preparado para los workers
- Launch template – Es una plantilla de tipos de EC2.
- Kubernetes labels
- Kubernetes taints - Es una propiedad, una clave, para que el pod sea lanzado o no.
- Tags

Después podemos configurar compute y la autoescalada.

- Tipo de AMI
- Tipo de capacidad – On-Demand o spot (Para pruebas lo más económico son las spot)
- Tipo de instancia
- Capacidad del disco
- Número de nodos deseados
- Númerio de nodos mínimos
- Número de nodos máximos
- Número de nodos indisponibles (Puedo indicar el número o el porcentaje)

En el siguiente paso indicamos las propiedades de red.

- Subnets
- Configuración del acceso por SSH

- Indicamos la Key pair
- Habilitar accesos para…
  - Grupos de seguridad
  - Todo

Al final podemos hacer una revisión de toda la configuración que hemos hecho en el asistente.

Ahora, dentro del clúster, en la pestaña de “Compute”, podremos ver el grupo de nodos creado. Si entramos veremos las siguientes pestañas:

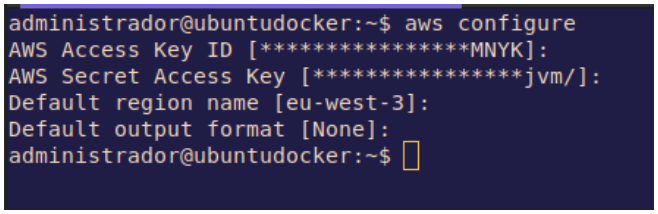
- Detalles
- Nodes
- Health issues
- Kubernetes labels
- Update configuraKubernetes taints –
- Update history
- Tags

En EC2 se podrán ver las instancias desplegadas. Al final, es un grupo de autoescalada dentro del EKS.

34.3. - AWS CLI con EKS

Deberíamos comprobar si tenemos las claves correcta y si estamos en la region.

aws configure



Para trabajar con eks es con el comando

aws eks

Documentación: https://awscli.amazonaws.com/v2/documentation/api/latest/reference/eks/index.html

Subcomandos:

associate-encryption-config
associate-identity-provider-config

create-addon
create-cluster
create-fargate-profile
create-nodegroup
delete-addon
delete-cluster
delete-fargate-profile
delete-nodegroup
deregister-cluster
describe-addon
describe-addon-versions
describe-cluster
describe-fargate-profile
describe-identity-provider-config
describe-nodegroup
describe-update
disassociate-identity-provider-config
get-token
list-addons
list-clusters
list-fargate-profiles
list-identity-provider-configs
list-nodegroups
list-tags-for-resource
list-updates
register-cluster
tag-resource
untag-resource
update-addon
update-cluster-config
update-cluster-version
update-kubeconfig
update-nodegroup-config
update-nodegroup-version
wait

Comandos destacados:

Listar los clústers activos:

```
aws eks list-clusters
```

Listar los grupos de nodos de un clúster

```
aws eks list-nodegroups –cluster-name <nombre-cluster>
```

Información de un cluster

aws eks describe-cluster --name <nombre-cluster>

Información sobre un grupo de nodos

aws eks describe-nodegroup --cluster-name <nombre-cluster> --nodegroup <nombre-nodegroup>

34.4. - kubectl

Para configurar la conexión desde kubectl al cluster deberemos utilizar:

- API server endpoint
- Certificate authority

Se puede configurar mediante el fichero config de .kubectl. Pero para hacerlo más fácil, también se puede hacer desde AWS CLI generando la configuración con el comando:

aws eks update-kubeconfig --name <nombre-cluster>

Si tenemos una configuración creada tendremos que borrar el fichero .kube/config para que se pueda añadir el nuevo.

En el archivo de configuración podemos ver los siguientes datos:

```
apiVersion: v1
clusters:
- cluster:
    certificate-authority-data: LS0tLS1CRUdJ**********************************
    server: https://6C**************.sk1.eu-west-3.eks.amazonaws.com
  name: arn:aws:eks:eu-west-3:99********:cluster/EKS1-redes-publicas
contexts:
- context:
    cluster: arn:aws:eks:eu-west-3:99********:cluster/EKS1-redes-publicas
    user: arn:aws:eks:eu-west-3:99********:cluster/EKS1-redes-publicas
  name: arn:aws:eks:eu-west-3:99********:cluster/EKS1-redes-publicas
```

```
current-context: arn:aws:eks:eu-west-3:99********:cluster/EKS1-redes-publicas
kind: Config
preferences: {}
users:
- name: arn:aws:eks:eu-west-3:99********:cluster/EKS1-redes-publicas
  user:
    exec:
      apiVersion: client.authentication.k8s.io/v1beta1
      args:
      - --region
      - eu-west-3
      - eks
      - get-token
      - --cluster-name
      - EKS1-redes-publicas
      command: aws
```

El current-context es importante porque indica cual es el cluster por defecto, en casa de tener más de uno. En principio se quedará siempre el último cluster creado.

Ahora, si pedimos la versión a kubectl nos dirá también si estamos conectados al cluster

```
kubectl version
```

Si le pedimos los nodos veremos los de AWS

```
kubectl get nodes
```

34.4.1. - Crear un despliegue con kubectl

Primero creamos un namespace

```
kubectl create namespace <nombre-namespace>
```

Ver los namespace

```
kubectl get ns
```

Vamos a poner el namespace creado por defecto

```
kubectl config set-context --current --namespace=<nombre-namespace>
```

Vamos a efectuar un despliegue

```
kubectl create deployment apache1 –image=httpd
```

Vamos a ver los despliegues

```
kubectl get deploy
```

Vemos los replicaset

kubectl get rs

Vemos los pods

kubectl get pods

Podemos ver logs del pod

kubectl logs <nombre-pod>

Ahora, si vamos al dashboard de AWS y entramos en el cluster, en la pestaña Resources, en el apartado Workloads/pods, veremos el pod creado. Podemos entrar dentro y ver detalles.

Para probar el apache tendremos que crear un servicio de kubernetes. Primero exponemos el despliegue

kubectl expose deploy apache1 --port=80 --name=apache1-svc –type=LoadBalancer

Podemos comprobarlo en

kubectl get svc

Aquí nos dirá la dirección para poder entrar en apache. Puede tardar unos minutos.

Vamos a cambiar la página index de apache.

Kubectl exec -it <nombre-pod> -- bash

Entonces entramos dentro del contenedor. Aquí vamos a cambiar el texto del index con

echo "Esto es una prueba" > htdocs/index.html

34.5. - Borrar cluster

Tenemos que seguir el sentido contrario de la creación. A veces nos genera algún error por los grupos de seguridad o por otros componentes. Es importante que esperemos el borrado de cada servicio/componente para seguir borrando el siguiente.

1. Borramos el nodegroup – Dentro del clúster, en la pestaña Compute
2. Borramos el cluster – (Dentro del cluster)
3. Borramos la red – Igual que lo creamos, desde CloudFormation borramos el stack

TEMA 35 - ECS Elastic Container Service

Comentario inicial: En el servicio ECS, a fecha de septiembre de 2022, su interfaz se encontraba en transición de diseño, con lo cuál tiene muchas posibilidades que en un periodo breve de tiempo no corresponda con las descripciones aquí mencionadas.

Un entorno para trabajar con contenedores de manera sencilla. Se pueden lanzar contenedores docker dentro de un entorno controlado, y da algunas funcionalidades más que Docker puro.

Se pueden crear:

- Clusters. Pero no son como Kubernetes. Ahora es un poco distinto el asistente, antes AWS daba 3 opciones: Solo con red, EC2 Linux + Red y EC2 Windows + Red
- Tareas – Son fichero json para ejecutar contenedores.

35.1. - Crear un cluster

Podemos configurar:

- Nombre y la red (VPC + subnet)
- Infraestructura
  - AWS Fargate (Serverless)
  - Amazon EC2
  - Instancias externas
- Monitorización y Tags

Una vez creado podemos entrar y movernos entre las pestañas:

- Services
- Tasks
- Infraestruture – Aparecen las instancias desplegadas.
- Metrics
- Tags

35.2. - Crear una tarea

Una tarea es una plantilla en donde definimos como será el despligue de los contenedores.

En Task definition escribiremos los detalles de la tarea. Tenemos que crear una nueva con las siguiente opciones:

- Nombre

- Datos del contenedor. Nombre, URI de la imagen, puertos, variables de entorno y fichero de entorno. Se puede añadir más contenedores
- Entorno
  - Entorno de la app – Podemos escoger Fargate (serverless) o Instancia EC2
  - Sistema operativo – Linux lo mejor.
  - Capacidad de la tarea (CPU y memoria). Se puede añadir otro contenedor
  - Rol de la tarea. Puede que los contenedores utilicen servicios de Amazon, se debería añadir un rol IAM que pueda hacer llamadas a AWS
  - Rol de ejecución de la tarea
  - Network mode – Bridge, Host, awsvpc, nome. Por defecto viene awsvpc
- Almacenamiento – Se puede especificar una almacenamiento efímero de entre 21 GiB y 200 GiB. Además, se pueden asociar volumenes
- Monitorización y logueo
- Etiquetas

Para lanzar la tarea vamos dentro del clúster, en la pestaña task, clicamos en “Run new task”. También podemos lanzarla en Task definition. Podemos indicar:

- El entorno -
  - Estrategia del proveedor de capacidad – Se le puede dar más detalles
  - Tipo de lanzamiento (Fargate, EC2 o External)
- Configuración del despliegue
  - Definimos el tipo de aplicación – Servicio o tarea.
  - Podemos decir las tareas deseadas.
- Red – Concretamos la VPC, subredes, grupo de seguridad y si queremos IP pública.
- Anulación de tareas -
  - Un rol que permite a los contenedores de la tarea realizar solicitudes a los servicios AWS.
  - Un rol de ejecución de la tarea
- Etiquetas

Una vez lanzada podemos ver la ip pública y si entramos veremos el servicio apache

Para borrar la tarea primero tendremos que pararlas. Hay que diferencias entre la «Definición de tarea» y la «Tarea».

35.3. - Crear un servicio

Tenemos que ubicarnos en el clúster, en la pestaña de servicios, y clicamos en “Deploy”. El asistente es el mismo que las tareas. Podremos configurar:

- El entorno -
  - Estrategia del proveedor de capacidad – Se le puede dar más detalles
  - Tipo de lanzamiento (Fargate, EC2 o External)
- Configuración del despliegue
  - Definimos el tipo de aplicación – Servicio o tarea.
  - Especificar la revisión manualmente.
  - Nombre del servicio
  - Tareas deseadas
  - Tareas mínimas en ejecución
  - Tareas máximas en ejecución
  - Utilice el interruptor de despliegue de Amazon ECS
    - Retirada en caso de fallo
- Red – Concretamos la VPC, subredes, grupo de seguridad y si queremos IP pública.
- Balanceador de carga
- Emplazamiento de tareas - personalizar cómo se colocan las tareas en las instancias de su clúster. https://docs.aws.amazon.com/es_es/AmazonECS/latest/developerguide/task-placement-strategies.html Podemos elegir: AZ balanced spread, AZ balanced binpack, Binpack, Una tarea por host, Al azar o Personalizado
- Etiquetas

Si entramos en el servicio podremos ver:

- Health & metrics
- Logs
- Configuration and tasks
- Deployments and events
- Networking
- Tags

TEMA 36 - ECR Elastic Container Registry

AWS cobra por el espacio de almacenamiento y por la entrada y salida. Tiene dos tipos de registros: Público y privado.

Dentro de los registros tendremos repositorios. Y dentro de cada repositorio tendremos distintas versiones de la misma imagen.

Un registro público tendrá un alias para facilitar el acceso desde internet. Hasta que no creemos el primer repositorio no tendremos el alias, que lo da AWS. Podemos tener uno personalizado si lo solicitamos a soporte. Lo primero es configurar un repositorio.

36.1. - Crear repositorio público

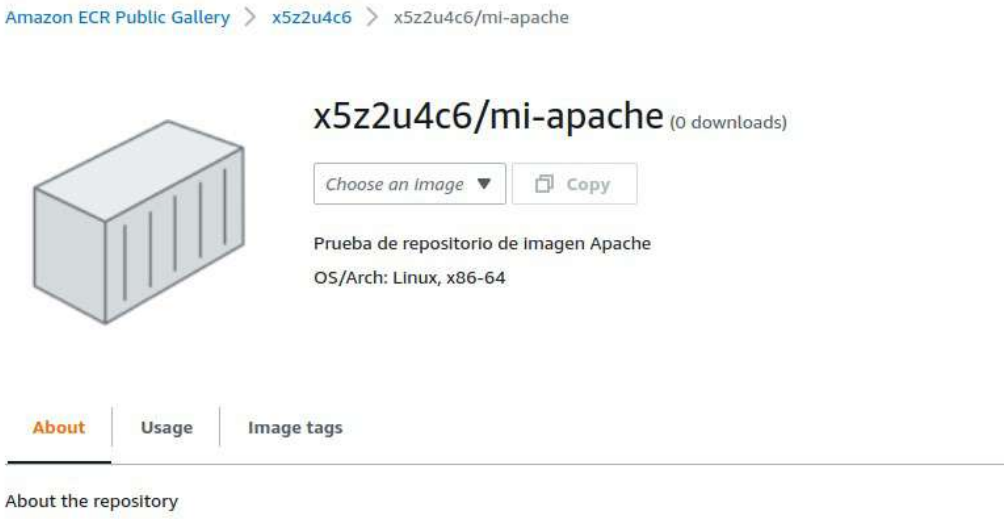
Podremos parametrizar los siguientes componentes:

- Privado o público – Una vez creado no se podrá cambiar.
- Nombre repositorio
- Subir logo
- Descripción
- Asociar a un sistema operativo y a una arquitectura
- Descripción más detallada
- Descripción del uso

La creación es inmediata. Una vez entramos podemos subir las imagenes. Podremos ver:

- Comandos push – Resumen de los comandos necesarios para subir imágenes
- Eliminar repositorio
- Acciones (Ver imagen, permisos, detalles o etiquetas. Editar).

Dentro del repositorio podremos ir a la lista pública que recuerda a docker hub. Es la ECR Public Gallery.



36.2. - Crear una imagen de Docker

Utilizaremos la imagen para probar push, siguiendo los comandos:

1. Recuperar token de autenticación y autenticar cliente Docker en su registro:

   aws ecr-public get-login-password --region us-east-1 | docker login --username AWS --password-stdin public.ecr.aws/x5z2u4c6

   Nota: Ante un error asegurar de que la versión de AWS CLI y de Docker son recientes.

2. Construir imagen Docker utilizando el siguiente comando (Tendremos que tener el Dockerfile preparado en la ubicación del prompt):

   docker build -t mi-apache .

   Comprobar que se ha creado:

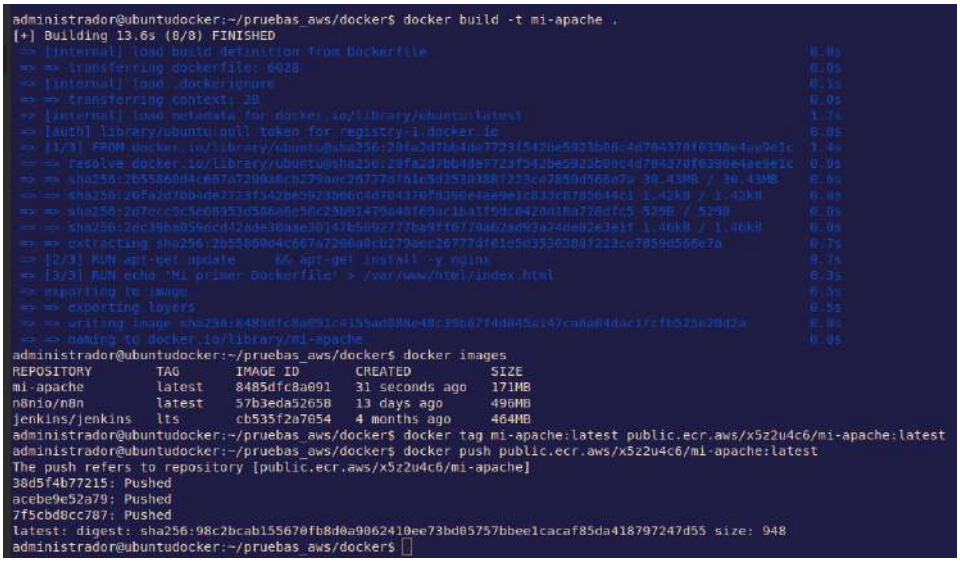
   docker images

3. Después de que la construcción se complete, etiquetar imagen para push al repositorio:

   docker tag mi-apache:latest public.ecr.aws/x5z2u4c6/mi-apache:latest

4. Ejecutar push de la imagen:

   docker push public.ecr.aws/x5z2u4c6/mi-apache:latest



Ahora ya lo tenemos en ECR Public Gallery

Desde la terminal podemos probar que se lanza la imagen con el comando:

docker run --name apache-1 -d -p 80:80 public.ecr.aws/x5z2u4c6/mi-apache

Podemos comprobar si está lanzada con

docker ps

En caso de que queramos descargar la imagen del registro público seria con

docker pull public.ecr.aws/x5z2u4c6/mi-apache:latest

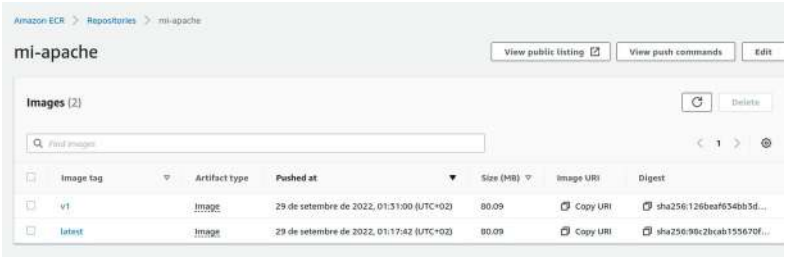
Se puede hacer una segunda versión modificando el Dockerfile y con el siguiente comando que ya indicamos directamente la etiqueta con el nombre completo del registro:

docker build -t public.ecr.aws/x5z2u4c6/mi-apache:v1 .

Y la volvemos a subir

docker push public.ecr.aws/x5z2u4c6/mi-apache:v1

Pero en esta ocasión solo subirá la última capa de cambios.



36.3. - Probar un despliegue con LightSail

https://lightsail.aws.amazon.com/

Nos vamos a los contenedores, creamos uno. Tendremos que configurarlo al gusto y en el espacio de deployment tendremos que especificar la imagen indicando:

- Nombre del contenedor que se creará
- Dirección de la imagen pública. La copiaremos de la imagen en el repositorio ECR o de ECR Public Gallery.
- Indicar más comandos si lo necesitamos
- Variables de entorno
- Puerto por donde escuchará el contenedor a la aplicación.

Después es importante indicar cuál es el contenedor que debe escuchar.

36.4. - Crear repositorio privado

Para poder utilizar las imágenes en un repositorio privado se requerirá permisos adecuados. Es muy parecida a la creación del repositorio público, las diferencias son:

- En el nombre aparecerá el id de la cuenta.
- Se puede activar la inmutabilidad de los tags.
- Se puede hacer una scaneo a la imagen en el momento de subirla. Se puede escanear a nivel repositorio (Deprecated) y a nivel de registro.
- Además, se puede utilizar la encriptación con su KMS.

Las instrucciones para el push son parecidas pero cambia el nombre de la autenticación.

Para descargar la imagen privada primero nos logueamos y después ya podremos efectuar el pull:

aws ecr get-login-password --region eu-west-3 | docker login --username AWS --password-stdin 992365247711.dkr.ecr.eu-west-3.amazonaws.com

docker pull 992365247711.dkr.ecr.eu-west-3.amazonaws.com/mi-apache:latest

36.5. - Asignar permisos de acceso

Se pueden dar permisos a nivel de registro o de repositorio para usuarios/grupos/roles concretos.

Dentro del repositorio privado, veremos los permisos en donde podemos subir un json con las políticas. Podemos seguir un asistente que dará al usuario que escojamos los permisos que configuramos de manera fácil.

No solo se pueden dar permisos de nuestra cuenta, también a usuario externos de AWS.

Clicamos en “Add Statement” y configuramos:

- Nombre statement
- Efecto – habilitado o denegado
- Avanzado – Podemos dar permisos a todo el mundo pero no es lo apropiado
- Principal del servicio – Podemos darle permisos a una entidad principal.
- AWS account Ids – Podemos darle permisos a una cuenta principal y se aplicarán a todos los usuarios.
- Entidad IAM – Podemos darselo a un usuario, grupo o rol.
- Actions – Decidimos las tareas concretas que le queremos dar. Por ejemplo, para dar permisos de solo lectura:
  ecr:DescribeImages
  ecr:DescribeRepositories
  ecr:GetDownloadUrlForLayer
  ecr:ListImages:
- También le podemos dar condiciones

Desde la consola de comandos podemos ver el usuario conectado con

```
aws sts get-caller-identity
```

Si nos conectamos con el usuario con el que tenemos los permisos anteriores, podríamos hacer un pull de la imagen a través de la url.

36.6. - AWS CLI

Los comandos principales para ECR son para trabajos con repositorios privados y públicos.

Repositorios privados:

aws ecr

Documentación: https://docs.aws.amazon.com/cli/latest/reference/ecr/index.html

Los subcomandos son:

batch-check-layer-availability
batch-delete-image
batch-get-image
batch-get-repository-scanning-configuration
complete-layer-upload
create-pull-through-cache-rule
create-repository
delete-lifecycle-policy
delete-pull-through-cache-rule
delete-registry-policy
delete-repository
delete-repository-policy
describe-image-replication-status
describe-image-scan-findings
describe-images
describe-pull-through-cache-rules
describe-registry
describe-repositories
get-authorization-token
get-download-url-for-layer
get-lifecycle-policy
get-lifecycle-policy-preview
get-login
get-login-password
get-registry-policy
get-registry-scanning-configuration
get-repository-policy
initiate-layer-upload
list-images
list-tags-for-resource
put-image
put-image-scanning-configuration
put-image-tag-mutability

put-lifecycle-policy
put-registry-policy
put-registry-scanning-configuration
put-replication-configuration
set-repository-policy
start-image-scan
start-lifecycle-policy-preview
tag-resource
untag-resource
upload-layer-part
wait

Repositorios públicos:

aws ecr-public

Documentación: https://docs.aws.amazon.com/cli/latest/reference/ecr-public/index.html

Los subcomandos son:

batch-check-layer-availability
batch-delete-image
complete-layer-upload
create-repository
delete-repository
delete-repository-policy
describe-image-tags
describe-images
describe-registries
describe-repositories
get-authorization-token
get-login-password
get-registry-catalog-data
get-repository-catalog-data
get-repository-policy
initiate-layer-upload
list-tags-for-resource
put-image
put-registry-catalog-data
put-repository-catalog-data
set-repository-policy
tag-resource
untag-resource
upload-layer-part

Ejemplos:

Para describir el registro privado:

aws ecr describe-registries

Para describir repositorios

aws ecr describe-repositories

Para listar las imágenes de un repositorio

aws ecr list-images –repository-name <nombre-repositorio>

Para crear un repositorio

aws ecr create-repository --repository-name <nombre-repositorio>

Se pueden subir imágenes con AWS CLI pero el mismo AWS recomienda que se use el comandos DOCKER.

Los comandos que tienen que ver con el subcomando *ecr-public* solo funcionan en la region us-east-1, con lo que hay que especificarla para poder utilizarlos.

Por ejemplo, para describir el registro público:

aws ecr-public describe-registries --region us-east-1

Podemos pedir los repositorios con:

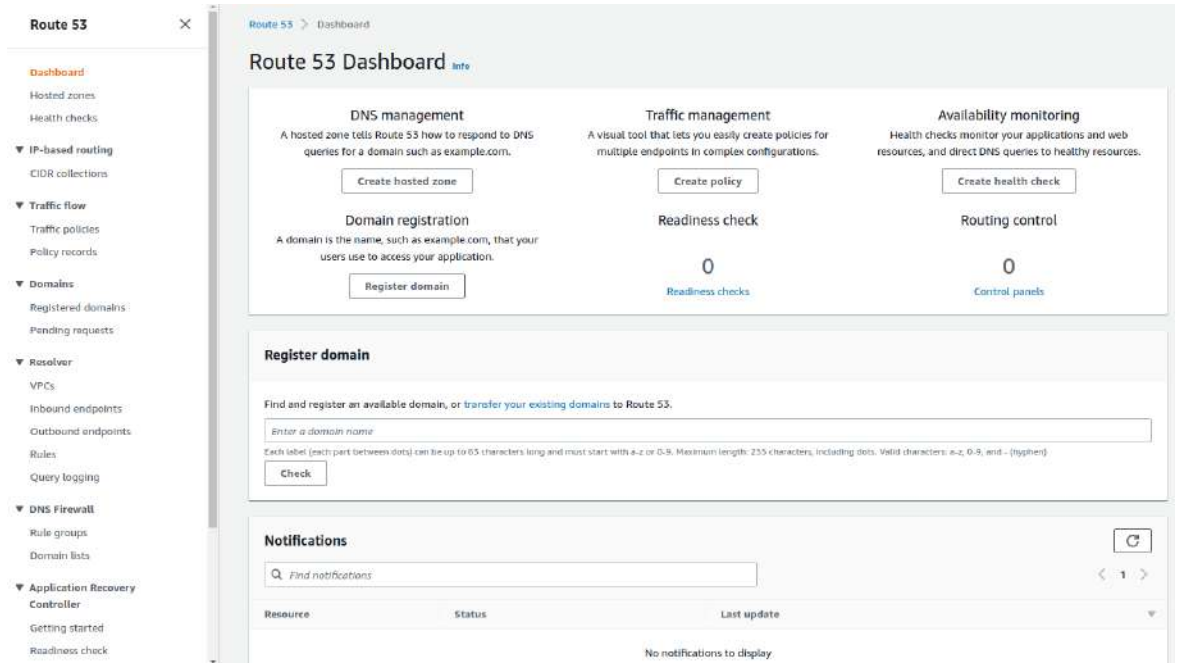
aws ecr-public describe-repositories --region us-east-1

TEMA 37 - Route 53

Es un DNS que se integra perfectamente con los productos de AWS. Características:

- Permite toda la **gestión de dominios**. Crear dominios y subdominios, etc
- Configuración de **zonas de host** para responder a peticiones concretas de los dominios.
- Controles de **seguridad** – Monitorización, etc
- **Traffic flow** – es una herramienta que mediante gráficos nos permite crear políticas de acceso de manera que podemos crear configuraciones muy complejas de manera fácil.
- **Resolver** – Va a permitir identificar hacia donde deben ir ciertas peticiones que lleguen a nuestra red.

El Dashboard es muy completo:



En el panel central tenemos las opciones más importantes:

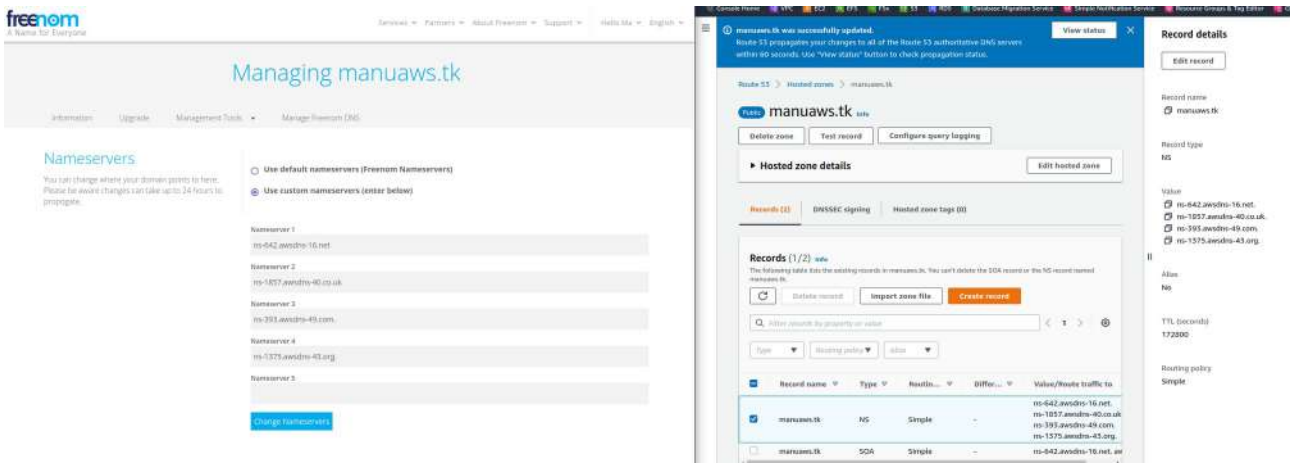
- Crear una zona de host
- Crear políticas de seguridad
- Crear testeos
- Registrar dominios

En el sidebar izquierdo tenemos más detalles.

Lo primero que debemos hacer es crear, añadir o registrar un nuevo dominio.

37.1. - Dominio

El registro de dominio sigue con el formato antiguo. AWS es caro para el registro del dominio. Personalmente he creado un dominio gratuito en freenom y apunto sus DNS a una zona de host creada en AWS. (También cobran por la zona de host, normalmente es un servicio gratuito cuando tienes un dominio o un hosting).



Este mecanismo tendrá que propagar las DNS y puede tardar hasta 48 horas. Normalmente se efectúa en menos de 24 horas.

En el caso de registrar o transferir un dominio, aparecerán las propiedades en el dashboard “Registered domains”. Tiene las mismas propiedades que cualquier otro registrador de dominios.

37.2. - Hosted zone

En una zona de host indicaremos la configuración de DNS para darle la ubicación a cada uno de los dominios y subdominios.

Las zonas pueden ser públicas (Para acceder desde Internet) o privadas (Para redirigir el tráfico interno de las VPC). Si creamos una zona nos pedirá nombre, descripción tipo (público o privado) y etiquetas.

Cuando se crea una zona, por defecto, nos crea 2 registros:

- Un NS simple con 4 servidores de nombres que son los autorizados para su zona alojada.
- Un SOA de autoridad con un servidor de nombres, la dirección del mail administrador, número de serie, el tiempo de actualización, el intervalo de reintento, el tiempo de intentos de un servidor secundario en completar una zona y el TTL.

CUIDADO, no se pueden borrar.

Documentación de los tipos de registros que soporta Route 53 (son los comunes): https://docs.aws.amazon.com/es_es/Route53/latest/DeveloperGuide/ResourceRecordTypes.html?icmpid=docs_console_unmapped

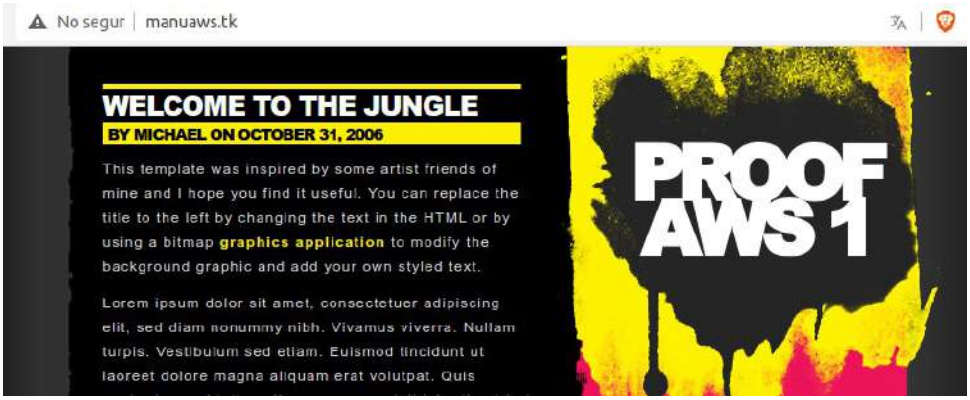
37.3. - Probar zona host con una instancia

Para probar la zona de host podemos abrirnos una instancia con un apache instalado, crear una pequeña web en /var/www/html

Luego hacemos un nuevo registro con la IP del servidor. Vamos a la zona de host y “Create record”.

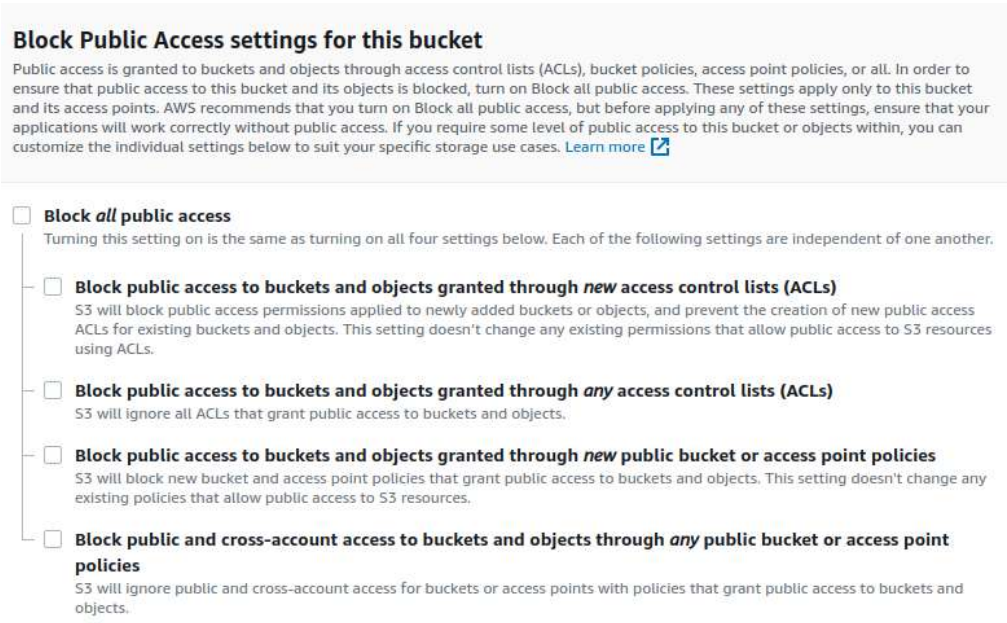
Ahora vamos a darle el subdominio www y de tipo A.

Indicamos la dirección IP de la instancia y mantenemos por defecto el TTL y la política de seguridad (Simple routing). Y ya lo tenemos



37.4. - Probar zona host con un bucket S3

Necesitaremos un bucket preparado para web. Le llamaremos igual que el dominio. Ponemos al region que queramos, le podemos dejar las ACLs deshabilitada y lo importante es desbloquear el acceso público



Confirmamos que somos conscientes que no es seguro el bucket. El resto lo dejo todo deshabilitado.

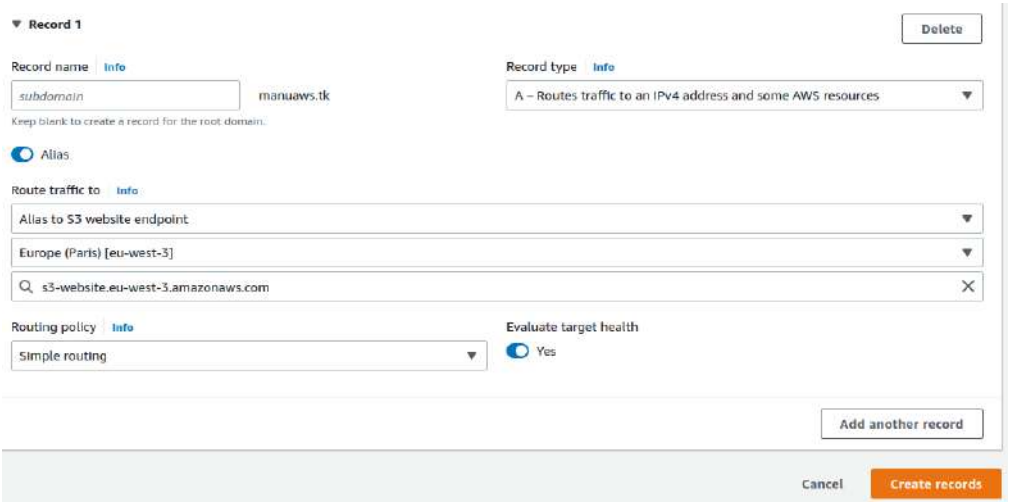
Vamos al final de las propiedades del bucket y habilitamos “Statis Website Hosting”. También indicamos el fichero index y si queremos el error.

Subimos los ficheros de una web estática y vamos a darle los permisos necesarios. Para hacerlo rápido, le damos un statement genérico:

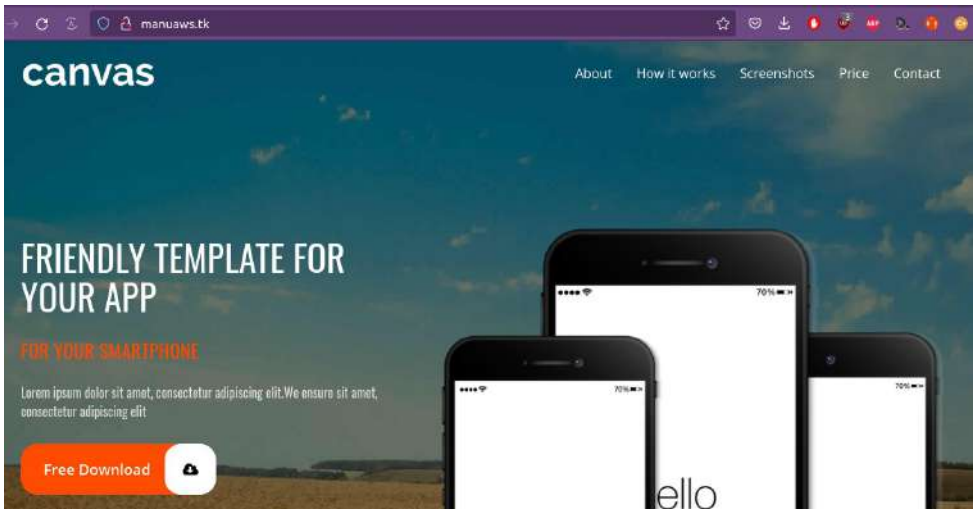
```
{
	"Version": "2012-10-17",
	"Statement": [
		{
			"Sid": "AddPerm",
			"Principal": "*",
			"Effect": "Allow",
			"Action": [
			   "s3:GetObject"
			],
			"Resource": [
			   "arn:aws:s3:::manuaws.tk/*"
			]
		}
	]
}
```

Ahora ya tenemos listo el bucket y le podemos dar la ubicación en la zona de host a las DNS. “Create record”

Esta vez no tenemos IP, vamos a apuntar a un recurso con alias, podremos escoger “S3 website endpoint”. Podremos ver que podemos escoger entre API Gateway, AppRunner, AppSync, CloudFont, etc



Creamos registro



37.5. - Probando subdominio

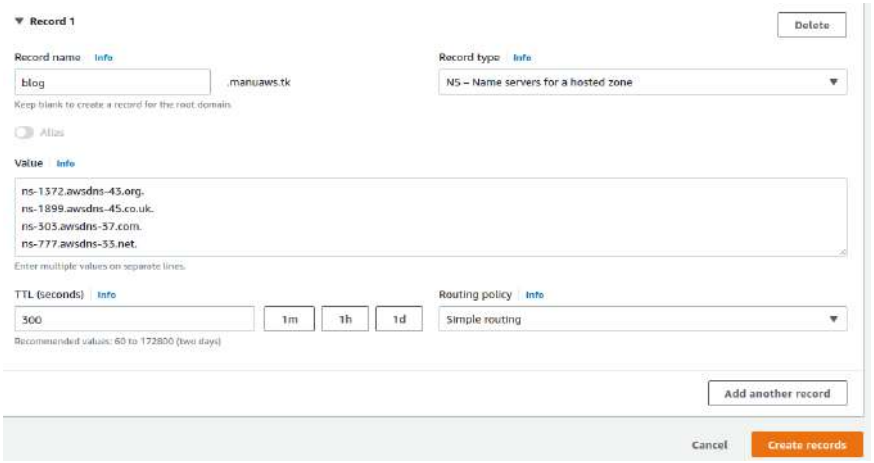
Utilizaremos una instancia con otra web, igual que antes la pondremos en el apache.

Dentro de las zonas alojadas crearemos otros record, le pondremos un nombre de subdominio. Le ponemos la IP de la instancia y listo.

Visto esto, vamos a **crear una zona de host para un subdominio**, lo que se denomina la delegación de responsabilidades a nivel de subdominio.

Vamos a crear una zona. Le ponemos el nombre del subdominio y la descripción que queramos. Luego dejamos la zona como pública y creamos. Tendremos los cuatro NS y el SOA.

Ahora tendremos que mapearla dentro de la zona principal, así que vamos a copiar los NS. Vamos al dominio principal, creamos registro con el nombre del subdominio, de tipo NS y con los valores de las NS de la zona del subdominio.



Para probarlo podemos crear un registro en la zona del subdominio apuntando a una de las instancias o buckets que ya tenemos.

37.6. - Apuntando a un load balancer

Tan solo se tiene que mapear a través del alias pero esto nos sirve para ver el failover. Debemos tener el balanceador de carga creado apuntando a un grupo de autoescalado. Para probar algo nuevo podemos probar el switch to wizar donde crearemos el registro con un asistente:

- Escogemos la política de enrutamiento. Simple routing
- Ahora definimos el registro
  - Podemos el subdominio que queramos.
  - Tipo de registro A
  - Enrutamos a un recurso de AWS, la región y la ALB que hemos creado.

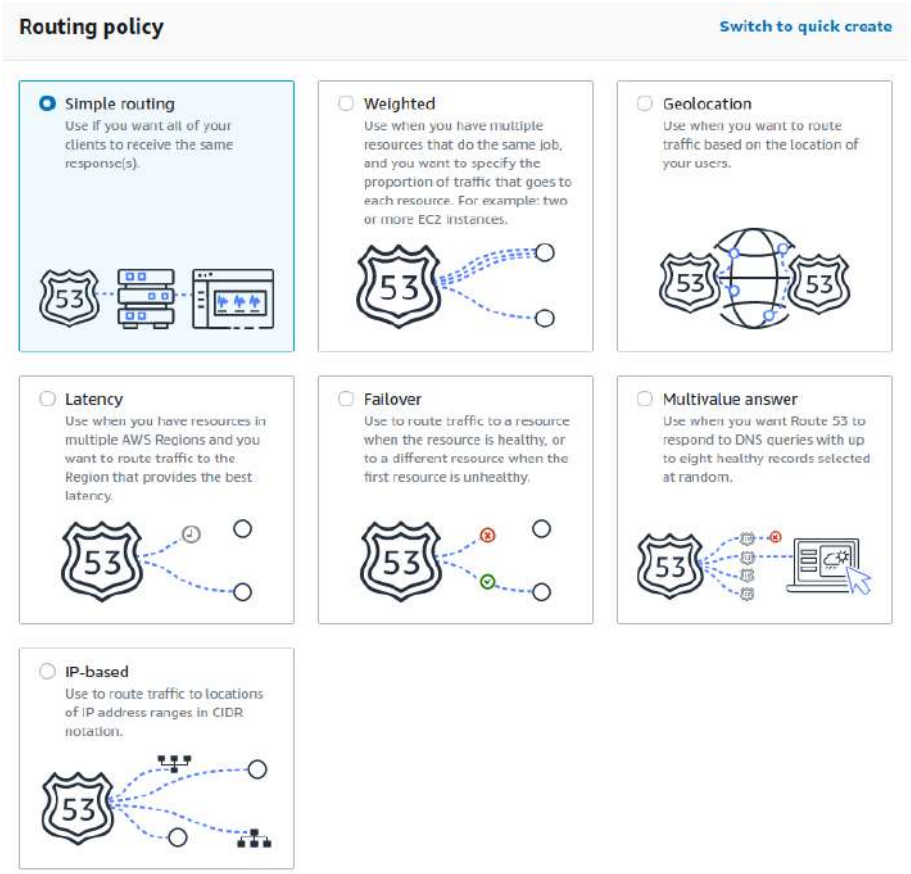
37.7. - Health Check

Tendremos que tener creada una DNS apuntando a un host con una web. Seleccionamos health Check en el sidebar izquierdo y creamos el Healt Check. Configuramos:

- El nombre
- Podemos monitorizar
  - Endpoint – un host concreto
  - Otros health check– Podemos agrupar otros chequeos
  - CloudWatch alarm
- Podemos ponerle una IP o un nombre de dominio. Podemos chequear cualquier dominio de Route53 o los dns públicos que asocia AWS a las instancias y otros servicios.
- Configuración avanzada
  - Cada cuanto tiempo hacemos el chequeo
  - Umbral para considerar que es un error.
  - String matching – podemos añadir el texto que devuelve el error.
  - Gráfico de latencia
  - Si queremos invertir el chequeo.
  - Desactivar el chequeo
  - Indicar las regiones desde donde se hace el chequeo.
- Además de las acciones anteriores podemos activar una alarma SNS. Si decimos que sí tendríamos que seleccionar el topic o crearlo.

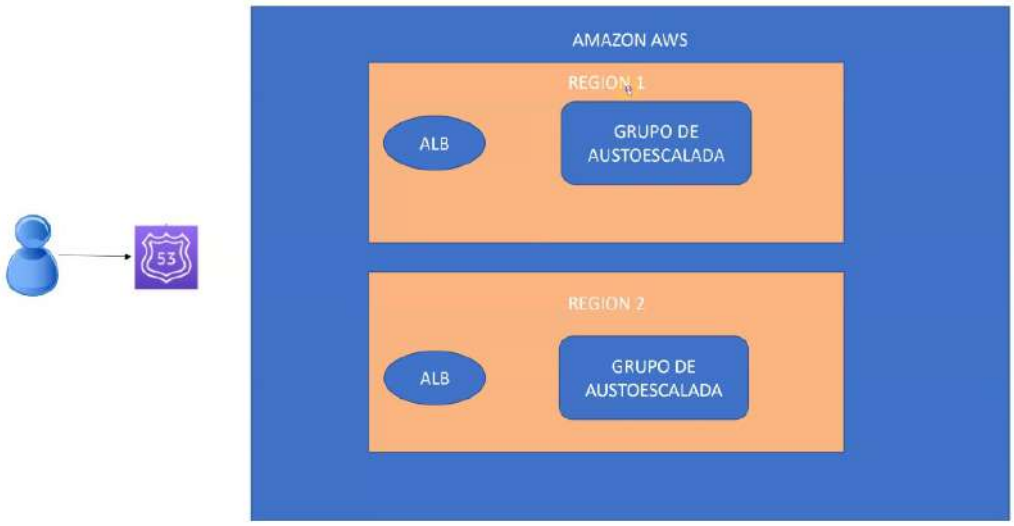
37.8. - Políticas de enrutamiento

Cuando se crea un registro en modo wizard se pueden ver bastante claras



- Simple routing
- Weighted – De tipo peso, que sirve para cuando tenemos distintos recursos haciendo el mismo trabajo, podremos enrutar según las necesidades que se tengan a través del peso.
- Geolocation – Hace que se enrute en la localización que este más cerca de los usuarios.
- Latency – Con recursos en distintas regiones el tráfico se enruta en la que menos latencia exista.
- Failover – Permite un entorno activo pasivo. Si cae el principal se enrutará el pasivo.
- Multivalue answer – Se va a responder con hasta 8 registros correctos que se escogen de manera aleatoria para que el cliente pueda escoger como acceder.
- IP-based - Se utiliza para enrutar el tráfico a ubicaciones de rangos de direcciones IP en notación CIDR.

37.8.1. - Failover



Vamos a tener un cliente conectado a través de route 53 que irá a dos regiones de tipo activo-pasivo. Con un balanceador de carga (ALB) en cada región apuntando a un grupo de autoescalada.

De tal manera que si falla una continuará la otra dando el servicio.

Entonces, tendremos que crear un Health Check apuntando a un balanceador de carga. ALB-A.

Crearemos otro Health Check apuntando al segundo balanceador de carga. ALB-B

Ahora vamos a la zona de host, creamos un registro con failover. Le damos un subdominio, enrutamos a un recurso de AWS. Escogemos "Alias to Application and Classic Load Balancer"/region y balanceador de carga.

El tipo de failover tendremos que indicar que Primary, que será el que responderá a las peticiones.

Tendremos que indicar un ID.

Ahora definimos el segundo, con la segunda region y le indicamos Secondary. Creamos.

Entonces, si entramos en el subdominio creado entraremos en ALB-A. Si eliminamos el grupo de autoescalada de ALB-A y volvemos a entrar, nos llevará a ALB-B.

37.8.2. - Enrutamiento por peso

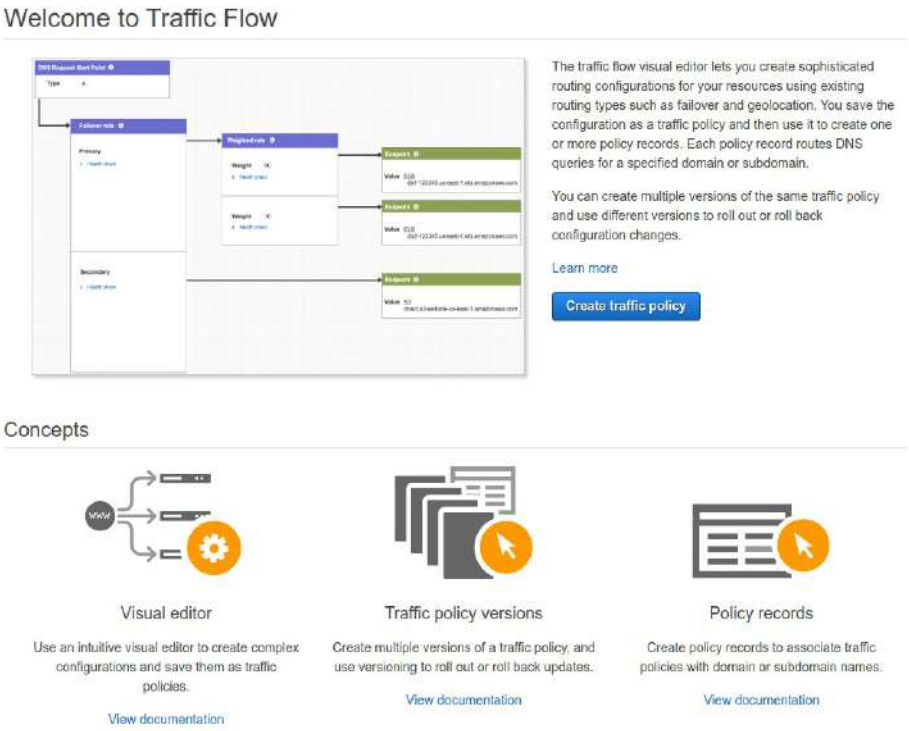
Probamos un entorno activo-activo. Necesitaremos 3 instancias con webs identificables.

Volvemos a Route 53, a nuestra zona de host y crearmos un registro con Weighted con un subdominio. Tendremos que definir el Weighted record de cada servidor, pondremos el peso entre 0 y 200. Si ponemos 0 no participará en este enrutamiento. También tendremos que indicar la ID.

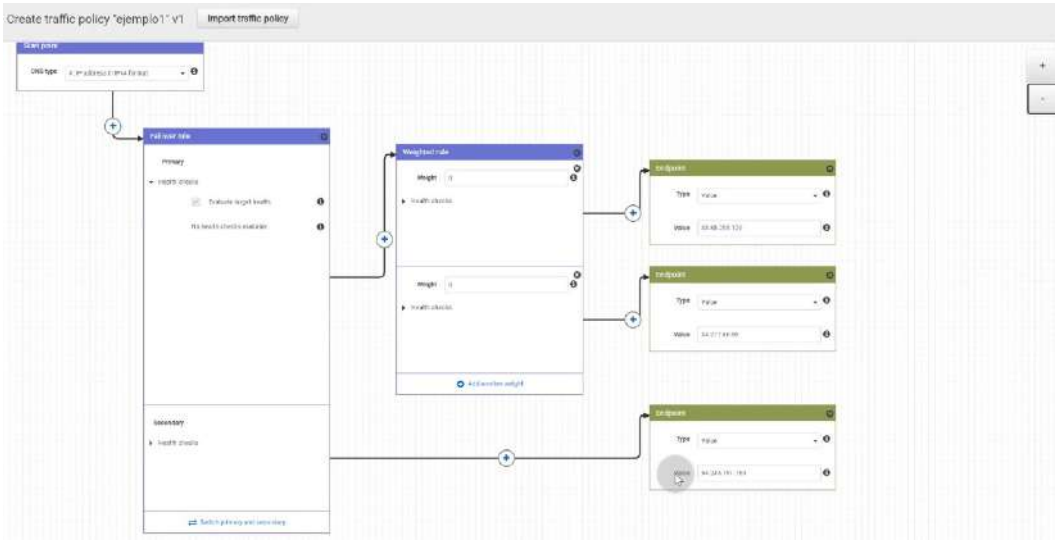
Esto lo tendremos que hacer con las tres instancias. Creamos el registro. Ahora entramos en el subdominio para ver los resultados.

37.9. - Traffic Flow

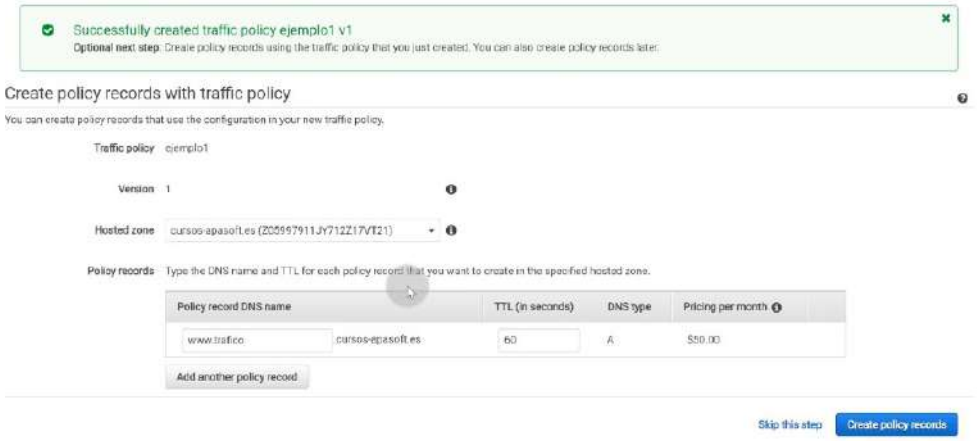
Si entramos en Traffic Flow veremos que es un editor visual para crear políticas de enrutamiento complejas.



Necesitaremos 3 instancias. Creamos una política de tráfico. Nos pedirá nombre y descripción, después ya podremos dibujar la política de tráfico. Podemos importar y exportar un json con todos los datos.



Una vez dibujado lo creamos y tendremos que crear otro registro que tenda esta política. CUIDADO Tendriamos que ver el coste porque suelen ser caro.


Successfully created traffic policy ejemplo1 v1
Optional next step: Create policy records using the traffic policy that you just created. You can also create policy records later.
Create policy records with traffic policy
You can create policy records that use the configuration in your new traffic policy.
Traffic policy ejemplo1
Version 1
Hosted zone cursos-apasoft.es (Z05997911JY712Z17VT21)
Policy records Type the DNS name and TTL for each policy record that you want to create in the specified hosted zone.
Policy record DNS name
TTL (in seconds)
DNS type
Pricing per month
www.trafico
cursos-epasoft.es
60
A
$50.00
Add another policy record
Skip this step
Create policy records

TEMA 38 - Resumen de todos los servicios

38.1. - Administración de costes

- Cost Explorer: Una buena vista de todos los costes de AWS. Importante revisarlo periódicamente y **crear alguna alarma**, para controlar los costes mensuales
- Budgets: Añade o crea **presupuestos y alarmas** para avisarte cuando tus costes pasan de un umbral.
- Marketplace Subscriptions: Aquí dispones de todo un catálogo de soluciones de terceros, listos para usar con la tecnología AWS, por ejemplo suscripciones de PFsese, Citrix, Microsoft o distribuciones específicas de linux.
- ABC: AWS Billing Conductor es más fácil que nunca para los equipos de FinOps configurar, generar y compartir las tarifas correctas con los usuarios finales, independientemente de las tarifas que el cliente haya negociado con AWS.

38.2. - Computación (máquina)

- EC2: Genera **máquinas virtuales** elásticas y escalables (Linux/Windows/Mac). De computo general, hasta 448 Vcpu y 12 TBytes RAM, o complejas con GPUs, alto rendimiento, P4d (para machine learning)...
- Lightsail: Las instacias y contenedores **virtuales básicos** (Linux y Windows), para competir en precio, más básicas, convertibles en EC2. Pago mensual al estilo tradicional, con Balanceador de Carga, Bases de datos Postgresql y Mysql, .
- Lambda: Las funciones en **serverless** (sin servidor), pago por uso en llamadas a las funciones (Node.js, Python, Java, C#, Go, Ruby).
- Batch: **Procesamiento** por lotes totalmente administrado para cualquier escala.Por ejemplo lanzar jobs de ejecución de un script o ejecutable linux en cientos de miles de máquinas.
- Elastic Beanstalk: Rápida forma de **implementar y escalar** servicios y aplicaciones web desarrollados con Java, NET, PHP, Node.js, Python, Ruby, Go y Docker en servidores familiares con Apache, Nginx, Passenger e IIS. (genera máquinas virtuales EC2).
- Amazon EKS: Amazon Elastic Container Service for Kubernetes es un servicio administrado que le permite ejecutar fácilmente **Kubernetes** en AWS sin necesidad de instalar ni usar sus propios clústeres de Kubernetes.
- Fargate: Te permite ejecutar **contenedores** sin tener que administrar servidores ni clústeres. Ya no tendrá que aprovisionar, configurar ni escalar clústeres de máquinas virtuales para ejecutar los contenedores
- Elastic Container Service: Servicio de administrador de contenedores compatible con **Dockers**. (puede ser alternativa a k8s)
- Outpost : Ejecuta infraestructura de **AWS en tu CPD**, es decir on-premise. Utilizas la misma API tanto para tus servidores como para la infraestructura AWS.
- Parallel Cluster: Herramienta de **administración de clústeres** de código abierto, mantenida y totalmente compatible que facilita a científicos, investigadores y administradores de TI la

implementación y administración de clústeres de informática de alto rendimiento (HPC) en la nube.
- Braket: Si quieres probar proyectos en máquinas cuánticas este servicio te permite explorar algoritmos para la informática cuántica.

38.3. - Almacenamiento

- S3: **Almacenamiento de objetos** con 99.999999999% durability (para guardar ficheros, backups, imágenes, código, etc) (no confundir con un dropbox/onedrive, aunque tiene similitudes). Primer servicio creado por AWS
- SFTP: Un servicio para S3 que permite acceder por el protocolo **seguro FTP** a los objetos S3.
- EBS: **Almacenamiento a nivel de bloque** de Sistemas Operativos, para las EC2.
- EFS: **Almacenamiento en red** para instancias EC2, como un NAS-iscsi.
- Glacier: Para **backups a largo plazo**, duradero y a super bajo coste (0,007$/mes/gbyte).
- Storage Gateway: Varios tipos de **conectores al S3**, (Hardware Appliance, con tu NAS on-premise), Tape Gateway y FileGateway para montar (on premise) desde Linux, Windows o Mac una unidad a un backet S3.
- Snowball: Solución de **transporte de datos (Hardware) a escala** de petabytes que utiliza dispositivos diseñados para ser seguros y para transferir grandes volúmenes de datos hacia y desde la nube de AWS
- Snowmobile: **Snowball pero a lo bestia** (a escala de exabytes), hasta 100 PB en un contenedor para transportarlo en un camión.
- Snowcone: Curioso dispositivo versátil, que permite guardar datos y/o calcular y/o transmitir a s3, de apenas 2.1 Kg.
- FSx for lustre: Sistema de ficheros pensado para trabajar con **cientos de Gigas por segundo**, millones de IOPs y latencias de milisegundos.
- AWS Backup: (Servicio de backup, **copia de seguridad,** centralizado y gestionado para automatizar copias de EFS, EBS, etc...
- Fsx for Windows File Server: Ofrece un sistema de **archivos de Microsoft Windows nativo** completamente administrado para que pueda migrar con facilidad sus aplicaciones basadas en Windows que requieren almacenamiento de archivos en AWS. Amazon FSx, basado en Windows Server

38.4. - Bases de datos

- RDS: Las **Bases de datos relacionales** administradas, MsSQL, MySQL, MariaDB, PostgresSQL, Oracle, todas escalables y con alta disponibilidad, ahora con **Storage Auto Scaling** etc...
- RDS Proxy: proxy de base de datos de alta disponibilidad y completamente administrado que hace que las aplicaciones sean más escalables, más resistentes a fallos de bases de datos y más seguras.

- Aurora: Base de datos Relacional, compatible MySQL, PostgresSQL. Aurora es hasta **cinco veces más rápida que las bases de datos** de MySQL estándar y tres veces más rápida que las bases de datos de PostgreSQL estándar.
- DynamoDB: **Base de datos no relacional** (no-sql), super-rápida y auto-escalable, con un 99.999 % SLA
- ElastiCache: Caches o **datastores en memoria**. Compatible con Redis y Memcached.
- Amazon Redshift: Para **almacenamiento de datos para BI** (permite ejecutar consultas complejas en petabytes).
- Neptune: Servicio de **base de datos de gráficos rápido**, de confianza y totalmente administrado que le permite crear y ejecutar fácilmente aplicaciones que funcionen con conjuntos de datos muy conectados
- DocumentDB: Base de datos no relacional, compatible con **mongoDB**, rápida, escalable y alta disponibilidad totalmente gestionada por Amazon.
- Timestream: Es un servicio de **base de datos de serie temporal rápida**, escalable y completamente administrado para IoT, etc, que facilita el almacenamiento y análisis de billones de eventos diarios
- SimpleDB: Pues estaba un poco escondido, pero es un servicio de AWS desde el 2007 que se parece mucho a **DynamoDB pero más básico y sencillo**.

38.5. - Red y contenido

- VPC: **Red Aislada Virtual**, dentro de tu entorno en la nube. similar a una vlan dentro de tu red de AWS
- CloudFront: El **CDN** de amazon con más de 300 ubicaciones de borde y 13 cachés de nivel medio regionales, distribuidos en más de 90 ciudades de 47 países, para acelerar tu contenido estático y mejorar las latencias y seguridad.
- Direct Connect: **Conexión de red dedicada** con amazon a su red/ubicación on-premise, sin pasar por Internet.
- Route 53: Un potente **gestor DNS**, incluso reparte la carga por ejemplo en función de geolocalización o latencia, balanceo.
- API Gateway: Facilita a los desarrolladores la creación, la publicación, el mantenimiento, la monitorización y la protección de **API** a cualquier escala
- AWS PrivateLink: Permite **conectar VPC** directamente sin pasar por Internet y válido para que on-premise use servicios de AWS de forma privada
- AWS GroundStation: Servicio gestionado con **conexión** con la mayoría de los actuales **satélites**, puedes conectar tu VPC a un satélite y procesar información, pagas por minutos.
- DATAsync: Servicio de **transferencia de datos**, para conectar EFS a tu on-premise
- Global Accelerator: Servicio de red que usa las propias redes de AWS para que tenga menos congestión y mayor disponibilidad, **un enrutamiento inteligente** en sus redes.
- Private 5G: Crear redes móviles privadas 5G nunca ha sido tan facil

38.6. - Migración

- AWS Migration Hub: Ofrece una ubicación única para realizar el seguimiento de los avances de las **migraciones de aplicaciones** en varias soluciones de AWS y otras empresas del sector.
- Application Discovery Service: Ayuda a los clientes empresariales a planificar proyectos de migración al r**ecopilar información sobre sus centros de dato**s on-premise.
- Database Migration Service: Ayuda a **migrar las bases de datos a AWS** de manera rápida, sencilla y segura.
- Server Migration Service: Servicio sin agente que le permite **migrar de forma más rápida** y sencilla miles de cargas de trabajo on-premise a AWS. Con AWS SMS, puede automatizar, programar y monitorizar replicaciones incrementales de volúmenes de servidores en vivo, lo que facilita la coordinación de migraciones de servidores a gran escala.
- Transit Gateway: Permite conectar sus Amazon Virtual Private Clouds (VPC) y sus **redes locales a una única gateway**. Funciona como un concentrador que controla la manera en la que el tráfico se direcciona a todas las redes conectadas que funcionan como radios.

38.7. - Herramientas de desarrollo

- SDK: Completo set de librerías para desarrolladores que permite que su aplicación obtenga acceso directo a los servicios de AWS. Hay librerías para node.js, Ruby, PHP, Go, C++, Python, JavaScrip, .NET, y Java
- Cloud9: Un IDE en la nube, permite escribir código, ejecutar y depurar solamente con un navegador, incluye un terminal con las CLI todo pre-configurado.
- CodeStar: Facilita la **configuración de toda la cadena de herramienta**s de desarrollo y entrega continua para codificar, compilar, probar e implementar el código de la aplicación. Para comenzar un proyecto, puede elegir entre varias plantillas de Amazon EC2, AWS Lambda y AWS Elastic Beanstalk.
- CodeCommit: Servicio de **control de código fuente** totalmente administrado que facilita a las empresas el hospedaje de repositorios Git privados, seguros y altamente escalables.
- CodeBuild: Es un servicio de creación totalmente administrado que crea **código fuente, ejecuta pruebas y produce paquetes** de software listos para su implementación. No es necesario aprovisionar, administrar y escalar sus propios servidores de creación.
- CodeDeploy: Servicio que **automatiza las implementaciones de código** en cualquier instancia, incluidas las instancias de Amazon EC2 y aquellas ejecutadas on-premise.
- CodePipeline: Es un servicio de **integración continua y entrega continua** para realizar actualizaciones de aplicaciones e infraestructura rápidas y de confianza. CodePipeline compila, prueba e implementa el código cada vez que se produce un cambio en este, de acuerdo con los modelos de procesamiento de la publicación que defina.
- X-Ray: Ayuda a **desarrolladores a analizar y depurar aplicaciones** distribuidas de producción, como las creadas con una arquitectura de micro-servicios

- Amplify: Conjunto de herramientas y características especialmente diseñadas que permite a los desarrolladores frontend web y móviles crear rápida y fácilmente aplicaciones integrandose en AWS.
- App Mash: Facilita la ejecución de **microservicios** al proporcionar una visibilidad constante y controles de tráfico de red para cada microservicio en una aplicación
- Corretto: Es una distribución sin costo, multiplataforma y lista para producción de Open Java Development Kit (**OpenJDK**)

38.8. - Herramientas de Administración

- CloudWatch: **Monitoriza todos** los servicios de AWS con métricas, disparadores, detección de anomalías (usa ML en tu infraestructura), etc...
- CLI: Interface de **línea de comandos** para Mac, linux, Windows, para poder controlar varios servicios de AWS desde la línea de comando y automatizarlos mediante secuencias de comandos.
- CloudFormation: **Infraestructura como código**, creador de mundos de AWS todo con scripts (yaml y json).
- CloudTrail: Permite realizar regulaciones y **auditorías operativas**, de riesgo/seguridad y conformidad en su cuenta de AWS.
- Config: Permite examinar, auditar y **evaluar las configuraciones** de tus recursos en AWS.
- OpsWorks: Administración de configuraciones que utiliza **Chef**, una plataforma de automatización que trata las configuraciones de servidor como código.
- Service Catalog: Permite a las organizaciones crear y administrar **catálogos de servicios** de TI aprobados para su uso en AWS. Incluye todo lo relacionado con imágenes, servidores, software y bases de datos de máquinas virtuales para completar las arquitecturas de aplicaciones multi-nivel.
- Trusted Advisor: Ayuda a **reducir los costos**, incrementar el desempeño y mejorar la seguridad al optimizar el entorno de AWS, proporciona asesoramiento a tiempo real para ayudarle a aprovisionar los recursos de acuerdo con las prácticas recomendadas de AWS.
- Systems Manager: **Recabe información** operativa e implemente acciones en recursos: Unifica, agrupa y automatiza tareas operativas de los recursos. Permite conectarse a las instancias (Windows/Linux) sin abrir puertos ni claves .PEM (Session Manager)
- Personal Health Dashboard proporciona **alertas y orientación** para solucionar problemas en caso de que AWS esté experimentando eventos que puedan afectarle
- Security HUB : Ver y administrar de **forma centralizada, las alertas de seguridad** y automatizar las verificaciones de cumplimiento.
- Cloud Map : Te permite **registrar cualquier recurso de la aplicación**, como bases de datos, colas, microservicios y otros recursos en la nube con nombres personalizados.
- Instance Scheduler 2.0 : Configuración de inicios y detección de las instancias EC2 y RDS para un ahorro de costes.

38.9. - Servicios Multimedia

- Elastic Transcoder: Realiza tareas de **trans-codificación** de contenido.
- Kinesis Video Streams: Aprendizaje automático y tareas analíticas
- Elemental MediaConvert: **Trans-codificador de vídeos basado en archivos** con características de emisión.
- Elemental MediaLive: **Procesamiento de vídeo** en directo para emisión
- Elemental MediaPackage: **Prepare y proteja fácilmente vídeos** para entrega bajo demanda y en directo.
- Elemental MediaStore: Ofrece nivel de desempeño, consistencia y **baja latencia para entrega de vídeo**.
- Elemental MediaTailor: **Inserta anuncios** dirigidos individualmente en sus transmisiones de vídeo.
- Elemental MediaConnect: Transferencia de **vídeos en directo** segura y fiable
- Interactive Video: Cree experiencias de transmisión en directo atractivas

38.10. - Identidad, Seguridad y Conformidad.

- IAM: Controle el acceso de forma segura a los servicios y recursos de AWS de sus **usuarios, grupos, roles, políticas..**
- Inspector: Servicio de **evaluación de seguridad** con y sin agente para máquinas EC2, evalúa si se puede comprometer. A diferencia de Guard Duty que vigila si hay malware en ejecución
- GuardDuty: Es un servicio de detección de amenazas administradas que monitoriza continuamente el comportamiento malicioso o no autorizado para proteger sus cuentas y cargas de trabajo de AWS. Supervisa la actividad, como llamadas API inusuales o implementaciones potencialmente no autorizadas que indican un posible compromiso de la cuenta. También detecta instancias potencialmente comprometidas.
- Certificate Manager: Permite aprovisionar, administrar e implementar con facilidad certificados de capa de conexión segura/seguridad de la capa de transporte (**SSL/TLS**) para su uso con servicios de AWS.
- Directory Service: **Microsoft AD** - Active Directory administrado en la nube de AWS. Migre con facilidad aplicaciones y cargas de trabajo compatibles con el directorio, disponible también como (Standard Edition)
- WAF & Shield: Un **cortafuegos de aplicación** web que permite monitorizar accesos a tu CloudFront o evitar DDoS.
- Artifact: Portal de **auditoria y conformidad** para el acceso bajo demanda, permite descargar los informes de conformidad (pdf) de AWS y administrar acuerdos seleccionados.
- Amazon Macie: Es un s**ervicio de seguridad que utiliza el aprendizaje automático** para descubrir, clasificar y proteger datos confidenciales automáticamente en AWS. Reconoce los datos confidenciales, como la información personalmente identificable (PII) o la propiedad intelectual y le proporciona paneles de control y alertas que aportan visibilidad acerca de cómo se están trasladando estos datos o se está accediendo a ellos.

- CloudHSM: Es un módulo de **seguridad de hardware (HSM)** basado en la nube que le permite generar con facilidad y usar sus propias claves de cifrado en la nube de AWS.
- Cognito: Permite agregar **el registro y el inicio de sesión** de forma sencilla a sus aplicaciones web y móviles. También tiene las opciones de autenticar a los usuarios a través de proveedores de identidad social, como Facebook, Twitter o Amazon, con soluciones de identidad SAML.
- Single Sign-on: Administre de manera centralizada el acceso con **inicio de sesión único** a varias aplicaciones empresariales de AWS, o conectable con Active Directory.
- AWS Control Tower: Automatiza la configuración de un entorno de referencia, o zona de almacenamiento, que es un **entorno con varias cuenta**s, seguro y bien diseñado de AWS. La configuración de la zona de almacenamiento está basada en las prácticas recomendadas que han sido establecidas al trabajar con miles de clientes empresariales para crear un entorno seguro que facilite controlar las cargas de trabajo de **AWS con reglas de seguridad, operaciones y cumplimiento.**

38.11. - Analítica

- Athena: fácil generador de consultas que facilita el **análisis de datos en Amazon S3 con SQL estándar para crear data Lakes**. Athena es serverless, de manera que no es necesario administrar infraestructura y solo paga por las consultas que ejecuta.
- EMR: Proporciona un marco **Hadoop** hospedado que facilita, acelera y rentabiliza procesar enormes cantidades de datos en instancias Amazon EC2 dinámicamente escalables.
- CloudSearch: servicio administrado en la nube de AWS que **facilita la configuració**n, la administración y el escalado rentables de una solución de búsqueda para su sitio web o aplicación.
- OpenSearch (Elasticsearch Service): Petabytes de texto y datos y facilita el uso y el escalado de **Elasticsearch** para el análisis de logs, la búsqueda de texto completo, la monitorización de aplicaciones y mucho más.
- Kinesis: Facilita la **recopilación, el procesamiento y el análisis de datos de streaming** en tiempo real para obtener conocimiento de manera oportuna y reaccionar rápidamente ante información nueva.
- Data Pipeline: Servicio web pensado para ayudarle a procesar datos y a transferirlos, de manera fiable y a intervalos definidos, entre diferentes servicios de almacenamiento e informática de AWS, así como entre orígenes de datos on-premise.
- QuickSight: El rápido y fácil BI (**Business Intelligence**) de amazon, a una décima parte del precio.
- AWS Glue: Servicio de extracción, transformación y carga **(ETL)** totalmente administrado que facilita a los clientes la preparación y carga de sus datos para su análisis, puede trabajar con Athena.

38.12. - Inteligencia Artificial - Machine Learning

- DeepRacer: Ofrece una **forma divertida** de comenzar a utilizar el aprendizaje por refuerzo (RL). El RL es una técnica avanzada de aprendizaje automático (ML) que adopta un enfoque muy diferente respecto de otros modelos de ML
- Lookout for Vision/Metrics/Equipment: Detecte defectos de productos utilizando visión artificial para automatizar la inspección de calidad.
- Textract: Servicio que **extrae automáticamente texto y datos de documentos** escaneados. Amazon Textract no se limita al simple reconocimiento óptico de caracteres (OCR), sino que también identifica el contenido de campos en formularios e información almacenada en tablas.
- Personalize: Personalización y **recomendaciones en tiempo real** basadas en la misma tecnología que se utiliza en Amazon.com
- Fraud Detector: Servicio completamente administrado que simplifica la tarea de **identificación de actividades en línea potencialmente fraudulentas**, como los fraudes de pagos en línea y la creación de cuentas falsas.
- CodeGuru: Servicio de aprendizaje automático para revisiones de código automatizadas y recomendaciones de rendimiento de aplicaciones. **Le ayuda a encontrar las líneas de código más "pesadas"** que perjudican el rendimiento de la aplicación.
- Kendra: Servicio de búsqueda empresarial altamente preciso y fácil de usar que funciona con **ML con capacidades de búsqueda en lenguaje natural** para sus sitios web y aplicaciones.
- Forecast: servicio administrado que utiliza ML para ofrecer pronósticos altamente precisos. Basado en la misma tecnología de **pronóstico de aprendizaje automático utilizada por Amazon.com**
- Lex: Servicio para crear interfaces de conversación en cualquier **aplicación con voz y texto**.
- Amazon Polly: Convierte **texto en habla realista**, lo que permite crear aplicaciones y categorías totalmente nuevas de productos con esta capacidad.
- Rekognition: Facilita la agregación de **análisis de imágenes** a sus aplicaciones. Puede detectar objetos, escenas, rostros; reconocer a famosos; e identificar contenido inapropiado en las imágenes.
- Tensorflow: Se ha convertido en una opción popular para la investigación de **aprendizaje profundo y desarrollo de aplicaciones**, en particular en áreas como la visión por computadora, la comprensión del lenguaje natural y la traducción del habla.
- Machine Learning: Sustituido o dentro del "mundo" de SageMaker
- Amazon Translate: Permite **traducir grandes volúmenes** de texto de manera sencilla y eficiente. Actualmente están disponibles 137 pares de idiomas.
- Amazon Transcribe: **Reconocimiento de voz automático** (ASR) y por supuesto con API .
- SageMaker : Cree, entrene e implemente modelos de **aprendizaje automático** a escala tambien dispone de SageMaker Autopilot
- DeepLens: La primera **cámara de vídeo** (Hardware) con aprendizaje profundo para programadores. Si quieres verla funcionando en Bilbao, llámame.

- Amazon Comprehend: Servicio de **procesamiento de lenguaje natural** (NLP) que usa el aprendizaje automático para encontrar información y relaciones en el texto
- Apache MXNet es un marco rápido de inferencia y entrenamiento de escala ajustable con un **API concisa y fácil de usar para aprendizaje automático**.
- Monitron: Es un servicio con sensórica, que usa ML para detectar anomalías en motores, ventiladores, Bombas, Compresores, etc
- Panorama: es un dispositivo de ML y un SDK (kit de desarrollo de software) que le permite agregar CV (video captura)

38.13. - Internet de las cosas (IoT)

- IoT Core: **Gestión principal** de Internet de las Cosas y conexión con el ecosistema de AWS.
- IoT 1 click: Permite a dispositivos sencillos activar funciones de **AWS Lambda** que pueden ejecutar una acción
- IoT Device Management: Incorpore, organice, monitorice y administre de manera remota dispositivos conectados a escala
- IoT Device Defender: Servicio completamente administrado que lo ayuda a **asegurar su flota de dispositivos de IoT**.
- FreeRTOS: Sistema Operativo con funciones **IoT para micro-controladores**
- AWS Greengrass: Es software que le permite ejecutar capacidades de **informática local**, mensajería, almacenamiento de datos en caché y sincronización para dispositivos conectados de manera segura para IoT.
- IoT Analytics: **Análisis** para dispositivos compatibles IoT

38.14. - Centro de contacto / Atención al cliente

- Amazon Connect: Servicio de centro de contacto o centralita telefónica (**chat/voz**) basado en la nube con modalidad autoservicio que le permite a cualquier empresa ofrecer un mejor servicio al cliente con facilidad y a un costo menor.
- Simple Email Service: Plataforma de **envío y recepción de emails**, super-escalable.
- Pinpoint Facilita el **envío de mensajes** a usuarios directamente desde su aplicación o servicio de backend, así como ejecutar campañas focalizadas para mejorar la interacción de los usuarios.

38.15. - Desarrollo de juegos

- Amazon GameLift: Servicio administrado para implementar, utilizar y escalar **servidores de videojuegos** dedicados para videojuegos multi-jugador basados en sesiones.
- Amazon Lumberyard: Es un **motor de videojuegos** AAA gratuito con un profundo nivel de integración con AWS y Twitch. Puedes descargar gratis la aplicación para Windows
- Amazon GameOn: Es un set de **API para desarrolladores de juegos** que permite tener a tus jugadores más entretenidos o conectado con el comercio electrónico de amazon para que los jugadores ganen premios.

- GameSparks: La forma sencilla de construir, desplegar y escalar el backends de los juegos.

38.16. - Servicios móviles

- Mobile Hub: Simplifica el proceso de construir, probar y **monitorizar aplicaciones móviles** que usan servicios de AWS.
- Device Farm: Mejore la calidad de las aplicaciones **iOS, Android y web probándolas** en dispositivos móviles reales en la nube de AWS.
- Mobile Analytics: Recopile, visualice y exporte **análisis de la aplicación**.
- AppSync: **Crea aplicaciones guiadas** por datos con capacidades en tiempo real y sin conexión.
- AWS Wavelength: Entregue aplicaciones con una latencia extremadamente baja para los dispositivos 5G

38.17. - AR / VR Realidad Aumentada / Realidad Virtual

- Sumerian: Crear **experiencias tridimensionales**, de VR y AR.

38.18. - Integración de aplicaciones

- Step Functions: Cree aplicaciones distribuidas con **flujos de trabajo visuales**.
- MQ: Servicio de **agente de mensajes** administrado para Apache ActiveMQ
- Simple Queue Service: **Colas de mensajes** completamente administradas para micro-servicios, sistemas distribuidos y aplicaciones sin servidor.
- Simple Notification Service: **Notificaciones móviles** y mensajes de publicación/suscripción para micro-servicios, sistemas distribuidos y aplicaciones sin servidor.
- SWF: Ayuda a los desarrolladores a diseñar, ejecutar y escalar trabajos de fondo que siguen **pasos paralelos o secuenciales**.

38.19. - Productividad Empresarial

- HoneyCode: Haz aplicaciones rápidas 100% administradas como un gestor de tareas, proyectos, TO-DOs, sin programar.
- Alexa para negocios: El "SIRI" de Amazon se llama **Alexa** y ahora está especializado para ayuda en el trabajo, reservas en el calendario, etc...:
- WorkDocs: Servicio empresarial seguro de **almacenamiento y uso compartido totalmente** administrado con controles administrativos estrictos y funciones de comentarios que mejoran la productividad de los usuarios, tipo dropbox.
- WorkMail: **Servicio de email y calendario empresarial** tipo webmail, seguro y administrado que soporta las aplicaciones clientes de email para dispositivos móviles y de escritorio existentes. pop3 y smtp.
- Amazon Chime: **Vídeo Conferencia** de amazon, alternativa a Skype.

- Amazon WorkLink: los empleados pueden obtener **acceso al contenido de un sitio web interno** con la misma facilidad que a cualquier sitio público, sin las molestias de conectarse a una red corporativa vpn.
- Nimble Studio: Permite a los estudios creativos producir efectos visuales, animación y contenido interactivo completamente en la nube.
- QnABot. Crea un chatBot multicanal, voz, SMS, chat para mejorar la experiencia del cliente.

38.20. - Escritorio y transmisión de aplicaciones

- WorkSpaces: Solución de **escritorio virtual** (Windows 7, Windows 10 o Amazon Linux 2) como servicio (DaaS) segura y completamente administrada que se ejecuta en AWS.
- AppStream 2.0: Permite enviar las **aplicaciones de Windows a cualquier dispositivo** por web, de forma desatendida.

38.21. - Blockchain

- Amazon Blockchain: Servicio totalmente gestionado que facilita la creación y gestión de redes de **blockchain** escalables utilizando los populares frameworks de código abierto Hyperledger Fabric and Ethereum.

38.22. - Robótica

- RoboMaker: Facilitan el desarrollo, las pruebas y la implementación de **aplicaciones de robótica inteligentes** a gran escala. Con RoboMaker se extiende el marco de software robótico de código abierto más utilizado, Robot Operating System (ROS, Sistema operativo de robots), con conectividad a servicios en la nube

38.23. - Licenciamiento

- License Manager: **Gestiona, Descubre y reporta las licencias** de Microsoft, SAP, etc. en AWS y on-premise

38.24. - Enlaces de interés:

- **Comparador** de los servicios Cloud en http://comparecloud.in/ (AWS / Azure / Google / IBM Cloud / Oracle / AlibaCloud / Huawei Cloud) de nuestro amigo Ilyas F
- Vídeo TOP 50 Servicios AWS en 10 minutos
- Infraestructura AWS de forma muy gráfica.
- pingaws.cdm.guru mide la **latencia de todas las regiones** de AWS desde tu ubicación, con un ping.
- Grupo de usuarios Meetup de AWS en **Bilbao (Spain)** bilbaws.com y otros grupos por el mundo:
- RSS de AWS: https://aws.amazon.com/es/about-aws/whats-new/recent/feed/