

AWS Certified Developer Associate

Por Stéphane Maarek y Joan Amengual



CURSO →



EXÁMENES PRÁCTICOS →

Descargo de responsabilidad: **Estas diapositivas están protegidas por derechos de autor y son estrictamente para uso personal**

- Este documento está reservado a las personas inscritas en el curso de [AWS Certified Developer Associate](#)
- **Por favor, no compartas este documento**, está destinado únicamente a uso personal y a la preparación de exámenes, gracias.
- Si has obtenido estas diapositivas de forma gratuita en un sitio web que no es el del curso, por favor, ponte en contacto con joan@blockstellart.com. ¡Gracias!
- **¡Mucha suerte para el examen y feliz aprendizaje!**

Curso de AWS Certified Developer Associate

¡Bienvenidos! Empezamos en 5 minutos



- Vamos a preparar el examen de **AWS Certified Developer - Associate**
- Es una certificación exigente, por lo que este curso será largo e interesante
- Cubriremos **más de 30 servicios de AWS**
- AWS / IT ¡Principiantes bienvenidos! (pero tómate tu tiempo, no es una carrera)
- No necesitas ser desarrollador para aprobar este examen
- **Aunque hayas hecho AWS Certified Solutions Architect, no te saltes las clases.**

¿Qué es AWS?



- AWS (Amazon Web Services) es un proveedor de Cloud
- Te proporcionan servidores y servicios que puedes utilizar bajo demanda y escalar fácilmente
- AWS ha revolucionado la IT a lo largo del tiempo
- AWS impulsa algunos de los mayores sitios web del mundo
 - Amazon.com
 - Netflix

Lo que aprenderemos en este curso



Amazon EC2



Amazon ECR



Amazon ECS



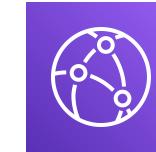
AWS Elastic Beanstalk



AWS Lambda



Elastic Load Balancing



Amazon CloudFront



Amazon Kinesis



Amazon Route 53



Amazon S3



Amazon RDS



Amazon Aurora



Amazon DynamoDB



Amazon ElastiCache



Amazon SQS



Amazon SNS



AWS Step Functions



Auto Scaling



Amazon API Gateway



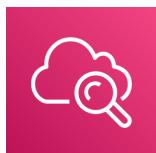
Amazon SES



Amazon Cognito



IAM



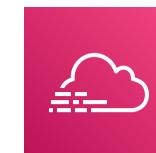
Amazon CloudWatch



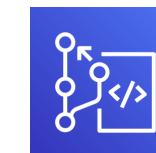
Amazon EC2 Systems Manager



AWS CloudFormation



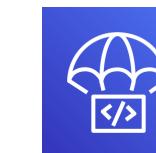
AWS CloudTrail



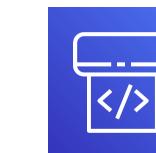
AWS CodeCommit



AWS CodeBuild



AWS CodeDeploy



AWS CodePipeline



AWS X-Ray



AWS KMS

Navegando por el plato de espaguetis de AWS



Sobre nosotros

- **¡Stephane Maarek!**
- Ha trabajado como consultor de IT y arquitecto de soluciones de AWS, desarrollador y SysOps
- Ha trabajado con AWS durante muchos años: ha construido sitios web, aplicaciones, plataformas de streaming
- Instructor veterano en AWS (Certificaciones, CloudFormation, Lambda, EC2...)



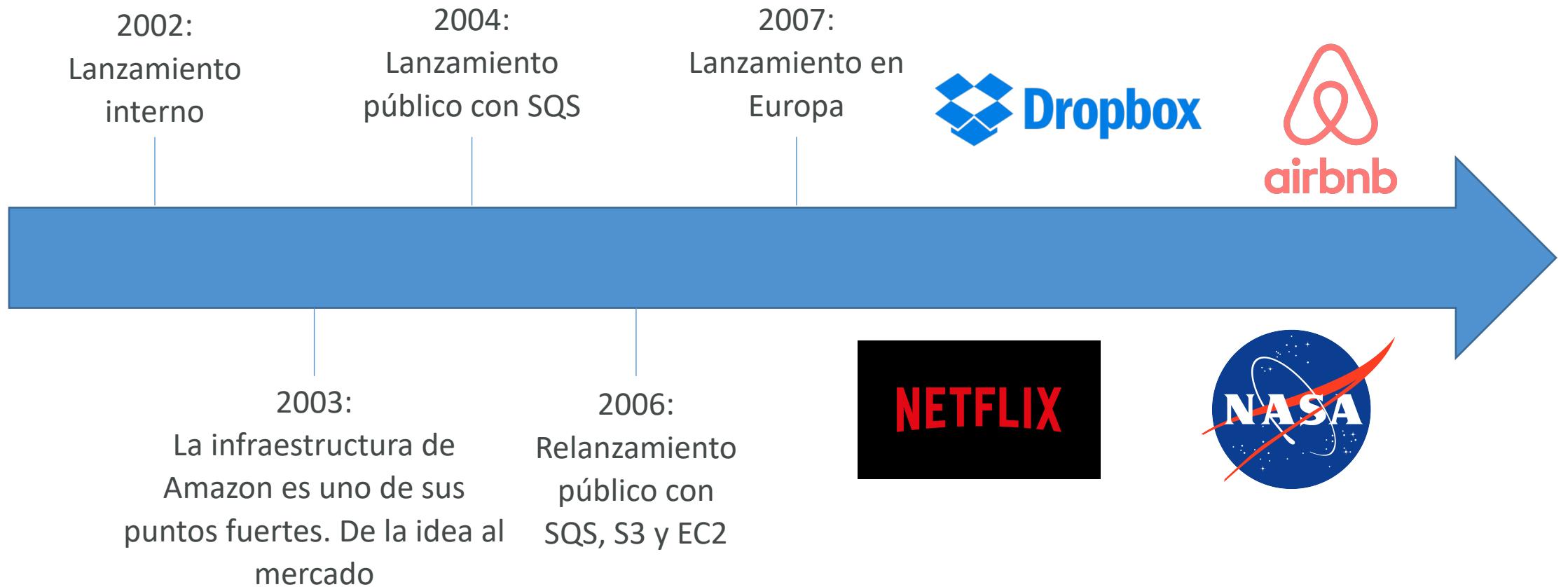
Sobre nosotros

- **¡Joan Amengual!**
- Ingeniero Full Stack en una empresa tecnológica en Silicon Valley, USA
- He trabajado con AWS varios años en diversas empresas para la migración y el escalado de servicios en el Cloud
- Premiado como Joven Talento en Ingeniería
- Puedes encontrarme en:
 - LinkedIn: <https://www.linkedin.com/in/joanamengual7>
 - Frecuentemente hago publicaciones interesantes sobre AWS



Cómo empezar con AWS

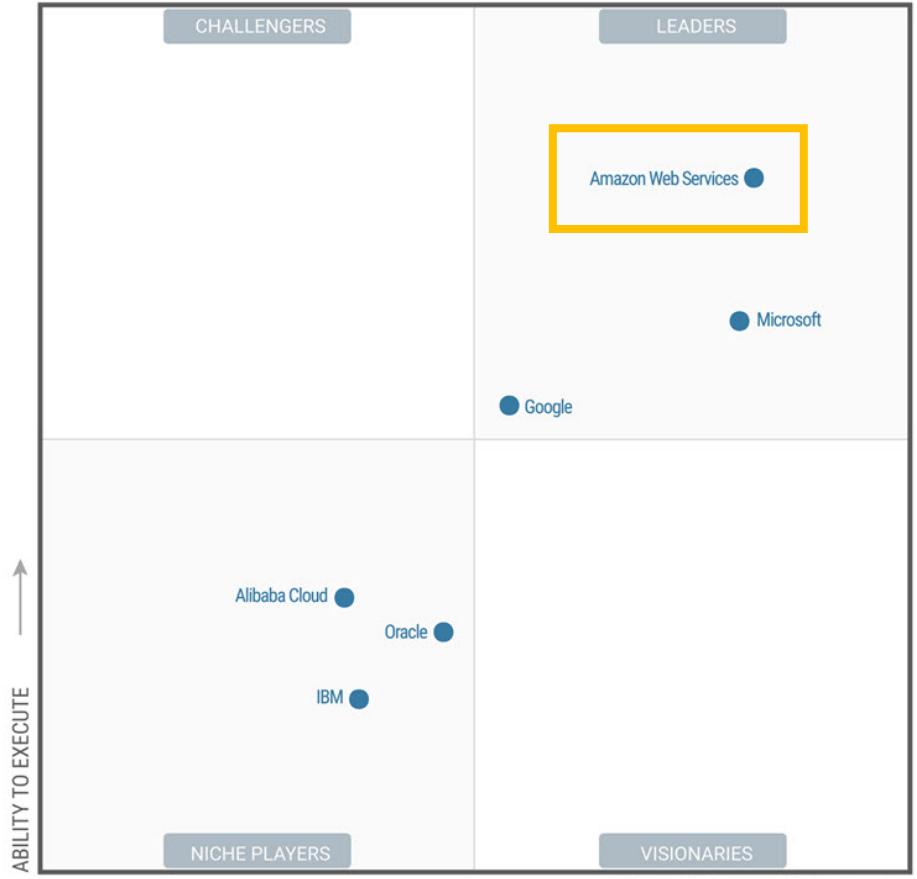
Historia del Cloud de AWS



Números de AWS Cloud

- En 2019, AWS tuvo 35.020 millones de dólares de ingresos anuales
- AWS representa el 47% del mercado en 2019 (Microsoft es el segundo con el 22%)
- Pionero y líder del mercado de Cloud de AWS por noveno año consecutivo
- Más de 1.000.000 de usuarios activos

Figure 1. Magic Quadrant for Cloud Infrastructure as a Service, Worldwide



Source: Gartner (July 2019)

Cuadrante mágico de Gartner

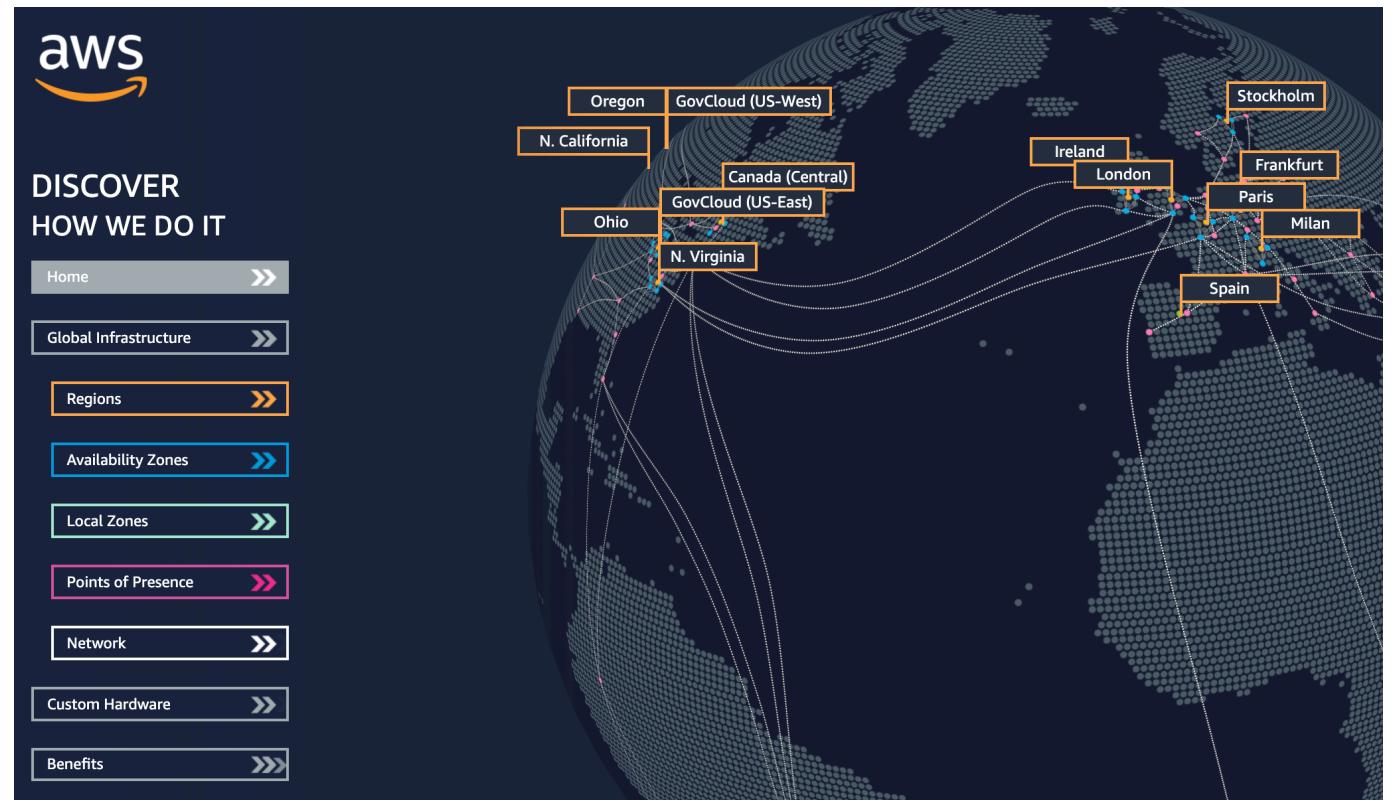
Casos de uso del Cloud de AWS

- AWS permite crear aplicaciones sofisticadas y escalables
- Aplicable a un conjunto diverso de industrias
- Los casos de uso incluyen
 - Enterprise IT, copias de seguridad y almacenamiento, análisis de Big Data
 - Alojamiento de sitios web, aplicaciones móviles y sociales
 - Juegos



Infraestructura global de AWS

- AWS Regions
- Regiones de AWS
- AWS Availability Zones
- Zonas de disponibilidad de AWS
- AWS Data Centers
- Centros de datos de AWS
- AWS Edge Locations / Points of Presence
- Puntos de presencia de AWS
- <https://infrastructure.aws/>



Regiones de AWS

- AWS tiene **Regiones** en todo el mundo
- Los nombres pueden ser us-east-1, eu-west-3...
- Una región es un **grupo de centros de datos**
- **La mayoría de los servicios de AWS son de ámbito regional**



<https://aws.amazon.com/about-aws/global-infrastructure/>

US East (N. Virginia) us-east-1

US East (Ohio) us-east-2

US West (N. California) us-west-1

US West (Oregon) us-west-2

Africa (Cape Town) af-south-1

Asia Pacific (Hong Kong) ap-east-1

Asia Pacific (Mumbai) ap-south-1

Asia Pacific (Seoul) ap-northeast-2

Asia Pacific (Singapore) ap-southeast-1

Asia Pacific (Sydney) ap-southeast-2

Asia Pacific (Tokyo) ap-northeast-1

Canada (Central) ca-central-1

Europe (Frankfurt) eu-central-1

Europe (Ireland) eu-west-1

Europe (London) eu-west-2

Europe (Paris) eu-west-3

Europe (Stockholm) eu-north-1

Middle East (Bahrain) me-south-1

South America (São Paulo) sa-east-1

¿Cómo elegir una región de AWS?

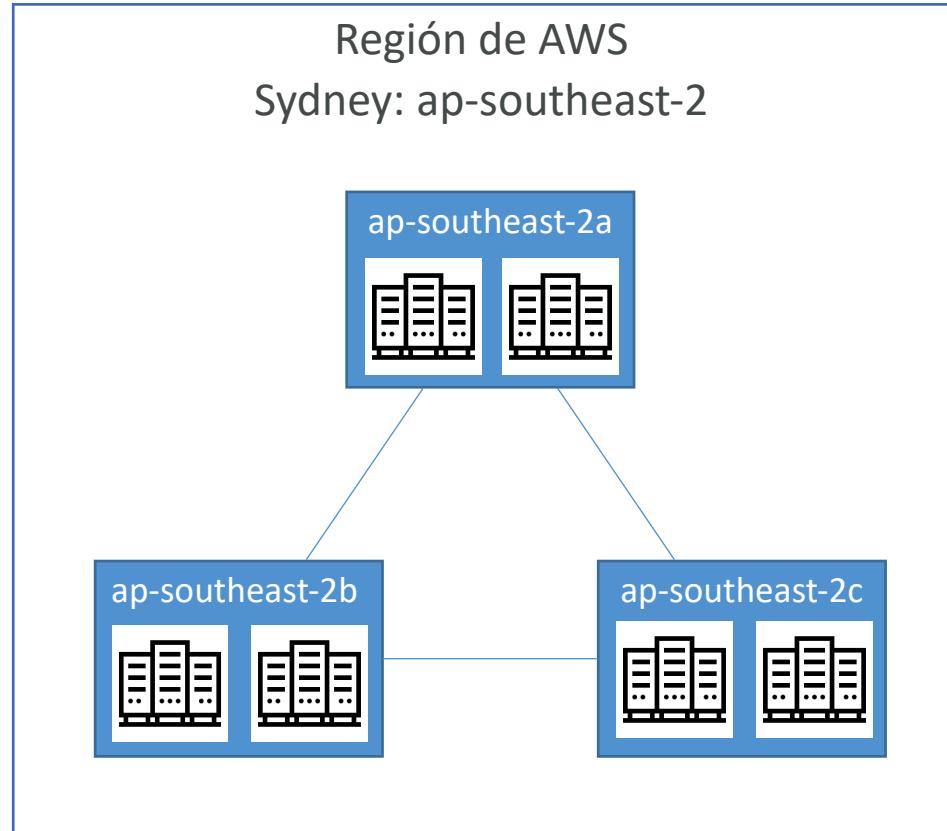
Si necesitas lanzar una nueva aplicación,
¿dónde debes hacerlo?



- **Cumplimiento de los requisitos legales y de gobernanza de datos:** los datos nunca salen de una región sin tu permiso explícito
- **Proximidad a los clientes:** latencia reducida
- **Servicios disponibles en una región:** los nuevos servicios y las nuevas funciones no están disponibles en todas las regiones
- **Precios:** los precios varían de una región a otra y son transparentes en la página de precios del servicio

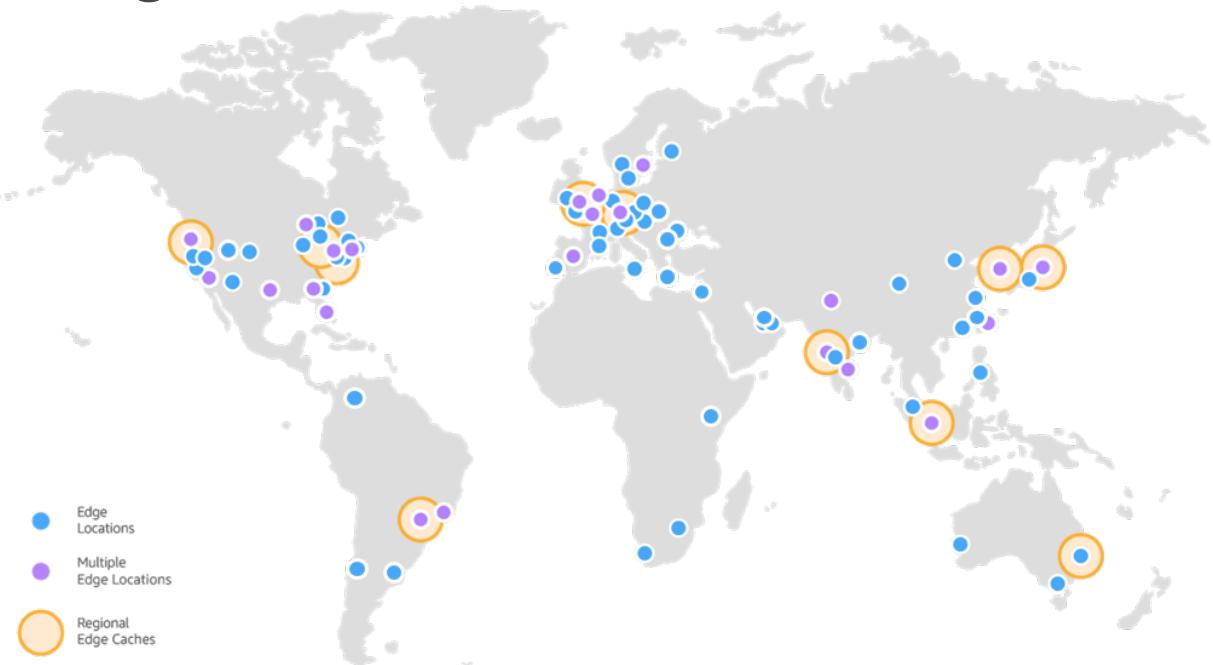
Zonas de disponibilidad de AWS

- Cada región tiene muchas zonas de disponibilidad (normalmente 3, el mínimo es 3, el máximo es 6).
Ejemplo:
 - ap-sudeste-2a
 - ap-sudeste-2b
 - ap-sudeste-2c
- Cada zona de disponibilidad (AZ) es uno o varios centros de datos discretos con alimentación, red y conectividad redundantes
- Están separadas unas de otras, de modo que están aisladas de las catástrofes
- Están conectadas con redes de alto ancho de banda y latencia ultrabaja



Puntos de presencia de AWS

- Amazon tiene 216 puntos de presencia (205 puntos de presencia y 11 cachés regionales) en 84 ciudades de 42 países
- El contenido se entrega a los usuarios finales con menor latencia



<https://aws.amazon.com/cloudfront/features/>

Tour por la consola de AWS



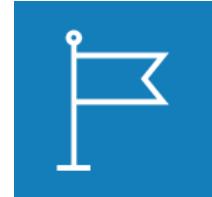
- **AWS cuenta con servicios globales:**

- Identity and Access Management (IAM)
- Route 53 (servicio DNS)
- CloudFront (Red de entrega de contenido)
- WAF (Firewall de aplicaciones web)



- **La mayoría de los servicios de AWS son de ámbito regional:**

- Amazon EC2 (Infraestructura como servicio)
- Elastic Beanstalk (Plataforma como servicio)
- Lambda (Función como servicio)
- Rekognition (Software como servicio)



- **Tabla de regiones:** <https://aws.amazon.com/about-aws/global-infrastructure/regional-product-services>

IAM

IAM: Usuarios y Grupos



- IAM = Identity and Access Management, servicio **global**
- **Cuenta root / raíz** creada por defecto, no debe ser utilizada ni compartida
- Los **usuarios** son personas dentro de tu organización, y pueden ser agrupados
- Los **grupos** sólo contienen usuarios, no otros grupos
- Los usuarios no tienen que pertenecer a un grupo, y el usuario puede pertenecer a varios grupos



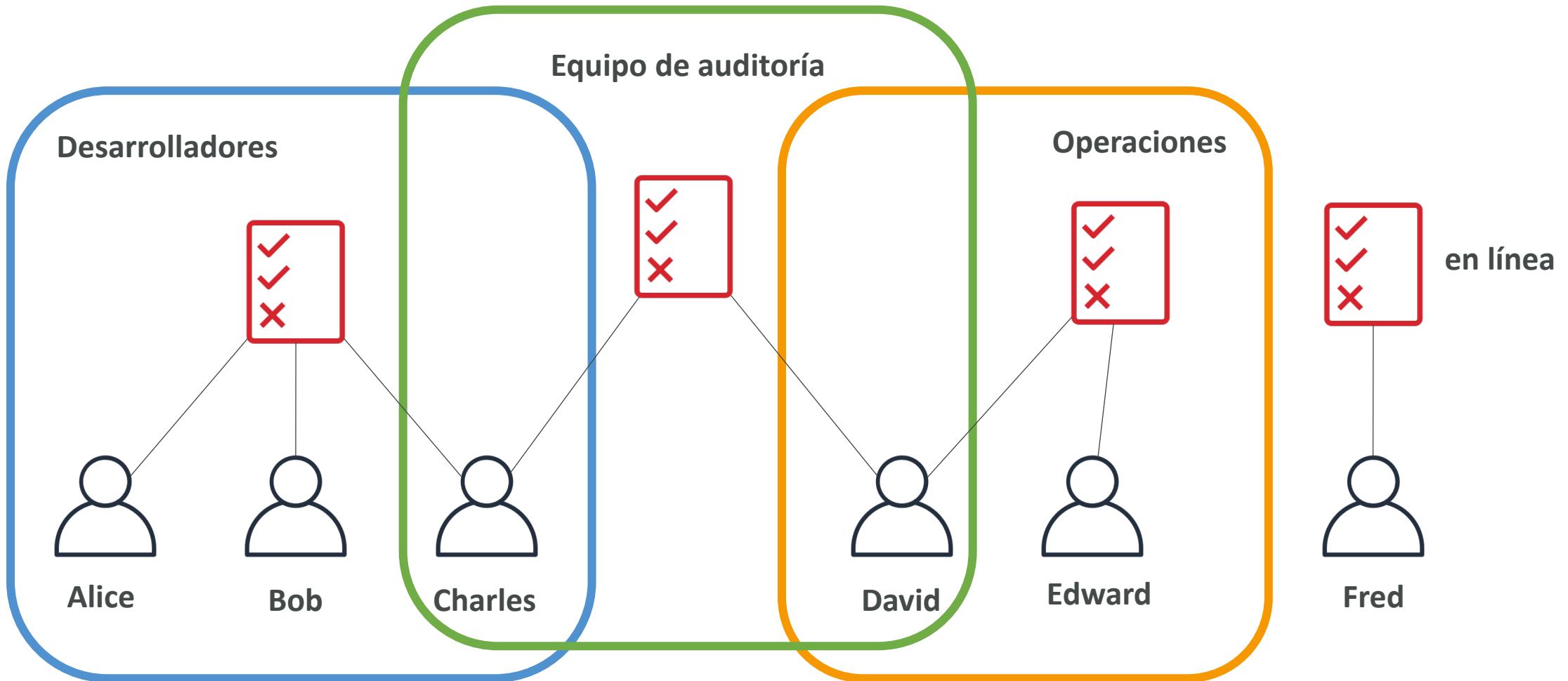
IAM: Permisos

- A los **usuarios o grupos** se les pueden asignar documentos JSON llamados políticas
- Estas políticas definen los **permisos** de los usuarios
- En AWS se aplica el **principio de mínimo privilegio**: no dar más permisos de los que un usuario necesita

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Effect": "Allow",  
            "Action": "ec2:Describe*",  
            "Resource": "*"  
        },  
        {  
            "Effect": "Allow",  
            "Action": "elasticloadbalancing:Describe*",  
            "Resource": "*"  
        },  
        {  
            "Effect": "Allow",  
            "Action": [  
                "cloudwatch>ListMetrics",  
                "cloudwatch:GetMetricStatistics",  
                "cloudwatch:Describe*"  
            ],  
            "Resource": "*"  
        }  
    ]  
}
```



Herencia de políticas IAM



Estructura de las políticas IAM

- Consta de:
 - **Version**: versión del lenguaje de la política, siempre incluye "2012-10-17"
 - **Id**: un identificador para la política (opcional)
 - **Statement**: una o más declaraciones individuales (obligatorio)
- Las declaraciones constan de:
 - **Sid**: un identificador para la declaración (opcional)
 - **Effect**: si la sentencia permite o deniega el acceso (Permitir, Denegar)
 - **Principal**: cuenta/usuario/rol al que se aplica esta política
 - **Action**: lista de acciones que esta política permite o deniega
 - **Resource**: lista de recursos a los que se aplican las acciones
 - **Condition**: condiciones para cuando esta política está en efecto (opcional)

```
{  
  "Version": "2012-10-17",  
  "Id": "S3-Account-Permissions",  
  "Statement": [  
    {  
      "Sid": "1",  
      "Effect": "Allow",  
      "Principal": {  
        "AWS": ["arn:aws:iam::123456789012:root"]  
      },  
      "Action": [  
        "s3:GetObject",  
        "s3:PutObject"  
      ],  
      "Resource": ["arn:aws:s3:::mybucket/*"]  
    }  
  ]  
}
```

IAM - Política de contraseñas

- Contraseñas fuertes = mayor seguridad para tu cuenta
- En AWS, puedes configurar una política de contraseñas:
 - Establecer una longitud mínima de contraseña
 - Requerir tipos de caracteres específicos:
 - incluyendo letras mayúsculas
 - letras minúsculas
 - números
 - caracteres no alfanuméricos
 - Permitir a todos los usuarios de IAM cambiar sus propias contraseñas
 - Requerir a los usuarios que cambien su contraseña después de un tiempo (caducidad de la contraseña)
 - Impedir la reutilización de la contraseña

Multi Factor Authentication - MFA



- Los usuarios tienen acceso a tu cuenta y posiblemente pueden cambiar configuraciones o eliminar recursos en tu cuenta de AWS
- **Quieres proteger tus cuentas root y los usuarios de IAM**
- MFA = contraseña que conoces + dispositivo de seguridad que posees



Contraseña +



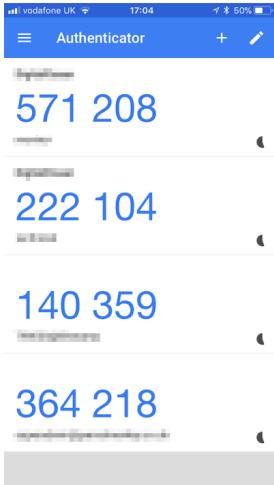
=>

Login exitoso

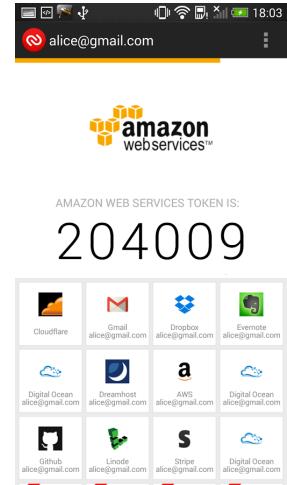
- **Principal beneficio de MFA:** si una contraseña es robada o hackeada, la cuenta no se ve comprometida

Opciones de dispositivos MFA en AWS

Dispositivo virtual MFA



Autenticador de Google
(sólo en el teléfono)



Authy
(multi-dispositivo)

Soporte para múltiples tokens en un solo dispositivo.

Clave de seguridad del segundo factor universal (U2F)

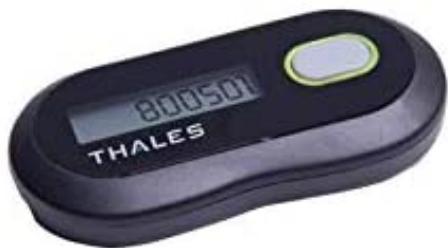


YubiKey de Yubico (3rd party)

Soporte para múltiples usuarios root e IAM utilizando una única clave de seguridad

Opciones de dispositivos MFA en AWS

Dispositivo MFA de llavero por hardware



Proporcionado por Gemalto (3rd party)

Dispositivo MFA de llavero por hardware para AWS GovCloud (US)



Proporcionado por SurePassID (3rd party)

¿Cómo pueden los usuarios acceder a AWS?



- Para acceder a AWS, tienes tres opciones:
 - **Consola de administración de AWS** (protegida por contraseña + MFA)
 - **Interfaz de línea de comandos de AWS (CLI)**: protegida por claves de acceso
 - **AWS Software Developer Kit (SDK)** - para el código: protegido por claves de acceso
 - Las claves de acceso se generan a través de la consola de AWS
- Los usuarios gestionan sus propias claves de acceso
- **Las claves de acceso son secretas, como una contraseña. No las compartas**
- ID de la clave de acceso ~ = nombre de usuario
- Clave de acceso secreta ~ = contraseña

Ejemplo de claves de acceso (falsas)

Access keys

Use access keys to make secure REST or HTTP Query protocol requests to AWS service APIs. For your protection, you should never share your secret keys with anyone. As a best practice, we recommend frequent key rotation. [Learn more](#)

[Create access key](#)

Access key ID	Created	Last used	Status	
AKIASK4E37PV4TU3RD6C	2020-05-25 15:13 UTC+0100	N/A	Active	Make inactive X

- ID de la llave de acceso: AKIASK4E37PV4983d6C
- Clave de acceso secreta: AZPN3z0jWozWCndljhB0Unh8239a1bzBzO5fqkZq
- **Recuerda: no compartas tus claves de acceso**

¿Qué es la CLI de AWS?

- Una herramienta que permite interactuar con los servicios de AWS mediante comandos en tu shell de línea de comandos
- Acceso directo a las API públicas de los servicios de AWS
- Puedes desarrollar scripts para gestionar tus recursos
- Es de código abierto <https://github.com/aws/aws-cli>
- Alternativa al uso de la consola de administración de AWS

```
→ ~ aws s3 cp myfile.txt s3://ccp-mybucket/myfile.txt
upload: ./myfile.txt to s3://ccp-mybucket/myfile.txt
→ ~ aws s3 ls s3://ccp-mybucket
2021-05-14 03:22:52          0 myfile.txt
→ ~ █
```

¿Qué es el SDK de AWS?



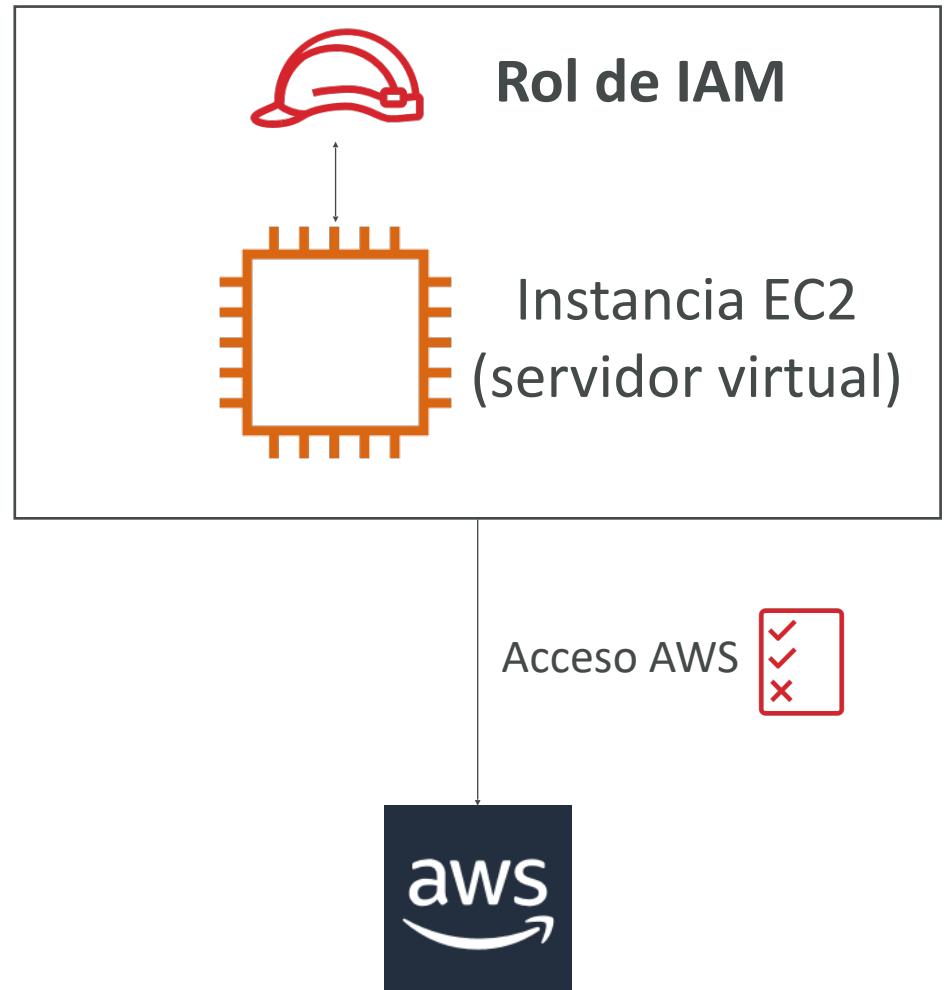
- Kit de desarrollo de software de AWS (AWS SDK)
- APIs específicas para cada lenguaje (conjunto de bibliotecas)
- Permite acceder y administrar los servicios de AWS mediante programación
- Integrado en la aplicación
- Admite:
 - SDKs (JavaScript, Python, PHP, .NET, Ruby, Java, Go, Node.js, C++)
 - SDKs para móviles (Android, iOS, ...)
 - SDKs para dispositivos IoT (Embedded C, Arduino, ...)
- Ejemplo: AWS CLI está construido sobre AWS SDK para Python



Tu aplicación

Roles IAM para los servicios

- Algún servicio de AWS tendrá que realizar acciones en tu nombre
- Para ello, asignaremos **permisos** a los servicios de AWS con **Roles IAM**
- Roles comunes:
 - Roles de Instancia EC2
 - Roles de la función Lambda
 - Roles para CloudFormation



Herramientas de seguridad IAM

- **IAM Credentials Report / Informe de credenciales de IAM (a nivel de cuenta)**
 - Un informe que enumera todos los usuarios de tu cuenta y el estado de tus diversas credenciales
- **IAM Access Advisor / Asesor de acceso de IAM (a nivel de usuario)**
 - Muestra los permisos de servicio concedidos a un usuario y cuando se accedió a esos servicios por última vez
 - Puedes utilizar esta información para revisar tus políticas

Directrices y buenas prácticas de IAM



- No utilices la cuenta root excepto para la configuración de la cuenta AWS
- Un usuario físico = Un usuario AWS
- **Asignar usuarios a grupos** y asignar permisos a grupos
- Crear una **política de contraseñas fuerte**
- Utilizar y reforzar el uso de la **autenticación multifactor (MFA)**
- Crear y utilizar **Roles** para dar permisos a los servicios de AWS
- Utilizar claves de acceso para el acceso programático (CLI / SDK)
- Revisar los permisos de tu cuenta con el informe de credenciales de IAM
- **No compartir nunca los usuarios de IAM ni las claves de acceso**

Modelo de responsabilidad compartida para IAM



Tú

- Infraestructura (seguridad de la red global)
- Análisis de configuración y vulnerabilidad
- Validación de la conformidad
- Gestión y supervisión de usuarios, grupos, roles y políticas
- Habilitar MFA en todas las cuentas
- Rota todas tus claves con frecuencia
- Utiliza las herramientas IAM para aplicar los permisos adecuados
- Analiza los patrones de acceso y revisa los permisos

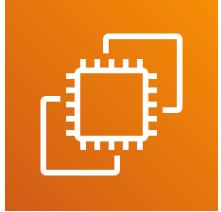
Resumen - IAM



- **Usuarios:** mapeado a un usuario físico, tiene una contraseña para la consola de AWS
- **Grupos:** contiene sólo usuarios
- **Políticas:** Documento JSON que describe los permisos para los usuarios o grupos
- **Roles:** para instancias EC2 o servicios AWS
- **Seguridad:** MFA + Política de contraseñas
- **AWS CLI:** gestiona tus servicios de AWS mediante la línea de comandos
- **AWS SDK:** gestiona tus servicios de AWS utilizando un lenguaje de programación
- **Claves de acceso:** accede a AWS mediante la CLI o el SDK
- **Auditoría:** Informes de credenciales de IAM y Asesor de acceso de IAM

EC2

Amazon EC2



- EC2 es una de las ofertas más populares de AWS
- EC2 = Elastic Compute Cloud = Infraestructura como servicio (IaaS)
- Consiste principalmente en la capacidad de :
 - Alquilar máquinas virtuales (EC2)
 - Almacenar datos en unidades virtuales (EBS)
 - Distribuir la carga entre las máquinas (ELB)
 - Escalar los servicios mediante un Auto Scaling Group (ASG) o también conocido en español como Grupo de Autoescalamiento
- Conocer EC2 es fundamental para entender el funcionamiento del Cloud

Opciones de tamaño y configuración de EC2

- Sistema operativo (**OS**): Linux, Windows o Mac OS
- Cuánta potencia de cálculo y núcleos (**CPU**)
- Cuánta memoria de acceso aleatorio (**RAM**)
- Cuánto espacio de almacenamiento:
 - Conectado a la red (**EBS y EFS**)
 - hardware (**EC2 Instance Store**)
- Tarjeta de red: velocidad de la tarjeta, dirección IP pública
- Reglas de firewall: **grupo de seguridad**
- Script de arranque (configurar en el primer lanzamiento): Datos de usuario de EC2

Datos del usuario de EC2

- Es posible arrancar nuestras instancias utilizando un script de datos de usuario de EC2.
- bootstrapping significa lanzar comandos cuando una máquina se inicia
- Ese script sólo se ejecuta una vez en el primer arranque de la instancia
- Los datos de usuario de EC2 se utilizan para automatizar tareas de arranque como:
 - Instalar actualizaciones
 - Instalación de software
 - Descarga de archivos comunes de Internet
 - Cualquier cosa que se te ocurra
- El script de datos de usuario de EC2 se ejecuta con el usuario root

Lanzamiento de una instancia EC2 con Linux

- Vamos a lanzar nuestro primer servidor virtual utilizando la consola de AWS
- Tendremos una primera aproximación de alto nivel a los distintos parámetros
- Veremos que nuestro servidor web se lanza utilizando los datos de usuario de EC2
- Aprenderemos a iniciar / parar / terminar nuestra instancia.

Tipos de instancias de EC2 - Visión general

- Puedes utilizar diferentes tipos de instancias EC2 optimizadas para diferentes casos de uso (<https://aws.amazon.com/ec2/instance-types/>)
- AWS tiene la siguiente convención de nombres:

m5.2xlarge

- m: clase de instancia
- 5: generación (AWS los mejora con el tiempo)
- 2xlarge: tamaño dentro de la clase de instancia

General Purpose
Compute Optimized
Memory Optimized
Accelerated Computing
Storage Optimized
Instance Features
Measuring Instance Performance

Tipos de instancias de EC2 - Propósito general

- Excelente para una diversidad de cargas de trabajo, como servidores web o repositorios de código
- Equilibrio entre:
 - Computación
 - Memoria
 - Red
- En el curso, utilizaremos la instancia t2.micro que es una instancia EC2 de propósito general

General Purpose

General purpose instances provide a balance of compute, memory and networking resources, and can be used for a variety of diverse workloads. These instances are ideal for applications that use these resources in equal proportions such as web servers and code repositories.

Mac	T4g	T3	T3a	T2	M6g	M5	M5a	M5n	M5zn	M4	A1
-----	-----	----	-----	----	-----	----	-----	-----	------	----	----

* Esta lista evolucionará con el tiempo, por favor, consulta el sitio web de AWS para obtener la información más reciente

Tipos de instancias EC2 - Computación optimizada

- Ideal para tareas de cálculo intensivo que requieren procesadores de alto rendimiento:
 - Cargas de trabajo de procesamiento por lotes
 - Transcodificación de medios
 - Servidores web de alto rendimiento
 - Computación de alto rendimiento (HPC)
 - Modelado científico y aprendizaje automático
 - Servidores dedicados a juegos

Compute Optimized

Compute Optimized instances are ideal for compute bound applications that benefit from high performance processors. Instances belonging to this family are well suited for batch processing workloads, media transcoding, high performance web servers, high performance computing (HPC), scientific modeling, dedicated gaming servers and ad server engines, machine learning inference and other compute intensive applications.

C6g C6gn C5 C5a C5n C4

* Esta lista evolucionará con el tiempo, por favor, consulta el sitio web de AWS para obtener la información más reciente

Tipos de instancias EC2 - Memoria optimizada

- Rápido rendimiento para cargas de trabajo que procesan grandes conjuntos de datos en memoria
- Casos de uso:
 - Alto rendimiento, bases de datos relacionales/no relacionales
 - Almacenes de caché distribuidos a escala web
 - Bases de datos en memoria optimizadas para BI (business intelligence)
 - Aplicaciones que realizan el procesamiento en tiempo real de grandes datos no estructurados

Memory Optimized

Memory optimized instances are designed to deliver fast performance for workloads that process large data sets in memory.

R6g

R5

R5a

R5b

R5n

R4

X1e

X1

High Memory

z1d

* Esta lista evolucionará con el tiempo, por favor, consulta el sitio web de AWS para obtener la información más reciente

Tipos de instancias EC2 - Almacenamiento optimizado

- Ideal para tareas de almacenamiento intensivo que requieran un acceso alto y secuencial de lectura y escritura a grandes conjuntos de datos en el almacenamiento local
- Casos de uso:
 - Sistemas de procesamiento de transacciones en línea (OLTP) de alta frecuencia
 - Bases de datos relacionales y NoSQL
 - Caché para bases de datos en memoria (por ejemplo, Redis)
 - Aplicaciones de almacenamiento de datos
 - Sistemas de archivos distribuidos

Storage Optimized

Storage optimized instances are designed for workloads that require high, sequential read and write access to very large data sets on local storage. They are optimized to deliver tens of thousands of low-latency, random I/O operations per second (IOPS) to applications.

I3	I3en	D2	D3	D3en	H1
----	------	----	----	------	----

* Esta lista evolucionará con el tiempo, por favor, consulta el sitio web de AWS para obtener la información más reciente

Tipos de instancias de EC2: ejemplo

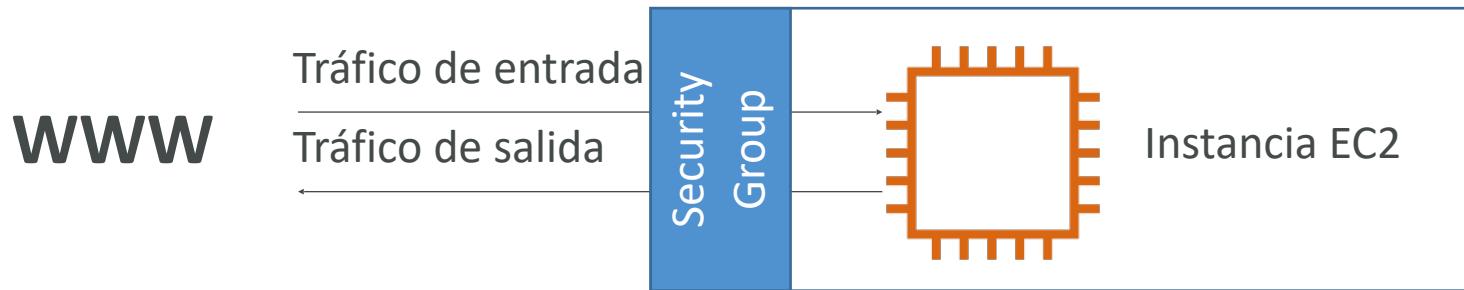
Instancia	vCPU	Mem (GiB)	Almacenamiento	Rendimiento de la red	Ancho de banda de EBS (Mbps)
t2.micro	1	1	Sólo EBS	Bajo a moderado	
t2.xlarge	4	16	Sólo EBS	Moderado	
c5d.4xlarge	16	32	1 x 400 NVMe SSD	Hasta 10 Gbps	4,750
r5.16xlarge	64	512	Sólo EBS	20 Gbps	13,600
m5.8xlarge	32	128	Sólo EBS	10 Gbps	6,800

t2.micro forma parte de la capa gratuita de AWS (hasta 750 horas al mes)

<https://instances.vantage.sh>

Introducción a los grupos de seguridad

- Los grupos de seguridad son la base de la seguridad de la red en AWS
- Controlan cómo se permite el tráfico dentro o fuera de nuestras Instancias EC2



- Los grupos de seguridad sólo contienen reglas de **permiso**
- Las reglas de los grupos de seguridad pueden hacer referencia por IP o por grupo de seguridad

Grupos de seguridad

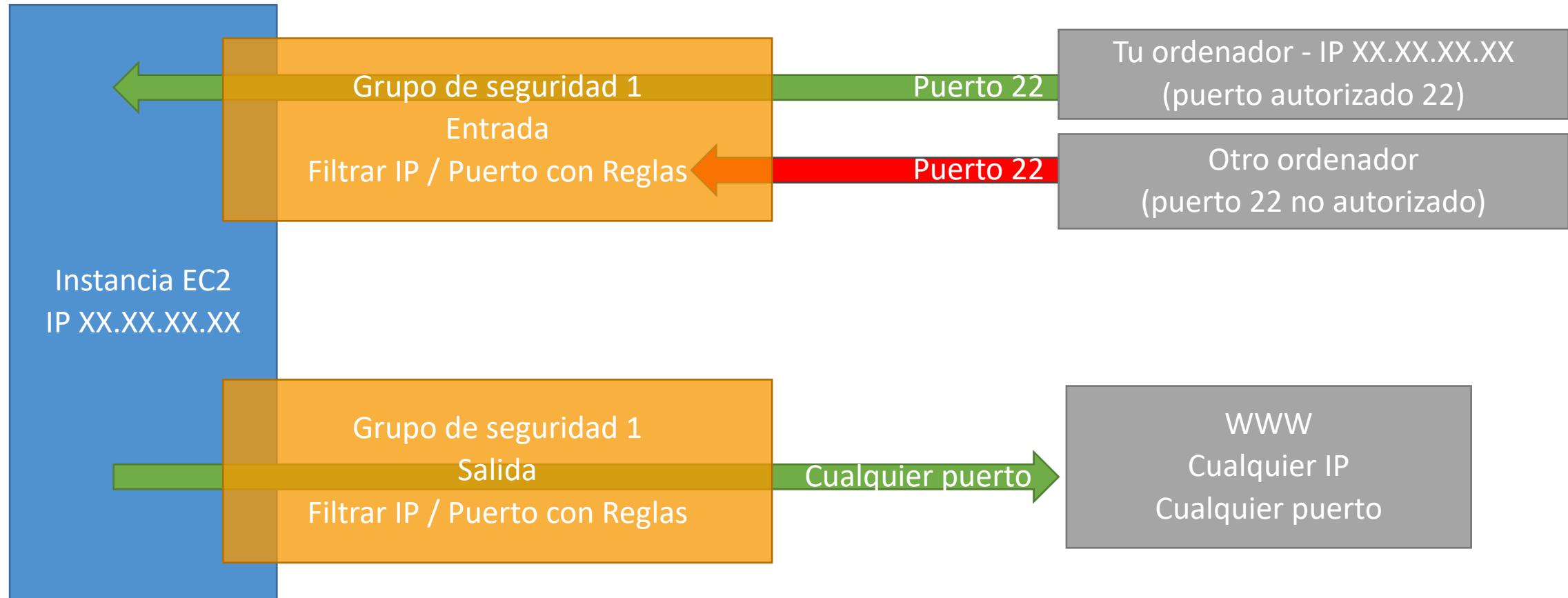
Inmersión más profunda

- Los grupos de seguridad actúan como un “firewall” en las instancias de EC2
- Regulan:
 - El acceso a los puertos
 - Rangos de IP autorizados - IPv4 e IPv6
 - Control de la red de entrada (de otros a la instancia)
 - Control de la red saliente (desde la instancia hacia otra)

Type	Protocol	Port Range	Source	Description
HTTP	TCP	80	0.0.0.0/0	test http page
SSH	TCP	22	122.149.196.85/32	
Custom TCP Rule	TCP	4567	0.0.0.0/0	java app

Grupos de seguridad

Diagrama



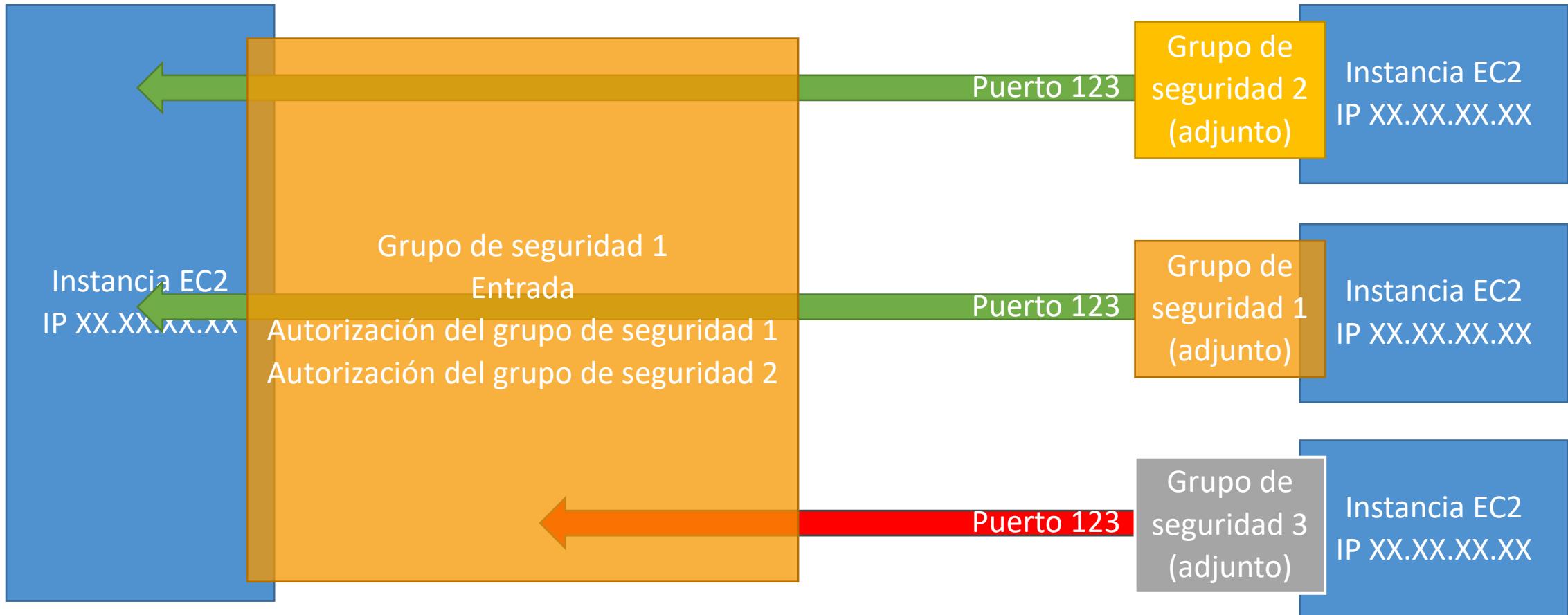
Grupos de seguridad

Es bueno saber

- Puede adjuntarse a múltiples instancias
- Bloqueado a una combinación de región / VPC
- Vive "fuera" del EC2 - si el tráfico está bloqueado, la instancia EC2 no lo verá
- Es bueno mantener un grupo de seguridad separado para el acceso SSH
- Si tu aplicación no es accesible (tiempo de espera), entonces es un problema de grupo de seguridad
- Si tu aplicación da un error de "conexión rechazada", entonces es un error de la aplicación o no se ha lanzado
- Todo el tráfico de entrada está **bloqueado** por defecto
- Todo el tráfico de salida está **autorizado** por defecto

Referencia a otros grupos de seguridad

Diagrama



Puertos clásicos que hay que conocer

- 22 = SSH (Secure Shell) - iniciar sesión en una instancia de Linux
- 21 = FTP (File Transfer Protocol) - subir archivos a un archivo compartido
- 22 = SFTP (Secure File Transfer Protocol) - subir archivos usando SSH
- 80 = HTTP - acceso a sitios web no seguros
- 443 = HTTPS - acceso a sitios web seguros
- 3389 = RDP (Remote Desktop Protocol) - iniciar sesión en una instancia de Windows

Tabla resumen SSH

	SSH	Putty	EC2 Instance Connect
Mac	✓		✓
Linux	✓		✓
Windows < 10		✓	✓
Windows >= 10	✓	✓	✓

Qué clases hay que ver

- **Mac / Linux:**
 - Clase de SSH en Mac/Linux
- **Windows:**
 - Clase sobre Putty
 - Si Windows 10: Clase sobre SSH en Windows 10
- **Todos los estudiantes:**
 - Clase de Instance Connect EC2

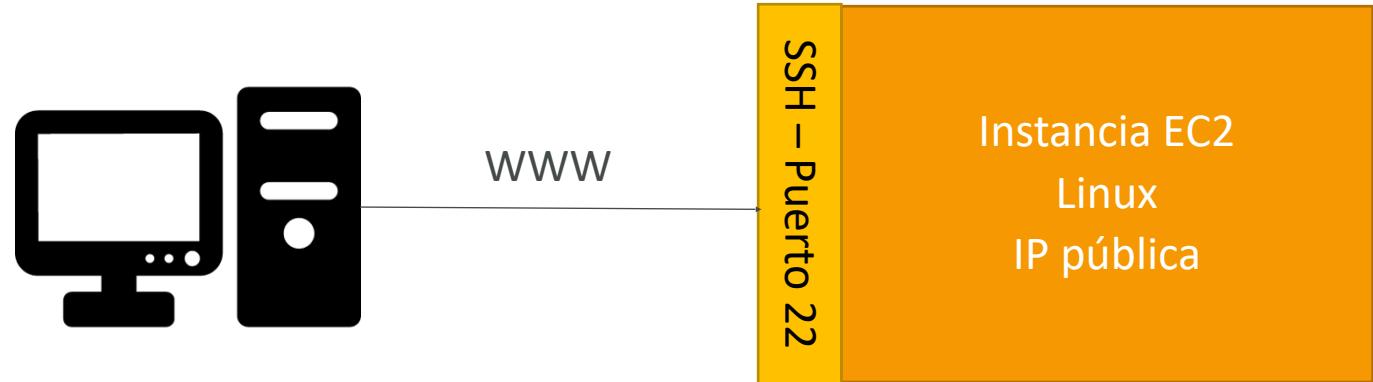
Solución de problemas de SSH

- **Los estudiantes son los que más problemas tienen con SSH**
- Si las cosas no funcionan...
 - Vuelve a ver la clase. Puede que te hayas perdido algo
 - Lee la guía de solución de problemas
 - Prueba con EC2 Instance Connect
- **Si uno de los métodos funciona (SSH, Putty o EC2 Instance Connect) estás bien**
- Si ningún método funciona, no pasa nada, el curso no utilizará mucho SSH

Cómo usar SSH en tu instancia EC2

Linux / Mac OS X

- Vamos a aprender cómo usar SSH en tu instancia EC2 usando [Linux / Mac](#)
- SSH es una de las funciones más importantes. Permite controlar una máquina remota, todo ello utilizando la línea de comandos.

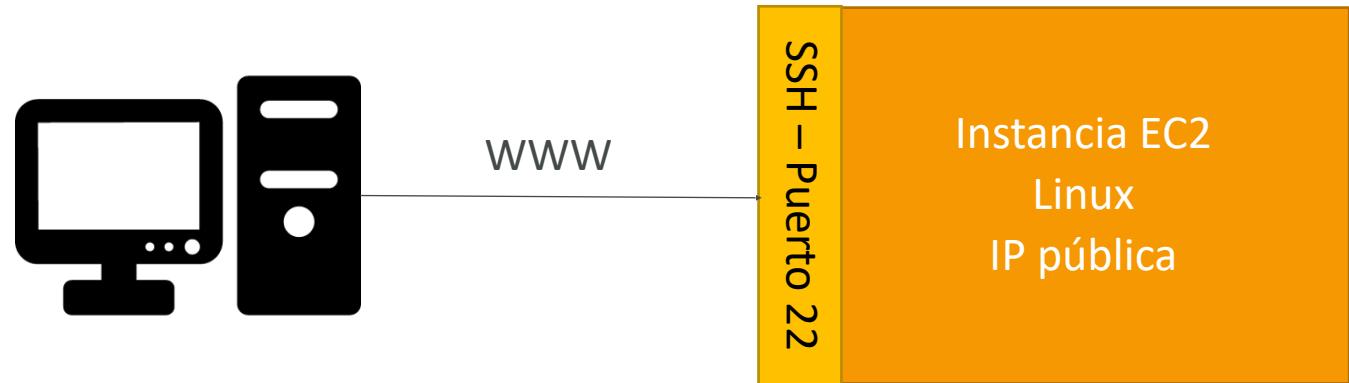


- Vamos a ver cómo podemos configurar OpenSSH [~/.ssh/config](#) para facilitar el SSH en nuestras instancias EC2

Cómo usar SSH en tu instancia EC2

Windows

- Vamos a aprender cómo usar SSH en tu instancia EC2 usando [Windows](#)
- SSH es una de las funciones más importantes. Permite controlar una máquina remota, todo ello utilizando la línea de comandos.



- Configuraremos todos los parámetros necesarios para hacer SSH en Windows utilizando la herramienta gratuita [Putty](#).

Instance Connect EC2

Conexión de instancias EC2

- Conéctate a tu instancia EC2 desde el navegador
- No es necesario utilizar el archivo de claves que se ha descargado
- La "magia" es que una clave temporal es cargada en EC2 por AWS
- **Funciona sólo out-of-the-box con Amazon Linux 2**
- Necesitas asegurarte de que el puerto 22 sigue abierto

Opciones de compra de instancias EC2

- **Instancias bajo demanda**: carga de trabajo corta, precio predecible, pago por segundos
- **Reservadas** (1 y 3 años)
 - **Instancias reservadas** - cargas de trabajo largas
 - **Instancias reservadas convertibles**: cargas de trabajo largas con instancias flexibles
- **Planes de ahorro** (1 y 3 años) - compromiso con una cantidad de uso, carga de trabajo larga
- **Instancias Spot** - cargas de trabajo cortas, baratas, pueden perder instancias (menos fiables)
- **Hosts dedicados**: reserva un servidor físico completo, controla la ubicación de las instancias
- **Instancias dedicadas** - ningún otro cliente compartirá tu hardware
- **Reservas de capacidad** - reserva de capacidad en una AZ específica para cualquier duración

EC2 bajo demanda

- Paga por lo que usas:
 - Linux o Windows - facturación por segundo, después del primer minuto
 - Todos los demás sistemas operativos: facturación por hora
 - Tiene el coste más elevado, pero no hay que pagar por adelantado
 - Sin compromiso a largo plazo
-
- Recomendado para **cargas de trabajo a corto plazo y sin interrupciones**, cuando no se puede predecir el comportamiento de la aplicación

Instancias reservadas de EC2

- Hasta un **72%** de descuento en comparación con el servicio bajo demanda
- Reserva de atributos de instancia específicos (**tipo de instancia, región, ocupación, sistema operativo**)
- **Periodo de reserva - 1 año** (+descuento) o **3 años** (+++descuento)
- **Opciones de pago - Sin pago inicial** (+), **Pago inicial parcial** (++) , **Pago inicial total** (+++)
- **Alcance de la instancia reservada** - Por **región** o por **zona** (capacidad de reserva en una AZ)
- Recomendado para aplicaciones de uso constante (piensa en una base de datos)
- Puedes comprar y vender en el Marketplace de instancias reservadas
- **Instancia reservada convertible:**
 - Puedes cambiar el tipo de instancia EC2, la familia de instancias, el SO, etc.
 - Hasta un **66%** de descuento

Nota: los % de descuento pueden ser diferentes a los del video ya que AWS los cambia con el tiempo - los números exactos no son necesarios para el examen. Esto es solo para fines ilustrativos ☺.

Planes de ahorro EC2

- Obtén un descuento basado en el uso a largo plazo (hasta el 72%)
- Comprométete a un determinado tipo de uso (10 \$/hora durante 1 o 3 años)
- El uso más allá de los planes de ahorro de EC2 se factura al precio bajo demanda
- Bloqueado a una familia de instancias específica y a una región de AWS (por ejemplo, M5 en us-east-1)
- Flexible a través de:
 - Tamaño de instancia (por ejemplo, m5.xlarge, m5.2xlarge)
 - Sistema operativo (por ejemplo, Linux, Windows)
 - Tenencia (Anfitrión, Dedicado, Por defecto)



Instancias EC2 Spot

- Puedes obtener un **descuento de hasta el 90%** en comparación con la demanda
- Instancias que puedes "perder" en cualquier momento si su precio máximo es inferior al precio spot actual
- Las instancias **MÁS rentables** de AWS
- **Útil para las cargas de trabajo que son resistentes a los fallos**
 - Trabajos por lotes (Batch Jobs)
 - Análisis de datos
 - Procesamiento de imágenes
 - Cualquier carga de trabajo distribuida
 - Cargas de trabajo con una hora de inicio y finalización flexible
- **No es adecuado para trabajos críticos o bases de datos**

Hosts dedicados EC2

- Un servidor físico con capacidad de instancia EC2 totalmente dedicado a su uso
- Permite abordar los requisitos de **normativas y utilizar licencias de software vinculadas al servidor existentes** (licencias de software por socket, por núcleo, por VM)
- Opciones de compra:
 - **Bajo demanda** - pago por segundo para el host dedicado activo
 - **Reservado** - 1 o 3 años (sin pago inicial, pago inicial parcial, pago inicial total)
- La opción más cara
- Útil para el software que tiene un modelo de licencia complicado (BYOL - Bring Your Own License)
- O para empresas que tienen fuertes necesidades de regulación o cumplimiento

Instancias dedicadas de EC2

- Las instancias se ejecutan en un hardware dedicado para ti
- Puedes compartir el hardware con otras instancias de la misma cuenta
- No hay control sobre la ubicación de las instancias (se puede mover el hardware después de la parada/arranque)

Characteristic	Dedicated Instances	Dedicated Hosts
Enables the use of dedicated physical servers	X	X
Per instance billing (subject to a \$2 per region fee)	X	
Per host billing		X
Visibility of sockets, cores, host ID		X
Affinity between a host and instance		X
Targeted instance placement		X
Automatic instance placement	X	X
Add capacity using an allocation request		X

Reservas de capacidad de EC2

- Reserva la capacidad de las instancias **bajo demanda** en una AZ específica para cualquier duración
- Siempre tendrás acceso a la capacidad de EC2 cuando la necesites
- **Sin compromiso de tiempo** (crear/cancelar en cualquier momento), **sin descuentos de facturación**
- Combina con las instancias regionales reservadas y los planes de ahorro para beneficiarte de descuentos en la facturación
- Se te cobra la tarifa bajo demanda tanto si ejecuta instancias como si no

- Adecuado para cargas de trabajo ininterrumpidas a corto plazo que necesitan estar en una AZ específica

¿Qué opción de compra me conviene?



- **Bajo demanda (On demand)**: venir y quedarse en el complejo cuando queramos, pagamos el precio completo
- **Reservada (Reserved)**: cómo planificar con antelación y si planeamos quedarnos durante mucho tiempo, podemos obtener un buen descuento
- **Planes de ahorro (Savings Plans)**: pagamos una cantidad por hora durante un periodo determinado y nos alojamos en cualquier tipo de habitación (por ejemplo, King, Suite, Vista al mar, ...)
- **Instancias de spot (Spot instances)**: el hotel permite que la gente puje por las habitaciones vacías y el mejor postor se queda con ellas. Puede ser expulsado en cualquier momento
- **Hosts dedicados (Dedicated Hosts)**: Se reserva un edificio entero del complejo turístico
- **Reservas de capacidad (Capacity Reservations)**: reservas una habitación por un periodo con el precio completo aunque no te alojes en ella

Comparación de precios

Ejemplo - m4.large - us-east-1

Tipo de precio	Precio (por hora)
Precio bajo demanda (On-demand)	0.10\$
Instancias de spot (Spot instances)	0.038\$ - 0.039\$ (hasta 61% de descuento)
Instancia reservada (1 año) (Reserved)	0,062\$ (sin anticipo) - 0,058\$ (todo por adelantado)
Instancia reservada (3 años) (Reserved)	0,043\$ (sin anticipo) - 0,037\$ (todo por adelantado)
Plan de ahorro EC2 (1 año) (Saving plan)	0,062\$ (sin anticipo) - 0,058\$ (todo por adelantado)
Instancia reservada convertible (1 año)	0,071\$ (sin anticipo) - 0,066\$ (todo por adelantado)
Host dedicado (Dedicated host)	Precio bajo demanda (On-demand)
Reserva de host dedicado (Dedicated host reservation)	Hasta el 70% de descuento
Reservas de capacidad (Capacity reservation)	Precio bajo demanda (On-demand)

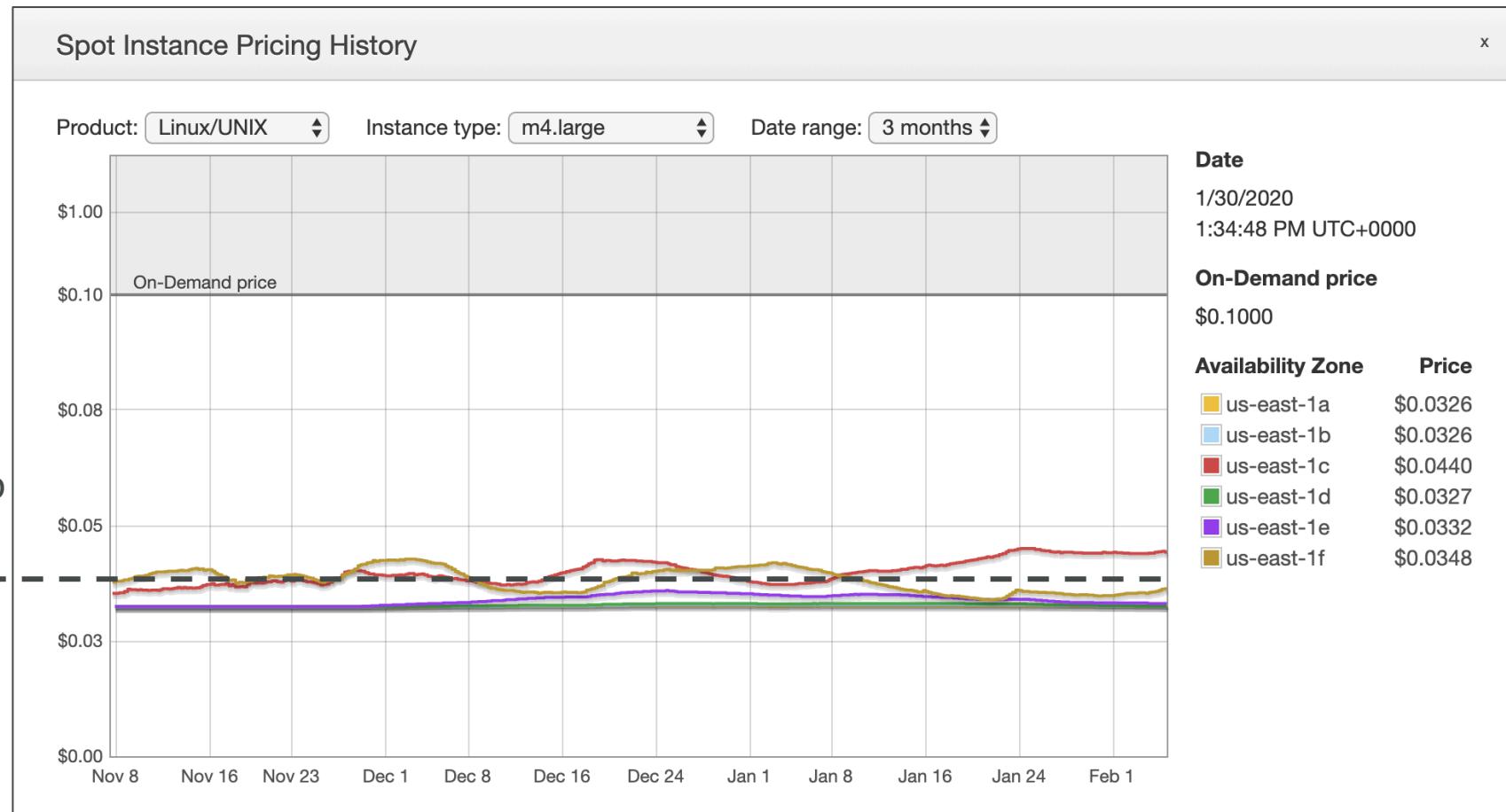


Peticiones de Instancias Spot de EC2

- Puedes obtener un descuento de hasta el 90% en comparación con la demanda
- Define el precio **spot máximo** y obtén la instancia mientras el precio **spot actual sea < máximo**
 - El precio spot por hora varía en función de la oferta y la capacidad
 - Si el precio spot actual > tu precio máximo, puedes elegir **parar** o **terminar** tu instancia con un periodo de gracia de 2 minutos.
- Otra estrategia: **Bloqueo de Spot**
 - "Bloquea" la instancia Spot durante un periodo de tiempo determinado (de 1 a 6 horas) sin interrupciones
 - En raras situaciones, la instancia puede ser reclamada
- **Se utiliza para trabajos por lotes, análisis de datos o cargas de trabajo resistentes a los fallos**
- **No es ideal para trabajos críticos o bases de datos**

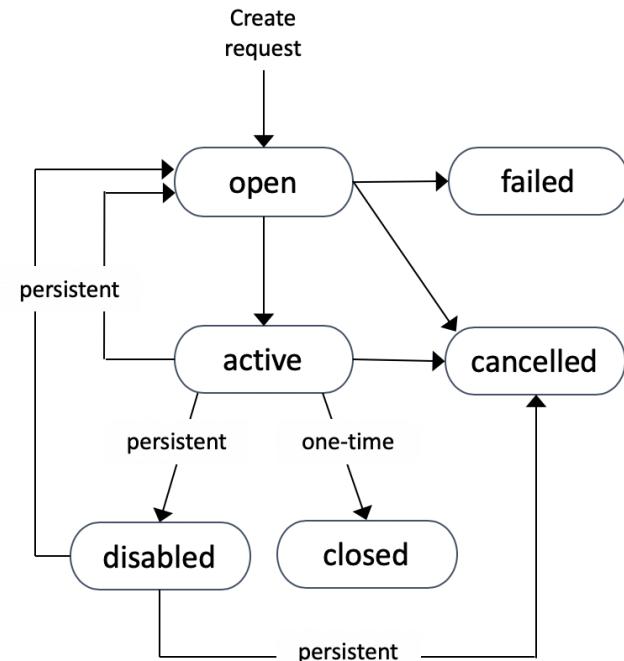
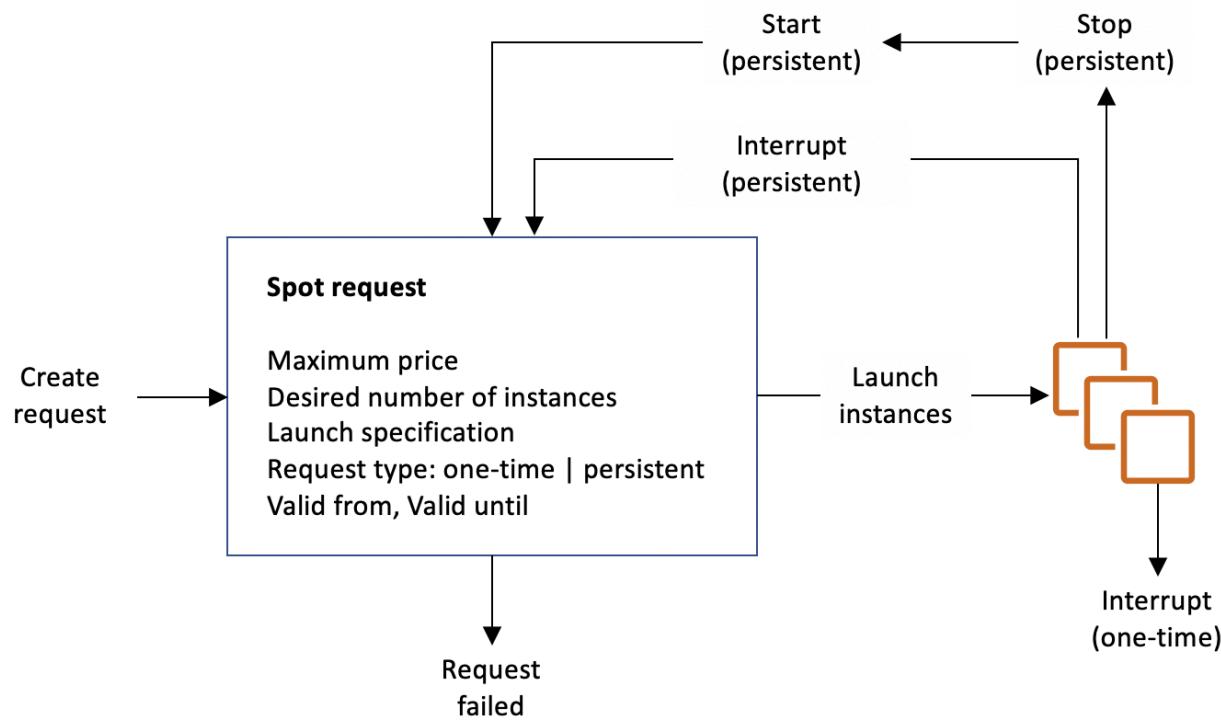
Precios de las Instancias Spot de EC2

Precio máximo definido por el usuario



<https://console.aws.amazon.com/ec2sp/v1/spot/home?region=us-east-1#>

¿Cómo terminar las Instancias Spot?



Sólo puedes cancelar las solicitudes de Instancias Spot que estén abiertas, activas o desactivadas.
La cancelación de una Solicitud Spot no termina las instancias
 Primero debes cancelar una Solicitud de Spot, y luego terminar las Instancias de Spot asociadas

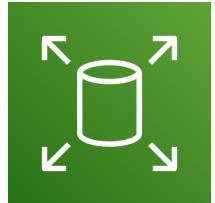
<https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/spot-requests.html>

Flotas Spot (Spot Fleets)

- Flotas Spot = conjunto de Instancias de Spot + (opcional) Instancias bajo demanda
- La Flota Spot tratará de alcanzar la capacidad objetivo con restricciones de precio
 - Define los posibles pools de lanzamiento: tipo de instancia (m5.large), SO, Zona de Disponibilidad
 - Puede tener varios pools de lanzamiento, para que la flota pueda elegir
 - La Flota Spot deja de lanzar instancias cuando alcanza la capacidad o el coste máximo
- Estrategias para asignar Instancias de Spot:
 - **bajo precio:** desde el pool con el precio más bajo (optimización de costes, carga de trabajo corta)
 - **diversificado:** distribución en todos los pools (gran disponibilidad, cargas de trabajo largas)
 - **capacidad optimizada:** pool con la capacidad óptima para el número de instancias
- Las flotas de Spot nos permiten solicitar automáticamente las Instancias de Spot con el precio más bajo

Almacenamiento de la instancia EC2

¿Qué es un volumen EBS?



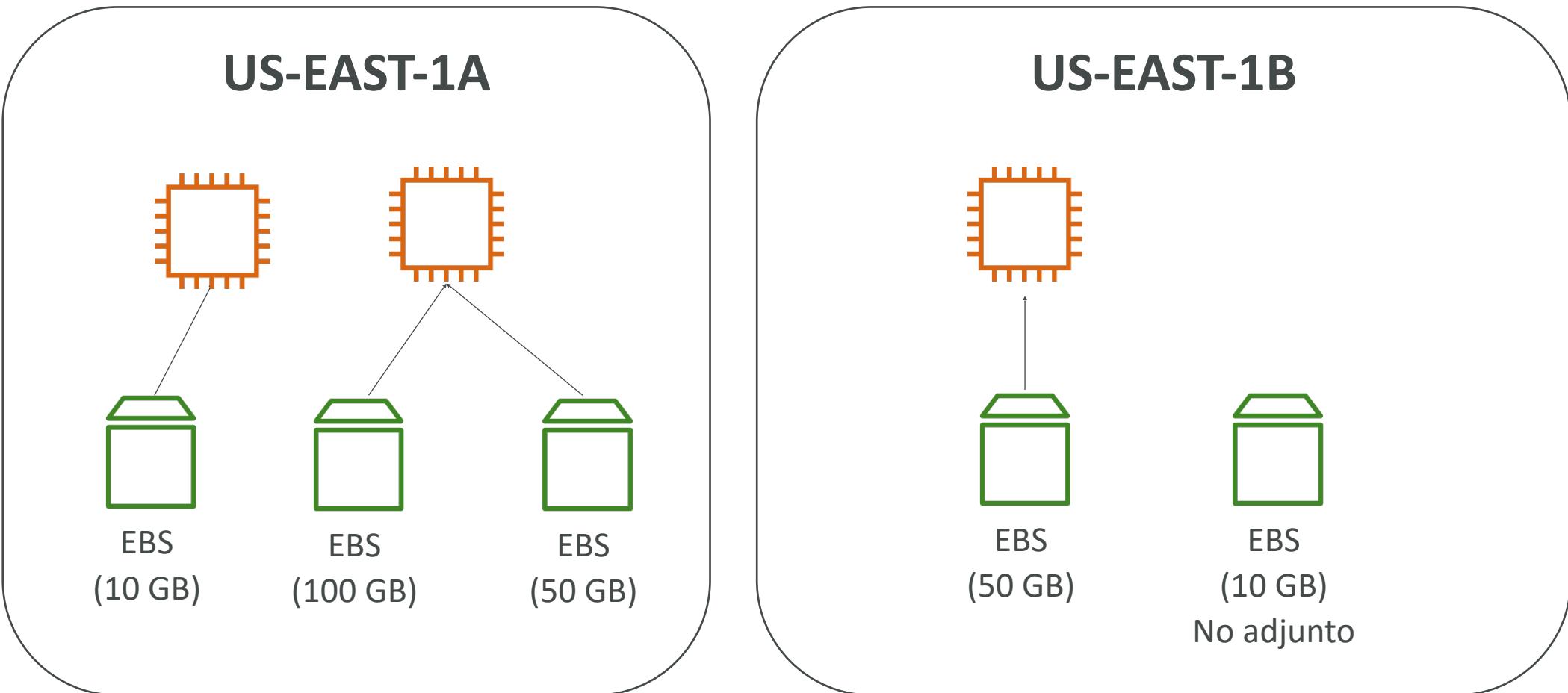
- Un **volumen EBS (Elastic Block Store)** es una **unidad de red** que puede adjuntar a las instancias mientras se ejecutan
- Permite que las instancias persistan los datos, incluso después de su finalización
- **Sólo pueden montarse en una instancia a la vez** (a nivel de CCP)
- Están vinculados **a una zona de disponibilidad específica**

- Analogía: Piensa en ellos como una "memoria USB de red"
- Nivel gratuito: 30 GB de almacenamiento EBS gratuito de tipo Propósito General (SSD) o Magnético al mes

Volumen EBS

- Es una unidad de red (es decir, no es una unidad física)
 - Utiliza la red para comunicar la instancia, lo que significa que puede haber un poco de latencia
 - Se puede separar de una instancia EC2 y conectarla a otra rápidamente
- Está bloqueado en una Zona de Disponibilidad (AZ)
 - Un volumen EBS en us-east-1a no puede adjuntarse a us-east-1b
 - Para trasladar un volumen, primero hay que hacer un snapshot del mismo
- Tener una capacidad provisionada (tamaño en GBs, e IOPS)
 - Se facturará toda la capacidad aprovisionada
 - Puede aumentar la capacidad de la unidad con el tiempo

Volumen EBS - Ejemplo



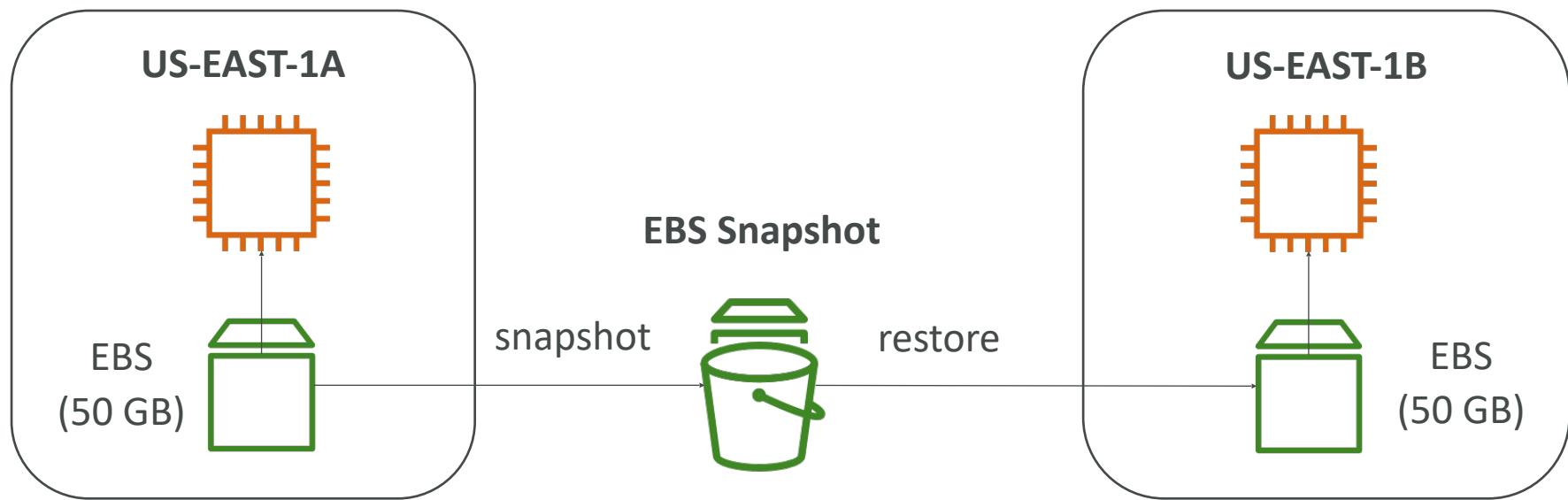
EBS - Atributo "Borrar al terminar"

Volume Type	Device	Snapshot	Size (GiB)	Volume Type	IOPS	Throughput (MB/s)	Delete on Termination	Encryption
Root	/dev/xvda	snap-09f18f682fd23a1b1	8	General Purpose SSD (gp2)	100 / 3000	N/A	<input checked="" type="checkbox"/>	Not Encrypted
EBS	/dev/sdb	Search (case-insensit)	8	General Purpose SSD (gp2)	100 / 3000	N/A	<input type="checkbox"/>	Not Encrypted
Add New Volume								

- Controla el comportamiento de EBS cuando una instancia EC2 termina
 - Por defecto, se elimina el volumen EBS root / raíz (atributo habilitado)
 - Por defecto, cualquier otro volumen EBS adjunto no se elimina (atributo deshabilitado)
- Esto puede ser controlado por la consola de AWS / AWS CLI
- **Caso de uso: preservar el volumen root / raíz cuando se termina la instancia**

Snapshot / Instantáneas de EBS

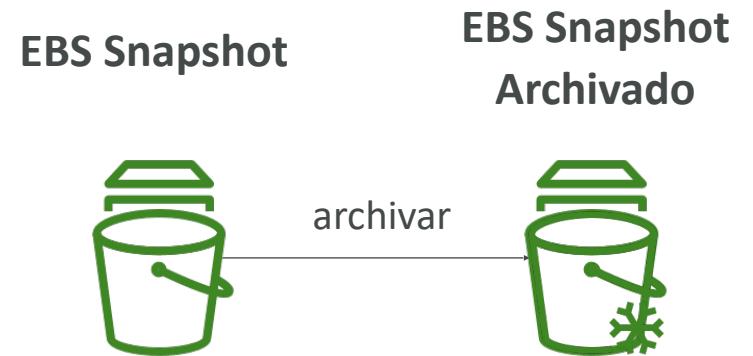
Haz una copia de seguridad (snapshot) de tu volumen EBS en un momento dado
No es necesario separar el volumen para hacer la instantánea, pero se recomienda
Puedes copiar las instantáneas a través de AZ o Región



Características de los Snapshots de EBS

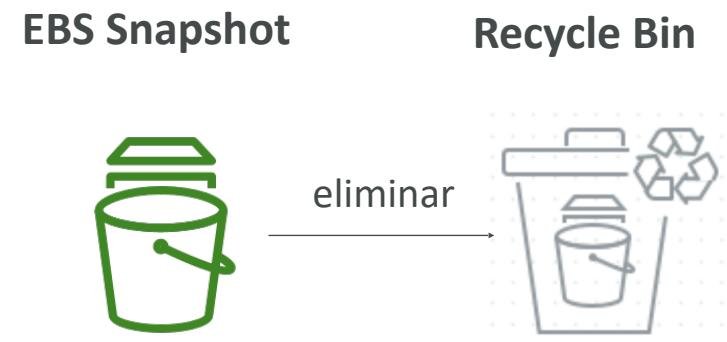
• Archivo de Snapshots de EBS

- Mover un snapshot a un "nivel de archivo" que es un 75% más barato
- La restauración del archivo tarda entre 24 y 72 horas



• Papelera de reciclaje para Snapshots EBS

- Configura reglas para retener los snapshots eliminados para poder recuperarlos después de un borrado accidental
- Especifica la retención (de 1 día a 1 año)



Visión general de AMI



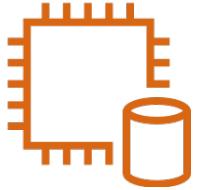
- AMI = Amazon Machine Image
- Las AMI son una **personalización** de una instancia EC2
 - Añades tu propio software, configuración, sistema operativo, monitorización...
 - Tiempo de arranque/configuración más rápido porque todo el software está preempaquetado
- Las AMI se construyen para una **región específica** (y pueden copiarse entre regiones)
- Puedes lanzar instancias EC2 desde:
 - **Una AMI pública:** proporcionada por AWS
 - **Tu propia AMI:** la creas y la mantienes tú mismo
 - **Una AMI de AWS Marketplace:** una AMI hecha por otra persona (y potencialmente vendida)

Proceso AMI (desde una instancia EC2)

- Iniciar una instancia EC2 y personalizarla
- Detener la instancia (para la integridad de los datos)
- Construir una AMI - esto también creará instantáneas de EBS
- Lanzar instancias desde otras AMIs



Almacén de instancias EC2



- Los volúmenes EBS son **unidades de red** con un rendimiento bueno pero “limitado”
- **Si necesitas un disco de hardware de alto rendimiento, utilizas EC2 Instance Store**
- Mejor rendimiento de E/S
- Los almacenes de instancias EC2 pierden su almacenamiento si se detienen (son efímeros)
- Bueno para el buffer / cache / datos de memoria virtual / contenido temporal
- Riesgo de pérdida de datos si el hardware falla
- Las copias de seguridad y la replicación son responsabilidad tuya

Almacén local de instancias EC2

Instance Size	100% Random Read IOPS	Write IOPS
i3.large *	100,125	35,000
i3.xlarge *	206,250	70,000
i3.2xlarge	412,500	180,000
i3.4xlarge	825,000	360,000
i3.8xlarge	1.65 million	720,000
i3.16xlarge	3.3 million	1.4 million
i3.metal	3.3 million	1.4 million
i3en.large *	42,500	32,500
i3en.xlarge *	85,000	65,000
i3en.2xlarge *	170,000	130,000
i3en.3xlarge	250,000	200,000
i3en.6xlarge	500,000	400,000
i3en.12xlarge	1 million	800,000
i3en.24xlarge	2 million	1.6 million
i3en.metal	2 million	1.6 million

IOPS muy altas

Tipos de volúmenes EBS

- Los volúmenes EBS vienen en 6 tipos
 - **gp2 / gp3 (SSD)**: Volumen SSD de uso general que equilibra el precio y el rendimiento para una amplia variedad de cargas de trabajo
 - **io1 / io2 (SSD)**: El volumen SSD de mayor rendimiento para cargas de trabajo de misión crítica de baja latencia o alto rendimiento
 - **st1 (HDD)**: Volumen de disco duro de bajo coste diseñado para cargas de trabajo de acceso frecuente y alto rendimiento
 - **sc1 (HDD)**: El volumen de disco duro más barato, diseñado para cargas de trabajo de acceso menos frecuente
- Los volúmenes EBS se caracterizan en Tamaño | Rendimiento | IOPS (I/O Ops Per Sec)
- En caso de duda, consulta siempre la documentación de AWS: ¡es buena!
- **Sólo se pueden utilizar gp2/gp3 y io1/io2 como volúmenes de arranque**

Tipos de volúmenes EBS - Casos de uso

SSD de uso general

- Almacenamiento rentable, baja latencia
- Volúmenes de arranque del sistema, escritorios virtuales, entornos de desarrollo y prueba
- 1 GiB - 16 TiB
- gp3:
 - Línea de base de 3.000 IOPS y rendimiento de 125 MiB/s
 - Puede aumentar las IOPS hasta 16.000 y el rendimiento hasta 1000 MiB/s de forma independiente
- gp2:
 - Los volúmenes gp2 pequeños pueden reventar las IOPS hasta 3.000
 - El tamaño del volumen y las IOPS están vinculados, las IOPS máximas son 16.000
 - 3 IOPS por GB, lo que significa que con 5.334 GB estamos en el máximo de IOPS

Tipos de volúmenes EBS - Casos de uso IOPS provisionadas (PIOPS) SSD

- Aplicaciones empresariales críticas con un rendimiento sostenido de IOPS
- O aplicaciones que necesitan más de 16.000 IOPS
- Excelente para las **cargas de trabajo de las bases de datos** (sensibles al rendimiento y la consistencia del almacenamiento)
- io1/io2 (4 GiB - 16 TiB):
 - PIOPS máximos: 64.000 para instancias Nitro EC2 y 32.000 para otras
 - Puede aumentar los PIOPS independientemente del tamaño del almacenamiento
 - io2 tiene más durabilidad y más IOPS por GiB (al mismo precio que io1)
- io2 Block Express (4 GiB - 64 TiB):
 - Latencia de menos de un milisegundo
 - PIOPS máximas: 256.000 con una relación IOPS:GiB de 1.000:1
- Soporta EBS Multi-attach

Tipos de volúmenes EBS - Casos de uso

Unidades de disco duro (HDD)

- No puede ser un volumen de arranque
- De 125 GiB a 16 TiB
- Disco duro de rendimiento optimizado (stl)
 - Big Data, almacenes de datos, procesamiento de logs
 - **Rendimiento máximo** de 500 MiB/s - IOPS máximo de 500
- Disco duro frío (sc1):
 - Para datos a los que se accede con poca frecuencia
 - Escenarios en los que el menor coste es importante
 - **Rendimiento máximo** de 250 MiB/s - IOPS máximas de 250

EBS - Resumen de los tipos de volúmenes

	General Purpose SSD	
Tipo de volumen	gp3	gp2
Durabilidad	99,8 % - 99,9 % de durabilidad (0,1 % - 0,2 % tasa anual de errores)	
Casos de uso	<ul style="list-style-type: none">• Aplicaciones interactivas de baja latencia• Entornos de desarrollo y pruebas• Escritorios virtuales• Bases de datos de tamaño mediano y una sola instancia• Volúmenes de arranque	
Tamaño del volumen	1 GiB - 16 TiB	
Máximo de IOPS por volumen (E/S de 16 KiB)**	16,000	
Rendimiento máximo por volumen**	1000 MiB/s	250 MiB/s *
Amazon EBS Multi-attach	No admitido	
Volumen de arranque	Soportado	

<https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ebs-volume-types.html#solid-state-drives>

EBS - Resumen de los tipos de volúmenes

	Provisioned IOPS SSD		
Tipo de volumen	io2 Block Express*	io2	io1
Durabilidad	99,999 % de durabilidad (0,001 % tasa anual de errores)	99,999 % de durabilidad (0,001 % tasa anual de errores)	99,8 % - 99,9 % de durabilidad (0,1 % - 0,2 % tasa anual de errores)
Casos de uso	<p>Cargas de trabajo que requieren lo siguiente:</p> <ul style="list-style-type: none"> • Latencia inferior a milisegundos • Rendimiento de IOPS sostenido • Más de 64 000 IOPS o 1000 MiB/s de rendimiento 	<ul style="list-style-type: none"> • Cargas de trabajo que requieren un rendimiento sostenido de IOPS o más de 16,000 IOPS • Cargas de trabajo de bases de datos con uso intensivo de operaciones de E/S 	
Tamaño del volumen	4 GiB - 64 TiB	4 GiB - 16 TiB	
IOPS máximo por volumen (E/S de 16 KiB)	256 000	64 000†	
Rendimiento máximo por volumen	4000 MiB/s	1000 MiB/s†	
Amazon EBS Multi-attach	Compatible		
Volumen de arranque	Compatible		

<https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ebs-volume-types.html#solid-state-drives>

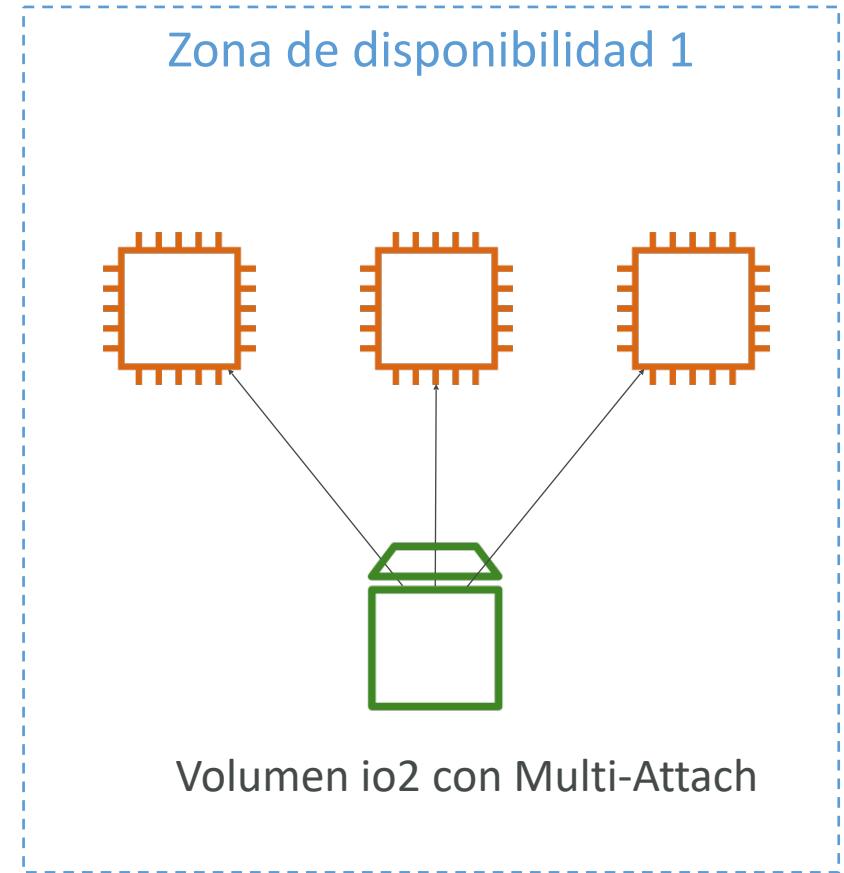
EBS - Resumen de los tipos de volúmenes

	HDD con rendimiento optimizado	HDD en frío
Tipo de volumen	st1	sc1
Durabilidad	99,8 % - 99,9 % de durabilidad (0,1 % - 0,2 % tasa anual de errores)	99,8 % - 99,9 % de durabilidad (0,1 % - 0,2 % tasa anual de errores)
Casos de uso	<ul style="list-style-type: none">• Big data• Data warehouses• Procesamiento de registros	<ul style="list-style-type: none">• Almacenamiento orientado al rendimiento para datos a los que se accede con poca frecuencia• Escenarios en los que es importante el costo de almacenamiento más bajo
Tamaño del volumen	125 GiB - 16 TiB	125 GiB - 16 TiB
IOPS máximo por volumen (E/S de 1 MiB)	500	250
Rendimiento máximo por volumen	500 MiB/s	250 MiB/s
Amazon EBS Multi-attach	No admitido	No admitido
Volumen de arranque	No admitido	No admitido

<https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ebs-volume-types.html#solid-state-drives>

EBS Multi-Attach - familia io1/io2

- Adjunta el mismo volumen EBS a varias instancias EC2 en la misma AZ
- Cada instancia tiene permisos completos de lectura y escritura en el volumen de alto rendimiento
- Caso de uso:
 - Conseguir una mayor disponibilidad de las aplicaciones en clusters de Linux (por ejemplo, Teradata)
 - Las aplicaciones deben gestionar operaciones de escritura concurrentes
- Hasta 16 instancias EC2 a la vez
- Debe utilizar un sistema de archivos que sea compatible con el clúster (no XFS, EX4, etc...)



Encriptación de EBS

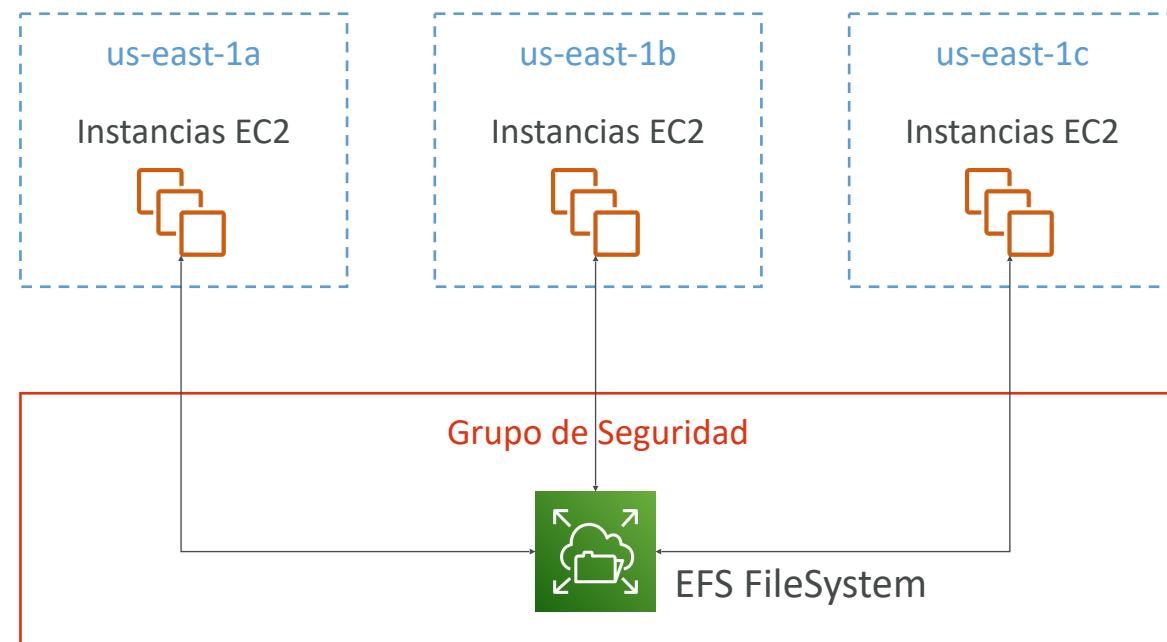
- Cuando creas un volumen EBS encriptado, obtienes lo siguiente:
 - Los datos en reposo están encriptados dentro del volumen
 - Todos los datos en movimiento entre la instancia y el volumen están encriptados
 - Todas las instantáneas están encriptadas
 - Todos los volúmenes creados a partir de la instantánea
- El cifrado y el descifrado se gestionan de forma transparente (no tienes que hacer nada)
- El cifrado tiene un impacto mínimo en la latencia
- El cifrado de EBS aprovecha las claves de KMS (AES-256)
- La copia de una Snapshot no cifrada permite el cifrado
- Las instantáneas de los volúmenes encriptados están encriptadas

Cifrado: cifrar un volumen EBS

- Crea una Snapshot de EBS del volumen
- Encripta la instantánea EBS (utilizando la copia)
- Crea un nuevo volumen EBS a partir de la instantánea (el volumen también estará encriptado)
- Ahora puedes adjuntar el volumen encriptado a la instancia original

Amazon EFS – Elastic File System

- NFS gestionado (sistema de archivos de red) que puede montarse en muchas EC2
- EFS funciona con instancias EC2 en multi-AZ
- Alta disponibilidad, escalable, caro (3x gp2), pago por uso



Amazon EFS – Elastic File System

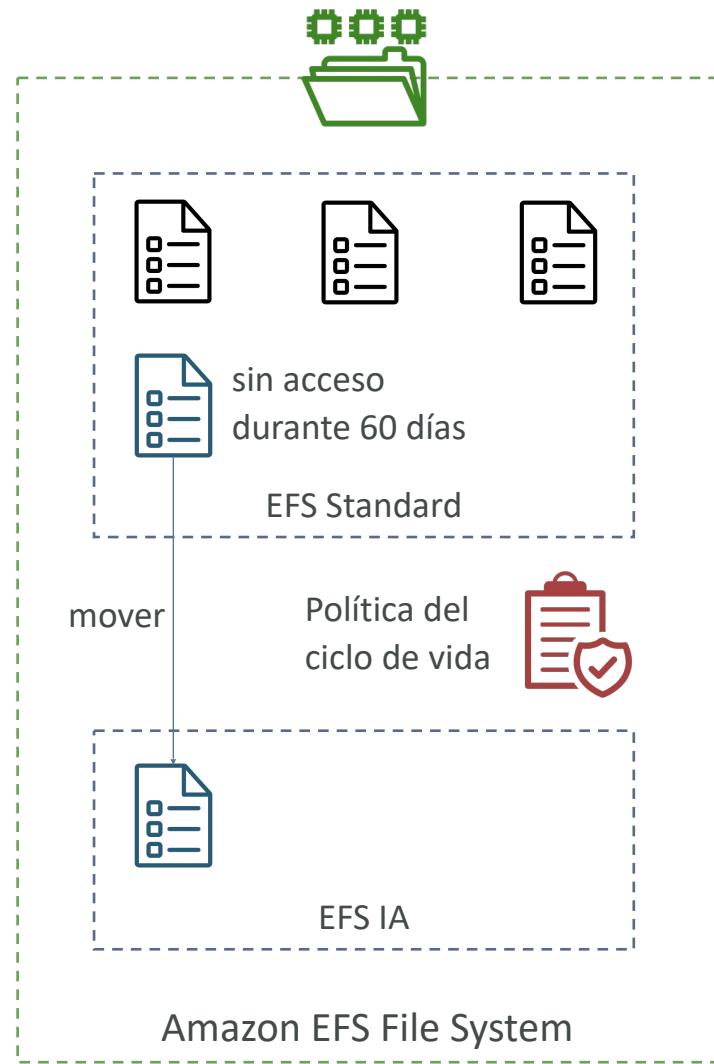
- Casos de uso: gestión de contenidos, servicio web, intercambio de datos, Wordpress
- Utiliza el protocolo NFSv4.1
- Utiliza el grupo de seguridad para controlar el acceso al EFS
- **Compatible con AMI basadas en Linux (no en Windows)**
- Cifrado en reposo mediante KMS
- Sistema de archivos POSIX (~Linux) que tiene una API de archivos estándar
- El sistema de archivos se escala automáticamente, paga por uso, ¡no hay que planificar la capacidad!

EFS - Clases de rendimiento y almacenamiento

- **Escala EFS**
 - 1000s de clientes NFS concurrentes, 10 GB+ /s de rendimiento
 - Crece hasta convertirse en un sistema de archivos en red a escala de petabytes, de forma automática
- **Modo de rendimiento (establecido en el momento de la creación del EFS)**
 - Propósito general (por defecto): casos de uso sensibles a la latencia (servidor web, CMS, etc.)
 - E/S máxima: mayor latencia, rendimiento, altamente paralelo (big data, procesamiento de medios)
- **Modo de rendimiento (Throughput)**
 - Ráfaga ($1\text{ TB} = 50\text{MiB/s} + \text{ráfaga de hasta } 100\text{MiB/s}$)
 - Aprovisionado: fija tu rendimiento independientemente del tamaño del almacenamiento, por ejemplo: 1 GiB/s para un almacenamiento de 1 TB

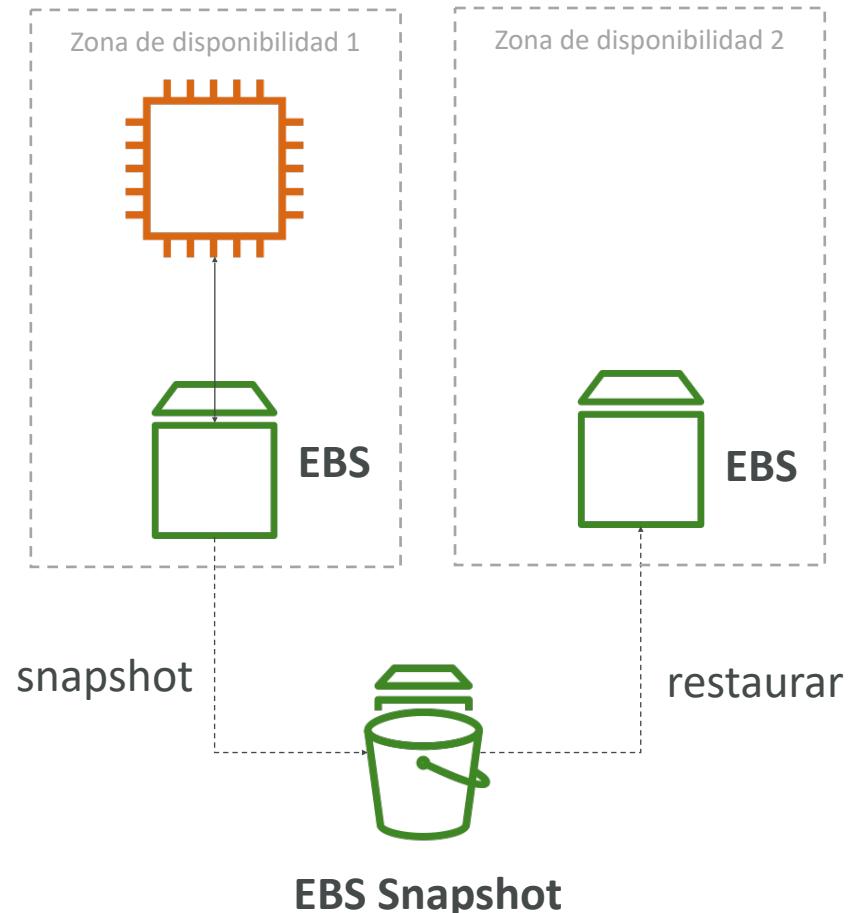
EFS - Clases de almacenamiento

- **Niveles de almacenamiento (función de gestión del ciclo de vida: mover el archivo después de N días)**
 - Estándar: para archivos de acceso frecuente
 - Acceso infrecuente (EFS-IA): coste de recuperación de los archivos, menor precio de almacenamiento. Habilita EFS-IA con una política de ciclo de vida
- **Disponibilidad y durabilidad**
 - Estándar: Multi-AZ, ideal para prod
 - Una zona: Una AZ, genial para dev, copia de seguridad activada por defecto, compatible con IA (EFS One Zone-IA)
- Más del 90% de ahorro de costes



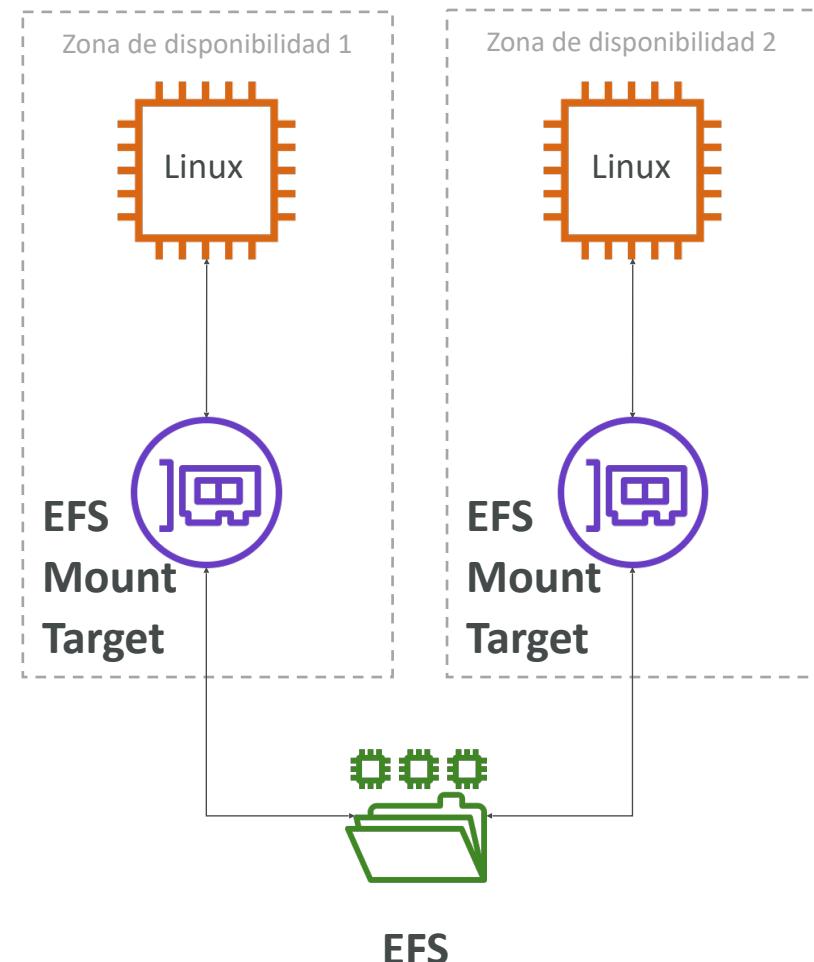
EBS vs EFS – Elastic Block Storage

- Los volúmenes EBS...
 - sólo pueden adjuntarse a una instancia a la vez
 - están bloqueados a nivel de Zona de Disponibilidad (AZ)
 - gp2: la IO aumenta si aumenta el tamaño del disco
 - io1: puede aumentar la IO de forma independiente
- Para migrar un volumen EBS a través de la AZ
 - Haz una Snapshot
 - Restaura la instantánea en otra AZ
 - Las copias de seguridad de EBS utilizan IO y no deberías ejecutarlas mientras tu aplicación esté manejando mucho tráfico
- Los volúmenes EBS root de las instancias se terminan por defecto si la instancia EC2 se termina. (puedes desactivarlo)



EBS vs EFS – Elastic File System

- Montar 100s de instancias a través de AZ
 - Compartir archivos del sitio web EFS (WordPress)
 - Sólo para instancias Linux (POSIX)
-
- EFS tiene un precio más elevado que EBS
 - Puede aprovechar EFS-IA para ahorrar costes
-
- Recuerda: EFS vs EBS vs Instance Store



Fundamentos de AWS – Parte II

Escalabilidad y alta disponibilidad

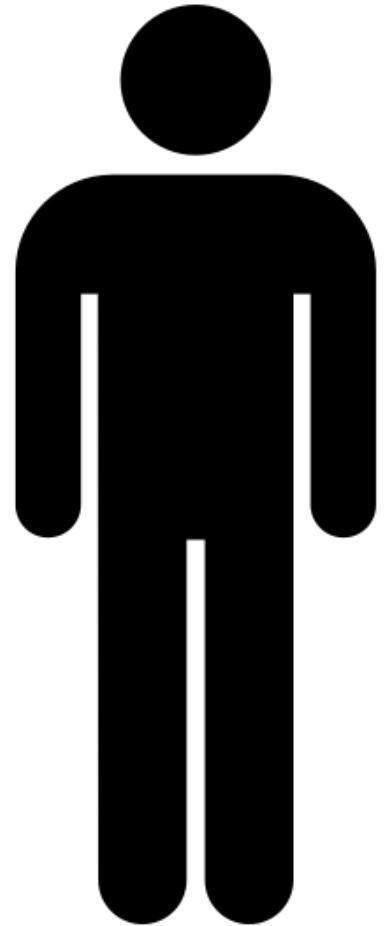
- La escalabilidad significa que una aplicación/sistema puede manejar mayores cargas adaptándose.
- Hay dos tipos de escalabilidad:
 - Escalabilidad vertical
 - Escalabilidad horizontal (= elasticidad)
- **La escalabilidad está vinculada pero es diferente a la Alta Disponibilidad**
- Vamos a profundizar en la distinción, utilizando un centro de llamadas como ejemplo

Escalabilidad vertical

- La escalabilidad vertical significa aumentar el tamaño de la instancia
- Por ejemplo, tu aplicación se ejecuta en un t2.micro
- Escalar esa aplicación verticalmente significa ejecutarla en un t2.large
- La escalabilidad vertical es muy común para los sistemas no distribuidos, como una base de datos.
- RDS, ElastiCache son servicios que pueden escalar verticalmente.
- Suele haber un límite en cuanto a la escalabilidad vertical (límite de hardware)



operador junior

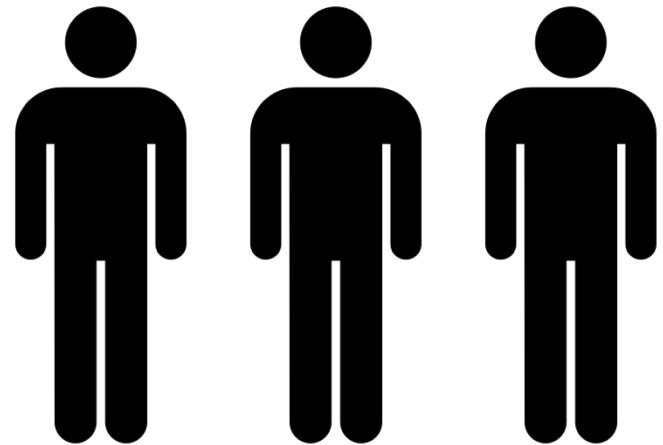
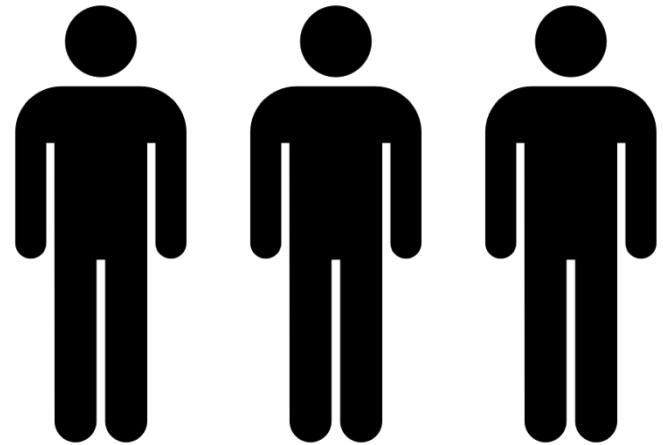


operador senior

Escalabilidad horizontal

- La escalabilidad horizontal significa aumentar el número de instancias / sistemas de tu aplicación
- El escalado horizontal implica sistemas distribuidos.
- Esto es muy común para las aplicaciones web / aplicaciones modernas
- Es fácil escalar horizontalmente gracias a las ofertas en el Cloud, como Amazon EC2

operador operador operador

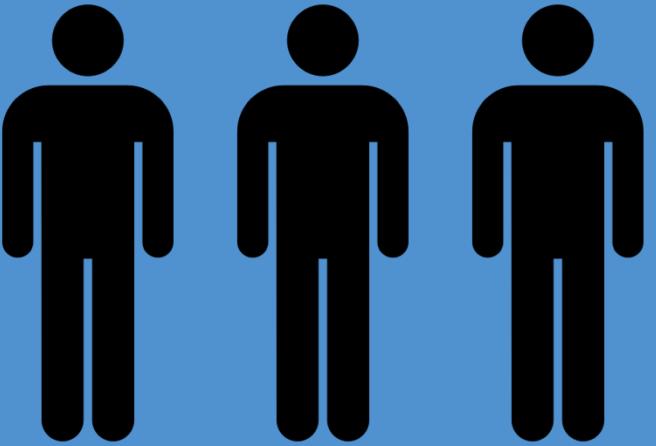


operador operador operador

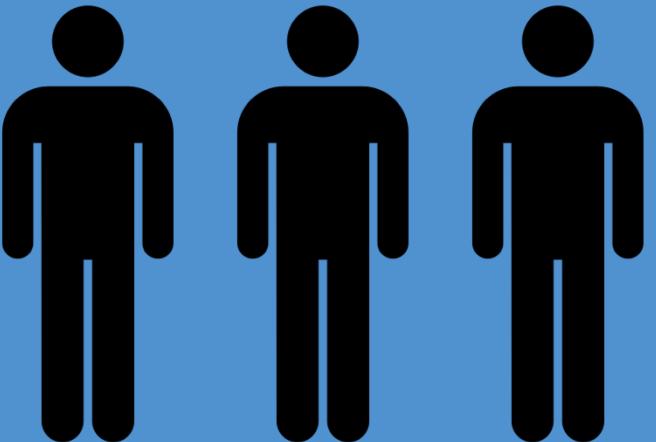
Alta disponibilidad

- La alta disponibilidad suele ir de la mano del escalado horizontal
- La alta disponibilidad significa ejecutar tu aplicación/sistema en al menos 2 centros de datos (== Zonas de Disponibilidad)
- El objetivo de la alta disponibilidad es sobrevivir a la pérdida de un centro de datos
- La alta disponibilidad puede ser pasiva (para RDS Multi AZ, por ejemplo)
- La alta disponibilidad puede ser activa (para el escalado horizontal)

primer edificio en Nueva York



segundo edificio en San Francisco

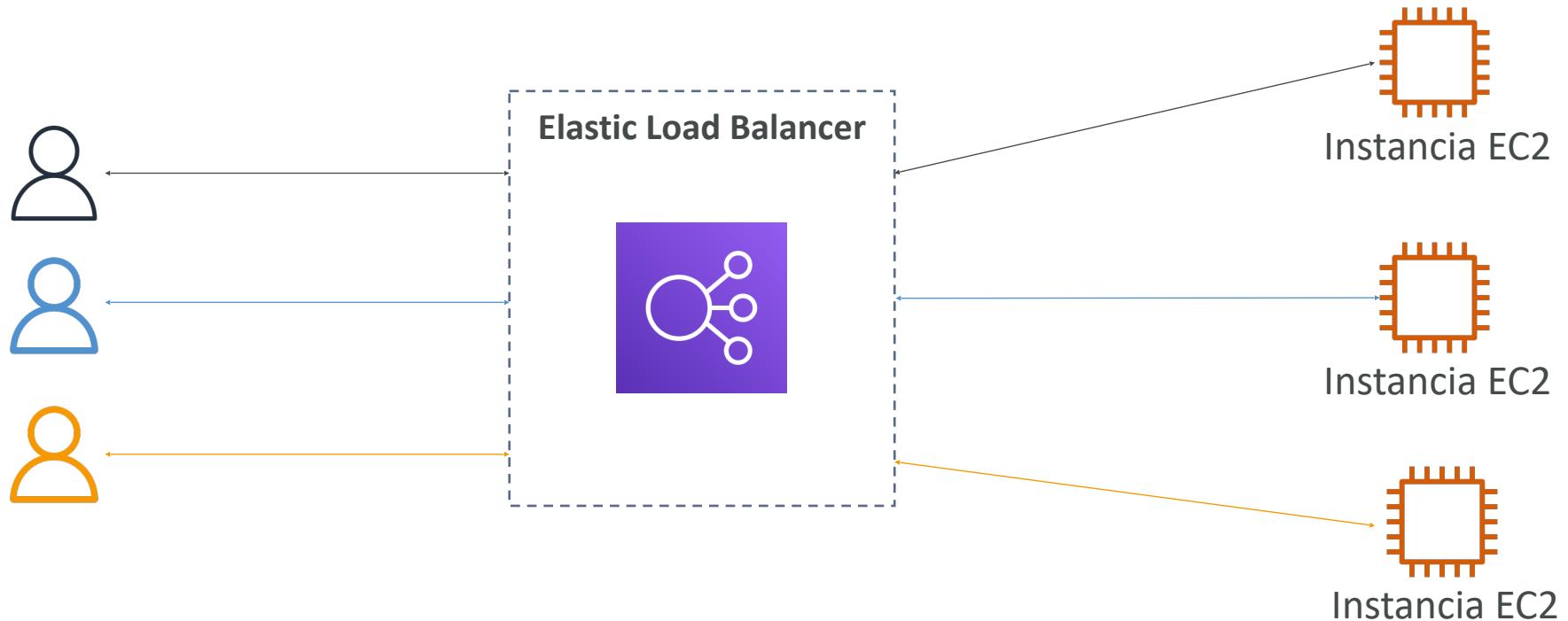


Alta disponibilidad y escalabilidad para EC2

- Escalado vertical: Aumentar el tamaño de la instancia (= escalar hacia arriba/hacia abajo)
 - De: t2.nano - 0,5G de RAM, 1 vCPU
 - A: u-12tb1.metal - 12,3 TB de RAM, 448 vCPUs
- Escalado horizontal: Aumenta el número de instancias (= escalar hacia fuera / hacia dentro)
 - Grupo de Auto Scaling
 - Load Balancer
- Alta disponibilidad: Ejecuta instancias para la misma aplicación a través de múltiples AZ
 - Auto Scaling Groups multi AZ
 - Load Balancer multi AZ

¿Qué es el balanceo de carga (load balancing)?

- Los Load Balancers son servidores que reenvían el tráfico a varios servidores (por ejemplo, instancias EC2) en sentido descendente



¿Por qué utilizar un Load Balancer?

- Repartir la carga entre varias instancias descendentes
- Exponer un único punto de acceso (DNS) a tu aplicación
- Manejar sin problemas los fallos de las instancias descendentes
- Realiza comprobaciones periódicas de la salud de tus instancias
- Proporcionar terminación SSL (HTTPS) para tus sitios web
- Imponer la adherencia con las cookies
- Alta disponibilidad entre zonas
- Separar el tráfico público del privado

¿Por qué utilizar un Elastic Load Balancer?

- Un Elastic Load Balancer es un **equilibrador de carga gestionado**
 - AWS garantiza su funcionamiento
 - AWS se encarga de las actualizaciones, el mantenimiento y la alta disponibilidad
 - AWS sólo proporciona unos pocos mandos de configuración
- Cuesta poco configurar tu propio balanceador de carga, y te supondrá una mejora en la disponibilidad y escalabilidad
- Está integrado con muchas ofertas/servicios de AWS
 - EC2, EC2 Auto Scaling Groups, Amazon ECS
 - AWS Certificate Manager (ACM), CloudWatch
 - Route 53, AWS WAF, AWS Global Accelerator

Controles de salud

- Las comprobaciones de salud son cruciales para los Load Balancer
- Permiten al Load Balancer saber si las instancias a las que reenvía el tráfico están disponibles para responder a las peticiones
- La comprobación de salud se realiza en un puerto y una ruta (/health es común)
- Si la respuesta no es 200 (OK), la instancia no está sana

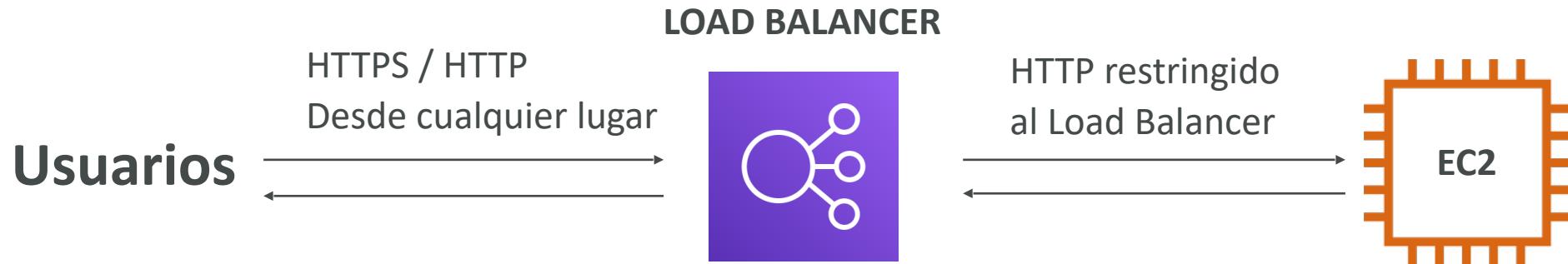




Tipos de Load Balancer en AWS

- AWS tiene **4 tipos de Load Balancer gestionados**
- **Classic Load Balancer** (v1 - antigua generación) - 2009 - CLB
 - HTTP, HTTPS, TCP, SSL (TCP seguro)
- **Application Load Balancer** (v2 - nueva generación) - 2016 - ALB
 - HTTP, HTTPS, WebSocket
- **Network Load Balancer** (v2 - nueva generación) - 2017 - NLB
 - TCP,TLS (TCP seguro), UDP
- **Gateway Load Balancer** - 2020 - GWLB
 - Funciona en la capa 3 (capa de red) - Protocolo IP
- En general, se recomienda utilizar los load balancer de nueva generación, ya que ofrecen más funciones
- Algunos Load Balancer pueden configurarse como ELB internos (privados) o externos (públicos)

Grupos de seguridad del Load Balancer



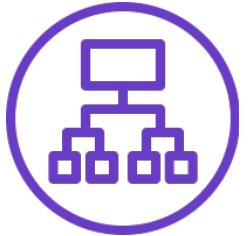
Grupo de seguridad del Load Balancer:

Type (i)	Protocol (i)	Port Range (i)	Source (i)	Description (i)
HTTP	TCP	80	0.0.0.0/0	Allow HTTP from an...
HTTPS	TCP	443	0.0.0.0/0	Allow HTTPS from a...

Grupo de seguridad de aplicaciones: Permitir el tráfico sólo desde el Load Balancer

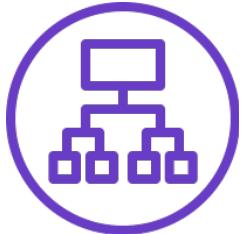
Type (i)	Protocol (i)	Port Range (i)	Source (i)	Description (i)
HTTP	TCP	80	sg-054b5ff5ea02f2b6e (load-b	Allow Traffic only...

Application Load Balancer (v2)



- El Application Load Balancer es de capa 7 (HTTP)
- Equilibrio de carga para múltiples aplicaciones HTTP en distintas máquinas (grupos de destino)
- Equilibrio de carga para múltiples aplicaciones en la misma máquina (por ejemplo, contenedores)
- Soporte para HTTP/2 y WebSocket
- Soporta redireccionamientos (de HTTP a HTTPS, por ejemplo)

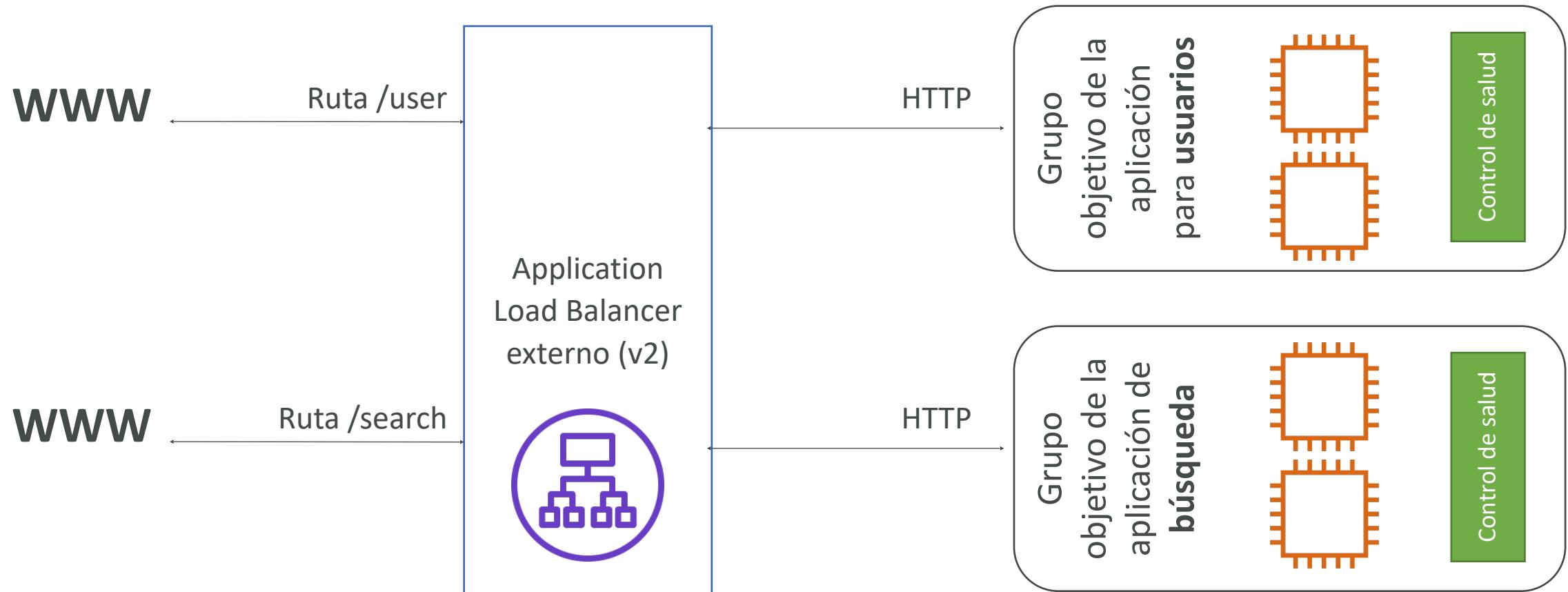
Application Load Balancer (v2)



- Tablas de enrutamiento a diferentes grupos de destino:
 - Enrutamiento basado en la ruta en la URL (example.com/users & example.com/posts)
 - Enrutamiento basado en el nombre de host en la URL (one.example.com & other.example.com)
 - Enrutamiento basado en la cadena de consulta, las cabeceras (example.com/users?id=123&order=false)
- Los ALB son muy adecuados para los microservicios y las aplicaciones basadas en contenedores (ejemplo: Docker y Amazon ECS)
- Tiene una función de mapeo de puertos para redirigir a un puerto dinámico en ECS
- En comparación, necesitaríamos varios Classic Load Balancer por aplicación

Application Load Balancer (v2)

Tráfico basado en HTTP



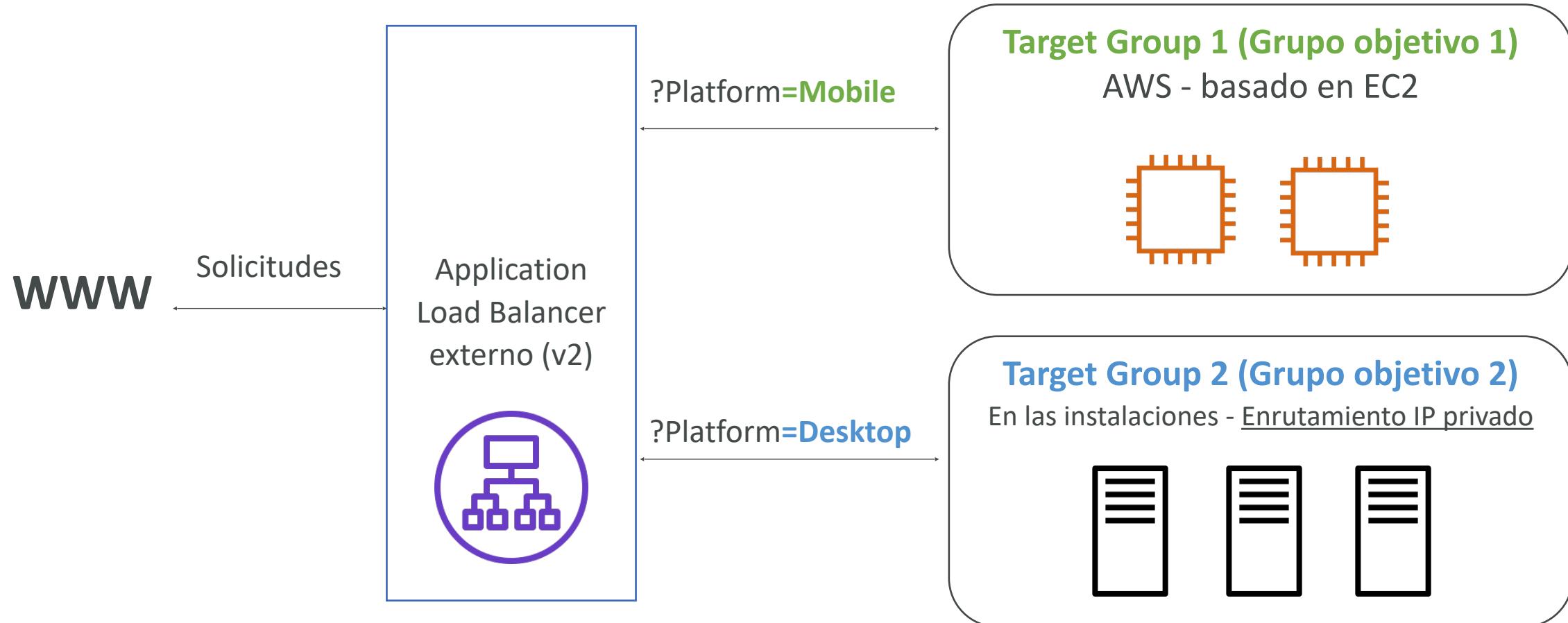
Application Load Balancer (v2)

Target Groups (Grupos objetivo)

- Instancias EC2 (pueden ser gestionadas por un Auto Scaling Groups) - HTTP
 - Tareas de ECS (gestionadas por el propio ECS) - HTTP
 - Funciones Lambda - La petición HTTP se traduce en un evento JSON
 - Direcciones IP - deben ser IPs privadas
-
- El ALB puede enrutar a múltiples grupos de destino
 - Las comprobaciones de salud son a nivel de grupo de destino

Application Load Balancer (v2)

Strings de consulta/enrutamiento de parámetros

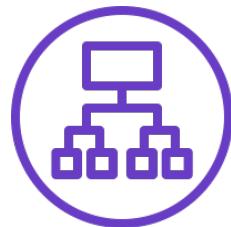


Application Load Balancer (v2)

Es bueno saber que

- Nombre de host fijo (XXX.region.elb.amazonaws.com)
- Los servidores de aplicaciones no ven directamente la IP del cliente
 - La verdadera IP del cliente se inserta en la cabecera X-Forwarded-For
 - También podemos obtener el puerto (X-Forwarded-Port) y el protocolo (X-Forwarded-Proto)

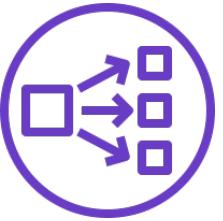
IP del cliente
12.34.56.78



IP del Load Balancer
(IP privada)



Terminación de la conexión

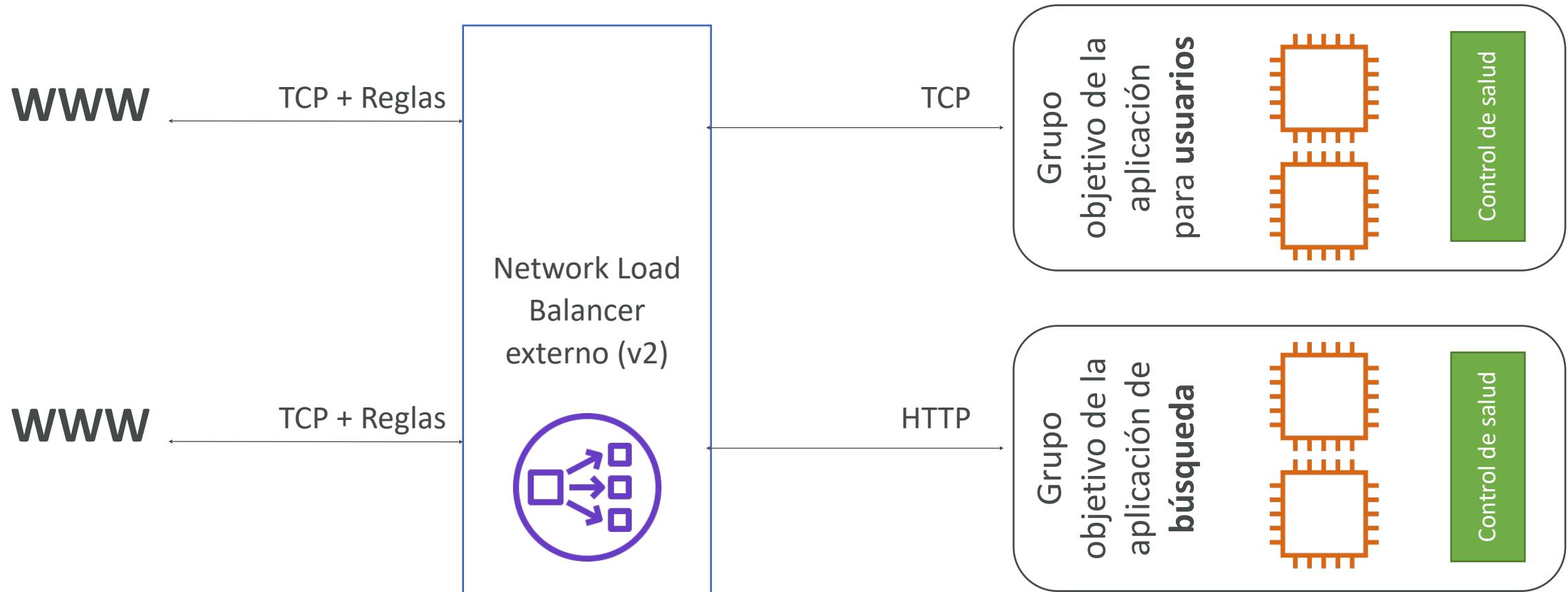


Network Load Balancer (v2)

- Los Network Load Balancer (Capa 4) permiten:
 - **Reenviar el tráfico TCP y UDP a tus instancias**
 - Manejar millones de peticiones por segundo
 - Menor latencia ~100 ms (frente a los 400 ms del ALB)
- **El NLB tiene una IP estática por AZ, y soporta la asignación de IP elástica**
(útil para poner en lista blanca una IP específica)
- Los NLB se utilizan para un rendimiento extremo, tráfico TCP o UDP
- **No está incluido en la capa gratuita de AWS**

Network Load Balancer (v2)

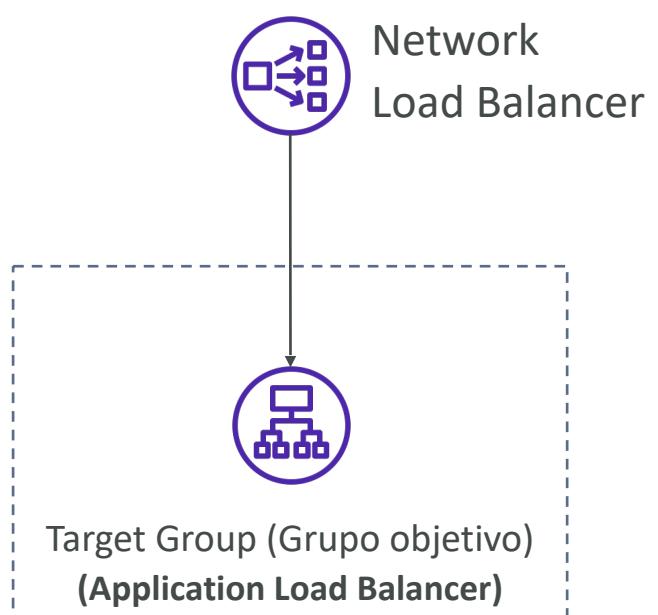
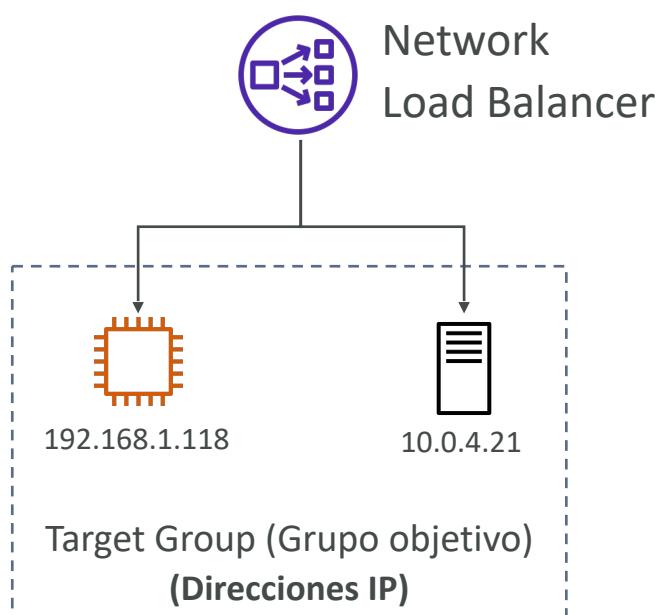
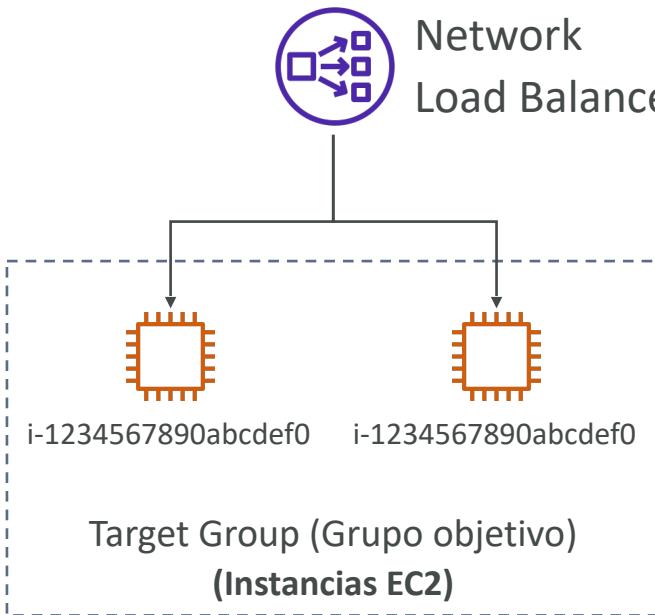
Tráfico basado en TCP (capa 4)



Network Load Balancer

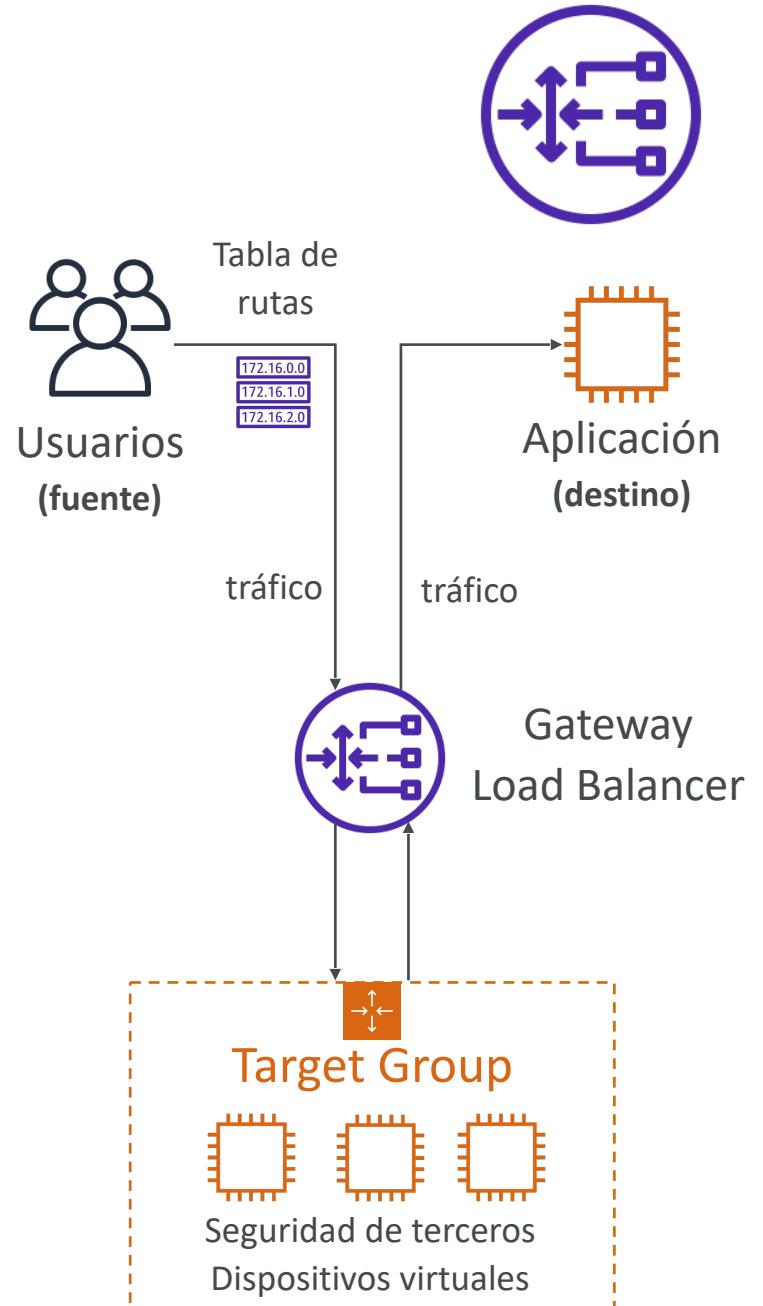
Target Groups (Grupos objetivo)

- **Instancias EC2**
- **Direcciones IP** - deben ser IPs privadas
- **Application Load Balancer**
- Los controles de salud soportan los **protocolos TCP, HTTP y HTTPS**



Gateway Load Balancer

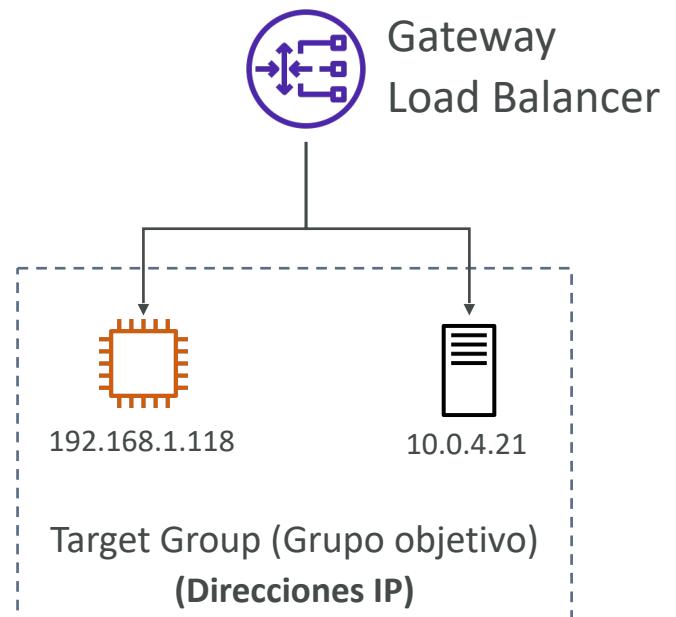
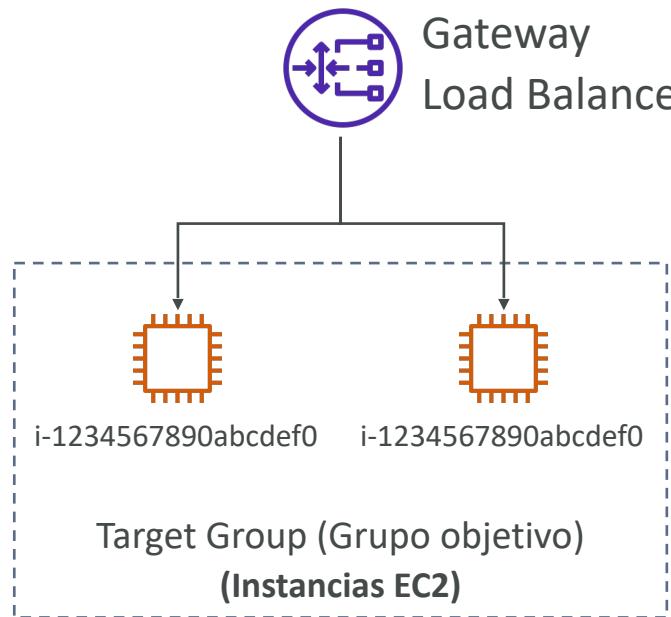
- Implementa, escala y administra una flota de dispositivos virtuales de red de terceros en AWS
- Ejemplo: Firewalls, Sistemas de Detección y Prevención de Intrusiones, Sistemas de Inspección Profunda de Paquetes, manipulación de cargas útiles, ...
- Opera en la Capa 3 (Capa de Red) - Paquetes IP
- Combina las siguientes funciones
 - **Gateway de Red Transparente** - entrada/salida única para todo el tráfico
 - **Load Balancer** - distribuye el tráfico a tus dispositivos virtuales
- Utiliza el protocolo **GENEVE** en el puerto 6081



Gateway Load Balancer

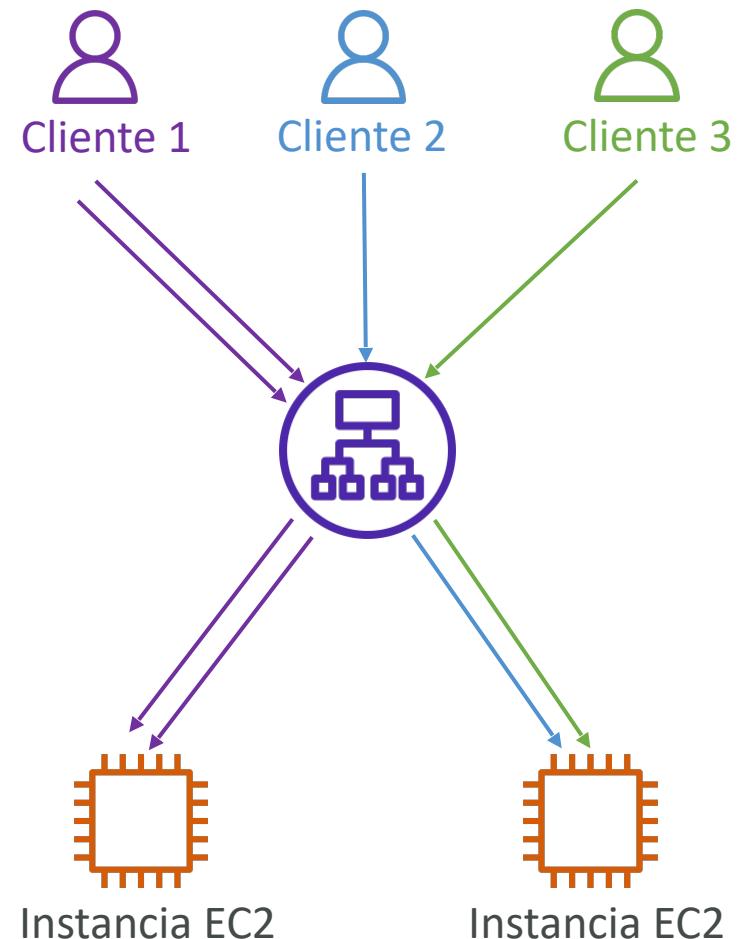
Target Groups (Grupos objetivo)

- **Instancias EC2**
- **Direcciones IP** - deben ser IPs privadas



Sticky Sessions (sesiones persistentes)

- Es posible implementar la “pegajosidad / adherencia” para que el mismo cliente sea siempre redirigido a la misma instancia detrás de un balanceador de carga
- Esto funciona para los Classic Load Balancer y los Application Load Balancer
- La "cookie" utilizada para la adherencia tiene una fecha de caducidad que tú controlas
- Caso de uso: asegurarse de que el usuario no pierde sus datos de sesión
- Activar la adherencia puede provocar un desequilibrio de la carga en las instancias EC2 del backend



Sticky Sessions – Nombres de las cookies

- **Cookies basadas en la aplicación**

- Cookie personalizada
 - Generada por el objetivo
 - Puede incluir cualquier atributo personalizado requerido por la aplicación
 - El nombre de la cookie debe especificarse individualmente para cada grupo de destino
 - No utilices **AWSALB**, **AWSALBAPP** o **AWSALBTG** (reservadas para el uso del ELB)
- Cookie de la aplicación
 - Generada por el Load Balancer
 - El nombre de la cookie es **AWSALBAPP**

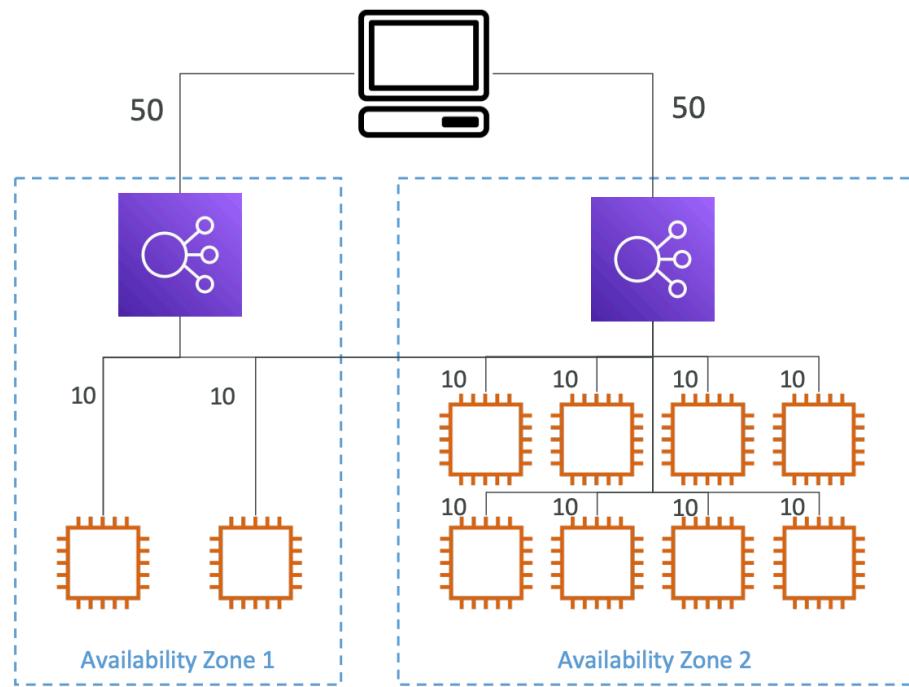
- **Cookies basadas en la duración**

- Cookie generada por el equilibrador de carga
- El nombre de la cookie es **AWSALB** para ALB, **AWSELB** para CLB

Load Balancer entre zonas

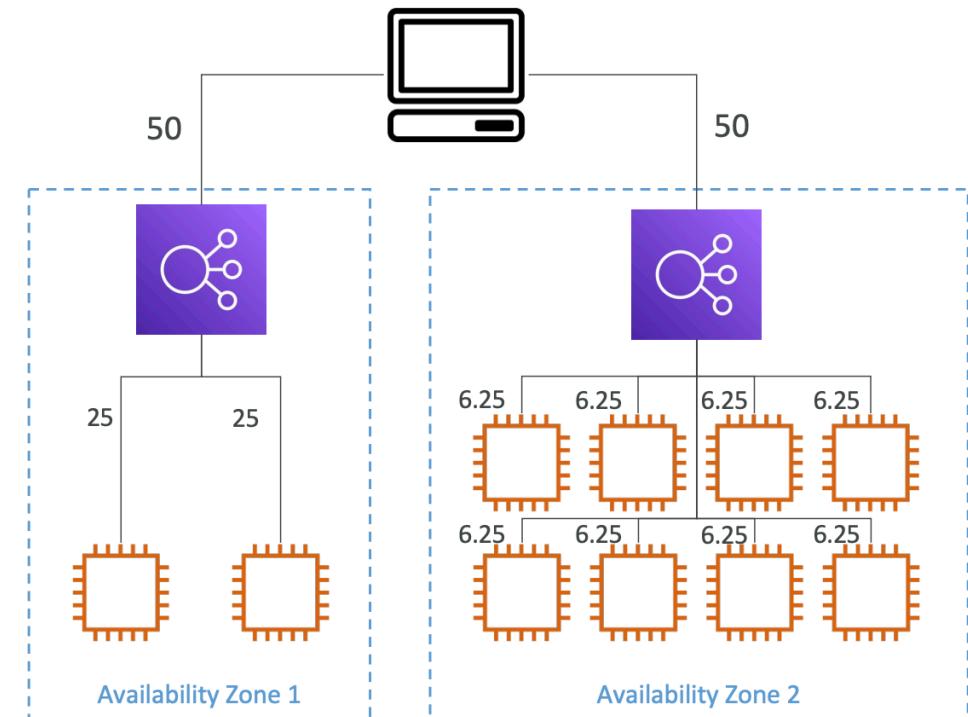
Con el Load Balancer de zona cruzada:

Cada instancia del Load Balancer distribuye uniformemente entre todas las instancias registradas en todas las AZ



Sin Load Balancer de zona cruzada:

Las solicitudes se distribuyen en las instancias del nodo del Elastic Load Balancer



Load Balancer entre zonas

- **Application Load Balancer**

- Siempre activado (no se puede desactivar)
- No se cobra por los datos inter AZ

- **Network Load Balancer & Gateway Load Balancer**

- Desactivado por defecto
- Si está activado, pagas una tarifa (\$) por los datos entre zonas geográficas

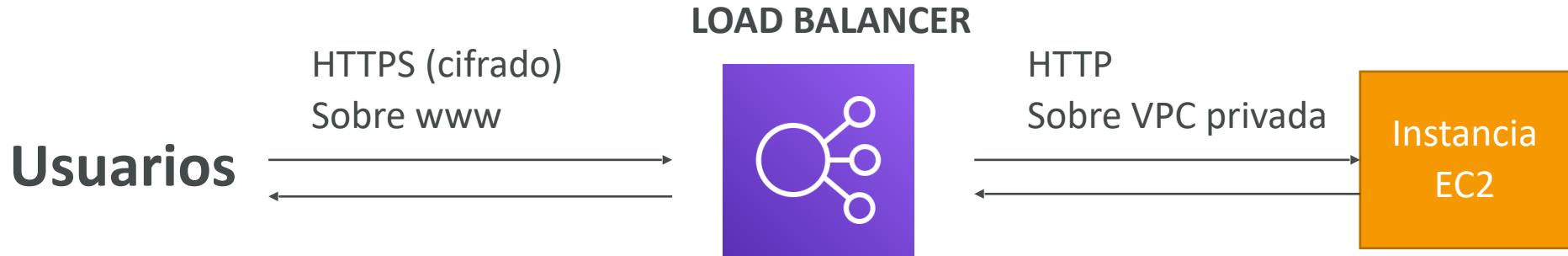
- **Classic Load Balancer**

- Desactivado por defecto
- No se cobra por los datos inter AZ si está activado

SSL/TLS - Conceptos básicos

- Un certificado SSL permite que el tráfico entre tus clientes y tu Load Balancer esté cifrado en tránsito (cifrado en vuelo)
- **SSL** hace referencia a Secure Sockets Layer, que se utiliza para cifrar las conexiones
- **TLS** se refiere a Transport Layer Security, que es una versión más reciente
- Hoy en día, **se utilizan principalmente los certificados TLS**, pero la gente sigue refiriéndose a ellos como SSL
- Los certificados SSL públicos son emitidos por las Autoridades de Certificación (CA)
- Comodo, Symantec, GoDaddy, GlobalSign, DigiCert, LetsEncrypt, etc.
- Los certificados SSL tienen una fecha de caducidad (que tú estableces) y deben ser renovados

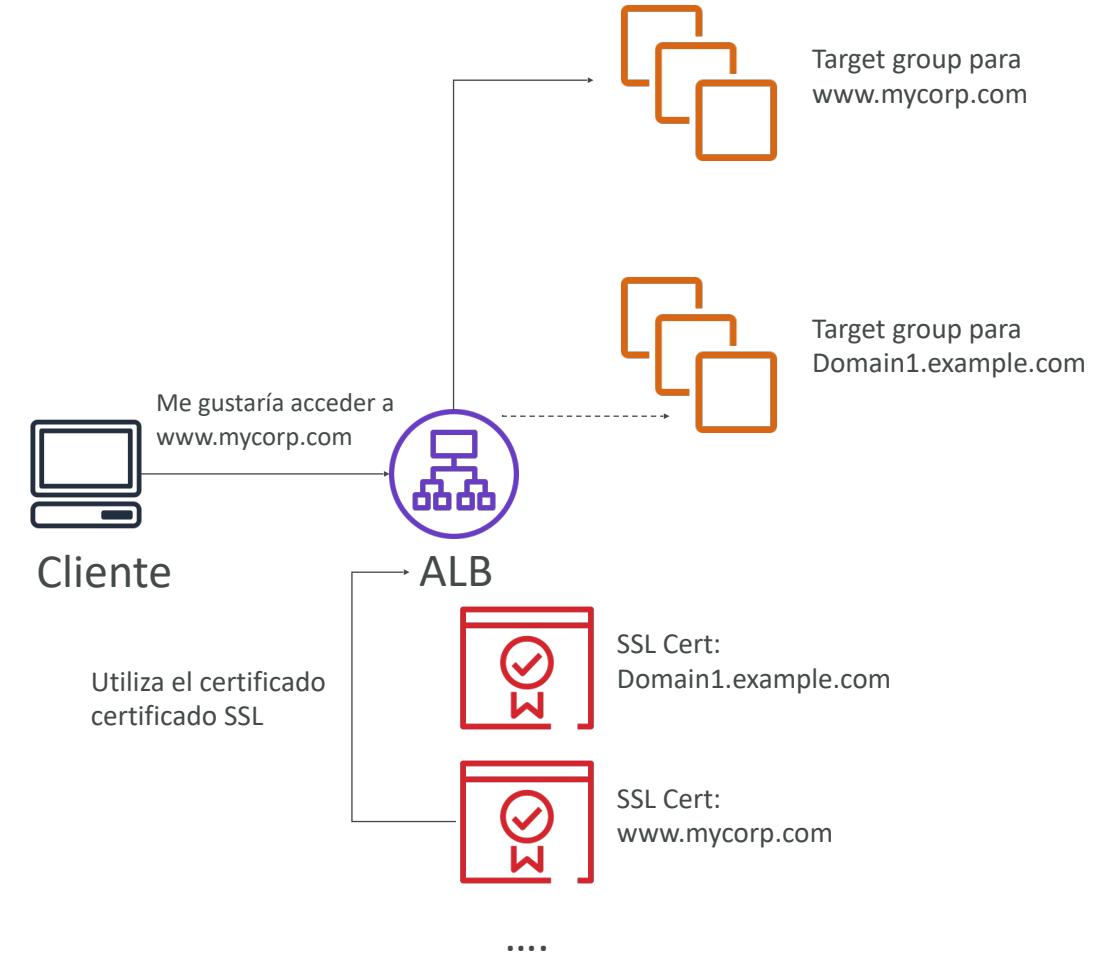
Load Balancer - Certificados SSL



- El Load Balancer utiliza un certificado X.509 (certificado de servidor SSL/TLS)
- Puedes gestionar los certificados mediante ACM (AWS Certificate Manager)
- También puedes crear y subir tus propios certificados
- Listener HTTPS:
 - Debes especificar un certificado por defecto
 - Puedes añadir una lista opcional de certificados para dar soporte a múltiples dominios
 - **Los clientes pueden utilizar SNI (Server Name Indication) para especificar el nombre de host al que llegan**
 - Posibilidad de especificar una política de seguridad para soportar versiones antiguas de SSL / TLS (clientes heredados)

SSL – Server Name Indication (SNI)

- SNI resuelve el problema de **cargar varios certificados SSL en un servidor web** (para servir a varios sitios web)
- Es un protocolo "más nuevo", y requiere que el cliente **indique** el nombre del servidor de destino en el apretón de manos SSL inicial (*handshake*)
- El servidor encontrará entonces el certificado correcto, o devolverá el predeterminado
- Nota:
 - Sólo funciona para ALB y NLB (generación más reciente), CloudFront
 - No funciona con CLB (generación anterior)



Elastic Load Balancers - Certificados SSL

- **Classic Load Balancer (v1)**

- Soporta sólo un certificado SSL
- Debe utilizar varios CLB para varios nombres de host con varios certificados SSL

- **Application Load Balancer (v2)**

- Soporta múltiples oyentes con múltiples certificados SSL
- Utiliza la indicación del nombre del servidor (SNI) para que funcione

- **Network Load Balancer (v2)**

- Soporta múltiples oyentes con múltiples certificados SSL
- Utiliza la Indicación del Nombre del Servidor (SNI) para hacerlo funcionar

Connection Draining (Drenaje de la conexión)

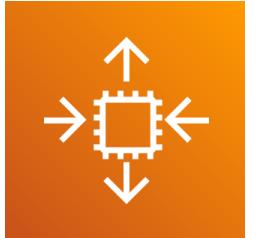
- **Nombre de la característica**

- Drenaje de la conexión - para el CLB
- Retraso en el desregistro - para ALB y NLB

- Tiempo para completar las "peticiones en vuelo" mientras la instancia se está desregistrando o no está sana
- Deja de enviar nuevas peticiones a la instancia EC2 que se está desregistrando
- Entre 1 y 3600 segundos (por defecto: 300 segundos)
- Se puede desactivar (fijar el valor en 0)
- Establece un valor bajo si tus peticiones son cortas

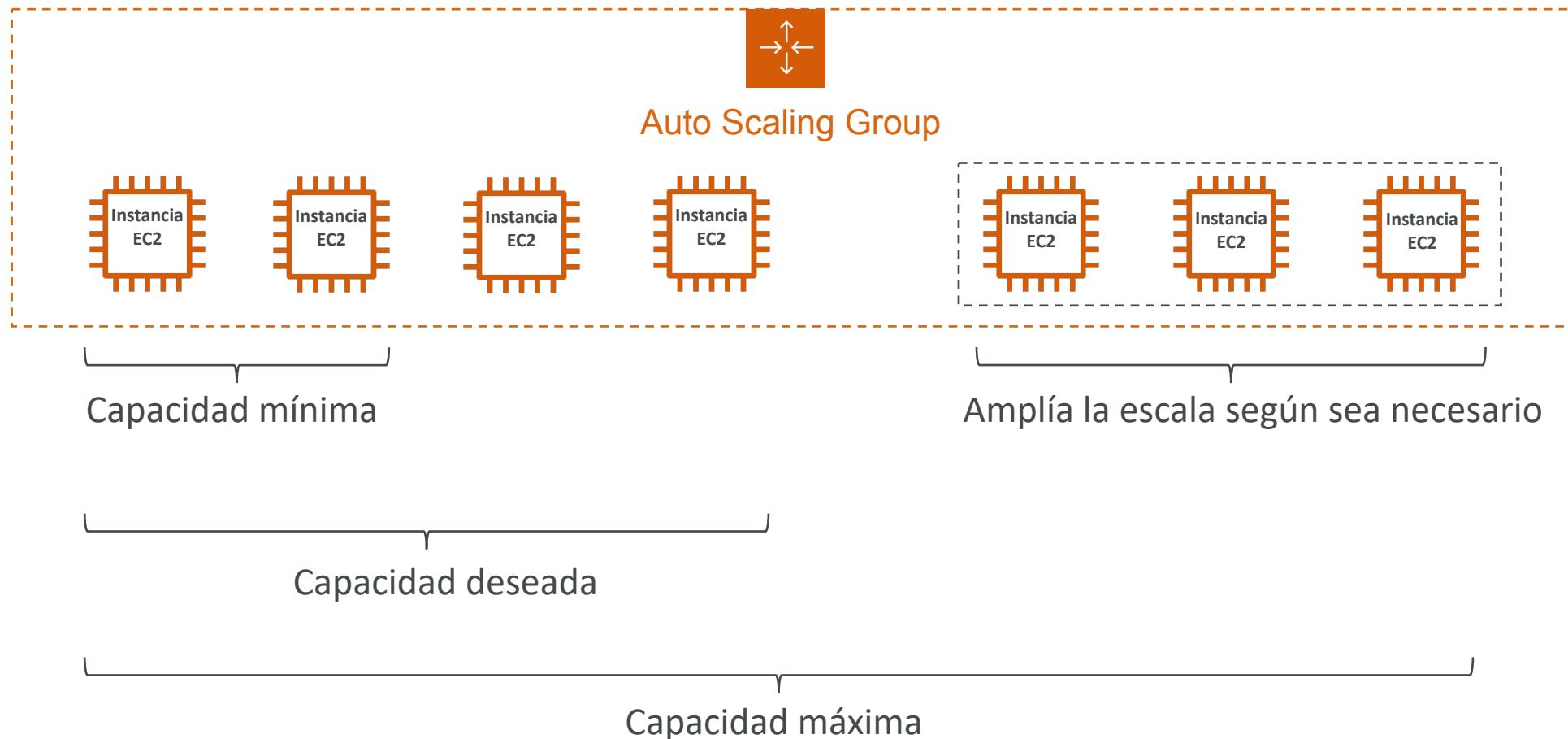


¿Qué es un Auto Scaling Group?

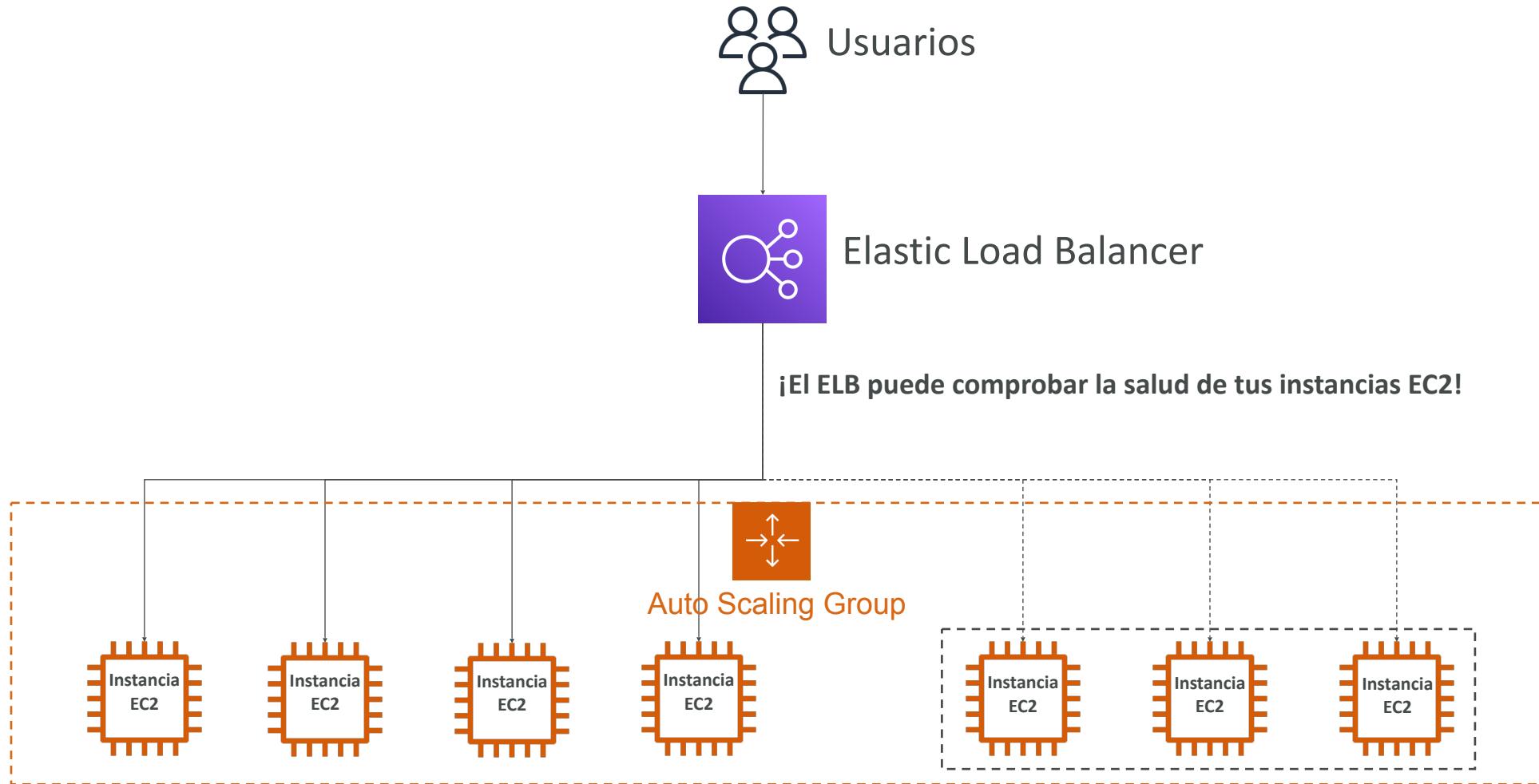


- En la vida real, la carga de tus sitios web y aplicaciones puede cambiar
- En el Cloud, puedes crear y deshacerte de servidores muy rápidamente
- El objetivo de un Auto Scaling Group (ASG) es
 - Reducir (añadir instancias EC2) para adaptarse a un aumento de la carga
 - Aumentar (eliminar instancias EC2) para que coincida con una disminución de la carga
 - Asegurar que tenemos un número mínimo y máximo de instancias EC2 en funcionamiento
 - Registrar automáticamente nuevas instancias en un Load Balancer
 - Volver a crear una instancia EC2 en caso de que se elimine una anterior (por ejemplo, si no está sana)
- Los ASG son gratuitos (sólo pagas por las instancias EC2 subyacentes)

Auto Scaling Group en AWS



Auto Scaling Group en AWS con Load Balancer



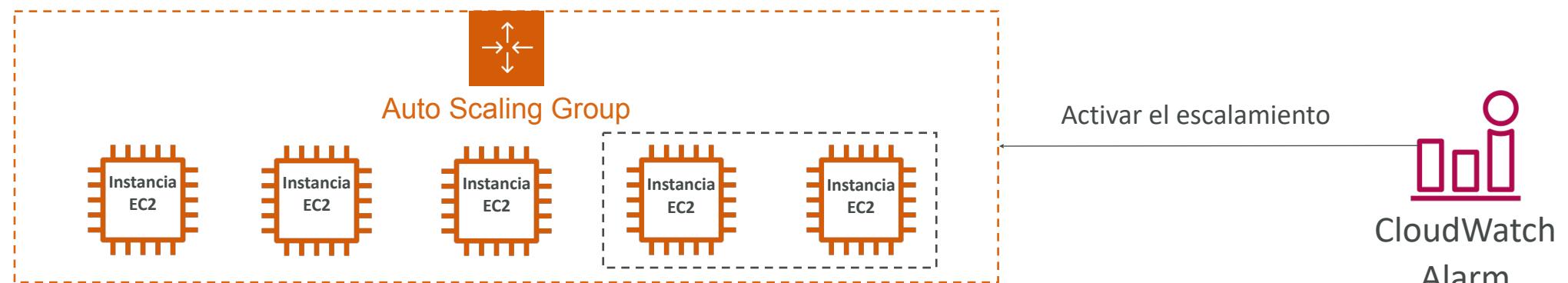
Atributos del Auto Scaling Group

- Una **Plantilla de Lanzamiento** (las antiguas "Configuraciones de Lanzamiento" están obsoletas)
 - AMI + Tipo de Instancia
 - Datos de usuario de EC2
 - Volúmenes EBS
 - Grupos de seguridad
 - Par de claves SSH
 - Roles IAM para tus instancias EC2
 - Información sobre la red y las subredes
 - Información del Load Balancer
- Tamaño mínimo / Tamaño máximo / Capacidad inicial
- Políticas de escalado



Escalado automático Alarms y escalado de CloudWatch

- Es posible escalar un ASG basándose en las alarmas de CloudWatch
- Una alarma monitoriza una métrica (como la CPU media, o una métrica personalizada)
- Las métricas, como la CPU media, se calculan para todas las instancias del ASG
- Basándonos en la alarma
 - Podemos crear políticas de escalado (aumentar el número de instancias)
 - Podemos crear políticas de ampliación (reducir el número de instancias)



Auto Scaling Group

Políticas de escalado dinámico

- **Escala de seguimiento de objetivos**

- Lo más sencillo y fácil de configurar
- Ejemplo: Quiero que la media de la CPU de ASG se mantenga en torno al 40%

- **Escalado simple / escalonado**

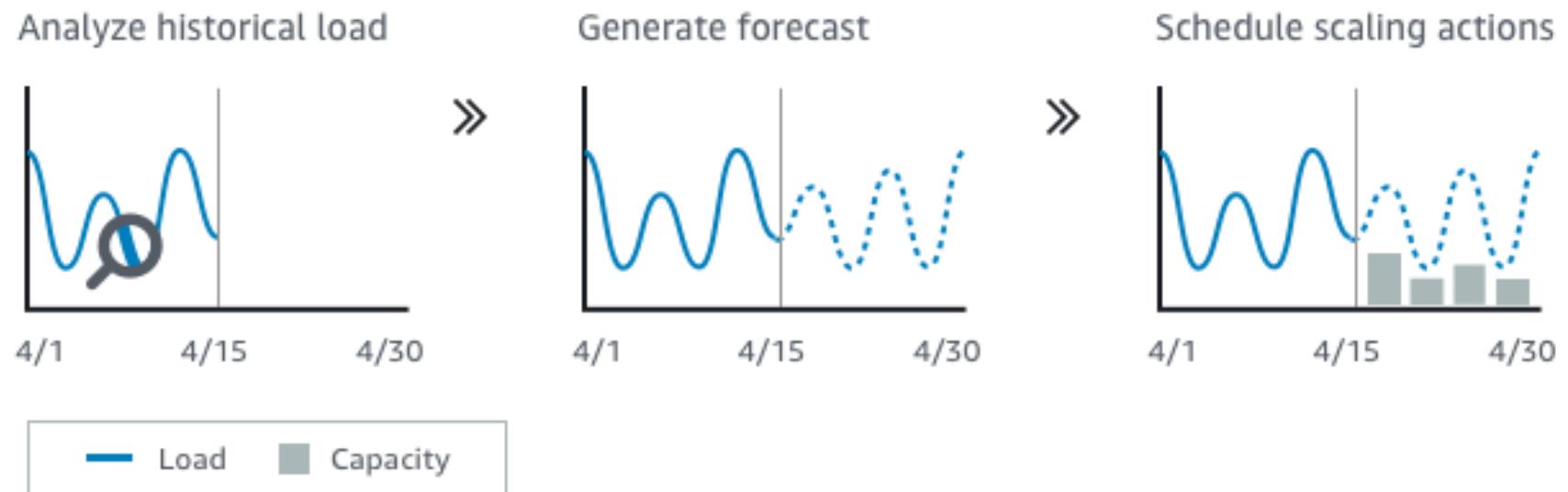
- Cuando se active una alarma de CloudWatch (por ejemplo, CPU > 70%), añade 2 unidades
- Cuando se active una alarma de CloudWatch (ejemplo CPU < 30%), entonces elimina 1

- **Acciones programadas**

- Anticipa un escalado basado en patrones de uso conocidos
- Ejemplo: aumentar la capacidad mínima a 10 a las 17 horas de los viernes

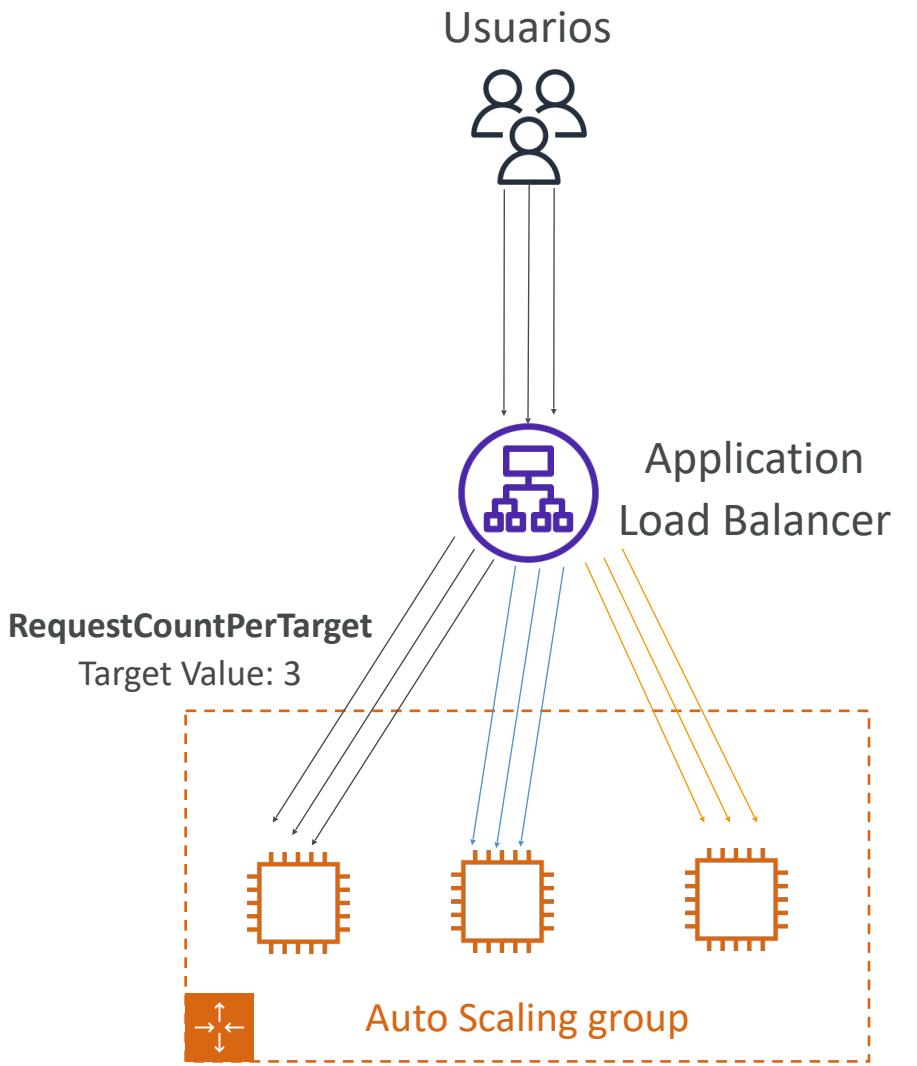
Auto Scaling Group - Escalado predictivo

- **Escalado predictivo:** previsión continua de la carga y programación del escalado por adelantado



Buenas métricas para escalar

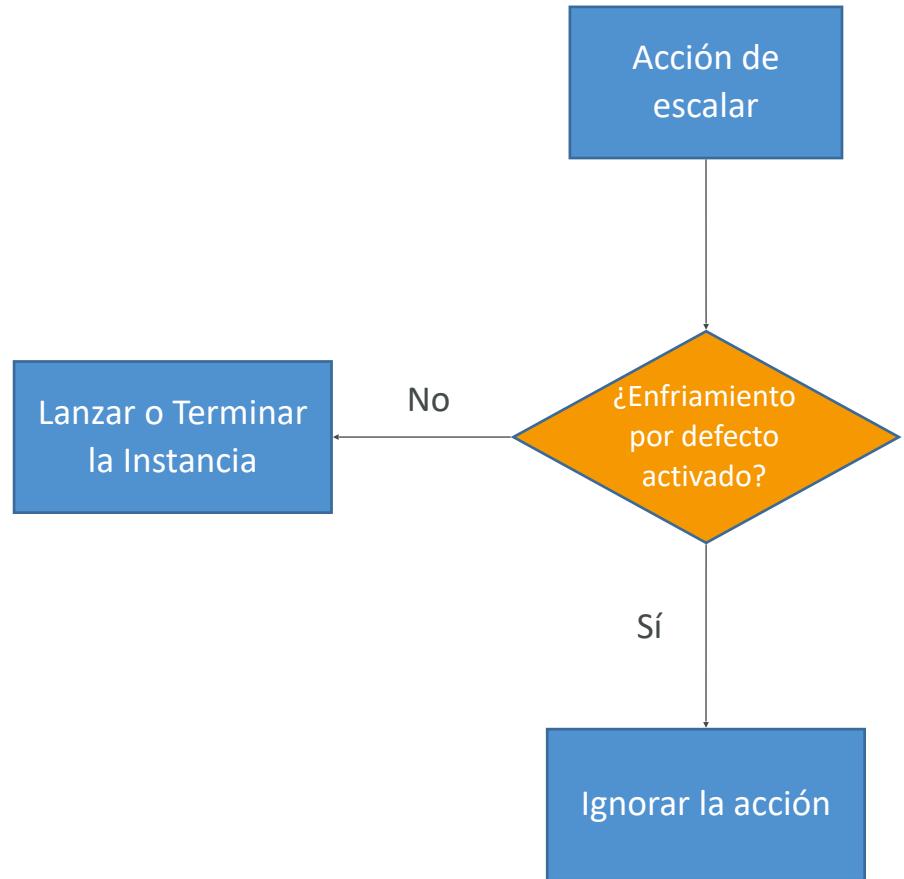
- **CPUUtilization**: Utilización media de la CPU en tus instancias
- **RequestCountPerTarget**: para asegurarse de que el número de peticiones por instancias EC2 es estable
- **Promedio de entrada/salida** de red (si tu aplicación está vinculada a la red)
- **Cualquier métrica personalizada** (que impulse con CloudWatch)



Auto Scaling Groups

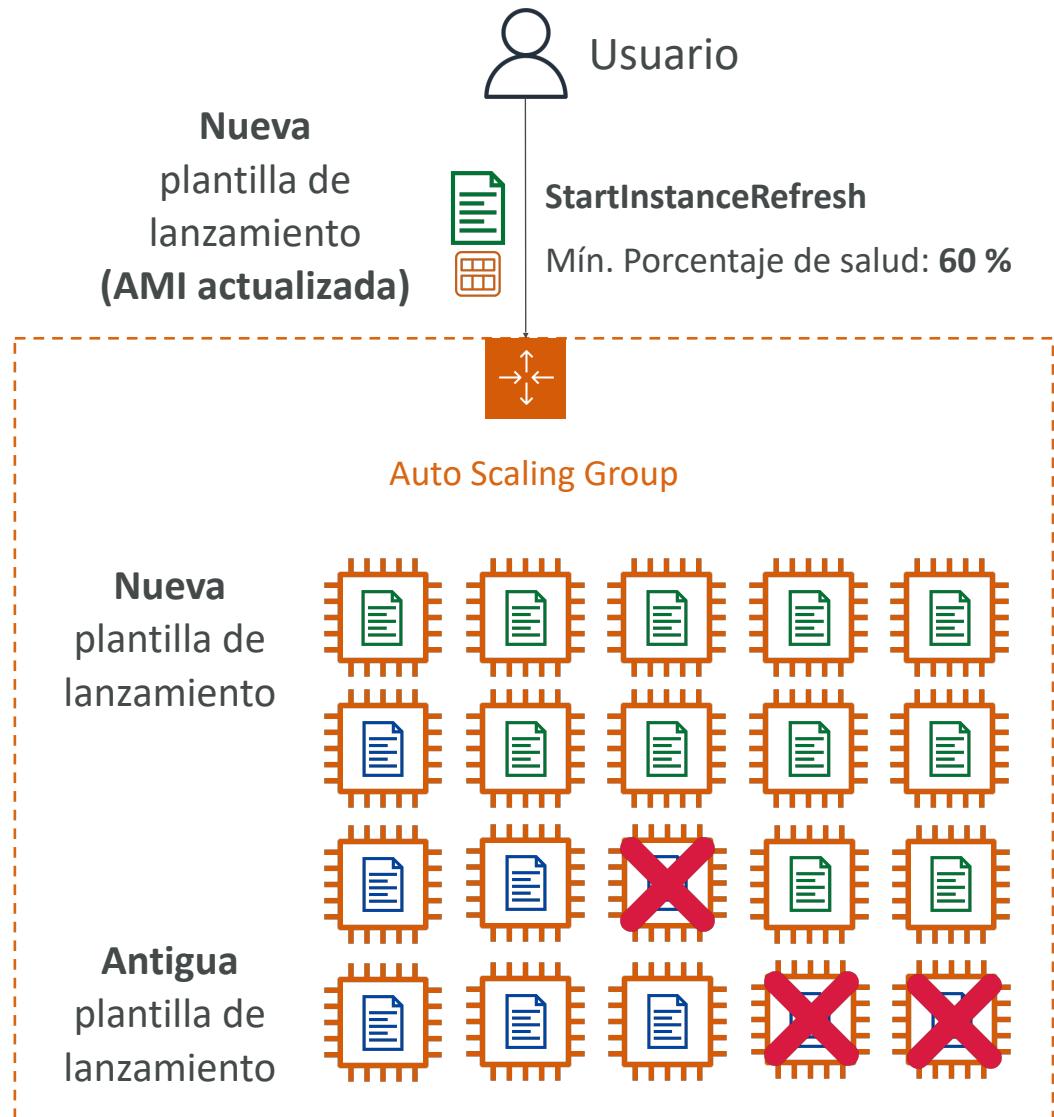
Enfriamiento de la escala

- Después de que se produzca una actividad de escalado, **estarás en el periodo de enfriamiento (por defecto, 300 segundos)**
- Durante el periodo de enfriamiento, el ASG no lanzará ni terminará instancias adicionales (para permitir que las métricas se estabilicen)
- Consejo: Utiliza una AMI lista para usar para reducir el tiempo de configuración y así poder servir las peticiones más rápidamente y reducir el periodo de enfriamiento



Escalado automático - Actualización de instancias

- Objetivo: actualizar la plantilla de lanzamiento y luego volver a crear todas las instancias EC2
- Para ello podemos utilizar la función nativa de Actualización de Instancias
- Establecer el porcentaje mínimo de salud
- Especificar el tiempo de calentamiento (cuánto tiempo hasta que la instancia esté lista para usarse)



Fundamentos de AWS - Parte III

RDS, Aurora y ElastiCache

Visión general de Amazon RDS



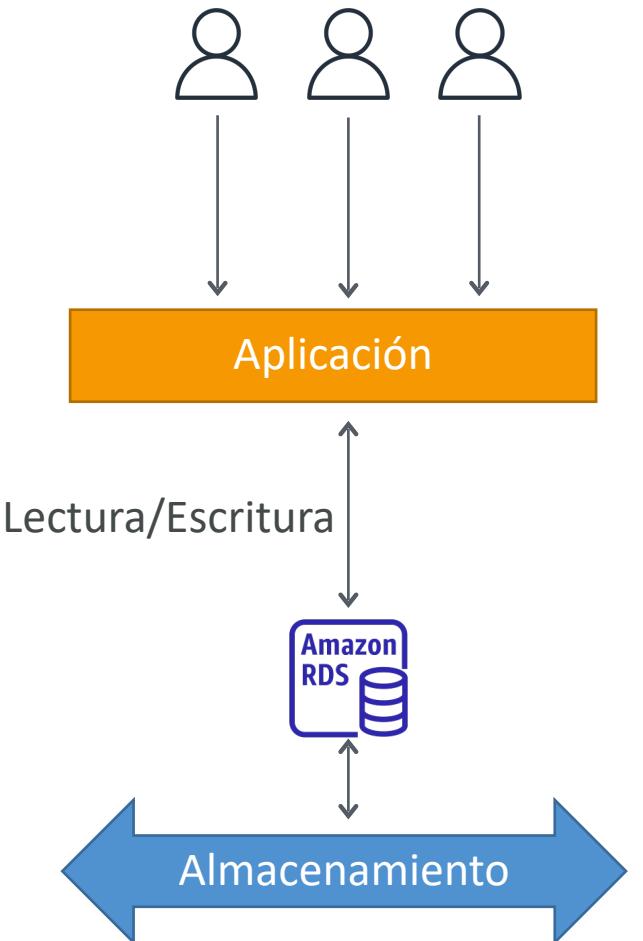
- RDS significa Servicio de Base de Datos Relacional
- Es un servicio de bases de datos gestionado para que las bases de datos utilicen SQL como lenguaje de consulta.
- Permite crear bases de datos en el Cloud que son gestionadas por AWS
 - Postgres
 - MySQL
 - MariaDB
 - Oracle
 - Microsoft SQL Server
 - Aurora (base de datos propia de AWS)

Ventaja sobre el uso de RDS frente al despliegue de la BD en EC2

- El RDS es un servicio gestionado:
 - Aprovisionamiento automatizado, parcheo del SO
 - Copias de seguridad continuas y restauración a una fecha determinada (Point in Time Restore)
 - Dashboards de monitorización
 - Rélicas de lectura para mejorar el rendimiento de lectura
 - Configuración multi AZ para DR (Disaster Recovery)
 - Ventanas de mantenimiento para actualizaciones
 - Capacidad de escalado (vertical y horizontal)
 - Almacenamiento respaldado por EBS (gp2 o io1)
- **PERO no puedes acceder por SSH a tus instancias**

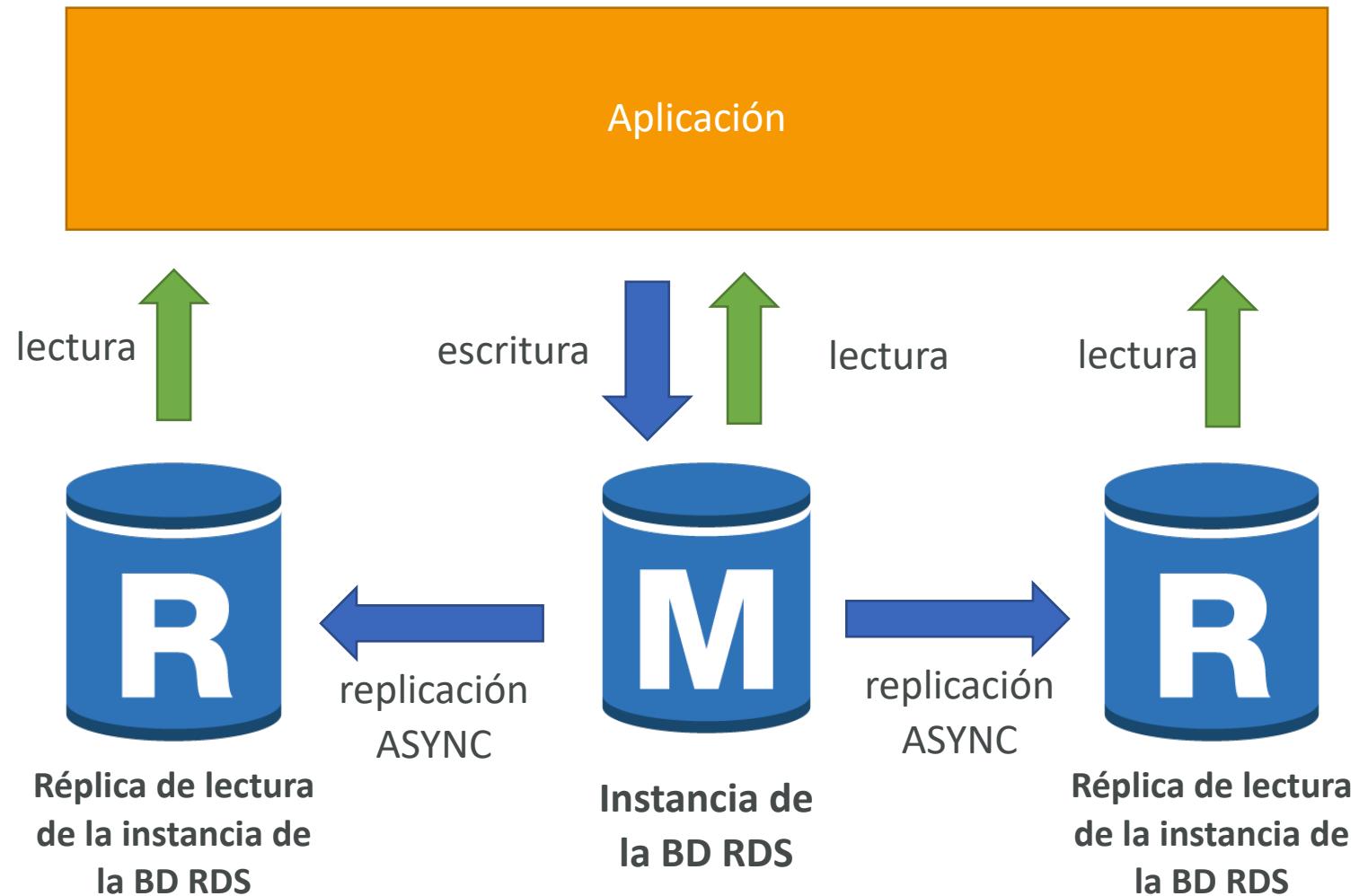
RDS – Autoescalamiento de almacenamiento

- Te ayuda a aumentar el almacenamiento de tu instancia de base de datos RDS de forma dinámica
- Cuando RDS detecta que te estás quedando sin almacenamiento gratuito en la base de datos, escala automáticamente
- Evita escalar manualmente el almacenamiento de tu base de datos
- Tienes que establecer el **Umbral Máximo de Almacenamiento** (límite máximo de almacenamiento de la BD)
- Modifica automáticamente el almacenamiento si
 - El almacenamiento gratuito es inferior al 10% del almacenamiento asignado
 - El almacenamiento bajo dura al menos 5 minutos
 - Han pasado 6 horas desde la última modificación
- Útil para aplicaciones con **cargas de trabajo imprevisibles**
- Soporta todos los motores de bases de datos RDS (MariaDB, MySQL, PostgreSQL, SQL Server, Oracle)



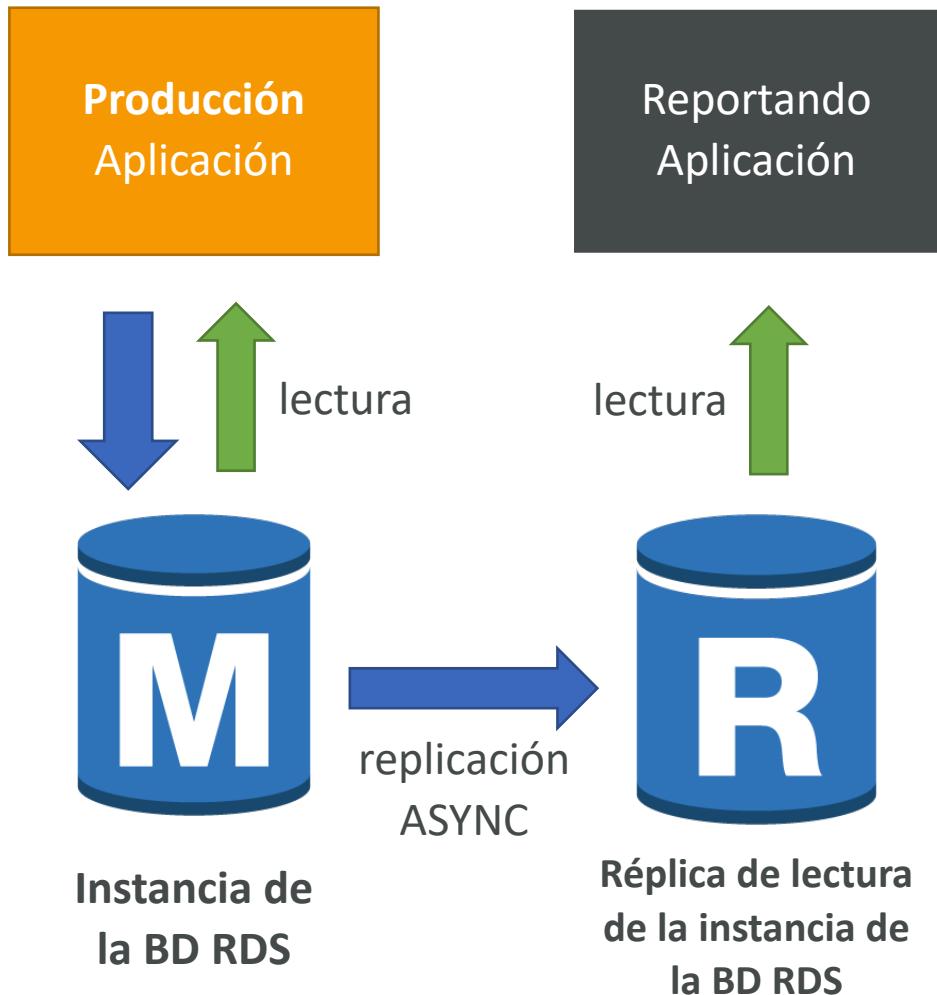
Réplicas de lectura RDS para la escalabilidad de lectura

- Hasta 5 réplicas de lectura
- Dentro de AZ, a través de AZ o a través de la región
- La replicación es **ASYNC**, por lo que las lecturas son finalmente consistentes
- Las réplicas pueden ser promovidas a su propia BD
- Las aplicaciones deben actualizar la cadena de conexión para aprovechar las réplicas de lectura



Réplicas de lectura RDS - Casos de uso

- Tienes una base de datos de producción que está recibiendo una carga normal
- Quieres ejecutar una aplicación de informes para realizar algunos análisis
- Creas una réplica de lectura para ejecutar allí la nueva carga de trabajo
- La aplicación de producción no se ve afectada
- Las réplicas de lectura se utilizan sólo para sentencias del tipo SELECT (=lectura) (no INSERT, UPDATE, DELETE)



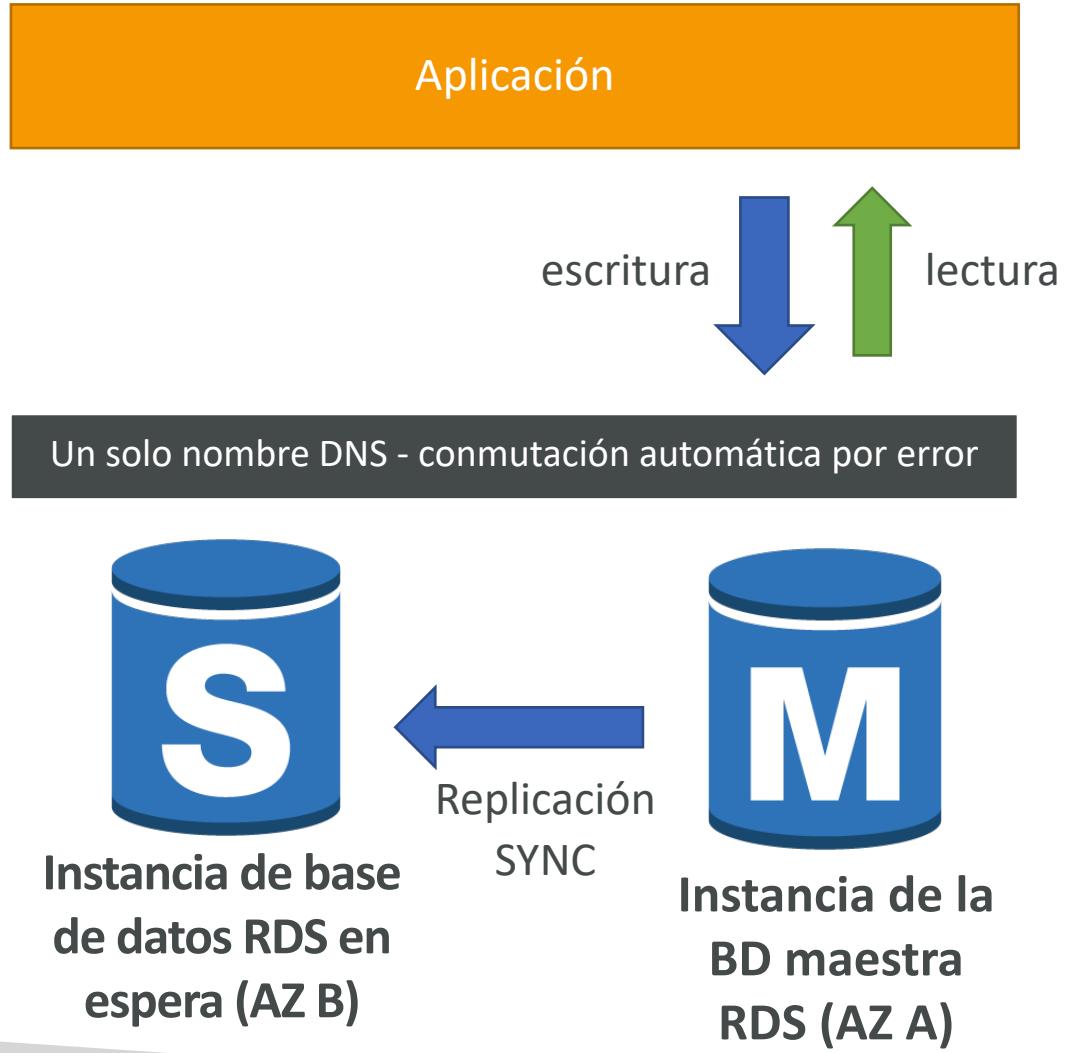
Réplicas de lectura RDS - Coste de la red

- En AWS hay un coste de red cuando los datos van de una AZ a otra
- Para las Réplicas de Lectura RDS dentro de la misma región, no pagas esa tarifa



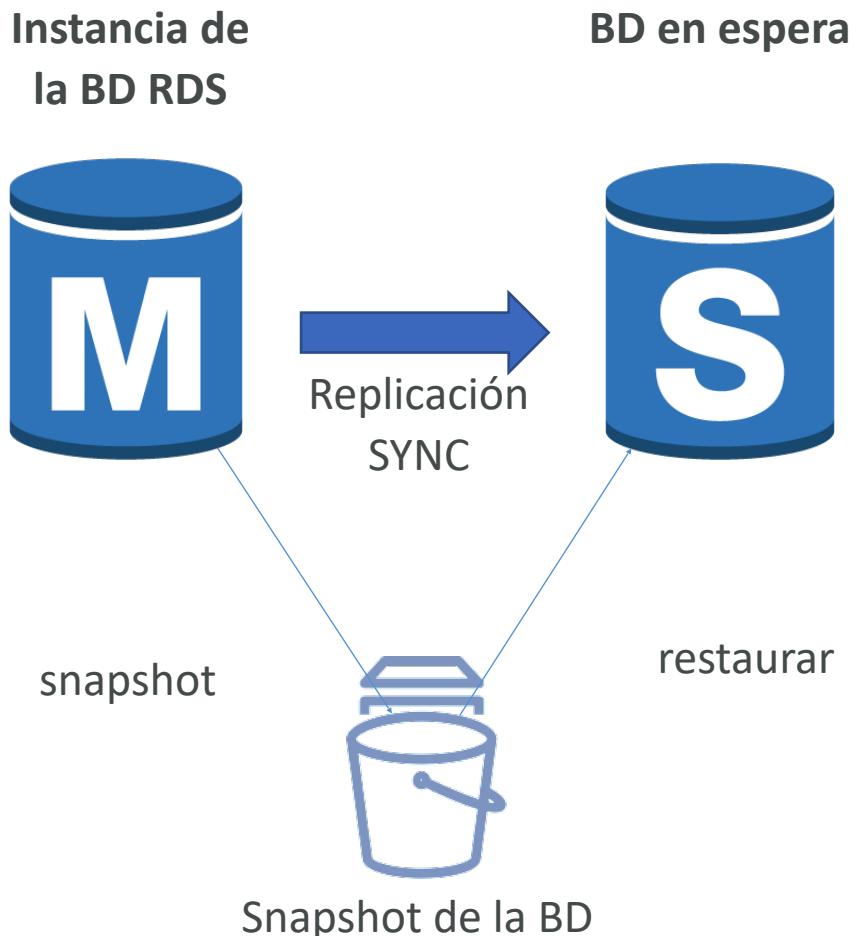
RDS Multi AZ (recuperación de desastres)

- Replicación SYNC
- Un nombre DNS - conmutación automática de la aplicación a la espera
- Aumenta la disponibilidad
- Conmutación por error en caso de pérdida de AZ, pérdida de red, fallo de instancia o de almacenamiento
- Sin intervención manual en las apps
- No se utiliza para escalar
- La replicación multi-AZ es gratis
- Nota: Las réplicas de lectura deben configurarse como Multi AZ para la recuperación de desastres (DR)



RDS – De una AZ a múltiples AZ

- Operación sin tiempo de inactividad (no es necesario parar la BD)
- Sólo tienes que hacer clic en "modificar" la base de datos
- Internamente ocurre lo siguiente
 - Se toma un Snapshot
 - Se restaura una nueva BD a partir del Snapshot en una nueva AZ
 - Se establece la sincronización entre las dos bases de datos



RDS Personalizada

- **Base de datos gestionada de Oracle y Microsoft SQL Server con personalización del sistema operativo y de la base de datos**
- RDS: automatiza la configuración, el funcionamiento y el escalado de la base de datos en AWS
- Personalizada: acceso a la base de datos subyacente y al SO para que puedas
 - Configurar los ajustes
 - Instalar parches
 - Habilitar las funciones nativas
 - Acceder a la instancia EC2 subyacente mediante **SSH** o **SSM Session Manager**
- **Desactivar el Modo de Automatización** para realizar tu personalización, mejor tomar una Snapshot de la BD antes
- RDS vs. RDS Personalizada
 - RDS: toda la base de datos y el SO serán gestionados por AWS
 - RDS Personalizada: acceso administrativo completo al SO subyacente y a la base de datos



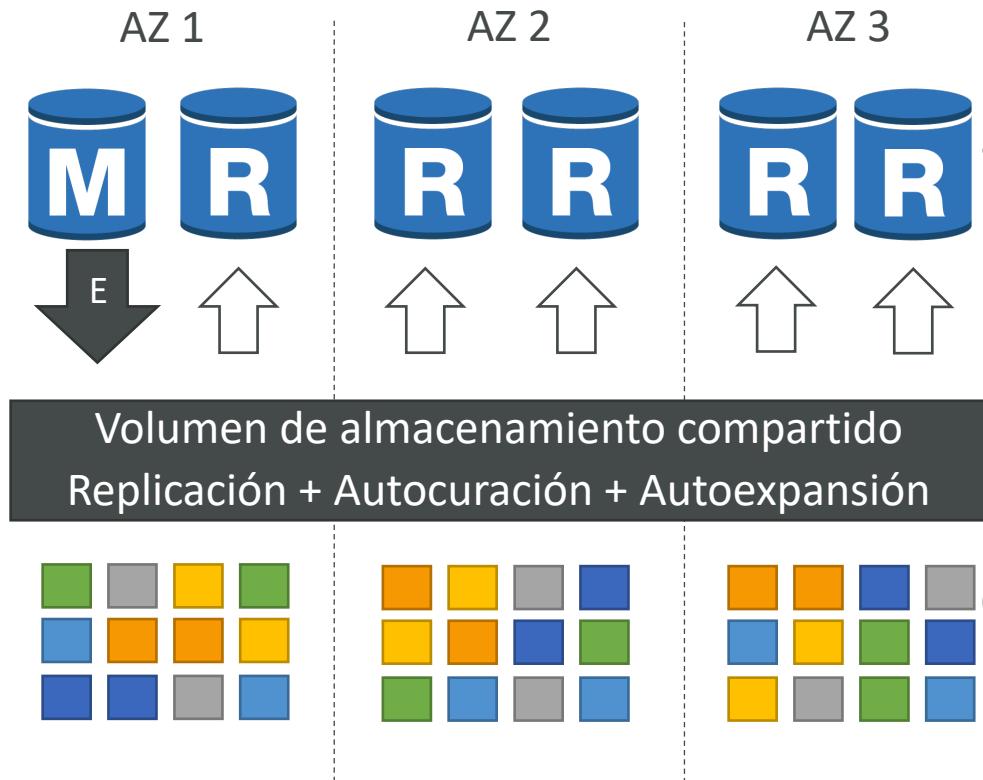


Amazon Aurora

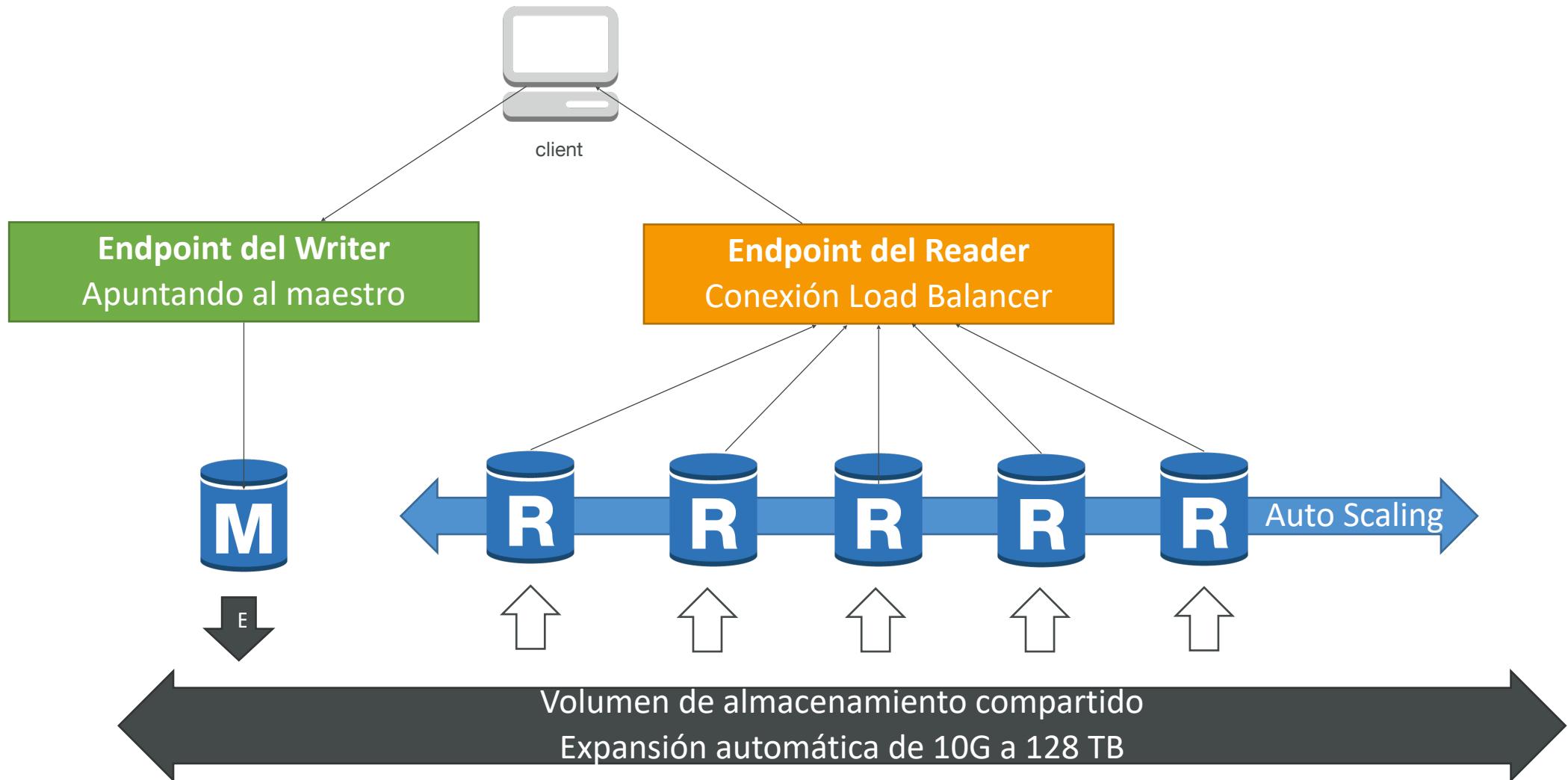
- Aurora es una tecnología propietaria de AWS (no de código abierto)
- Tanto Postgres como MySQL se soportan como base de datos de Aurora (eso significa que tus controladores funcionarán como si Aurora fuera una base de datos Postgres o MySQL)
- Aurora está "optimizada para la Cloud de AWS" y afirma que mejora 5 veces el rendimiento de MySQL en RDS, y más de 3 veces el rendimiento de Postgres en RDS
- El almacenamiento de Aurora crece automáticamente en incrementos de 10 GB, hasta 128 TB.
- Aurora puede tener 15 réplicas, mientras que MySQL tiene 5, y el proceso de replicación es más rápido (retardo de réplica inferior a 10 ms)
- La conmutación por error en Aurora es instantánea. Es nativo HA (Alta Disponibilidad).
- Aurora cuesta más que RDS (un 20% más), pero es más eficiente.

Alta disponibilidad y escalado de lectura de Aurora

- 6 copias de tus datos en 3 AZ:
 - 4 copias de las 6 necesarias para las escrituras
 - 3 copias de las 6 necesarias para las lecturas
 - Autoreparación con replicación entre pares
 - El almacenamiento está dividido en 100 volúmenes
- Una instancia de Aurora se encarga de las escrituras (maestra)
- Recuperación automática del maestro en menos de 30 segundos
- El maestro + hasta 15 réplicas de lectura de Aurora realizan lecturas
- Soporte para la replicación entre regiones



Cluster de BD Aurora



Características de Aurora

- Comutación automática por error
- Copia de seguridad y recuperación
- Aislamiento y seguridad
- Cumplimiento de la normativa del sector
- Escalado con un botón
- Parches automáticos con cero tiempo de inactividad
- Supervisión avanzada
- Mantenimiento rutinario
- Backtrack: restaura los datos en cualquier momento sin usar copias de seguridad

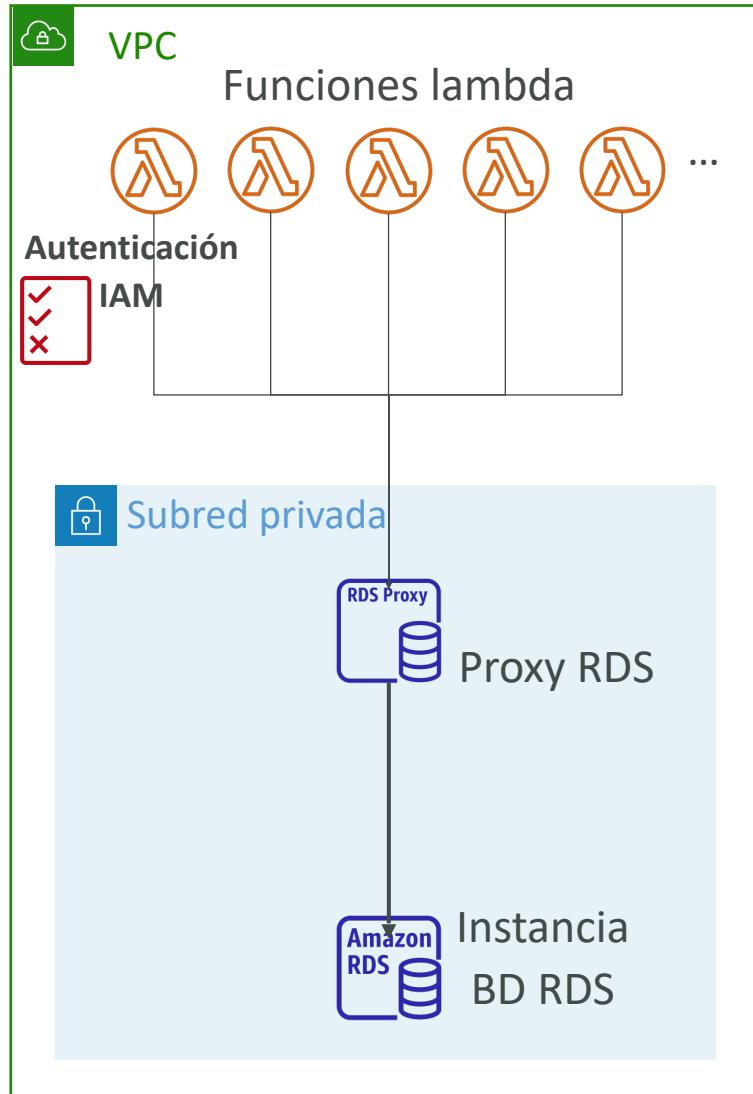
Seguridad RDS y Aurora

- **Cifrado en reposo:**
 - Cifrado de la base de datos maestra y de las réplicas mediante AWS KMS - debe definirse en el momento del lanzamiento.
 - Si la base maestra no está cifrada, las réplicas de lectura no pueden ser cifradas
 - Para cifrar una base de datos no cifrada, pasa por un Snapshot de la base de datos y restaura como cifrada
- **Cifrado en vuelo:** Preparado para TLS por defecto, utiliza certificados root del lado del cliente de AWS TLS
- **Autenticación IAM:** Roles de IAM para conectarse a tu base de datos (en lugar de nombre de usuario/pw)
- **Grupos de seguridad:** Controla el acceso de red a tu RDS / Aurora DB
- **No hay SSH disponible** excepto en RDS Custom
- **Los logs de auditoría pueden ser activados** y enviados a CloudWatch Logs para una mayor retención

Proxy de Amazon RDS



- Proxy de base de datos totalmente gestionado para RDS
- Permite a las apps agrupar y compartir las conexiones a la base de datos establecidas
- **Mejora la eficiencia de la base de datos reduciendo el estrés de los recursos de la base de datos (por ejemplo, CPU, RAM) y minimizando las conexiones abiertas (y los tiempos de espera)**
- Sin servidor, con autoescalado y alta disponibilidad (multi-AZ)
- Reduce el tiempo de comutación por error de RDS y Aurora hasta en un 66%
- **Soporta RDS (MySQL, PostgreSQL, MariaDB) y Aurora (MySQL, PostgreSQL)**
- No se requieren cambios de código para la mayoría de las aplicaciones
- **Aplica la autenticación IAM para la base de datos y almacena de forma segura las credenciales en AWS Secrets Manager**
- **El proxy RDS nunca es accesible al público (debe accederse desde la VPC)**





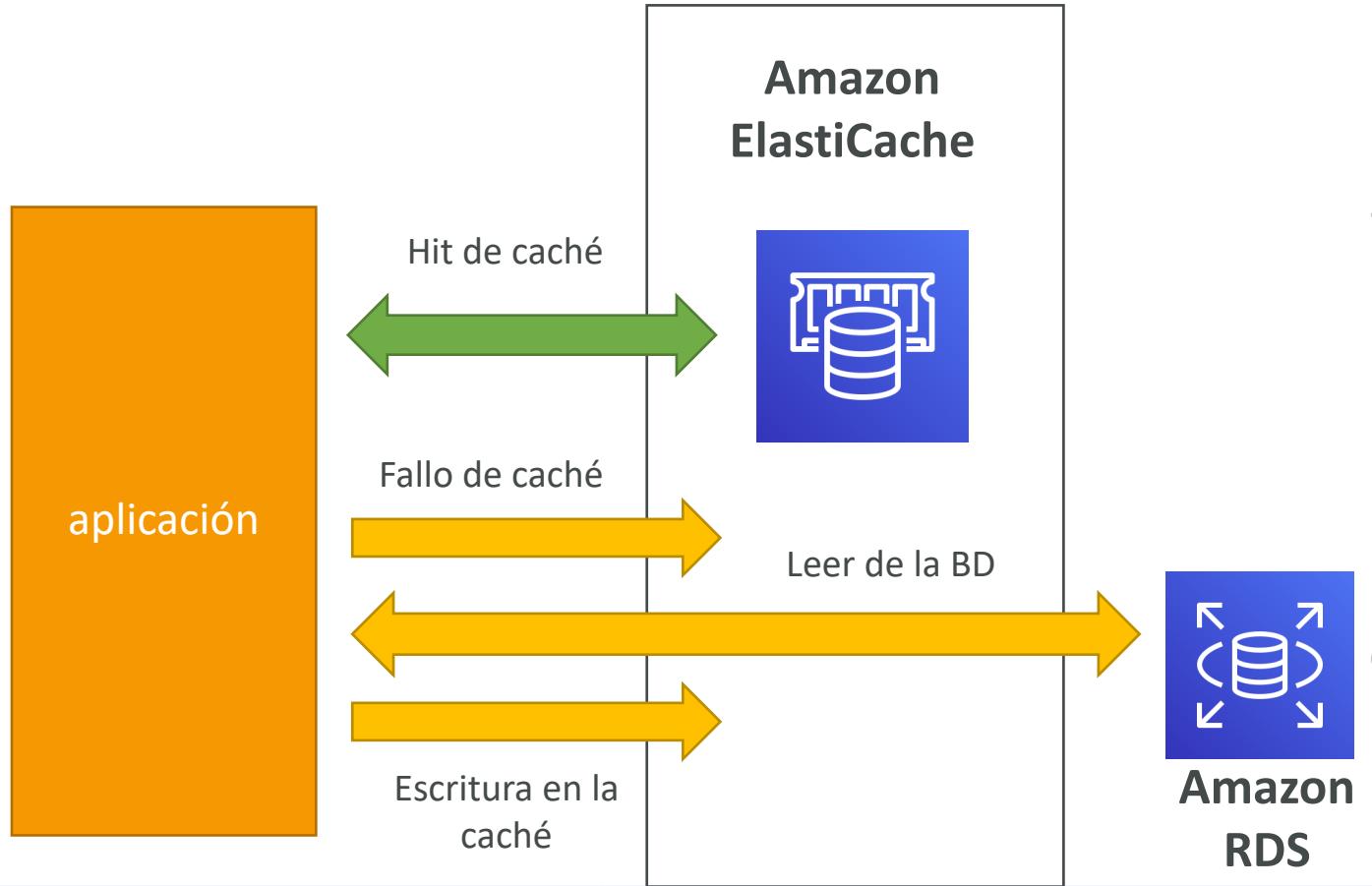
Visión general de Amazon ElastiCache

- De la misma manera que RDS es para conseguir bases de datos relacionales gestionadas...
- ElastiCache es para obtener Redis o Memcached gestionados
- Las cachés son bases de datos en memoria con un rendimiento realmente alto y baja latencia
- Ayuda a reducir la carga de las bases de datos para cargas de trabajo de lectura intensiva
- Ayuda a que tu aplicación no tenga estado
- AWS se encarga del mantenimiento/parche del sistema operativo, las optimizaciones, la instalación, la configuración, la supervisión, la recuperación de fallos y las copias de seguridad
- **El uso de ElastiCache implica grandes cambios en el código de la aplicación**

ElastiCache

Solución de arquitectura - DB Cache

- Las aplicaciones consultan ElastiCache, si no está disponible, lo obtienen del RDS y lo almacenan en ElastiCache.
- Ayuda a aliviar la carga en RDS
- La caché debe tener una estrategia de invalidación para asegurarse de que sólo se utilizan allí los datos más actuales.



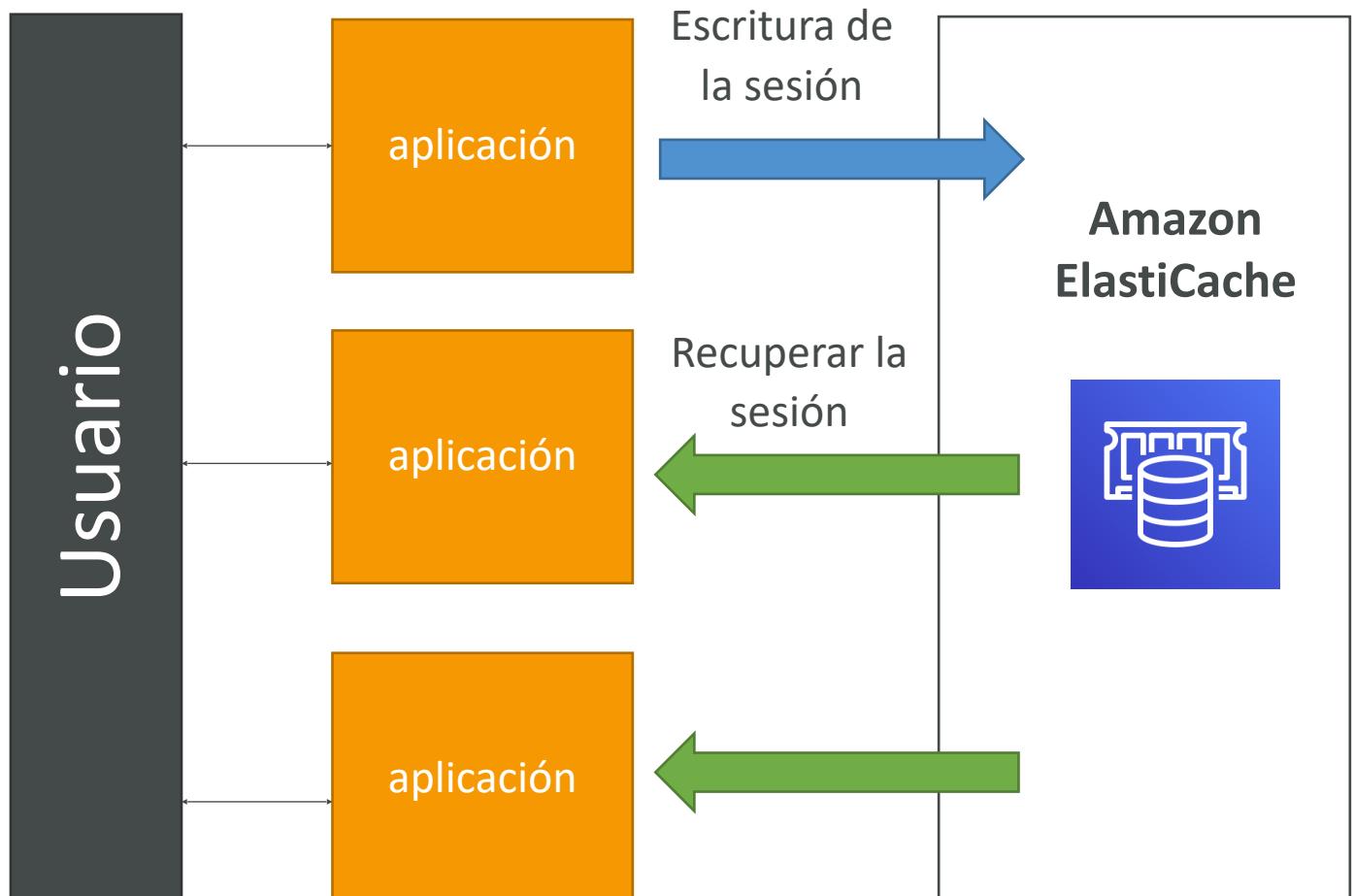
Un **hit de caché** es un estado en el que los datos solicitados para el procesamiento por un componente o aplicación se encuentran en la memoria caché.

Un **fallo de caché** es cuando los datos que solicita un sistema o una aplicación no se encuentran en la memoria caché.

ElastiCache

Solución de arquitectura - Almacén de sesiones de usuario

- El usuario se loguea en cualquiera de las aplicaciones
- La aplicación escribe los datos de la sesión en ElastiCache
- El usuario accede a otra instancia de nuestra aplicación
- La instancia recupera los datos y el usuario ya ha iniciado la sesión



ElastiCache – Redis vs Memcached

REDIS

- **Multi AZ** con Auto-Failover
- **Rélicas de lectura** para escalar las lecturas y tener **alta disponibilidad**
- Durabilidad de los datos mediante la persistencia AOF
- **Funciones de copia de seguridad y restauración**



MEMCACHED

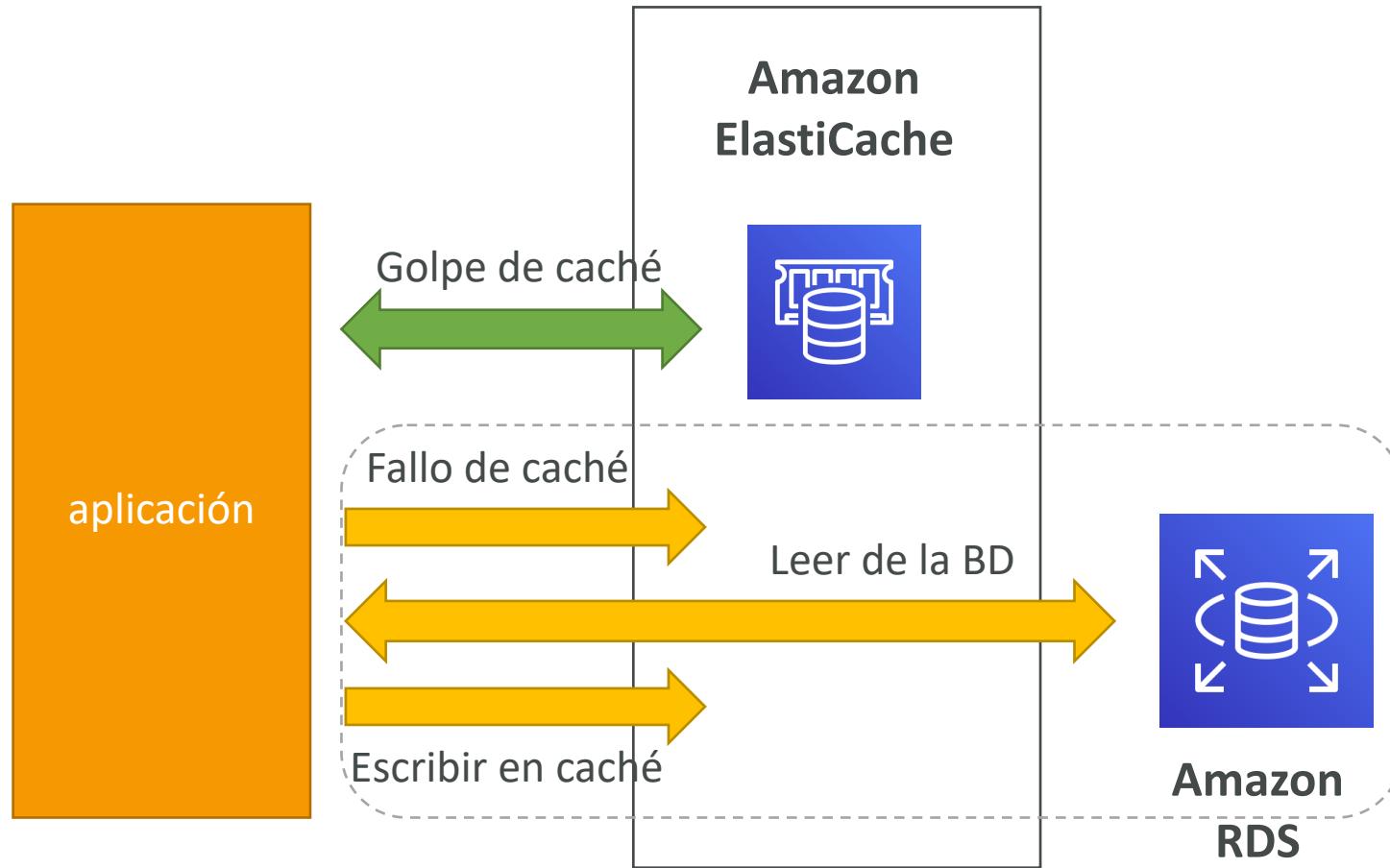
- Múltiples nodos para la partición de datos (sharding)
- **Sin alta disponibilidad (replicación)**
- **No es persistente**
- **No hay copia de seguridad ni restauración**
- Arquitectura multihilo



Consideraciones sobre la implementación del caché

- Lee más en: <https://aws.amazon.com/caching/implementation-considerations/>
- **¿Es seguro almacenar datos en caché?**
 - Los datos pueden quedar desfasados, y con el tiempo no ser coherentes
- **¿Es eficaz el almacenamiento en caché para esos datos?**
 - Datos que cambian lentamente = se necesitan con frecuencia pocas claves
 - Datos que cambian rápidamente = se necesita con frecuencia todo un gran espacio de claves
- **¿Están bien estructurados los datos para su almacenamiento en caché?**
- **¿Qué patrón de diseño de caché es el más adecuado?**

Lazy Loading / Cache-Aside / Lazy Population



- Ventajas
 - Sólo se almacenan en caché los datos solicitados (la caché no se llena de datos no utilizados)
 - Los fallos de los nodos no son fatales (sólo aumenta la latencia para calentar la caché)
- Contras
 - Penalización por fallo de caché que provoca 3 viajes de ida y vuelta, retraso notable para esa petición
 - Datos obsoletos: los datos pueden actualizarse en la base de datos y quedar obsoletos en la caché

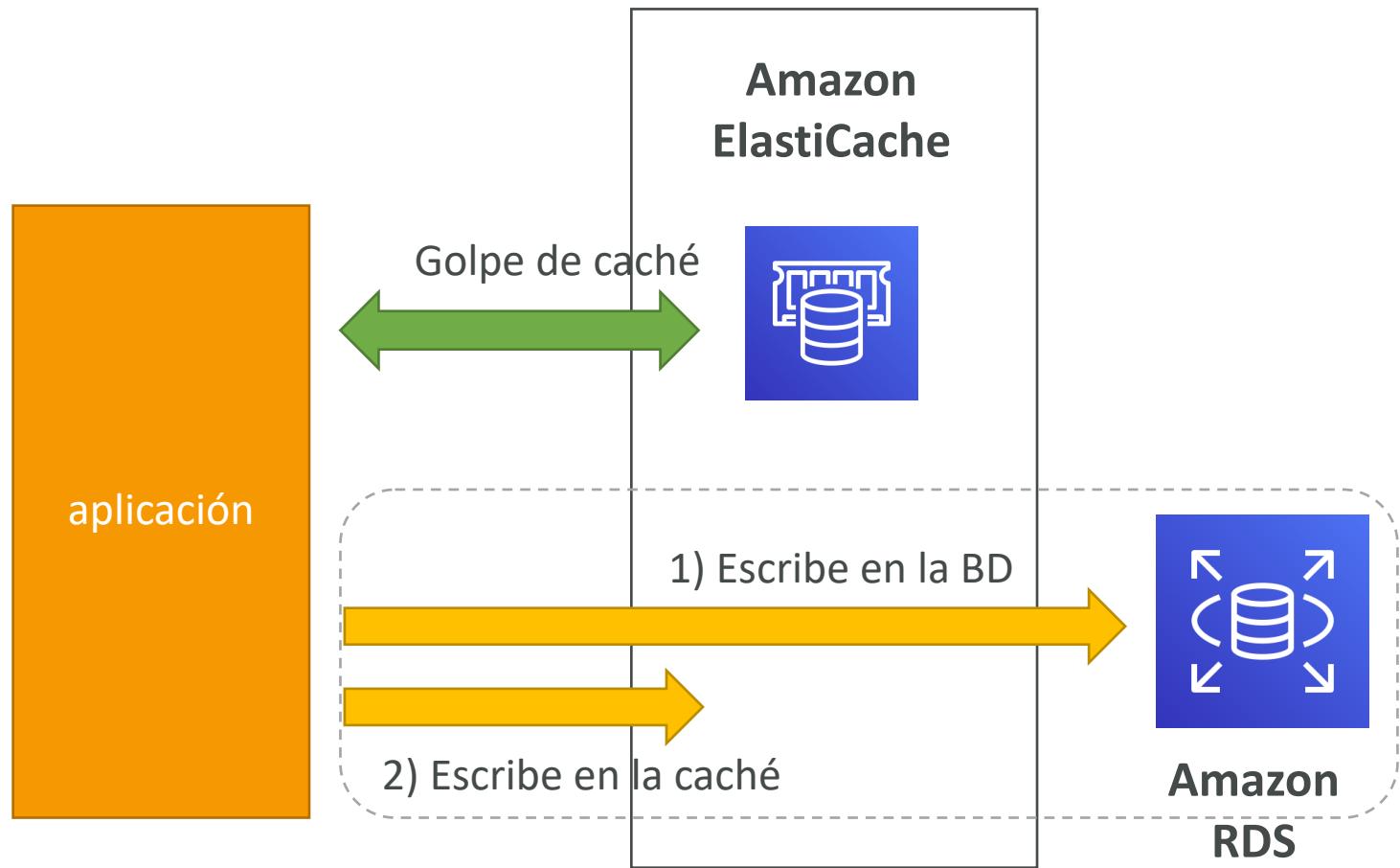
Lazy Loading / Cache-Aside / Lazy Population

Pseudocódigo en Python

```
1 # Python
2
3 def get_user(user_id):
4     # Check the cache
5     record = cache.get(user_id)
6
7     if record is None:
8         # Run a DB query
9         record = db.query("select * from users where id = ?", user_id)
10        # Populate the cache
11        cache.set(user_id, record)
12        return record
13    else:
14        return record
15
16 # App code
17 user = get_user(17)
```

Write Through (Escritura directa) -

Añadir o actualizar caché cuando se actualiza la base de datos



- Ventajas:
 - Los datos de la caché nunca caducan, **las lecturas son rápidas**
 - **Penalización de escritura** frente a penalización de lectura (cada escritura requiere 2 llamadas)
- Contras:
 - Datos perdidos hasta que se añaden / actualizan en la BD. La mitigación consiste en aplicar también la estrategia de carga perezosa.
 - Muchos de los datos nunca se leerán

Escritura Directa / Write-Through Pseudocódigo Python

```
1 # Python
2
3 def save_user(user_id, values):
4
5     # Save to DB
6
7     record = db.query("update users ... where id = ?", user_id, values)
8
9     # Push into cache
10
11    cache.set(user_id, record)
12
13    return record
14
15 # App code
16
17 user = save_user(17, {"name": "Nate Dogg"})
```

Invalidación de cachés y tiempo de vida (TTL)

- El desalojo de la caché puede producirse de tres formas:
 - Eliminas el elemento explícitamente de la caché
 - El elemento se expulsa porque la memoria está llena y no se ha utilizado recientemente (LRU)
 - Estableces un **tiempo de vida del elemento (o TTL)**
- Los TTL son útiles para cualquier tipo de datos:
 - Tablas de clasificación
 - Comentarios
 - Flujos (streams) de actividad
- El TTL puede variar desde unos segundos hasta horas o días
- Si se producen demasiados desalojos debido a la memoria, debes escalar hacia arriba o hacia abajo

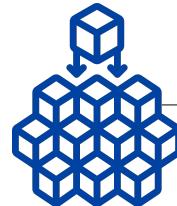
Palabras finales de “sabiduría”

- Lazy Loading / Cache-aside es fácil de implementar y funciona para muchas situaciones como base, especialmente en el lado de la lectura
- Write-through suele combinarse con Lazy Loading como objetivo para las consultas o cargas de trabajo que se benefician de esta optimización
- Establecer un TTL no suele ser una mala idea, excepto cuando utilizas Escritura directa / Write-through. Fíjalo en un valor razonable para tu aplicación.
- Almacena en caché sólo los datos que tengan sentido (perfiles de usuario, blogs, etc...)
- *Cita: Sólo hay dos cosas difíciles en Informática: invalidar la caché y nombrar las cosas.*

Amazon MemoryDB para Redis



- **Servicio de base de datos en memoria, duradero** y compatible con Redis
- **Rendimiento ultrarrápido con más de 160 millones de peticiones/segundo**
- Almacenamiento duradero de datos en memoria con logs transaccionales Multi-AZ
- Escala sin problemas de 10 GBs a 100 TBs de almacenamiento
- Casos de uso: aplicaciones web y móviles, juegos en línea, streaming multimedia, ...



Aplicaciones de microservicios
Web, móvil, comercio minorista, juegos, medios de comunicación y entretenimiento, banca, finanzas y mucho más ...



Route 53

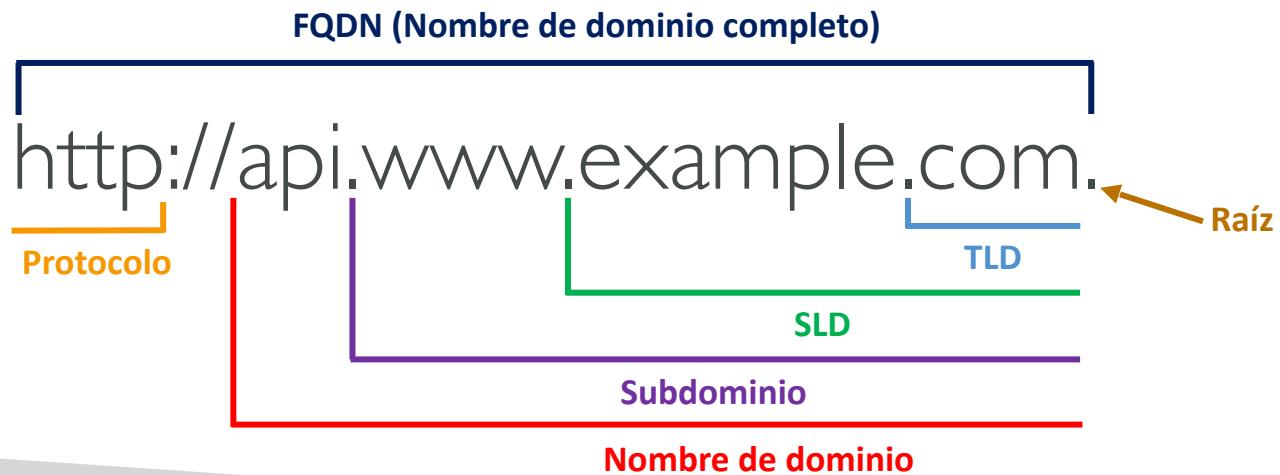
¿Qué es DNS?

- Sistema de nombres de dominio que traduce los nombres de host amigables con el ser humano en las direcciones IP de las máquinas
- www.google.com => 172.217.18.36
- El DNS es la columna vertebral de Internet
- El DNS utiliza una estructura jerárquica de nombres

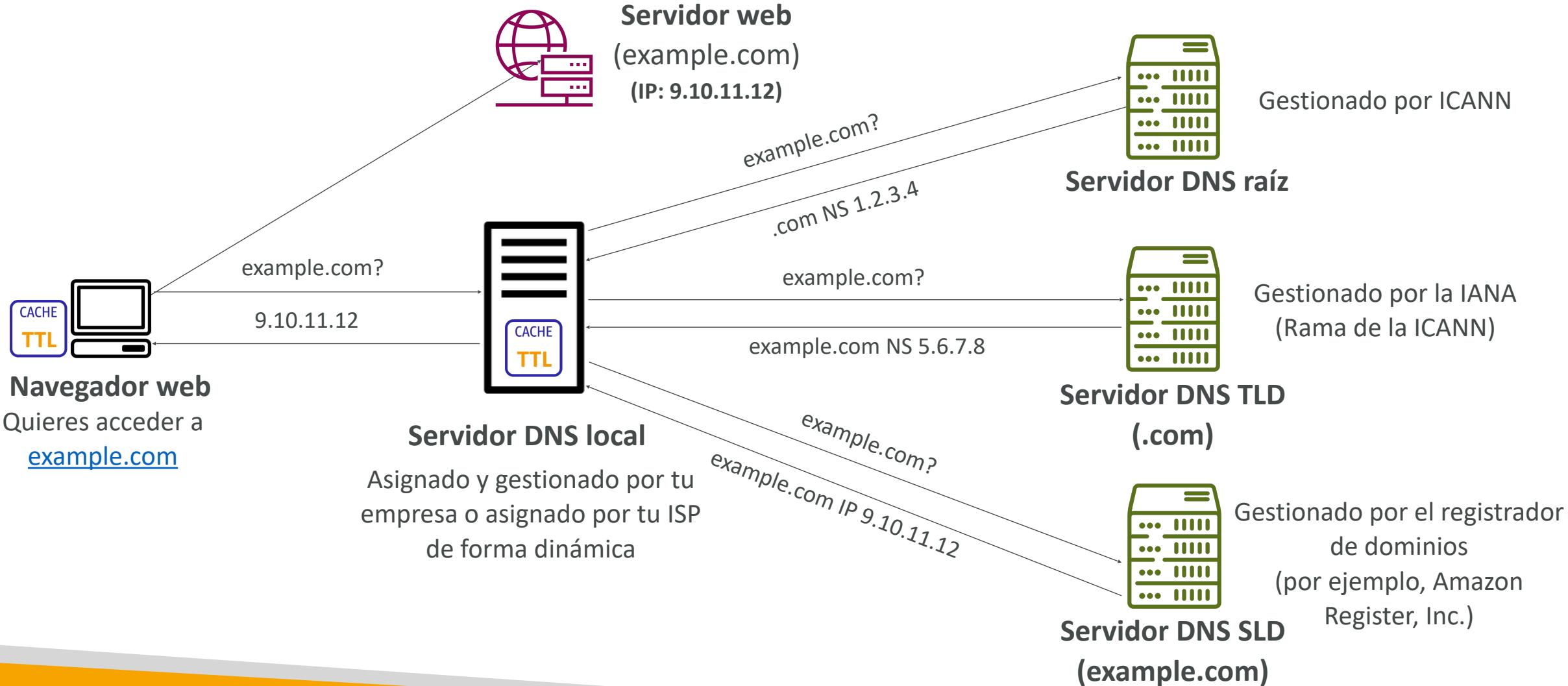
.com
example.com
www.example.com
api.example.com

Terminología de DNS

- **Registrador de dominios:** Amazon Route 53, GoDaddy, ...
- **Registros DNS:** A, AAAA, CNAME, NS, ...
- **Archivo de zona:** contiene registros DNS
- **Servidor de nombres:** resuelve las consultas DNS (autorizadas o no autorizadas)
- **Dominio de primer nivel (TLD):** .com, .us, .in, .gov, .org, ...
- **Dominio de segundo nivel (SLD):** amazon.com, google.com, ...

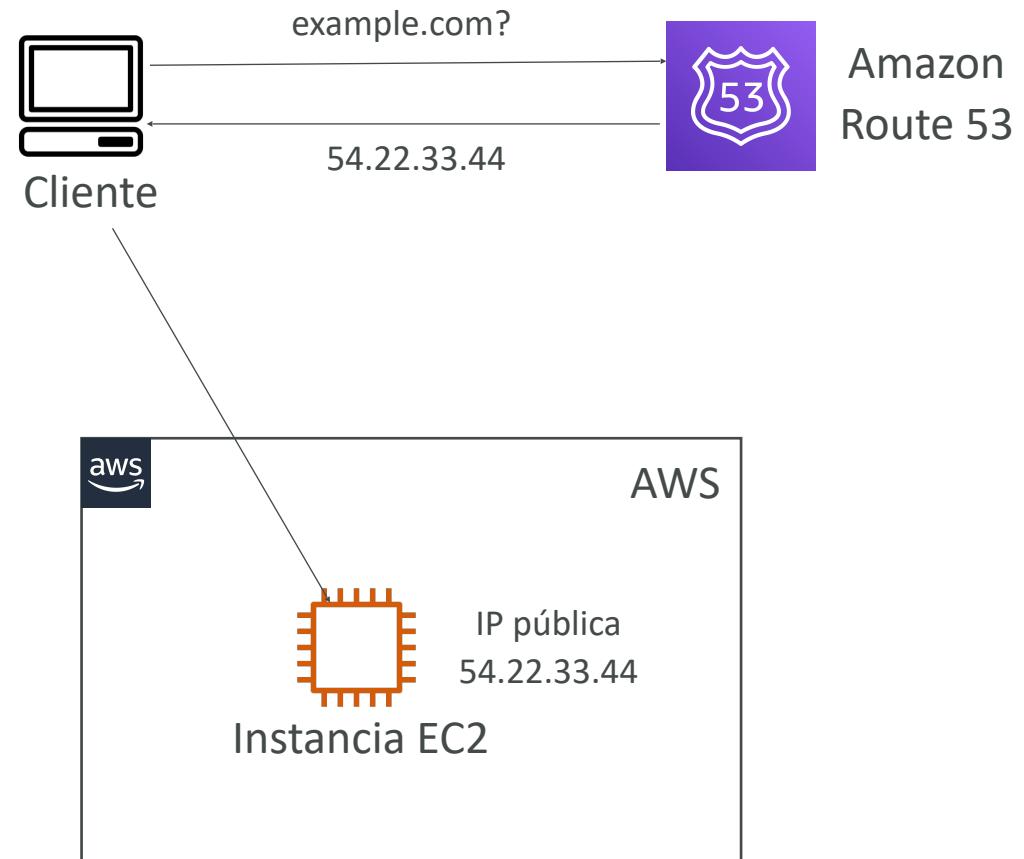


Cómo funciona el DNS



Amazon Route 53

- Un DNS altamente disponible, escalable, totalmente gestionado y autoritativo
 - Autoritario = el cliente (tú) puedes actualizar los registros DNS
- Route 53 también es un registrador de dominios
- Posibilidad de comprobar la salud de tus recursos
- El único servicio de AWS que ofrece un SLA de disponibilidad del 100%
- ¿Por qué Route 53? 53 es una referencia al puerto DNS tradicional

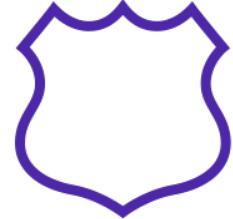


Route 53 - Registros

- Cómo quieras dirigir el tráfico de un dominio
- Cada registro contiene
 - **Nombre del dominio/subdominio** - por ejemplo, ejemplo.com
 - **Tipo de registro** - por ejemplo, A o AAAA
 - **Valor** - por ejemplo, 12.34.56.78
 - **Política de enrutamiento** - cómo responde Route 53 a las consultas
 - **TTL** - cantidad de tiempo que el registro se almacena en caché en los Resolvers DNS
- Route 53 soporta los siguientes tipos de registros DNS:
 - (obligatorio) A / AAAA / CNAME / NS
 - (avanzado) CAA / DS / MX / NAPTR / PTR / SOA / TXT / SPF / SRV

Route 53 - Tipos de registro

- **A** - asigna un nombre de host a IPv4
- **AAAA** - asigna un nombre de host a IPv6
- **CNAME** - asigna un nombre de host a otro nombre de host
 - El objetivo es un nombre de dominio que debe tener un registro A o AAAA
 - No puedes crear un registro CNAME para el nodo superior de un espacio de nombres DNS (Zona Apex)
 - Ejemplo: no puedes crear para example.com, pero sí para www.example.com
- **NS** - Servidores de nombres para la Zona Alojada
 - Controla cómo se enruta el tráfico de un dominio

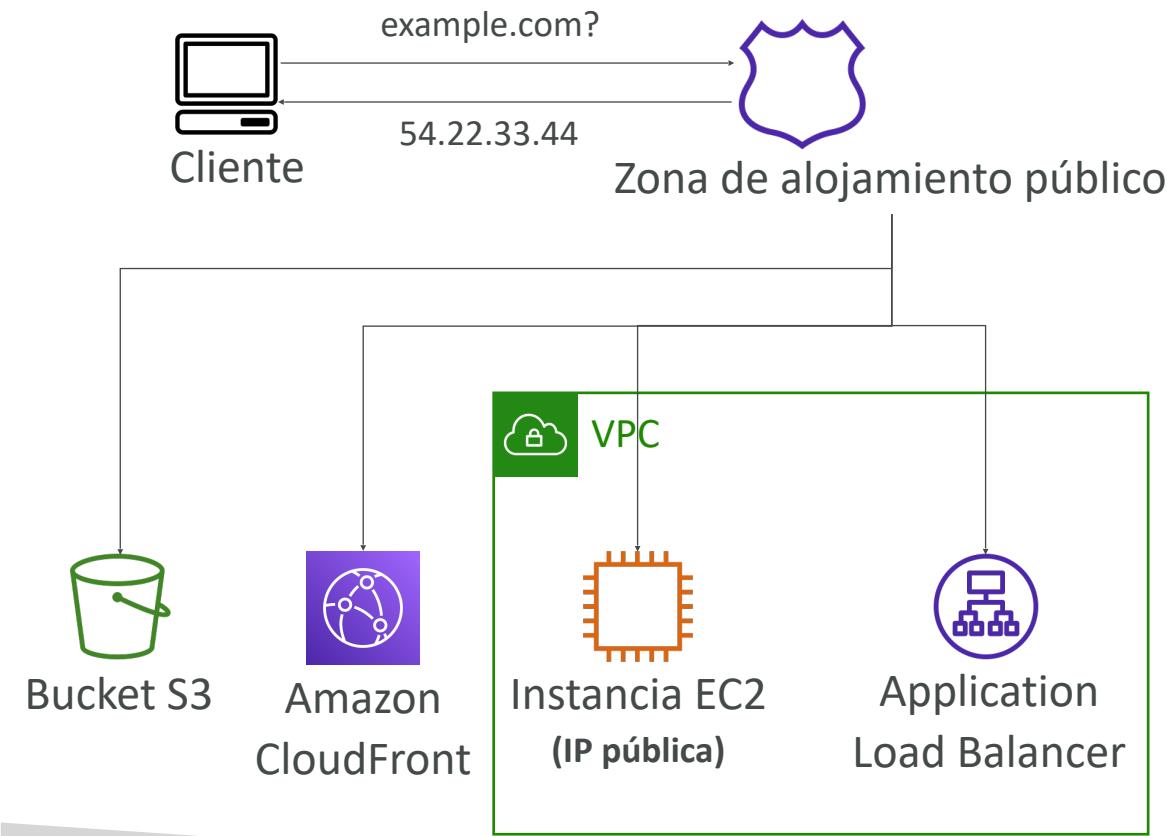


Route 53 - Zonas de alojamiento

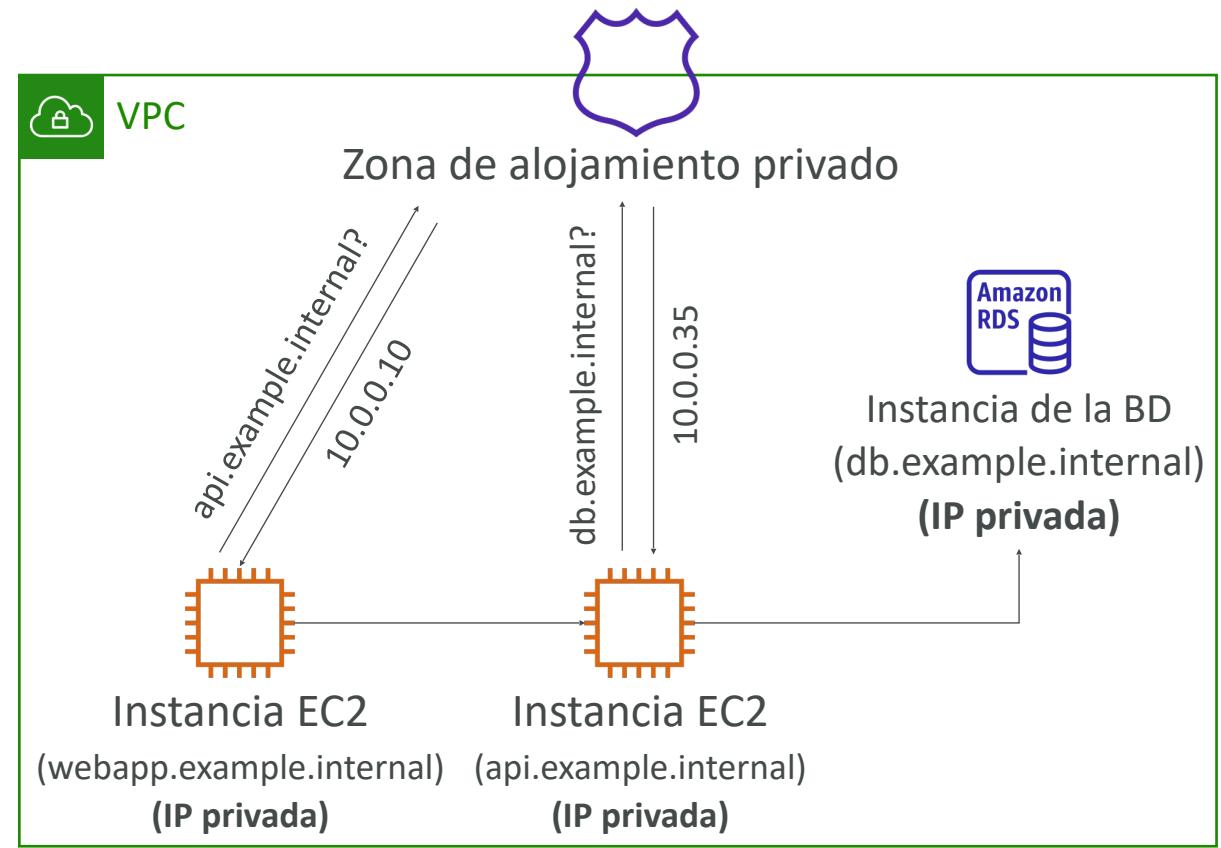
- Un contenedor para los registros que definen cómo dirigir el tráfico a un dominio y sus subdominios
- **Zonas de alojamiento público:** contiene registros que especifican cómo enrutar el tráfico en Internet (nombres de dominio público)
application1.mypublicdomain.com
- **Zonas de alojamiento privadas:** contienen registros que especifican cómo enrutar el tráfico dentro de una o más VPC (nombres de dominio privados)
application1.company.internal
- Pagas 0,50\$ al mes por zona alojada

Route 53 - Zonas de alojamiento públicas frente a privadas

Zona de alojamiento público

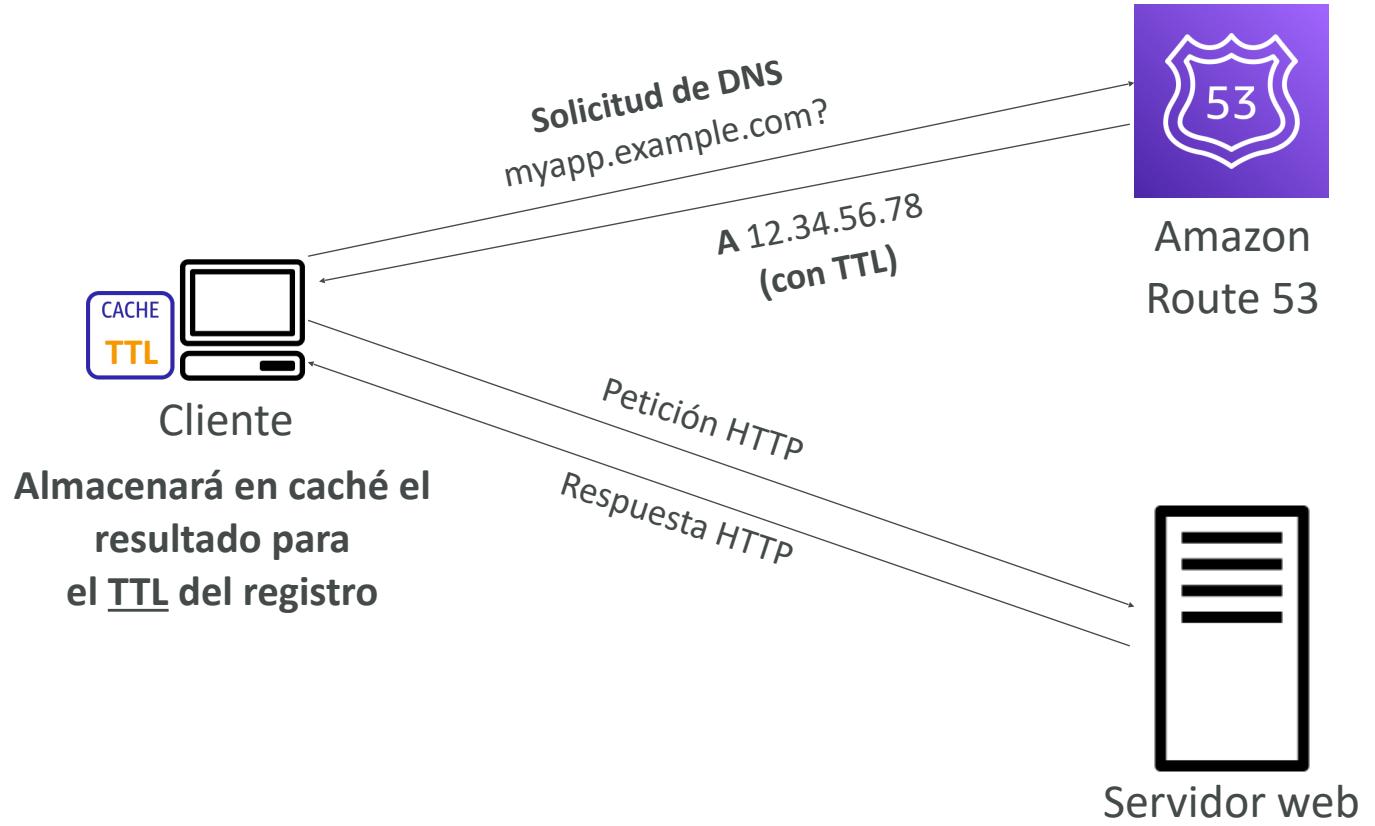


Zona de alojamiento privado



Route 53 - Registros TTL (Tiempo de vida)

- **TTL alto - por ejemplo, 24 horas**
 - Menos tráfico en Route 53
 - Registros posiblemente obsoletos
- **TTL bajo - por ejemplo, 60 seg.**
 - Más tráfico en la Route 53 (\$\$)
 - Los registros están desfasados durante menos tiempo
 - Facilidad para cambiar los registros
- **Excepto los registros de Alias, el TTL es obligatorio para cada registro DNS**



CNAME vs Alias

- Los recursos de AWS (Load Balancer, CloudFront...) exponen un nombre de host de AWS:
 - **IbI-I234.us-east-2.elb.amazonaws.com** y quieres **myapp.midominio.com**
- CNAME:
 - Apunta un nombre de host a cualquier otro nombre de host. (app.midominio.com => blabla.algo.com)
 - **SÓLO PARA DOMINIOS NO ROOT (algo.midominio.com)**
- Alias:
 - Apunta un nombre de host a un recurso de AWS (app.mydomain.com => blabla.amazonaws.com)
 - **Funciona para DOMINIO RAÍZ y DOMINIO NO RAÍZ (mydomain.com)**
 - Gratis
 - Comprobación de salud nativa

Route 53 - Registros con Alias

- Asigna un nombre de host a un recurso de AWS
- Una extensión de la funcionalidad del DNS
- Reconoce automáticamente los cambios en las direcciones IP del recurso
- A diferencia de CNAME, puede utilizarse para el nodo superior de un espacio de nombres DNS (Zona Apex), por ejemplo: example.com
- El Registro Alias es siempre del tipo A/AAAA para los recursos AWS (IPv4 / IPv6)
- No puedes establecer el TTL



Route 53 – Objetivos Registros con Alias

- Elastic Load Balancers
- Distribuciones CloudFront
- API Gateway
- Entornos Elastic Beanstalk
- Sitios web S3
- Endpoints de interfaz VPC
- Global Accelerator
- Registro Route 53 en la misma Zona alojada
- **No puedes establecer un registro ALIAS para un nombre DNS de EC2**



Elastic
Load Balancer



Amazon
CloudFront



Amazon
API Gateway



Elastic Beanstalk



Sitios web S3



Interfaz VPC
Endpoints



Global Accelerator



Registro de la Route 53
(misma Zona de alojamiento)

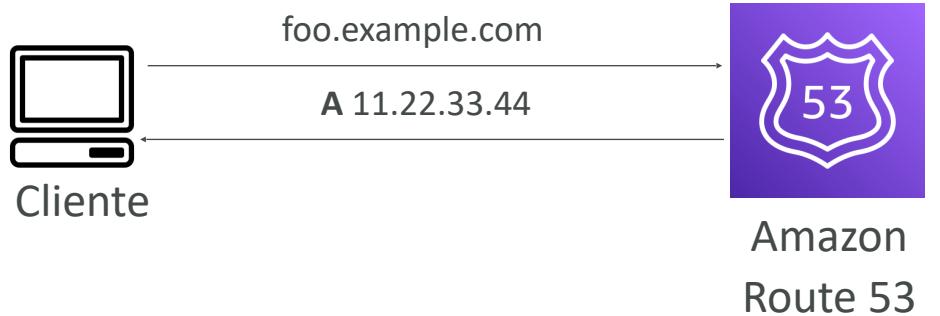
Route 53 - Políticas de enrutamiento

- Definir cómo responde Route 53 a las consultas DNS
- No te confundas con la palabra "*Enrutamiento*"
 - No es lo mismo que el enrutamiento del Load Balancer, que enruta el tráfico
 - El DNS no enruta ningún tráfico, sólo responde a las consultas del DNS
- Route 53 soporta las siguientes políticas de enrutamiento
 - Simple
 - Ponderada
 - Comutación por error
 - Basada en la latencia
 - Geolocalización
 - Respuesta multivalente
 - Geoproximidad (utilizando la función de flujo de tráfico de Route 53)

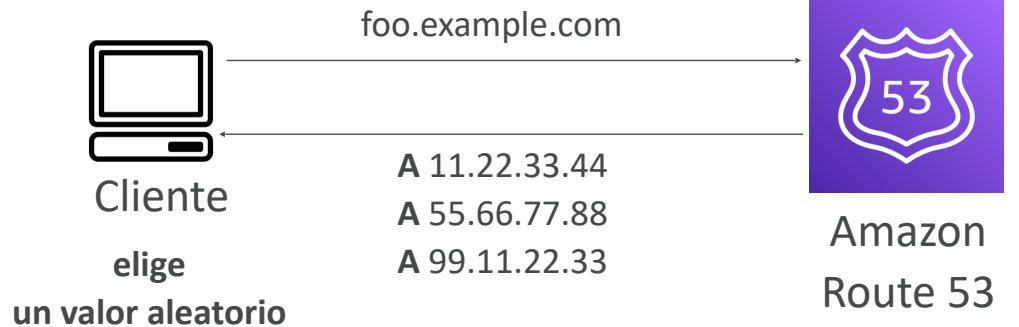
Políticas de enrutamiento - Simple

- Normalmente, dirige el tráfico a un solo recurso
- Puede especificar varios valores en el mismo registro
- **Si se devuelven varios valores, el cliente elige uno al azar**
- Cuando se habilita el Alias, sólo se puede especificar un recurso de AWS
- No se puede asociar a los controles de salud

Valor único



Valor múltiple

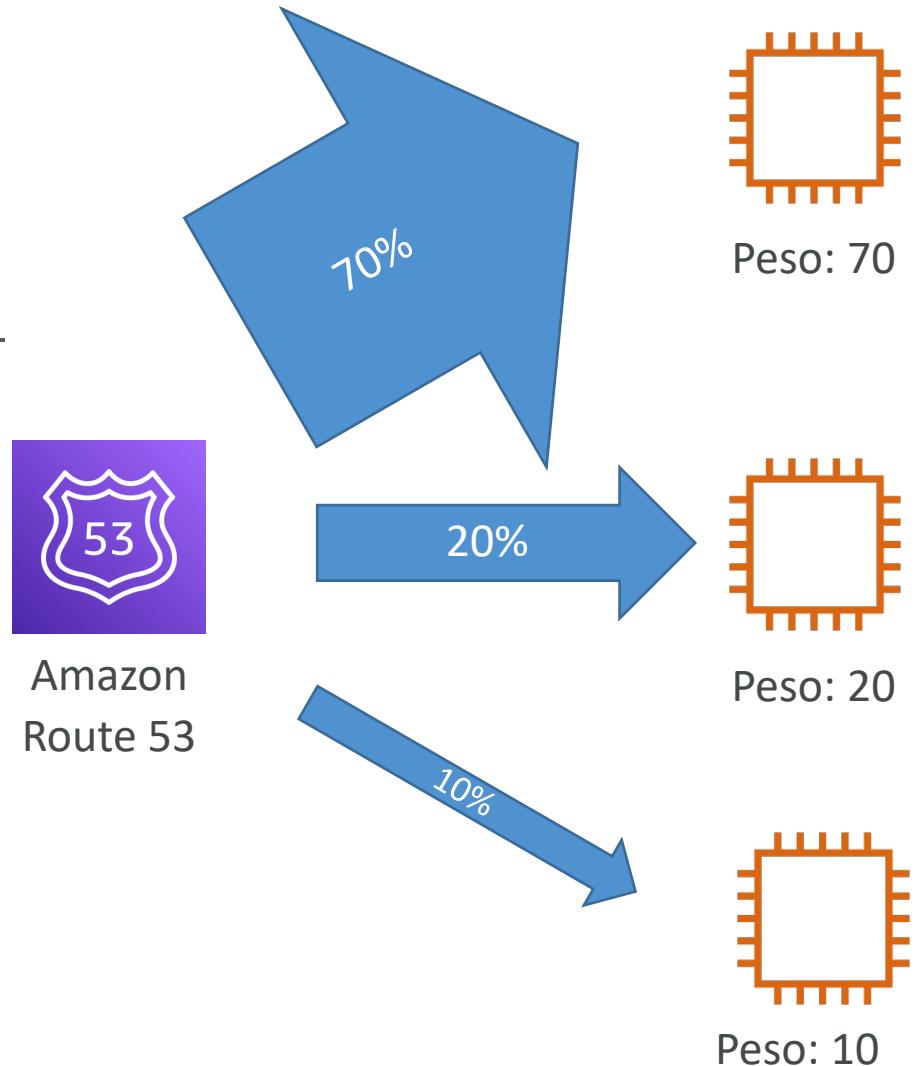


Políticas de enrutamiento - Ponderadas

- Controla el % de las solicitudes que van a cada recurso específico
- Asigna a cada registro un peso relativo:

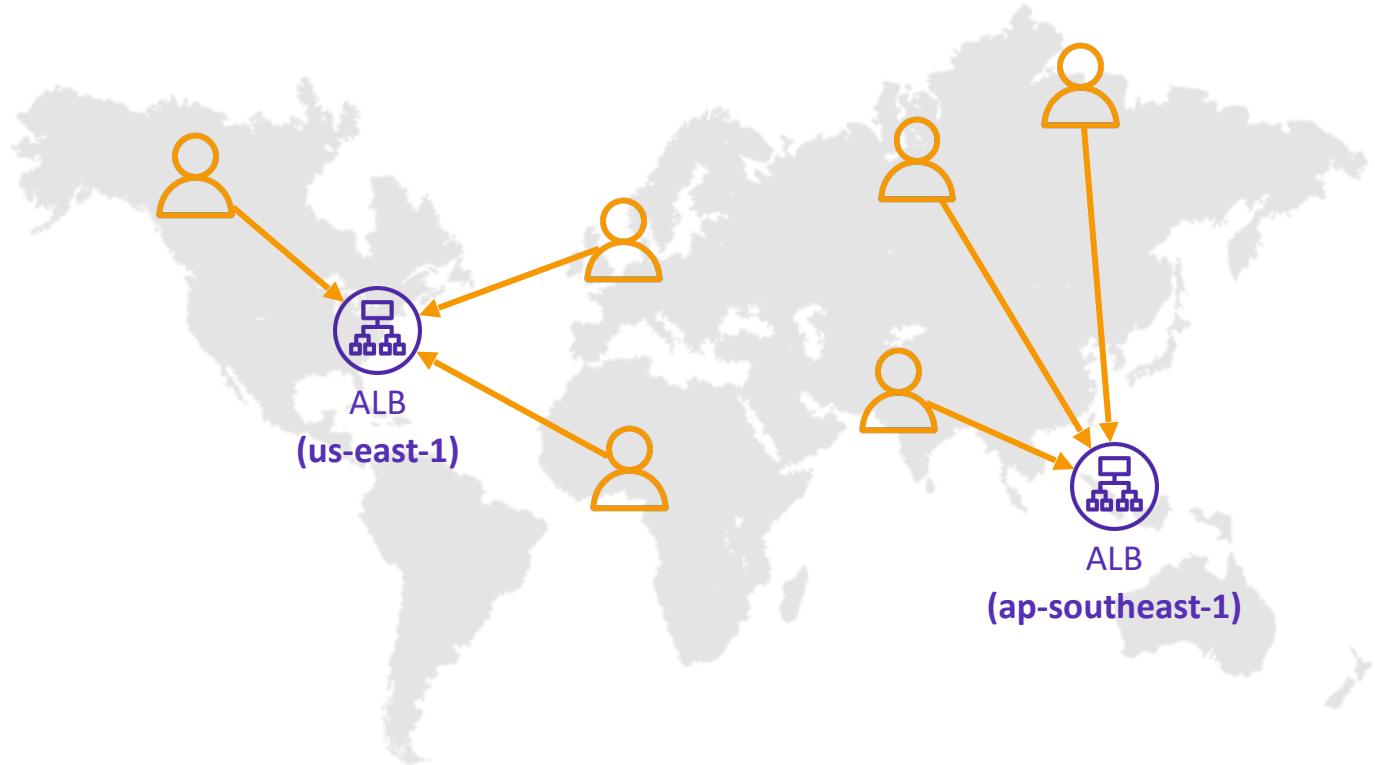
$$\bullet \text{ traffic } (\%) = \frac{\text{Weight for a specific record}}{\text{Sum of all the weights for all records}}$$

- No es necesario que los pesos sumen 100
- Los registros DNS deben tener el mismo nombre y tipo
- Pueden asociarse a las comprobaciones de salud
- Casos de uso: equilibrar la carga entre regiones, probar nuevas versiones de aplicaciones...
- **Asigna un peso de 0 a un registro para dejar de enviar tráfico a un recurso**
- **Si todos los registros tienen un peso de 0, se devolverán todos los registros por igual**



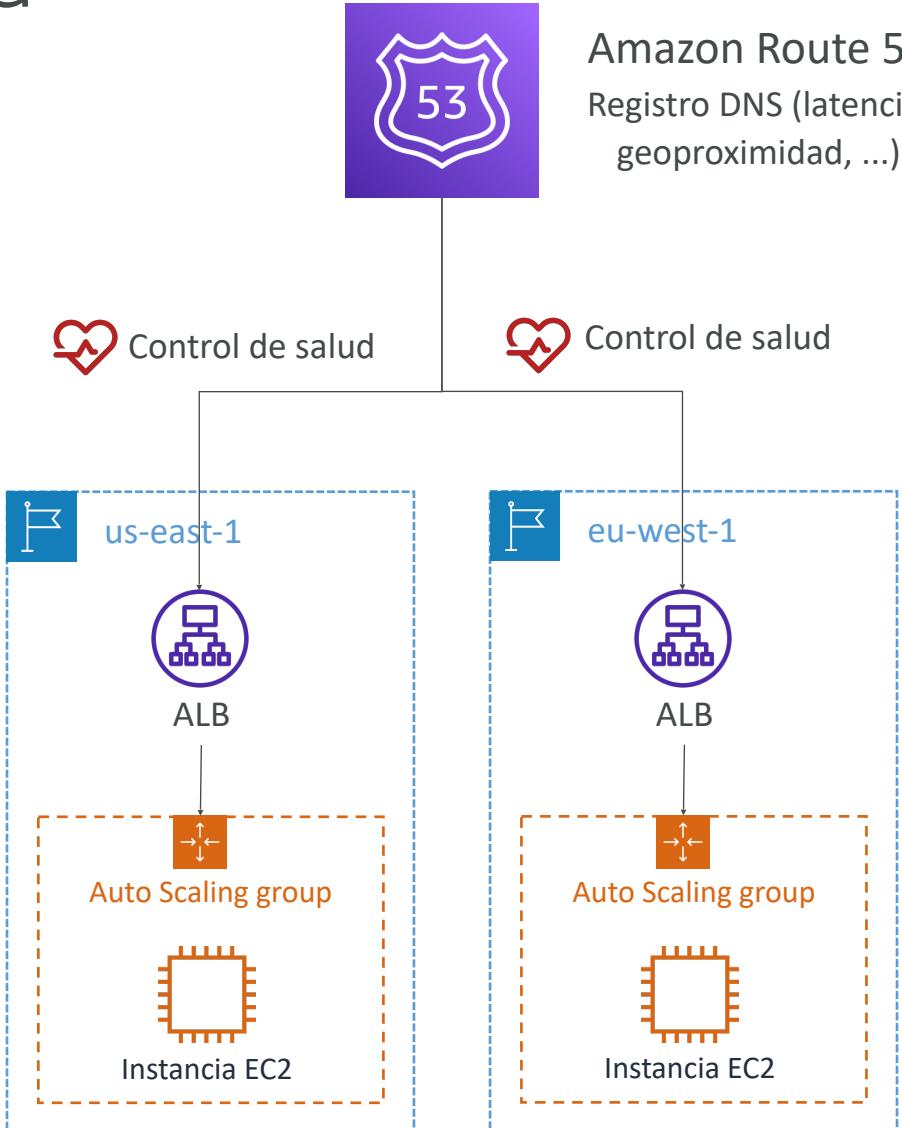
Políticas de enrutamiento - basadas en la latencia

- Redirigir al recurso que tenga la menor latencia cerca de nosotros
- Muy útil cuando la latencia para los usuarios es una prioridad
- **La latencia se basa en el tráfico entre los usuarios y las regiones de AWS**
- Los usuarios de Alemania pueden ser dirigidos a EEUU (si esa es la latencia más baja)
- Se puede asociar a los controles de salud (tiene capacidad de commutación por error)



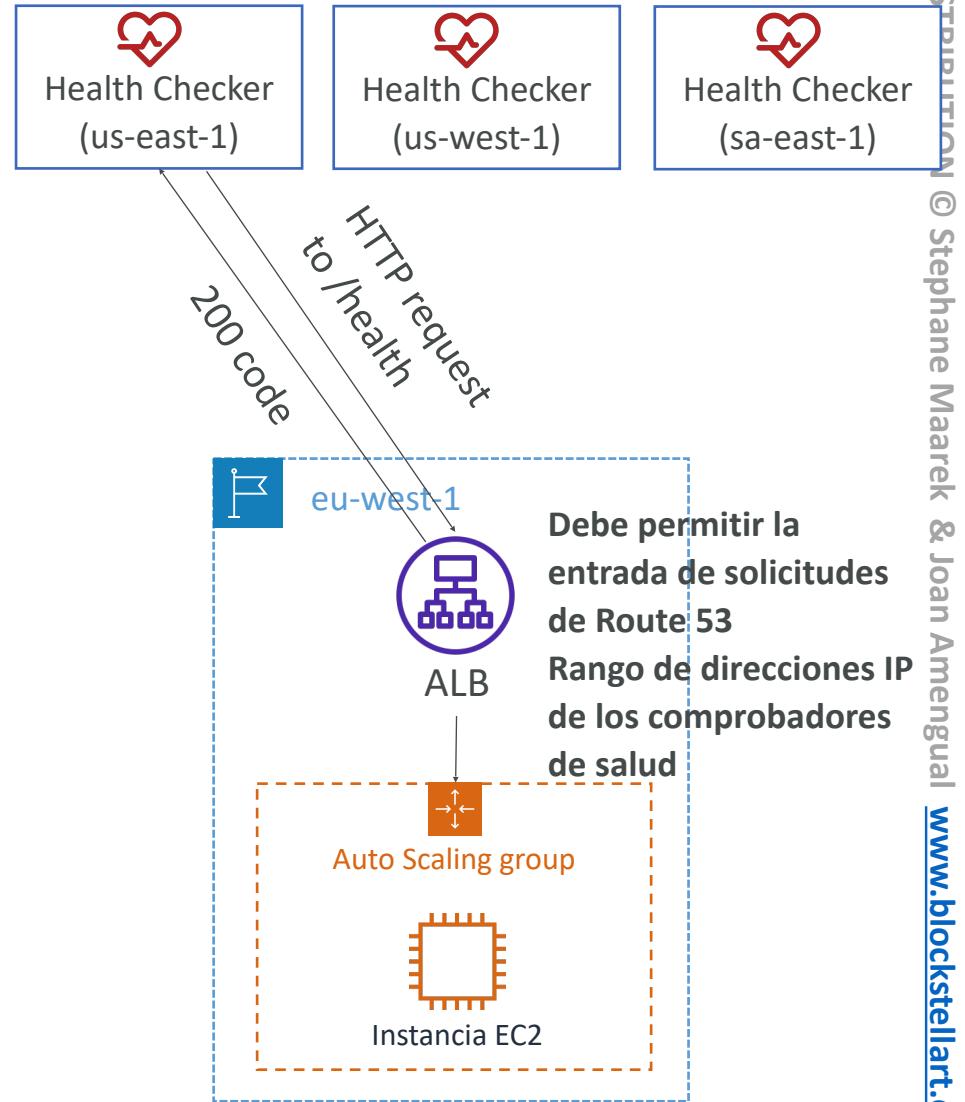
Route 53 - Controles de salud

- Las comprobaciones de salud HTTP son sólo para los **recursos públicos**
- Comprobación de la salud => Conmutación por error de DNS automatizada:
 1. Comprobaciones de salud que supervisan un endpoint (aplicación, servidor, otro recurso de AWS)
 2. Controles de salud que controlan otros controles de salud (Controles de salud calculados)
 3. Controles de salud que supervisan las alarmas de CloudWatch (¡control total!) - por ejemplo, alarmas en RDS, métricas personalizadas, ... (útil para recursos privados)
- Los Controles de Salud se integran con las métricas del CloudWatch



Comprobaciones de salud - Monitorizar un endpoint

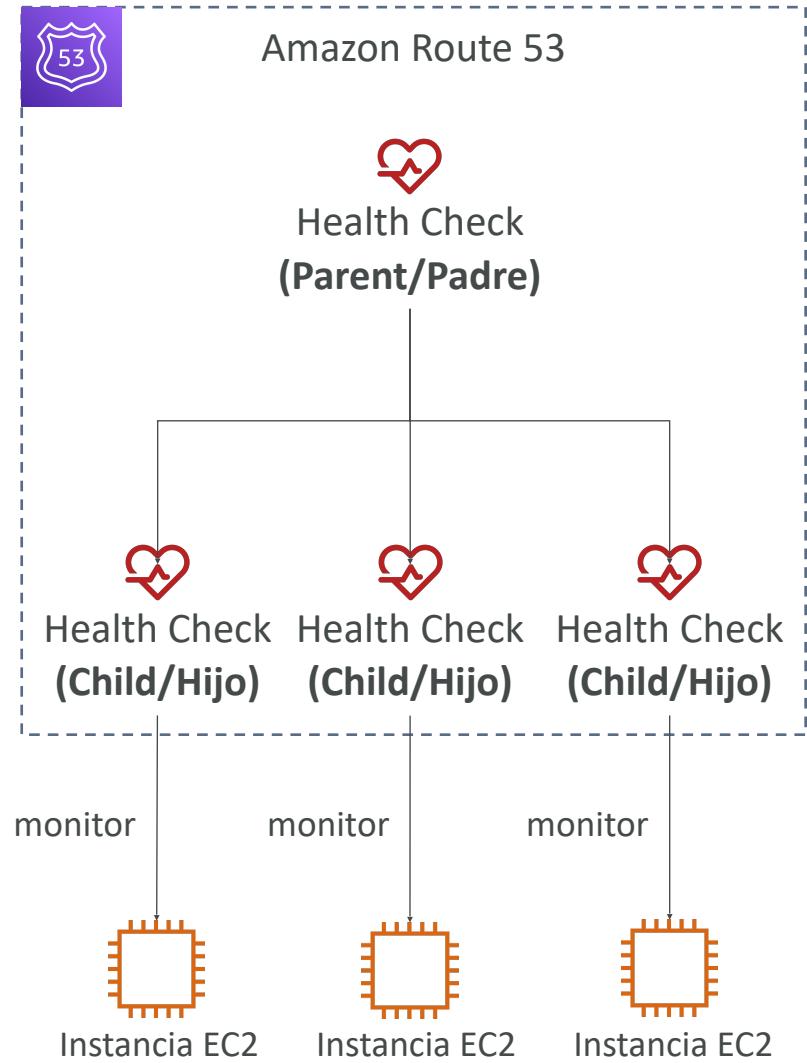
- **Unos 15 verificadores de salud globales comprobarán la salud del endpoint**
 - Umbral de salud/no salud - 3 (por defecto)
 - Intervalo - 30 segundos (se puede ajustar a 10 segundos - mayor coste)
 - Protocolo soportado: HTTP, HTTPS y TCP
 - Si > 18% de los comprobadores de salud informan de que el endpoint está sano, Route 53 lo considera **sano**. En caso contrario, se considera **no saludable**.
 - Posibilidad de elegir qué ubicaciones quieras que utilice Route 53
- Las comprobaciones de salud sólo pasan cuando el endpoint responde con los códigos de estado 2xx y 3xx
- Las comprobaciones de salud pueden configurarse para que pasen/no pasen en función del texto de los primeros **5120** bytes de la respuesta
- Configura tu router/firewall para permitir las solicitudes entrantes de los Health Checkers de Route 53



<https://ip-ranges.amazonaws.com/ip-ranges.json>

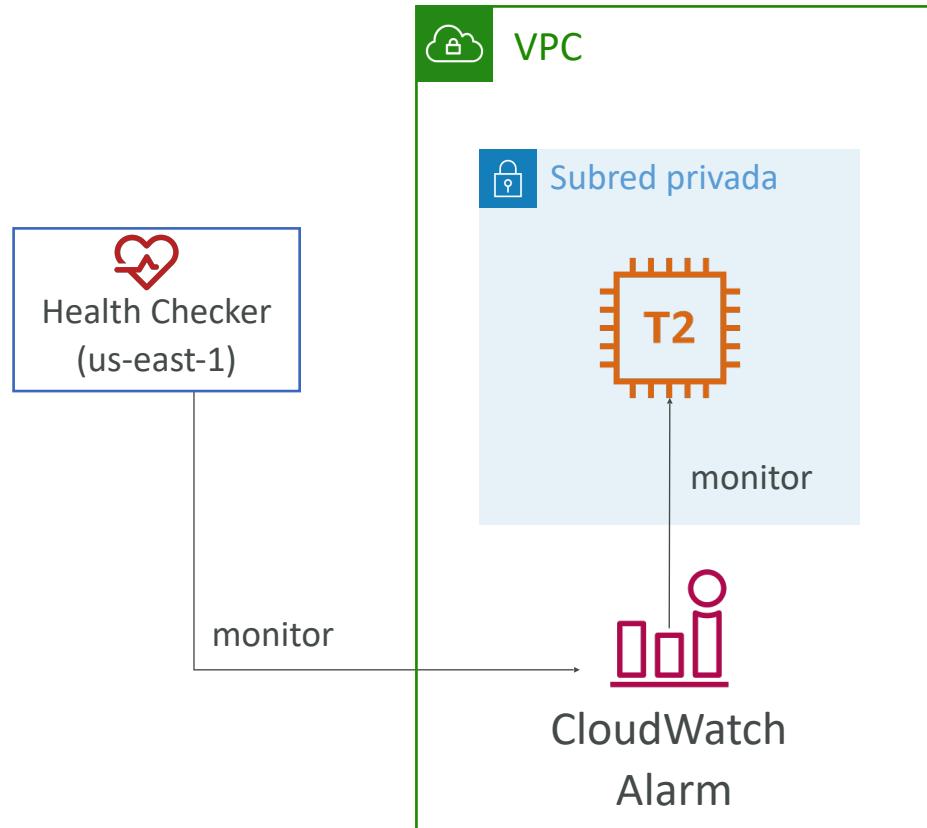
Route 53 - Controles de salud calculados

- Combinar los resultados de varios chequeos de salud en un solo chequeo de salud
- Puedes utilizar **OR, AND o NOT**
- Puedes controlar hasta 256 chequeos médicos de los “hijos” (child)
- Especifica cuántas de las comprobaciones de salud deben pasar para que “el padre” (parent) pase
- Uso: realiza el mantenimiento de tu sitio web sin que fallen todas las comprobaciones de salud

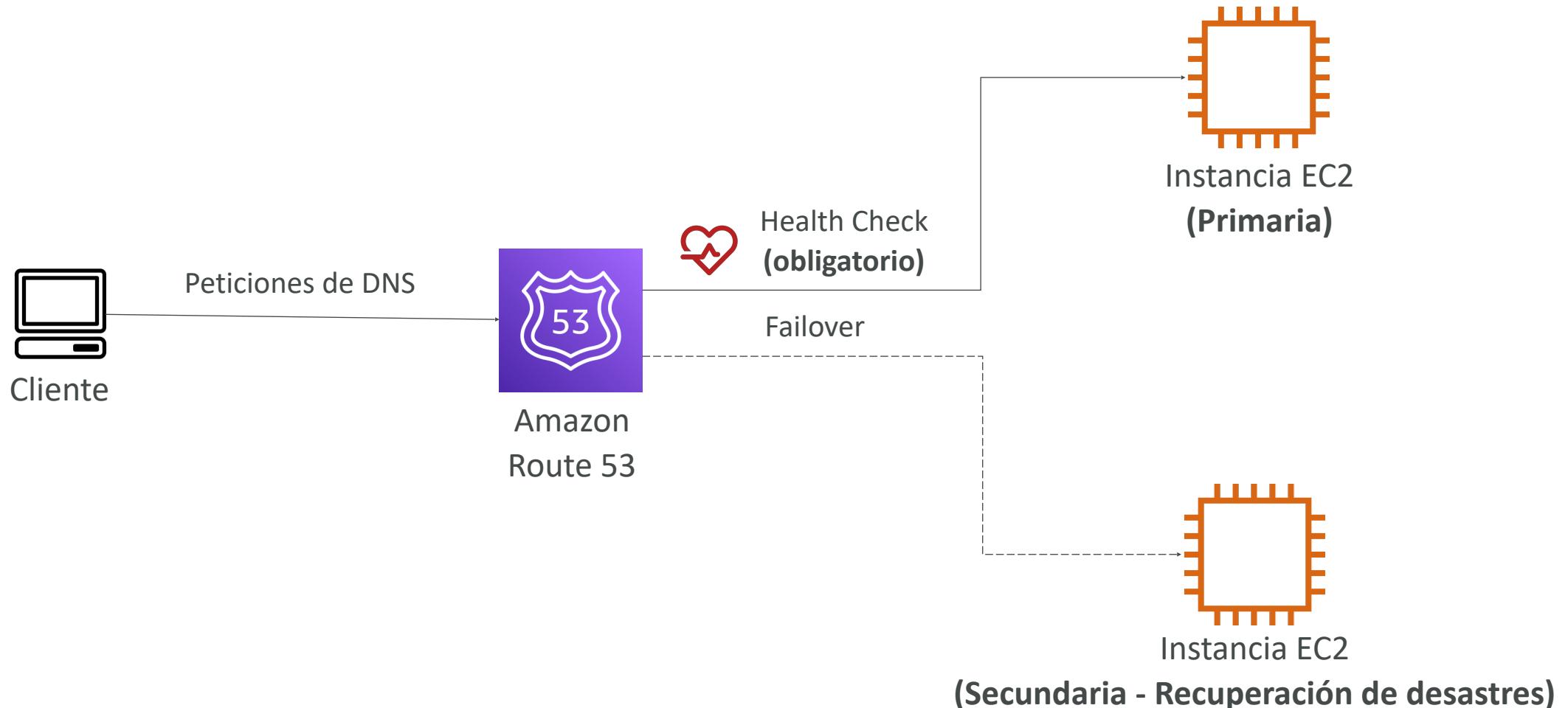


Controles de salud - Zonas de alojamiento privadas

- Los comprobadores de salud (Health Checker) de Route 53 están fuera de la VPC
- No pueden acceder a endpoints **privados** (VPC privada o recurso local)
- Puedes crear una **métrica de CloudWatch** y asociar una **alarma de CloudWatch**, y luego crear un chequeo de salud que compruebe la propia alarma

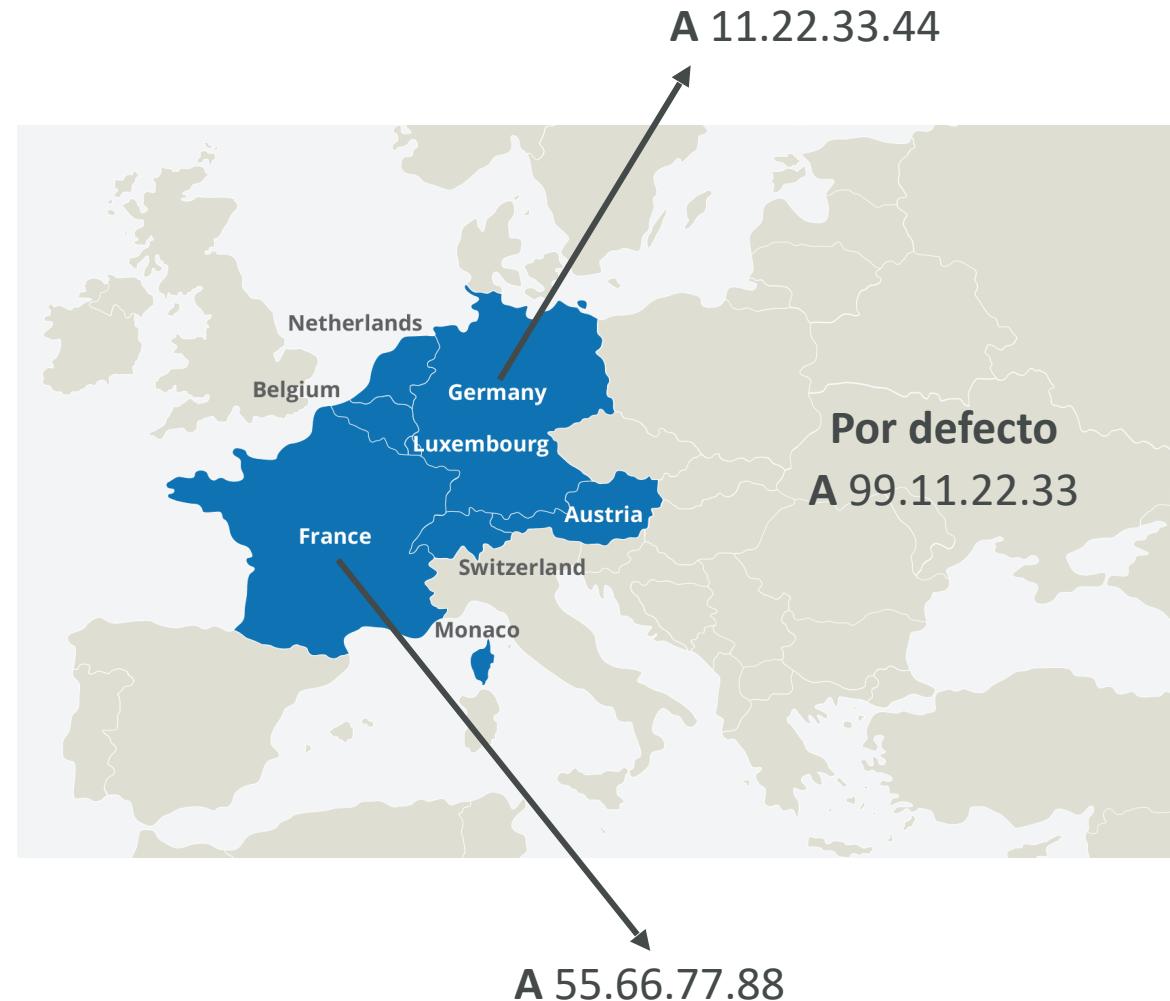


Políticas de enrutamiento - Comutación por error (activo-pasivo)



Políticas de enrutamiento - Geolocalización

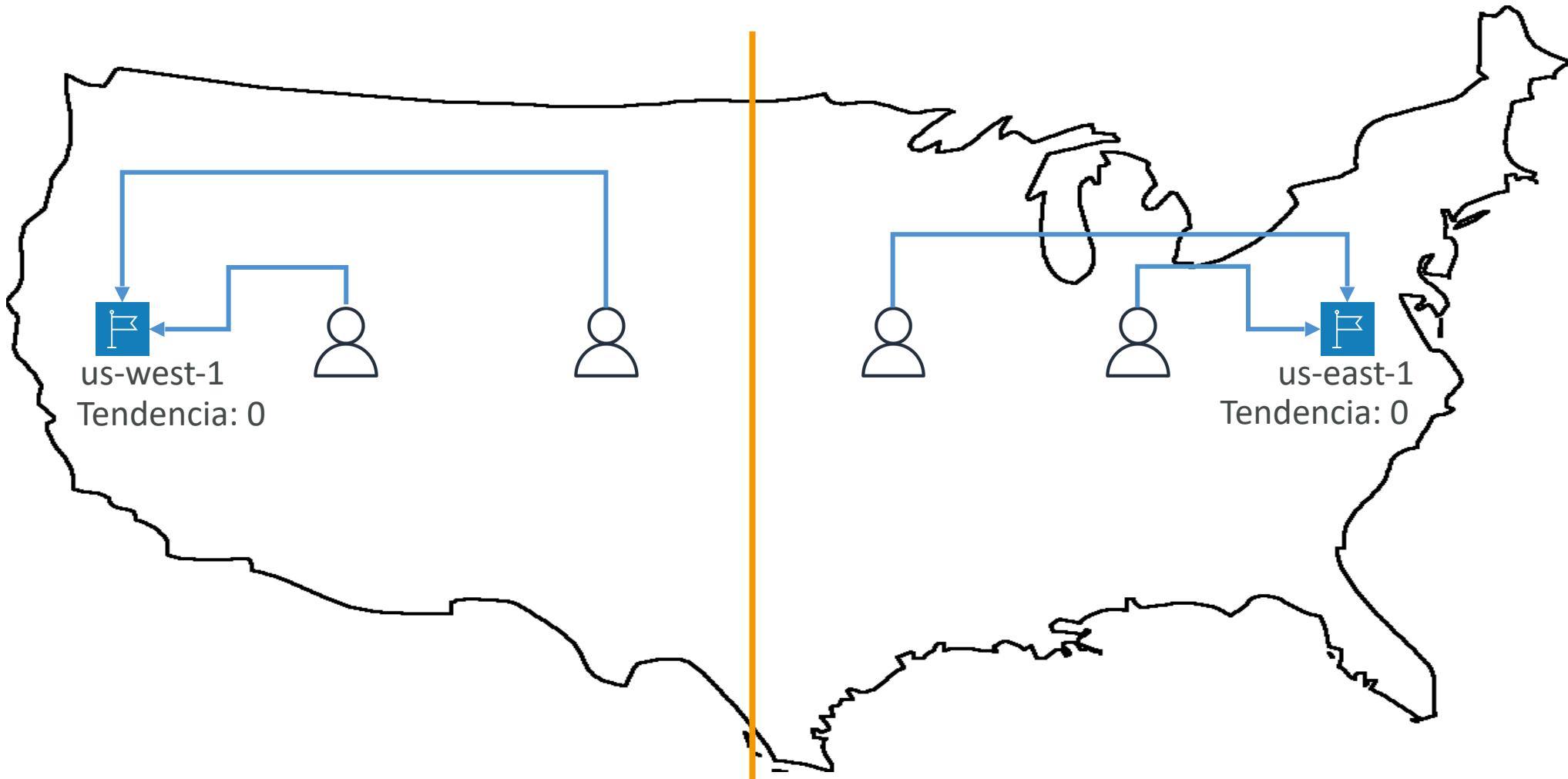
- ¡Diferente al basado en la latencia!
- **Este enrutamiento se basa en la ubicación del usuario**
- Especifica la ubicación por continente, país o estado de EE.UU. (si hay solapamiento, se selecciona la ubicación más precisa)
- Debe crear un registro "**por defecto**" (en caso de que no haya coincidencia en la ubicación)
- Casos de uso: localización de sitios web, restringir la distribución de contenidos, equilibrar la carga, ...
- Puede asociarse a los controles de salud



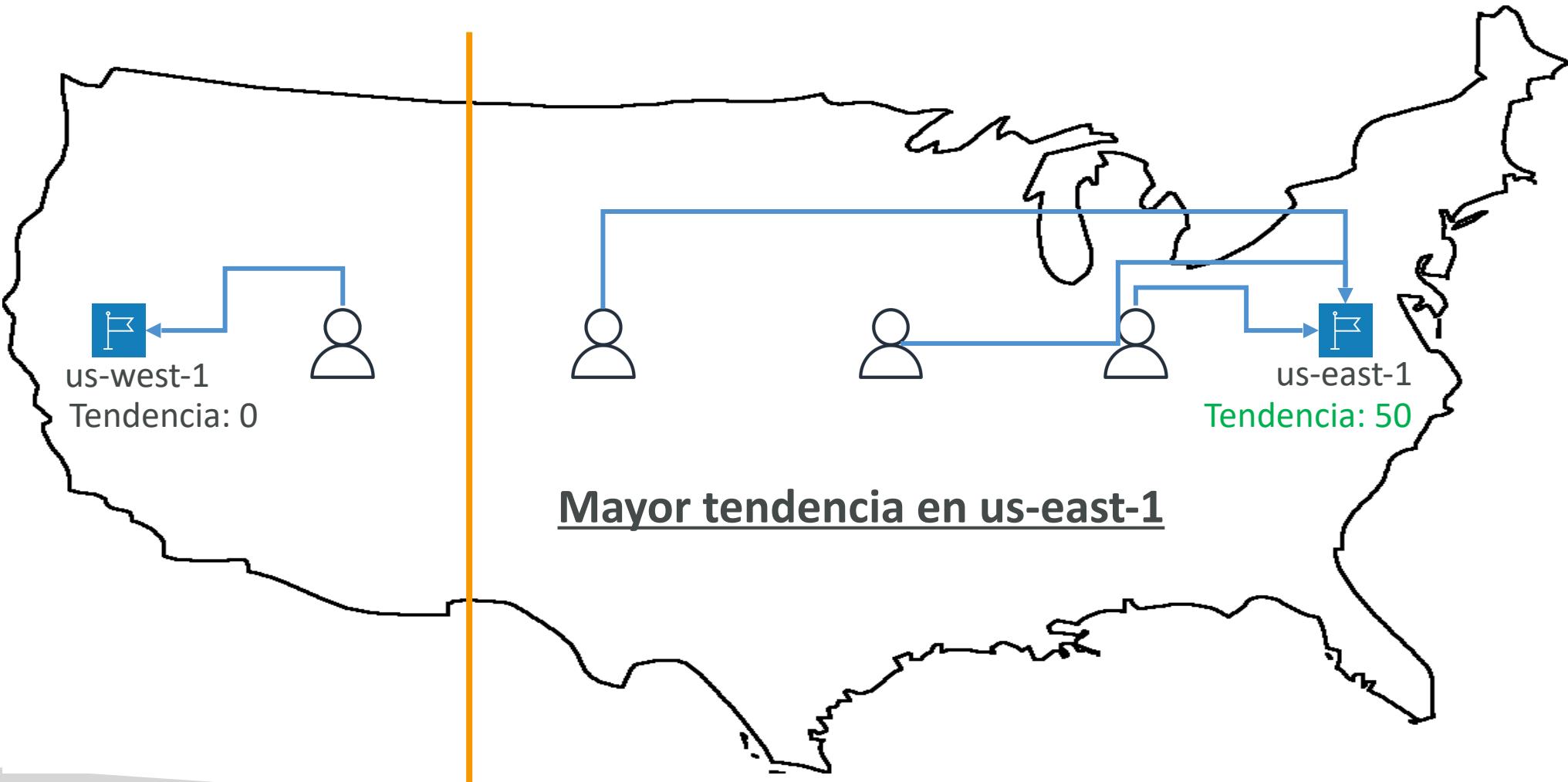
Políticas de enrutamiento - Geoproximidad

- Dirige el tráfico a tus recursos en función de la ubicación geográfica de los usuarios y los recursos
- Posibilidad de desplazar más tráfico a los recursos en función del sesgo definido
- Para cambiar el tamaño de la región geográfica, especifica los valores del sesgo:
 - Para ampliar (1 a 99) - más tráfico hacia el recurso
 - Para reducir (-1 a -99) - menos tráfico hacia el recurso
- Los recursos pueden ser
 - Recursos AWS (especifica la región AWS)
 - Recursos no AWS (especifica latitud y longitud)
- Debes utilizar el flujo de tráfico de Route 53 para utilizar esta función

Políticas de enrutamiento - Geoproximidad

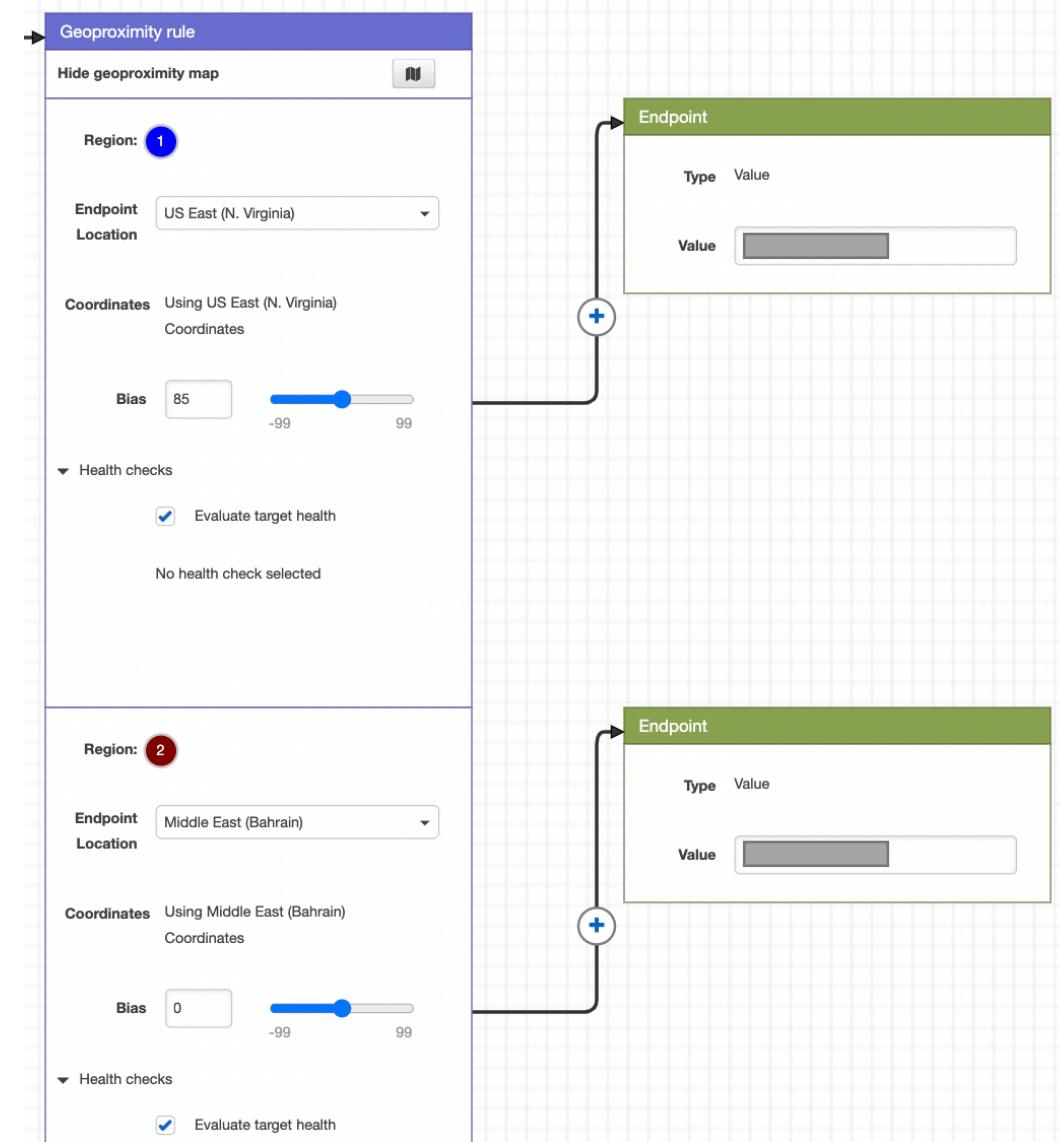


Políticas de enrutamiento - Geoproximidad



Route 53 - Flujo de tráfico

- Simplifica el proceso de creación y mantenimiento de registros en configuraciones grandes y complejas
- Editor visual para gestionar árboles de decisión de encaminamiento complejos
- Las configuraciones se pueden guardar como **Política de Flujo de Tráfico**
 - Puede aplicarse a diferentes Zonas de alojamiento Route 53 (diferentes nombres de dominio)
 - Soporta la creación de versiones



Políticas de enrutamiento - Multivalores

- Utilízalo cuando dirijas el tráfico a múltiples recursos
- Route 53 devuelve múltiples valores/recursos
- Puede asociarse a comprobaciones de salud (devuelve sólo los valores de los recursos sanos)
- Se devuelven hasta 8 registros sanos por cada consulta Multi-Value
- **El Multi-Valor no sustituye a tener un ELB**

Name	Type	Value	TTL	Set ID	Health Check
www.example.com	A Record	192.0.2.2	60	Web1	A
www.example.com	A Record	198.51.100.2	60	Web2	B
www.example.com	A Record	203.0.113.2	60	Web3	C

Registrar el dominio vs. Servicio DNS

- Compras o registras tu nombre de dominio con un Registrador de Dominios, normalmente pagando una cuota anual (por ejemplo, GoDaddy, Amazon Registrar Inc., ...)
- El Registrador de dominios suele proporcionarte un servicio de DNS para gestionar tus registros de DNS
- Pero puedes utilizar otro servicio DNS para gestionar tus registros DNS
- Ejemplo: compra el dominio a GoDaddy y utiliza Route 53 para gestionar tus registros DNS



GoDaddy como registrador y Route 53 como servicio DNS



Records

We can't display your DNS information because your nameservers aren't managed by us.

Nameservers

Using custom nameservers [Change](#)

Nameserver
ns-1083.awsdns-07.org
ns-932.awsdns-52.net
ns-1911.awsdns-46.co.uk
ns-481.awsdns-60.com



Amazon
Route 53

Zona de alojamiento público
jamengual.com

▼ Hosted zone details [Edit hosted zone](#)

Hosted zone ID	Type	Name servers
Z30IJCCWPKZUV	Public hosted zone	ns-252.awsdns-31.com ns-1468.awsdns-55.org ns-633.awsdns-15.net ns-1800.awsdns-33.co.uk
Description	Record count	
HostedZone created by Route53 Registrar	22	
Query log		

Registrador de terceros con Amazon Route 53

- **Si compras tu dominio en un registrador de terceros, puedes seguir utilizando Route 53 como proveedor de servicios DNS**
 1. Crear una Zona de alojamiento en Route 53
 2. Actualizar los registros NS en el sitio web de terceros para utilizar los servidores de nombres de Route 53
- **Registrador de dominios != Servicio DNS**
- Pero todos los Registradores de Dominio suelen tener algunas funciones DNS

Fundamentos de la VPC

Visión general de VPC

VPC - Curso intensivo

- VPC es algo que debes conocer en profundidad para el AWS Certified Solutions Architect Associate & AWS Certified SysOps Administrator
- **En el nivel AWS Certified Developer, debes conocer:**
 - VPC, Subredes, Gateways de Internet y Gateways NAT
 - Grupos de seguridad, ACL de red (NACL), Logs de flujo de VPC
 - VPC Peering, VPC Endpoints
 - VPN Site-to-Site (Sitio a Sitio) y Direct Connect (Conexión Directa)
- Sólo te daré una visión general, menos de 1 o 2 preguntas en tu examen.
- Más adelante en el curso, destacaré cuándo son útiles los conceptos de VPC

Introducción a la VPC y las subredes

- **VPC**: red privada para desplegar tus recursos (recurso regional)
- Las **subredes** te permiten particionar tu red dentro de tu VPC (recurso Zona de Disponibilidad)
- Una **subred pública** es una subred accesible desde Internet
- Una **subred privada** es una subred no accesible desde Internet
- Para definir el acceso a Internet y entre subredes, utilizamos **tablas de enrutamiento**.

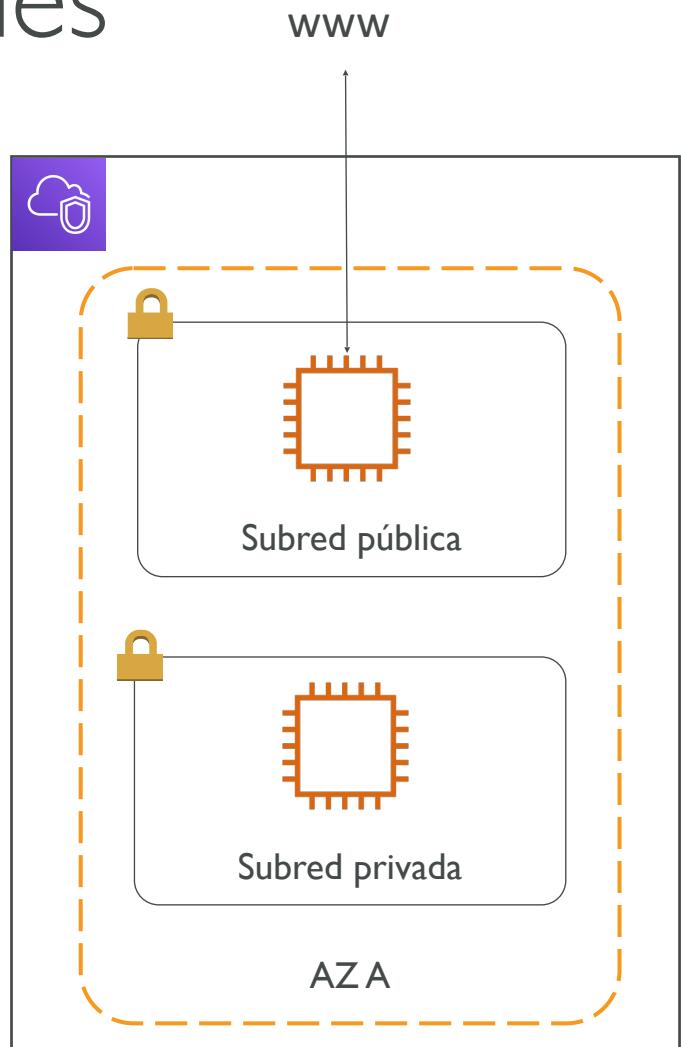
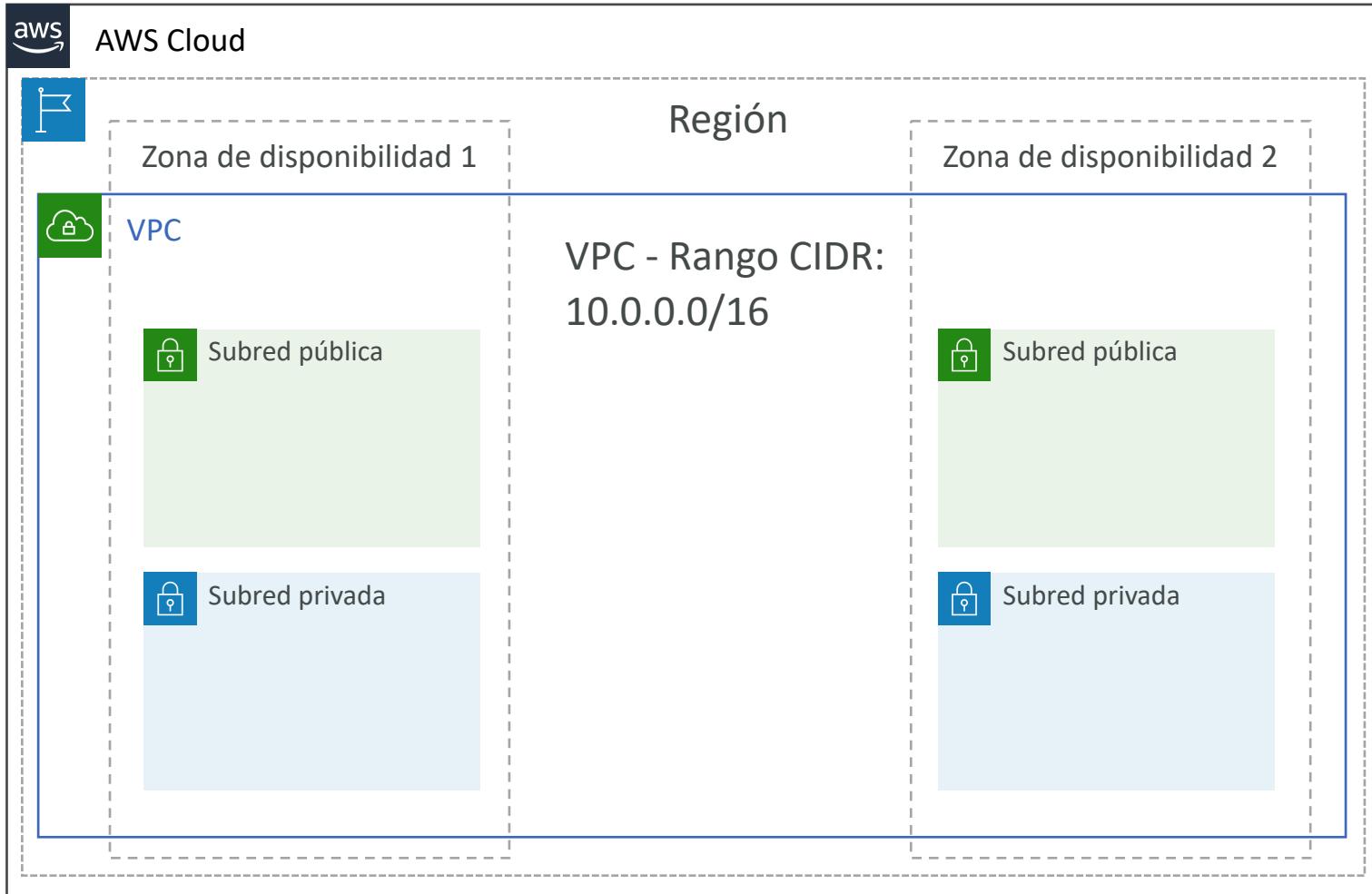
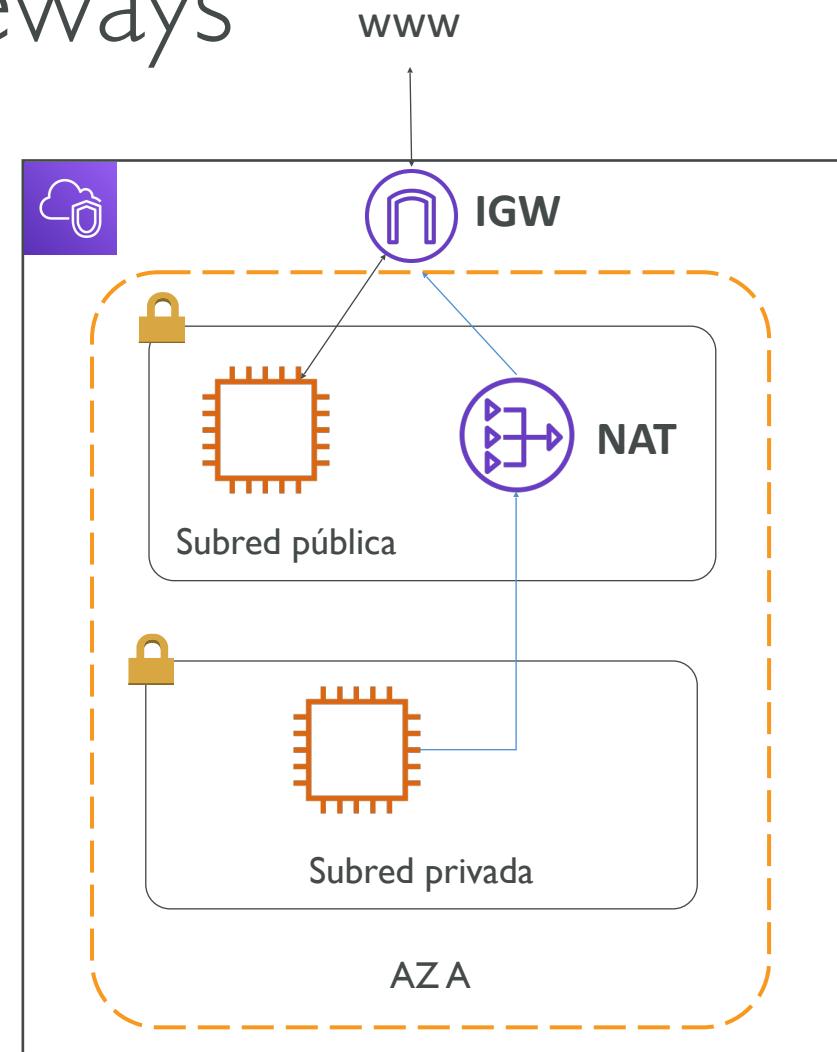


Diagrama VPC



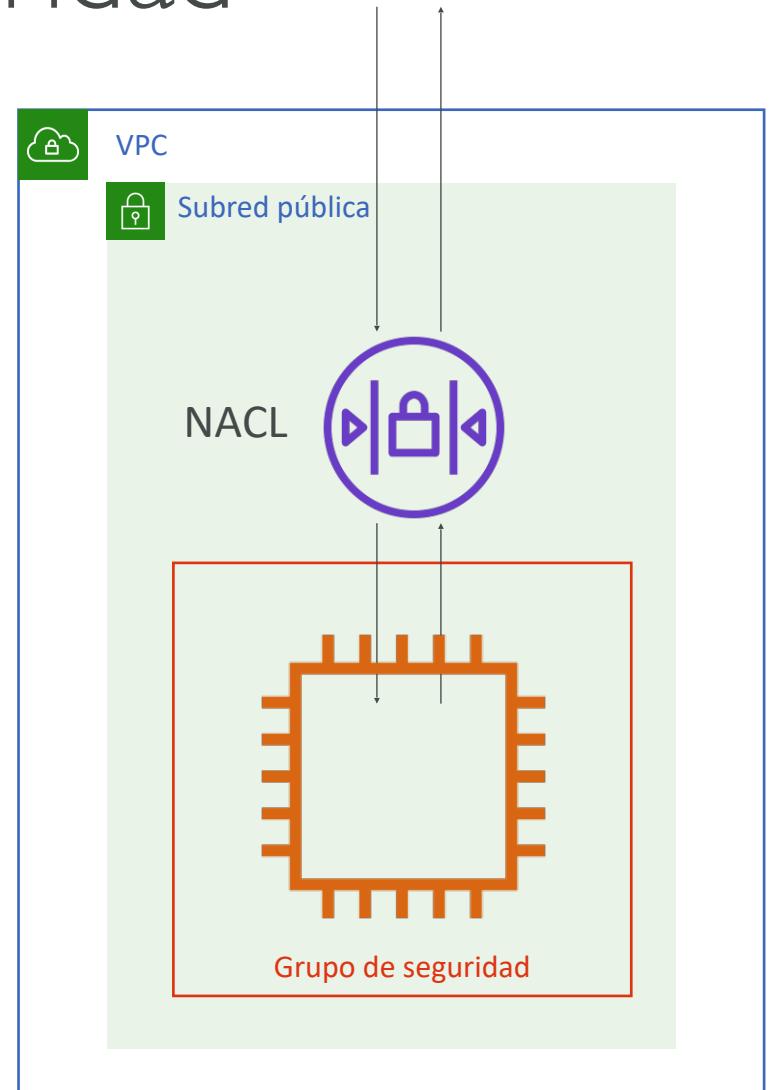
Internet Gateway y NAT Gateways

- Los **Gateways de Internet** ayudan a nuestras instancias VPC a conectarse con Internet
- Las subredes públicas tienen una ruta al Gateway de Internet.
- Los **Gateways NAT** (gestionados por AWS) y las **Instancias NAT** (autogestionadas) permiten a tus instancias en tus **subredes privadas** acceder a Internet sin dejar de ser privadas



ACL de red y Grupos de seguridad

- **NACL** (ACL de red)
 - Un firewall que controla el tráfico desde y hacia la subred
 - Puede tener reglas ALLOW y DENY
 - Se adjuntan a nivel de **subred**
 - Las reglas sólo incluyen direcciones IP
- **Grupos de seguridad**
 - Un firewall que controla el tráfico hacia y desde una **ENI / una Instancia EC2**
 - Sólo puede tener reglas de permiso (ALLOW)
 - Las reglas incluyen direcciones IP y otros grupos de seguridad



ACLs de red vs Grupos de seguridad

Security group (Grupo de seguridad)	ACL de red
Opera en el nivel de la instancia	Opera en el nivel de la subred
Se aplica a una instancia solo si está asociada a la instancia	Se aplica a todas las instancias implementadas en la subred asociada (proporciona una capa de defensa adicional si las reglas del grupo de seguridad son demasiado permisivas)
Solo admite reglas de permiso	Admite reglas de permiso y de denegación
Evalúa todas las normas antes de decidir si permitir el tráfico	Evalúa las reglas en orden, a partir de la regla numerada más baja, al decidir si permitir el tráfico
Con estado: el tráfico de retorno se permite de manera automática, independientemente de las reglas	Sin estado: las reglas deben permitir de forma explícita el tráfico de retorno

https://docs.aws.amazon.com/vpc/latest/userguide/VPC_Security.html#VPC_Security_Comparison

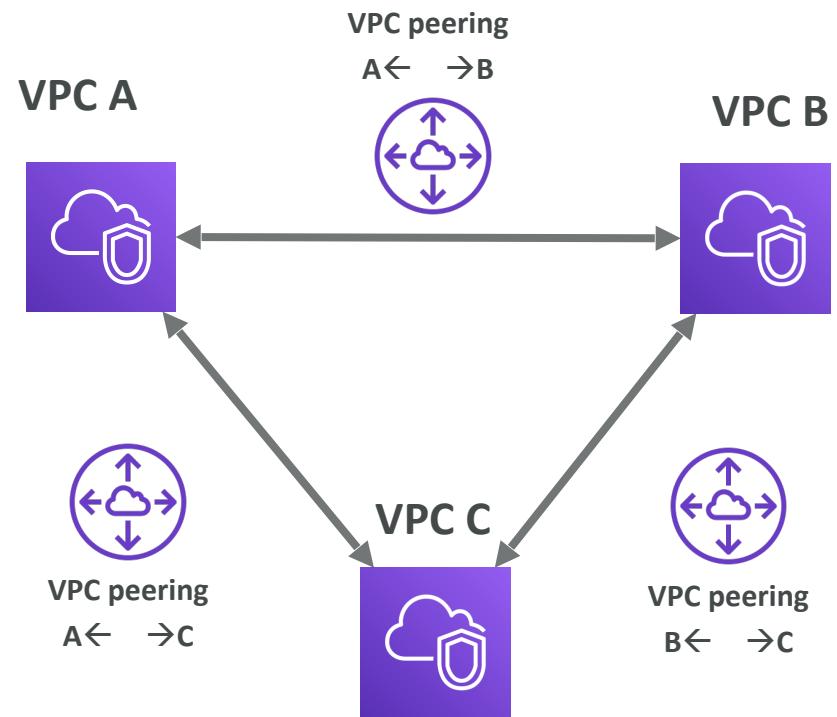
VPC Flow Logs / Logs de flujo de la VPC



- Captura información sobre el tráfico IP que entra en tus interfaces:
 - Logs de flujo de **VPC**
 - Logs de flujo de **subred**
 - Logs de flujo de **Elastic Network Interface (ENI)**
- Ayuda a supervisar y solucionar problemas de conectividad. Ejemplo:
 - Subredes a Internet
 - Subredes a subredes
 - Internet a subredes
- Captura también información de red de interfaces gestionadas por AWS: Elastic Load Balancers, ElastiCache, RDS, Aurora, etc...
- Los datos de logs de VPC Flow pueden ir a S3 / CloudWatch Logs

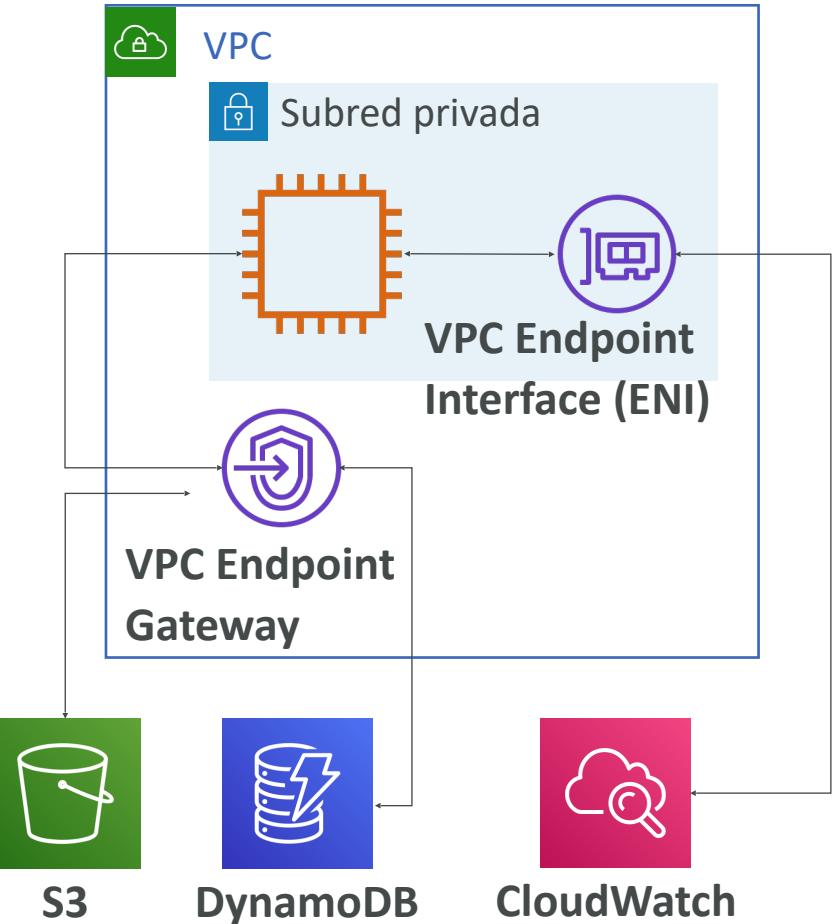
VPC Peering

- Conecta dos VPC, de forma privada utilizando la red de AWS
- Haz que se comporten como si estuvieran en la misma red
- No deben tener CIDR (rango de direcciones IP) solapados
- La conexión de VPC Peering **no es transitiva** (debe establecerse para cada VPC que necesite comunicarse entre sí)



VPC Endpoints

- Los endpoints te permiten conectarte a los servicios de AWS **utilizando una red privada** en lugar de la red www pública
- Esto te proporciona mayor seguridad y menor latencia para acceder a los servicios de AWS
- VPC Endpoint Gateway: S3 y DynamoDB
- VPC Endpoint Interface: el resto
- Sólo se utiliza dentro de tu VPC



VPN Site-to-Site (Sitio a Sitio) y Direct Connect

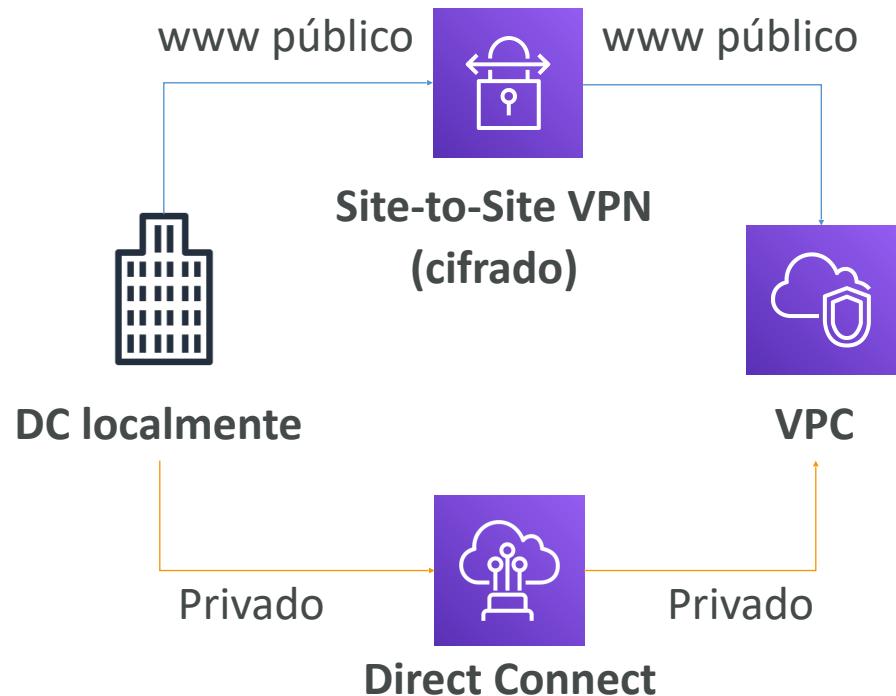
- **VPN Site-to-Site (Sitio a Sitio)**

- Conecta una VPN local a AWS
- La conexión se cifra automáticamente
- Pasa por el Internet público

- **Direct Connect (DX)**

- Establece una conexión física entre las instalaciones y AWS
- La conexión es privada, segura y rápida
- Pasa por una red privada
- Tarda al menos un mes en establecerse

- Nota: VPN Site-to-Site y Direct Connect no pueden acceder a endpoints VPC



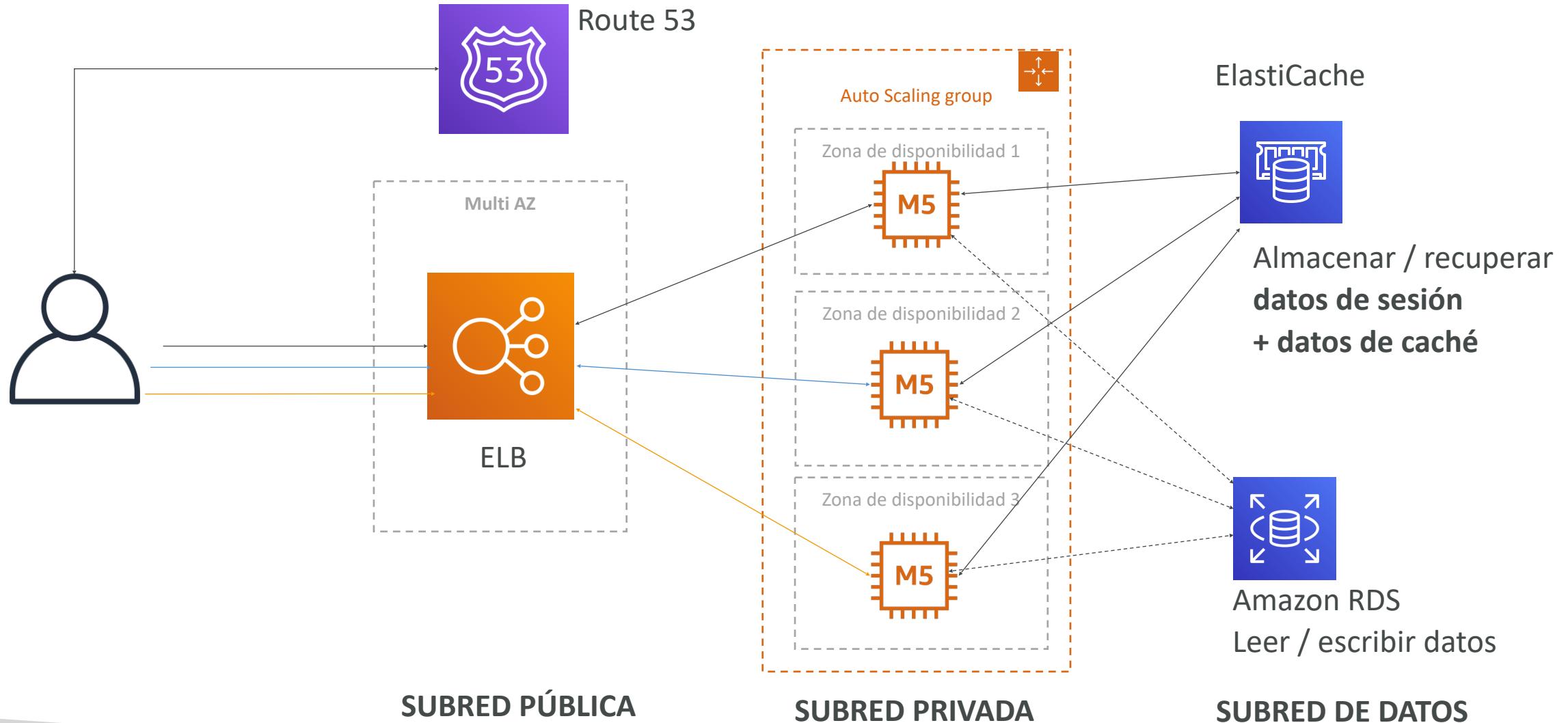
VPC - Comentarios finales

- **VPC:**Virtual Private Cloud (Nube Privada Virtual)
- **Subredes:**Vinculadas a una AZ, partición de red de la VPC
- **Gateway de Internet:**a nivel de la VPC, proporcionan acceso a Internet
- **Gateways NAT o Instancias NAT:** dan acceso a Internet a subredes privadas
- **NACL:** sin estado, reglas de subred para entrantes y salientes
- **Grupos de seguridad:** Con estado, operan a nivel de instancia EC2 o ENI
- **VPC Peering:** Conecta dos VPC con rangos de IP no solapados, no transitivo
- **Endpoints de VPC:** Proporcionan acceso privado a los servicios de AWS dentro de la VPC
- **VPC Flow Logs:** logs de tráfico de red
- **VPN Site to Site (Sitio a Sitio):**VPN a través de Internet pública entre el DC local y AWS
- **Direct Connect (DX):** conexión privada directa a AWS

Nota sobre VPC - AWS Certified Developer

- No te estreses si no lo has entendido todo en esta sección
- En el curso destacaré las características específicas de la VPC que necesitamos.
- Siéntete libre de volver a visitar esa sección cuando hayas terminado el curso.
- Continuamos ☺

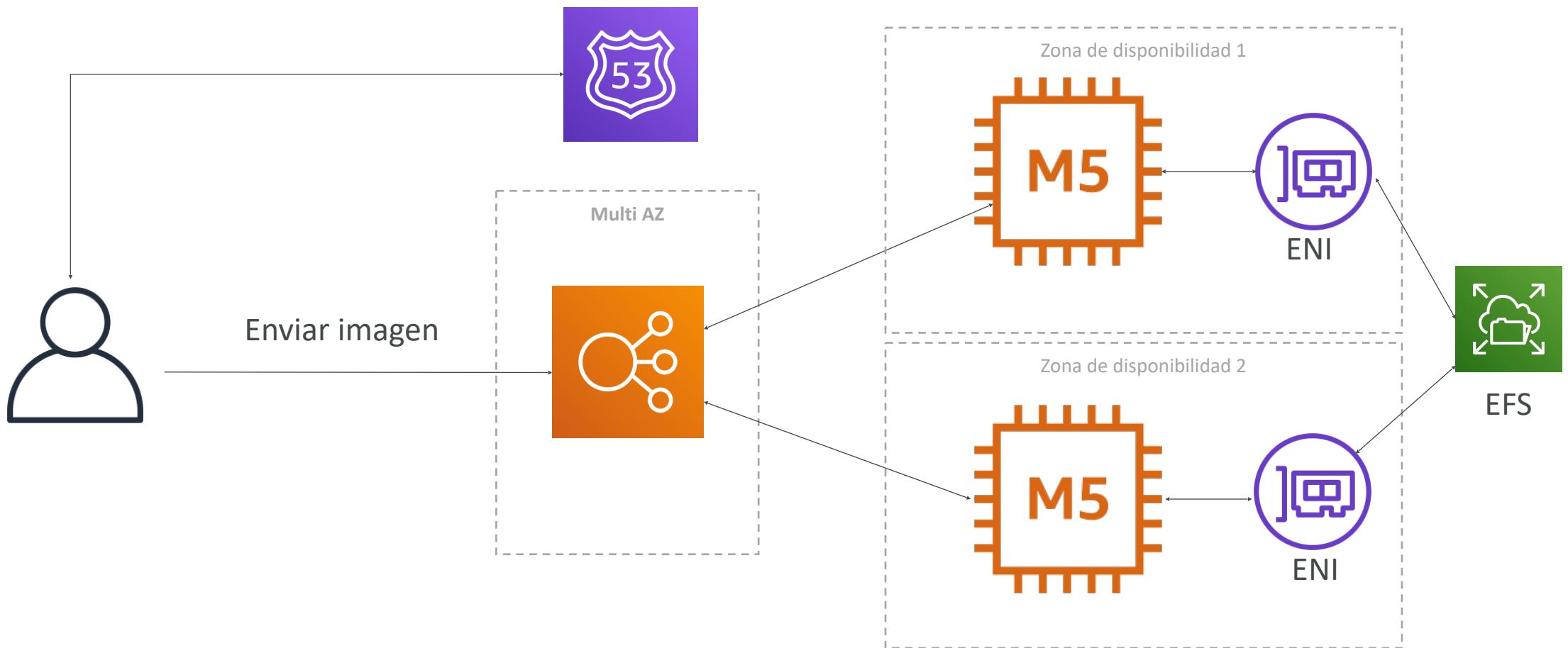
Arquitectura típica de una solución de 3 niveles



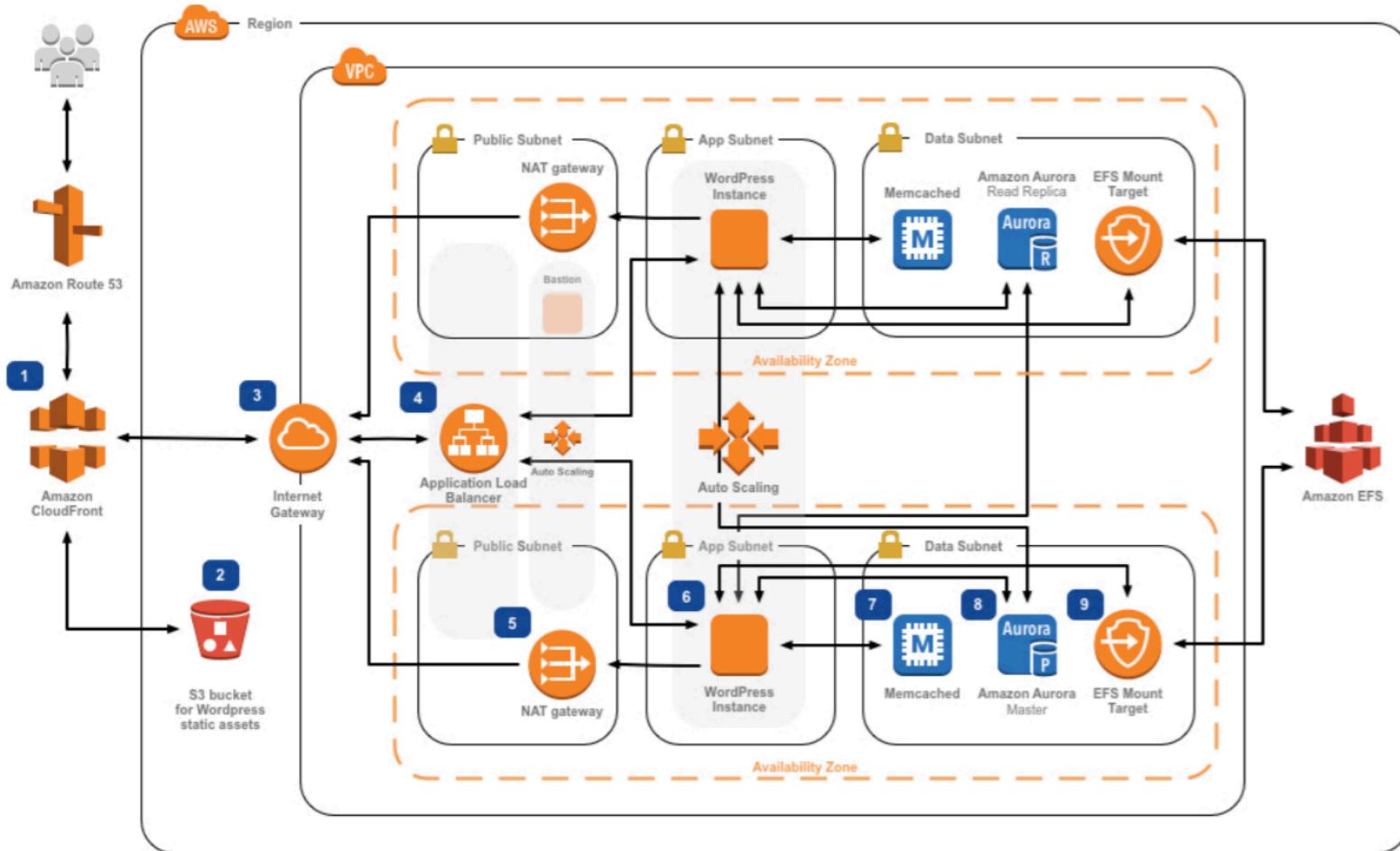
LAMP Stack en EC2

- **L**inux: SO para instancias EC2
 - **A**pache: Servidor web que se ejecuta en Linux (EC2)
 - **M**ySQL: base de datos en RDS
 - **P**HP: Lógica de aplicación (que se ejecuta en EC2)
-
- Puedes añadir Redis / Memcached (**ElastiCache**) para incluir una tecnología de caché
 - Para almacenar los datos y el software de la aplicación local:
 - Unidad **EBS** (root)

Wordpress en AWS



WordPress en AWS (más complicado)



<https://aws.amazon.com/blogs/architecture/wordpress-best-practices-on-aws/>

Amazon S3

Introducción de la sección



- Amazon S3 es uno de los principales bloques de construcción de AWS
- Se anuncia como almacenamiento de "escala infinita".

- Muchos sitios web utilizan Amazon S3 como columna vertebral
- Muchos servicios de AWS utilizan Amazon S3 como una integración también

- Tendremos una aproximación paso a paso a S3

S3 Casos de uso

- Copia de seguridad y almacenamiento
- Recuperación de desastres
- Almacenamiento en el Cloud híbrido
- Alojamiento de aplicaciones
- Alojamiento de medios
- Data Lakes y análisis de big data
- Entrega de software
- Sitio web estático



Nasdaq almacena 7 años de datos en S3 Glacier



Sysco analiza sus datos y obtiene información comercial

Visión general de Amazon S3 - Buckets

- Amazon S3 permite almacenar objetos (archivos) en "buckets" (directorios)
- Los buckets deben tener un **nombre único a nivel global (en todas las regiones, todas las cuentas)**
- Los buckets se definen a nivel de región
- S3 parece un servicio global pero los buckets se crean en una región
 - Convención de nombres
 - Sin mayúsculas
 - Sin guiones bajos
 - 3-63 caracteres de longitud
 - No es una IP
 - Debe comenzar con una letra minúscula o un número



Amazon S3 - Objetos

- Los objetos (archivos) tienen una clave
- La **clave** es la ruta **COMPLETA**:
 - s3://mi-bucket/**mi-archivo.txt**
 - s3://mi-bucket/**mi_carpetal/otra_carpetal/mi_archivo.txt**
- La clave se compone de **prefijo** + **nombre del objeto**
 - s3://mi-bucket/**mi_carpetal/otra_carpetal/mi_fichero.txt**
- No existe el concepto de "directorios" dentro de los buckets (aunque la interfaz de usuario te hará pensar lo contrario)
- Sólo claves con nombres muy largos que contienen barras inclinadas ("/")

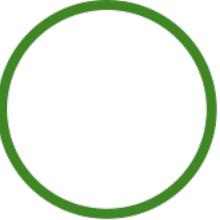


Object



Bucket S3
con objetos

Amazon S3 – Objetos (cont.)

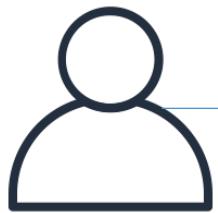


- Los valores de los objetos son el contenido del cuerpo:
 - El tamaño máximo del objeto es de 5TB (5000GB)
 - Si se suben más de 5GB, se debe usar "carga de varias partes"
- Metadatos (lista de pares clave/valor de texto - metadatos del sistema o del usuario)
- Etiquetas (par clave/valor Unicode - hasta 10) - útil para la seguridad/ciclo de vida
- ID de la versión (si el versionado está activado)

Amazon S3 – Seguridad

- **Basada en el usuario**
 - Políticas de IAM: qué llamadas a la API deben permitirse para un usuario específico desde la consola de IAM
- **Basadas en recursos**
 - Políticas de bucket - reglas a nivel de bucket desde la consola de S3 - permite cuentas cruzadas
 - Lista de control de acceso a objetos (ACL)
 - Lista de control de acceso a buckets (ACL)
- **Nota:** un usuario de IAM puede acceder a un objeto de S3 si
 - los permisos IAM del usuario lo permiten Y la política de recursos lo PERMITE
 - Y no hay una DENEGACIÓN explícita
- **Encriptación:** cifra los objetos en Amazon S3 utilizando claves de encriptación

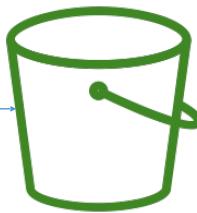
Ejemplo: Acceso público Política de uso de bucket



Visitante anónimo del sitio web www



Política de bucket S3
Permite el acceso público



S3 Bucket

Ejemplo: Acceso del usuario al S3 Permisos IAM



Ejemplo: Acceso a la instancia EC2 Utilizar roles IAM



Avanzado: Acceso entre cuentas

Usar política de bucket

Usuario IAM

Otra cuenta de AWS



Política de bucket S3
Permite las cuentas cruzadas



S3 Bucket

Políticas de bucket S3

- **Políticas basadas en JSON:**

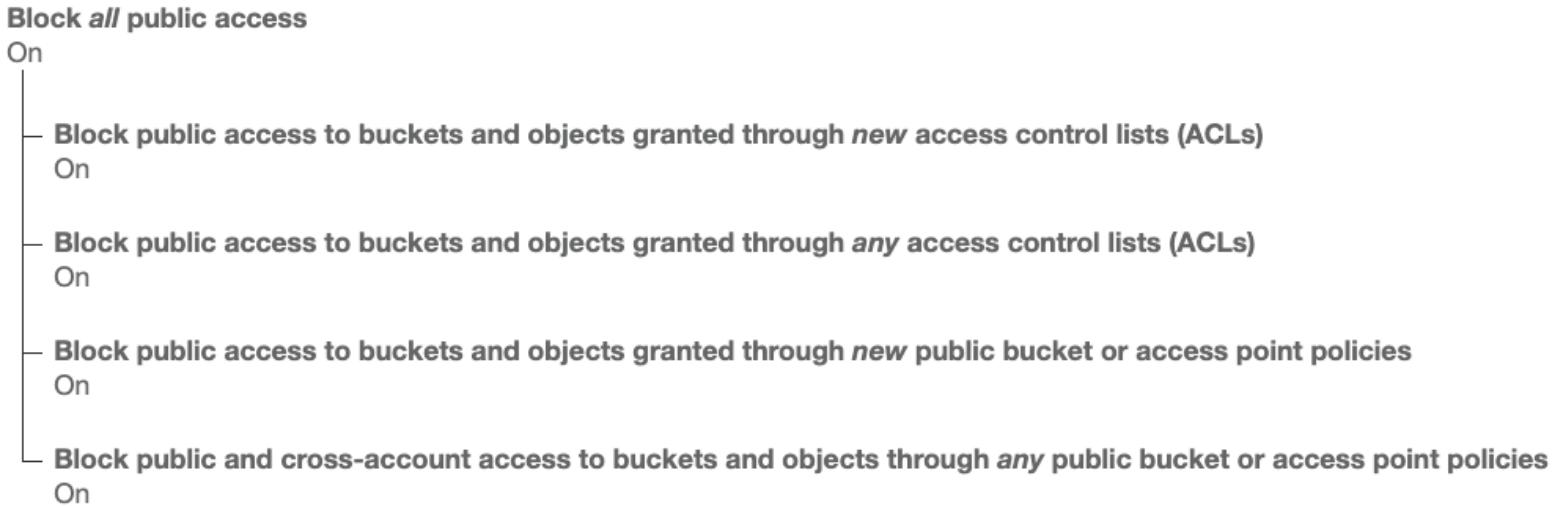
- Resources: buckets y objetos
- Effect: Permitir / Denegar
- Actions: Conjunto de API para permitir o denegar
- Principal: La cuenta o usuario al que aplicar la política

- **Usar el bucket S3 para aplicar la política:**

- Conceder acceso público al bucket
- Forzar que los objetos se cifren al subirlos
- Conceder acceso a otra cuenta (cuenta cruzada)

```
{  
  "Version": "2012-10-17",  
  "Statement": [  
    {  
      "Sid": "PublicRead",  
      "Effect": "Allow",  
      "Principal": "*",  
      "Action": [  
        "s3:GetObject"  
      ],  
      "Resource": [  
        "arn:aws:s3:::examplebucket/*"  
      ]  
    }  
  ]  
}
```

Configuración del bucket para bloquear el acceso público



- Estos ajustes se crearon para evitar la filtración de datos de la empresa
- Si sabes que tu bucket no debe ser nunca público, déjalo activado
- Pueden establecerse a nivel de cuenta

Amazon S3

Alojamiento de sitios web estáticos

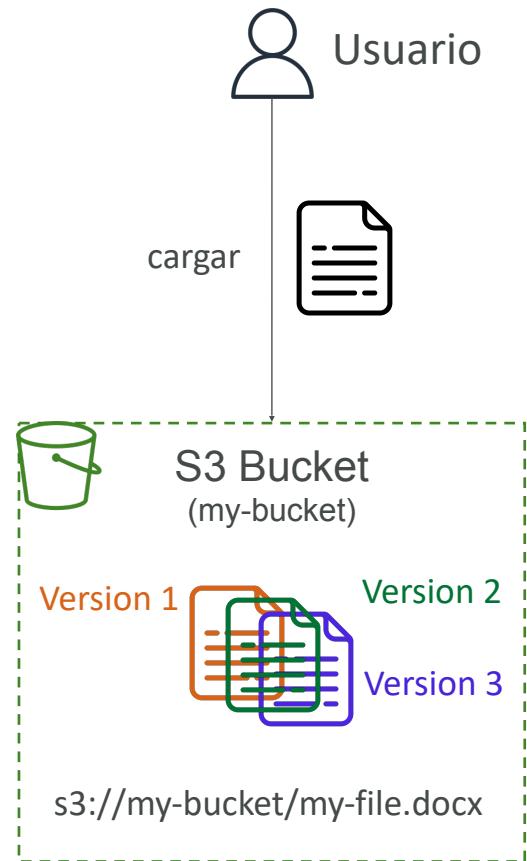
- S3 puede alojar sitios web estáticos y tenerlos accesibles en Internet
- La URL del sitio web será (según la región)
 - [http://bucket-name.s3-website-*aws-region*.amazonaws.com](http://bucket-name.s3-website-us-west-2.amazonaws.com)
 -
 - [http://bucket-name.s3-website.*aws-region*.amazonaws.com](http://bucket-name.s3-website.us-west-2.amazonaws.com)
- Si obtienes un error 403 Prohibido, ¡asegúrate de que la política del bucket permite las lecturas públicas!

<http://demo-bucket.s3-website-us-west-2.amazonaws.com>
<http://demo-bucket.s3-website.us-west-2.amazonaws.com>



Amazon S3 - Versionado

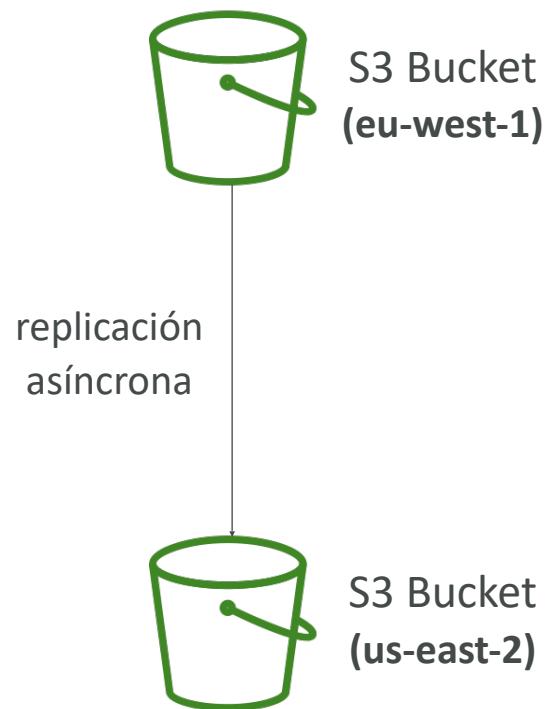
- Puedes versionar tus archivos en Amazon S3
- Se activa a nivel de bucket
- La misma clave de sobrescritura cambiará la "versión": 1, 2, 3...
- Es una buena práctica versionar tus buckets
 - Protege contra los borrados involuntarios (posibilidad de restaurar una versión)
 - Facilidad para volver a la versión anterior
- Notas:
 - Cualquier archivo que no esté versionado antes de activar el versionado tendrá la versión "nula"
 - Suspender el versionado no elimina las versiones anteriores



Amazon S3 - Replicación (CRR y SRR)



- Debe activar el control de versiones en los buckets de origen y destino
- Replicación entre regiones (CRR)
- Replicación en la misma región (SRR)
- Los buckets pueden estar en diferentes cuentas de AWS
- La copia es asíncrona
- Debes dar los permisos IAM adecuados a S3
- Casos de uso:
 - CRR - normativa, acceso de menor latencia, replicación entre cuentas
 - SRR - agregación de logs, replicación en vivo entre cuentas de producción y de prueba



Amazon S3 - Replicación (Notas)

- Después de activar la Replicación, sólo se replican los objetos nuevos
- Opcionalmente, puedes replicar los objetos existentes utilizando la **Replicación por lotes de S3**
 - Replica los objetos existentes y los objetos que fallaron en la replicación
- Para las operaciones de borrado
 - **Puede replicar los marcadores de borrado** del origen al destino (configuración opcional)
 - Los borrados con un ID de versión no se replican (para evitar borrados maliciosos)
- **No hay "encadenamiento" de la replicación**
 - Si el bucket 1 tiene replicación en el bucket 2, que tiene replicación en el bucket 3
 - Entonces los objetos creados en el bucket 1 no se replican en el bucket 3

Clases de almacenamiento S3

- Amazon S3 Standard - General Purpose
 - Amazon S3 Standard-Infrequent Access (IA)
 - Amazon S3 One Zone-Infrequent Access (IA)
 - Amazon S3 Glacier Instant Retrieval
 - Amazon S3 Glacier Flexible Retrieval
 - Amazon S3 Glacier Deep Archive
 - Amazon S3 Intelligent Tiering
-
- Se puede pasar de una clase a otra manualmente o utilizando las configuraciones del ciclo de vida de S3

S3 Durabilidad y disponibilidad

- **Durabilidad:**

- Alta durabilidad (99,99999999%, 11 9's) de los objetos a través de múltiples AZ
- Si almacenas 10.000.000 de objetos con Amazon S3, puedes esperar una media de pérdida de un solo objeto una vez cada 10.000 años
- Lo mismo para todas las clases de almacenamiento

- **Disponibilidad:**

- Mide la disponibilidad de un servicio
- Varía en función de la clase de almacenamiento
- Ejemplo: El Estándar S3 tiene una disponibilidad del 99,99% = no está disponible 53 minutos al año

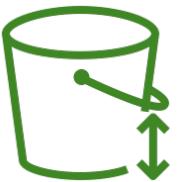


Estándar S3 - Propósito general

- Disponibilidad del 99,99%.
 - Se utiliza para datos de acceso frecuente
 - Baja latencia y alto rendimiento
 - Soporta 2 fallos concurrentes de la instalación
-
- Casos de uso: Análisis de Big Data, aplicaciones móviles y de juegos, distribución de contenidos...

Clases de almacenamiento S3 - Acceso infrecuente

- Para datos a los que se accede con menos frecuencia, pero que requieren un acceso rápido cuando se necesitan
- Coste inferior al de S3 Estándar
- **Amazon S3 Standard-Infrequent Access (S3 Standard-IA)**
 - Disponibilidad del 99,9%.
 - Casos de uso: Recuperación de desastres, copias de seguridad
- **Amazon S3 One Zone-Infrequent Access (S3 One Zone-IA)**
 - Alta durabilidad (99,999999999%) en una sola AZ; los datos se pierden cuando se destruye la AZ
 - Disponibilidad del 99,5%.
 - Casos de uso: Almacenamiento de copias de seguridad secundarias de datos locales, o de datos que puedes recrear



Clases de almacenamiento de Amazon S3 Glacier

- Almacenamiento de objetos de bajo coste pensado para archivar / hacer copias de seguridad
- Precio: precio del almacenamiento + coste de recuperación del objeto
- **Amazon S3 Glacier Instant Retrieval**
 - Recuperación en milisegundos, ideal para datos a los que se accede una vez al trimestre
 - Duración mínima de almacenamiento de 90 días
- **Amazon S3 Glacier Flexible Retrieval** (antes Amazon S3 Glacier)
 - Acelerada (de 1 a 5 minutos), Estándar (de 3 a 5 horas), Masiva (de 5 a 12 horas) - gratis
 - Duración mínima de almacenamiento de 90 días
- **Amazon S3 Glacier Deep Archive** - para almacenamiento a largo plazo:
 - Estándar (12 horas), Masiva (48 horas)
 - Duración mínima de almacenamiento de 180 días





S3 Intelligent-Tiering

- Pequeña cuota mensual de monitorización y jerarquización automática
 - Mueve los objetos automáticamente entre los niveles de acceso en función del uso
 - No hay cargos por recuperación en S3 Intelligent-Tiering
-
- *Frequent Access tier (automático)*: nivel por defecto
 - *Infrequent Access tier (automático)*: objetos no accedidos durante 30 días
 - *Archive Instant Access tier (automático)*: objetos no accedidos durante 90 días
 - *Archive Access tier (opcional)*: configurable de 90 a más de 700 días
 - *Deep Archive Access tier (opcional)*: configurable de 180 días a 700+ días

Comparación de las clases de almacenamiento de S3

	Standard	Intelligent-Tiering	Standard-IA	One Zone-IA	Glacier Instant Retrieval	Glacier Flexible Retrieval	Glacier Deep Archive
Durabilidad	99.999999999% == (11 9's)						
Disponibilidad	99.99%	99.9%	99.9%	99.5%	99.9%	99.99%	99.99%
Acuerdo de nivel de servicio de disponibilidad	99.9%	99%	99%	99%	99%	99.9%	99.9%
Zonas de disponibilidad	>= 3	>= 3	>= 3	1	>= 3	>= 3	>= 3
Min. Duración del almacenamiento	Ninguno	Ninguno	30 Días	30 Días	90 Días	90 Días	180 Días
Min. Tamaño del objeto facturable	Ninguno	Ninguno	128 KB	128 KB	128 KB	40 KB	40 KB
Tasa de recuperación	Ninguno	Ninguno	Por GB recuperado	Por GB recuperado	Por GB recuperado	Por GB recuperado	Por GB recuperado

<https://aws.amazon.com/s3/storage-classes/>

Clases de almacenamiento S3 - Comparación de precios

Ejemplo: us-east-1

	Standard	Intelligent-Tiering	Standard-IA	One Zone-IA	Glacier Instant Retrieval	Glacier Flexible Retrieval	Glacier Deep Archive
Coste de almacenamiento (por GB al mes)	0.023\$	0.0025\$ - 0.023\$	%0.0125	0.01\$	0.004\$	0.0036\$	0.00099\$
Coste de recuperación (por cada 1000 solicitudes)	GET: 0.0004\$ POST: 0.005\$	GET: 0.0004\$ POST: 0.005\$	GET: 0.001\$ POST: 0.01\$	GET:\$0.001\$ POST: 0.01\$	GET: 0.01\$ POST: 0.02\$	GET: 0.0004\$ POST: 0.03\$ Expedited: 10\$ Standard: 0.05\$ Bulk: gratis	GET: 0.0004\$ POST: 0.05\$ Standard: 0.10\$ Bulk: 0.025\$
Tiempo de recuperación	Instantáneo					Expedited (1 – 5 mins) Standard (3 – 5 hours) Bulk (5 – 12 hours)	Standard (12 horas) Bulk (48 horas)
Coste de la monitorización (1000 objetos)		0.0025\$					

<https://aws.amazon.com/s3/pricing/>

Desarrollar en AWS

CLI, SDK y Políticas IAM

Pruebas en la CLI de AWS - Dry Runs

- A veces, sólo queremos asegurarnos de que tenemos los permisos...
- ¡Pero no ejecutar realmente los comandos!
- Algunos comandos CLI de AWS (como EC2) pueden resultar caros si tienen éxito, por ejemplo, si quisiéramos intentar crear una instancia EC2
- Algunos comandos CLI de AWS (no todos) contienen una opción:
 - **--dry-run** para simular llamadas a la API
- ¡A practicar!

Errores de descodificación del STS en la CLI de AWS

- Cuando ejecutas llamadas a la API y fallan, puedes obtener un largo mensaje de error
- Este mensaje de error puede descodificarse utilizando la línea de comandos **STS**:
 - `sts decode-authorization-message`
- ¡Vamos a practicar!

Metadatos de la instancia de AWS EC2

- Los metadatos de las instancias de AWS EC2 son potentes pero una de las características menos conocidas por los desarrolladores
- Permite que las instancias de AWS EC2 "aprendan sobre sí mismas" **sin necesidad de utilizar un rol de IAM para ello.**
- La URL es <http://169.254.169.254/latest/meta-data>
- Puedes recuperar el nombre del rol IAM de los metadatos, pero NO puedes recuperar la política IAM.
- Metadatos = Información sobre la instancia EC2

MFA con CLI

- Para utilizar MFA con la CLI, debes crear una sesión temporal
- Para ello, debes ejecutar la llamada a la API **STS GetSessionToken**
- **aws sts get-session-token** --serial-number arn-of-the-mfa-device --token-code code-from-token --duration-seconds 3600

```
{  
  "Credentials": {  
    "SecretAccessKey": "secret-access-key",  
    "SessionToken": "temporary-session-token",  
    "Expiration": "expiration-date-time",  
    "AccessKeyId": "access-key-id"  
  }  
}
```

Visión general del SDK de AWS

- ¿Y si quieres realizar acciones en AWS directamente desde el código de tus aplicaciones? (sin usar la CLI).
- ¡Puedes utilizar un SDK (kit de desarrollo de software) !
- Los SDK oficiales son...
 - Java
 - .NET
 - Node.js
 - PHP
 - Python (llamado boto3 / botocore)
 - Go
 - Ruby
 - C++

Visión general del SDK de AWS

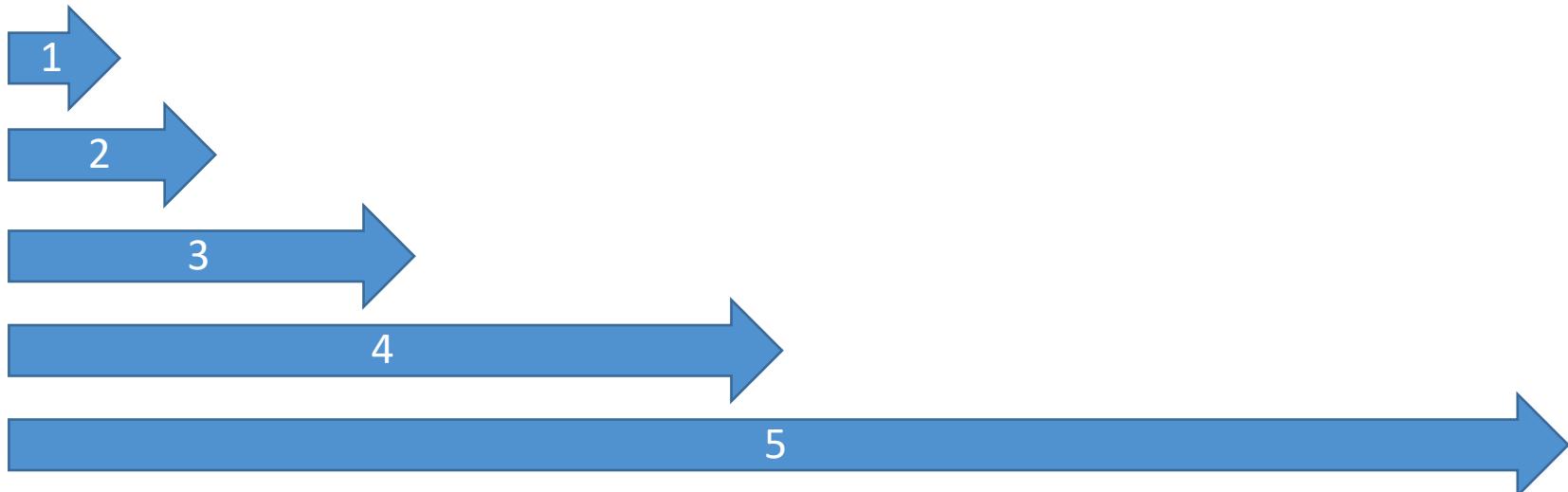
- Tenemos que utilizar el SDK de AWS cuando programamos contra servicios de AWS como DynamoDB
- Dato curioso... la CLI de AWS utiliza el SDK de Python (boto3)
- El examen espera que sepas cuándo debes utilizar un SDK
- Practicaremos el SDK de AWS cuando lleguemos a las funciones Lambda
- Es bueno saberlo: si no especificas o configuras una región por defecto, se elegirá por defecto us-east-1

AWS Limits (Quotas)

- Límites de velocidad de la API
 - La API **DescribeInstances** para EC2 tiene un límite de 100 llamadas por segundo
 - **GetObject** en S3 tiene un límite de 5500 GET por segundo por prefijo
 - Para errores intermitentes: implementa el Backoff Exponencial
 - Para errores constantes: solicita un aumento del límite de estrangulamiento de la API
- Cuotas de Servicio (Límites de Servicio)
 - Ejecución de instancias estándar bajo demanda: 1152 vCPU
 - Puedes solicitar un aumento del límite de servicio **abriendo un ticket**
 - Puedes solicitar un aumento de la cuota de servicio utilizando:
 - **Service Quotas API**

Backoff exponencial (cualquier servicio de AWS)

- Si obtienes **ThrottlingException** de forma intermitente, utiliza backoff exponencial
- El mecanismo de reintento ya está incluido en las llamadas a la API del SDK de AWS
- Debes implementarlo tú mismo si utilizas la API de AWS tal cual o en casos específicos
 - **Sólo debes implementar los reintentos en errores de servidor 5xx y “estrangulamiento”**
 - No implementar en los errores de cliente 4xx



Cadena de proveedores de credenciales de la CLI de AWS

- La CLI buscará las credenciales en este orden
 1. **Opciones de la línea de comandos** - --region, --output y --profile
 2. **Variables de entorno** - AWS_ACCESS_KEY_ID, AWS_SECRET_ACCESS_KEY y AWS_SESSION_TOKEN
 3. **Archivo de credenciales de la CLI** - aws configure ~/.aws/credentials en Linux / Mac y C:\Users\user\.aws\credentials en Windows
 4. **Archivo de configuración CLI** - aws configure ~/.aws/config en Linux / macOS y C:\Users\USERNAME\.aws\config en Windows
 5. **Credenciales del contenedor** - para tareas ECS
 6. **Credenciales del perfil de instancia** - para perfiles de instancia EC2

Cadena del proveedor de credenciales predeterminado del SDK de AWS

- El SDK de Java (ejemplo) buscará las credenciales en este orden
 1. **Propiedades del sistema Java** - aws.accessKeyId y aws.secretKey
 2. **Variables de entorno** - AWS_ACCESS_KEY_ID y AWS_SECRET_ACCESS_KEY
 3. **El archivo de perfiles de credenciales por defecto** - ex at: `~/.aws/credentials`, compartido por muchos SDK
 4. **Credenciales de contenedor de Amazon ECS** - para contenedores ECS
 5. **Credenciales de perfil de instancia** - utilizadas en instancias EC2

Escenario de credenciales de AWS

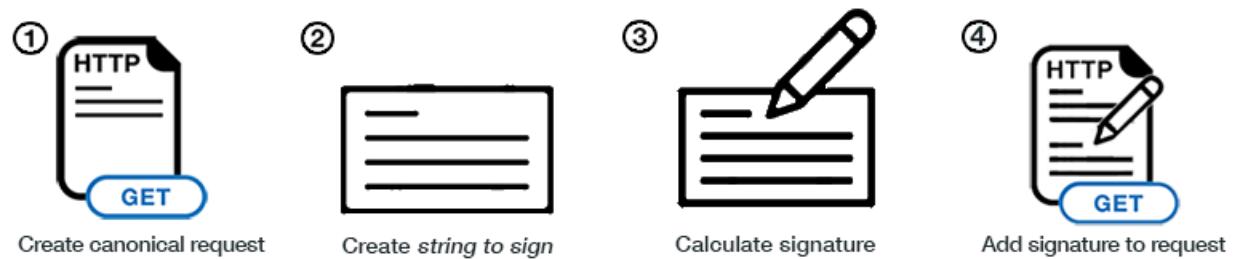
- Una aplicación desplegada en una instancia EC2 está utilizando variables de entorno con credenciales de un usuario IAM para llamar a la API de Amazon S3.
- El usuario IAM tiene permisos S3FullAccess.
- La aplicación sólo utiliza un bucket de S3, por lo que de acuerdo con las mejores prácticas:
 - Se creó un rol IAM y un perfil de instancia EC2 para la instancia EC2
 - Al rol se le asignaron los permisos mínimos para acceder a ese único bucket de S3.
- **Se asignó el Perfil de Instancia IAM a la instancia EC2, pero seguía teniendo acceso a todos los buckets S3. ¿Por qué?**
 - la cadena de credenciales sigue dando prioridad a las variables de entorno

Mejores prácticas para las credenciales de AWS

- **NUNCA ALMACENES CREDENCIALES DE AWS EN TU CÓDIGO**
- La mejor práctica es que las credenciales se hereden de la cadena de credenciales
- **Si trabajas en AWS, utiliza roles IAM**
 - => Roles de Instancias EC2 para Instancias EC2
 - => Roles ECS para tareas ECS
 - => Roles Lambda para funciones Lambda
- Si trabajas fuera de AWS, utiliza variables de Entorno / perfiles con nombre

Firmar peticiones a la API de AWS

- Cuando llamas a la API HTTP de AWS, firmas la petición para que AWS pueda identificarte, utilizando tus credenciales de AWS (clave de acceso y clave secreta)
- Nota: algunas peticiones a Amazon S3 no necesitan ser firmadas
- Si utilizas el SDK o la CLI, las peticiones HTTP se firman por ti
- Debes firmar una petición HTTP de AWS utilizando la Firma v4 (SigV4)



Ejemplos de peticiones SigV4

- Opción de encabezado HTTP (firma en el encabezado Autorización)

```
GET https://iam.amazonaws.com/?Action=ListUsers&Version=2010-05-08 HTTP/1.1
Authorization: AWS4-HMAC-SHA256 Credential=AKIDEXAMPLE/20150830/us-east-1/iam/aws4_request,
SignedHeaders=content-type;host;x-amz-date,
Signature=5d672d79c15b13162d9279b0855cfba6789a8edb4c82c400e06b5924a6f2b5d7
content-type: application/x-www-form-urlencoded; charset=utf-8
host: iam.amazonaws.com
x-amz-date: 20150830T123600Z
```

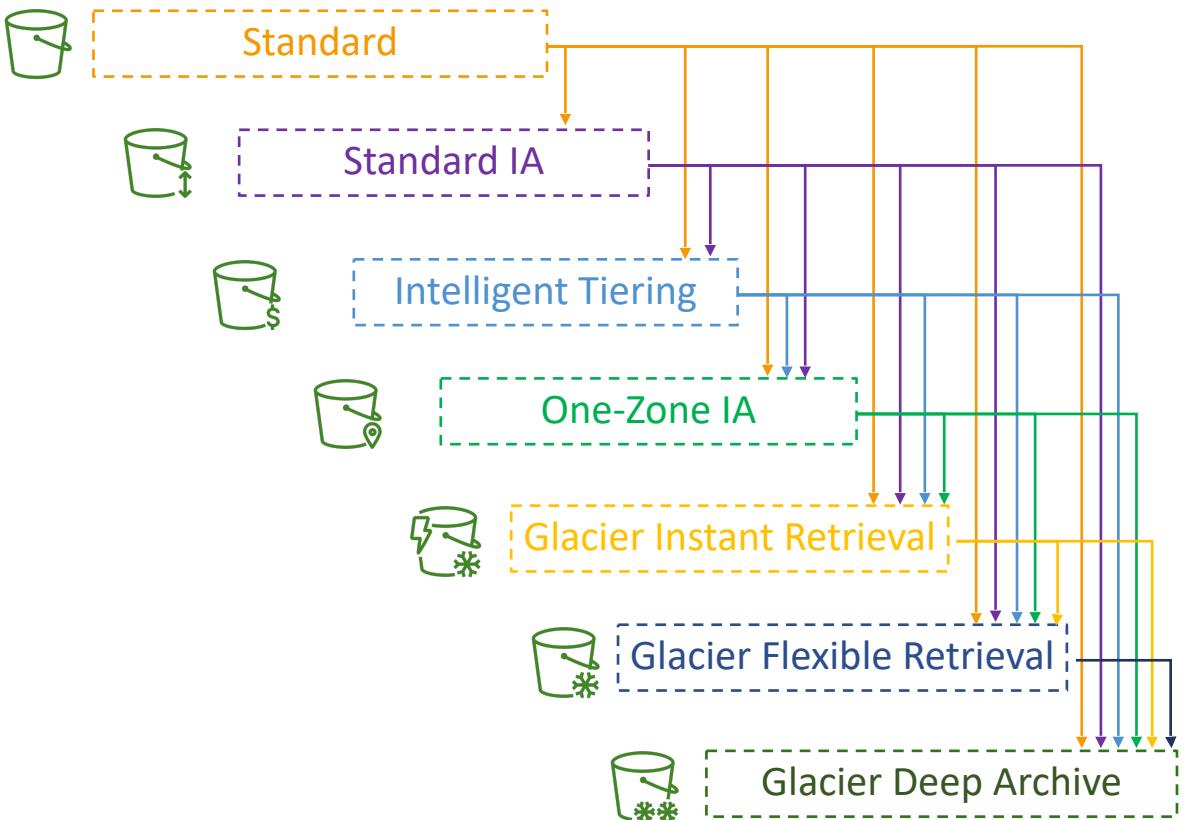
- Opción de cadena de consulta, ej: URL pre-firmadas S3 (firma en X-Amz-Signature)

```
GET https://iam.amazonaws.com?Action=ListUsers&Version=2010-05-08&
X-Amz-Algorithm=AWS4-HMAC-SHA256&
X-Amz-Credential=AKIDEXAMPLE%2F20150830%2Fus-east-1%2Fiam%2Faws4_request&
X-Amz-Date=20150830T123600Z&X-Amz-Expires=60&X-Amz-SignedHeaders=content-type%3Bhost&
X-Amz-Signature=37ac2f4fde00b0ac9bd9eadeb459b1bbee224158d66e7ae5fcadb70b2d181d02 HTTP/1.1
content-type: application/x-www-form-urlencoded; charset=utf-8
host: iam.amazonaws.com
```

S3 avanzado

Amazon S3 - Movimiento entre clases de almacenamiento

- Puedes pasar los objetos entre las clases de almacenamiento
- Para los objetos a los que se accede con poca frecuencia, muévelos a IA Estándar
- Para los objetos de archivo a los que no necesitas acceder rápidamente, muévelos a Glacier o Glacier Deep Archive
- El movimiento de los objetos puede automatizarse mediante las Reglas del Ciclo de Vida



Amazon S3 - Reglas del ciclo de vida



- **Acciones de transición:** configura los objetos para que pasen a otra clase de almacenamiento
 - Mover los objetos a la clase IA Estándar 60 días después de su creación
 - Mover a Glacier para archivar después de 6 meses
- **Acciones de expiración** - configura los objetos para que caduquen (se eliminen) después de un tiempo
 - Los archivos de logs de acceso pueden configurarse para que se eliminen después de 365 días
 - **Se puede utilizar para eliminar versiones antiguas de archivos (si el versionado está activado)**
 - Se puede utilizar para eliminar subidas incompletas de Multipartes
- Se pueden crear reglas para un determinado prefijo (ejemplo: s3://mybucket/mp3/*)
- Se pueden crear reglas para determinados objetos Etiquetas (ejemplo: Departamento: Finanzas)

Amazon S3 - Reglas del ciclo de vida (escenario I)

- Tu aplicación en EC2 crea imágenes en miniatura después de subir las fotos del perfil a Amazon S3. Estas miniaturas pueden recrearse fácilmente, y sólo deben conservarse durante 60 días. Las imágenes de origen deben poder recuperarse inmediatamente durante estos 60 días, y después, el usuario puede esperar hasta 6 horas. ¿Cómo diseñarías esto?
- Las imágenes de origen de S3 pueden estar en **Estándar**, con una configuración del ciclo de vida para que pasen a **Glacier** después de 60 días
- Las miniaturas de S3 pueden estar en **IA de Zona Única**, con una configuración del ciclo de vida para que caduquen (se eliminen) después de 60 días

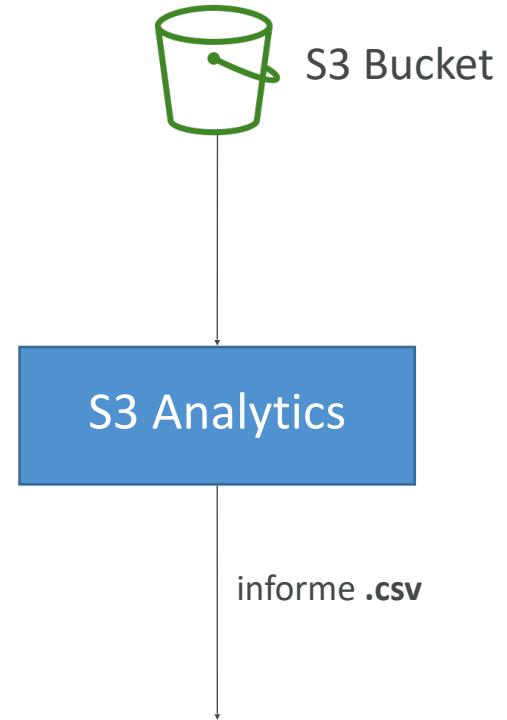
Amazon S3 - Reglas del ciclo de vida (escenario 2)

- Una norma de tu empresa establece que deberías poder recuperar tus objetos S3 eliminados inmediatamente durante 30 días, aunque esto puede ocurrir en raras ocasiones. Después de este tiempo, y durante un máximo de 365 días, los objetos eliminados deberían poder recuperarse en 48 horas.
- **Activa el Versionado de S3** para tener versiones de los objetos, de modo que los "objetos eliminados" queden de hecho ocultos por un "marcador de eliminación" y puedan ser recuperados
- Transitar las "versiones no actuales" del objeto a la **IA estándar**
- Transita después las "versiones no actuales" a **Glacier Deep Archive**

Amazon S3 Analytics

Análisis de la clase de almacenamiento

- Te ayuda a decidir cuándo pasar los objetos a la clase de almacenamiento adecuada
- Recomendaciones para **IA estándar** y **estándar**
 - NO sirve para IA de una Zona o Glacier
- El informe se actualiza diariamente
- De 24 a 48 horas para empezar a ver el análisis de los datos
- Buen primer paso para elaborar las Reglas del Ciclo de Vida (o mejorarlas)

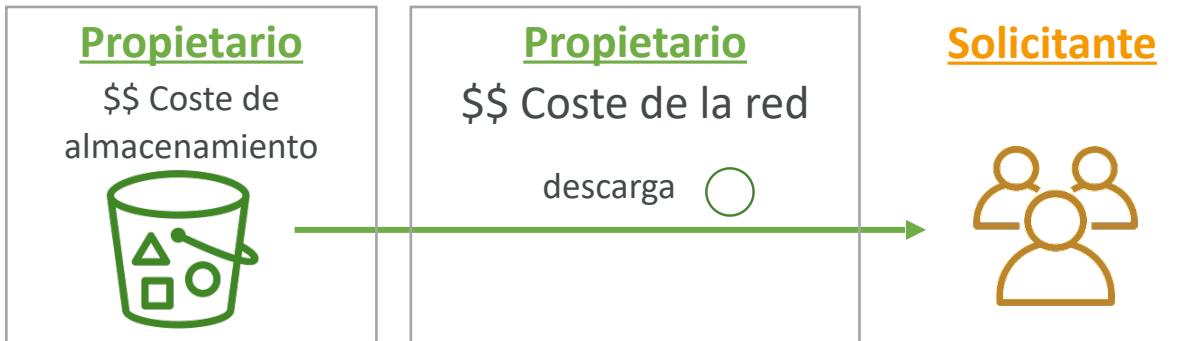


Fecha	StorageClass	ObjectAge
8/22/2022	STANDARD	000-014
8/25/2022	STANDARD	030-044
9/6/2022	STANDARD	120-149

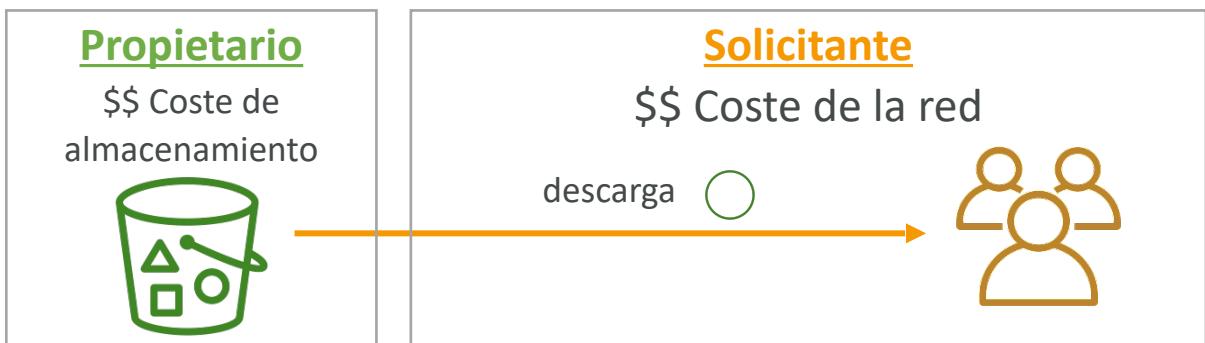
S3 - El solicitante paga

- En general, los propietarios de los buckets pagan todos los costes de almacenamiento y transferencia de datos de Amazon S3 asociados a su bucket
- **Con los buckets donde el solicitante paga**, el solicitante en lugar del propietario del bucket, paga el coste de la petición y la descarga de datos del bucket
- Es útil cuando quieras compartir grandes conjuntos de datos con otras cuentas
- El solicitante debe estar autenticado en AWS (no puede ser anónimo)

Bucket estándar

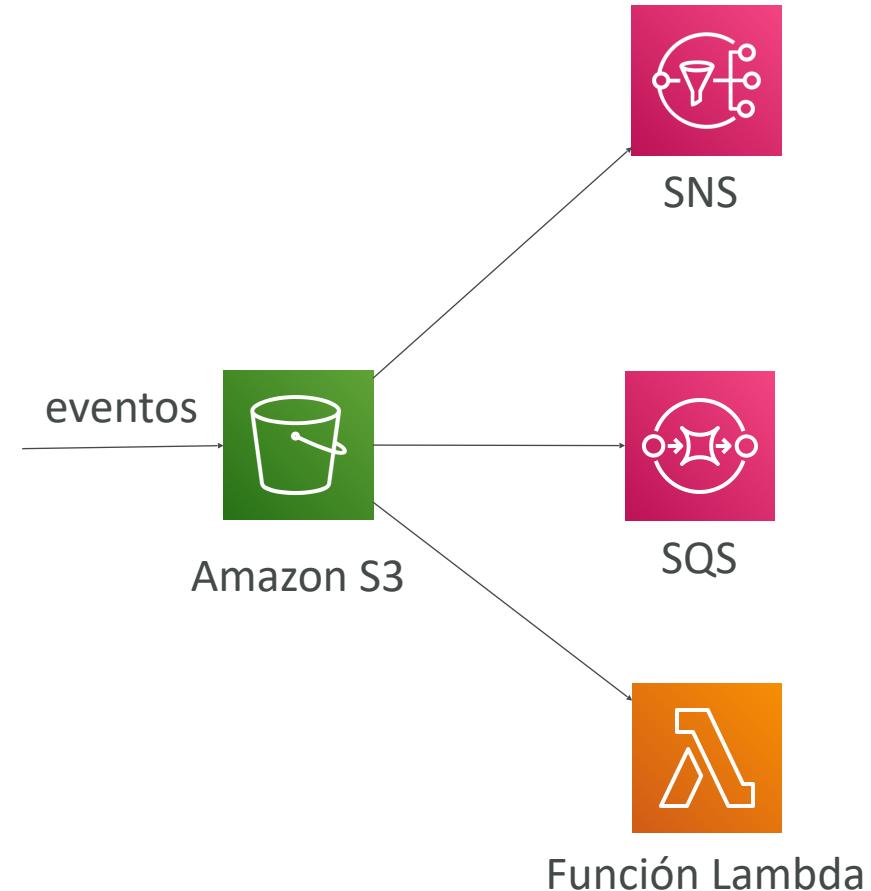


El solicitante paga el bucket

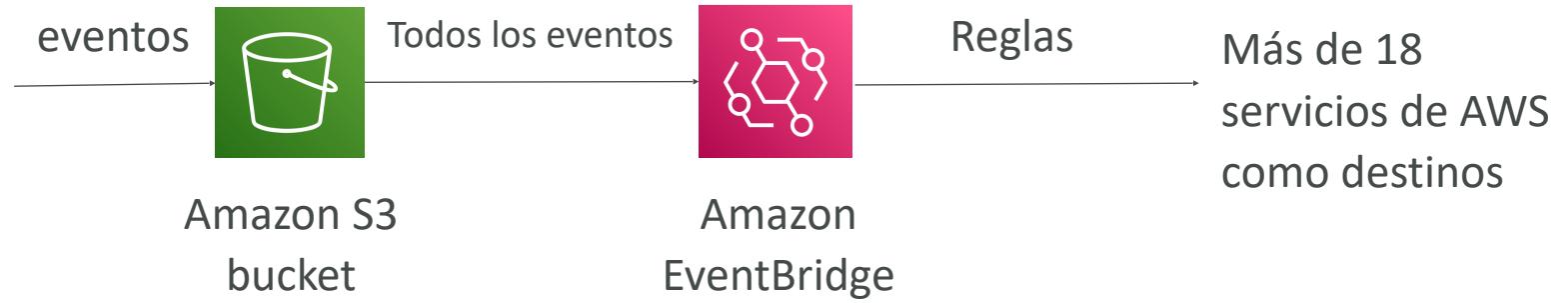


Notificaciones de eventos S3

- S3:ObjectCreated, S3:ObjectRemoved, S3:ObjectRestore, S3:Replication...
- Posibilidad de filtrar el nombre del objeto (*.jpg)
- Caso de uso: generar miniaturas de imágenes subidas a S3
- **Se pueden crear tantos "eventos S3" como se desee**
- Las notificaciones de eventos S3 suelen entregar los eventos en segundos, pero a veces pueden tardar un minuto o más



Notificaciones de eventos S3 con Amazon EventBridge



- **Opciones avanzadas** de filtrado con reglas JSON (metadatos, tamaño del objeto, nombre...)
- **Múltiples destinos** - Step Functions, Kinesis Streams / Firehose...
- **Capacidades de EventBridge** - Repetición de eventos, entrega fiable

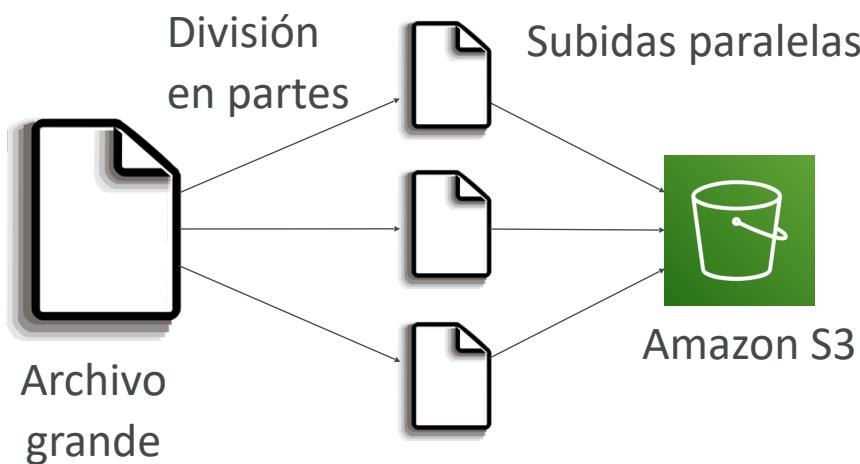
S3 - Rendimiento básico

- Amazon S3 escala automáticamente a altas tasas de petición, latencia 100-200 ms
- Tu aplicación puede alcanzar al menos **3.500 peticiones PUT/COPY/POST/DELETE y 5.500 GET/HEAD por segundo por prefijo en un bucket.**
- No hay límites en el número de prefijos de un bucket.
- Ejemplo (ruta del objeto => prefijo):
 - bucket/carpeta1/sub1/fichero => /carpeta1/sub1/
 - bucket/carpeta1/sub2/fichero => /carpeta1/sub2/
 - bucket/1/fichero => /1/
 - bucket/2/fichero => /2/
- Si distribuyes las lecturas entre los cuatro prefijos de manera uniforme, puedes conseguir 22.000 peticiones por segundo para GET y HEAD

Rendimiento S3 (Performance)

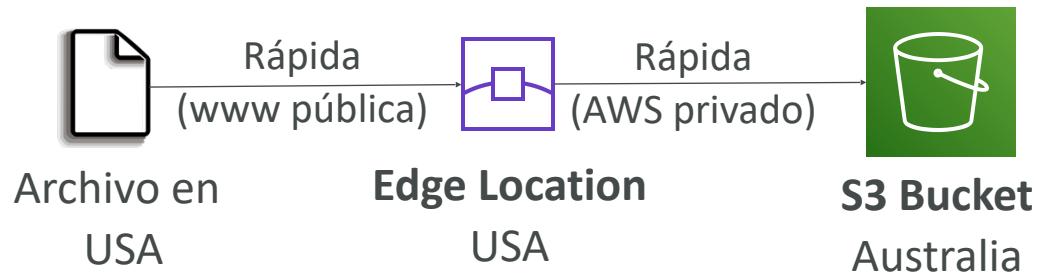
- **Carga de varias partes (Multipartes):**

- Recomendado para archivos > 100MB, obligatorio para archivos > 5GB
- Puede ayudar a paralelizar las subidas (acelerar las transferencias)



- **Aceleración de la transferencia en S3:**

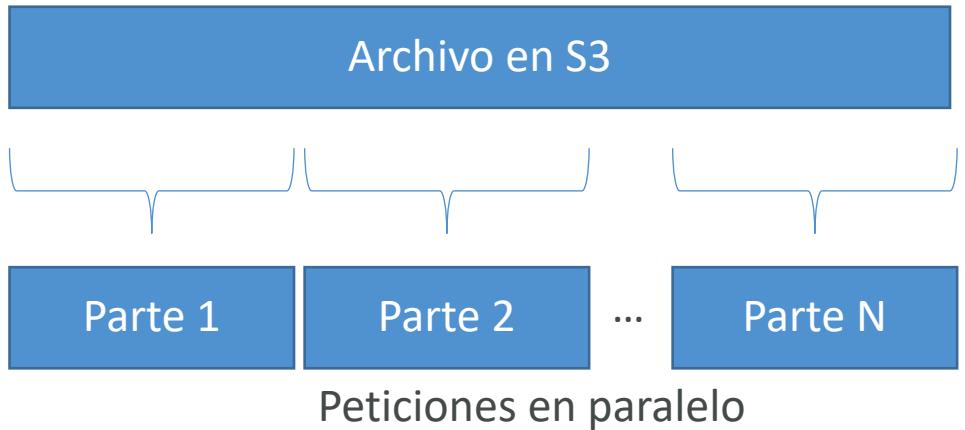
- Aumenta la velocidad de transferencia transfiriendo el archivo a un Edge Location de AWS que reenviará los datos al bucket de S3 en la región de destino
- Compatible con la carga de varias partes



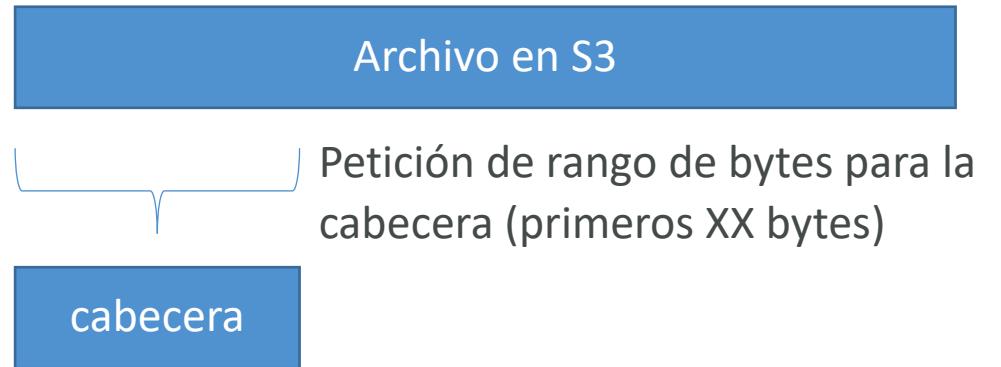
Rendimiento del S3

Recuperación del rango de bytes del S3

- Paraleliza los GETs solicitando rangos de bytes específicos
- Mejor resiliencia en caso de fallos
- Puede utilizarse para acelerar las descargas

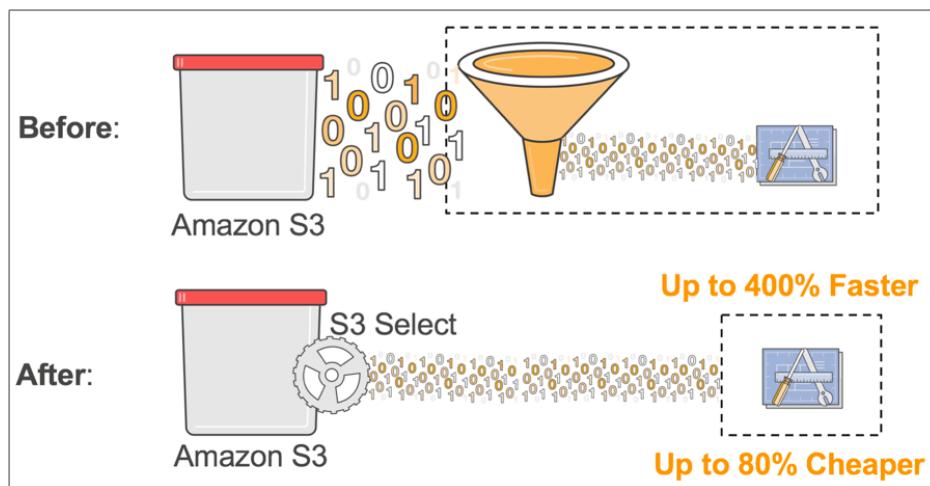


Puede utilizarse para recuperar sólo datos parciales



S3 Select y Glacier Select

- Recupera menos datos mediante SQL realizando un filtrado del lado del servidor
- Puedes filtrar por filas y columnas (simples sentencias SQL)
- Menos transferencia de red, menos coste de CPU en el lado del cliente



<https://aws.amazon.com/blogs/aws/s3-glacier-select/>



Filtrado del lado del servidor

Metadatos de objetos definidos por el usuario de S3 y etiquetas de objetos de S3

- **Metadatos de objetos definidos por el usuario en S3**

- Al subir un objeto, también puedes asignarle metadatos
- Pares clave-valor
- Los nombres de metadatos definidos por el usuario deben comenzar por "x-amz-meta-"
- Amazon S3 almacena las claves de metadatos definidos por el usuario en minúsculas
- Los metadatos pueden recuperarse mientras se recupera el objeto

- **Etiquetas de objetos de S3**

- Pares clave-valor para objetos en Amazon S3
- Útil para permisos de grano fino (sólo acceder a objetos específicos con etiquetas específicas)
- Útiles para fines analíticos (utilizar S3 Analytics para agrupar por etiquetas)

- **No puedes buscar en los metadatos de los objetos ni en las etiquetas de los objetos**

- En su lugar, debes utilizar una BD externa como índice de búsqueda. (search index), como DynamoDB

Metadatos

Clave	Valor
Content-Length	7.5 KB
Content-Type	html
x-amz-meta-origin	paris

Etiquetas

Clave	Valor
Project	Blue
PHI	True



Objeto S3

datos del índice en la Tabla DDB
(se pueden buscar)

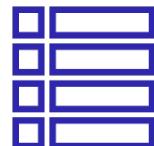


Tabla DynamoDB

Seguridad de Amazon S3

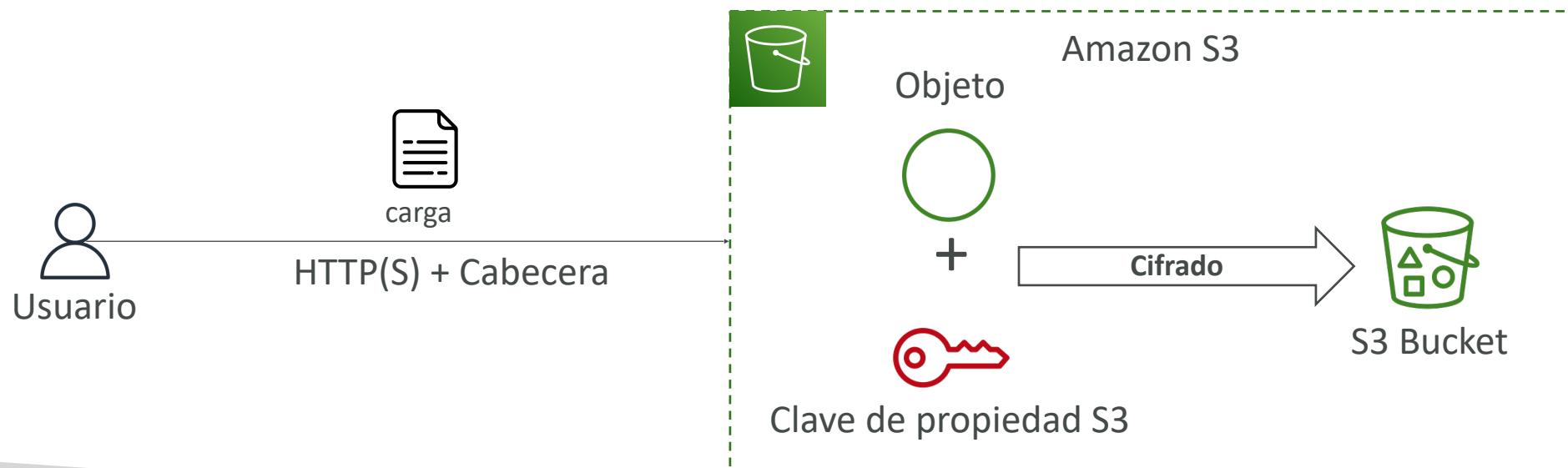


Amazon S3 - Cifrado de objetos

- Puedes cifrar objetos en buckets de S3 utilizando uno de los 4 métodos siguientes
- **Cifrado del lado del servidor (SSE)**
 - **Cifrado del lado del servidor con claves gestionadas por Amazon S3 (SSE-S3) -**
 - Activado por defecto
 - Cifra los objetos de S3 utilizando claves manejadas, gestionadas y propiedad de AWS
 - **Cifrado del lado del servidor con claves KMS almacenadas en AWS KMS (SSE-KMS)**
 - Aprovecha el servicio de administración de claves de AWS (AWS KMS) para gestionar las claves de cifrado
 - **Cifrado del lado del servidor con claves proporcionadas por el cliente (SSE-C)**
 - Cuando quieras gestionar tus propias claves de cifrado
- **Cifrado del lado del cliente**
- Es importante entender cuáles son para cada situación para el examen

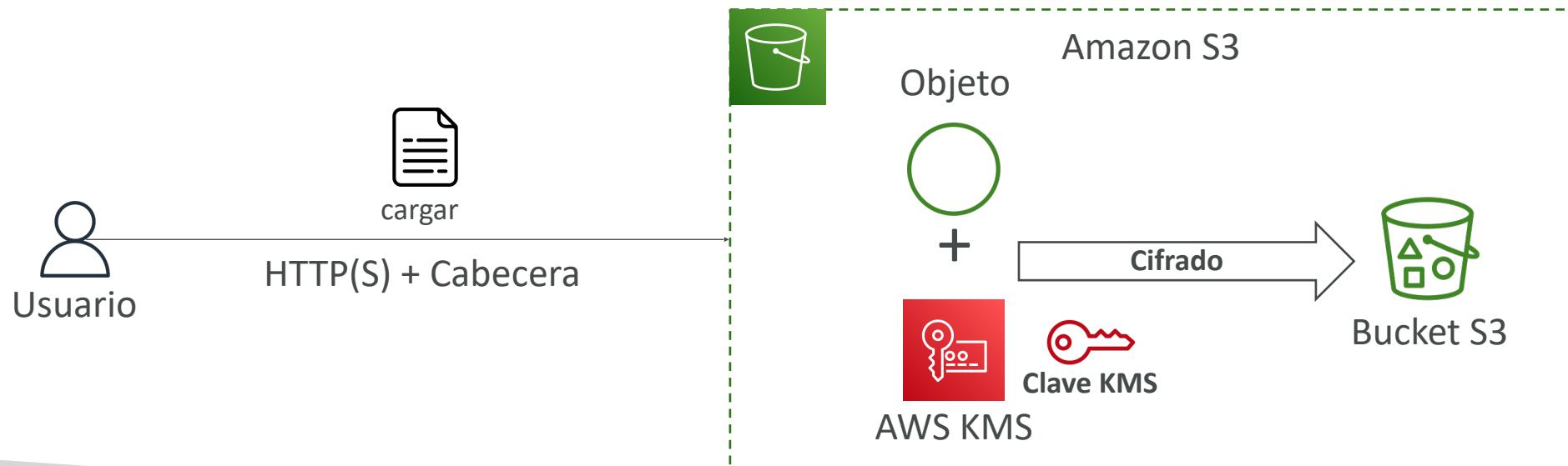
Cifrado de Amazon S3 - SSE-S3

- Cifrado mediante claves manejadas, gestionadas y propiedad de AWS
- El objeto está cifrado en el lado del servidor
- El tipo de cifrado es **AES-256**
- Debe establecerse la cabecera "x-amz-server-side-encryption": "AES256"
- **Activado por defecto para nuevos buckets y nuevos objetos**



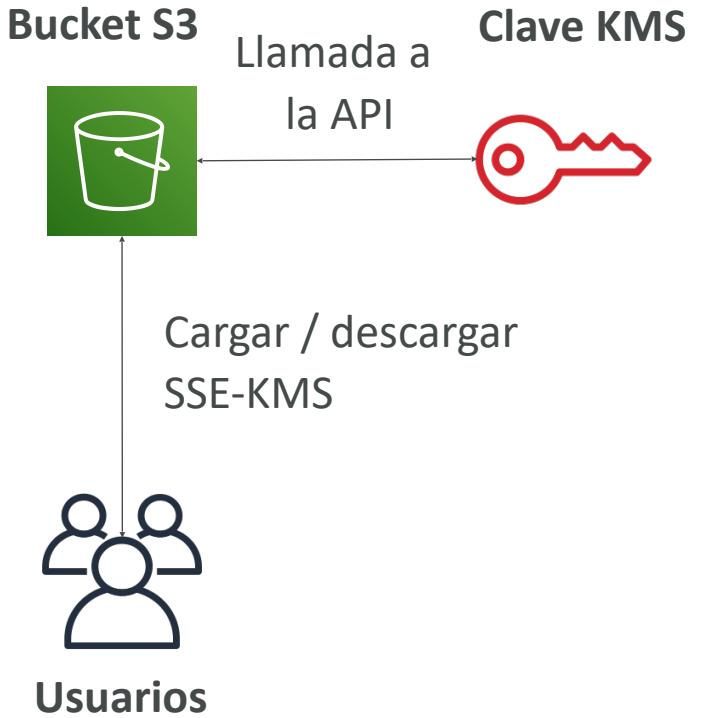
Cifrado de Amazon S3 - SSE-KMS

- Cifrado mediante claves manejadas y gestionadas por AWS KMS (Key Management Service)
- Ventajas del KMS: control del usuario + auditoría del uso de las claves mediante CloudTrail
- El objeto está cifrado en el lado del servidor
- Debes establecer la cabecera "**x-amz-server-side-encryption": "aws:kms"**



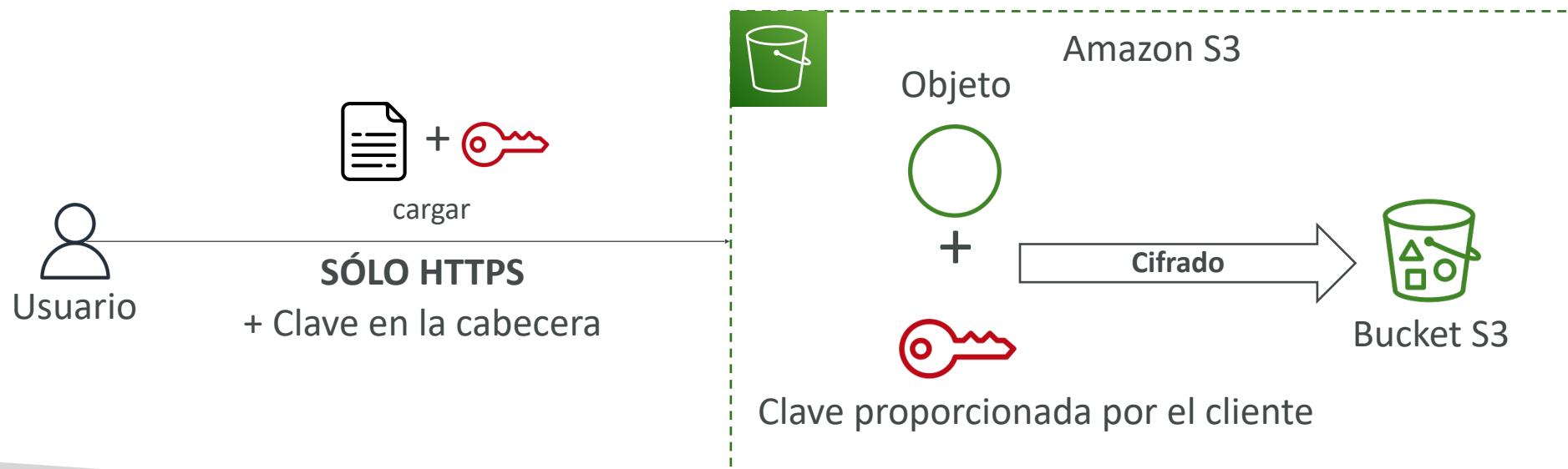
Limitación de SSE-KMS

- Si utilizas SSE-KMS, puede que te afecten los límites del KMS
- Cuando subes, llama a la API KMS **GenerateDataKey**
- Cuando descargas, llama a la API KMS **Decrypt**
- Cuenta para la cuota KMS por segundo (5500, 10000, 30000 req/s según la región)
- Puedes solicitar un aumento de cuota mediante la Consola de Cuotas de Servicio



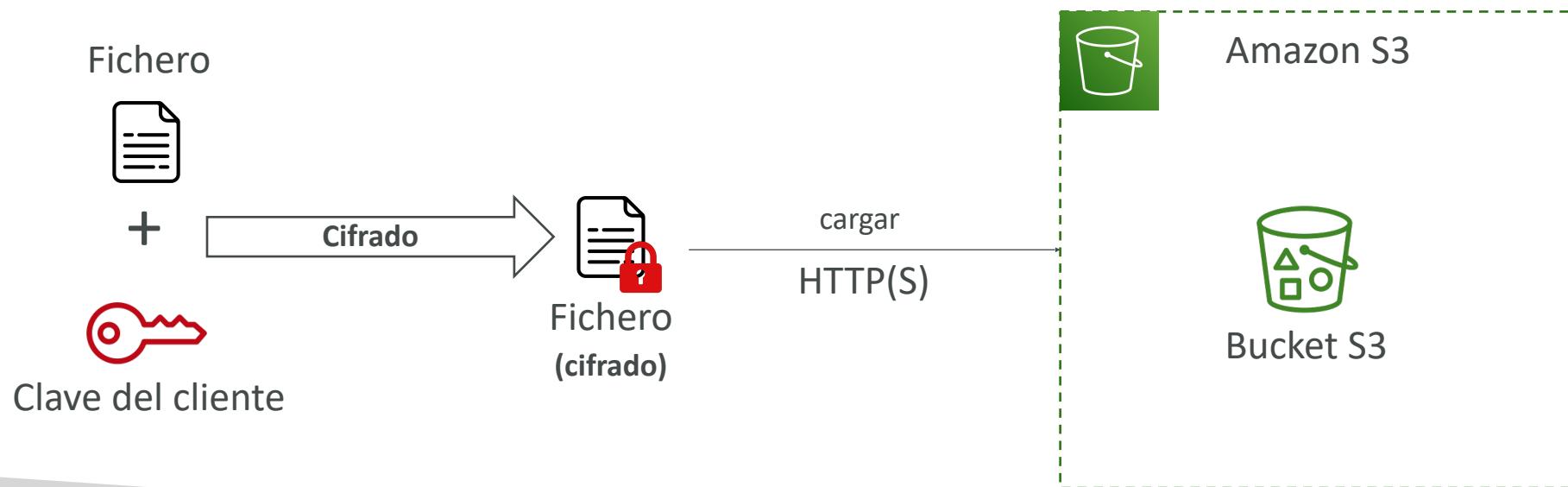
Cifrado de Amazon S3 - SSE-C

- Cifrado del lado del servidor mediante claves totalmente gestionadas por el cliente fuera de AWS
- Amazon S3 **NO** almacena la clave de cifrado que proporcionas
- **Se tiene que utilizar HTTPS**
- La clave de cifrado debe proporcionarse en las cabeceras HTTP, para cada petición HTTP realizada



Cifrado de Amazon S3 - Cifrado del lado del cliente

- Utiliza bibliotecas de clientes como la **biblioteca de cifrado del lado del cliente de Amazon S3**
- Los clientes deben cifrar los datos ellos mismos antes de enviarlos a Amazon S3
- Los clientes deben descifrar los datos ellos mismos al recuperarlos de Amazon S3
- El cliente gestiona completamente las claves y el ciclo de cifrado



Amazon S3 - Cifrado en tránsito (SSL/TLS)

- El cifrado en vuelo también se llama SSL/TLS
- Amazon S3 expone dos endpoints:
 - **HTTP Endpoint** - no cifrado
 - **Endpoint HTTPS** - cifrado en vuelo
- **Se recomienda HTTPS**
- **HTTPS es obligatorio para SSE-C**
- La mayoría de los clientes usarán el endpoint HTTPS por defecto



Amazon S3 - Políticas de cifrado por defecto vs. bucket

- Una forma de "forzar el cifrado" es utilizar una política de bucket y rechazar cualquier llamada a la API para PUT un objeto S3 sin cabeceras de cifrado

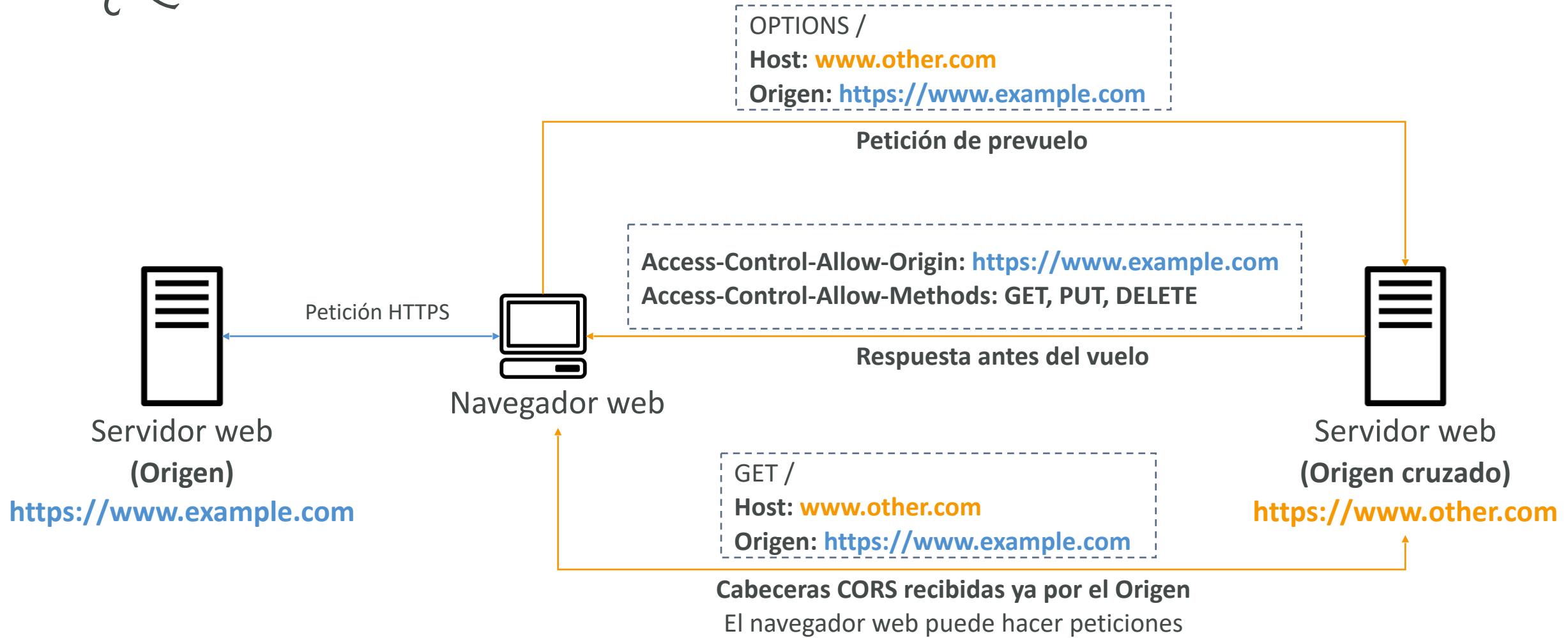
```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Sid": "DenyIncorrectDecryptionHeader",  
            "Effect": "Deny",  
            "Principal": "*",  
            "Action": [ "s3:PutObject" ],  
            "Resource": [ "arn:aws:s3:::examplebucket/*" ],  
            "Condition": {  
                "StringNotEquals": {  
                    "s3:x-amz-server-side-encryption": "AES256"  
                }  
            }  
        }  
    ]  
}  
  
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Sid": "DenyUnencryptedObjectUploads",  
            "Effect": "Deny",  
            "Principal": "*",  
            "Action": [ "s3:PutObject" ],  
            "Resource": [ "arn:aws:s3:::examplebucket/*" ],  
            "Condition": {  
                "Null": {  
                    "s3:x-amz-server-side-encryption": true  
                }  
            }  
        }  
    ]  
}
```

- Otra forma es utilizar la opción de "cifrado por defecto" en S3
- Nota: Las políticas de los buckets se evalúan antes del "cifrado por defecto"

¿Qué es el CORS?

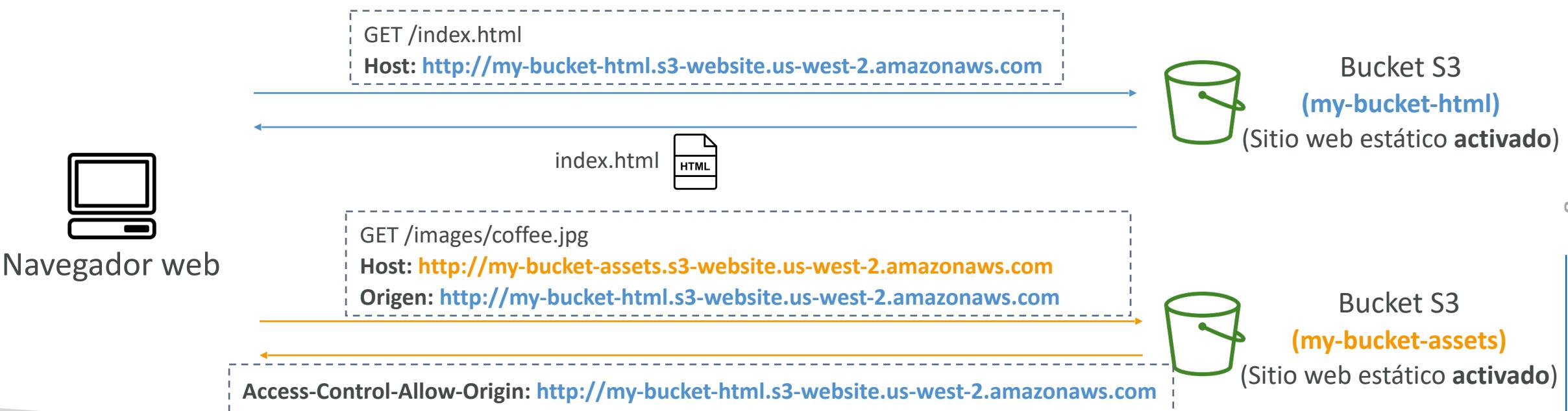
- **Compartir recursos entre orígenes (CORS)**
- **Origen = esquema (protocolo) + host (dominio) + puerto**
 - Ejemplo: <https://www.example.com> (el puerto implícito es 443 para HTTPS, 80 para HTTP)
- Mecanismo **basado en el navegador web** para permitir peticiones a otros orígenes mientras se visita el origen principal
- El mismo origen: <http://example.com/app1> y <http://example.com/app2>
- Diferentes orígenes: <http://www.example.com> y <http://other.example.com>
- Las peticiones no se cumplirán a menos que el otro origen permita las peticiones, utilizando **cabeceras CORS** (ejemplo: **Access-Control-Allow-Origin**)

¿Qué es el CORS?



Amazon S3 – CORS

- Si un cliente hace una petición de origen cruzado en nuestro bucket de S3, tenemos que habilitar las cabeceras CORS correctas
- Es una pregunta de examen muy popular
- Puedes permitir un origen específico o * (todos los orígenes)



Amazon S3 - Eliminación de MFA

- **MFA (Autenticación de Factores Múltiples)**: obliga a los usuarios a generar un código en un dispositivo (normalmente un teléfono móvil o un hardware) antes de realizar operaciones importantes en el S3
- MFA será necesario para:
 - Eliminar permanentemente una versión de un objeto
 - Suspender el control de versiones en el bucket
- MFA no será necesario para:
 - Habilitar el control de versiones
 - Listar las versiones eliminadas
- Para utilizar MFA Delete, **el control de versiones debe estar activado** en el bucket
- **Sólo el propietario del bucket (cuenta root) puede activar/desactivar MFA Delete**



Autenticador de Google



Dispositivo de hardware MFA

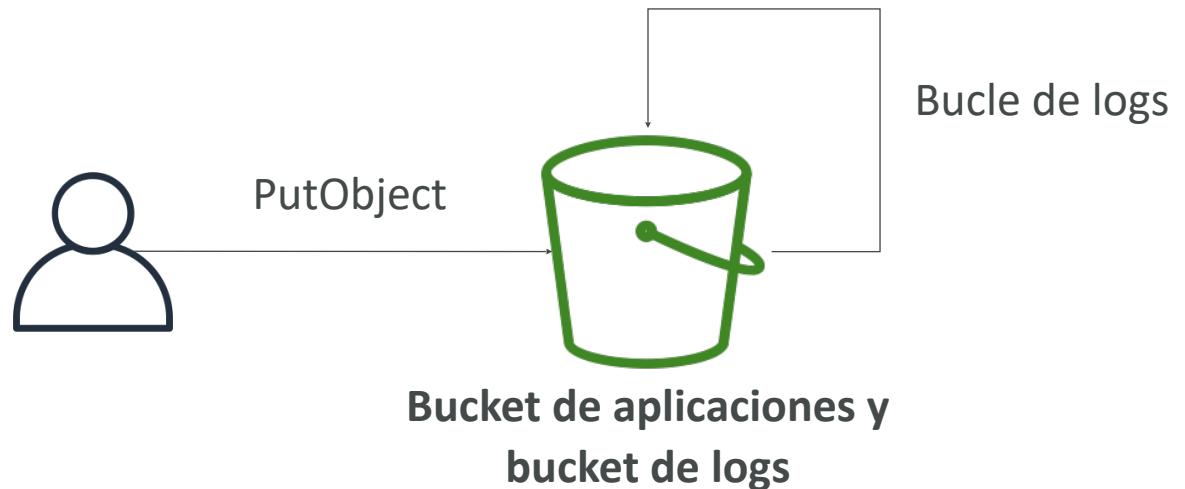
S3 Access Logs

- Para fines de auditoría, es posible que quieras registrar todos los accesos a los buckets de S3
- Cualquier petición realizada a S3, desde cualquier cuenta, autorizada o denegada, se registrará en otro bucket de S3
- Esos datos pueden ser analizados con herramientas de análisis de datos...
- El bucket de logs de destino debe estar en la misma región de AWS
- El formato de los logs está en: <https://docs.aws.amazon.com/AmazonS3/latest/dev/LogFormat.html>



S3 Access Logs: Warning

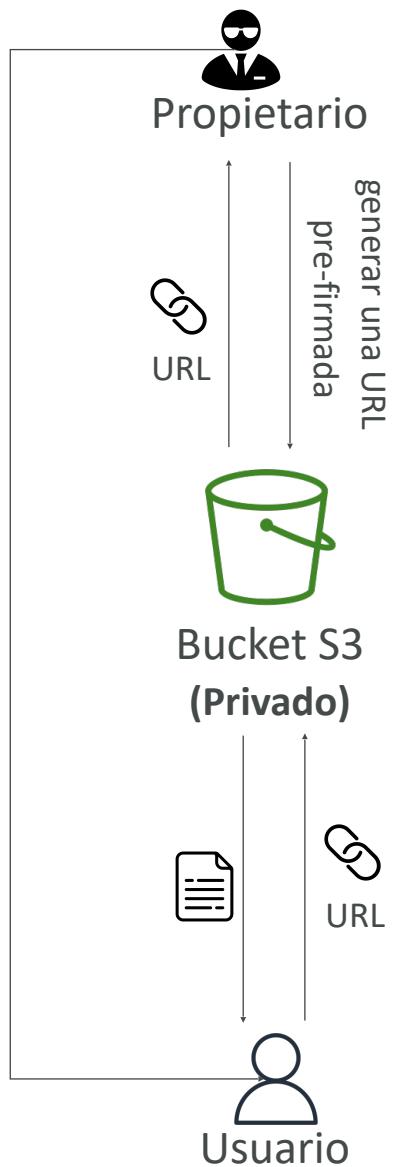
- No configures tu bucket de logs para que sea el bucket monitorizado
- Se creará un bucle de logs, y **tu bucket crecerá exponencialmente**



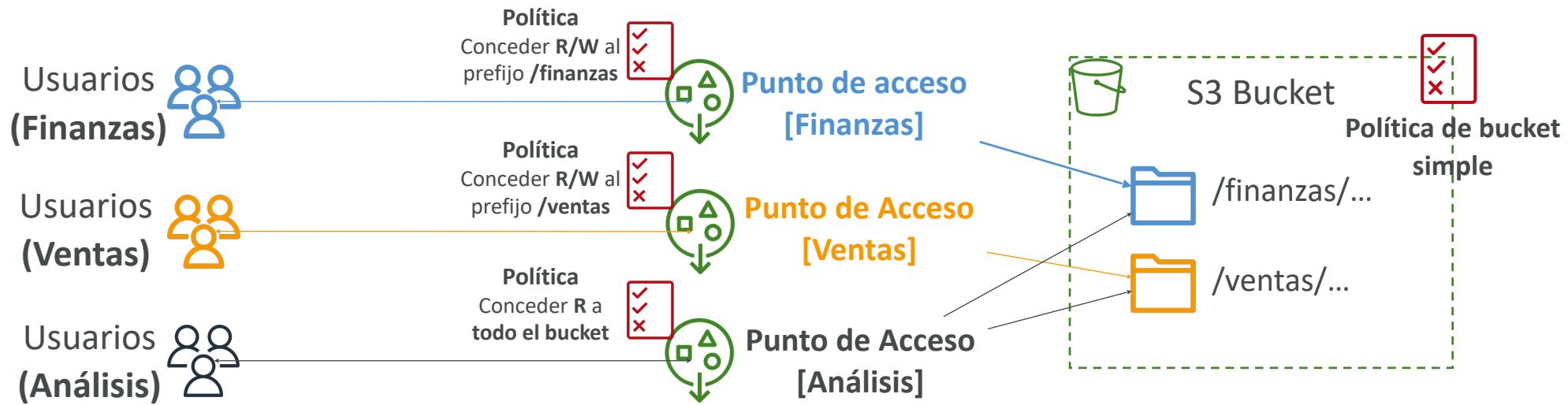
No lo intentes en casa ☺

Amazon S3 - URLs pre-firmadas

- Generar URLs pre-firmadas usando la consola de **S3, la CLI de AWS o el SDK**
- **Expiración de la URL**
 - **Consola S3** - de 1 minuto a 720 minutos (12 horas)
 - **CLI de AWS** - configurar la caducidad con el parámetro --expires-in en segundos (por defecto 3600 segs, máx. 604800 segs ~ 168 horas)
- Los usuarios a los que se les da una URL pre-firmada heredan los permisos del usuario que generó la URL para GET / PUT
- Ejemplos:
 - Permite que sólo los usuarios que han iniciado sesión descarguen un vídeo premium de tu bucket de S3
 - Permitir que una lista cambiante de usuarios descargue archivos generando URLs dinámicamente
 - Permitir temporalmente que un usuario suba un archivo a una ubicación precisa en tu bucket de S3



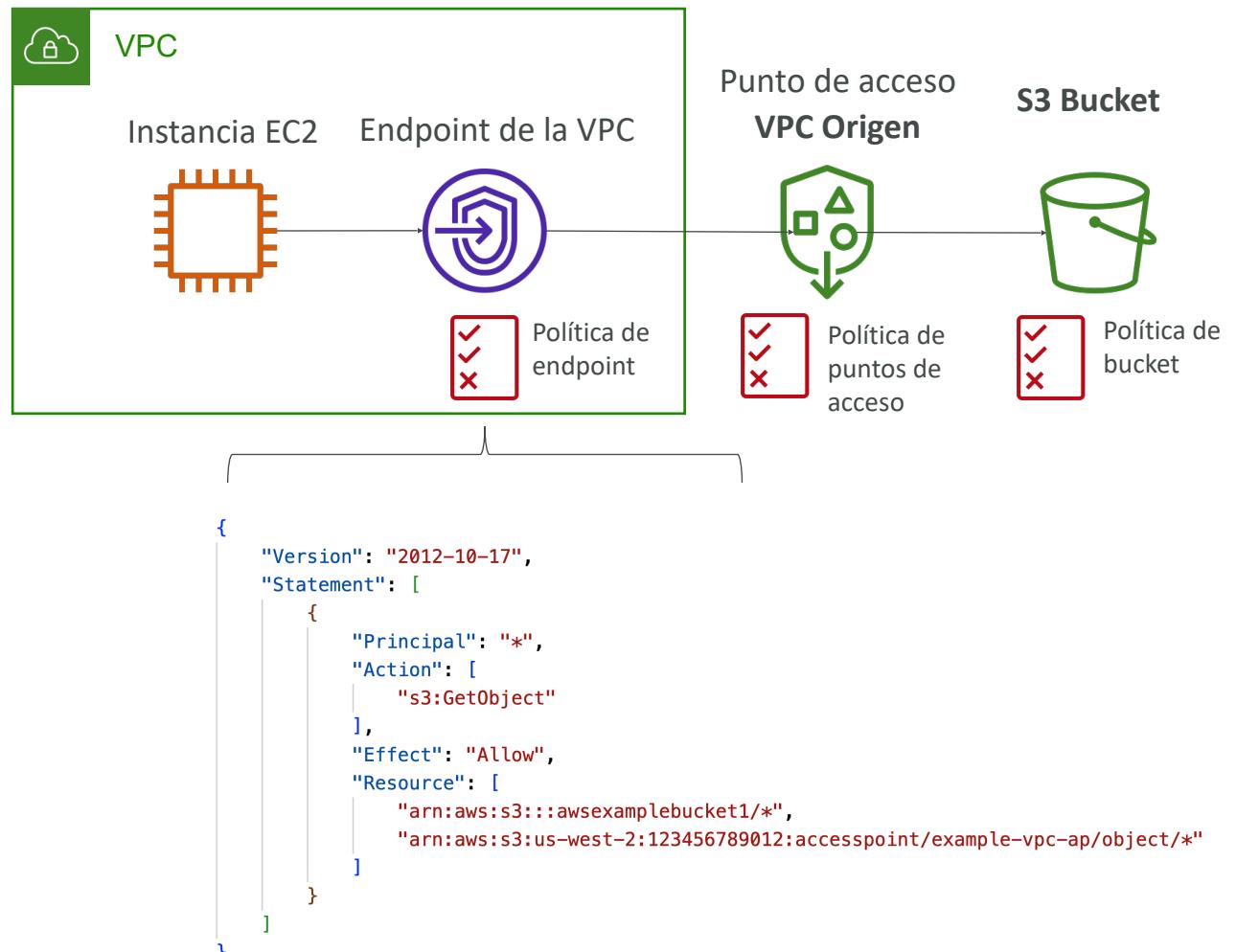
S3 - Puntos de acceso



- Los puntos de acceso simplifican la gestión de la seguridad de los buckets S3
- Cada punto de acceso tiene
 - su propio nombre DNS (Origen Internet u Origen VPC)
 - una política de punto de acceso (similar a la política de bucket) - gestionar la seguridad a escala

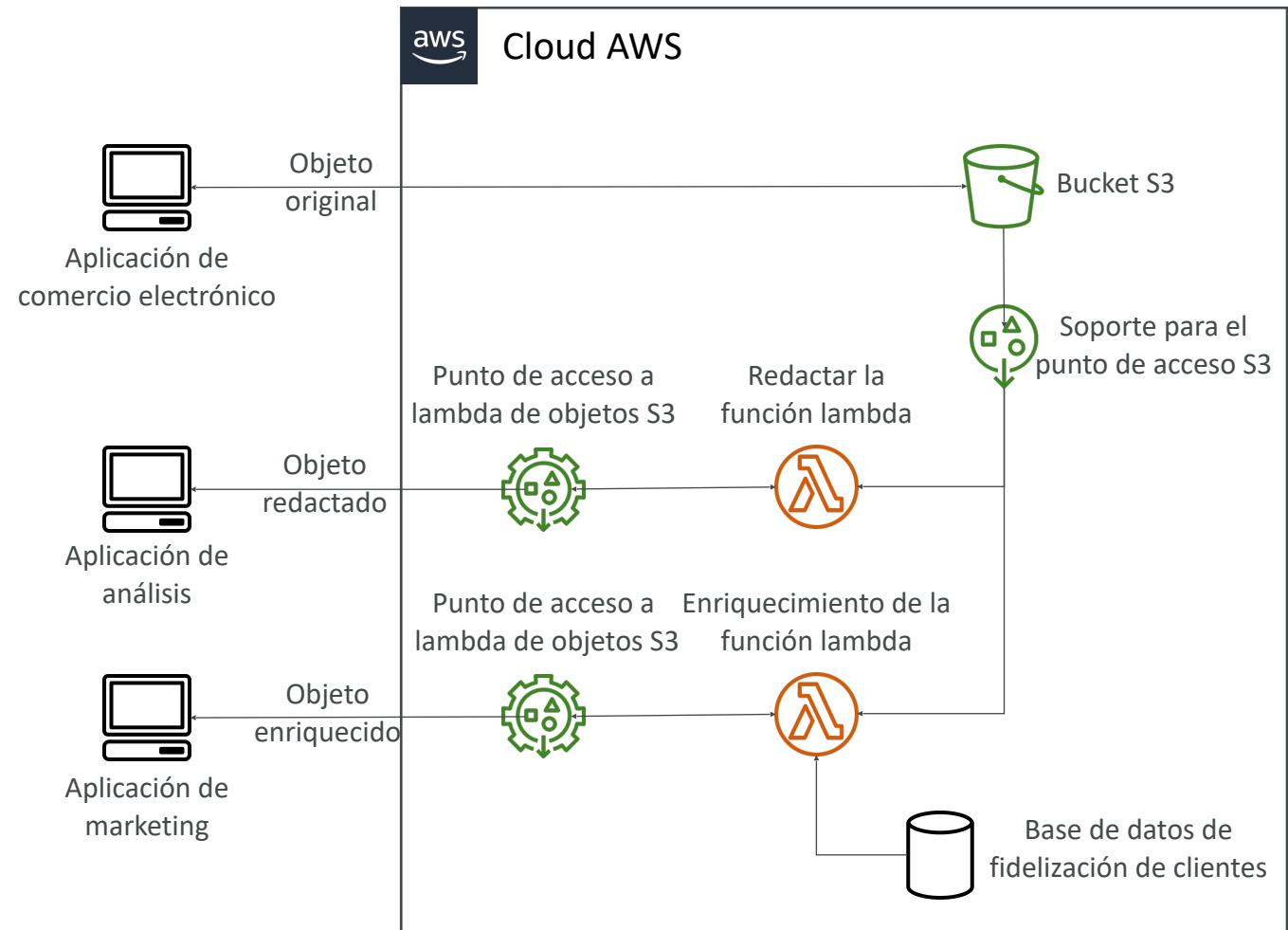
S3 - Puntos de acceso - VPC Origen

- Podemos definir el punto de acceso para que sólo sea accesible desde dentro de la VPC
- Debes crear un VPC Endpoint para acceder al punto de acceso (Gateway o Interface Endpoint)
- La política del endpoint de la VPC debe permitir el acceso al bucket de destino y al punto de acceso



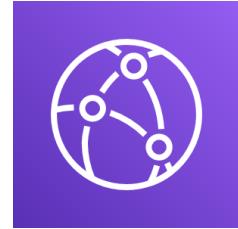
Objeto S3 Lambda

- Utiliza las Funciones Lambda de AWS para modificar el objeto antes de que lo recupere la aplicación que lo llama
- Sólo se necesita un bucket de S3, sobre el que creamos **puntos de acceso de S3** y **puntos de acceso de S3 Object Lambda**.
- Casos de uso:
 - Redactar información de identificación personal para entornos de análisis o de no producción.
 - Convertir entre formatos de datos, como convertir XML a JSON.
 - Redimensionar y poner marcas de agua a las imágenes sobre la marcha utilizando detalles específicos de la persona que llama, como el usuario que solicitó el objeto.

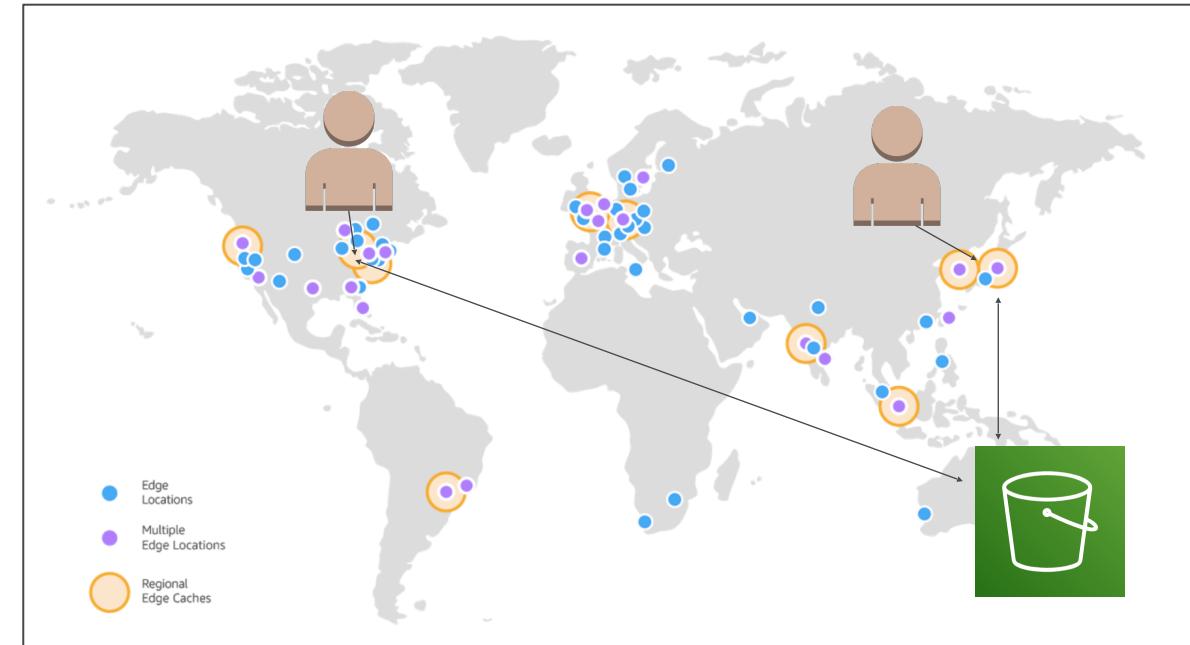


CloudFront

AWS CloudFront



- Red de entrega de contenidos (CDN)
- **Mejora el rendimiento de lectura, el contenido se almacena en caché en edge location**
- Mejora la experiencia de los usuarios
- 216 puntos de presencia a nivel mundial (ubicaciones edge)
- **Protección DDoS, integración con Shield, AWS Web Application Firewall**



Fuente: <https://aws.amazon.com/cloudfront/features/?nc=sn&loc=2>

CloudFront - Orígenes

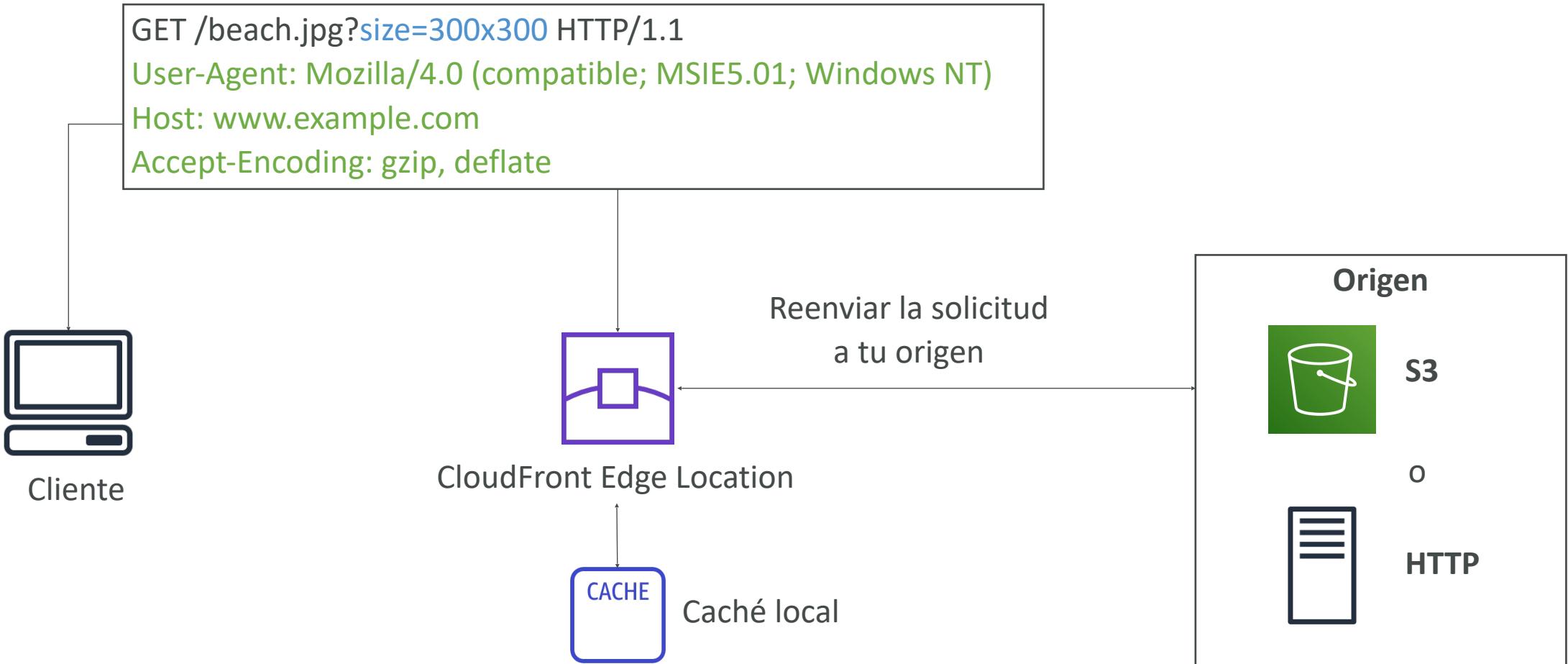
- **Bucket S3**

- Para distribuir archivos y almacenarlos en caché en el borde
- Seguridad mejorada con CloudFront Origin Access Identity (OAI)
- OAC sustituye a Origin Access Identity (OAI)
- CloudFront puede utilizarse como entrada (para subir archivos a S3)

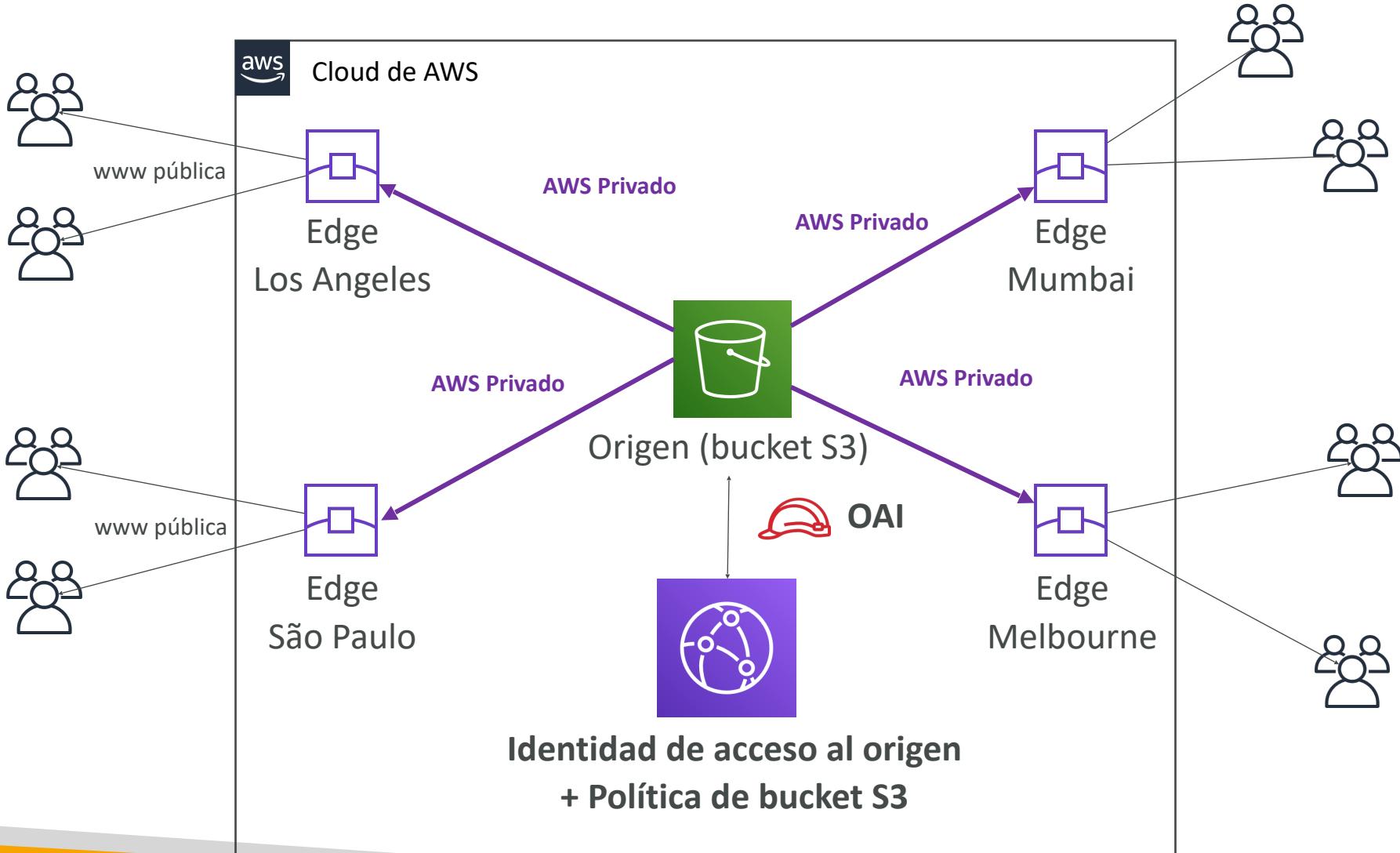
- **Origen personalizado (HTTP)**

- Application Load Balancer
- Instancia EC2
- Sitio web de S3 (primero debes habilitar el bucket como sitio web estático de S3)
- Cualquier backend HTTP que quieras

CloudFront a alto nivel



CloudFront - S3 como origen

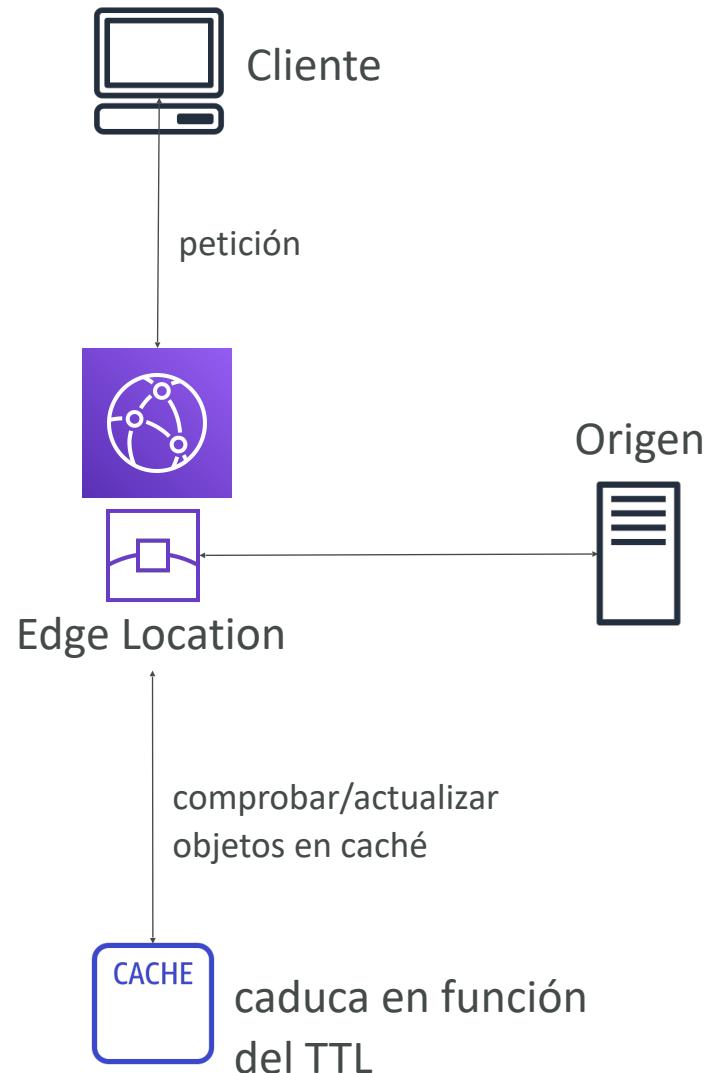


CloudFront vs S3 Cross Region Replication (CRR)

- CloudFront:
 - Red Global Edge
 - Los archivos se almacenan en caché durante un TTL (quizás un día)
 - **Es ideal para contenidos estáticos que deben estar disponibles en todas partes**
- S3 Cross Region Replication (CRR):
 - Debe configurarse para cada región en la que quieras que se produzca la replicación
 - Los archivos se actualizan casi en tiempo real
 - Sólo lectura
 - **Ideal para contenidos dinámicos que deben estar disponibles con baja latencia en pocas regiones**

Almacenamiento en caché CloudFront

- La caché vive en cada **Edge Location** de CloudFront
- CloudFront identifica cada objeto de la caché mediante la **clave de caché** (véase la siguiente diapositiva)
- Quieres maximizar el ratio del golpe de caché (Cache Hit) para minimizar las peticiones al origen
- Puedes invalidar parte de la caché utilizando la API **CreateInvalidation**



¿Qué es la clave de caché de CloudFront?

- Un identificador único para cada objeto de la caché
- Por defecto, consiste en el **hostname + la parte del recurso de la URL**
- Si tienes una aplicación que sirve contenidos que varían en función del usuario, dispositivo, idioma, ubicación...
- Puedes añadir otros elementos (cabeceras HTTP, cookies, cadenas de consulta) a la clave de caché mediante las **políticas de caché de CloudFront**



Políticas de CloudFront - Política de caché

- Caché basada en:
 - **Cabeceras HTTP:** Ninguna - Lista blanca
 - **Cookies:** Ninguna - Lista blanca - Incluir todas/Excepto - Todas
 - **Strings de consulta:** Ninguna - Lista blanca - Incluir Todas/Excepto - Todas
- Controla el TTL (de 0 segundos a 1 año), lo puede establecer el origen mediante la cabecera **Cache-Control** o la cabecera **de caducidad...**
- Crea tu propia política o utiliza políticas gestionadas predefinidas
- **Todas las cabeceras HTTP, cookies y strings de consulta que incluyas en la clave de caché se incluyen automáticamente en las peticiones de origen**

CloudFront Caching - Política de caché

Cabeceras HTTP

GET /blogs/myblog.html HTTP/1.1

Host: mywebsite.com

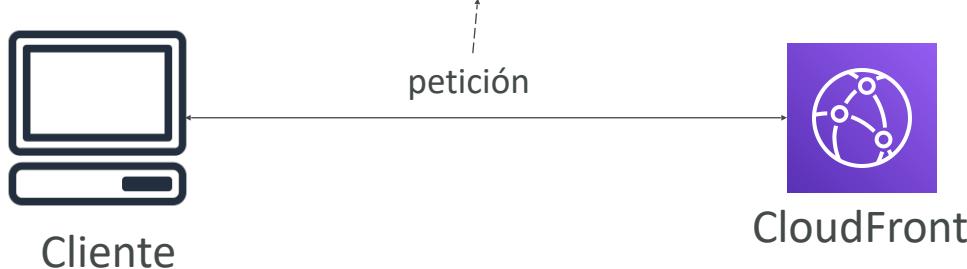
User-Agent: Mozilla/5.0 (Mac OS X 10_15_2....)

Date: Tue, 28 Jan 2021 17:01:57 GMT

Authorization: SAPISIDHASH fdd00ecee39fe....

Keep-Alive: 300

Language: fr-fr



- **Ninguna:**

- No incluir ninguna cabecera en la clave caché (excepto por defecto)
- Las cabeceras no se reenvían (excepto por defecto)
- Mejor rendimiento de la caché
- Las cabeceras especificadas también se reenvían al Origen

- **Whitelist (Lista blanca):**

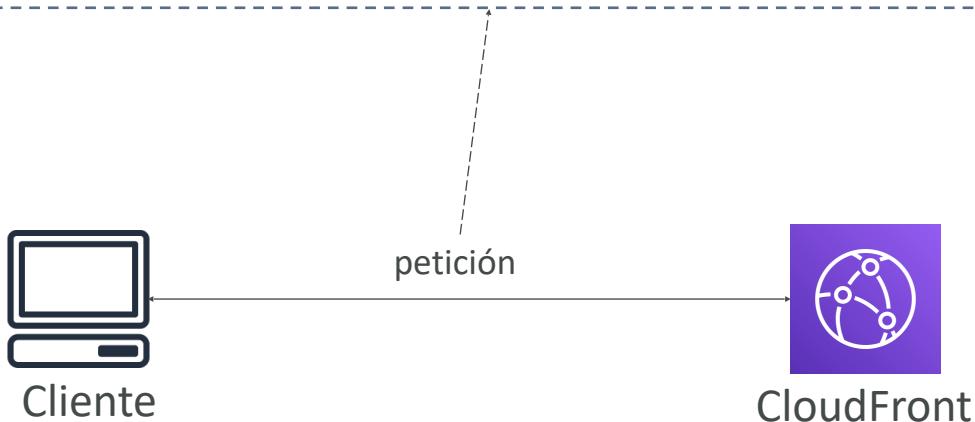
- sólo **cabeceras especificadas** incluida en la clave de caché
- Las cabeceras especificadas también se reenvían al origen

Caché CloudFront - Política de caché

Strings de consulta

GET /image/cat.jpg?border=red&**size=large** HTTP/1.1

...



- **Ninguna**

- No incluir ninguna cadena de consulta en la clave de caché
- Las strings de consulta no se reenvían

- **Lista blanca**

- Sólo se incluyen en la clave de caché las strings de consulta especificadas
- Sólo se reenvían las strings de consulta especificadas

- **Incluir todo excepto**

- Incluir todas las strings de consulta en la clave de caché excepto la lista especificada
- Se reenvían todas las strings de consulta excepto la lista especificada

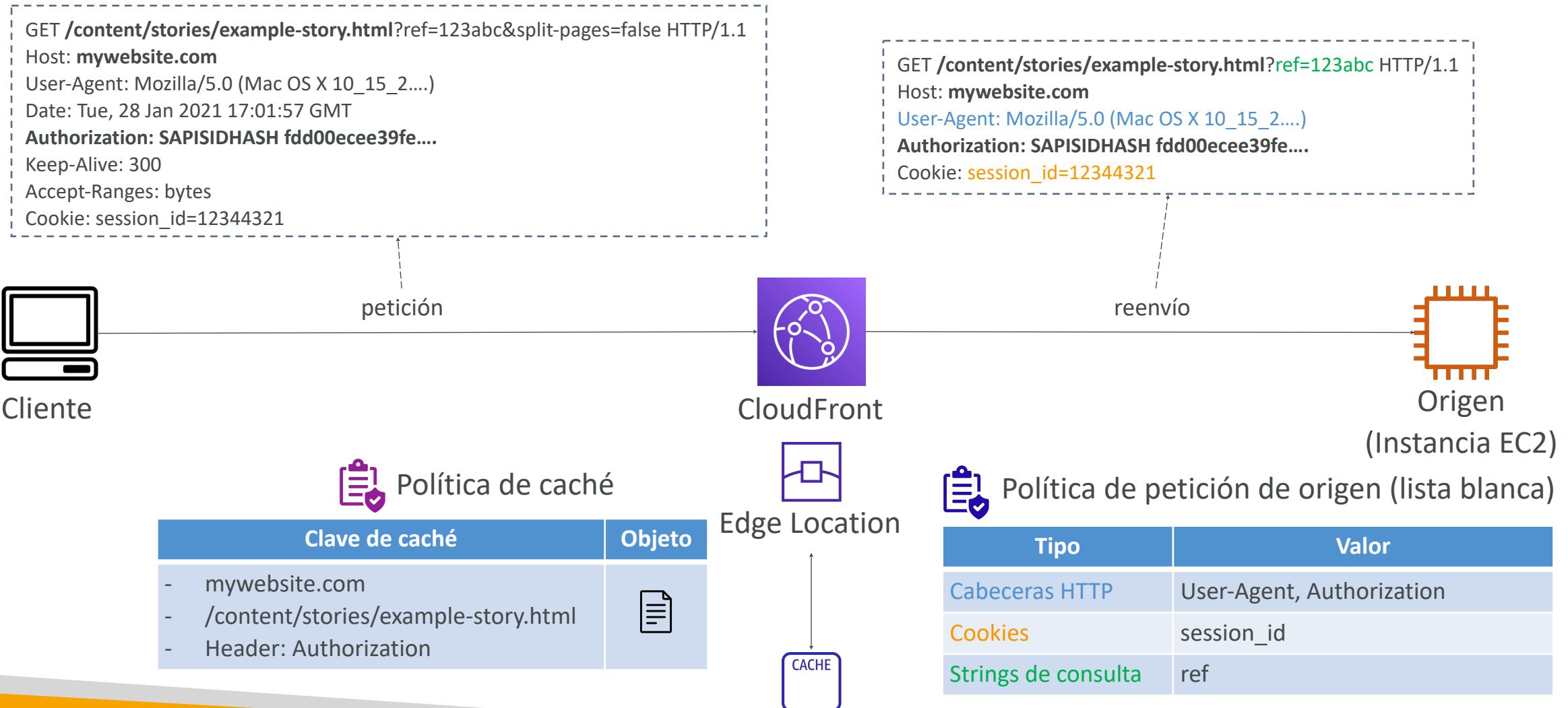
- **Todas**

- Incluir todas las strings de consulta en la clave caché
- Se reenvían todas las strings de consulta
- Peor rendimiento de la caché

Políticas de CloudFront - Política de petición de origen

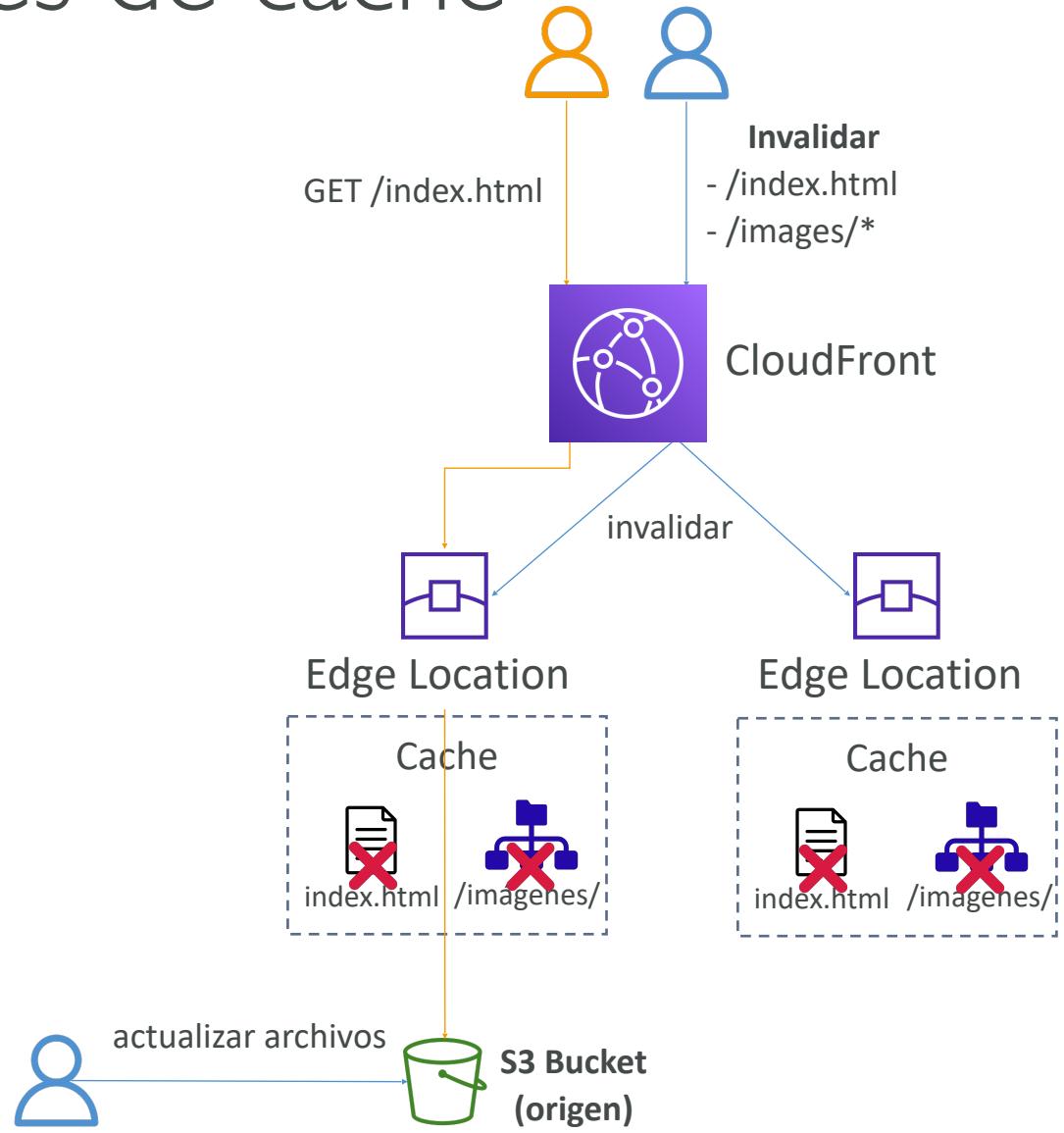
- Especifica los valores que quieras incluir en las peticiones de origen **sin incluirlos en la clave de caché (sin contenido duplicado en caché)**
- Puedes incluir
 - **Cabeceras HTTP:** Ninguna - Lista blanca - Todas las opciones de cabeceras del “viewer”
 - **Cookies:** Ninguna - Lista blanca - Todas
 - **Strings de consulta:** Ninguna - Lista blanca - Todas
- Posibilidad de añadir cabeceras HTTP de CloudFront y cabeceras personalizadas a una petición de origen que no se incluyeron en la petición del visor
- Crea tu propia política o utiliza Políticas Gestiónadas Predefinidas

Política de caché vs. Política de petición de origen



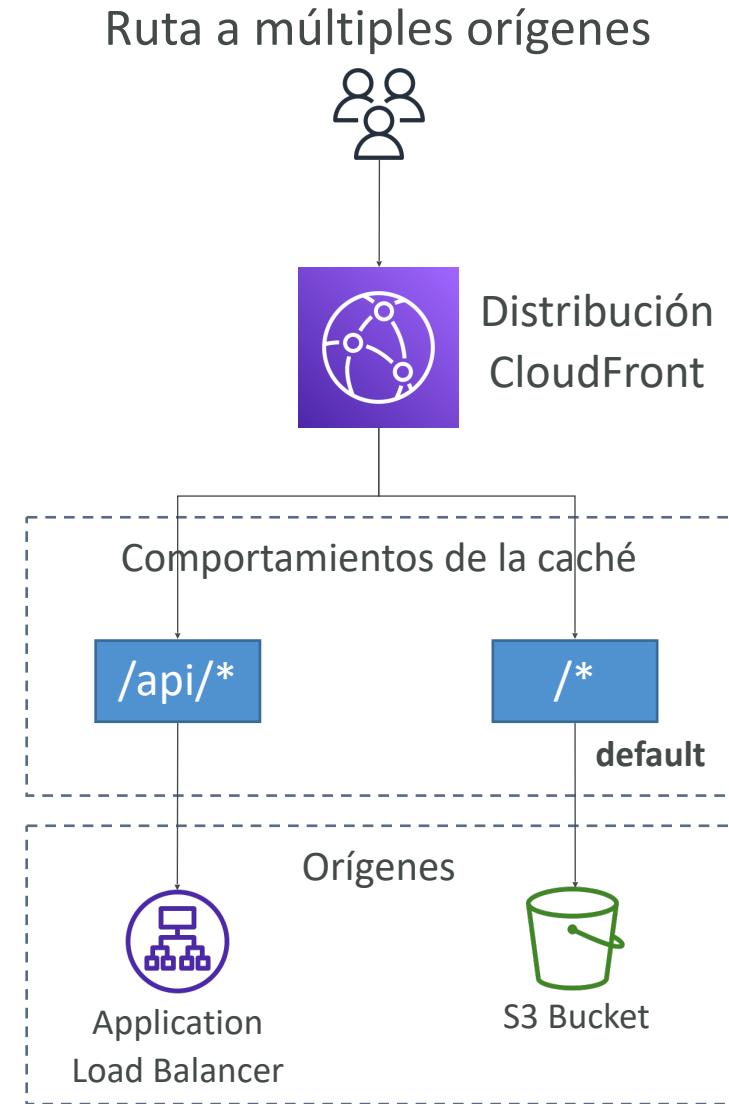
CloudFront - Invalidaciones de caché

- En caso de que actualices el origen del back-end, CloudFront no lo sabe y sólo obtendrá el contenido refrescado cuando el TTL haya expirado
- Sin embargo, puedes forzar una actualización total o parcial de la caché (obviando así el TTL) realizando una **Invalidación de CloudFront**
- Puedes invalidar todos los archivos (*) o una ruta especial (/imágenes/*)



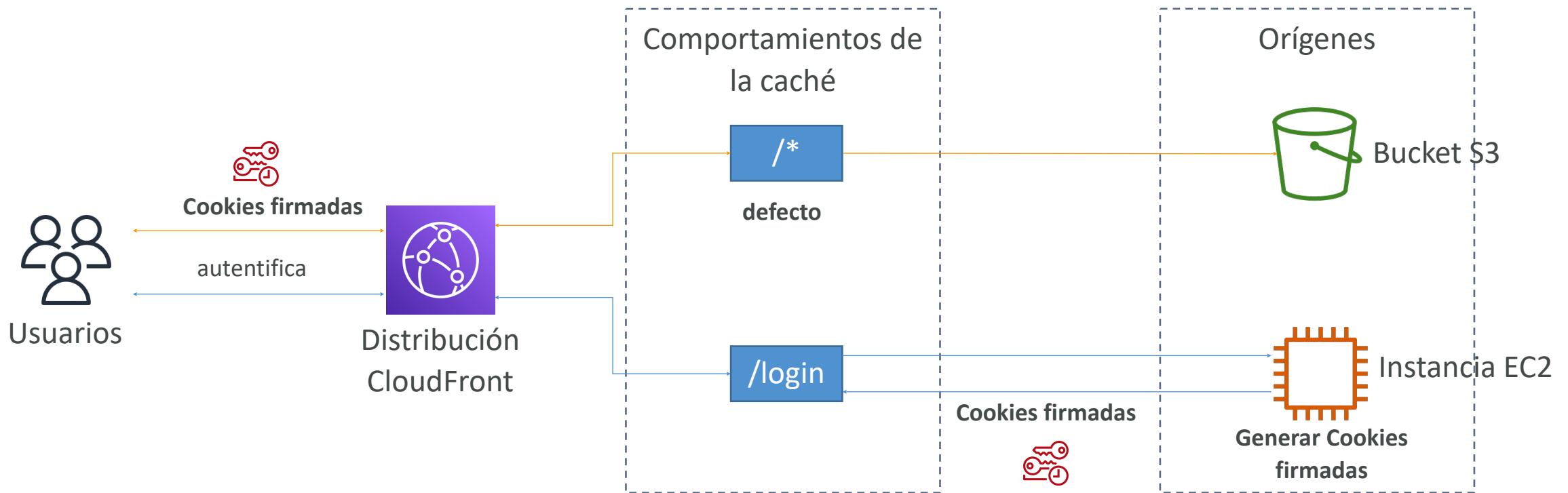
CloudFront - Comportamientos de la caché

- Configurar diferentes ajustes para un determinado patrón de ruta URL
- Ejemplo: un comportamiento de caché específico para archivos **images/*.jpg** en tu servidor web de origen
- Enrutar a diferentes tipos de orígenes/grupos de orígenes en función del tipo de contenido o del patrón de ruta
 - /images/*
 - /api/*
 - /* (comportamiento por defecto de la caché)
- Al añadir comportamientos de caché adicionales, el comportamiento de caché por defecto siempre es el último en procesarse **y siempre es /***



CloudFront - Comportamientos de la caché

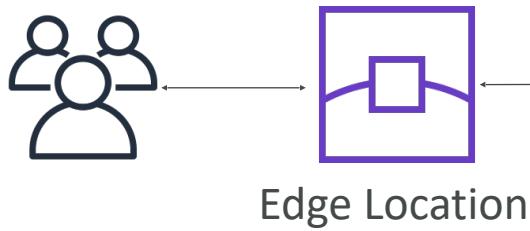
Página de inicio de sesión



CloudFront - Maximiza las visitas a la caché separando las distribuciones estáticas y dinámicas

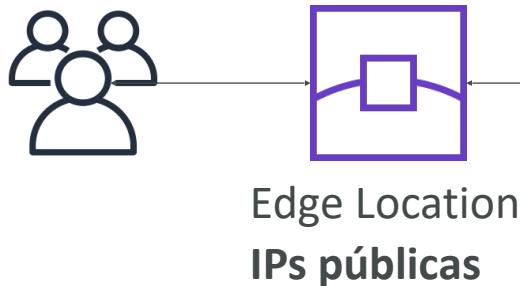


CloudFront - ALB o EC2 como origen



Permitir IP pública de Edge Locations

<http://d7uri8nf7uskq.cloudfront.net/tools/list-cloudfront-ips>



Permitir la IP pública
de Edge Locations



Permitir Grupo de
Seguridad del
Load Balancer



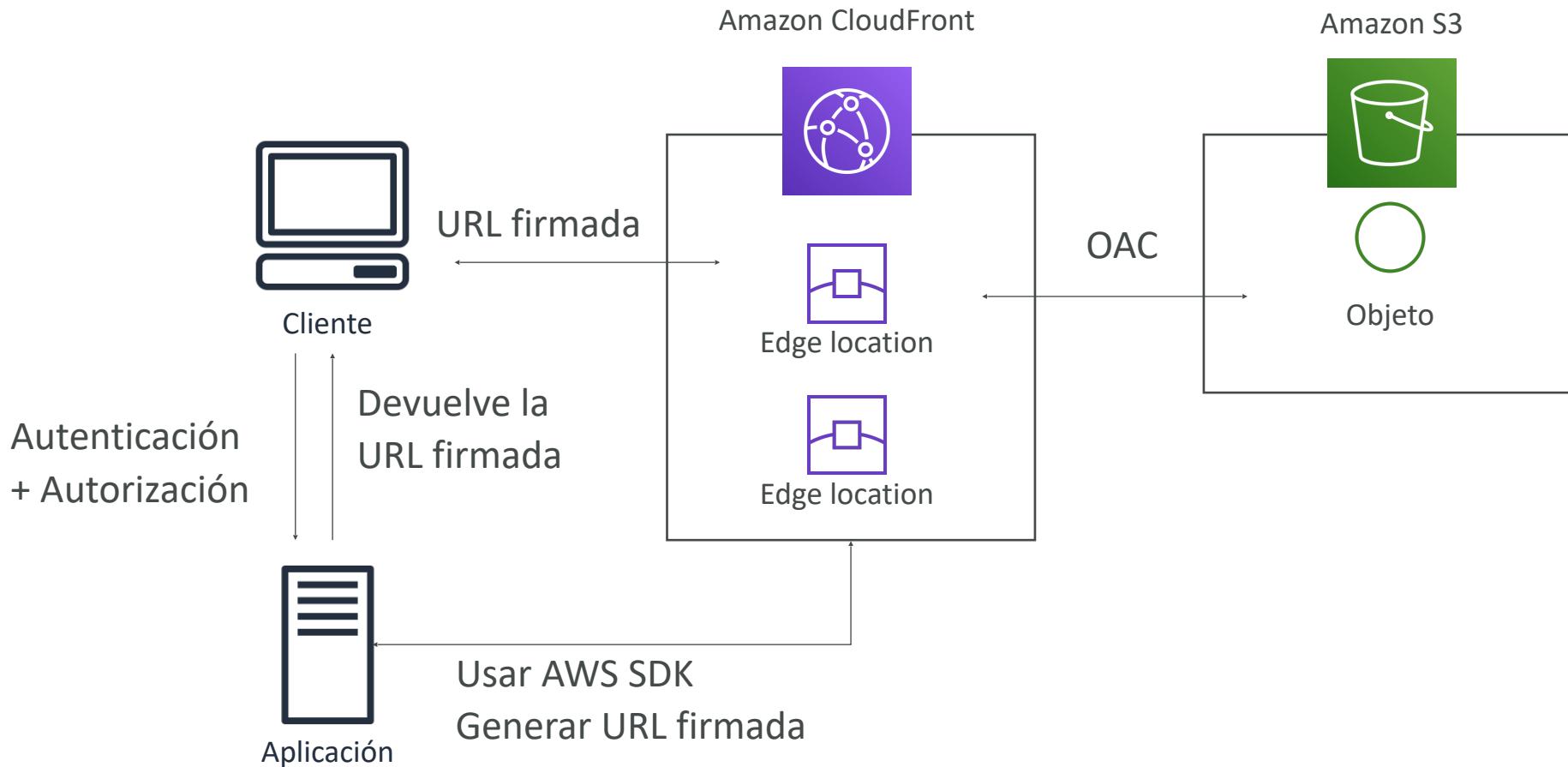
Restricción geográfica de CloudFront

- Puedes restringir quién puede acceder a tu distribución
 - **Lista de permitidos:** Permite que tus usuarios accedan a tu contenido sólo si están en uno de los países de una lista de países aprobados.
 - **Lista de bloqueo:** Evita que tus usuarios accedan a tu contenido si se encuentran en uno de los países de la lista de países prohibidos.
- El "país" se determina utilizando una base de datos Geo-IP de terceros
- Caso de uso: Leyes de derechos de autor para controlar el acceso a los contenidos

URL firmada de CloudFront / Cookies firmadas

- Quieres distribuir contenido compartido de pago a usuarios premium de todo el mundo
- Podemos utilizar URL firmadas de CloudFront / Cookies. Adjuntamos una política con:
 - Incluye la caducidad de la URL
 - Incluye rangos de IP desde los que acceder a los datos
 - Firmantes de confianza (qué cuentas de AWS pueden crear URL firmadas)
- ¿Durante cuánto tiempo debe ser válida la URL?
 - Contenido compartido (película, música): que sea breve (unos minutos)
 - Contenido privado (privado para el usuario): puedes hacer que dure años
- URL firmada = acceso a archivos individuales (una URL firmada por archivo)
- Cookies firmadas = acceso a varios archivos (una cookie firmada para muchos archivos)

Diagrama de URL firmada por CloudFront



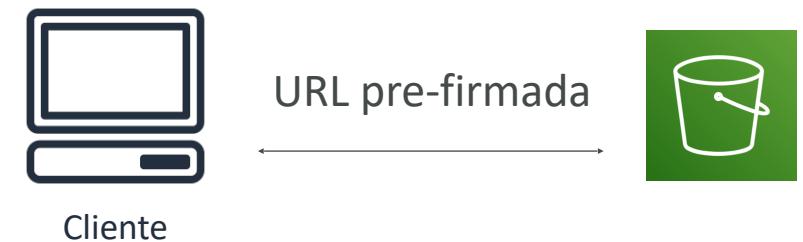
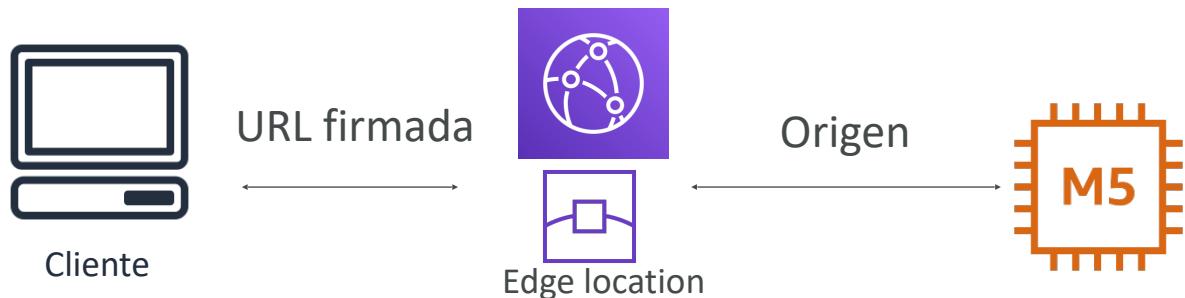
URL firmada de CloudFront vs URL prefirmada de S3

- **URL firmada por CloudFront:**

- Permite el acceso a una ruta, sin importar el origen
- Par de claves para toda la cuenta, sólo el root puede gestionarlo
- Puede filtrar por IP, ruta, fecha, caducidad
- Puede aprovechar las funciones de caché

- **URL pre-firmada de S3:**

- Emite una petición como la persona que ha pre-firmado la URL
- Utiliza la clave IAM de la entidad de seguridad IAM firmante
- Tiempo de vida limitado



Proceso de URL firmada de CloudFront

- **Dos tipos de firmantes:**
 - Un grupo de claves de confianza (recomendado)
 - Puede aprovechar las API para crear y rotar claves (e IAM para la seguridad de las API)
 - Una cuenta de AWS que contenga un par de claves de CloudFront
 - Necesitas gestionar las claves utilizando **la cuenta root y la consola de AWS**
 - No se recomienda porque no deberías utilizar la cuenta root para esto
- En tu distribución de CloudFront, crea uno o más **grupos de claves de confianza**
- Genera tu propia clave pública / privada
 - La clave privada la utilizan tus aplicaciones (por ejemplo, EC2) para firmar URLs
 - La clave pública (cargada) es utilizada por CloudFront para verificar URLs

CloudFront - Precios

- Las Edge Locations de CloudFront están por todo el mundo
- El coste de los datos por Edge Locations varía

Per Month	United States, Mexico, & Canada	Europe & Israel	South Africa, Kenya, & Middle East	South America	Japan	Australia & New Zealand	Hong Kong, Philippines, Singapore, South Korea, Taiwan, & Thailand	India
First 10TB	\$0.085	\$0.085	\$0.110	\$0.110	\$0.114	\$0.114	\$0.140	\$0.170
Next 40TB	\$0.080	\$0.080	\$0.105	\$0.105	\$0.089	\$0.098	\$0.135	\$0.130
Next 100TB	\$0.060	\$0.060	\$0.090	\$0.090	\$0.086	\$0.094	\$0.120	\$0.110
Next 350TB	\$0.040	\$0.040	\$0.080	\$0.080	\$0.084	\$0.092	\$0.100	\$0.100
Next 524TB	\$0.030	\$0.030	\$0.060	\$0.060	\$0.080	\$0.090	\$0.080	\$0.100
Next 4PB	\$0.025	\$0.025	\$0.050	\$0.050	\$0.070	\$0.085	\$0.070	\$0.100
Over 5PB	\$0.020	\$0.020	\$0.040	\$0.040	\$0.060	\$0.080	\$0.060	\$0.100

bajo  más alto

CloudFront - Clases de precios

- Puedes reducir el número de Edge Locations para **reducir los costes**
- Tres clases de precios:
 1. Clase de precio Todos: todas las regiones - mejor rendimiento
 2. Clase de precio 200: la mayoría de las regiones, pero excluye las regiones más caras
 3. Clase de precio 100: sólo las regiones menos caras

Edge Locations Included Within	United States, Mexico, & Canada	Europe & Israel	South Africa, Kenya, & Middle East	South America	Japan	Australia & New Zealand	Hong Kong, Philippines, Singapore, South Korea, Taiwan, & Thailand	India
Price Class All	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Price Class 200	Yes	Yes	Yes	x	Yes	x	Yes	Yes
Price Class 100	Yes	Yes	x	x	x	x	x	x

CloudFront - Clase de precio

Precios Clase: 100



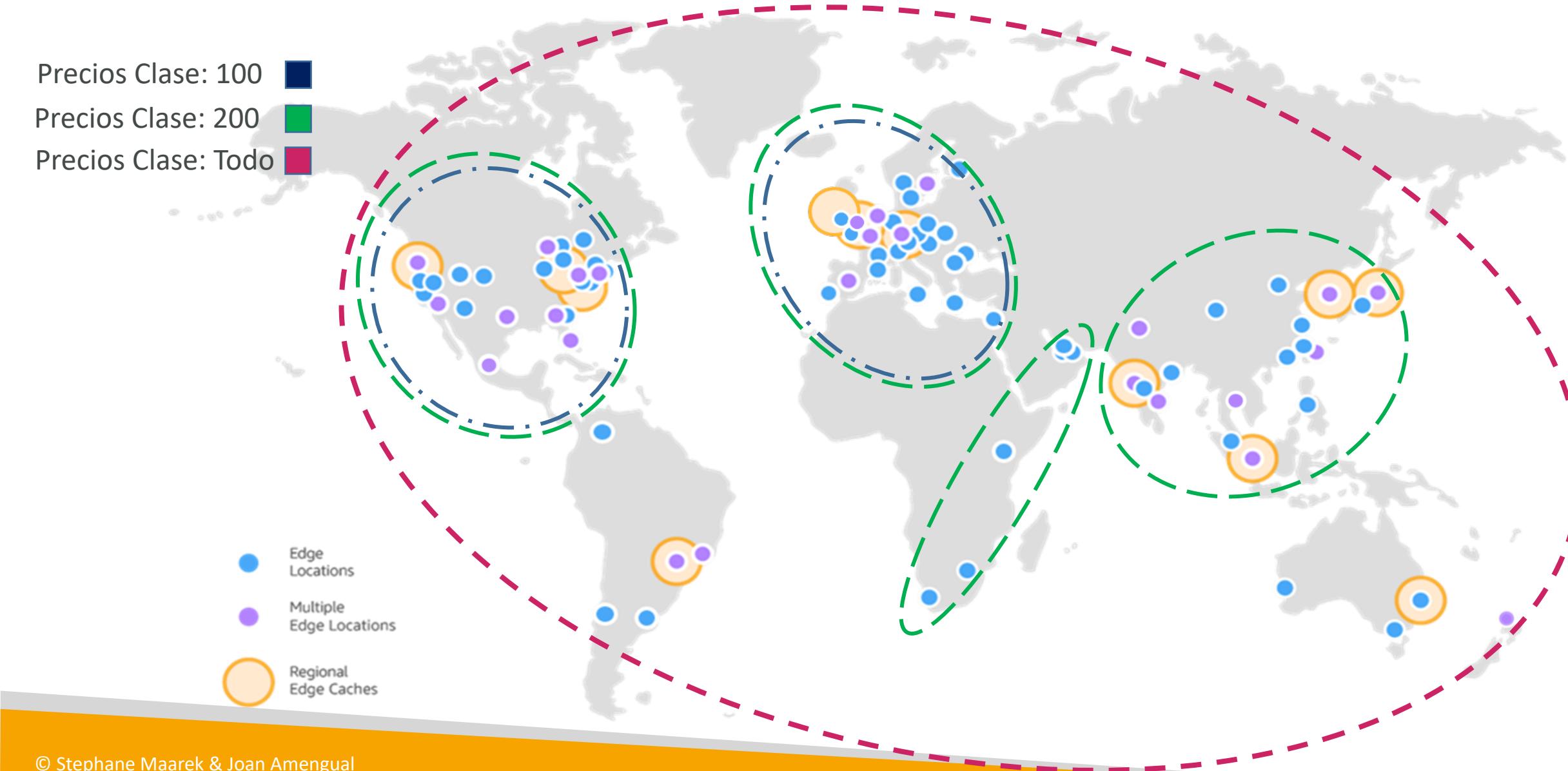
Precios Clase: 200



Precios Clase: Todo

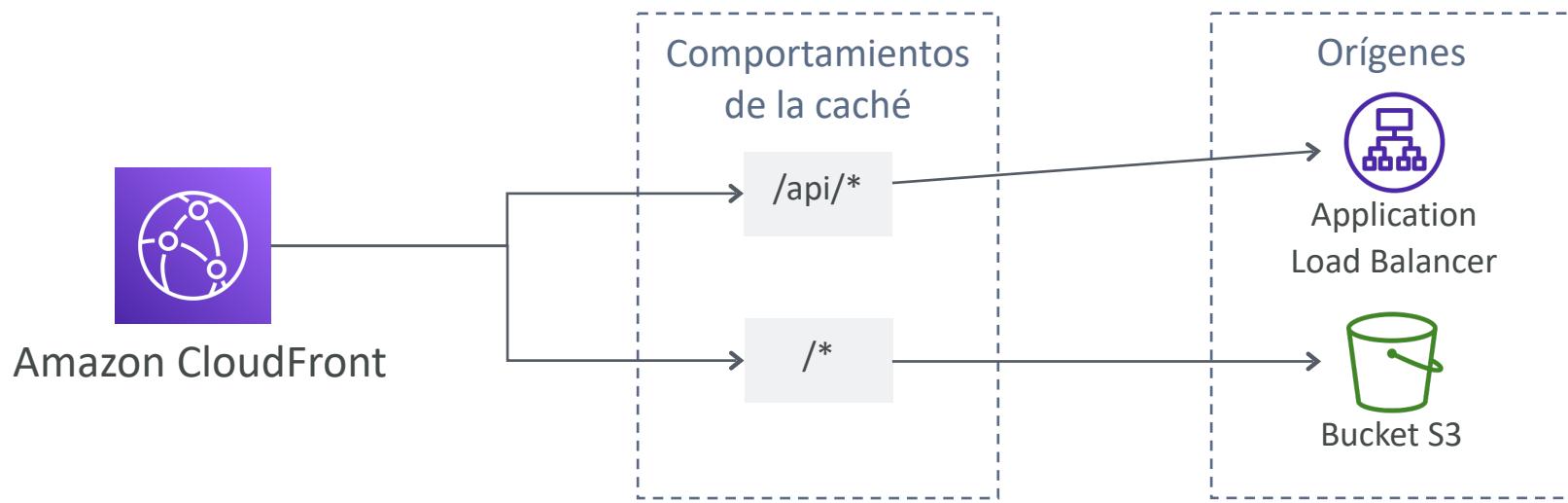


- Edge Locations
- Multiple Edge Locations
- Regional Edge Caches



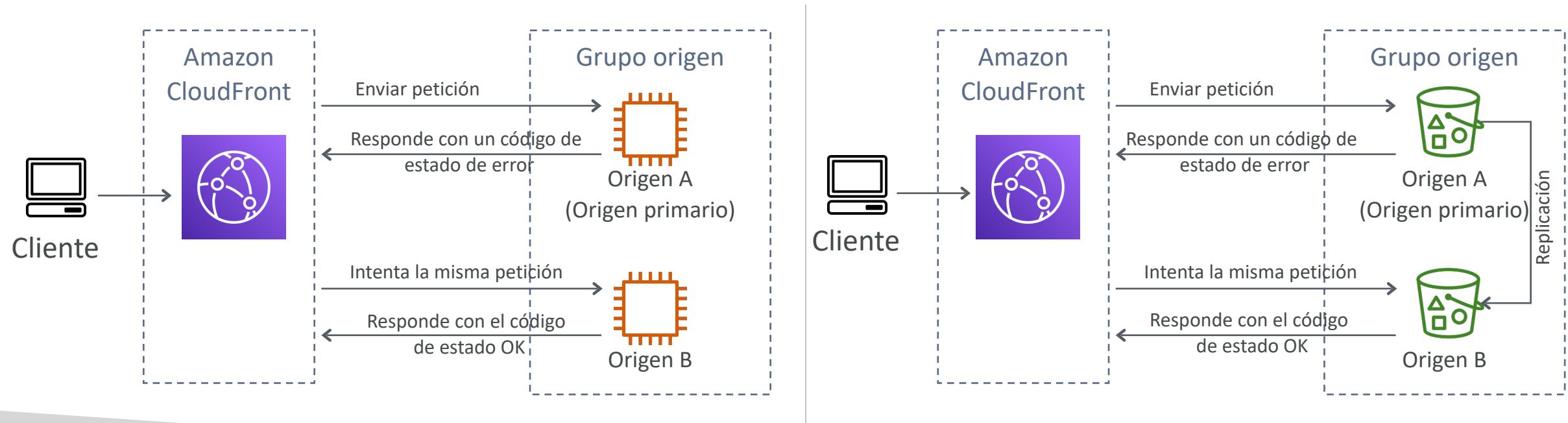
CloudFront - Origen múltiple

- Enrutar a distintos tipos de orígenes en función del tipo de contenido
- Basado en el patrón de ruta:
 - /images/*
 - /api/*
 - /*



CloudFront - Grupos de orígenes

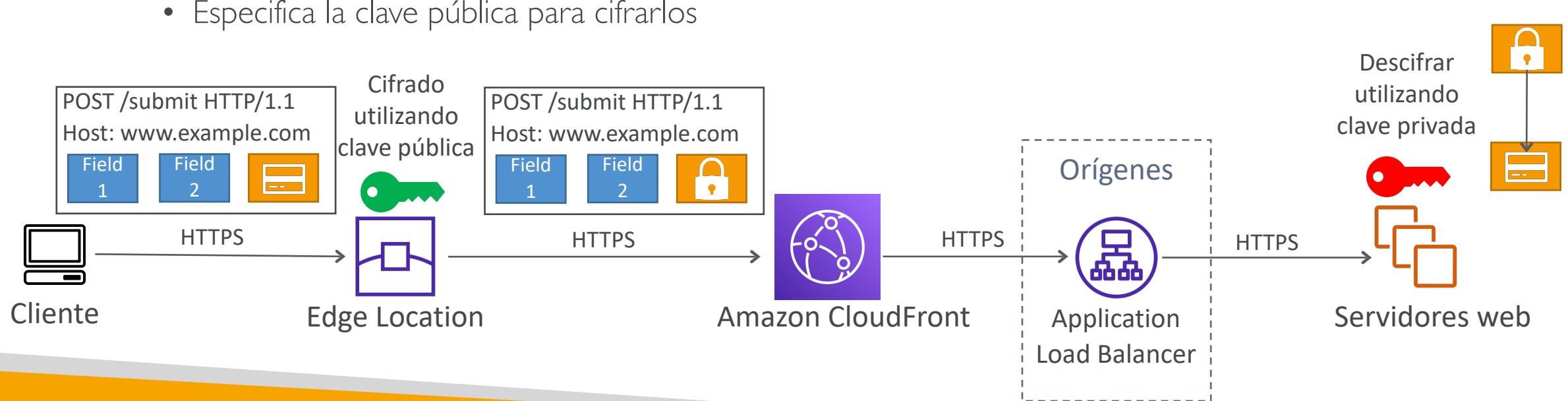
- Para aumentar la alta disponibilidad y hacer commutación por error
- Grupo de orígenes: un origen primario y otro secundario
- Si falla el origen primario, se utiliza el segundo



S3 + CloudFront - Alta disponibilidad a nivel regional

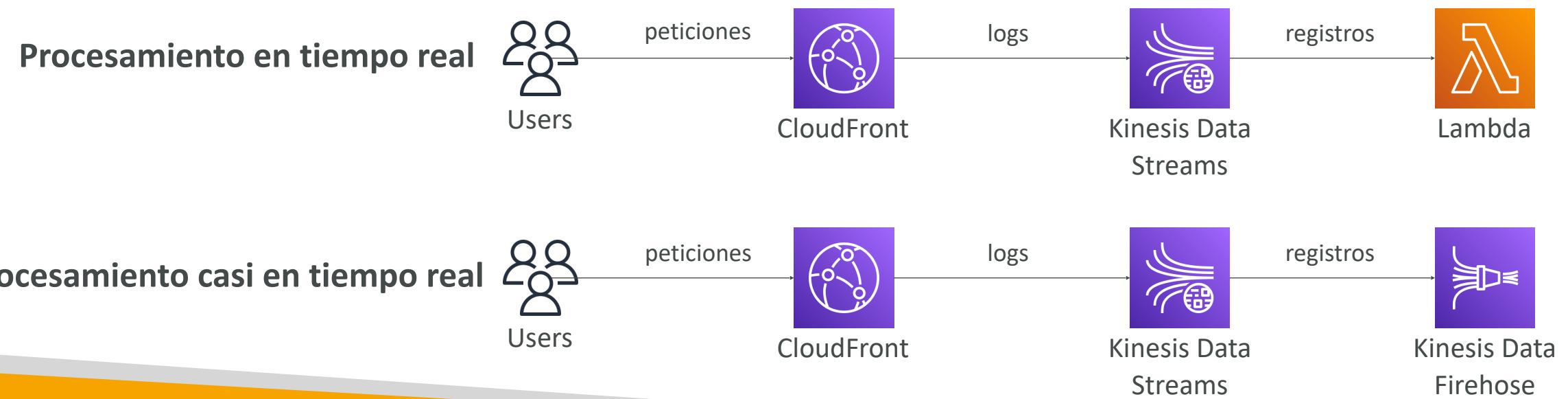
CloudFront - Cifrado a nivel de campo

- Protege la información sensible del usuario mediante la aplicación stack
- Añade una capa adicional de seguridad junto con HTTPS
- Información sensible cifrada en el extremo cercano al usuario
- Utiliza cifrado asimétrico
- Utilización:
 - Especifica el conjunto de campos de las peticiones POST que quieras que se cifren (hasta 10 campos)
 - Especifica la clave pública para cifrarlos

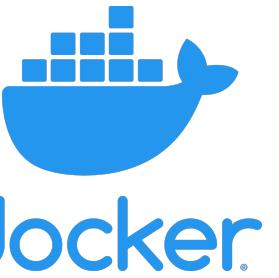


CloudFront - Logs en tiempo real

- Obtén en tiempo real las peticiones recibidas por CloudFront enviadas a Kinesis Data Streams
- Supervisa, analiza y toma medidas en función del rendimiento de la entrega de contenidos
- Te permite elegir:
 - **Tasa de muestreo** - porcentaje de peticiones para las que quieras recibir
 - Campos específicos y comportamientos de caché específicos (patrones de ruta)



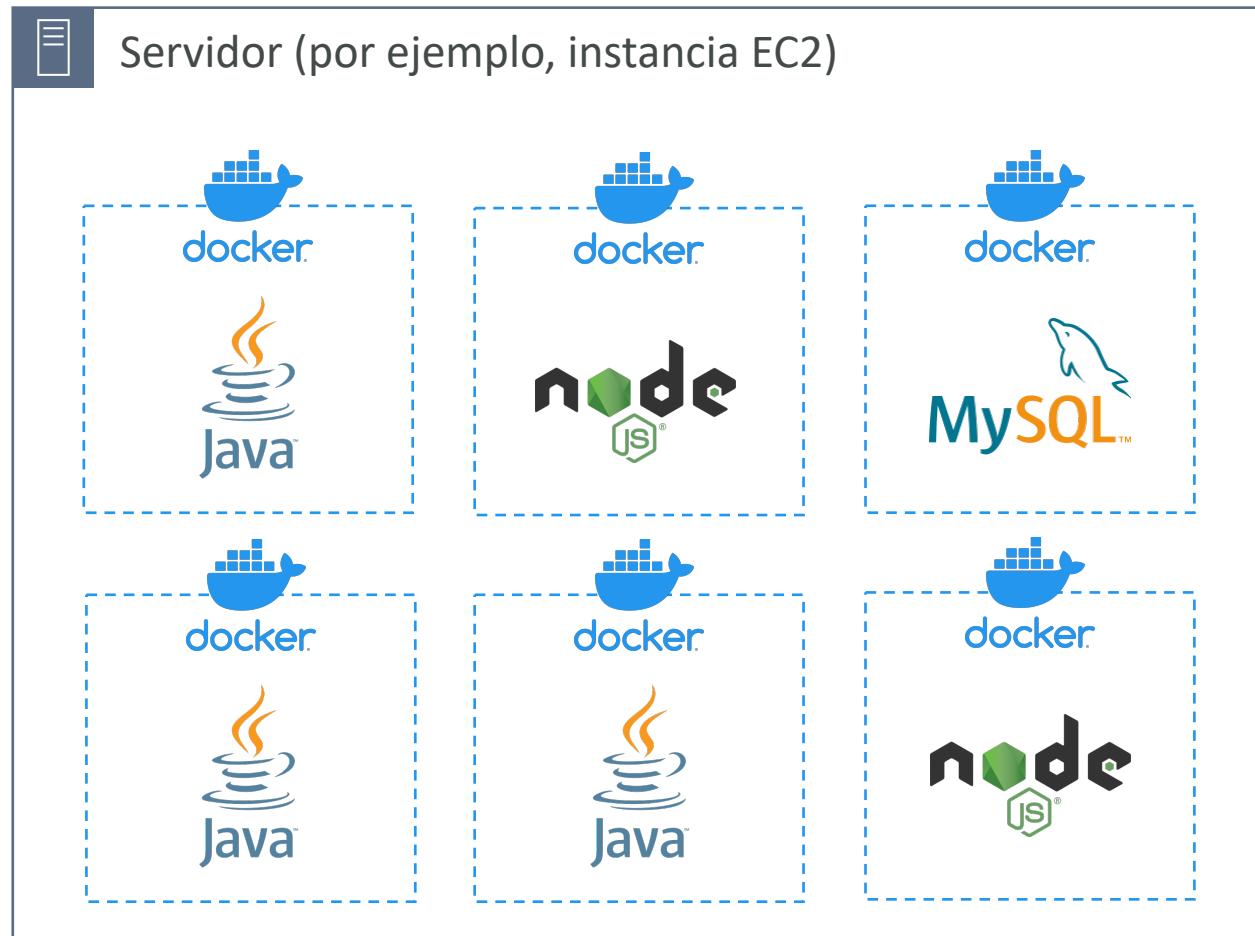
Contenedores



¿Qué es Docker?

- Docker es una plataforma de desarrollo de software para desplegar aplicaciones.
- Las aplicaciones se empaquetan en **contenedores** que pueden ejecutarse en cualquier sistema operativo.
- Las aplicaciones se ejecutan igual, independientemente de dónde se ejecuten.
 - Cualquier máquina
 - Sin problemas de compatibilidad
 - Comportamiento predecible
 - Menos trabajo
 - Más fácil de mantener e implantar
 - Funciona con cualquier lenguaje, sistema operativo y tecnología
- Casos de uso: arquitectura de microservicios, aplicaciones lift-and-shift de on-premises a la nube de AWS, ...

Docker en un Sistema Operativo

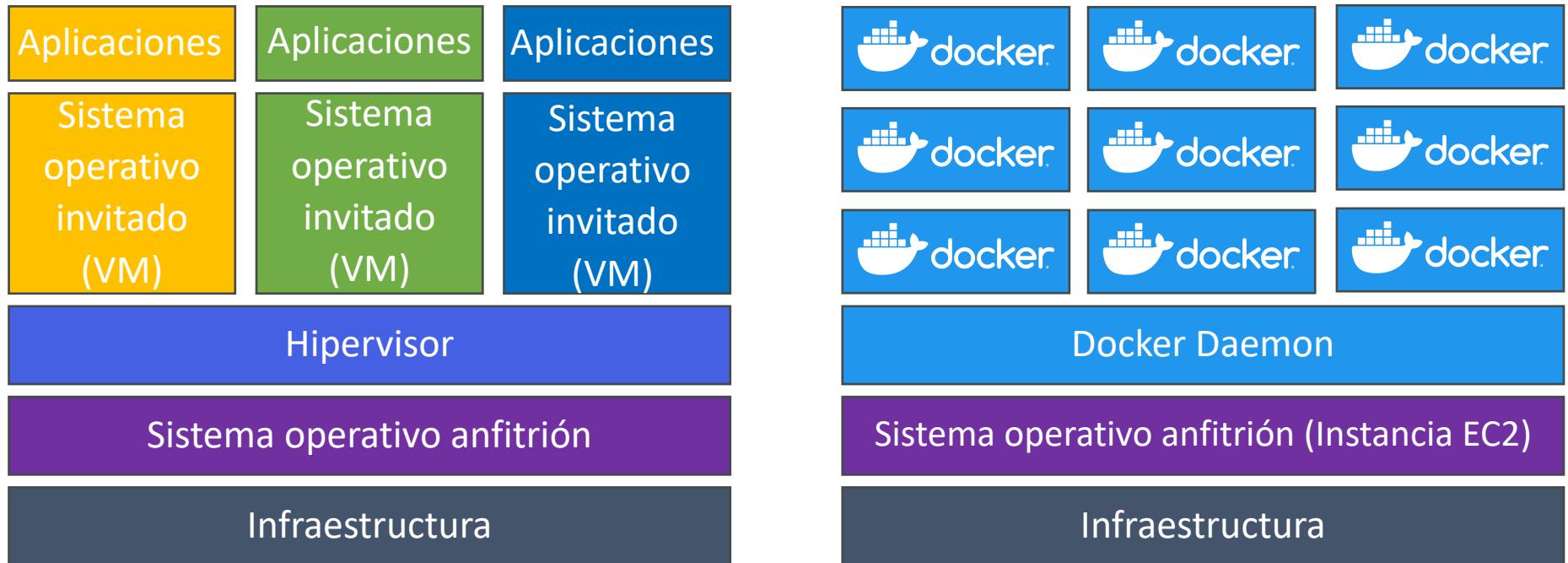


¿Dónde se almacenan las imágenes Docker?

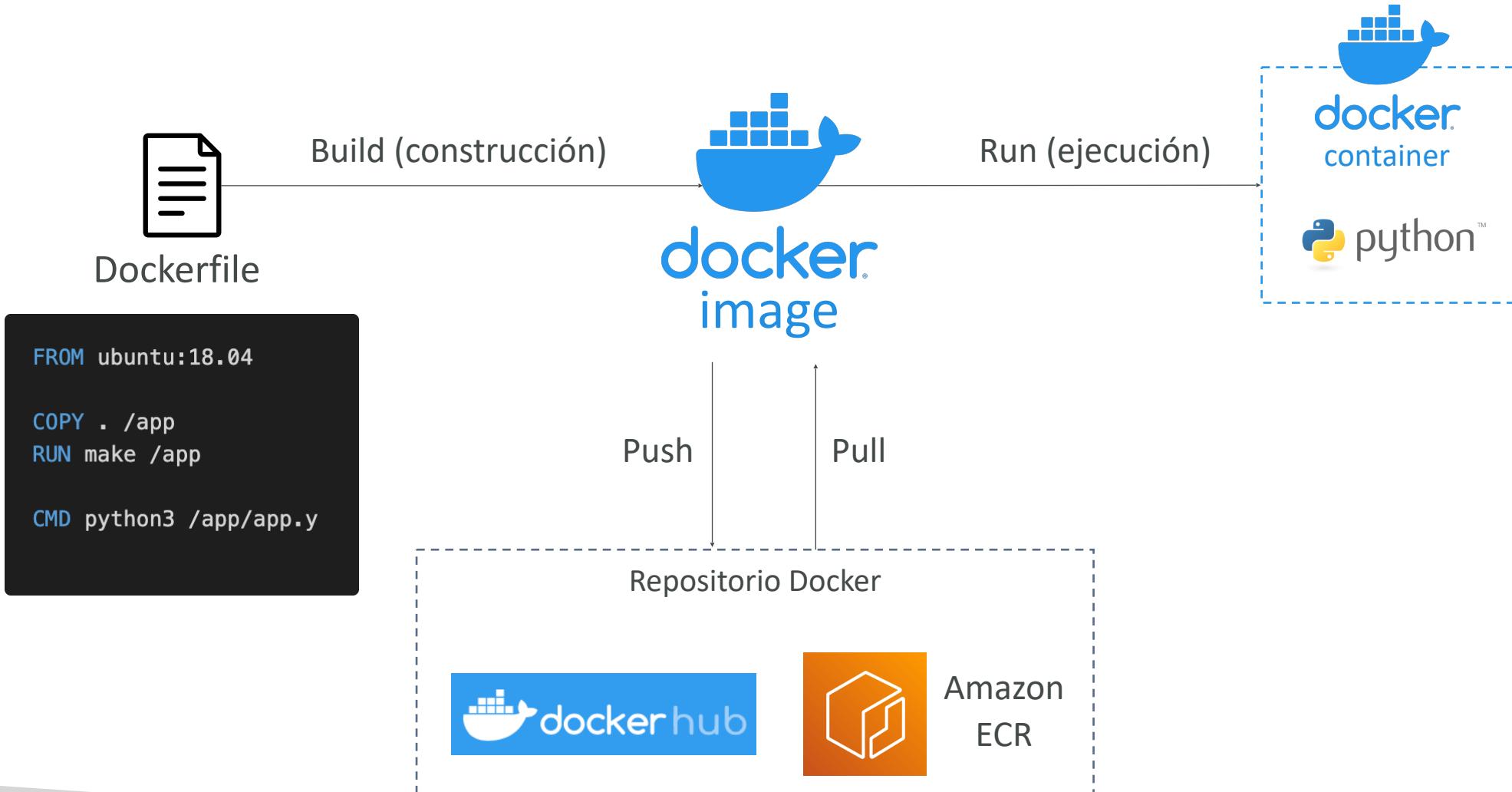
- Las imágenes Docker se almacenan en repositorios Docker
- **Docker Hub (<https://hub.docker.com>)**
 - Repositorio **público**
 - Encuentre imágenes base para muchas tecnologías o sistemas operativos (por ejemplo, Ubuntu, MySQL, ...)
- **Amazon ECR (Registro elástico de contenedores de Amazon)**
 - Repositorio **privado**
 - Repositorio **público** (**Galería pública de Amazon ECR** <https://gallery.ecr.aws>)

Docker vs máquinas virtuales

- Docker es "algo así" como una tecnología de virtualización, pero no exactamente
- Los recursos se comparten con el host => muchos contenedores en un servidor



Primeros pasos con Docker



Gestión de contenedores Docker en AWS

- **Amazon Elastic Container Service (Amazon ECS)**

- Plataforma de contenedores propia de Amazon



Amazon ECS

- **Servicio Amazon Elastic Kubernetes (Amazon EKS)**

- Kubernetes administrado por Amazon (código abierto)



Amazon EKS

- **AWS Fargate**

- Plataforma de contenedores sin servidor propia de Amazon
- Funciona con ECS y con EKS



AWS Fargate

- **Amazon ECR**

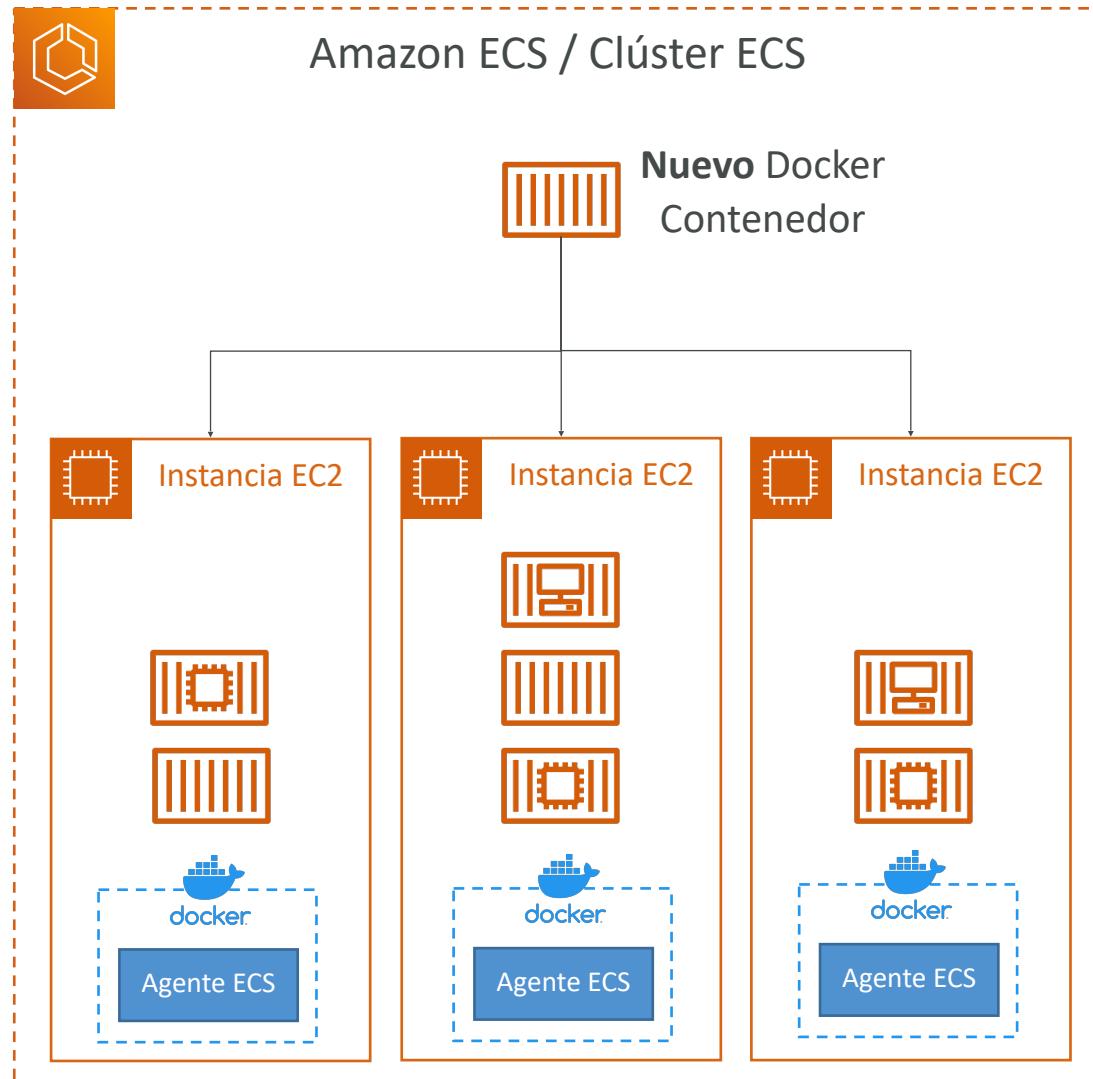
- Almacena imágenes de contenedores



Amazon ECR

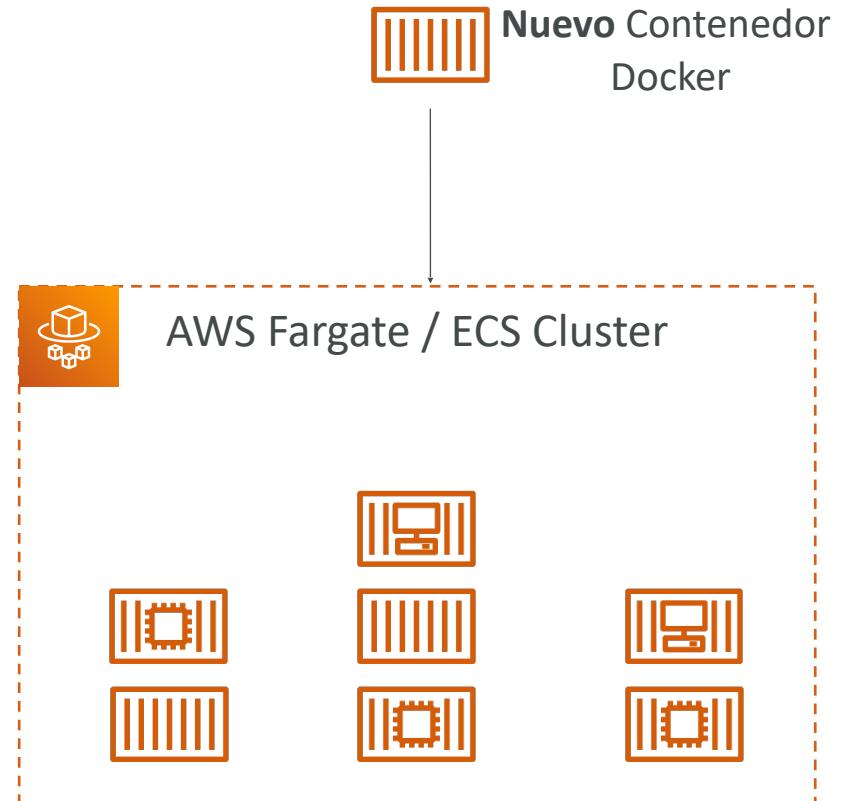
Amazon ECS - Tipo de lanzamiento EC2

- ECS = Elastic Container Service
- Lanzar contenedores Docker en AWS = Lanzar **tareas ECS** en clústeres ECS
- **Tipo de lanzamiento EC2: debe aprovisionar y mantener la infraestructura (las instancias EC2)**
- Cada Instancia EC2 debe ejecutar el Agente ECS para registrarse en el Cluster ECS
- AWS se encarga de iniciar / detener los contenedores



Amazon ECS - Tipo de lanzamiento Fargate

- Lanzar contenedores Docker en AWS
- **No aprovisionas la infraestructura (no hay instancias EC2 que administrar)**
- **¡Todo es Serverless!**
- Sólo tienes que crear definiciones de tareas
- AWS ejecuta las tareas ECS por ti en función de la CPU / RAM que necesites.
- Para escalar, basta con aumentar el número de tareas. Simple - no más instancias EC2



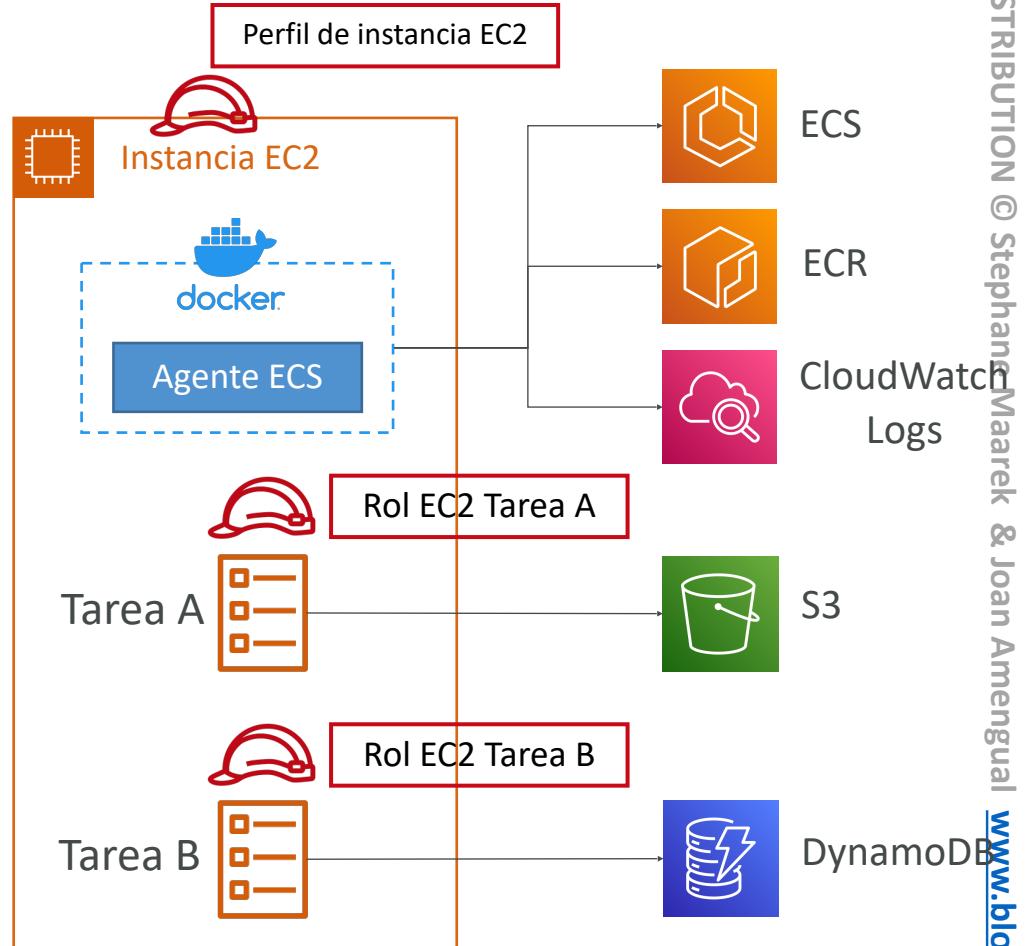
Amazon ECS - Roles IAM para ECS

- **Perfil de instancia EC2 (sólo tipo de lanzamiento EC2):**

- Utilizado por el agente ECS
- Realiza llamadas API al servicio ECS
- Envía logs de contenedores a CloudWatch Logs
- Extrae imagen de Docker de ECR
- Hace referencia a datos sensibles en Secrets Manager o SSM Parameter Store

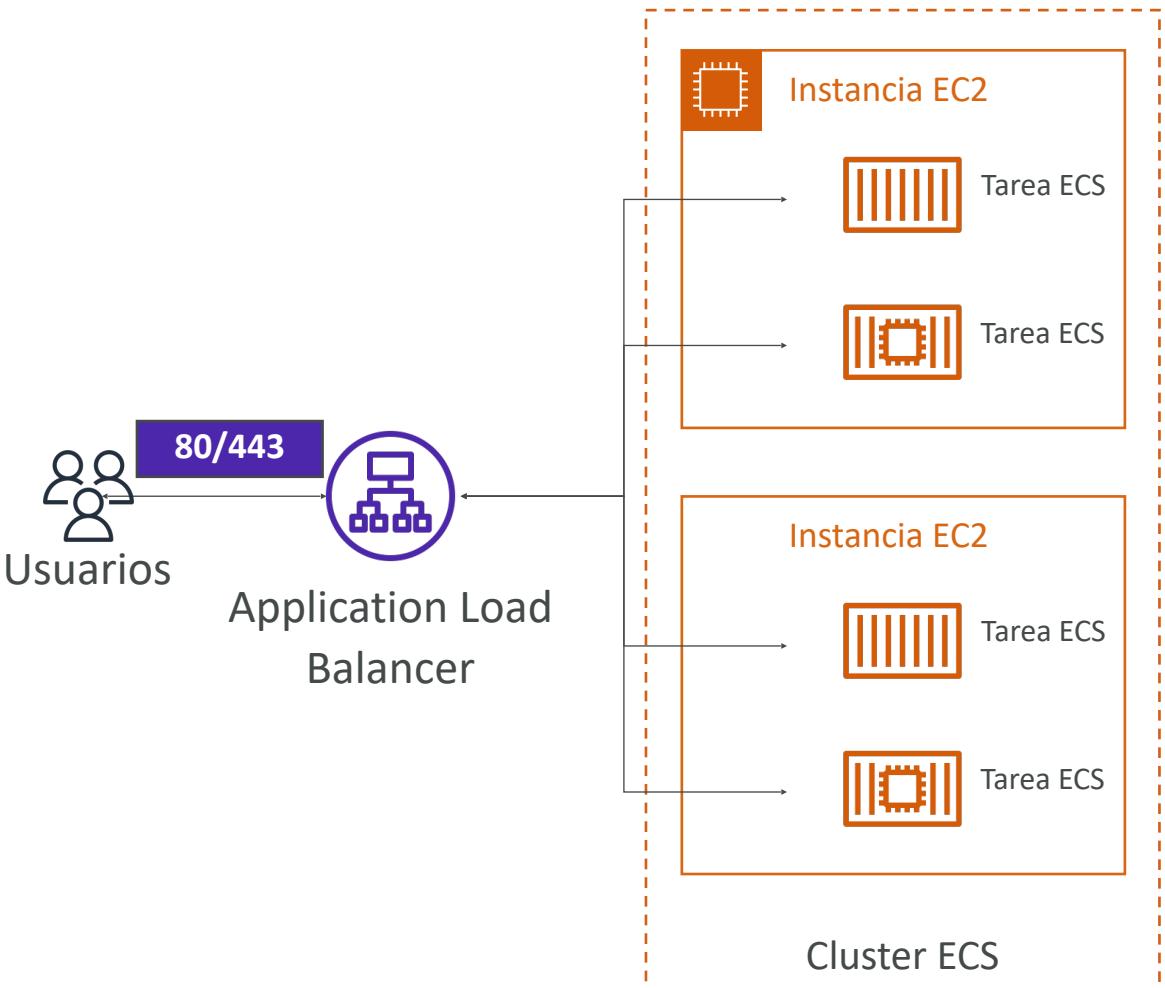
- **Rol de tarea ECS:**

- Permite que cada tarea tenga un rol específico
- Utiliza diferentes roles para los diferentes Servicios ECS que ejecute
- El rol de la tarea se define en la definición de la tarea



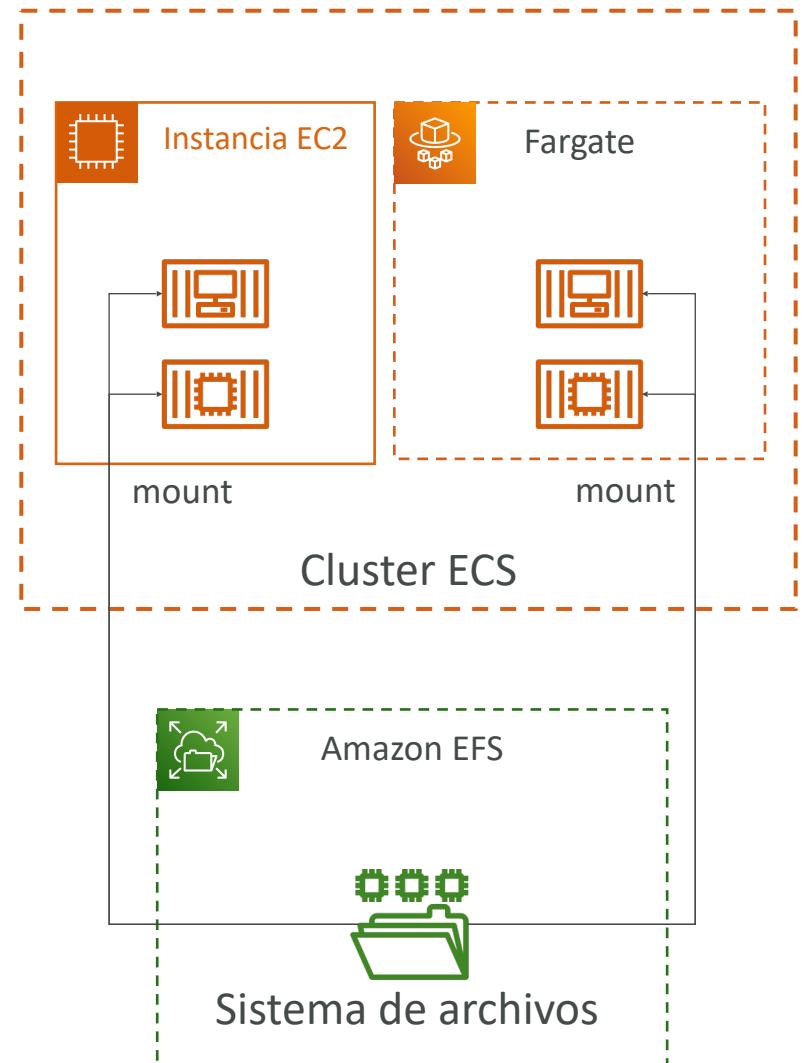
Amazon ECS - Integraciones de balanceadores de carga

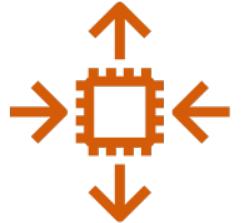
- **Application Load Balancer** es compatible y funciona para la mayoría de los casos de uso.
- **Network Load Balancer** recomendado solo para casos de uso de alto rendimiento o para combinarlo con AWS Private Link
- **Elastic Load Balancer** es compatible pero no se recomienda (sin características avanzadas - sin Fargate)



Amazon ECS - Volúmenes de datos (EFS)

- Montar sistemas de archivos EFS en tareas ECS
- Funciona tanto para los tipos de lanzamiento **EC2** como **Fargate**
- Las tareas que se ejecuten en cualquier AZ compartirán los mismos datos en el sistema de archivos EFS
- **Fargate + EFS = Sin servidor**
- Casos de uso: almacenamiento compartido multi-AZ persistente para sus contenedores
- Nota:
 - Amazon S3 no se puede montar como sistema de archivos





Escalado automático del servicio ECS

- Aumentar/disminuir automáticamente el número deseado de tareas ECS
- Amazon ECS Auto Scaling utiliza **AWS Application Auto Scaling**
 - Utilización media de la CPU del servicio ECS
 - Utilización media de memoria del servicio ECS - Escalado en RAM
 - Recuento de solicitudes de ALB por objetivo - métrica procedente del ALB
- **Seguimiento de objetivo** - escala basada en el valor objetivo para una métrica específica de CloudWatch
- **Escalado por pasos** - escalado basado en una alarma CloudWatch específica
- **Escalado programado** - escalado basado en una fecha/hora especificada (cambios predecibles)
- Autoescalado del servicio ECS (nivel de tarea) \neq Autoescalado de EC2 (nivel de instancia de EC2)
- Fargate Auto Scaling es mucho más fácil de configurar (porque es **Serverless**)

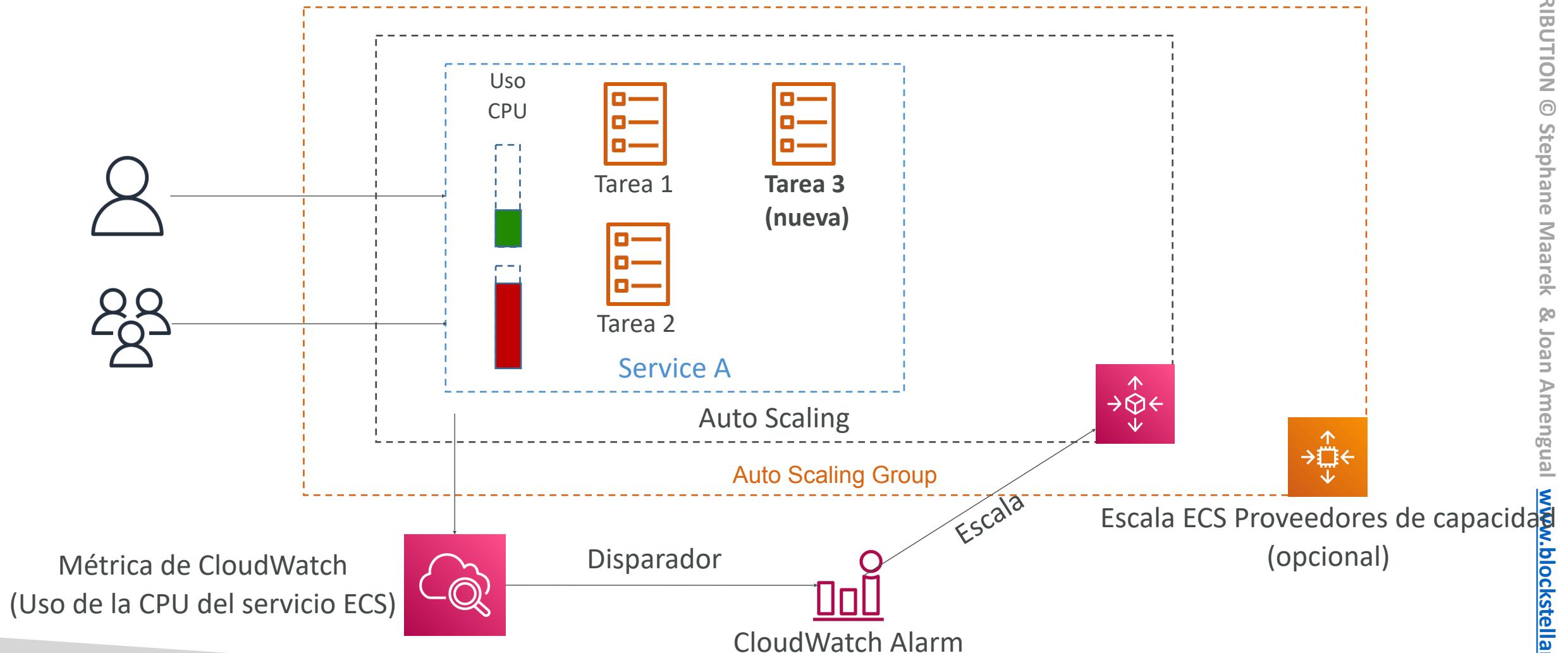
Tipo de lanzamiento EC2

Escalado automático de instancias EC2

- Acomodar el escalado de servicios ECS añadiendo instancias EC2 subyacentes
- **Escalado automático de grupos (Auto Scaling Group Scaling)**
 - Escala el ASG en función de la utilización de la CPU
 - Añadir instancias EC2 con el tiempo
- **Proveedor de capacidad de clúster ECS**
 - Se utiliza para aprovisionar y escalar automáticamente la infraestructura para tareas ECS
 - Proveedor de capacidad emparejado con un Auto Scaling Group
 - Añade instancias EC2 cuando falte capacidad (CPU, RAM...)

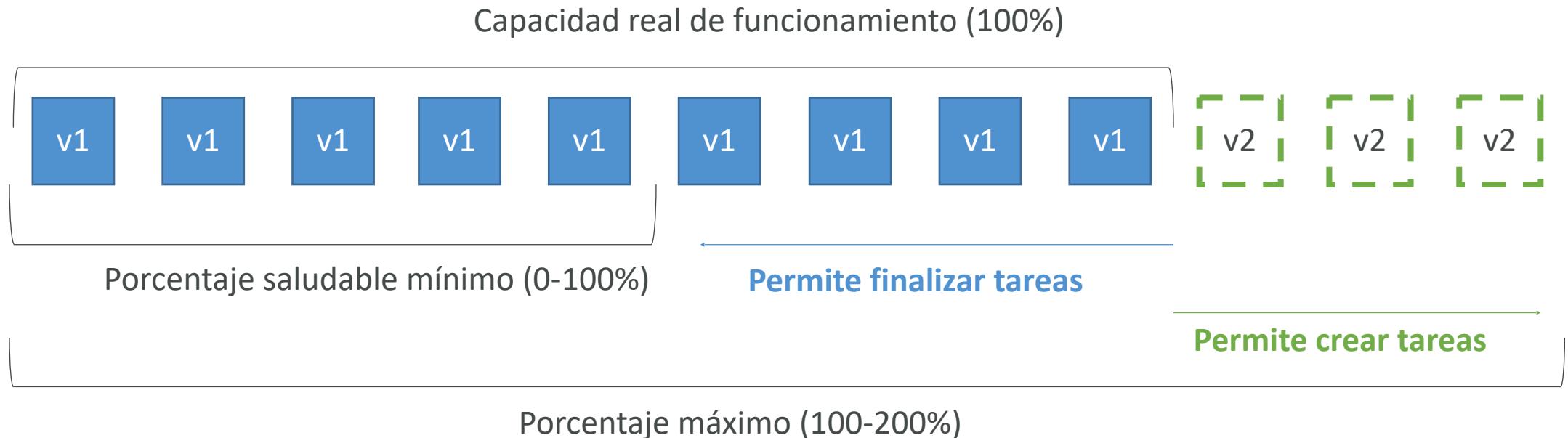
Escalado ECS

Ejemplo de uso de CPU de servicio



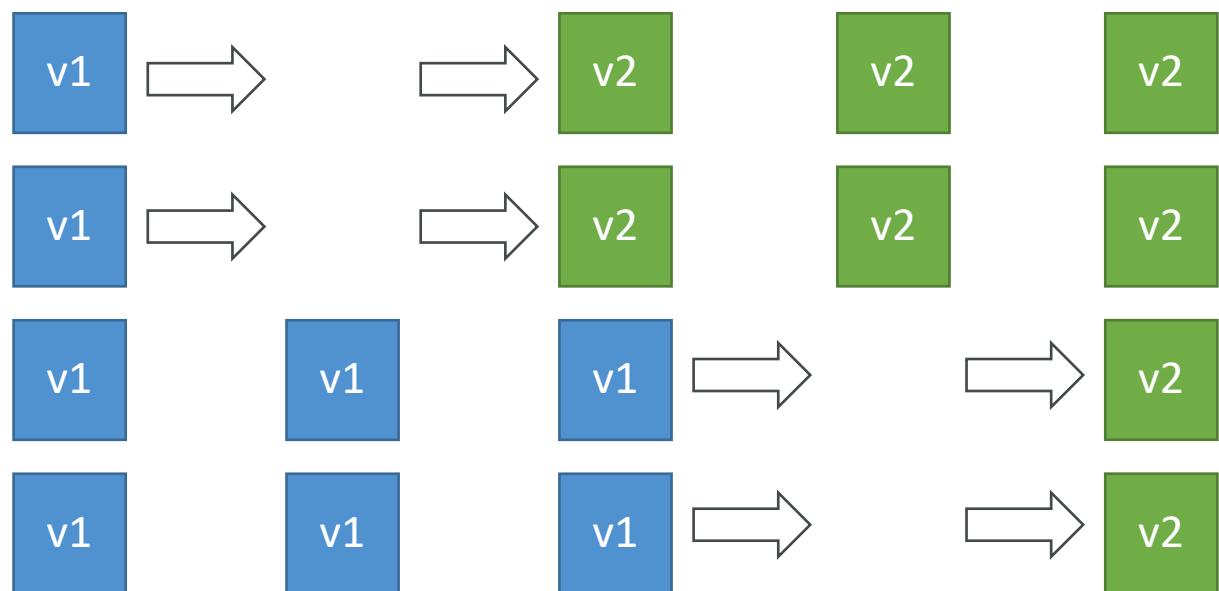
Actualizaciones continuas de ECS

- Al actualizar de v1 a v2, podemos controlar cuántas tareas pueden iniciarse y detenerse, y en qué orden



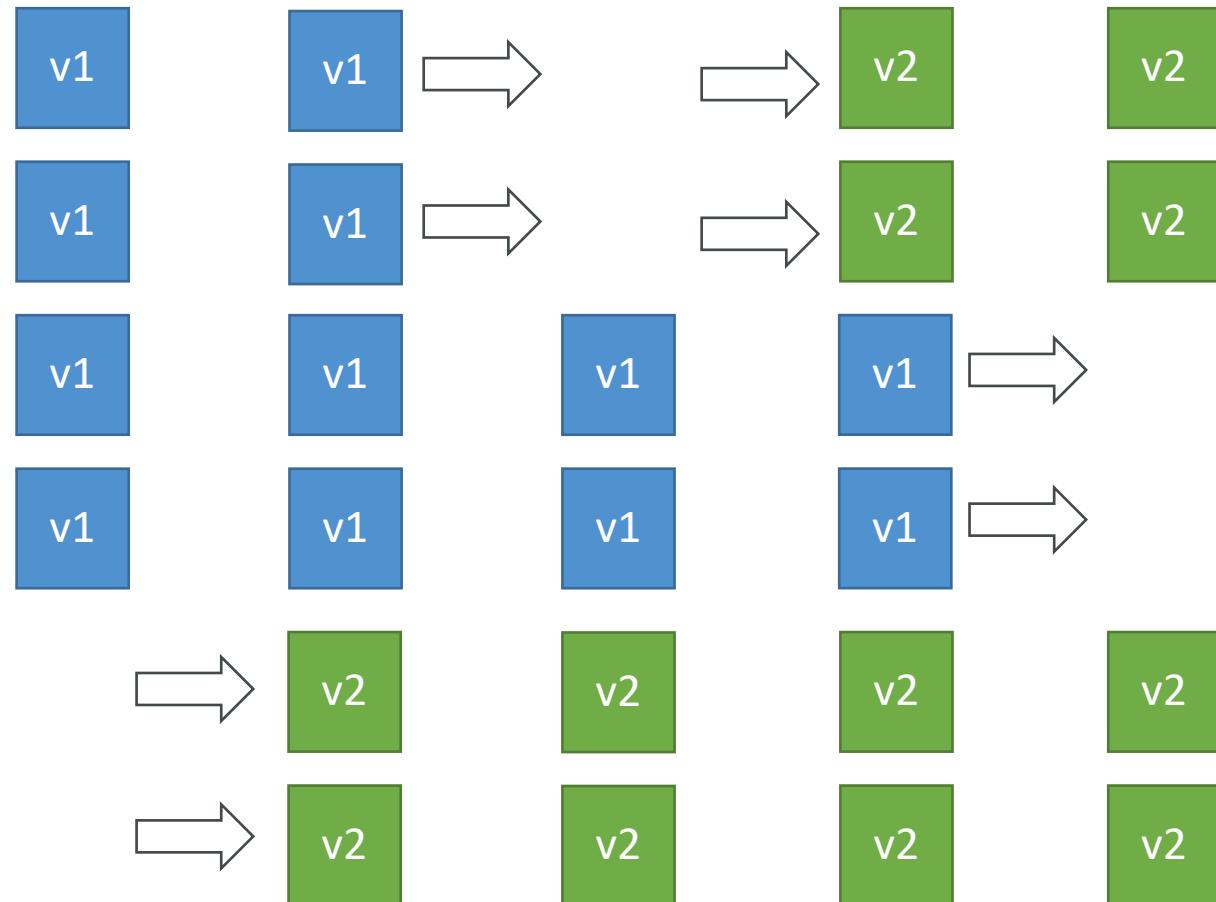
Actualización continua de ECS - Mín 50%, Máx 100%

- Número inicial de tareas: 4

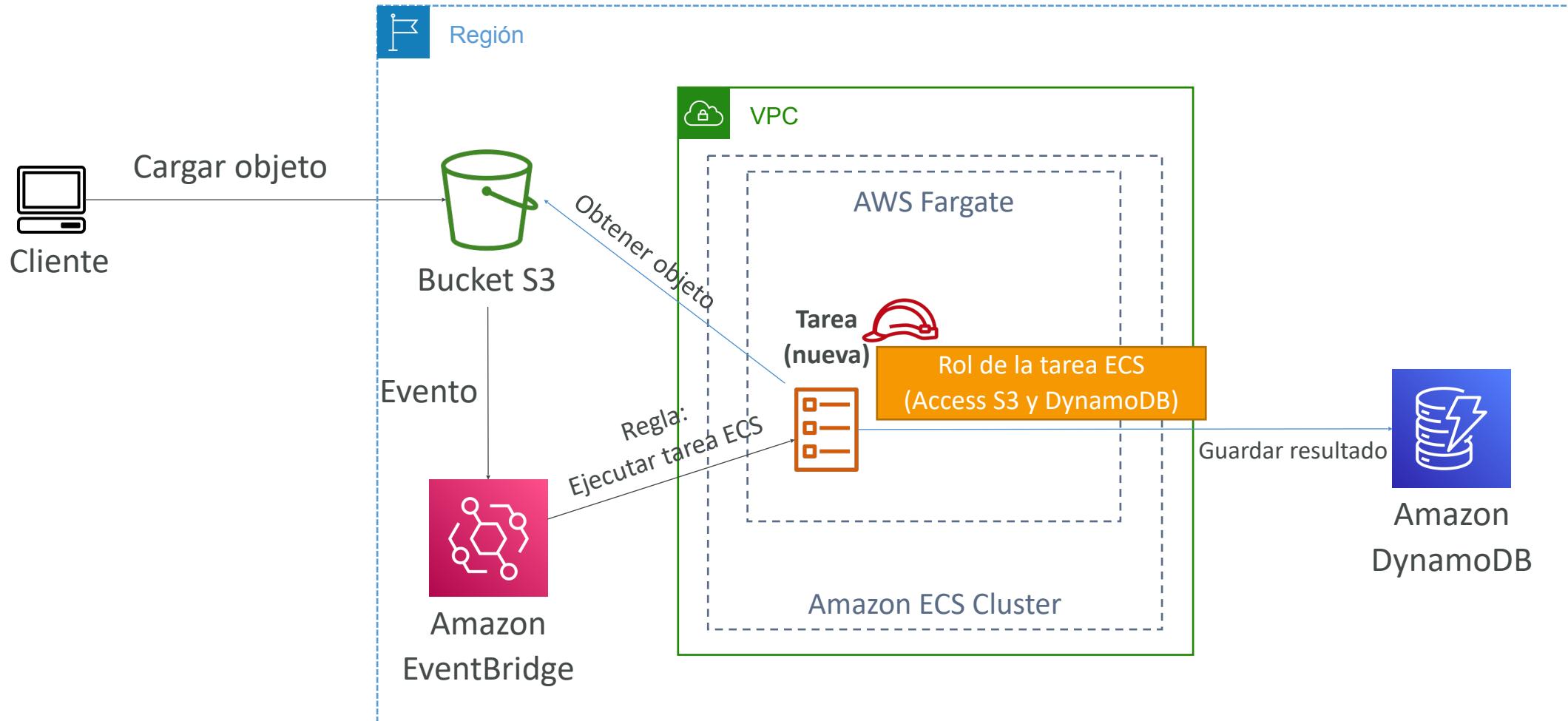


Actualización continua de ECS - Mín 100%, Máx 150%

- Número inicial de tareas: 4



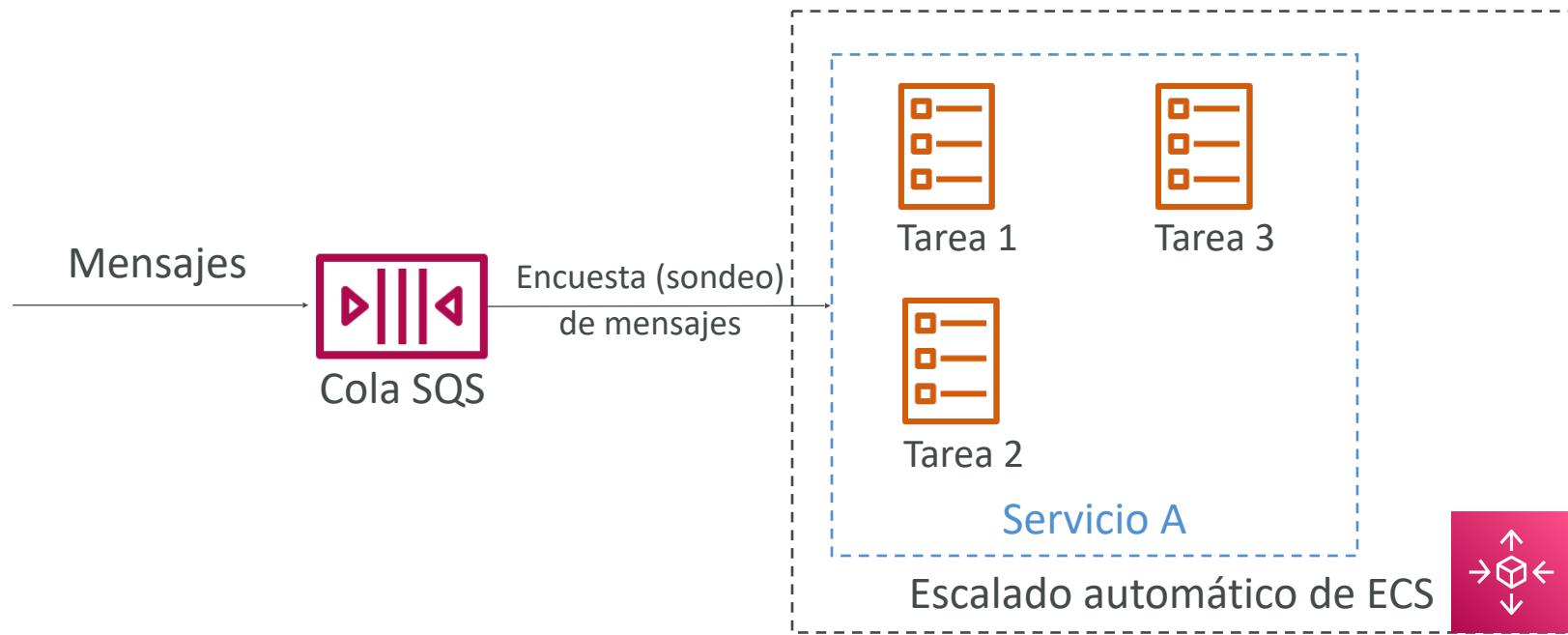
Tareas ECS invocadas por EventBridge



Tareas ECS invocadas por EventBridge



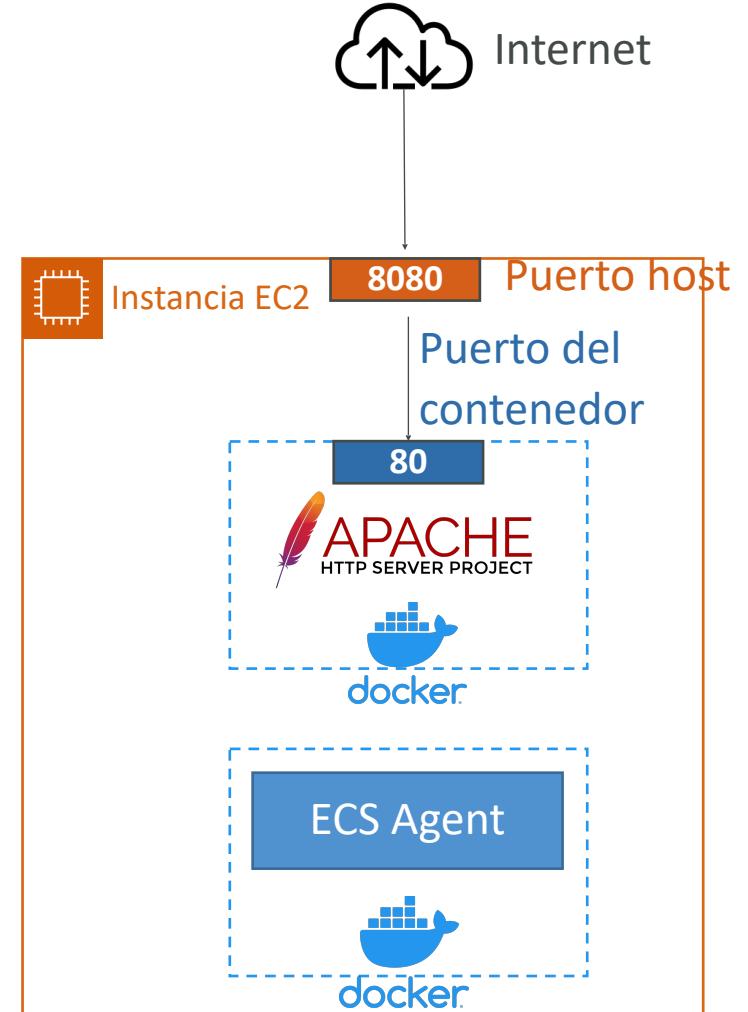
ECS - Ejemplo de cola SQS



Amazon ECS - Definiciones de tareas

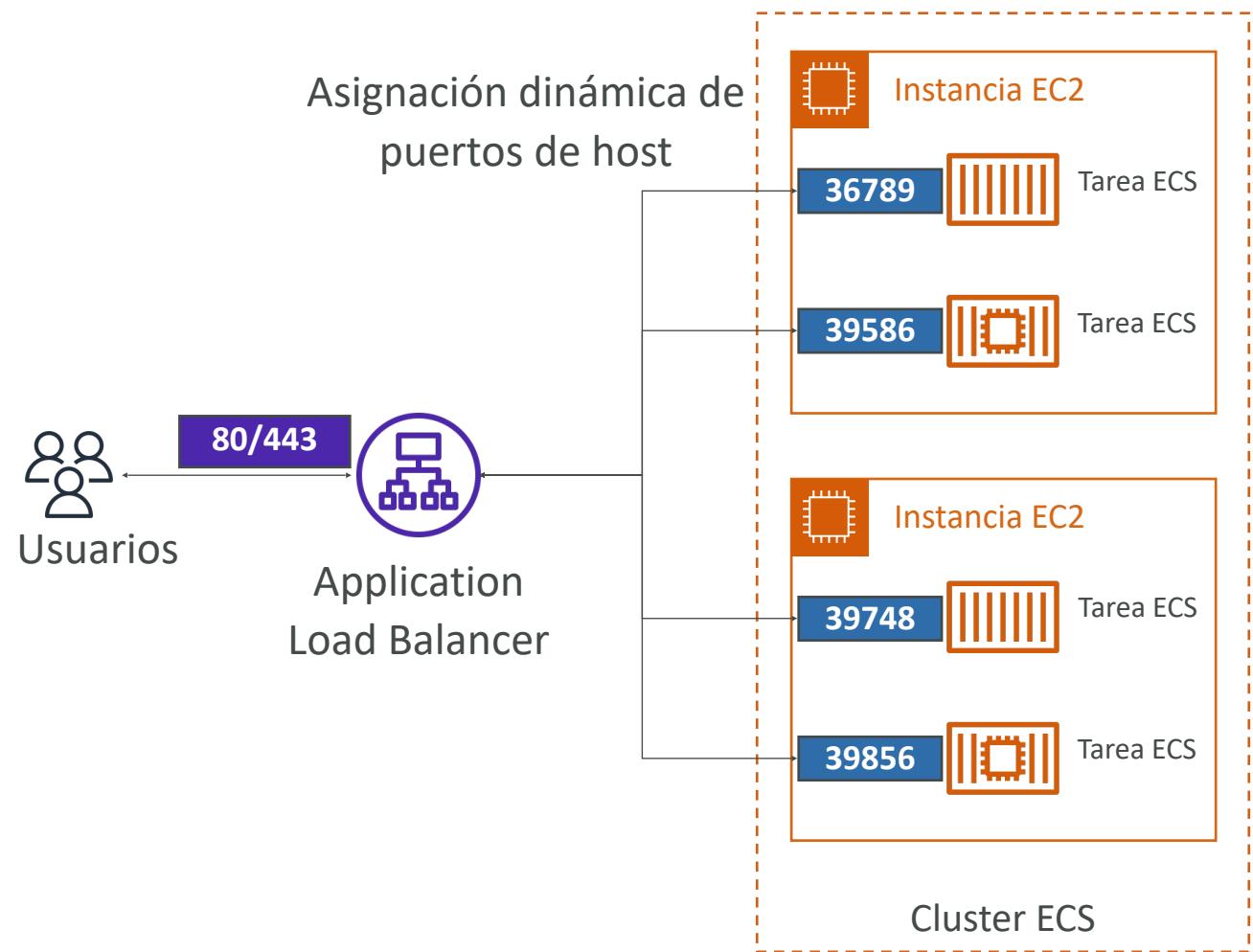


- Las definiciones de tareas son metadatos en **formato JSON** que indican a ECS cómo ejecutar un contenedor Docker
- Contiene información crucial, como:
 - Nombre de la imagen
 - Enlace de puertos para el contenedor y el host
 - Memoria y CPU necesarias
 - Variables de entorno
 - Información de red
 - Rol IAM
 - Configuración de logs (por ejemplo, CloudWatch)
- Puedes definir hasta 10 contenedores en una definición de tarea



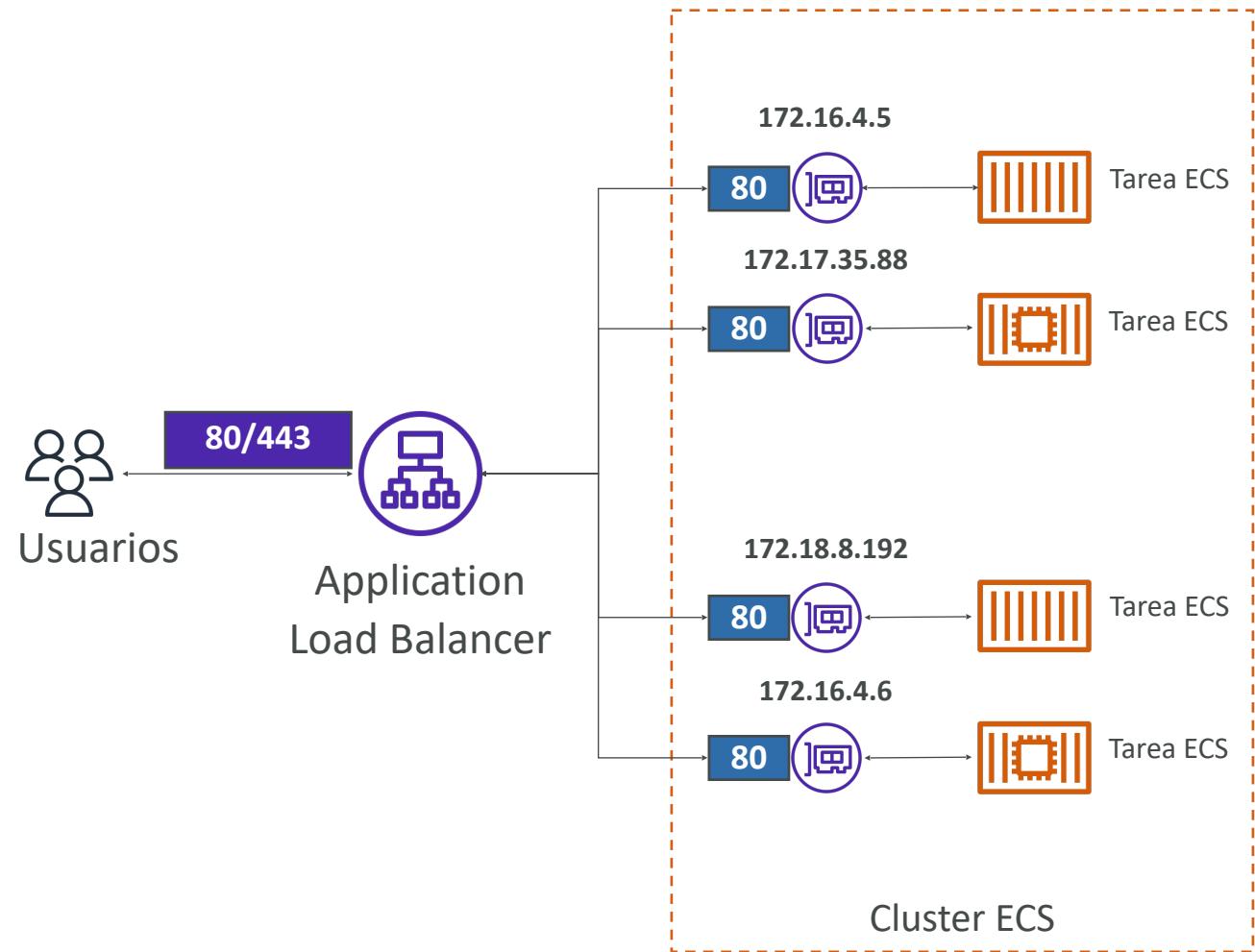
Amazon ECS - Load Balancer (Tipo de lanzamiento EC2)

- Obtenemos un **mapeo dinámico del puerto del host si defines sólo el puerto del contenedor en la definición de la tarea**
- El ALB encuentra el puerto correcto en tus Instancias EC2
- **Debes permitir en el grupo de seguridad de la instancia EC2 cualquier puerto del grupo de seguridad del ALB**



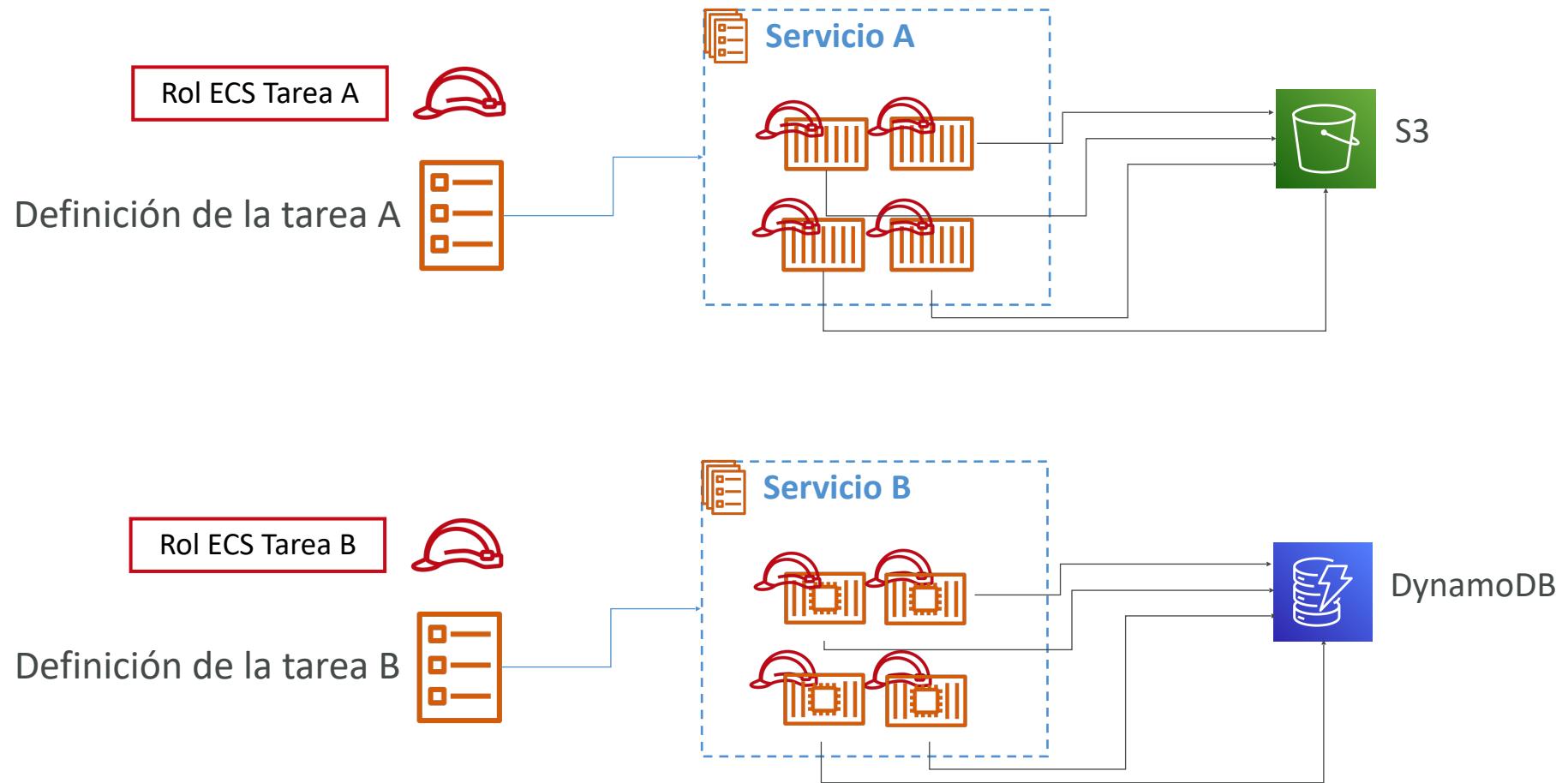
Amazon ECS – Load Balancing (Fargate)

- Cada tarea tiene **una IP privada única**
- **Define sólo el puerto del contenedor** (el puerto del host no es aplicable)
- Ejemplo
 - **Grupo de seguridad ECS ENI**
 - Permitir puerto 80 desde el ALB
 - **Grupo de seguridad ALB**
 - Permitir el puerto 80/443 desde la web



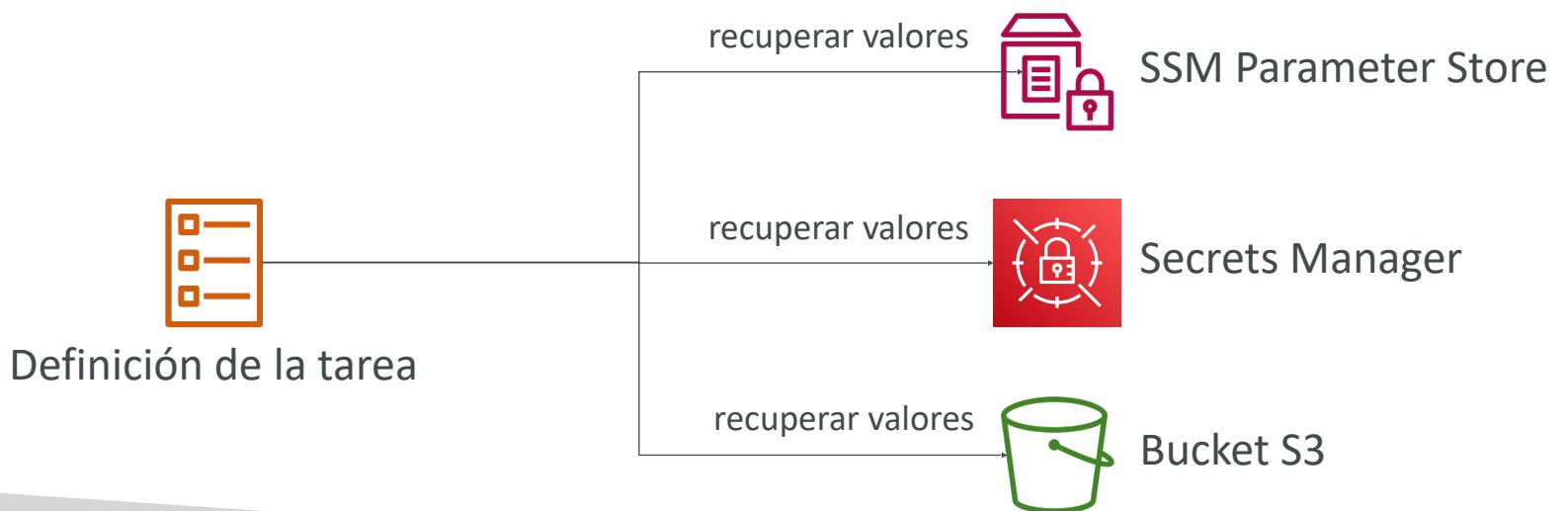
Amazon ECS

Un rol IAM por definición de tarea



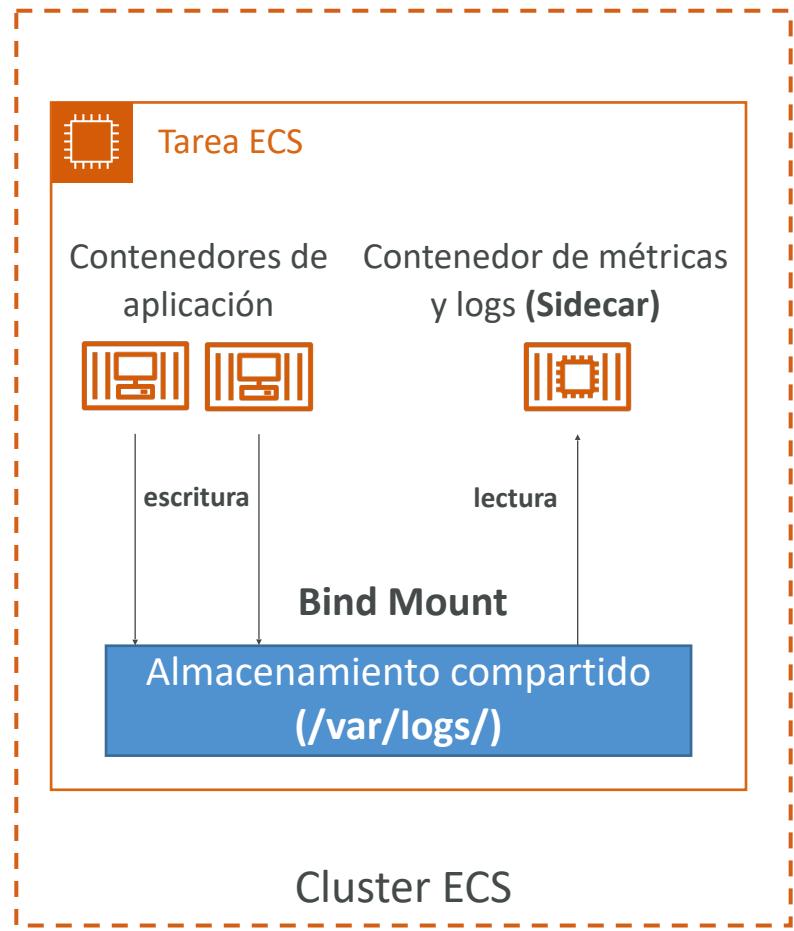
Amazon ECS - Variables de entorno

- Variable de entorno
 - **Hardcoded** - por ejemplo, URLs
 - **Almacén de Parámetros SSM** - variables sensibles (por ejemplo, claves API, configuraciones compartidas)
 - **Secrets Manager** - variables sensibles (por ejemplo, contraseñas de BD)
- Archivos de entorno (masivos) - Amazon S3



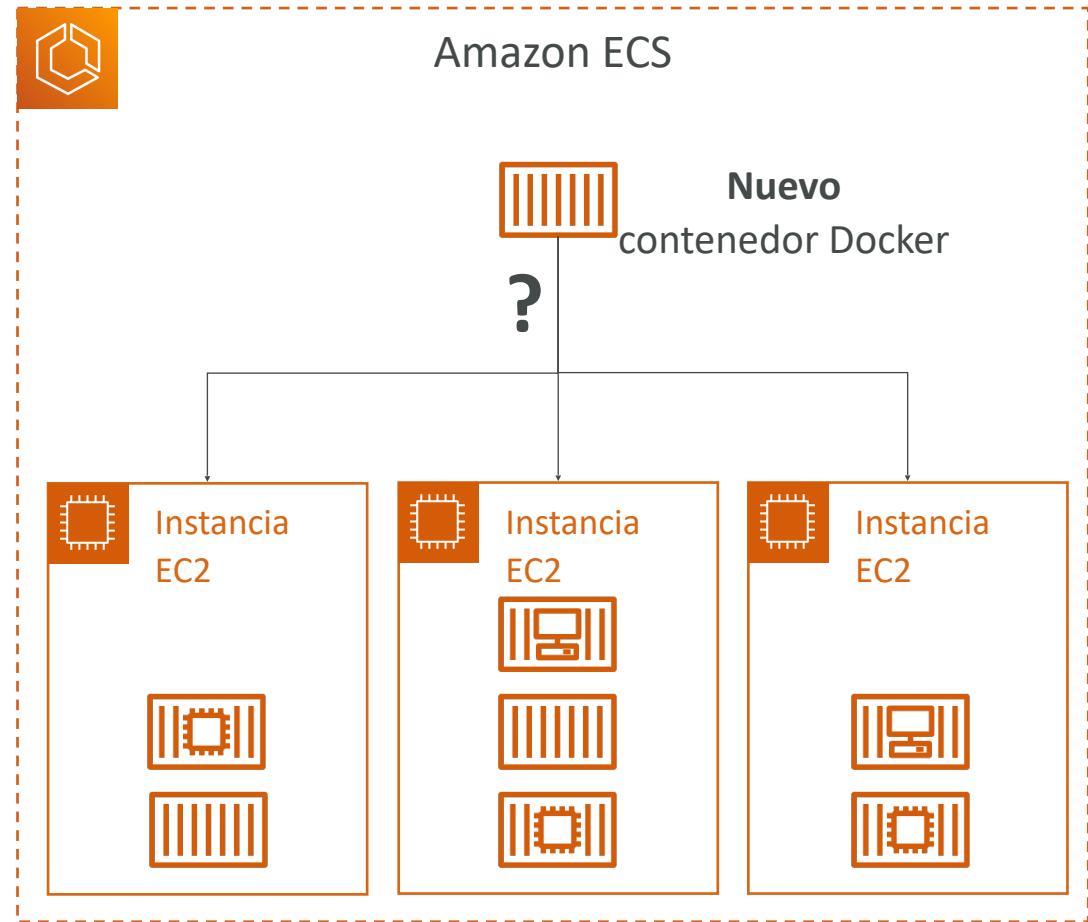
Amazon ECS - Volúmenes de datos (Bind Mounts)

- Comparte datos entre varios contenedores en la misma Definición de Tarea
- Funciona tanto para tareas **EC2** como **Fargate**
- **Tareas EC2** - utilizando almacenamiento de instancia EC2
 - Los datos están ligados al ciclo de vida de la instancia EC2
- **Tareas Fargate** - utilizando almacenamiento efímero
 - Los datos están ligados al contenedor o contenedores que los utilizan
 - 20 GiB - 200 GiB (por defecto 20 GiB)
- Casos de uso:
 - Compartir datos efímeros entre varios contenedores
 - Patrón de contenedor "sidecar", en el que el contenedor "sidecar" se utiliza para enviar métricas/logs a otros destinos



Amazon ECS - Colocación de tareas

- Cuando se inicia una tarea ECS con EC2 Launch Type, ECS debe determinar dónde colocarla, con las limitaciones de **CPU, memoria y puerto disponible**.
- Del mismo modo, cuando se escala un servicio, ECS debe determinar qué tarea terminar
- Puedes definir
 - **Estrategia de colocación de tareas**
 - **Restricciones de colocación de tareas**
- Nota: sólo para ECS con EC2, no soporta Fargate



Amazon ECS - Proceso de colocación de tareas

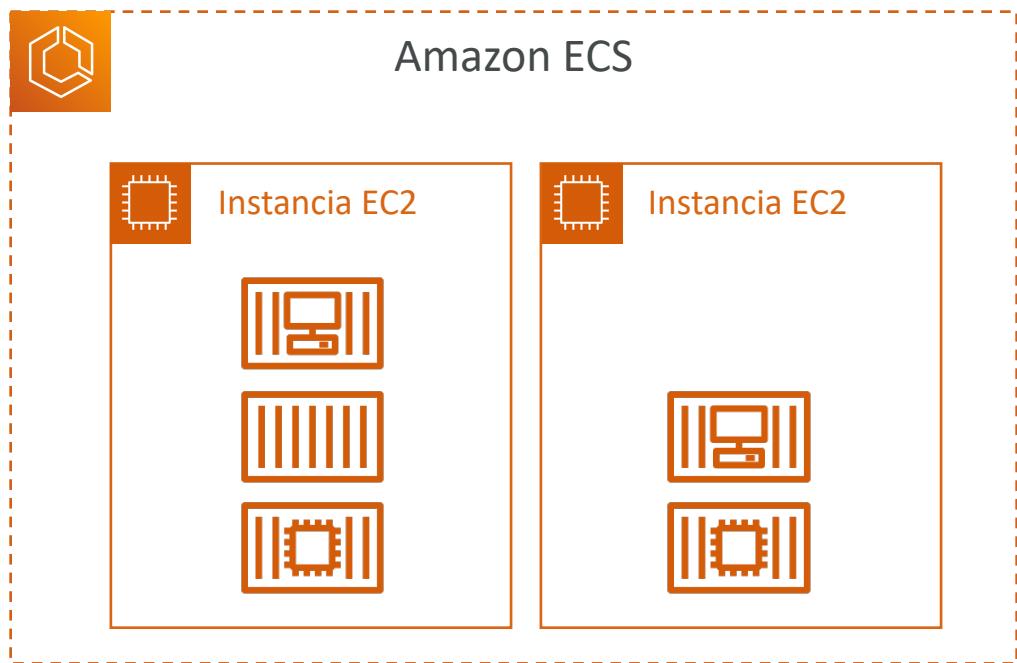
- Las estrategias de colocación de tareas son un mejor esfuerzo
- Cuando Amazon ECS coloca una tarea, utiliza el siguiente proceso para seleccionar la instancia de contenedor EC2 adecuada:
 1. Identificar qué instancias satisfacen los requisitos de CPU, memoria y puerto
 2. Identificar qué instancias satisfacen las restricciones de colocación de tareas
 3. Identificar qué instancias satisfacen las estrategias de colocación de tareas
 4. Seleccionar las instancias

Amazon ECS - Estrategias de colocación de tareas

- **Binpack**

- Las tareas se colocan en la menor cantidad disponible de CPU y memoria
- Minimiza el número de instancias EC2 en uso (ahorro de costes)

```
"placementStrategy": [  
    {  
        "type": "binpack",  
        "field": "memory"  
    }  
]
```

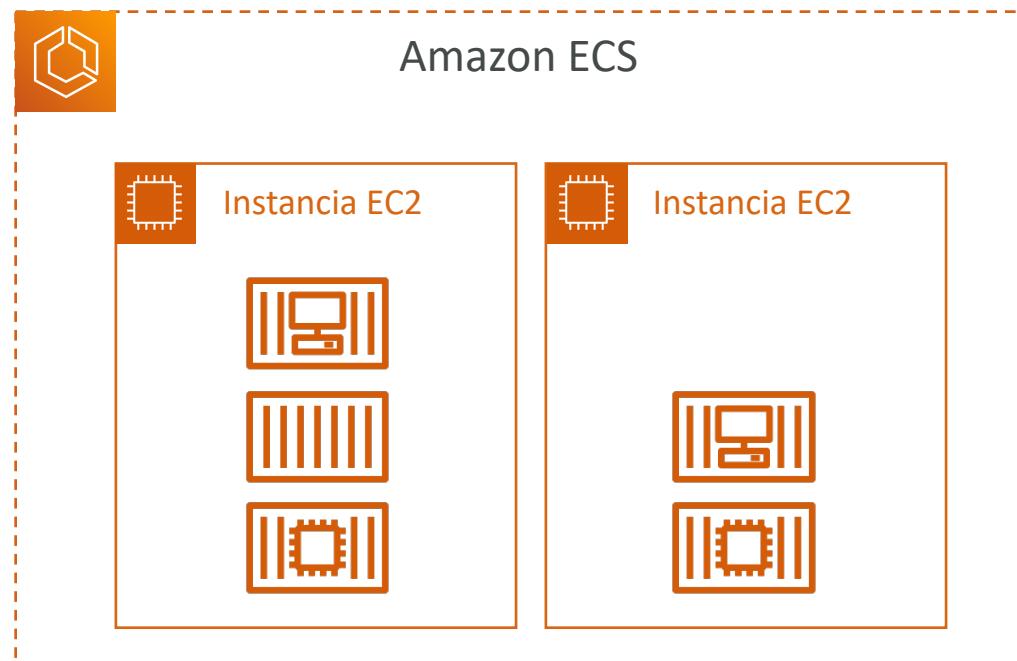


Amazon ECS - Estrategias de colocación de tareas

- **Random**

- Las tareas se colocan aleatoriamente

```
"placementStrategy": [  
    {  
        "type": "random"  
    }  
]
```

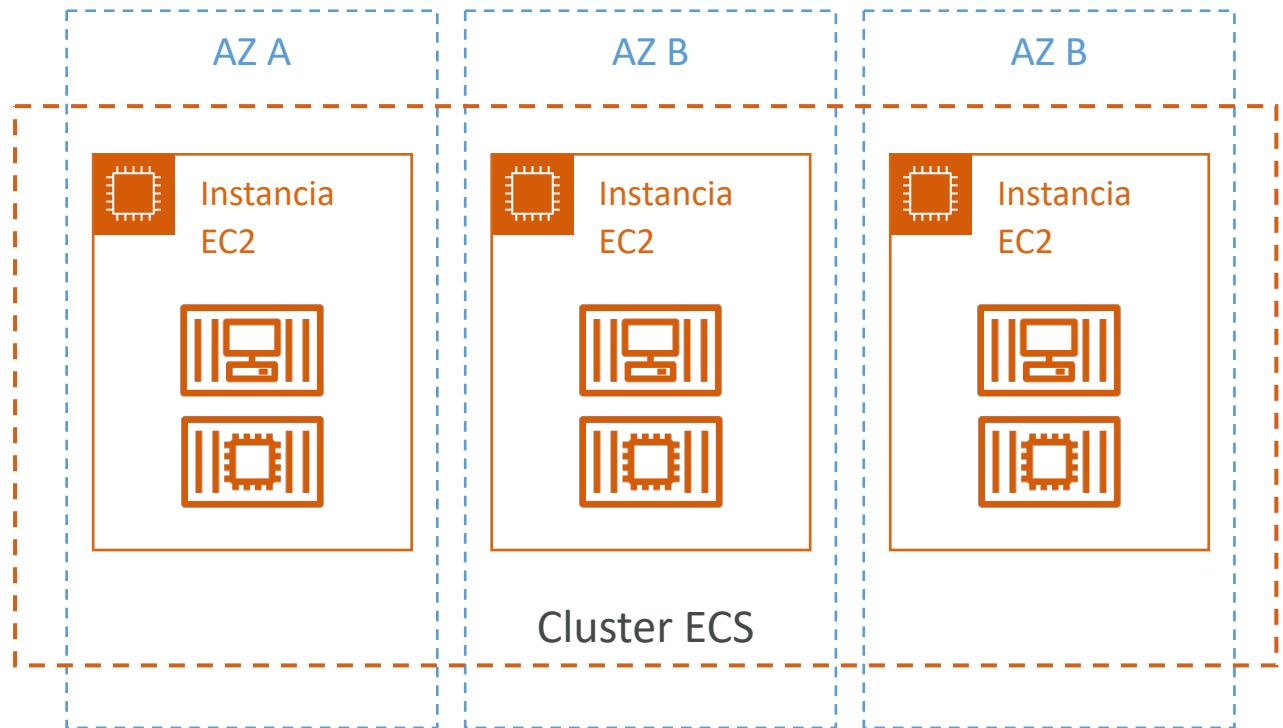


Amazon ECS - Estrategias de colocación de tareas

- **Spread**

- Las tareas se colocan uniformemente en función del valor especificado
- Ejemplo: instanceId, attribute:ecs.availability-zone, ...

```
"placementStrategy": [
  {
    "type": "spread",
    "field": "attribute:ecs.availability-zone"
  }
]
```



Amazon ECS - Estrategias de colocación de tareas

- Puedes mezclarlos

```
"placementStrategy": [  
    {  
        "type": "spread",  
        "field": "attribute:ecs.availability-zone"  
    },  
    {  
        "type": "spread",  
        "field": "instanceId"  
    }  
]
```

```
"placementStrategy": [  
    {  
        "type": "spread",  
        "field": "attribute:ecs.availability-zone"  
    },  
    {  
        "type": "binpack",  
        "field": "memory"  
    }  
]
```

Amazon ECS - Restricciones de colocación de tareas

- **distinctInstance:**

- Las tareas se colocan en una instancia EC2 diferente

```
"placementConstraints": [  
    {  
        "type": "distinctInstance"  
    }  
]
```

- **memberOf:**

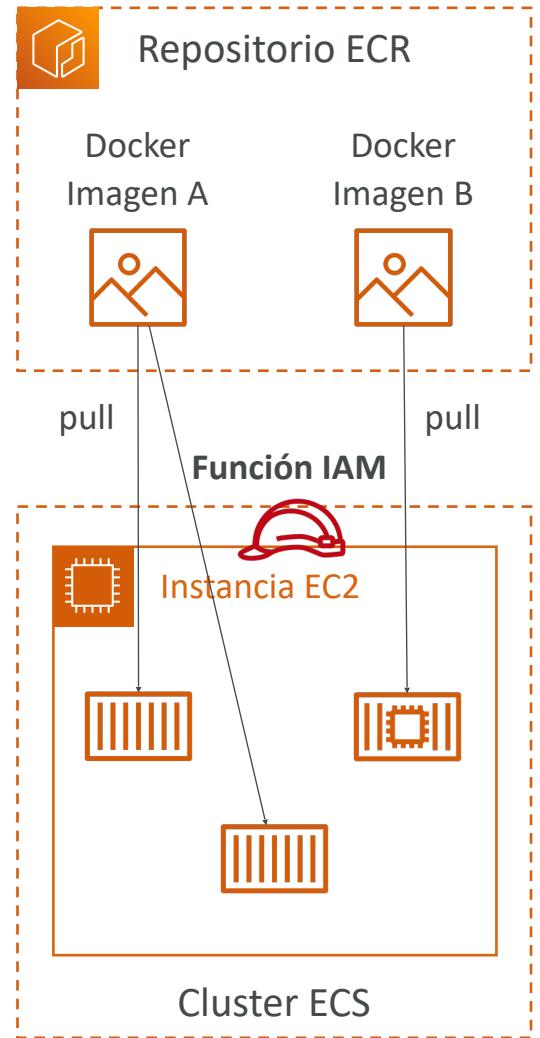
- Las tareas se colocan en instancias EC2 que satisfacen una expresión especificada
- Utiliza el Cluster Query Language (avanzado)

```
"placementConstraints": [  
    {  
        "type": "memberOf",  
        "expression": "attribute:ecs.instance-type =~ t2.*"  
    }  
]
```

Amazon ECR



- ECR = Registro elástico de contenedores
- Almacenar y administrar imágenes Docker en AWS
- Repositorio **privado** y **público** (**Amazon ECR Public Gallery** <https://gallery.ecr.aws>)
- Totalmente integrado con ECS, respaldado por Amazon S3
- El acceso se controla a través de IAM (errores de permiso => política)
- Soporta escaneo de vulnerabilidades de imágenes, versionado, etiquetas de imágenes, ciclo de vida de imágenes, ...



Amazon ECR - Uso de la CLI de AWS

- **Comando de inicio de sesión**

- CLI de AWS v2

```
aws ecr get-login-password --region region | docker login --username AWS  
--password-stdin aws_account_id.dkr.ecr.region.amazonaws.com
```

- **Comandos Docker**

- Push

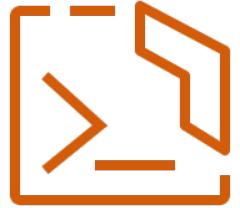
```
docker push aws_account_id.dkr.ecr.region.amazonaws.com/demo:latest
```

- Pull

```
docker pull aws_account_id.dkr.ecr.region.amazonaws.com/demo:latest
```

- **En caso de que una instancia EC2 (o tú) no pueda extraer una imagen Docker, comprueba los permisos IAM**

AWS Copilot

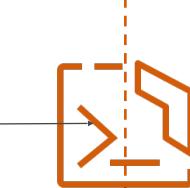


- Herramienta CLI para crear, publicar y utilizar aplicaciones en contenedores listas para la producción
- Ejecuta tus aplicaciones en **AppRunner**, **ECS** y **Fargate**
- Te ayuda a centrarte en crear aplicaciones en lugar de configurar la infraestructura
- Proporciona toda la infraestructura necesaria para las aplicaciones en contenedores (ECS, VPC, ELB, ECR...)
- Despliegues automatizados con un comando utilizando CodePipeline
- Despliega en múltiples entornos
- Solución de problemas, logs, estado de salud...

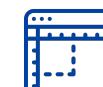


Arquitectura de microservicios

Utiliza CLI o YAML para describir la arquitectura de tus aplicaciones



CLI para aplicaciones en contenedores



Configuración de infraestructura bien diseñada



Pipeline de despliegue



Operaciones eficaces y resolución de problemas



Amazon ECS



AWS Fargate



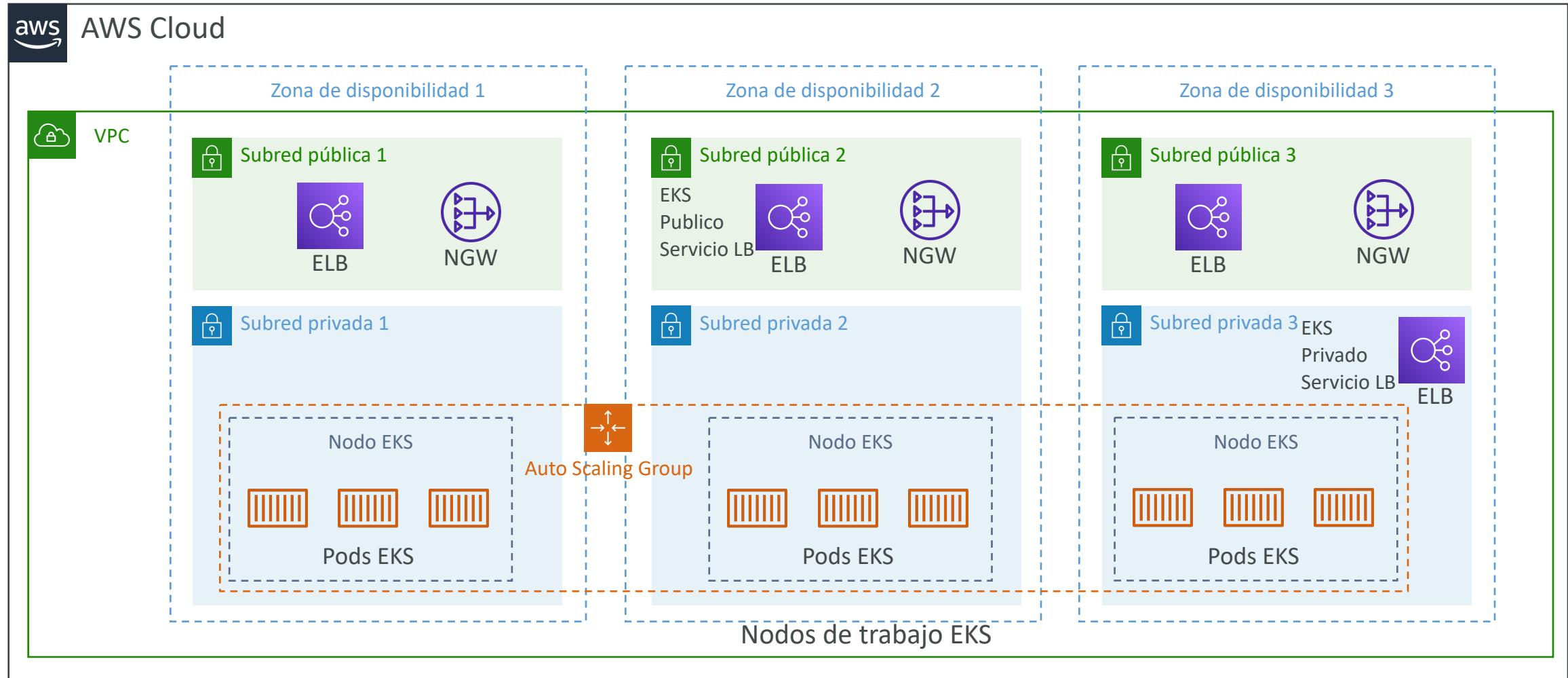
AWS App Runner

Visión general de Amazon EKS



- Amazon EKS = Servicio Amazon Elastic **Kubernetes**
- Es una forma de lanzar **clústeres Kubernetes administrados en AWS**
- Kubernetes es un sistema de código abierto para el despliegue, escalado y gestión automáticos de aplicaciones en contenedores (normalmente Docker)
- Es una alternativa a ECS, objetivo similar pero API diferente
- EKS soporta **EC2** si quieres desplegar nodos trabajadores o **Fargate** para desplegar contenedores sin servidor
- **Caso de uso:** si tu empresa ya utiliza Kubernetes on-premises o en otra nube, y quiere migrar a AWS utilizando Kubernetes
- **Kubernetes es agnóstico a la nube (puede utilizarse en cualquier nube - Azure, GCP...)**

Amazon EKS - Diagrama



Amazon EKS - Tipos de nodos

- **Grupos de nodos gestionados**

- Crea y gestiona Nodos (instancias EC2) para ti
- Los nodos forman parte de un ASG gestionado por EKS
- Admite instancias bajo demanda o puntuales

- **Nodos autogestionados**

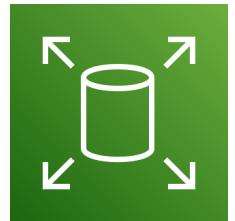
- Nodos creados por ti y registrados en el clúster EKS y gestionados por un ASG
- Puede utilizar AMI preconstruidas - Amazon EKS Optimized AMI
- Admite instancias bajo demanda o puntuales

- **AWS Fargate**

- No requiere mantenimiento; no se administran nodos

Amazon EKS - Volúmenes de datos

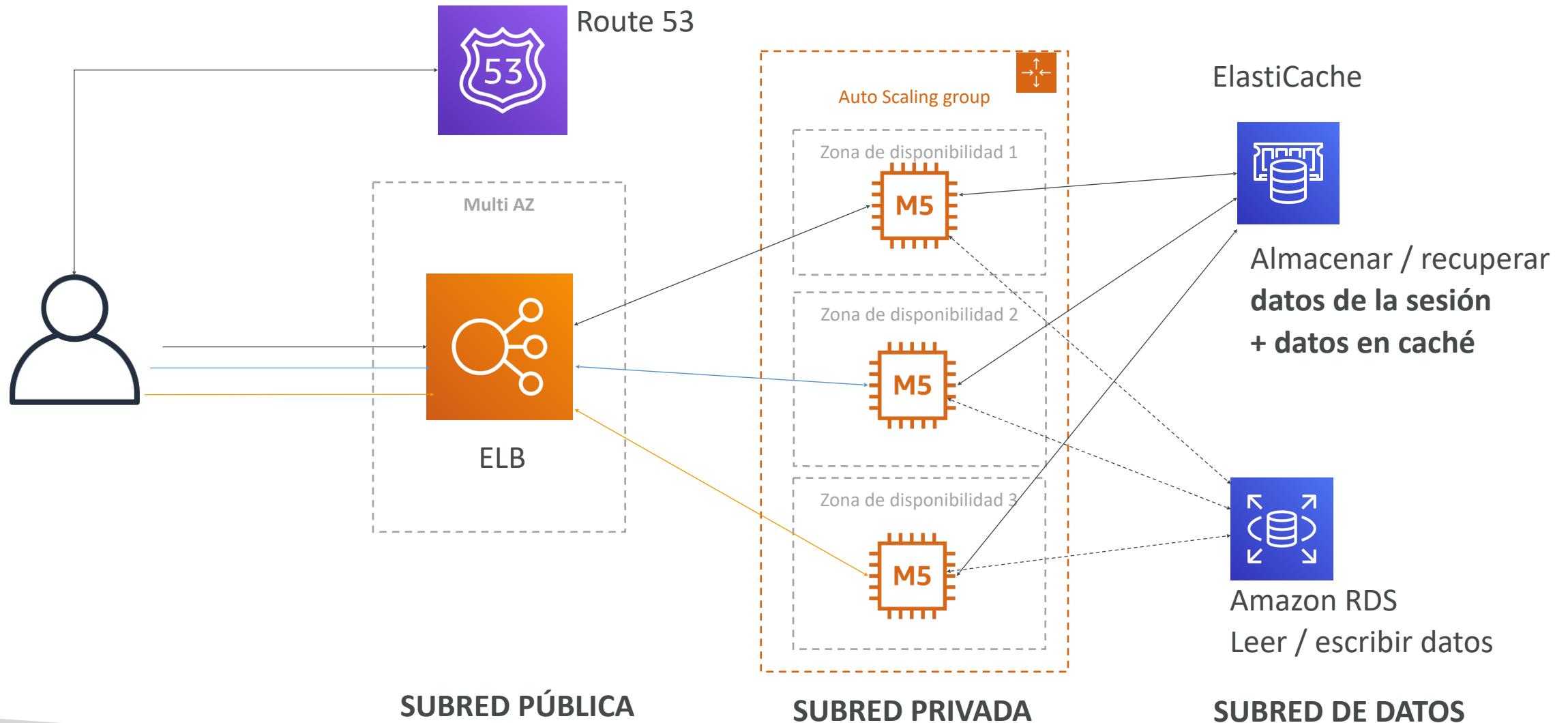
- Necesidad de especificar el **StorageClass** en el clúster EKS
- Aprovecha un controlador compatible con **Container Storage Interface (CSI)**
- Compatible con...
 - Amazon EBS
 - Amazon EFS (funciona con Fargate)
 - Amazon FSx para Lustre
 - Amazon FSx para NetApp ONTAP



AWS Elastic Beanstalk

Implementar aplicaciones en AWS de forma segura y predecible

Arquitectura típica: Web App de 3 niveles



Problemas de los desarrolladores en AWS

- Gestionar la infraestructura
 - Despliegue del código
 - Configuración de todas las bases de datos,平衡adores de carga, etc.
 - Problemas de escalabilidad
-
- La mayoría de las aplicaciones web tienen la misma arquitectura (ALB + ASG)
 - Lo único que quieren los desarrolladores es que su código se ejecute
 - Posiblemente, de forma coherente en diferentes aplicaciones y entornos

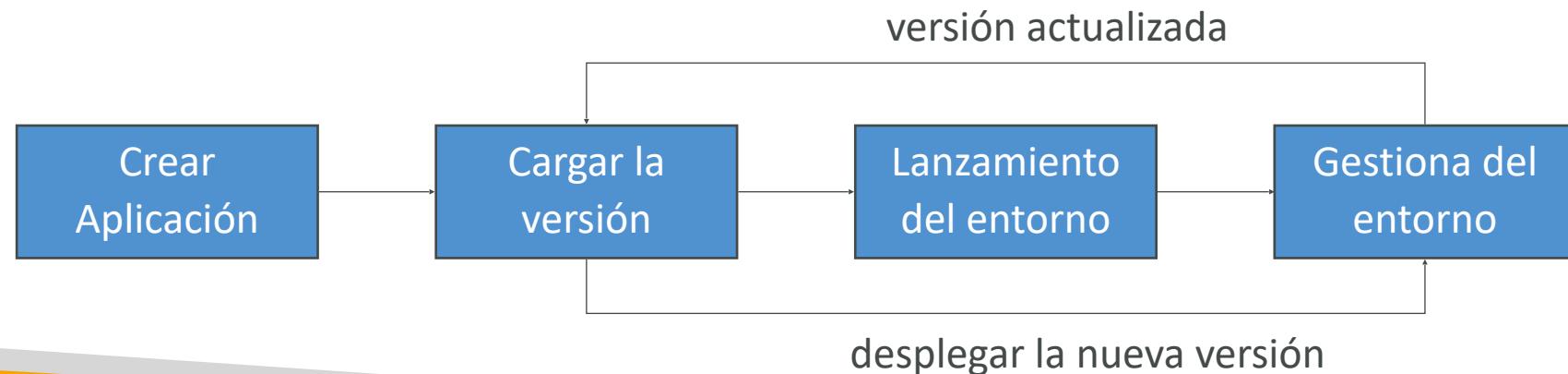
Elastic Beanstalk - Visión general



- Elastic Beanstalk es una visión centrada en el desarrollador de la implementación de una aplicación en AWS
- Utiliza todos los componentes que hemos visto antes: EC2, ASG, ELB, RDS, ...
- Servicio gestionado
 - Gestiona automáticamente el aprovisionamiento de capacidad, el equilibrio de carga, el escalado, la supervisión del estado de la aplicación, la configuración de las instancias, ...
 - Sólo el código de la aplicación es responsabilidad del desarrollador
 - Seguimos teniendo el control total de la configuración
 - Beanstalk es gratis pero pagas por las instancias subyacentes

Elastic Beanstalk - Componentes

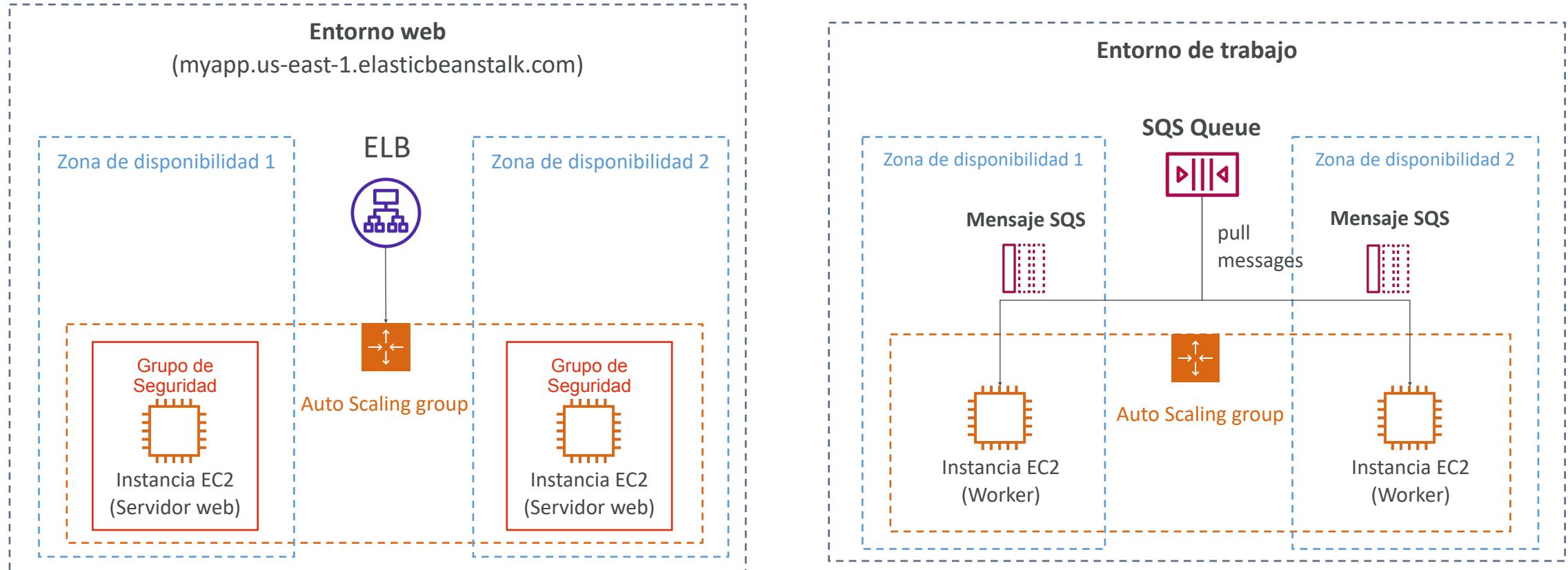
- **Aplicación:** colección de componentes de Elastic Beanstalk (entornos, versiones, configuraciones, ...)
- **Versión de la aplicación:** una iteración del código de tu aplicación
- **Entorno**
 - Colección de recursos de AWS que ejecutan una versión de la aplicación (sólo una versión de la aplicación a la vez)
 - **Niveles:** Nivel de entorno del servidor web y nivel de entorno del trabajador
 - Puedes crear varios entornos (dev, test, prod, ...)



Elastic Beanstalk - Plataformas soportadas

- Go
- Java SE
- Java con Tomcat
- .NET Core en Linux
- .NET en Windows Server
- Node.js
- PHP
- Python
- Ruby
- Packer Builder
- Contenedor único Docker
- Docker multicontenedor
- Docker Preconfigurado
- Si no tiene soporte, puedes escribir tu plataforma personalizada (avanzada)

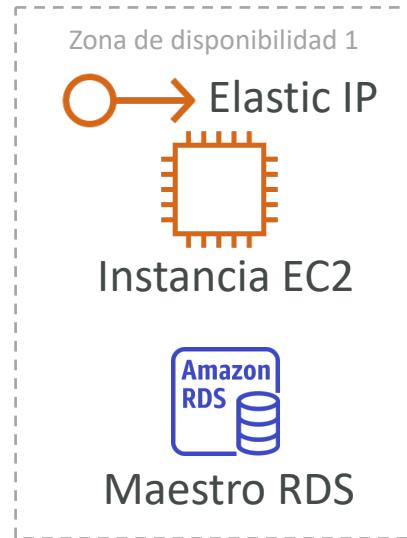
Nivel de entorno web vs. entorno de trabajo



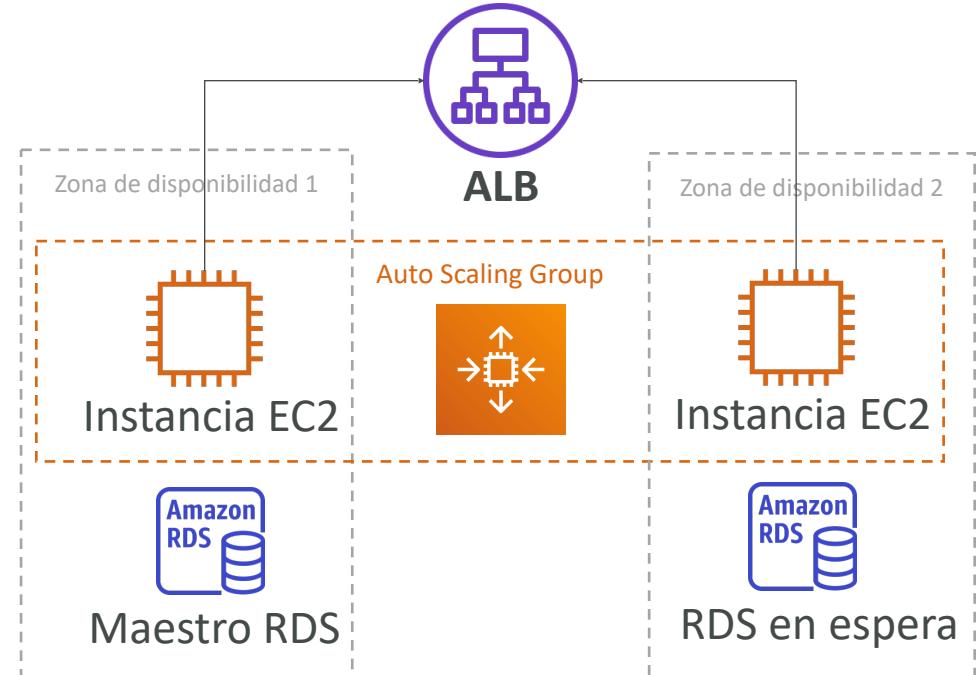
- Escala basada en el número de mensajes SQS
- Puede enviar mensajes a la cola SQS desde otro nivel de servidor web

Modos de despliegue de Elastic Beanstalk

Instancia única
Ideal para desarrolladores



Alta Disponibilidad con Load Balancer
Genial para prod



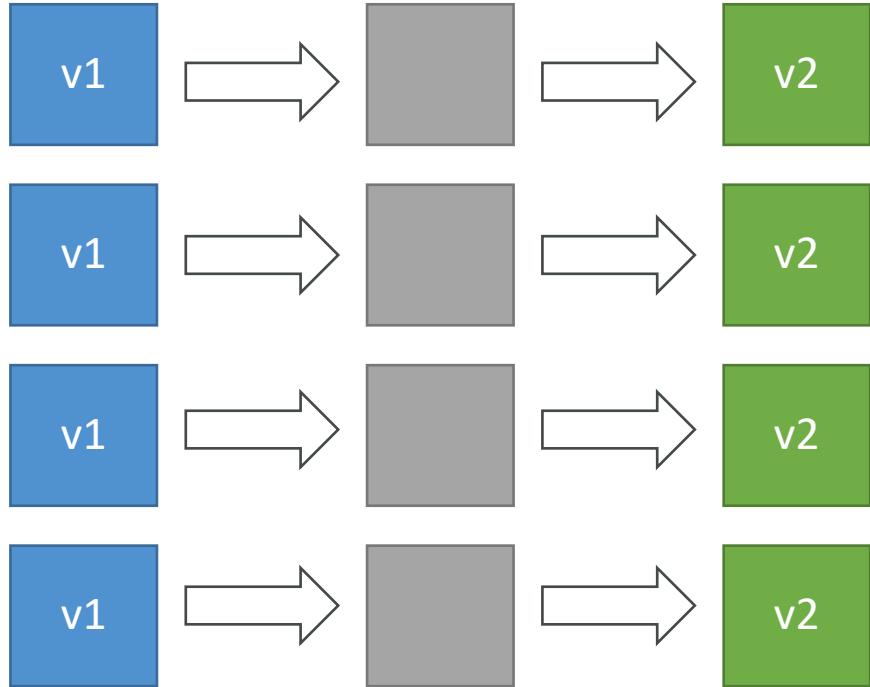
Opciones de despliegue de actualizaciones de Beanstalk

- **Todo a la vez (desplegar todo de una vez)** - más rápido, pero las instancias no están disponibles para servir tráfico durante un tiempo (tiempo de inactividad)
- **Continuo (Rolling)**: actualiza unas pocas instancias a la vez (bucket), y luego pasa al siguiente bucket una vez que el primero esté sano
- **Continuo (Rolling) con lotes adicionales**: es como el rolling, pero crea nuevas instancias para mover el lote (de modo que la aplicación antigua siga disponible).
- **Inmutable**: crea nuevas instancias en un nuevo ASG, despliega la versión en esas instancias y luego intercambia todas las instancias cuando todo esté en orden.
- **Blue/Green**: crea un nuevo entorno y cámbialo cuando esté listo
- **División del tráfico**: Test “canario” - envía un pequeño % del tráfico al nuevo despliegue

Despliegue de Elastic Beanstalk

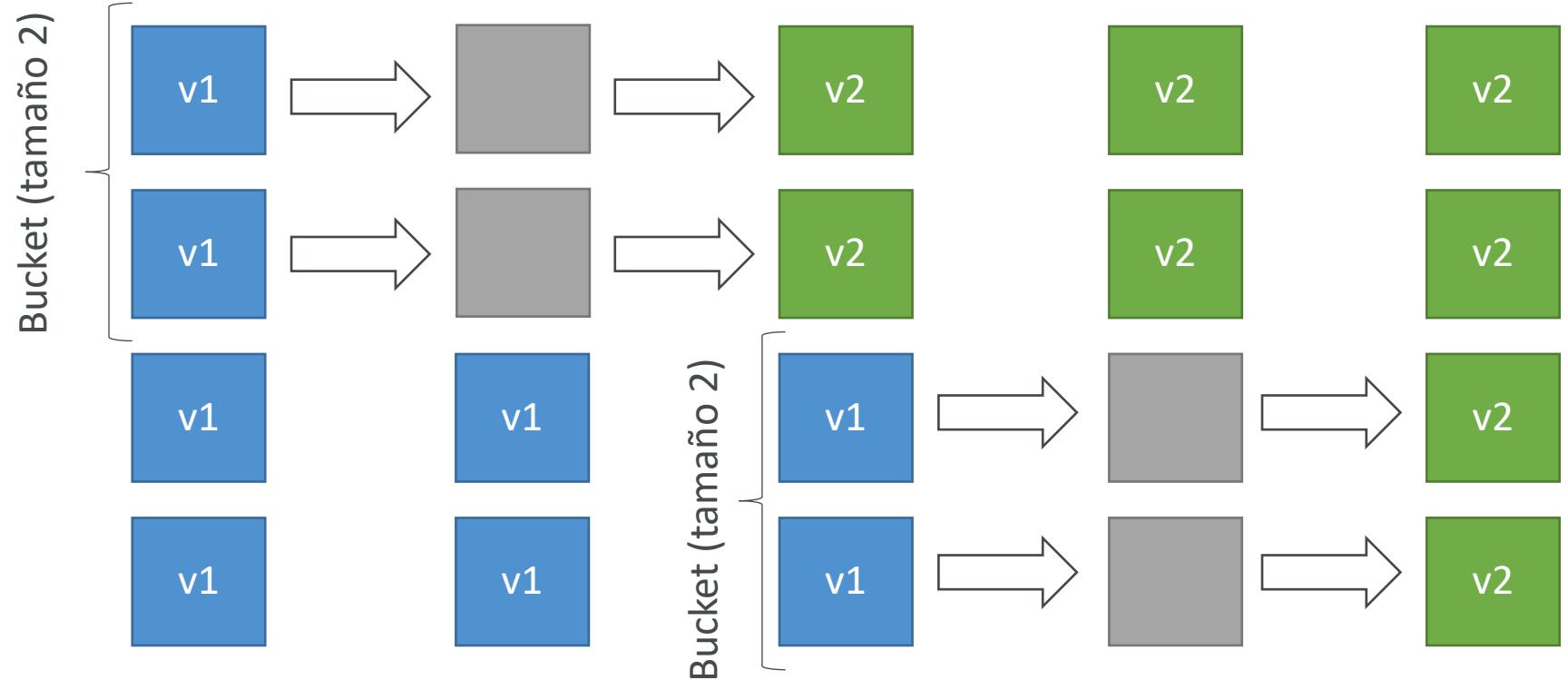
Todo a la vez

- Despliegue más rápido
- La aplicación tiene tiempo de inactividad
- Ideal para iteraciones rápidas en entornos de desarrollo
- Sin coste adicional



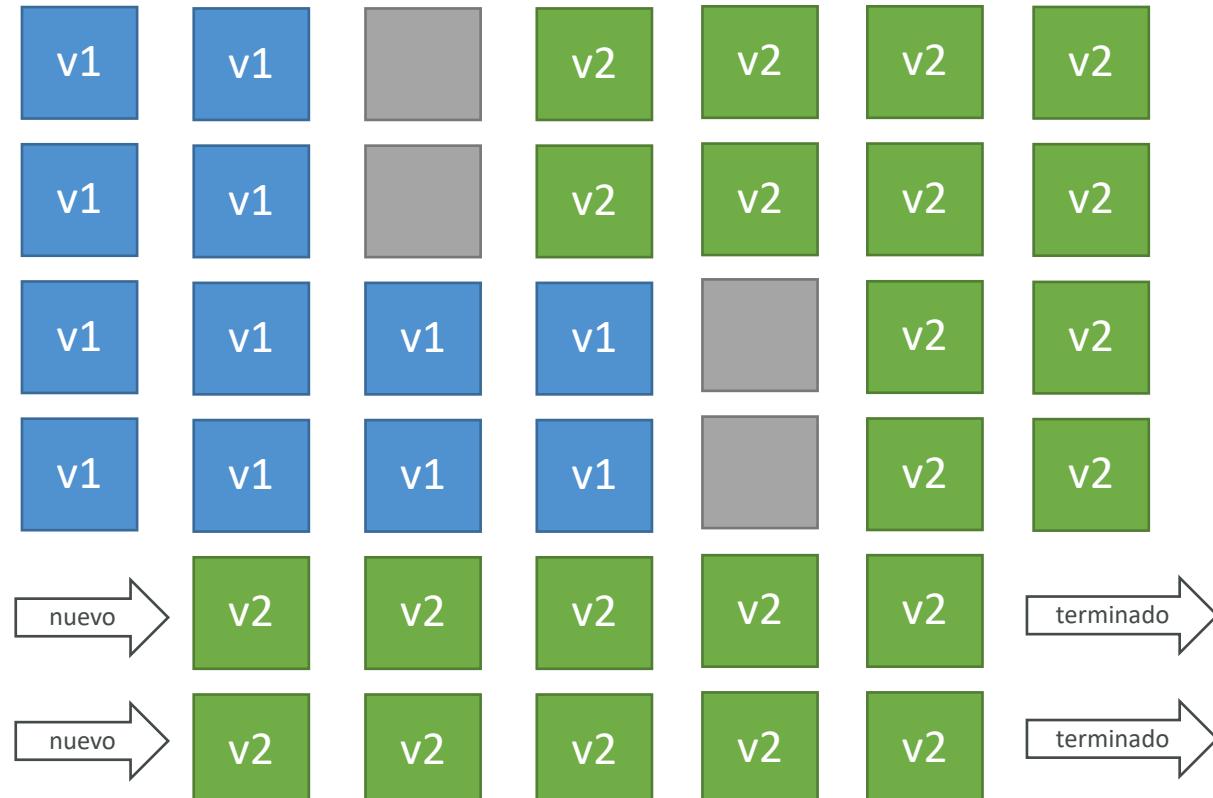
Despliegue de Elastic Beanstalk Continuo (Rolling)

- La aplicación funciona por debajo de su capacidad
 - Puedes ajustar el tamaño del bucket
 - La aplicación está ejecutando ambas versiones simultáneamente
 - Sin coste adicional
 - Despliegue prolongado



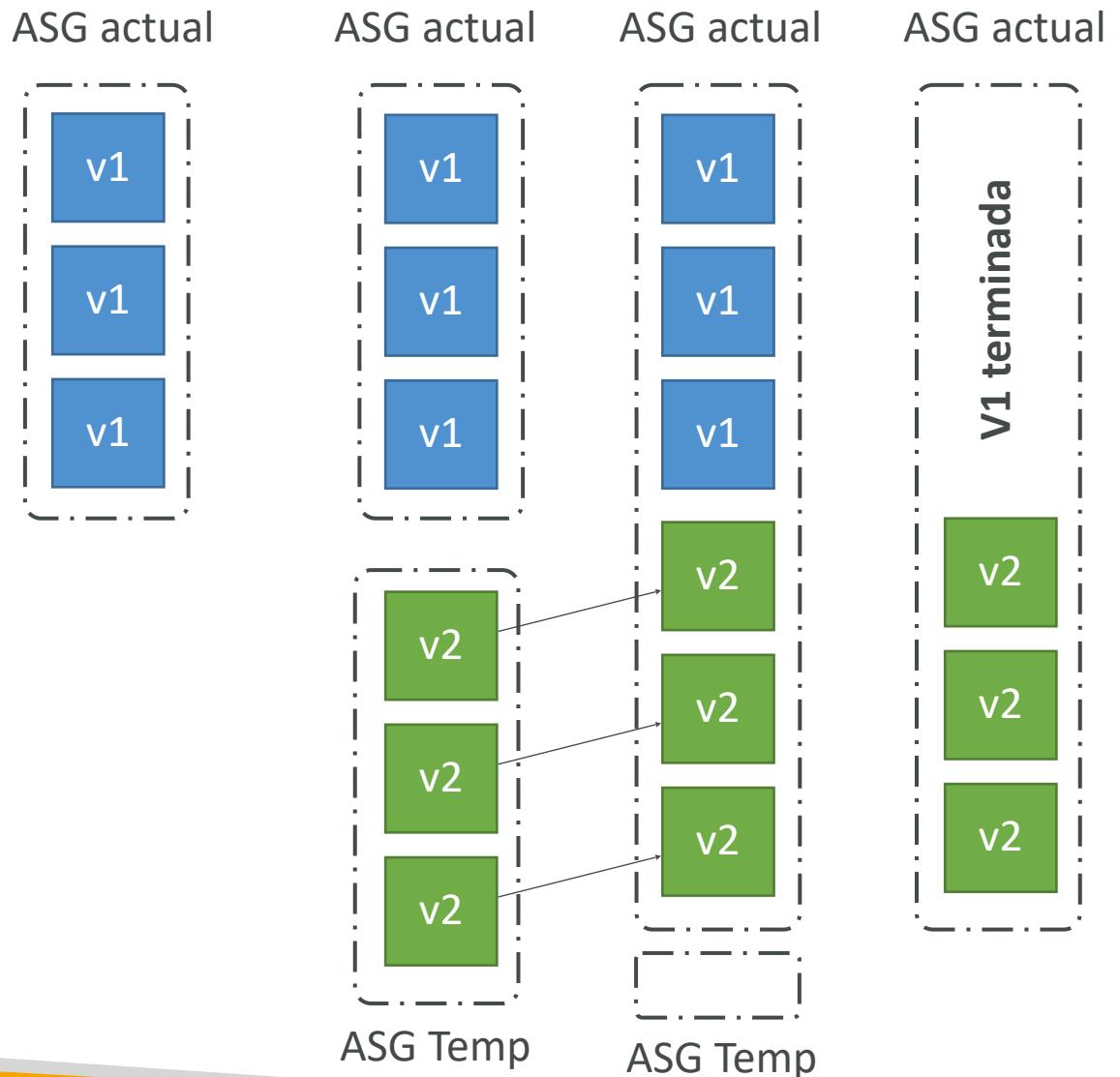
Despliegue de Elastic Beanstalk Continuo (Rolling) con lotes adicionales

- La aplicación funciona al máximo de su capacidad
- Puedes ajustar el tamaño del bucket
- La aplicación ejecuta ambas versiones simultáneamente
- Pequeño coste adicional
- El lote adicional se elimina al final del despliegue
- Despliegue más largo
- Bueno para prod



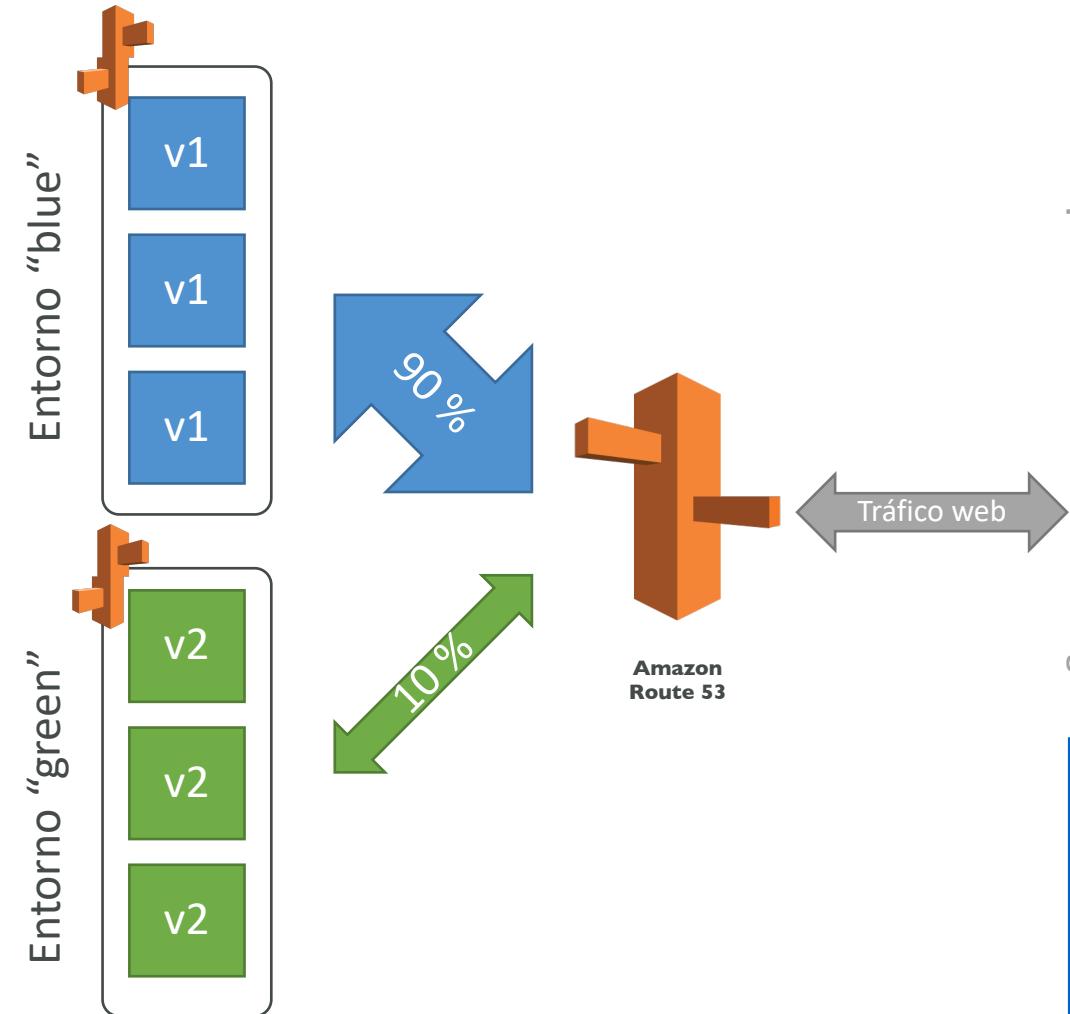
Despliegue de Elastic Beanstalk Immutable

- Tiempo de inactividad cero
- El nuevo código se despliega en nuevas instancias en un ASG temporal
- Alto coste, doble capacidad
- Despliegue más largo
- Retroceso rápido en caso de fallos (basta con dar de baja el nuevo ASG)
- Ideal para producción



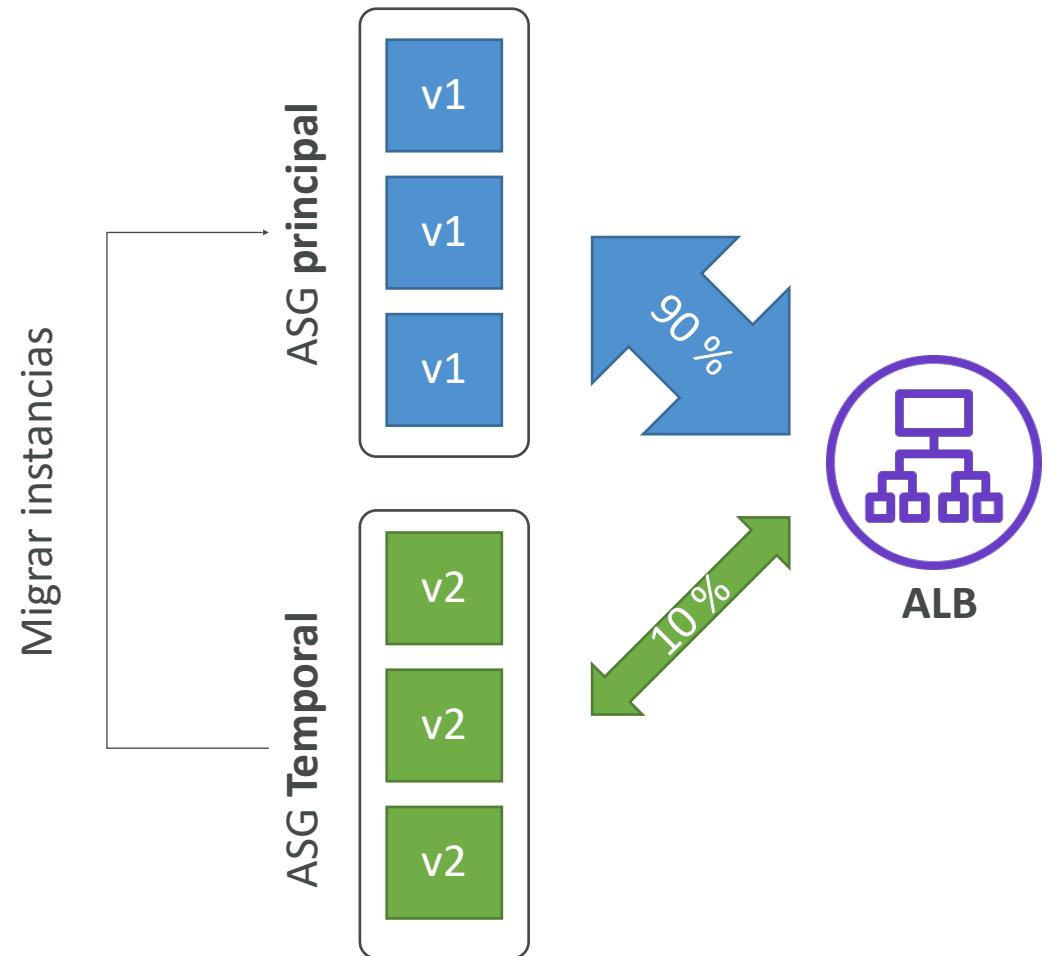
Despliegue de Elastic Beanstalk Blue / Green

- No es una "característica directa" de Elastic Beanstalk
- Cero tiempo de inactividad y facilidad de liberación
- Crea un nuevo entorno "escenario" y despliega allí la v2
- El nuevo entorno (verde) se puede validar de forma independiente y hacer rollback si surgen problemas
- Route 53 puede configurarse utilizando políticas ponderadas para redirigir un poco de tráfico al entorno de escenario
- Utilizando Beanstalk, "intercambia URLs" cuando termines de probar el entorno



Elastic Beanstalk - División del tráfico

- **Tests en “Canario”**
- Se despliega una nueva versión de la aplicación en un ASG temporal con la misma capacidad
- Se envía un pequeño % del tráfico al ASG temporal durante un tiempo configurable
- Se supervisa la salud del despliegue
- Si se produce un fallo en el despliegue, se activa una **reversión automática (muy rápida)**
- No hay tiempo de inactividad de la aplicación
- Las nuevas instancias se migran del ASG temporal al original
- Se cancela la versión antigua de la aplicación



Resumen de implementación de Elastic Beanstalk de AWS Doc

- <https://docs.aws.amazon.com/elasticbeanstalk/latest/dg/using-features.deploy-existing-version.html>

Métodos de implementación						
Método	Repercusión de la implementación con errores	Tiempo de implementación	Sin inactividad	Sin cambios de DNS	Proceso de restauración	El código se implementa en
Todo a la vez	Tiempo de inactividad.	⊕	✗ No	✓ Sí	Nueva implementación manual	Instancias existentes
Continua	Un lote fuera de servicio y los lotes implementados correctamente antes del error ejecutan la nueva versión de la aplicación.	⊕ ⊕ †	✓ Sí	✓ Sí	Nueva implementación manual	Instancias existentes
Continua con un lote adicional	Mínima si se produce un error en el primer lote; de lo contrario, similar a Rolling (Continua) .	⊕ ⊕ ⊕ †	✓ Sí	✓ Sí	Nueva implementación manual	Instancias nuevas y existentes
Inmutable	Mínima	⊕ ⊕ ⊕ ⊕	✓ Sí	✓ Sí	Terminar nuevas instancias	Instancias nuevas
División de tráfico	Porcentaje de tráfico del cliente dirigido a la nueva versión afectada temporalmente	⊕ ⊕ ⊕ ⊕ ††	✓ Sí	✓ Sí	Redireccionar tráfico y finalizar nuevas instancias	Instancias nuevas
Azul/verde	Mínima	⊕ ⊕ ⊕ ⊕	✓ Sí	✗ No	Intercambio de URL	Instancias nuevas

CLI de Elastic Beanstalk

- Podemos instalar una CLI adicional llamada "EB cli" que facilita el trabajo con Beanstalk desde la CLI
- Los comandos básicos son:
 - eb create
 - eb status
 - eb health
 - eb events
 - eb logs
 - eb open
 - eb deploy
 - eb config
 - eb terminate
- ¡Es útil para tus procesos automatizados de despliegue!

Proceso de despliegue de Elastic Beanstalk

- Describe las dependencias (requirements.txt para Python, package.json para Node.js)
- Empaqueta el código como zip y describe las dependencias
 - Python: requirements.txt
 - Node.js: package.json
- **Consola:** carga el archivo zip (crea una nueva versión de la aplicación), y luego despliega
- **CLI:** crea una nueva versión de la aplicación utilizando la CLI (carga el archivo zip) y, a continuación, despliega la aplicación.
- Elastic Beanstalk desplegará el zip en cada instancia EC2, resolverá las dependencias e iniciará la aplicación

Política del ciclo de vida de Beanstalk

- Elastic Beanstalk puede almacenar como máximo 1000 versiones de aplicaciones
- Si no eliminas las versiones antiguas, ya no podrás desplegarlas
- Para eliminar versiones antiguas de aplicaciones, utiliza una **política de ciclo de vida**
 - Basada en el tiempo (se eliminan las versiones antiguas)
 - Basada en el espacio (cuando tienes demasiadas versiones)
- Las versiones que se utilizan actualmente no se eliminarán
- Opción de no eliminar el paquete fuente en S3 para evitar la pérdida de datos

Extensiones de Elastic Beanstalk

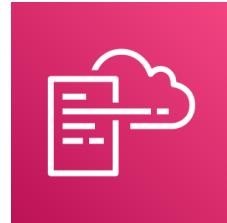
- Hay que desplegar en Elastic Beanstalk un archivo zip que contenga nuestro código
- Todos los parámetros establecidos en la interfaz de usuario pueden configurarse con código mediante archivos
- Requisitos:
 - En el directorio .ebextensions/ en la raíz del código fuente
 - Formato YAML / JSON
 - Extensiones **.config** (ejemplo: logs.config)
 - Posibilidad de modificar algunos ajustes por defecto mediante: option_settings
 - Posibilidad de añadir recursos como RDS, ElastiCache, DynamoDB, etc...
- Los recursos gestionados por las extensiones .ebextensions se eliminan si el entorno desaparece

Elastic Beanstalk bajo el capó

- Bajo el capó, Elastic Beanstalk depende de CloudFormation
- CloudFormation se utiliza para aprovisionar otros servicios de AWS (lo veremos más adelante)



Elastic Beanstalk



CloudFormation

- Caso de uso: puedes definir recursos CloudFormation en tus **.ebextensions** para aprovisionar ElastiCache, un bucket S3, ¡lo que quieras!
- ¡Vamos a echar un vistazo!

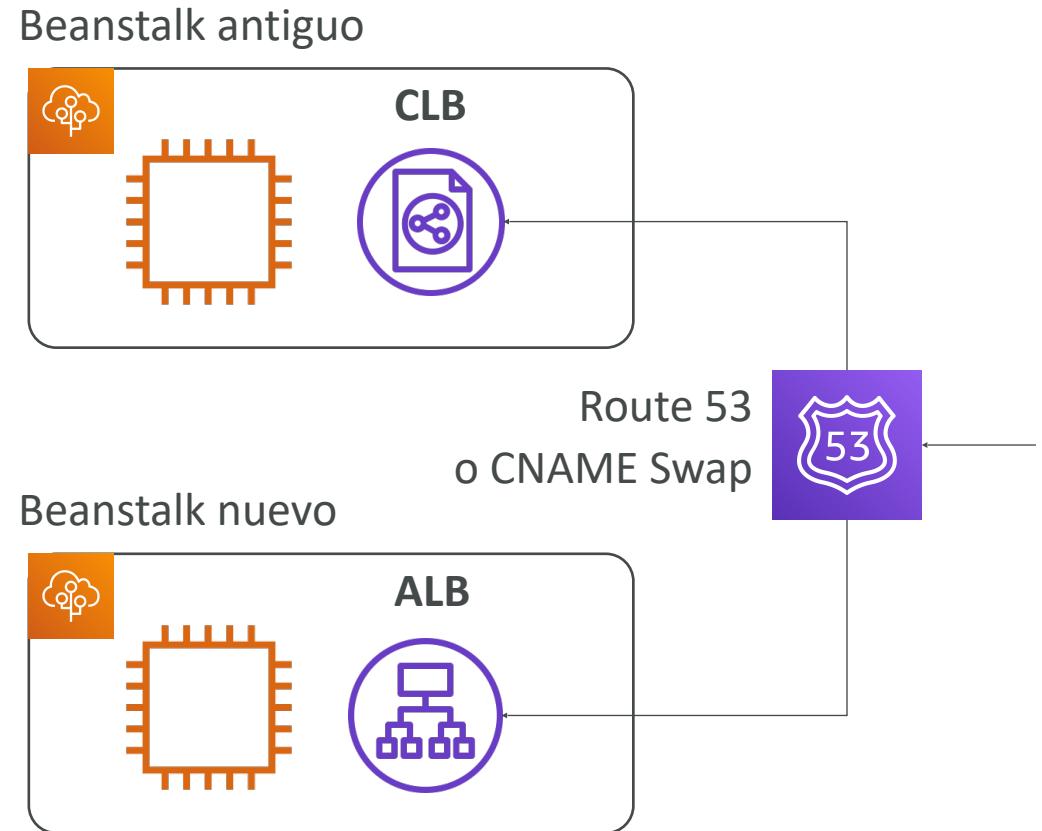
Clonación de Elastic Beanstalk

- Clona un entorno con exactamente la misma configuración
- Útil para desplegar una versión "de test" de tu aplicación
- Se conservan todos los recursos y la configuración:
 - Tipo y configuración del Load Balancer
 - Tipo de base de datos RDS (pero no se conservan los datos)
 - Variables de entorno
- Después de clonar un entorno, puedes cambiar la configuración



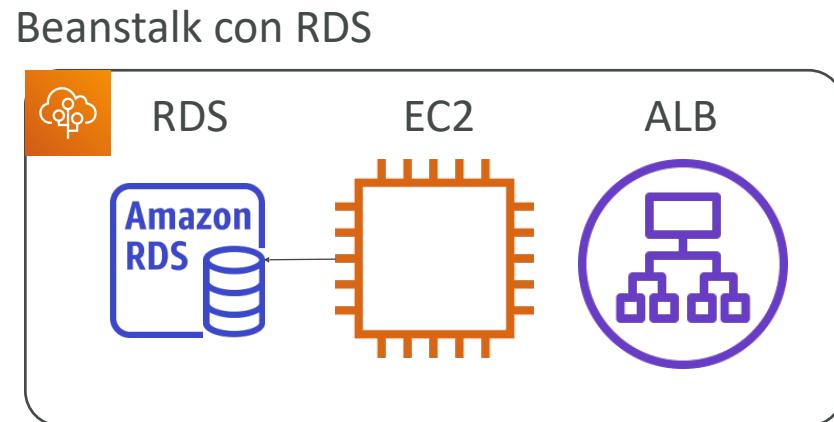
Migración a Elastic Beanstalk: Load Balancer

- Despues de crear un entorno Elastic Beanstalk, **no puedes cambiar el tipo de Elastic Load Balancer** (sólo la configuración)
- Para migrar
 1. crea un nuevo entorno con la misma configuración excepto LB (no se puede clonar)
 2. despliega tu aplicación en el nuevo entorno
 3. realiza un intercambio de CNAME o una actualización de Route 53



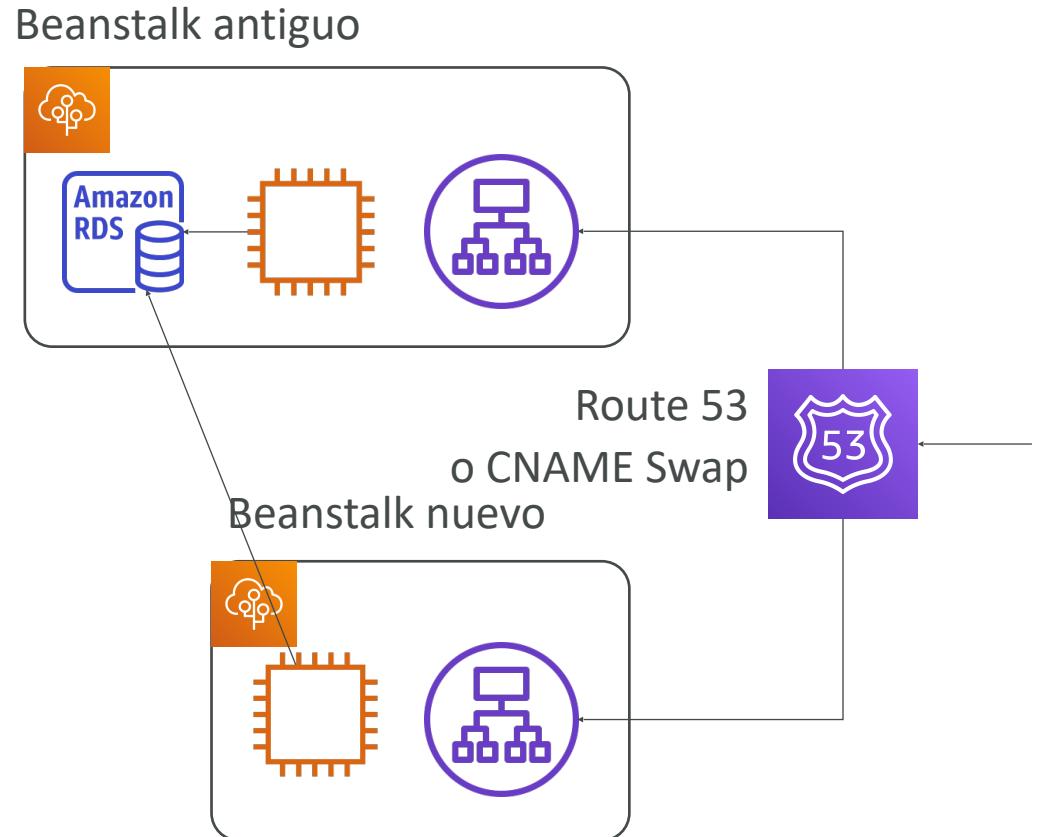
RDS con Elastic Beanstalk

- RDS puede aprovisionarse con Beanstalk, lo que es estupendo para desarrollo y tests.
- Esto no es bueno para producción, ya que el ciclo de vida de la base de datos está ligado al ciclo de vida del entorno Beanstalk.
- Lo mejor para prod es crear por separado una base de datos RDS y proporcionar a nuestra aplicación EB la cadena de conexión.



Migración a Elastic Beanstalk: Desacoplar RDS

1. Crea una Snapshot de la BD RDS (como salvaguarda)
2. Ve a la consola RDS y protege la base de datos RDS para que no se borre
3. Crea un nuevo entorno Elastic Beanstalk, sin RDS, apunta tu aplicación al RDS existente
4. Realiza un intercambio CNAME (blue/green) o una actualización Route 53, confirma que funciona
5. Finaliza el entorno antiguo (el RDS no se eliminará)
6. Elimina el stack de CloudFormation (en estado `DELETE_FAILED`)



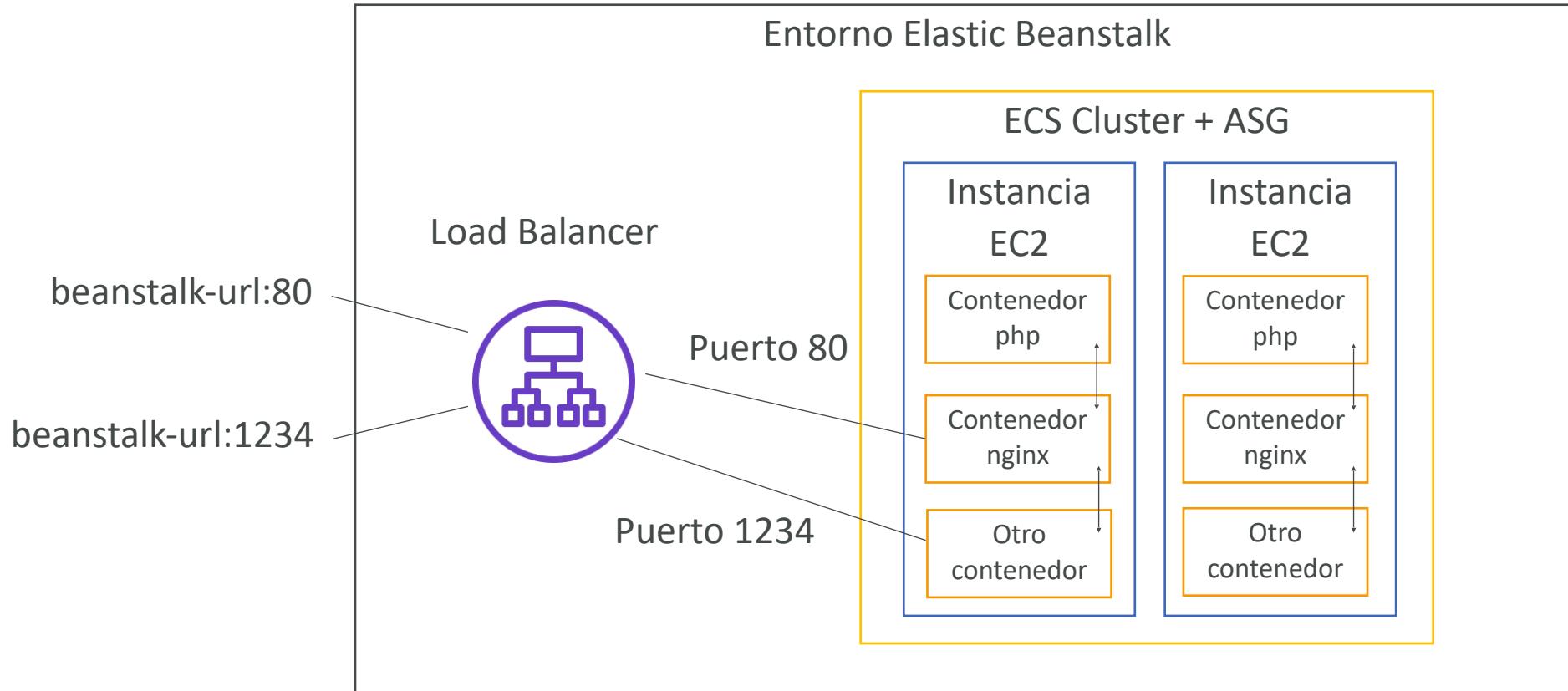
Elastic Beanstalk - Un solo Docker

- Ejecuta tu aplicación como un único contenedor Docker
- O bien proporciona
 - **Dockerfile**: Elastic Beanstalk creará y ejecutará el contenedor Docker
 - **Dockerrun.aws.json (v1)**: Describe dónde está la imagen Docker *ya construida*
 - Imagen
 - Puertos
 - Volúmenes
 - Logs
 - Etc...
- **Beanstalk en un único contenedor Docker no utiliza ECS**

Elastic Beanstalk - Contenedor Docker Múltiple

- Multi Docker ayuda a ejecutar múltiples contenedores por instancia EC2 en EB
- Esto creará para ti
 - Cluster ECS
 - Instancias EC2, configuradas para utilizar el Cluster ECS
 - Load Balancer (en modo de alta disponibilidad)
 - Definición y ejecución de tareas
- Requiere un config **Dockerrun.aws.json (v2)** en la raíz del código fuente
- **Dockerrun.aws.json** se utiliza para generar la **definición de la tarea ECS**
- Tus imágenes Docker deben estar preconstruidas y almacenadas en ECR, por ejemplo

Elastic Beanstalk + Multi Docker ECS



Elastic Beanstalk y HTTPS

- **Beanstalk con HTTPS**

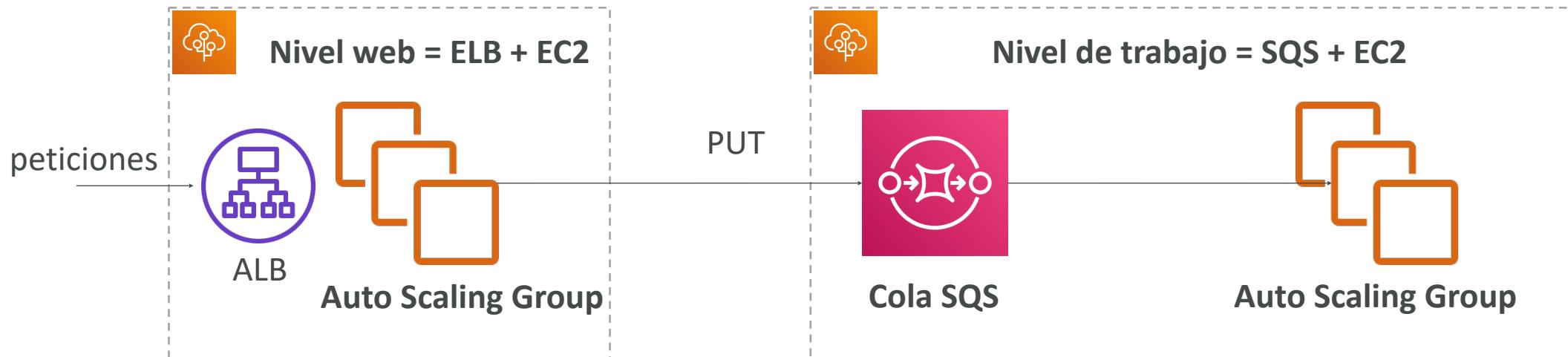
- Idea: Carga el certificado SSL en el Load Balancer
- Puede hacerse desde la Consola (consola EB, configuración del Load Balancer)
- Se puede hacer desde el código .ebextensions/securelistener-alb.config
- El certificado SSL se puede aprovisionar mediante ACM (AWS Certificate Manager) o CLI
- Debes configurar una regla de grupo de seguridad para permitir la entrada del puerto 443 (puerto HTTPS)

- **Beanstalk redirige HTTP a HTTPS**

- Configura tus instancias para redirigir HTTP a HTTPS: <https://github.com/awsdocs/elastic-beanstalk-samples/tree/master/configuration-files/aws-provided/security-configuration/https-redirect>
- O configura el Application Load Balancer (sólo ALB) con una regla
- Asegúrate de que las comprobaciones de estado no se redirigen (para que sigan dando 200 OK)

Servidor web vs entorno de trabajo

- Si tu aplicación realiza tareas que tardan mucho en completarse, descarga estas tareas a un **entorno de trabajador dedicado**
- **Desacoplar** tu aplicación en dos niveles es habitual
- Ejemplo: procesar un vídeo, generar un archivo zip, etc.
- Puedes definir tareas periódicas en un archivo cron.yaml



Elastic Beanstalk - Plataforma personalizada (Avanzada)

- Las plataformas personalizadas son muy avanzadas, permiten definir desde cero:
 - El Sistema Operativo (SO)
 - Software adicional
 - Scripts que Beanstalk ejecuta en estas plataformas
- **Caso de uso:** el lenguaje de la aplicación es incompatible con Beanstalk y no utiliza Docker
- Para crear tu propia plataforma
 - Define una AMI utilizando el archivo **Platform.yaml**
 - Construye esa plataforma utilizando el **software Packer (herramienta de código abierto para crear AMIs)**
- Plataforma personalizada frente a imagen personalizada (AMI):
 - La imagen personalizada es para modificar una Plataforma Beanstalk existente
 - Python, Node.js, Java...
 - Plataforma personalizada es crear una plataforma Beanstalk completamente nueva

AWS CloudFormation

Gestionar tu infraestructura como código

Infraestructura como código

- Actualmente, hemos estado haciendo mucho trabajo manual
- Todo este trabajo manual será muy difícil de reproducir:
 - En otra región
 - En otra cuenta de AWS
 - Dentro de la misma región si se borrara todo
- ¿No sería genial que toda nuestra infraestructura fuera... código?
- Ese código se desplegaría y crearía / actualizaría / eliminaría nuestra infraestructura

Qué es CloudFormation

- CloudFormation es una forma declarativa de esbozar tu Infraestructura de AWS, para cualquier recurso (la mayoría de ellos están soportados).
- Por ejemplo, dentro de una plantilla de CloudFormation, dices:
 - Quiero un grupo de seguridad
 - Quiero dos máquinas EC2 que utilicen este grupo de seguridad
 - Quiero dos IPs elásticas para estas máquinas EC2
 - Quiero un bucket S3
 - Quiero un Load Balancer (ELB) delante de estas máquinas
- Entonces CloudFormation los crea por ti, en el **orden correcto**, con la configuración **exacta que especifiques**

Ventajas de AWS CloudFormation (1/2)

- Infraestructura como código
 - No se crean recursos manualmente, lo que es excelente para el control
 - El código puede controlarse mediante versiones, por ejemplo, utilizando git
 - Los cambios en la infraestructura se revisan a través del código
- Coste
 - Cada recurso dentro de la pila se etiqueta con un identificador para que puedas ver fácilmente cuánto te cuesta un stack
 - Puedes estimar los costes de tus recursos utilizando la plantilla CloudFormation
 - Estrategia de ahorro: en Dev, podrías automatizar la eliminación de plantillas a las 5 de la tarde y volver a crearlas a las 8 de la mañana, de forma segura

Ventajas de AWS CloudFormation (2/2)

- Productividad
 - Capacidad para destruir y volver a crear una infraestructura en el Cloud sobre la marcha
 - Generación automatizada de diagramas para tus plantillas
 - Programación declarativa (sin necesidad de averiguar el orden y la orquestación)
- Separación de intereses: crea muchos stacks para muchas aplicaciones, y muchas capas. Ej:
 - Stack 1:VPC
 - Stack 2: Red
 - Stack 3: Aplicaciones
- No reinvenes la rueda
 - Aprovecha las plantillas existentes en la web
 - Aprovecha la documentación

Cómo funciona CloudFormation

- Las plantillas tienen que cargarse en S3 y luego referenciarse en CloudFormation
- Para actualizar una plantilla, no podemos editar las anteriores. Tenemos que volver a subir una nueva versión de la plantilla a AWS
- Los stacks se identifican con un nombre
- Al borrar un stack se borran todos y cada uno de los artefactos creados por CloudFormation.

Desplegar plantillas de CloudFormation

- De forma manual:
 - Edición de plantillas en el Diseñador de CloudFormation
 - Utilizando la consola para introducir parámetros, etc.
- De forma automatizada:
 - Editando plantillas en un archivo YAML
 - Usando la CLI (Interfaz de Línea de Comandos) de AWS para desplegar las plantillas
 - Forma recomendada cuando quieres automatizar completamente tu flujo

Bloques de CloudFormation

Componentes de las plantillas (una sección del curso para cada uno):

I. **Recursos: tus recursos AWS declarados en la plantilla (OBLIGATORIO)**

2. Parámetros: las entradas dinámicas de tu plantilla
3. Asignaciones: las variables estáticas de tu plantilla
4. Salidas: Referencias a lo que se ha creado
5. Condicionales: Lista de condiciones para realizar la creación de recursos
6. Metadatos

Plantillas de ayuda:

1. Referencias
2. Funciones

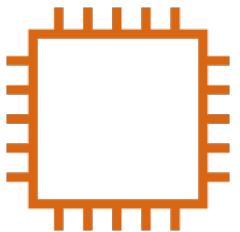
Nota:

Esto es una introducción a CloudFormation

- Aprender y dominar correctamente CloudFormation puede llevar más de 3 horas.
 - Esta sección está pensada para que te hagas una idea de cómo funciona.
 - Seremos un poco menos prácticos que en otras secciones
-
- Aprenderemos todo lo necesario para responder a las preguntas del examen
 - El examen no requiere que escribas realmente CloudFormation
 - El examen espera que entiendas cómo leer CloudFormation

Ejemplo introductorio

- Vamos a crear una instancia EC2 sencilla.
 - Luego vamos a crearla para añadirle una IP elástica
 - Y vamos a añadirle dos grupos de seguridad
 - Por ahora, olvídate de la sintaxis del código.
 - Veremos la estructura de los archivos más adelante
-
- ¡Veremos cómo en un abrir y cerrar de ojos somos capaces de empezar a trabajar con CloudFormation!



Instancia EC2

Curso acelerado de YAML

```
1  invoice:      34843
2  date   :     2001-01-23
3  bill-to:
4    given  :   Chris
5    family :  Dumars
6    address:
7      lines: |
8        458 Walkman Dr.
9        Suite #292
10       city   : Royal Oak
11       state  : MI
12       postal : 48046
13 product:
14   - sku        : BL394D
15     quantity   : 4
16     description: Basketball
17     price      : 450.00
18   - sku        : BL4438H
19     quantity   : 1
20     description: Super Hoop
21     price      : 2392.00
```

- YAML y JSON son los lenguajes que puedes utilizar para CloudFormation.
 - **JSON es horrible para CF**
 - **YAML es genial en muchos sentidos**
 - ¡Aprendamos un poco sobre ello!
-
- Pares clave-valor
 - Objetos anidados
 - Soporta matrices
 - Strings de varias líneas
 - Puedes incluir comentarios

¿Qué son los recursos?

- Los recursos son el núcleo de tu plantilla de CloudFormation (OBLIGATORIO)
 - Representan los diferentes Componentes de AWS que se crearán y configurarán
 - Los recursos se declaran y pueden referenciarse entre sí
-
- AWS se encarga de crear, actualizar y eliminar recursos por nosotros
 - Hay más de 224 tipos de recursos (!)
 - Los identificadores de los tipos de recursos son de la forma:

AWS::aws-product-name::data-type-name

¿Cómo encuentro la documentación de los recursos?

- No puedo enseñarte todos los 224 recursos, pero puedo enseñarte a aprender a utilizarlos.
- Puedes encontrar todos los recursos aquí:
[http://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/
aws-template-resource-type-ref.html](http://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/aws-template-resource-type-ref.html)
- Entonces, nos limitamos a leer la documentación ☺.
- Ejemplo aquí (para una instancia EC2):
[http://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/
aws-properties-ec2-instance.html](http://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/aws-properties-ec2-instance.html)

Análisis de la plantilla CloudFormation

- Volviendo al ejemplo de la sección introductoria, veamos por qué se escribió así.
- Puedes encontrar la documentación pertinente aquí:
 - <http://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/aws-properties-ec2-instance.html>
 - <http://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/aws-properties-ec2-security-group.html>
 - <http://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/aws-properties-ec2-eip.html>

Preguntas frecuentes sobre recursos - FAQ

- ¿Puedo crear una cantidad dinámica de recursos?

➤ No, no puedes. Todo en la plantilla de CloudFormation tiene que estar declarado. No puedes realizar la generación de código allí

- ¿Se soportan todos los servicios de AWS?

➤ Casi. Sólo unos pocos nichos selectos no están ahí todavía

➤ Puedes evitarlo utilizando los recursos personalizados de AWS Lambda

¿Qué son los parámetros?

- Los parámetros son una forma de proporcionar entradas a tu plantilla de AWS CloudFormation
- Es importante conocerlos si
 - Quieres reutilizar tus plantillas en toda la empresa
 - Algunas entradas no pueden determinarse de antemano
- Los parámetros son extremadamente potentes, controlables y pueden evitar que se produzcan errores en tus plantillas gracias a los tipos.

¿Cuándo debes utilizar un parámetro?

- Pregúntate lo siguiente:
 - ¿Es probable que esta configuración de recursos de CloudFormation cambie en el futuro?
 - Si es así, conviértelo en un parámetro.
- No tendrás que volver a cargar una plantilla para cambiar su contenido ☺.

Parameters:

SecurityGroupDescription:

Description: Security Group Description
(Simple parameter)

Type: String

Configuración de los parámetros

Todos estos ajustes pueden controlar los parámetros:

- **Tipo:**
 - String
 - Número
 - Lista delimitada por comas
 - Lista <Tipo>
 - Parámetro de AWS (para ayudar a detectar valores no válidos - comparar con valores existentes en la cuenta de AWS)
- **Descripción**
- **Restricciones**
 - ConstraintDescription (String)
 - Longitud Mín/Máx
 - Valor Mín/Máx
 - Valores por defecto
 - AllowedValues (matriz)
 - AllowedPattern (regexp)
 - NoEcho (booleano)

Cómo hacer referencia a un parámetro

- La función Fn::Ref para hacer referencia a los parámetros
- Los parámetros se pueden utilizar en cualquier parte de una plantilla.
- La abreviatura de esto en YAML es !Ref
- La función también puede hacer referencia a otros elementos dentro de la plantilla

```
DbSubnet1:  
  Type: AWS::EC2::Subnet  
  Properties:  
    VpcId: !Ref MyVPC
```

Concepto: Pseudoparámetros

- AWS nos ofrece pseudoparámetros en cualquier plantilla de CloudFormation.
- Se pueden utilizar en cualquier momento y están activados por defecto

Valor de referencia	Ejemplo de valor de retorno
AWS::AccountId	1234567890
AWS::NotificationARNS	[arn:aws:sns:us-east-1:123456789012:MyTopic]
AWS::NoValue	Does not return a value.
AWS::Region	us-east-2
AWS::StackId	arn:aws:cloudformation:us-east-1:123456789012:stack/MyStack/1c2fa620-982a-11e3-aff7-50e2416294e0
AWS::StackName	MyStack

¿Qué son los mapeos?

- Las asignaciones son variables fijas dentro de tu plantilla de CloudFormation.
- Son muy útiles para diferenciar entre distintos entornos (desarrollo y producción), regiones (regiones de AWS), tipos de AMI, etc.
- Todos los valores están codificados en la plantilla
- Ejemplo:

```
Mappings:  
  Mapping01:  
    Key01:  
      Name: Value01  
    Key02:  
      Name: Value02  
    Key03:  
      Name: Value03
```

```
RegionMap:  
  us-east-1:  
    "32": "ami-6411e20d"  
    "64": "ami-7a11e213"  
  us-west-1:  
    "32": "ami-c9c7978c"  
    "64": "ami-cfc7978a"  
  eu-west-1:  
    "32": "ami-37c2f643"  
    "64": "ami-31c2f645"
```

¿Cuándo utilizarías mapeos frente a parámetros?

- Los mapeos son estupendos cuando conoces de antemano todos los valores que se pueden tomar y que se pueden deducir de variables como
 - Región
 - Zona de Disponibilidad
 - Cuenta de AWS
 - Entorno (dev vs prod)
 - Etc...
- Permiten un control más seguro sobre la plantilla.
- Utiliza parámetros cuando los valores sean realmente específicos del usuario

Fn::FindInMap

Acceder a los valores de mapeo

- Utilizamos **Fn::FindInMap** para devolver un valor con nombre a partir de una clave concreta
- **!FindInMap [MapName, TopLevelKey, SecondLevelKey]**

```
AWSTemplateFormatVersion: "2010-09-09"
Mappings:
  RegionMap:
    us-east-1:
      "32": "ami-6411e20d"
      "64": "ami-7a11e213"
    us-west-1:
      "32": "ami-c9c7978c"
      "64": "ami-cfc7978a"
    eu-west-1:
      "32": "ami-37c2f643"
      "64": "ami-31c2f645"
    ap-southeast-1:
      "32": "ami-66f28c34"
      "64": "ami-60f28c32"
    ap-northeast-1:
      "32": "ami-9c03a89d"
      "64": "ami-a003a8a1"
Resources:
  myEC2Instance:
    Type: "AWS::EC2::Instance"
    Properties:
      ImageId: !FindInMap [RegionMap, !Ref "AWS::Region", 32]
      InstanceType: m1.small
```

¿Qué son los outputs (salidas)?

- La sección Outputs declara valores de salida *opcionales* que podemos importar a otros stacks (¡si los exportas primero!).
- También puedes ver las salidas en la consola de AWS o utilizando la CLI de AWS
- Son muy útiles, por ejemplo, si defines una red CloudFormation, y das salida a variables como el ID de la VPC y los ID de tus subredes
- Es la mejor forma de realizar alguna colaboración cross stack, ya que dejas que los expertos manejen su propia parte del stack
- No puedes eliminar un stack de CloudFormation si sus salidas están siendo referenciadas por otro stack de CloudFormation

Ejemplo de Outputs (salidas)

- Crear un grupo de seguridad SSH como parte de una plantilla
- Creamos una salida que haga referencia a ese grupo de seguridad

Outputs:

StackSSHSecurityGroup:

Description: The SSH Security Group for our Company

Value: !Ref MyCompanyWideSSHSecurityGroup

Export:

Name: SSHSecurityGroup

Referencia cruzada de Stack

- A continuación, creamos una segunda plantilla que aprovecha ese grupo de seguridad
- Para ello, utilizamos la función **Fn::ImportValue**
- No puedes eliminar el stack subyacente hasta que no se eliminan también todas las referencias.

```
Resources:  
  MySecureInstance:  
    Type: AWS::EC2::Instance  
    Properties:  
      AvailabilityZone: us-east-1a  
      ImageId: ami-a4c7edb2  
      InstanceType: t2.micro  
      SecurityGroups:  
        - !ImportValue SSHSecurityGroup
```

¿Para qué se utilizan las condiciones?

- Las condiciones se utilizan para controlar la creación de recursos o salidas en función de una condición.
- Las condiciones pueden ser lo que tú quieras, pero las más comunes son
 - Entorno (dev / test / prod)
 - Región de AWS
 - Cualquier valor de parámetro
- Cada condición puede hacer referencia a otra condición, valor de parámetro o asignación

¿Cómo definir una condición?

Conditions:

```
| CreateProdResources: !Equals [ !Ref EnvType, prod ]
```

- El ID lógico lo eliges tú. Es cómo nombras la condición
- La función intrínseca (lógica) puede ser cualquiera de las siguientes:
 - Fn::And
 - Fn::Equals
 - Fn::If
 - Fn::Not
 - Fn::Or

Utilizar una condición

- Las condiciones pueden aplicarse a recursos / salidas / etc...

```
Resources:
```

```
  MountPoint:
```

```
    Type: "AWS::EC2::VolumeAttachment"
```

```
    Condition: CreateProdResources
```

CloudFormation

Funciones intrínsecas que debes conocer

- Ref
- Fn::GetAtt
- Fn::FindInMap
- Fn::ImportValue
- Fn::Join
- Fn::Sub
- Funciones de condición (Fn::If, Fn::Not, Fn::Equals, etc...)

Fn::Ref

- La función Fn::Ref puede aprovecharse para hacer referencia a
 - Parámetros => devuelve el valor del parámetro
 - Recursos => devuelve el ID físico del recurso subyacente (ej: ID de EC2)
- La abreviatura para esto en YAML es !Ref

```
DbSubnet1:  
  Type: AWS::EC2::Subnet  
  Properties:  
    VpcId: !Ref MyVPC
```

Fn::GetAtt

- Los atributos se adjuntan a cualquier recurso que crees
- Para conocer los atributos de tus recursos, lo mejor es consultar la documentación.
- Por ejemplo: ¡la AZ de una máquina EC2!

```
Resources:
```

```
EC2Instance:
```

```
Type: "AWS::EC2::Instance"
```

```
Properties:
```

```
ImageId: ami-1234567
```

```
InstanceType: t2.micro
```

```
NewVolume:
```

```
Type: "AWS::EC2::Volume"
```

```
Condition: CreateProdResources
```

```
Properties:
```

```
Size: 100
```

```
AvailabilityZone:
```

```
!GetAtt EC2Instance.AvailabilityZone
```

Fn::FindInMap

Acceder a los valores de mapeo

- Utilizamos **Fn::FindInMap** para devolver un valor con nombre a partir de una clave concreta
- **!FindInMap [MapName , TopLevelKey , SecondLevelKey]**

```
AWSTemplateFormatVersion: "2010-09-09"
Mappings:
  RegionMap:
    us-east-1:
      "32": "ami-6411e20d"
      "64": "ami-7a11e213"
    us-west-1:
      "32": "ami-c9c7978c"
      "64": "ami-cfc7978a"
    eu-west-1:
      "32": "ami-37c2f643"
      "64": "ami-31c2f645"
    ap-southeast-1:
      "32": "ami-66f28c34"
      "64": "ami-60f28c32"
    ap-northeast-1:
      "32": "ami-9c03a89d"
      "64": "ami-a003a8a1"
Resources:
  myEC2Instance:
    Type: "AWS::EC2::Instance"
    Properties:
      ImageId: !FindInMap [RegionMap, !Ref "AWS::Region", 32]
      InstanceType: m1.small
```

Fn::ImportValue

- Importar valores exportados en otras plantillas
- Para ello, utilizamos la función **Fn::ImportValue**

```
Resources:  
  MySecureInstance:  
    Type: AWS::EC2::Instance  
    Properties:  
      AvailabilityZone: us-east-1a  
      ImageId: ami-a4c7edb2  
      InstanceType: t2.micro  
      SecurityGroups:  
        - !ImportValue SSHSecurityGroup
```

Fn::Join

- Unir valores con un delimitador

```
!Join [ delimiter, [ comma-delimited list of values ] ]
```

- Esto crea "a:b:c"

```
!Join [ ":", [ a, b, c ] ]
```

Función Fn::Sub

- Fn::Sub, o !Sub como abreviatura, se utiliza para sustituir variables de un texto. Es una función muy práctica que te permitirá personalizar completamente tus plantillas.
- Por ejemplo, puedes combinar Fn::Sub con Referencias o Pseudo variables AWS
- Los strings deben contener \${VariableName} y los sustituirá

```
!Sub
  - String
  - { Var1Name: Var1Value, Var2Name: Var2Value }
```

```
!Sub String
```

Funciones de condición

Conditions:

| CreateProdResources: !Equals [!Ref EnvType, prod]

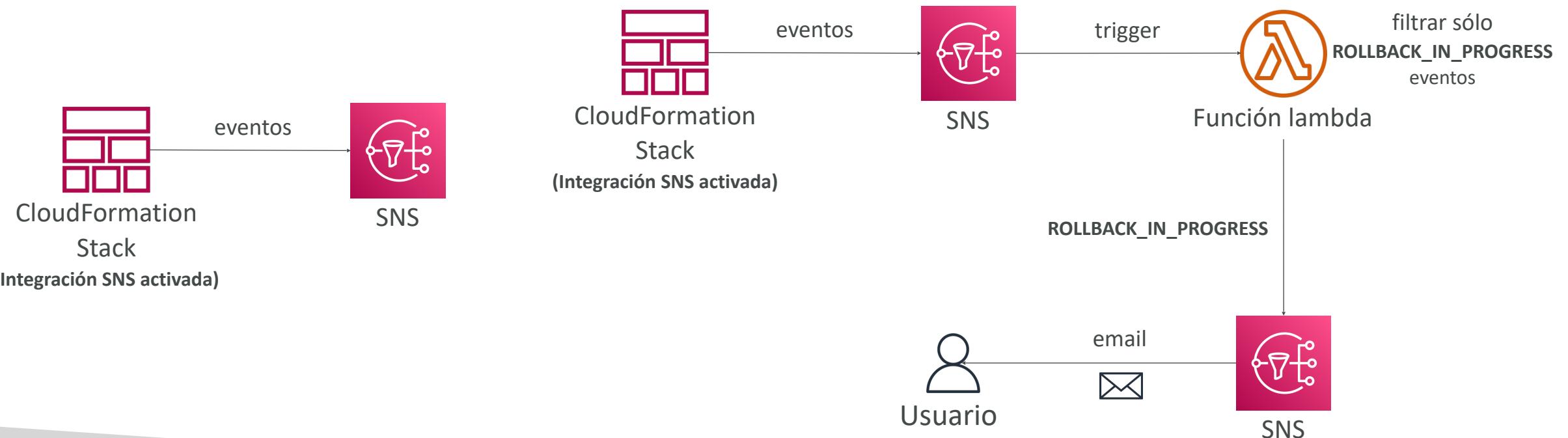
- El ID lógico lo eliges tú. Es cómo nombras la condición
- La función intrínseca (lógica) puede ser cualquiera de las siguientes
 - Fn::And
 - Fn::Equals
 - Fn::If
 - Fn::Not
 - Fn::Or

Retrocesos (rollback) de CloudFormation

- Falla la creación de stacks:
 - Por defecto: todo rueda hacia atrás (se borra). Podemos mirar los logs
 - Opción para desactivar la reversión y solucionar el problema
- Falla la actualización del stack:
 - El stack retrocede automáticamente al estado de funcionamiento anterior conocido
 - Posibilidad de ver en los logs lo que ha ocurrido y los mensajes de error

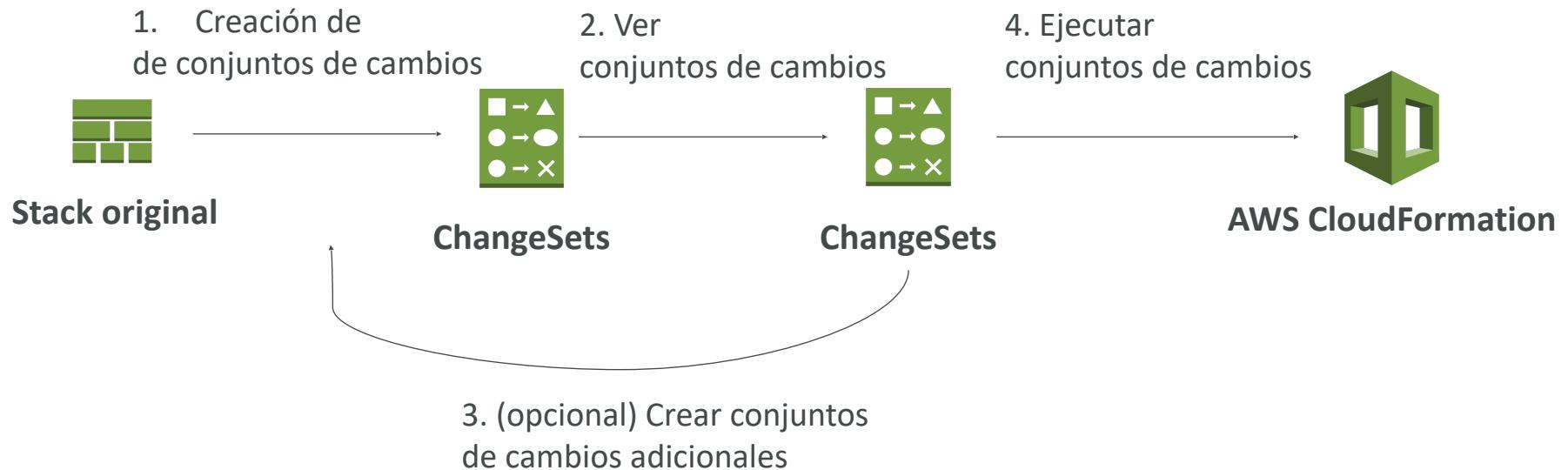
Notificaciones de stack de CloudFormation

- Enviar eventos de stack a SNS Topic (Email, Lambda, ...)
- Activar la integración SNS mediante opciones de stack



Conjuntos de cambios de Stacks

- Cuando actualizas un stack, necesitas saber qué cambia antes de que ocurra para tener más confianza
- Los cambios en los conjuntos no dicen si la actualización tendrá éxito



Fuente: <https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/using-cfn-updating-stacks-changesets.html>

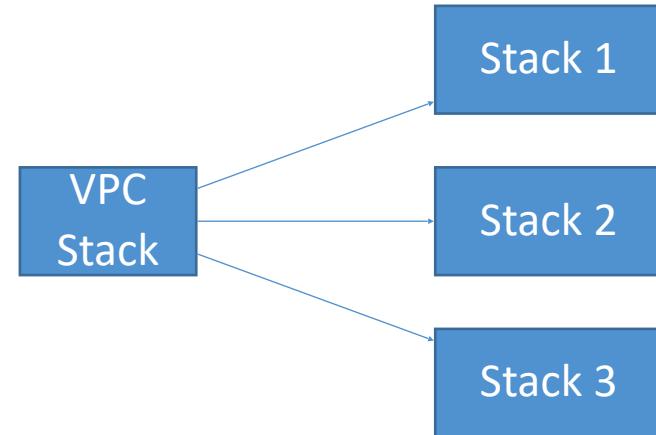
Stacks anidados / jerarquizados

- Los stacks anidados son stacks que forman parte de otros stacks
- Te permiten aislar patrones repetidos / componentes comunes en stacks separados y llamarlos desde otros stacks
- Ejemplo:
 - Configuración del Load Balancer que se reutiliza
 - Grupo de seguridad que se reutiliza
- Los stacks anidados se consideran la mejor práctica
- Para actualizar un stack anidado, actualiza siempre el padre (stack raíz)

CloudFormation - Stacks cruzados vs anidados

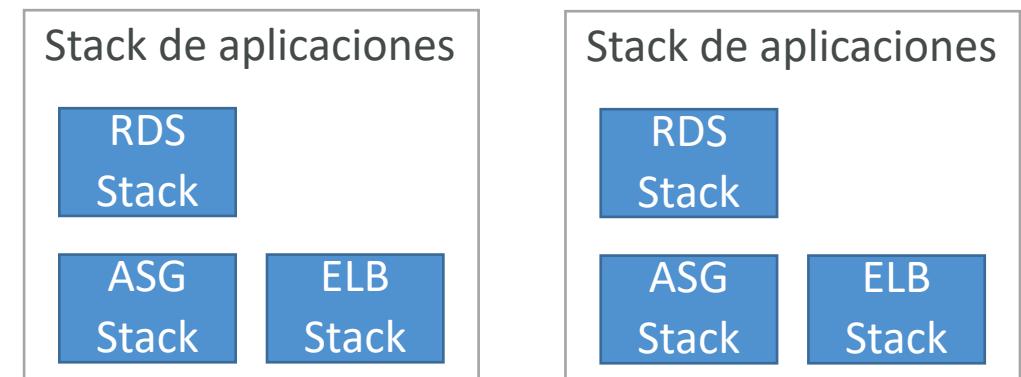
• Stacks cruzados

- Útil cuando los stacks tienen ciclos de vida diferentes
- Utiliza *Outputs Export* y *Fn::ImportValue*
- Cuando necesites pasar valores de exportación a muchos stacks (VPC Id, etc...)



• Stacks anidados

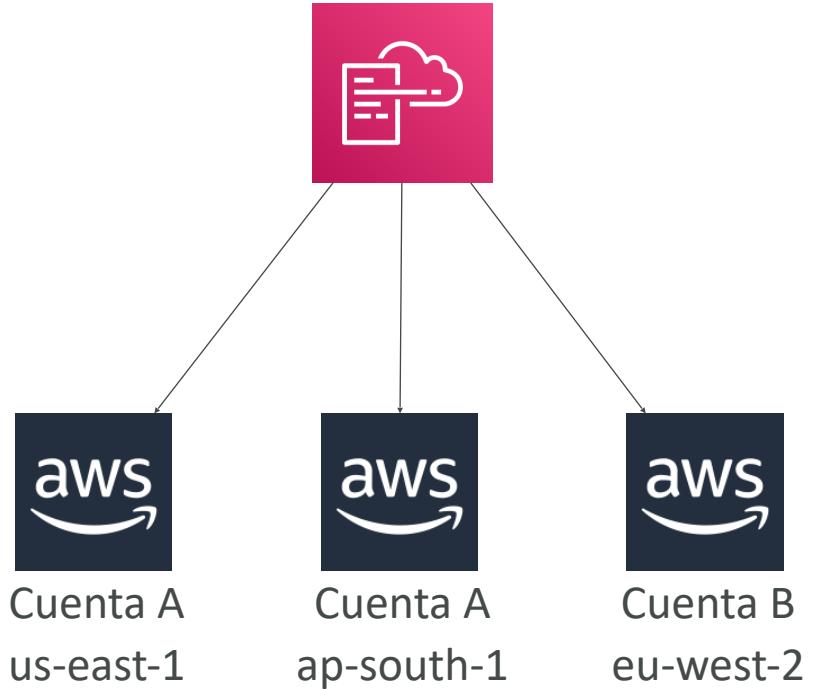
- Útiles cuando hay que reutilizar componentes
- Ej: reutilizar cómo configurar correctamente un Application Load Balancer
- El stack anidado sólo es importante para el stack de nivel superior (no se comparte)



CloudFormation - StackSets

- Crea, actualiza o elimina stacks **en varias cuentas y regiones** con una sola operación
- Cuenta de administrador para crear StackSets
- Cuentas de confianza para crear, actualizar, eliminar instancias de stack de StackSets
- Cuando actualizas un conjunto de pilas, todas las instancias de pila asociadas se actualizan en todas las cuentas y regiones.

CloudFormation **StackSet**
Cuenta de administrador



Desviaciones de CloudFormation

- También conocido por su término: *Drift* = Desviación
- CloudFormation te permite crear infraestructura
- Pero no te protege contra los cambios manuales de configuración
- ¿Cómo sabemos si nuestros recursos han **derivado**?
- ¡Podemos utilizar el drift de CloudFormation!
- Aún no se soportan todos los recursos: <https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/using-cfn-stack-drift-resource-list.html>

Políticas de stack de CloudFormation

- Durante una actualización de CloudFormation Stack, se permiten todas las acciones de actualización en todos los recursos (por defecto)
- **Una Política de pila es un documento JSON que define las acciones de actualización que se permiten en recursos específicos durante las actualizaciones de pila**
- Protege los recursos de actualizaciones involuntarias
- Cuando estableces una política de stack, todos los recursos de la pila están protegidos por defecto
- Especifica un PERMISO explícito para los recursos que quieras que se permita actualizar

```
{  
  "Statement": [  
    {  
      "Effect": "Allow",  
      "Action": "Update:*",  
      "Principal": "*",  
      "Resource": "*"  
    },  
    {  
      "Effect": "Deny",  
      "Action": "Update:*",  
      "Principal": "*",  
      "Resource": "LogicalResourceId/ProductionDatabase"  
    }  
  ]  
}
```

Permitir actualizaciones en todos los recursos **excepto** en *ProductionDatabase*

Monitorización, resolución de problemas y auditoría de AWS

CloudWatch, X-Ray y CloudTrail

Por qué es importante la monitorización

- Sabemos cómo desplegar aplicaciones
 - De forma segura
 - Automáticamente
 - Utilizando la Infraestructura como código
 - Aprovechando los mejores componentes de AWS
- Nuestras aplicaciones se despliegan, y a nuestros usuarios no les importa cómo lo hicimos...
- ¡A nuestros usuarios sólo les importa que la aplicación funcione!
 - Latencia de la aplicación: ¿aumentará con el tiempo?
 - Caídas de la aplicación: la experiencia del cliente no debe degradarse
 - Que los usuarios se pongan en contacto con el departamento de IT o se quejen no es un buen resultado
 - Solución de problemas
- Supervisión interna:
 - ¿Podemos prevenir los problemas antes de que ocurran?
 - Rendimiento y coste
 - Tendencias (patrones de escalado)
 - Aprendizaje y mejora

Monitorización en AWS

- **AWS CloudWatch:**

- Métricas: Recopila y realiza un seguimiento de las métricas clave
- Logs: Recopila, monitoriza, analiza y almacena logs
- Eventos: Envía notificaciones cuando ocurran determinados eventos en tu AWS
- Alarmas: Reacciona en tiempo real ante métricas / eventos

- **AWS X-Ray:**

- Solución de problemas de rendimiento y errores de la aplicación
- Rastreo distribuido de microservicios

- **AWS CloudTrail:**

- Monitorización interna de las llamadas a la API que se realizan
- Auditoría de los cambios realizados por tus usuarios en los recursos de AWS

Métricas de AWS CloudWatch



- CloudWatch proporciona métricas para todos los servicios de AWS
- La **métrica** es una variable a monitorizar (CPUUtilization, NetworkIn...)
- Las métricas pertenecen a **espacios de nombres**
- **Dimensión** es un atributo de una métrica (id de instancia, entorno, etc...)
- Hasta 30 dimensiones por métrica
- Las métricas tienen **marcas de tiempo**
- Puedes crear dashboards de métricas de CloudWatch

Monitorización detallada de EC2

- Las métricas de las instancias EC2 tienen métricas "cada 5 minutos".
- Con la monitorización detallada (por un coste), obtienes datos "cada 1 minuto"
- ¡Utiliza la monitorización detallada si quieres escalar más rápido tu ASG!
- La capa gratuita de AWS nos permite disponer de 10 métricas de monitorización detallada
- Nota: el uso de memoria de EC2 no se envía por defecto (debe enviarse desde dentro de la instancia como métrica personalizada)

Métricas personalizadas de CloudWatch

- Posibilidad de definir y enviar tus propias métricas personalizadas a CloudWatch
- Ejemplo: uso de memoria (RAM), espacio en disco, número de usuarios logs ...
- Utiliza la llamada a la API **PutMetricData**
- Posibilidad de utilizar dimensiones (atributos) para segmentar las métricas
 - Instance.id
 - Environment.name
- Resolución métrica (parámetro de la API **StorageResolution** - dos valores posibles):
 - Estándar: 1 minuto (60 segundos)
 - Alta resolución: 1/5/10/30 segundo(s) - Mayor coste
- **Importante:** Acepta puntos de datos métricos dos semanas en el pasado y dos horas en el futuro (asegúrate de configurar correctamente la hora de tu instancia EC2)



Registros de CloudWatch (Logs)

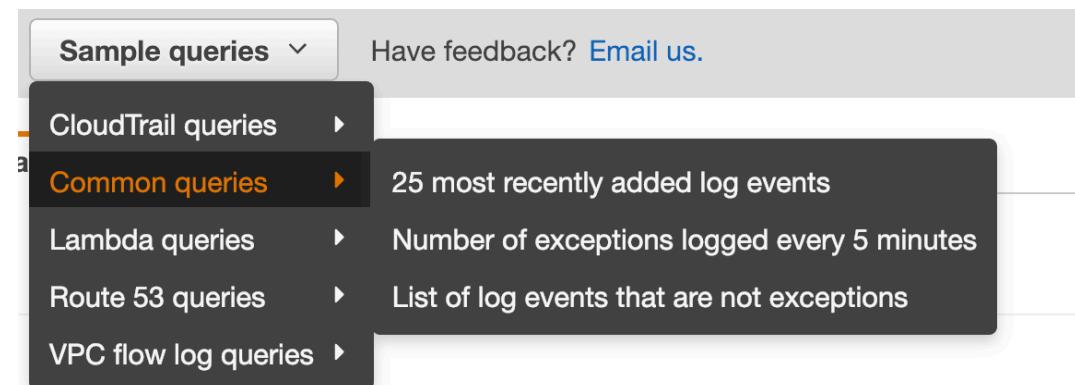
- **Grupos de registro:** nombre arbitrario, normalmente representando una aplicación
- **Flujo de registro:** instancias dentro de la aplicación / archivos de registro / contenedores
- Puede definir políticas de expiración de logs (nunca expiran, 30 días, etc..)
- **CloudWatch Logs puede enviar logs a:**
 - Amazon S3 (exportaciones)
 - Flujos de datos de Kinesis
 - Kinesis Data Firehose
 - AWS Lambda
 - ElasticSearch

CloudWatch Logs - Fuentes

- SDK, agente de CloudWatch Logs, agente unificado de CloudWatch
- Elastic Beanstalk: recogida de logs desde la aplicación
- ECS: recopilación desde contenedores
- AWS Lambda: recopilación de registros de funciones
- Registros de flujo de VPC: Registros específicos de VPC
- API Gateway
- CloudTrail basado en filtro
- Route53: registro de consultas DNS

Filtro de métricas e información de CloudWatch Logs

- CloudWatch Logs puede utilizar expresiones de filtro
 - Por ejemplo, encontrar una IP específica dentro de un log
 - O contar ocurrencias de "ERROR" en sus registros
- Los filtros de métricas pueden utilizarse para activar alarmas de CloudWatch Metrics.
- CloudWatch Logs Insights puede utilizarse para consultar registros y añadir consultas a CloudWatch Dashboards.

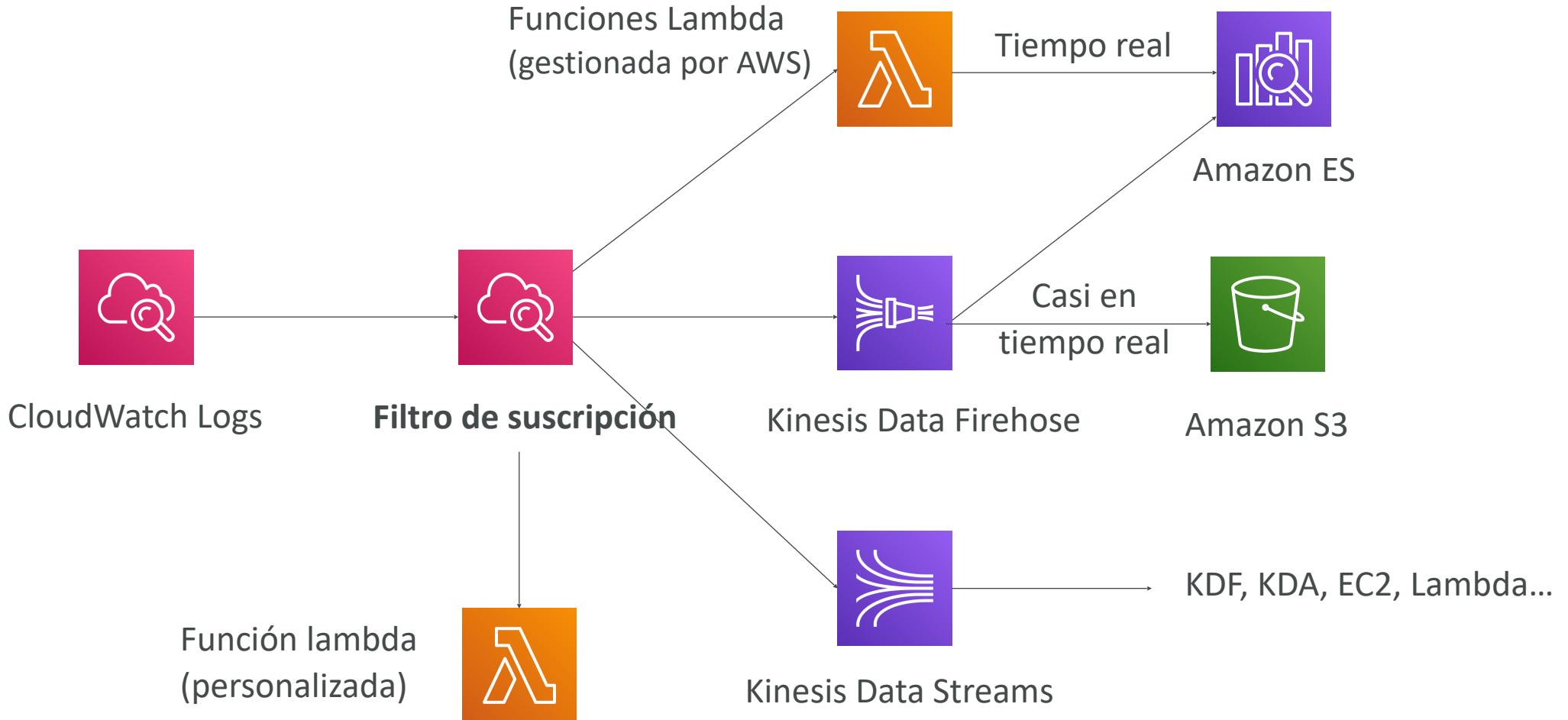


CloudWatch Logs - Exportación a S3

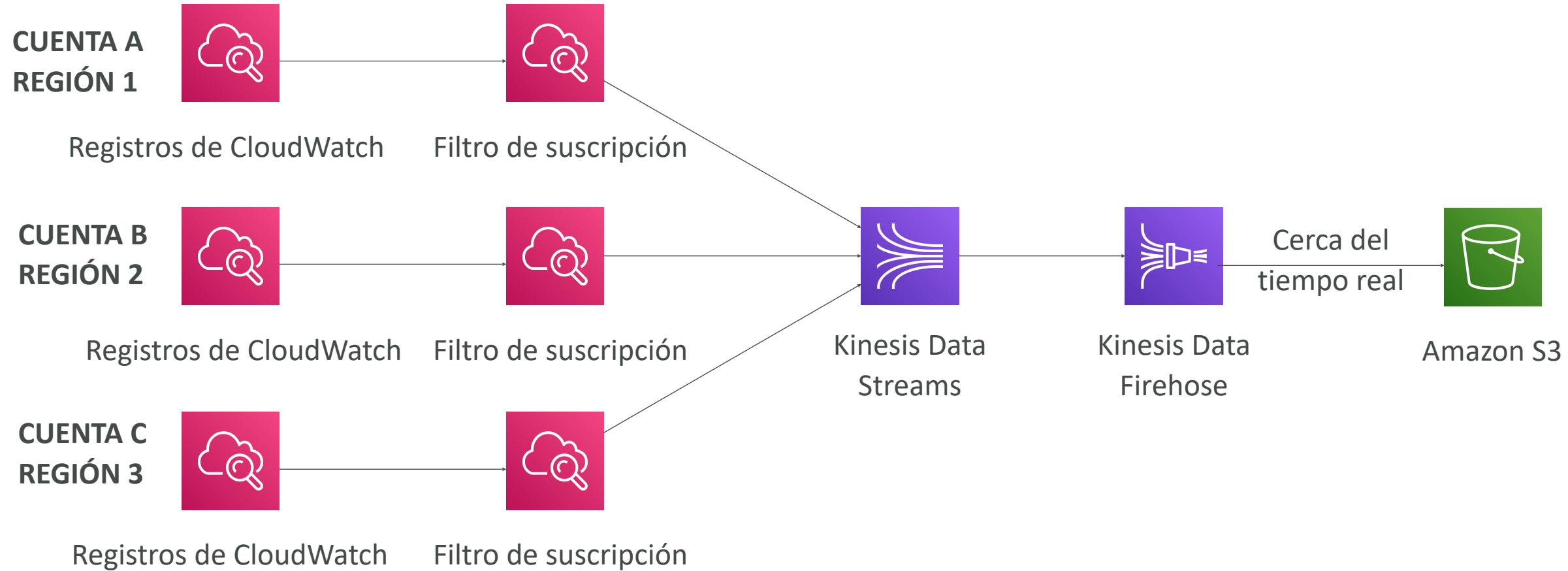


- Los datos de registro pueden tardar **hasta 12 horas** en estar disponibles para su exportación.
- La llamada a la API es **CreateExportTask**
- No en tiempo casi real ni en tiempo real... utilice suscripciones a registros en su lugar

Suscripciones a CloudWatch Logs

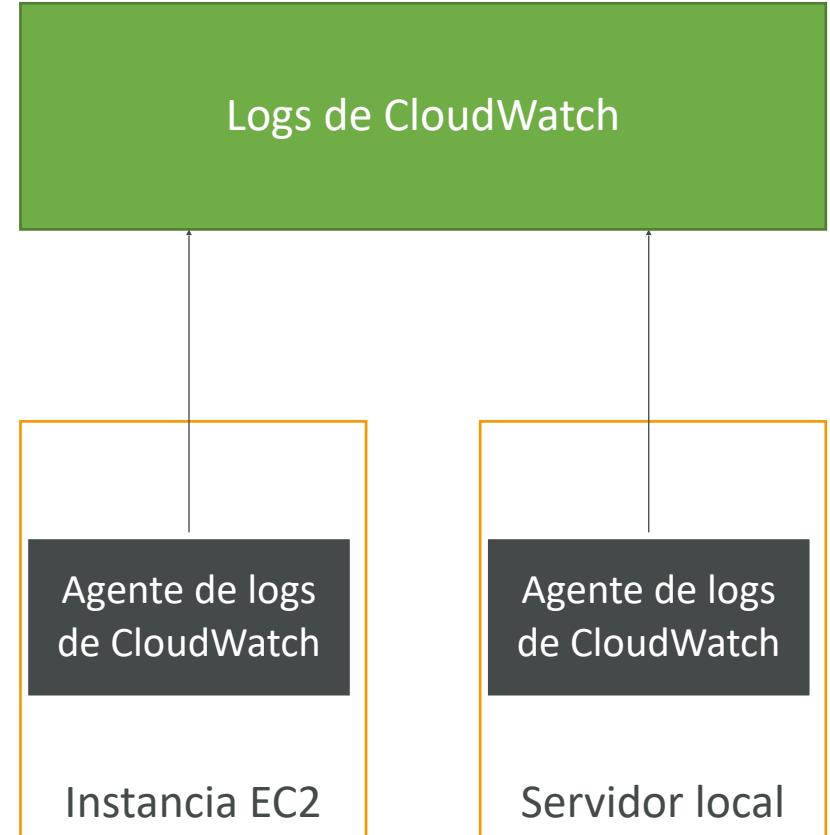


Agregación de CloudWatch Logs Multi-Cuenta y Multi-Región



CloudWatch Logs para EC2

- Por defecto, los logs de tu máquina EC2 no irán a CloudWatch
- Necesitas ejecutar un agente de CloudWatch en EC2 para enviar los logs que deseas.
- Asegúrate de que los permisos IAM son correctos
- El agente de logs de CloudWatch también puede configurarse en local



Agente CloudWatch Logs y Agente Unificado

- Para servidores virtuales (instancias EC2, servidores locales...)
- **Agente de logs de CloudWatch**
 - Versión antigua del agente
 - Sólo puede enviar a CloudWatch Logs
- **Agente Unificado CloudWatch**
 - Recoge métricas adicionales a nivel de sistema, como RAM, procesos, etc...
 - Recoge logs para enviarlos a CloudWatch Logs
 - Configuración centralizada mediante el Almacén de Parámetros SSM

Agente Unificado CloudWatch - Métricas

- Recopilados directamente en tu servidor Linux / instancia EC2
- **CPU** (activa, huésped, inactiva, sistema, usuario)
- **Métricas de disco** (libre, usado, total), IO de disco (escrituras, lecturas, bytes, iops)
- **RAM** (gratis, inactiva, usada, total, en caché)
- **Netstat** (número de conexiones TCP y UDP, paquetes netos, bytes)
- **Procesos** (totales, muertos, bloqueados, inactivos, en ejecución, en reposo)
- **Espacio de intercambio** (gratis, usado, % usado)

Filtro de métricas de los logs de CloudWatch

- CloudWatch Logs puede utilizar expresiones de filtro
 - Por ejemplo, encontrar una IP específica dentro de un logs
 - O contar ocurrencias de "ERROR" en tus logs
 - Los filtros métricos pueden utilizarse para activar alarmas
- **Los filtros no filtran datos retroactivamente. Los filtros sólo publican los puntos de datos métricos de los eventos que ocurren después de la creación del filtro.**



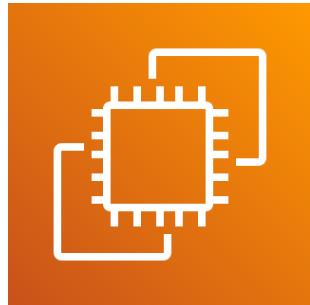
Alarmas de CloudWatch



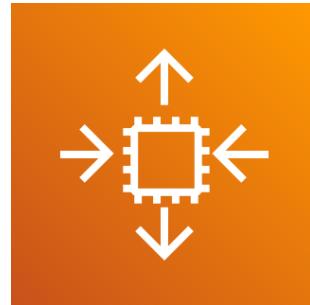
- Las alarmas se utilizan para activar notificaciones para cualquier métrica
- Varias opciones (muestreo, %, máx, mín, etc...)
- Estados de alarma:
 - OK
 - INSUFFICIENT_DATA
 - ALARM
- Periodo:
 - Tiempo en segundos para evaluar la métrica
 - Métricas personalizadas de alta resolución: 10 seg, 30 seg o múltiplos de 60 seg

Objetivos de alarma de CloudWatch

- Detener, Terminar, Reiniciar o Recuperar una Instancia EC2
- Activar la acción de Autoescalado
- Enviar notificación a SNS (desde donde puedes hacer prácticamente cualquier cosa)



Amazon EC2



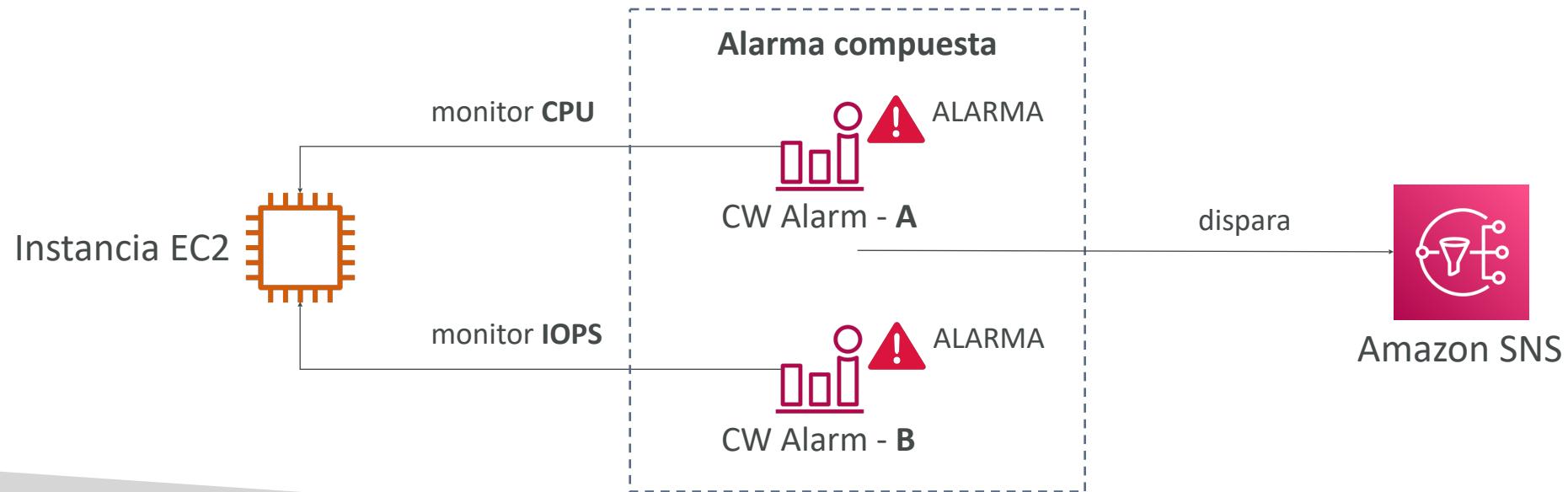
Auto Scaling Group
de EC2



Amazon SNS

Alarmas de CloudWatch - Alarmas compuestas

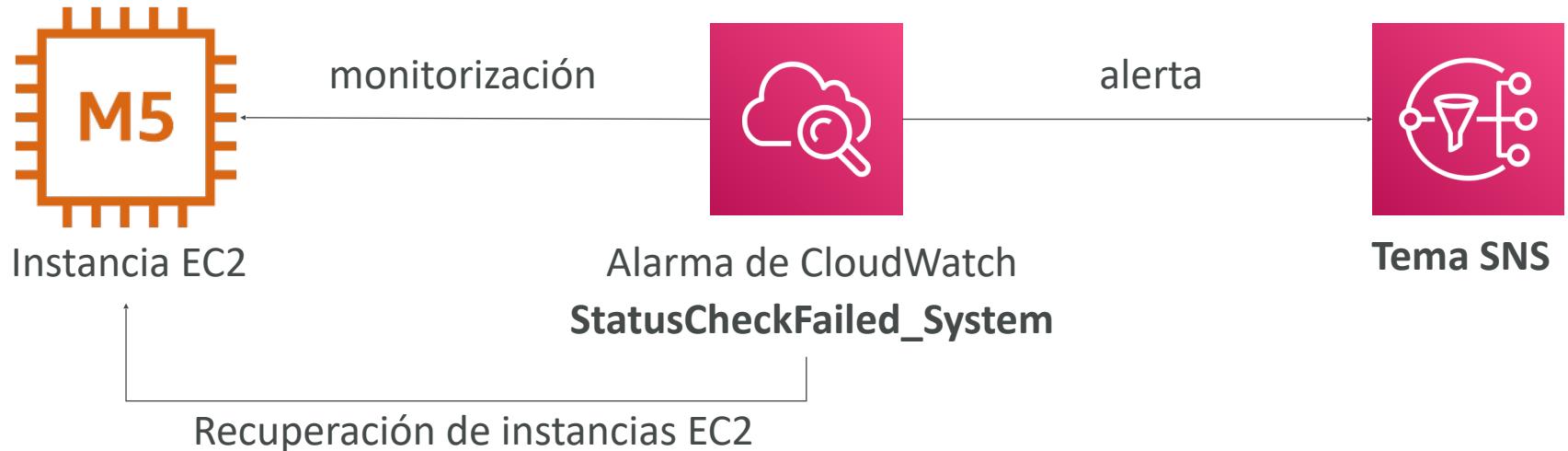
- Las Alarmas CloudWatch son sobre una única métrica
- **Las alarmas compuestas supervisan los estados de otras alarmas múltiples**
- Condiciones AND y OR
- Útiles para reducir el "ruido de alarma" creando alarmas compuestas complejas



Recuperación de instancias EC2

- Comprobación de estado:

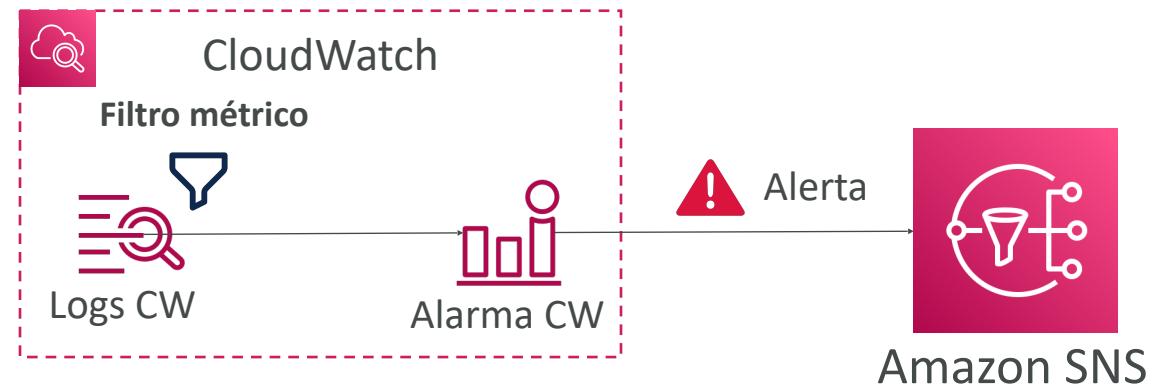
- Estado de la instancia = comprueba la máquina virtual EC2
- Estado del sistema = comprueba el hardware subyacente



- **Recuperación:** La misma IP privada, pública, elástica, metadatos, grupo de colocación

Alarma de CloudWatch: es bueno saber que...

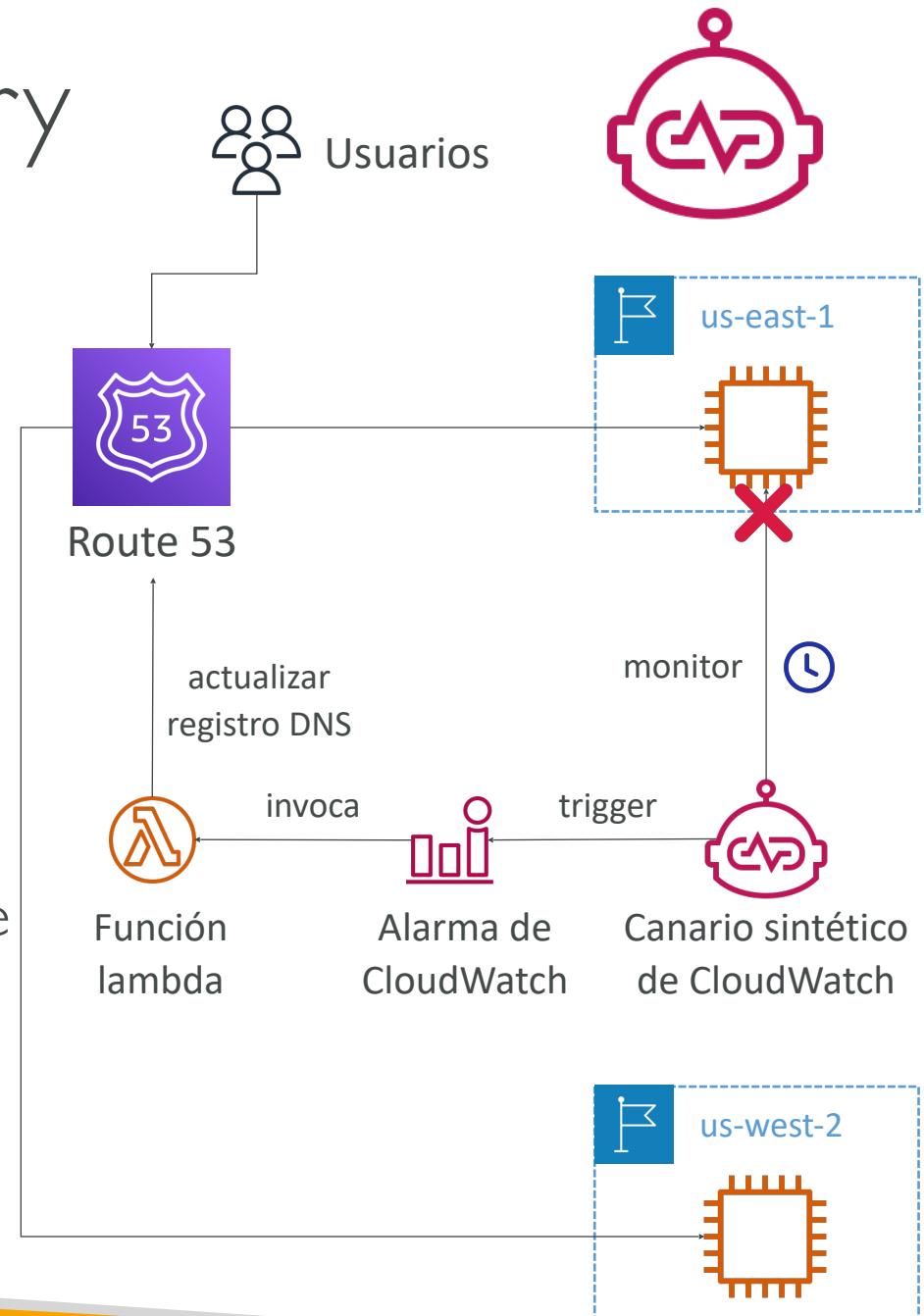
- Se pueden crear alarmas basadas en los filtros de métricas de CloudWatch Logs



- Para probar las alarmas y notificaciones, establece el estado de alarma mediante la CLI aws cloudwatch **set-alarm-state --alarm-name "myalarm" --state-value ALARM --state-reason "fines de prueba"**

CloudWatch Synthetics Canary

- Script configurable que supervisa tus API, URL, sitios web, ...
- Reproduce programáticamente lo que hacen tus clientes para encontrar problemas antes de que éstos se vean afectados
- Comprueba la disponibilidad y latencia de tus endpoints y puede almacenar datos de tiempo de carga y capturas de pantalla de la interfaz de usuario
- Integración con las alarmas de CloudWatch
- Scripts escritos en Node.js o Python
- Acceso programático a un navegador Google Chrome headless
- Puede ejecutarse una vez o de forma programada



CloudWatch Synthetics Canary Blueprints

- **Heartbeat Monitor (Monitor de latidos)**- carga URL, almacena capturas de pantalla y un archivo HTTP
- **API Canary** - prueba las funciones básicas de lectura y escritura de las API REST
- **Comprobador de enlaces rotos** - comprueba todos los enlaces dentro de la URL que estás probando
- **Monitorización visual** - compara una captura de pantalla tomada durante una ejecución canaria con una captura de pantalla de referencia
- **Canary Recorder (Grabador canario)** - se utiliza con el Grabador sintético de CloudWatch (graba tus acciones en un sitio web y genera automáticamente un script para ello)
- **GUI Workflow Builder** - verifica que se pueden realizar acciones en tu página web (por ejemplo, prueba una página web con un formulario de inicio de sesión)

Amazon EventBridge



- EventBridge es la siguiente evolución de CloudWatch Events
- **Bus de eventos predeterminado** - generado por servicios de AWS (CloudWatch Events)
- **Bus de eventos de socios** - recibe eventos de servicios SaaS o aplicaciones (Zendesk, DataDog, Segment, Auth0...)
- **Buses de eventos personalizados** - para tus propias aplicaciones
- Otras cuentas de AWS pueden acceder a los buses de eventos
- Puedes **archivar eventos** (todos/filtro) enviados a un bus de eventos (indefinidamente o por un periodo determinado)
- Posibilidad de **reproducir eventos archivados**
- **Reglas:** cómo procesar los eventos (como CloudWatch Events)

Amazon EventBridge - Registro de esquemas

- EventBridge puede analizar los eventos de tu bus e inferir el **esquema**
- El **Registro de esquemas** te permite generar código para tu aplicación, que sabrá de antemano cómo se estructuran los datos en el bus de eventos
- El esquema puede versionarse

The screenshot shows the AWS Schema Registry interface. At the top, it displays the schema name: `aws.codepipeline@CodePipelineActionExecutionStateChange`. Below this, the "Schema details" section provides the following information:

Schema name	Last modified	Schema ARN
<code>aws.codepipeline@CodePipelineActionExecutionStateChange</code>	Dec 1, 2019, 12:11 AM GMT	-
Description	Schema for event type CodePipelineActionExecutionStateChange, published by AWS service aws.codepipeline	Schema registry aws.events Number of versions 1 Schema type OpenAPI 3.0

Below the details, there is a section titled "Version 1" with a timestamp of "Created on Dec 1, 2019, 12:11 AM GMT". It includes an "Action" button and a "Download code bindings" button. The code bindings are displayed as a JSON-like snippet:

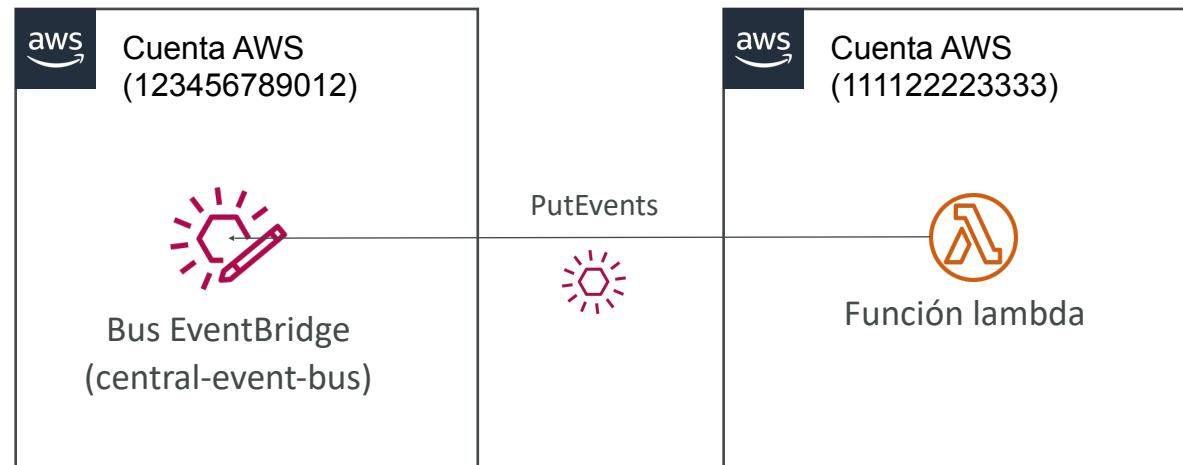
```
1 {
2   "openapi": "3.0.0",
3   "info": {
4     "version": "1.0.0",
5     "title": "CodePipelineActionExecutionStateChange"
6   },
7   "paths": {},
8   "components": {
9     "schemas": {
10       "AWSEvent": {
```

Amazon EventBridge - Política basada en recursos

- Gestionar permisos para un bus de eventos específico
- Ejemplo: permitir/denegar eventos de otra cuenta AWS o región AWS
- Caso práctico: agregar todos los eventos de tu Organización AWS en una única cuenta AWS o región AWS

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Effect": "Allow",  
            "Action": "events:PutEvents",  
            "Principal": { "AWS": "111122223333" },  
            "Resource": "arn:aws:events:us-east-1:123456789012:  
event-bus/central-event-bus"  
        }  
    ]  
}
```

Permitir **eventos** desde otra cuenta AWS

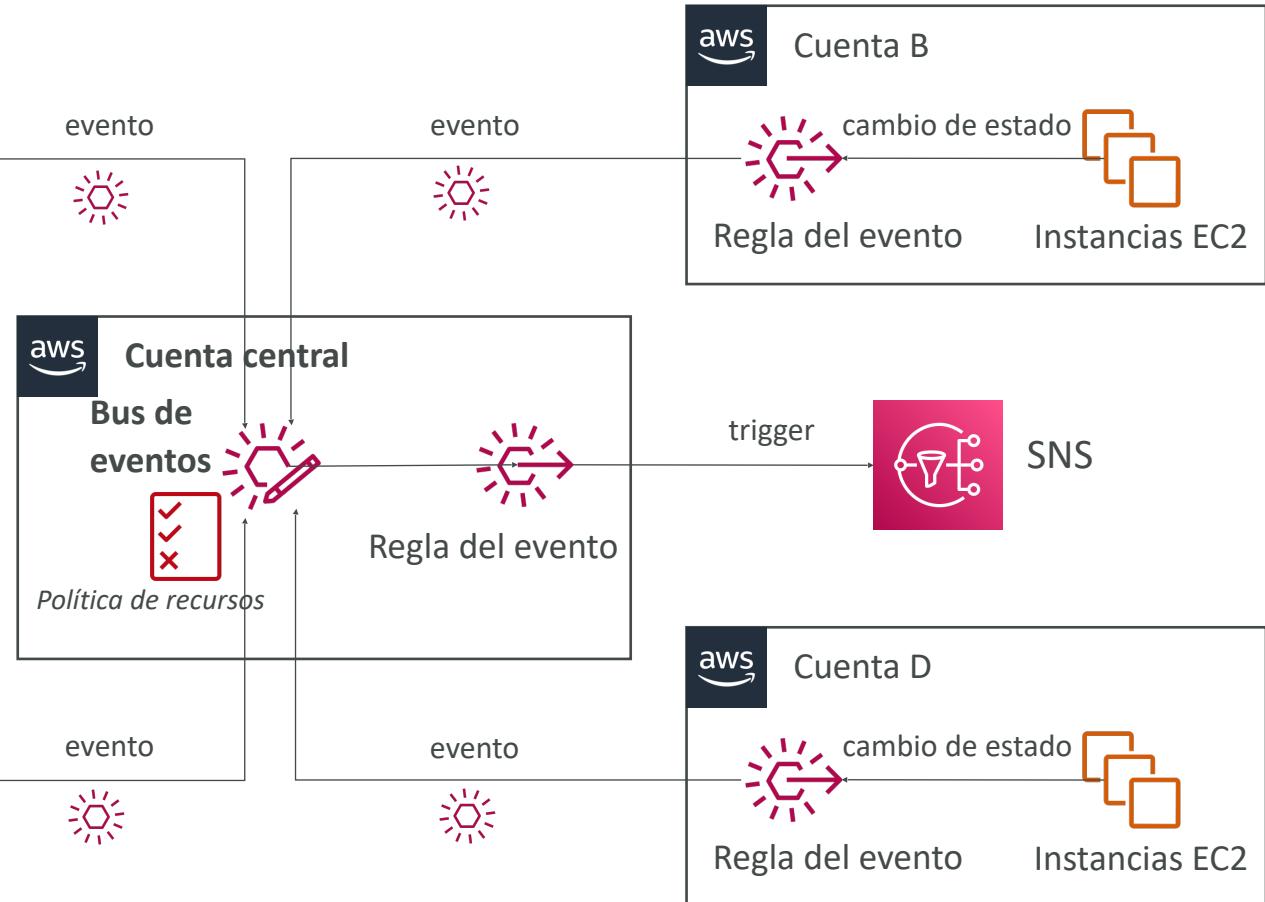
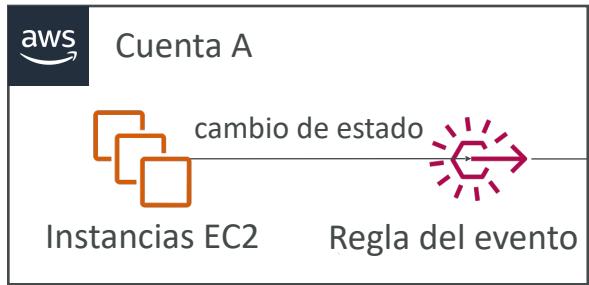


Amazon EventBridge vs CloudWatch Events

- Amazon EventBridge se basa en CloudWatch Events y lo amplía.
- Utiliza la misma API y endpoint de servicio, y la misma infraestructura de servicio subyacente.
- EventBridge permite la ampliación para añadir buses de eventos para tus aplicaciones personalizadas y tus aplicaciones SaaS de terceros.
- Event Bridge tiene la capacidad de Registro de esquemas
- EventBridge tiene un nombre diferente para marcar las nuevas capacidades
- Con el tiempo, el nombre CloudWatch Events será sustituido por EventBridge.

EventBridge - Agregación multicuenta

```
{
    "id": "7bf73129-1428-4cd3-a780-95db273d1602",
    "detail-type": "EC2 Instance State-change Notification",
    "source": "aws.ec2",
    "account": "123456789012",
    "time": "2021-11-11T21:29:54Z",
    "region": "us-east-1",
    "resources": [
        "arn:aws:ec2:us-east-1:123456789012:instance/i-abcd1111"
    ],
    "detail": {
        "instance-id": "i-abcd1111",
        "state": "pending"
    }
}
```



AWS X-Ray

- Depurar en producción, a la antigua usanza:
 - Test localmente
 - Añade logs por todas partes
 - Vuelve a desplegar en producción
- Los formatos de logs difieren entre las aplicaciones que utilizan CloudWatch y el análisis es difícil.
- Depuración: monolito "fácil", servicios distribuidos "difícil".
- No hay vistas comunes de toda tu arquitectura
- Entra... ¡AWS X-Ray!

AWS X-Ray

Análisis visual de nuestras aplicaciones



Ventajas de AWS X-Ray

- Resolución de problemas de rendimiento (cuellos de botella)
- Comprender las dependencias en una arquitectura de microservicios
- Localizar problemas de servicio
- Revisar el comportamiento de las peticiones
- Encontrar errores y excepciones
- ¿Cumplimos el SLA de tiempo?
- ¿Dónde estoy estrangulado?
- Identifica a los usuarios afectados

Compatibilidad con X-Ray

- AWS Lambda
- Elastic Beanstalk
- ECS
- ELB
- API Gateway
- Instancias EC2 o cualquier servidor de aplicaciones (incluso in situ)

AWS X-Ray aprovecha el rastreo

- El rastreo es una forma integral de seguir una "petición".
- Cada componente que se ocupa de la petición añade su propio "seguimiento".
- El seguimiento se compone de segmentos (+ subsegmentos)
- Se pueden añadir anotaciones a las trazas para proporcionar información adicional
- Posibilidad de rastrear:
 - Cada petición
 - Muestra de petición (en %, por ejemplo, o en tasa por minuto)
- Seguridad X-Ray:
 - IAM para autorización
 - KMS para cifrado en reposo

AWS X-Ray

¿Cómo activarlo?

I) Tu código (Java, Python, Go, Node.js, .NET) debe importar el SDK de AWS X-Ray

- Se necesita muy poca modificación del código
- A continuación, el SDK de la aplicación capturará
 - Llamadas a servicios de AWS
 - Peticiones HTTP / HTTPS
 - Llamadas a bases de datos (MySQL, PostgreSQL, DynamoDB)
 - Llamadas a colas (SQS)

2) Instala el demonio (daemon) X-Ray o activa la integración de X-Ray en AWS

El demonio X-Ray funciona como un interceptor de paquetes UDP de bajo nivel (Linux / Windows / Mac...)

AWS Lambda / otros servicios de AWS ya ejecutan el demonio X-Ray por ti

Cada aplicación debe tener los derechos IAM para escribir datos en X-Ray



La magia de X-Ray

- El servicio X-Ray recoge datos de todos los diferentes servicios
- El mapa de servicios se calcula a partir de todos los segmentos y trazas
- X-Ray es gráfico, por lo que incluso personas no técnicas pueden ayudar a solucionar problemas



Solución de problemas de AWS X-Ray

- Si X-Ray no funciona en EC2
 - Asegúrate de que el rol IAM de EC2 tiene los permisos adecuados
 - Asegúrate de que la instancia EC2 está ejecutando el demonio X-Ray
- Para habilitarlo en AWS Lambda
 - Asegúrate de que tiene un rol de ejecución IAM con la política adecuada (AWSX-RayWriteOnlyAccess)
 - Asegúrate de que X-Ray se importa en el código
 - Habilitar el **rastreo activo de X-Ray en Lambda**

Instrumentación de X-Ray en tu código

- Instrumentar significa medir el rendimiento del producto, diagnosticar errores y escribir información de rastreo.
- Para instrumentar el código de tu aplicación, utiliza el **SDK de X-Ray**
- Muchos SDK sólo requieren cambios de configuración
- Puedes modificar el código de tu aplicación para personalizar y anotar los datos que el SDK envía a X-Ray, utilizando **interceptores, filtros, manejadores, middleware...**

Ejemplo para Node.js y Express

```
var app = express();

var AWSXRay = ...;
app.use(AWSXRay.express.openSegment('MyApp'));

app.get('/', function (req, res) {
  res.render('index');
});

app.use(AWSXRay.express.closeSegment());
```

Conceptos de X-Ray

- Segmentos: cada aplicación / servicio los enviará
- Subsegmentos: si necesitas más detalles en tu segmento
- Rastreo: segmentos reunidos para formar un rastreo de extremo a extremo
- Muestreo: disminuye la cantidad de peticiones enviadas a X-Ray, reduce el coste
- Anotaciones: Pares clave-valor utilizados para **indexar** las trazas y utilizarlas con **filtros**
- Metadatos: Pares de valores clave, no indexados, no utilizados para búsquedas
- El demonio / agente de X-Ray tiene una configuración para enviar trazas a cuentas cruzadas:
 - Asegúrate de que los permisos IAM son correctos - el agente asumirá el rol
 - Esto permite tener una cuenta central para todas las trazas de tu aplicación

Reglas de muestreo de X-Ray

- Con las reglas de muestreo, controlas la cantidad de datos que registras
 - Puedes modificar las reglas de muestreo sin cambiar tu código
-
- Por defecto, el SDK de X-Ray registra la primera petición **cada segundo**, y **cinco por ciento** de las peticiones adicionales.
 - **Una petición por segundo es el reservoir (déposito)**, lo que garantiza que se registre al menos una traza cada segundo mientras el servicio esté sirviendo peticiones.
 - El **5% es el rate (tasa)** a la que se muestran las peticiones adicionales que superan el tamaño de la reserva.

Reglas de muestreo personalizadas de X-Ray

- Puedes crear tus propias reglas con la función `reservoir` (depósito) y el rate (tasa)

Example Higher minimum rate for POSTs

- Rule name – `POST minimum`
- Priority – `100`
- Reservoir – `10`
- Rate – `0.10`
- Service name – `*`
- Service type – `*`
- Host – `*`
- HTTP method – `POST`
- URL path – `*`
- Resource ARN – `*`

Example Debugging rule to trace all requests for a problematic route

A high-priority rule applied temporarily for debugging.

- Rule name – `DEBUG - history updates`
- Priority – `1`
- Reservoir – `1`
- Rate – `1`
- Service name – `Scorekeep`
- Service type – `*`
- Host – `*`
- HTTP method – `PUT`
- URL path – `/history/*`
- Resource ARN – `*`

APIs de escritura de X-Ray (utilizadas por el demonio X-Ray)

```
"Effect": "Allow",
"Action": [
    "xray:PutTraceSegments",
    "xray:PutTelemetryRecords",
    "xray:GetSamplingRules",
    "xray:GetSamplingTargets",
    "xray:GetSamplingStatisticSummaries"
],
"Resource": [
    "*"
]
```

arn:aws:iam::aws:policy/AWSXrayWriteOnlyAccess

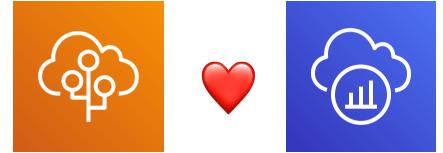
- **PutTraceSegments:** Sube documentos de segmentos a AWS X-Ray
- **PutTelemetryRecords:** Utilizado por el demonio de AWS X-Ray para cargar telemetría.
 - SegmentsReceivedCount, SegmentsRejectedCounts, BackendConnectionErrors...
- **GetSamplingRules:** Recupera todas las reglas de muestreo (para saber qué/cuándo enviar)
- GetSamplingTargets & GetSamplingStatisticSummaries: avanzado
- El demonio X-Ray necesita tener una política IAM que autorice las llamadas correctas a la API para funcionar correctamente

API de lectura de X-Ray - continuación

```
"Effect": "Allow",
"Action": [
    "xray:GetSamplingRules",
    "xray:GetSamplingTargets",
    "xray:GetSamplingStatisticSummaries",
    "xray:BatchGetTraces",
    "xray:GetServiceGraph",
    "xray:GetTraceGraph",
    "xray:GetTraceSummaries",
    "xray:GetGroups",
    "xray:GetGroup",
    "xray:GetTimeSeriesServiceStatistics"
],
"Resource": [
    "*"
]
```

- **GetServiceGraph**: gráfico principal
- **BatchGetTraces**: Recupera una lista de trazas especificadas por ID. Cada traza es una colección de documentos de segmentos que se originan a partir de una única petición.
- **GetTraceSummaries**: Recupera los ID y las anotaciones de las trazas disponibles para un periodo de tiempo especificado utilizando un filtro opcional. Para obtener las trazas completas, pasa los ID de las trazas a BatchGetTraces.
- **GetTraceGraph**: Recupera un gráfico de servicio para uno o más ID de traza específicos.

X-Ray con Elastic Beanstalk



- Las plataformas AWS Elastic Beanstalk incluyen el demonio X-Ray
- Puedes ejecutar el demonio estableciendo una opción en la consola de Elastic Beanstalk o con un archivo de configuración (en .ebextensions/xray-daemon.config)

```
option_settings:  
  aws:elasticbeanstalk:xray:  
    XRayEnabled: true
```

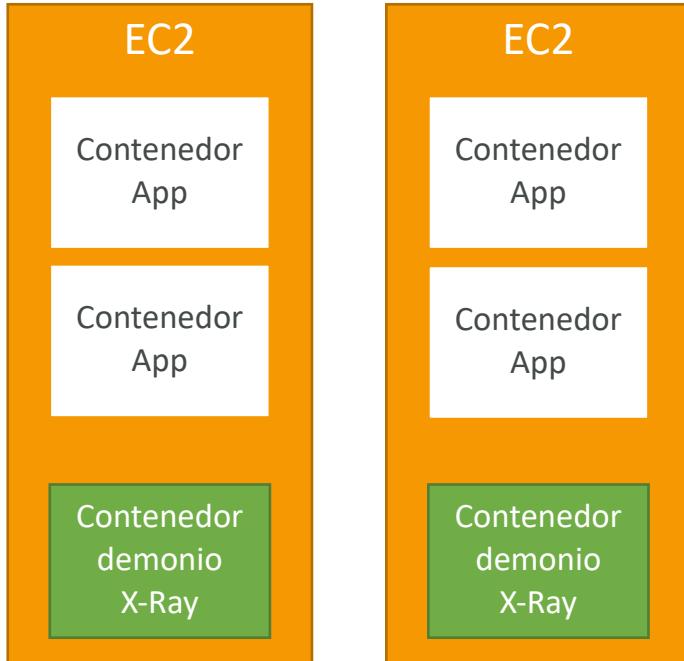
- Asegúrate de dar a tu perfil de instancia los permisos IAM correctos para que el demonio X-Ray pueda funcionar correctamente
- A continuación, asegúrate de que el código de tu aplicación está instrumentado con el SDK de X-Ray
- Nota: El demonio X-Ray no está disponible para Docker Multicontainer.

Opciones de integración ECS + X-Ray



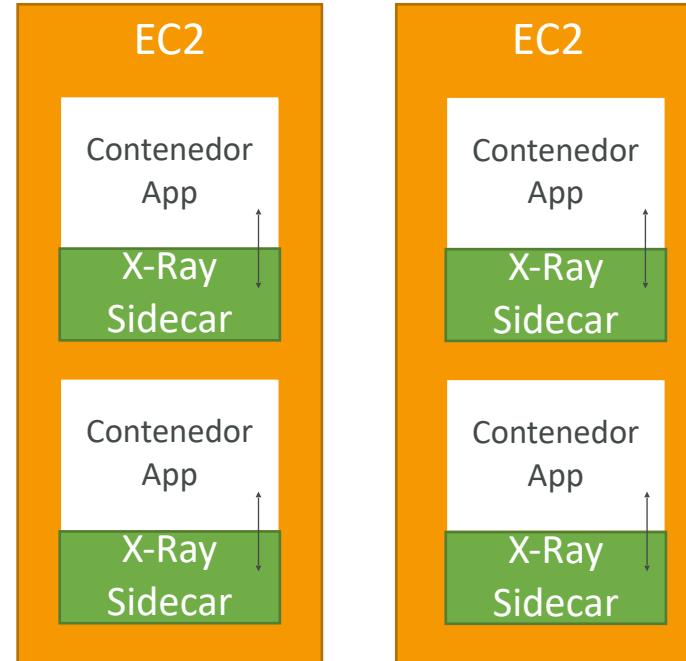
Cluster ECS

Contenedor demonio de X-Ray



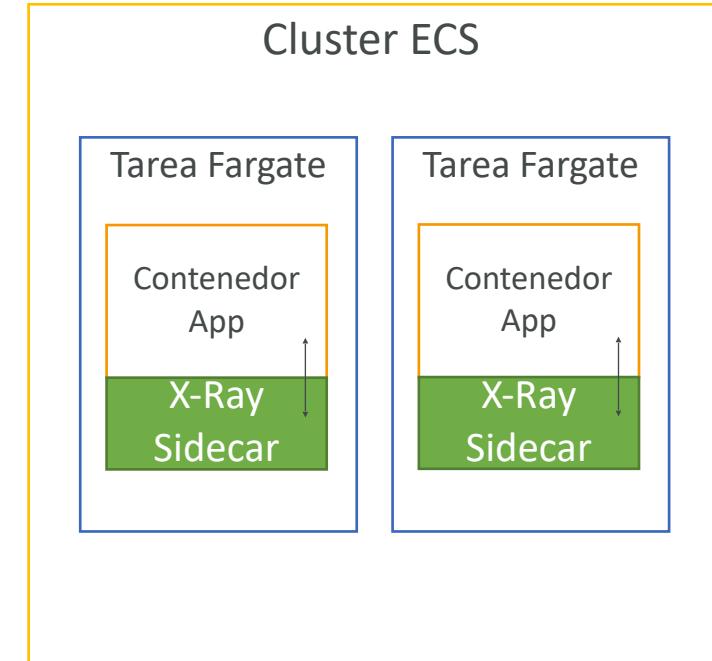
Cluster ECS

Contenedor X-Ray como Sidecar



Cluster Fargate

Contenedor de X-Ray como Sidecar



ECS + X-Ray: Ejemplo de definición de tarea

```
{  
    "name": "xray-daemon",  
    "image": "123456789012.dkr.ecr.us-east-2.amazonaws.com/xray-daemon",  
    "cpu": 32,  
    "memoryReservation": 256,  
    "portMappings" : [  
        {  
            "hostPort": 0,  
            "containerPort": 2000,  
            "protocol": "udp"  
        },  
    ],  
    {  
        "name": "scorekeep-api",  
        "image": "123456789012.dkr.ecr.us-east-2.amazonaws.com/scorekeep-api",  
        "cpu": 192,  
        "memoryReservation": 512,  
        "environment": [  
            { "name" : "AWS_REGION", "value" : "us-east-2" },  
            { "name" : "NOTIFICATION_TOPIC", "value" : "arn:aws:sns:us-east-2:123456789012:scorekeep-notifications" },  
            { "name" : "AWS_XRAY_DAEMON_ADDRESS", "value" : "xray-daemon:2000" }  
        ],  
        "portMappings" : [  
            {  
                "hostPort": 5000,  
                "containerPort": 5000  
            }  
        ],  
        "links": [  
            "xray-daemon"  
        ]  
    }  
}
```

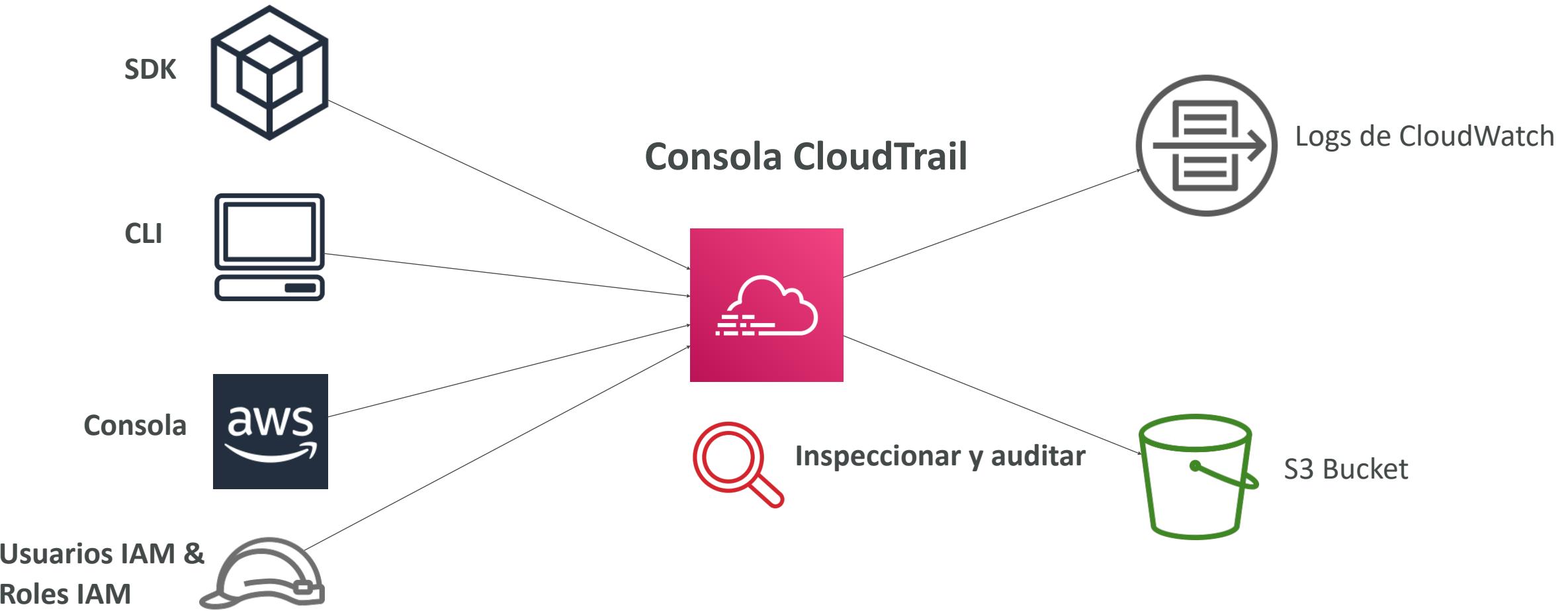
<https://docs.aws.amazon.com/xray/latest/devguide/xray-daemon-ecs.html#xray-daemon-ecs-build>

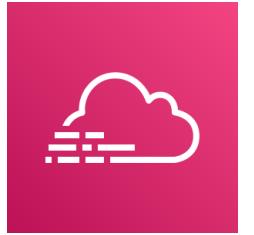
AWS CloudTrail



- **Proporciona gobernanza, normativa y auditoría para tu cuenta de AWS**
- CloudTrail está activado por defecto
- Obtén **un historial de eventos / llamadas a la API realizadas dentro de tu Cuenta de AWS** por:
 - Consola
 - SDK
 - CLI
 - Servicios de AWS
- Puedes poner logs de CloudTrail en CloudWatch Logs o S3
- **Un rastro (trail) puede aplicarse a todas las regiones (por defecto) o a una sola región**
- Si se elimina un recurso en AWS, ¡investiga primero en CloudTrail!

Diagrama de CloudTrail





Eventos CloudTrail

- **Eventos de gestión:**

- Operaciones que se realizan en los recursos de tu cuenta de AWS
- Ejemplos:
 - Configurar la seguridad (IAM **AttachRolePolicy**)
 - Configurar reglas para enrutar datos (Amazon EC2 **CreateSubnet**)
 - Configurar logs (AWS CloudTrail **CreateTrail**)
- **Por defecto, los trails están configurados para logs de eventos de gestión.**
- Pueden separar los **eventos de lectura** (que no modifican los recursos) de los Eventos de Escritura (que pueden modificar los recursos)

- **Eventos de datos:**

- **Por defecto, los eventos de datos no se registran (porque son operaciones de gran volumen)**
- Actividad a nivel de objeto de Amazon S3 (ej: **GetObject**, **DeleteObject**, **PutObject**): puede separar los Eventos de Lectura de los de Escritura
- Actividad de ejecución de funciones de AWS Lambda (la API **Invoke**)

- **CloudTrail Insights Events:**

- Ver siguiente diapositiva ☺

CloudTrail Insights

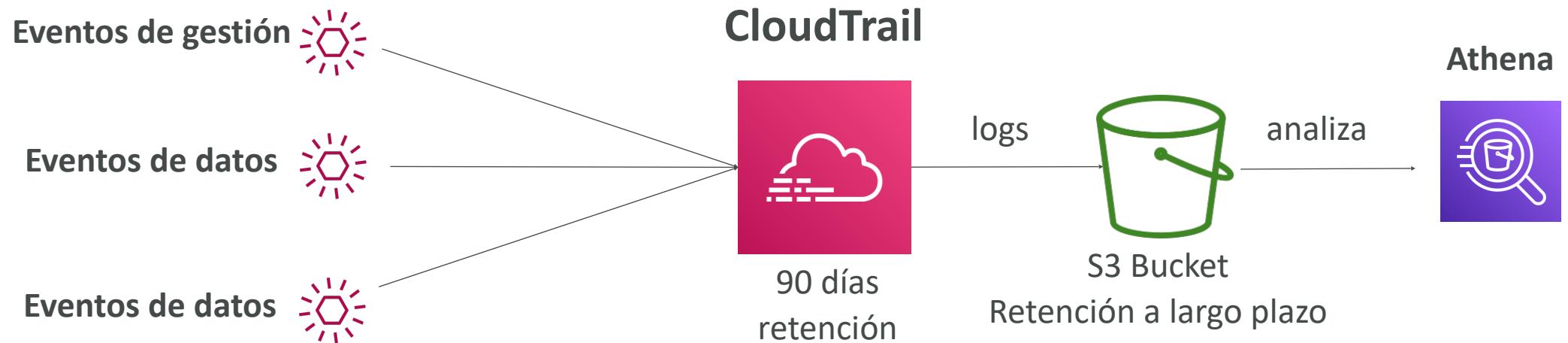


- Activa **CloudTrail Insights** para detectar actividad inusual en tu cuenta:
 - aprovisionamiento inexacto de recursos
 - superación de los límites de servicio
 - ráfagas de acciones de AWS IAM
 - lagunas en la actividad de mantenimiento periódico
- CloudTrail Insights analiza los eventos de gestión normales para crear una línea de base
- Y después **analiza continuamente los eventos de escritura para detectar patrones inusuales**
 - Las anomalías aparecen en la consola de CloudTrail
 - El evento se envía a Amazon S3
 - Se genera un evento EventBridge (para necesidades de automatización)



Retención de Eventos CloudTrail

- Los eventos se almacenan durante 90 días en CloudTrail
- Para conservar los eventos más allá de este periodo, regístralos en S3 y utiliza Athena



CloudTrail vs CloudWatch vs X-Ray

- CloudTrail:
 - Auditoría de llamadas a la API realizadas por usuarios / servicios / consola de AWS
 - Útil para detectar llamadas no autorizadas o la causa raíz de los cambios
- CloudWatch:
 - CloudWatch Metrics a lo largo del tiempo para monitorización
 - CloudWatch Logs para almacenar logs de aplicaciones
 - CloudWatch Alarms para enviar notificaciones en caso de métricas inesperadas
- X-Ray:
 - Análisis de Trazas Automatizado y Visualización de Mapas de Servicios Centrales
 - Análisis de latencia, errores y fallos
 - Seguimiento de peticiones en sistemas distribuidos

Integración y mensajería de AWS

SQS, SNS & Kinesis

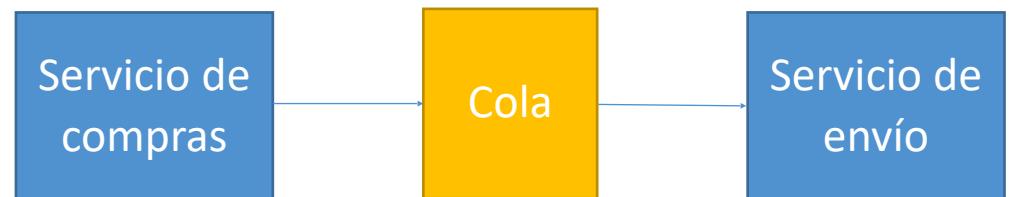
Introducción a la sección

- Cuando empezamos a desplegar varias aplicaciones, es inevitable que tengan que comunicarse entre sí.
- Existen dos patrones de comunicación entre aplicaciones

**1) Comunicaciones sincrónicas
(de aplicación a aplicación)**



**2) Asíncrono / basado en eventos
(aplicación a cola a aplicación)**

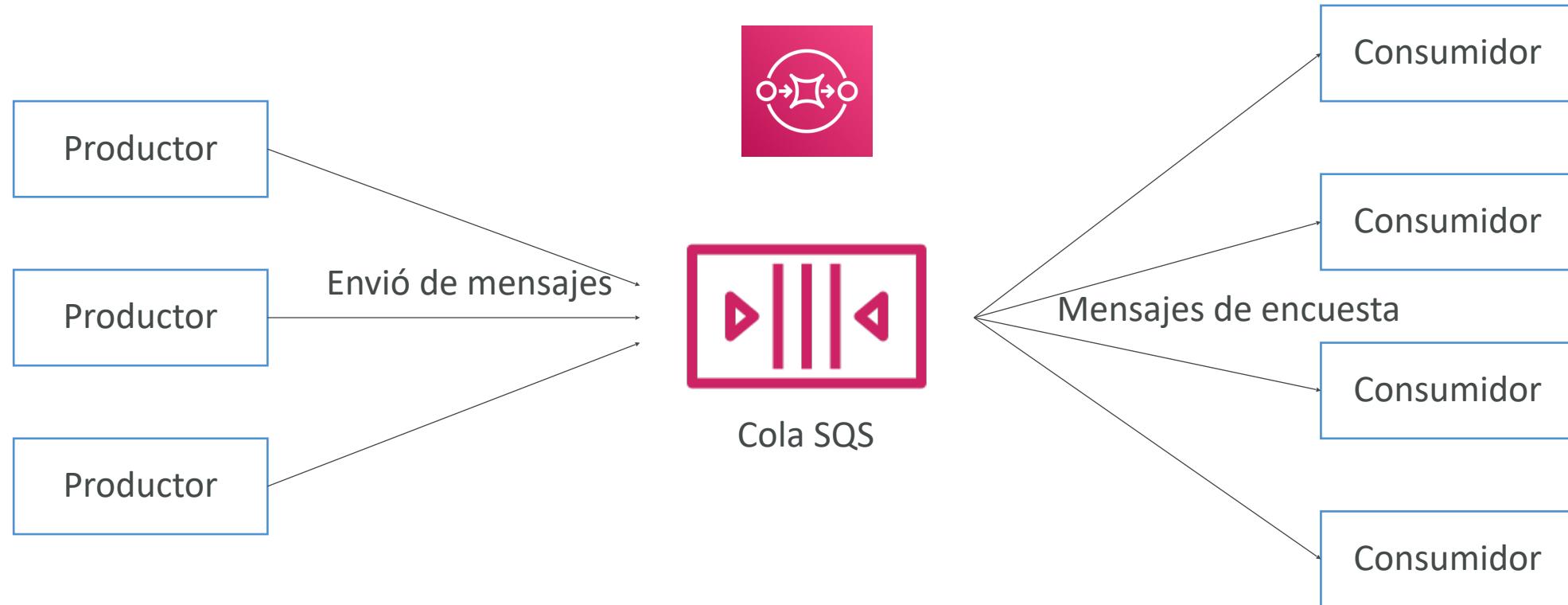


Introducción a la sección

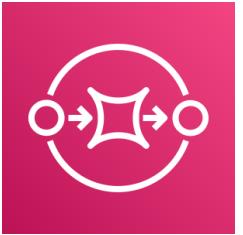
- La sincronización entre aplicaciones puede ser problemática si hay picos repentinos de tráfico
- ¿Qué pasa si de repente necesitas codificar 1000 vídeos pero normalmente son 10?
- En ese caso, es mejor **desacoplar** tus aplicaciones,
 - utilizando SQS: modelo de cola
 - utilizando SNS: modelo pub/sub
 - usando Kinesis: modelo de streaming en tiempo real
- ¡Estos servicios pueden escalar independientemente de nuestra aplicación!

Amazon SQS

¿Qué es una cola?



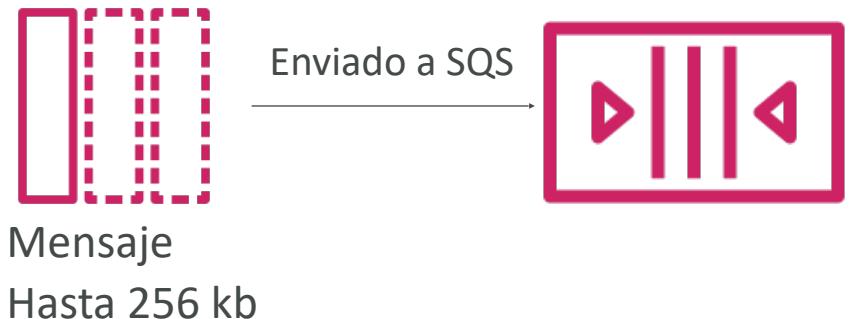
Amazon SQS – Cola estándar



- La oferta más antigua (más de 10 años)
- Servicio totalmente gestionado, utilizado para **desacoplar aplicaciones**
- Atributos:
 - Rendimiento ilimitado, número ilimitado de mensajes en cola
 - Retención de mensajes por defecto 4 días, máximo de 14 días
 - Baja latencia (<10 ms en publicación y recepción)
 - Limitación de 256 KB por mensaje enviado
- Puede haber mensajes duplicados (al menos una entrega, ocasionalmente)
- Puede haber mensajes fuera de orden (orden de mejor esfuerzo)

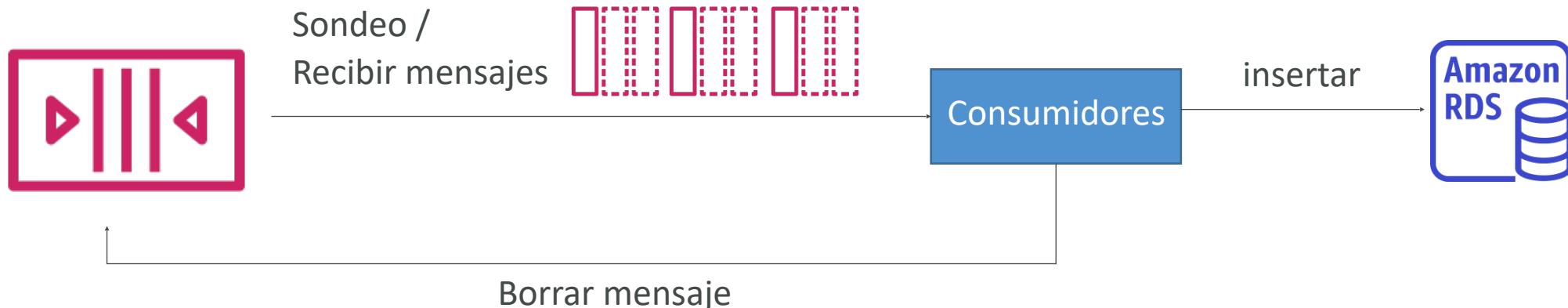
SQS - Producción de mensajes

- Producido a SQS utilizando el SDK (API SendMessage)
- El mensaje se **conserva** en SQS hasta que un consumidor lo elimina
- Retención del mensaje: por defecto 4 días, hasta 14 días
- Ejemplo: enviar un pedido para ser procesado
 - Id de pedido
 - Id de cliente
 - Los atributos que quieras
- SQS estándar: rendimiento ilimitado



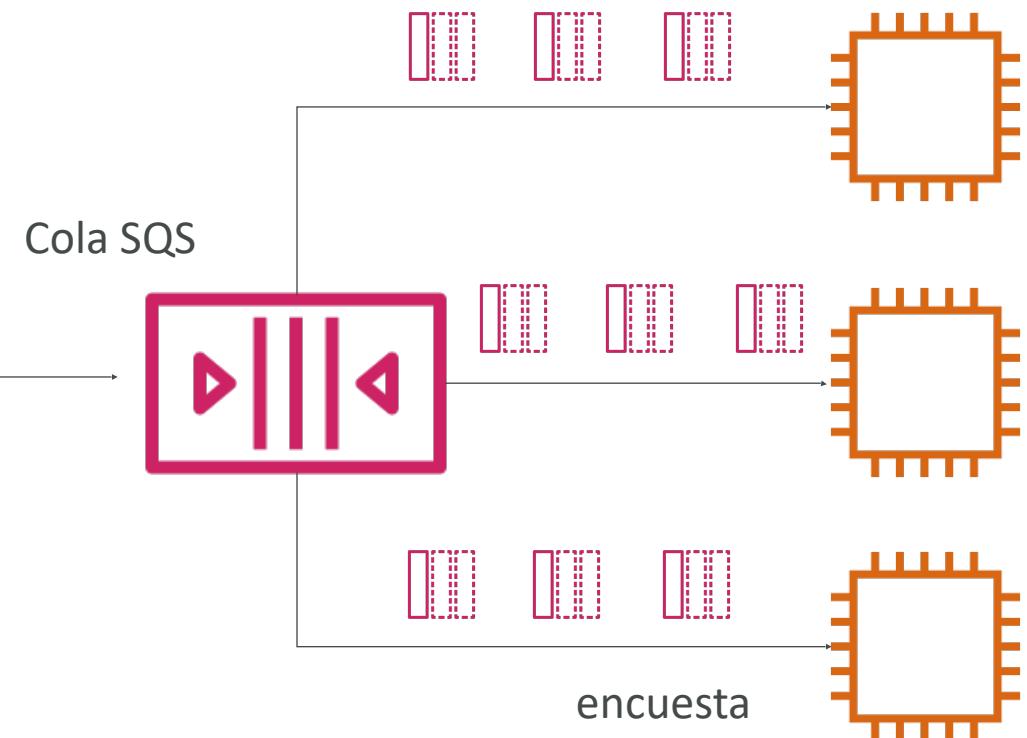
SQS – Consumir mensajes

- Consumidores (ejecutándose en instancias EC2, servidores o AWS Lambda)...
- Sondeo (encuestas) SQS en busca de mensajes (recibir hasta 10 mensajes a la vez)
- Procesar los mensajes (ejemplo: insertar el mensaje en una base de datos RDS)
- Eliminar los mensajes utilizando la API DeleteMessage



SQS

Consumidores de múltiples instancias EC2

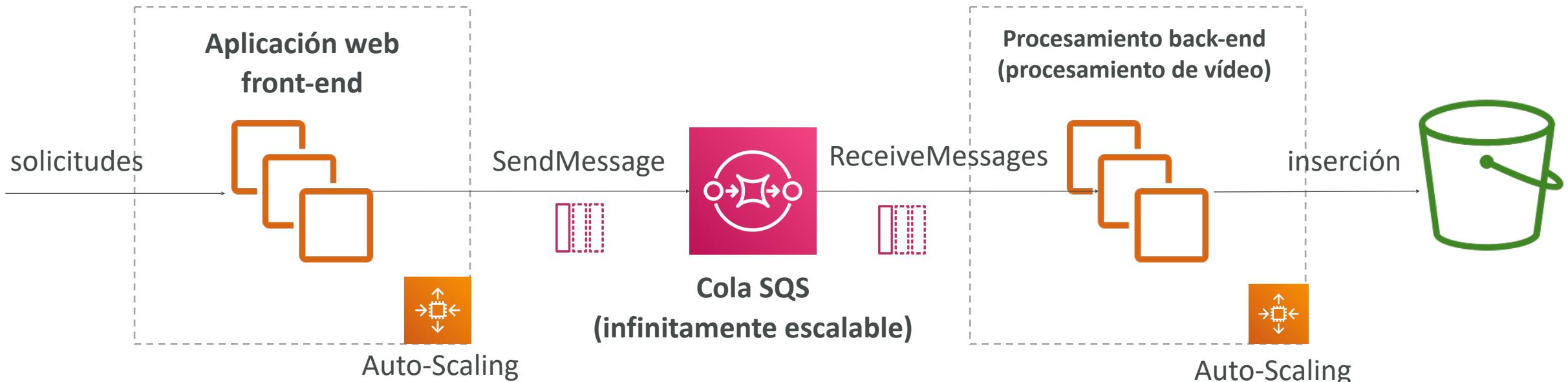


- Los consumidores reciben y procesan los mensajes en paralelo
- Al menos una entrega
- Ordenación de mensajes al mejor esfuerzo
- Los consumidores borran los mensajes después de procesarlos
- Podemos escalar los consumidores horizontalmente para mejorar el rendimiento del procesamiento

SQS con Auto Scaling Group (ASG)



SQS para **desacoplar** los niveles de aplicación

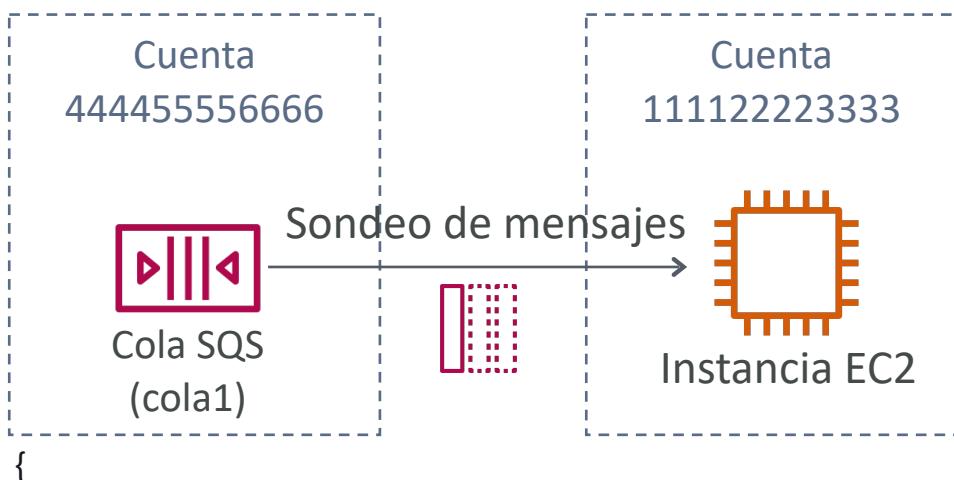


Amazon SQS - Seguridad

- **Cifrado:**
 - Cifrado en vuelo mediante API HTTPS
 - Cifrado en reposo mediante claves KMS
 - Cifrado del lado del cliente si el cliente desea realizar el cifrado/descifrado por sí mismo
- **Controles de acceso:** Políticas IAM para regular el acceso a la API SQS
- **Políticas de acceso a SQS** (similares a las políticas de bucket de S3)
 - Útil para el acceso entre cuentas a las colas SQS
 - Útil para permitir a otros servicios (SNS, S3...) escribir en una cola SQS

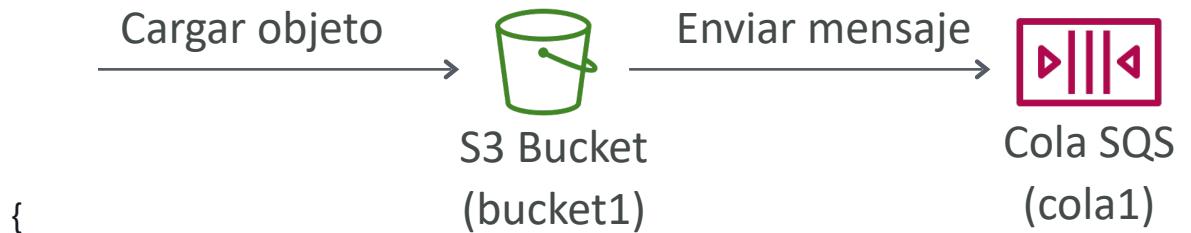
Política de acceso a la cola SQS

Acceso cruzado a cuentas



```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": { "AWS": [ "111122223333" ] },
      "Action": [ "sns:ReceiveMessage" ],
      "Resource": "arn:aws:sns:us-east-1:444455556666:queue1"
    }
  ]
}
```

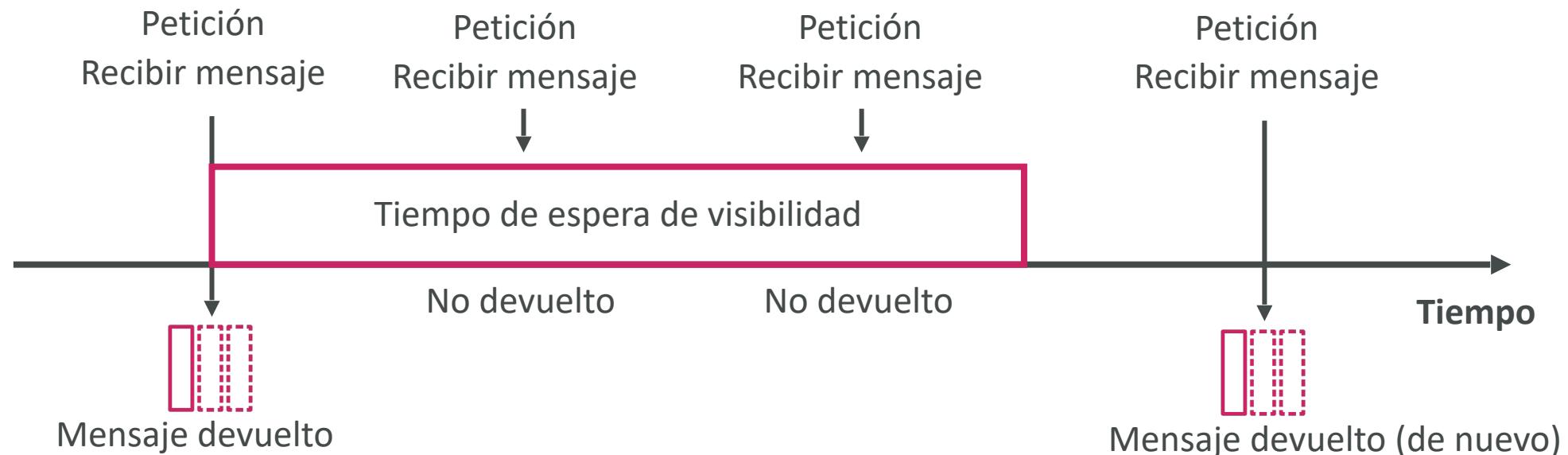
Publicar notificaciones de eventos S3 a la cola SQS



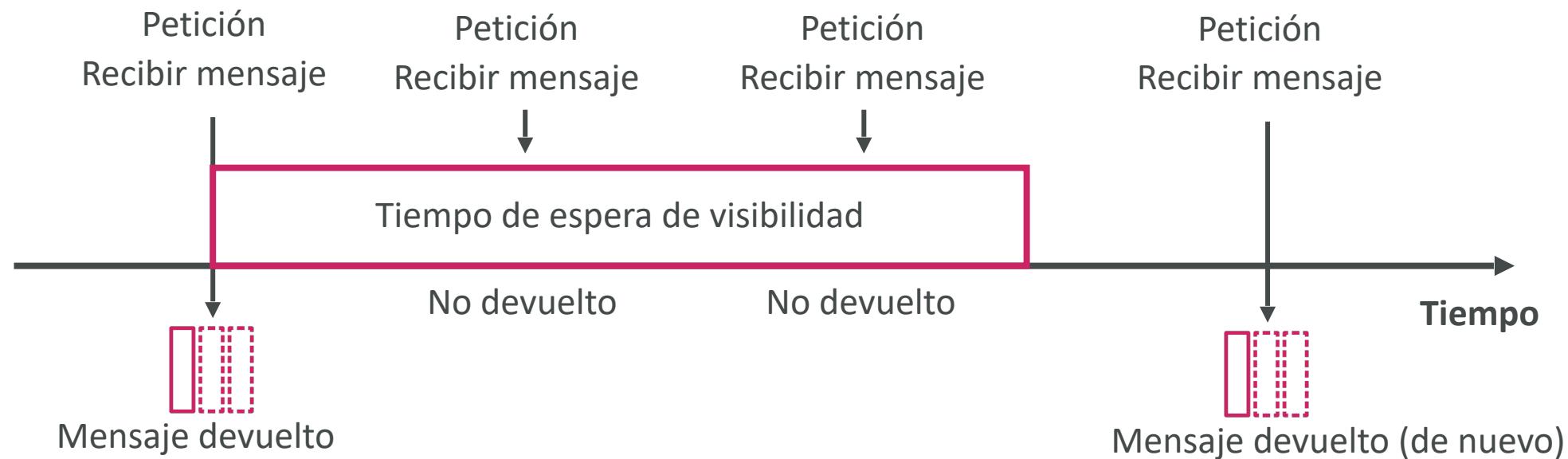
```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": { "AWS": "*" },
      "Action": [ "sns:Publish" ],
      "Resource": "arn:aws:sns:us-east-1:444455556666:queue1",
      "Condition": {
        "ArnLike": { "aws:SourceArn": "arn:aws:s3::*:bucket1" },
        "StringEquals": { "aws:SourceAccount": "<bucket1_owner_account_id>" }
      }
    }
  ]
}
```

SQS –Tiempo de espera de visibilidad de mensajes

- Despues de que un consumidor sondee un mensaje, éste se vuelve **invisible** para los demás consumidores
- Por defecto, el "tiempo de visibilidad del mensaje" **es de 30 segundos**
- Esto significa que el mensaje tiene 30 segundos para ser procesado
- Una vez transcurrido el tiempo de espera, el mensaje es "visible" en SQS.



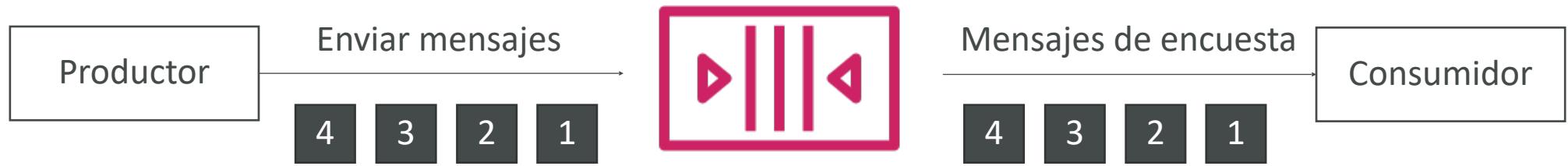
SQS – Tiempo de espera de visibilidad de mensajes



- Si un mensaje no se procesa dentro del tiempo de visibilidad, se procesará **dos veces**.
- El consumidor puede llamar a la API **ChangeMessageVisibility** para obtener más tiempo
- Si el tiempo de espera de visibilidad es alto (horas) y el consumidor se bloquea, el reprocesamiento llevará tiempo.
- Si el tiempo de visibilidad es demasiado bajo (segundos), puede haber duplicados.

Amazon SQS - Cola FIFO

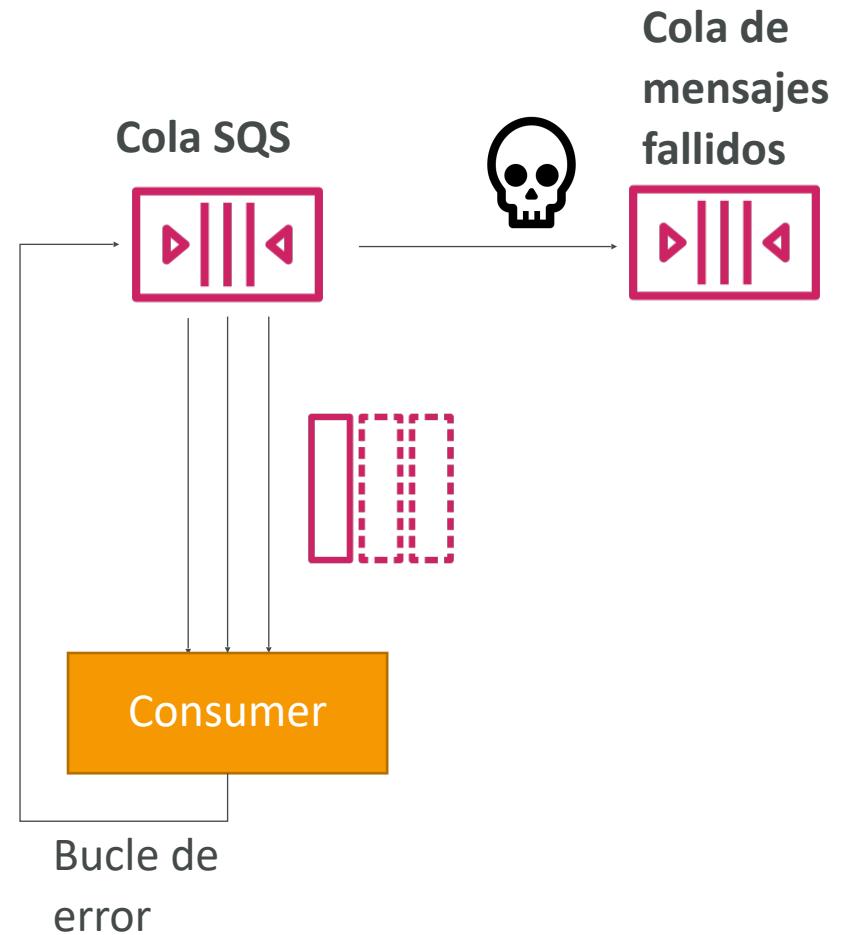
- FIFO = First In First Out (ordenación de los mensajes en la cola)



- Rendimiento limitado: 300 msg/s sin procesamiento por lotes, 3000 msg/s con procesamiento por lotes
- Capacidad de envío exactamente una vez (eliminando los duplicados)
- El consumidor procesa los mensajes en orden

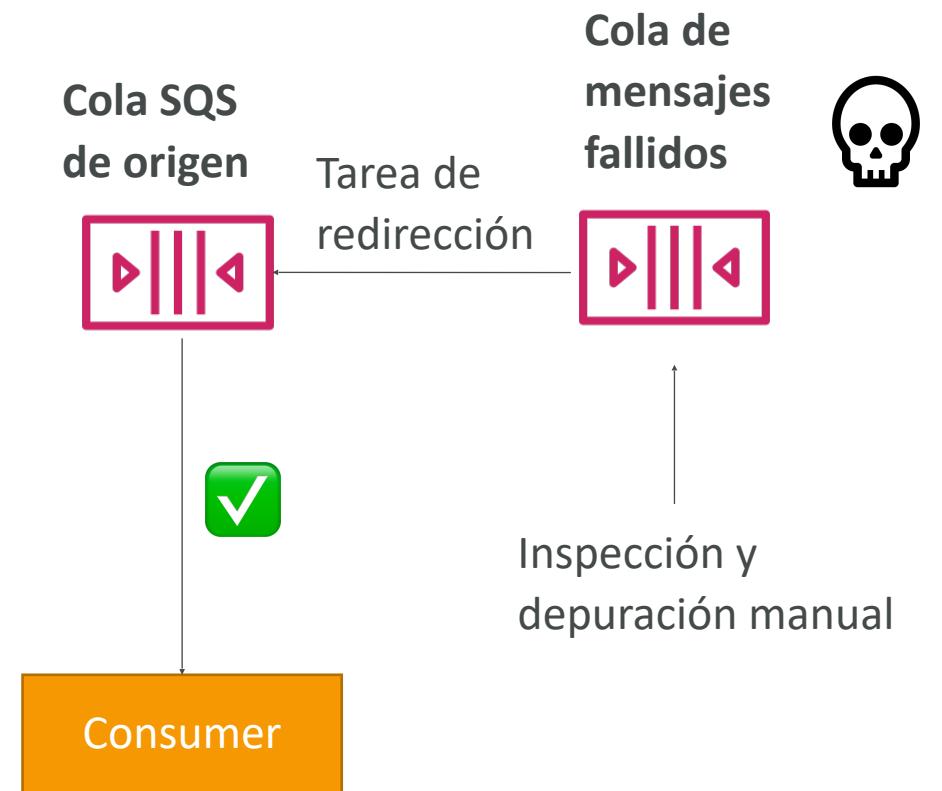
Amazon SQS – Cola de mensajes fallidos (DLQ)

- Si un consumidor no procesa un mensaje dentro del tiempo de espera de visibilidad... ¡el mensaje vuelve a la cola!
- Podemos establecer un umbral de cuántas veces puede volver un mensaje a la cola
- Una vez superado el umbral de **MaximumReceives**, el mensaje pasa a una cola de mensajes fallidos (DLQ)
- ¡Útil para depurar!
- **DLQ de una cola FIFO debe ser también una cola FIFO**
- **DLQ de una cola Estándar también debe ser una cola Estándar**
- Asegúrate de procesar los mensajes de la DLQ antes de que caduquen:
 - Conviene fijar una retención de 14 días en la DLQ



SQS DLQ - Redirigir al origen

- Característica que ayuda a consumir mensajes en la DLQ para entender qué les pasa
- Cuando nuestro código esté arreglado, podremos volver a conducir los mensajes de la DLQ a la cola de origen (o a cualquier otra cola) por lotes sin escribir código personalizado



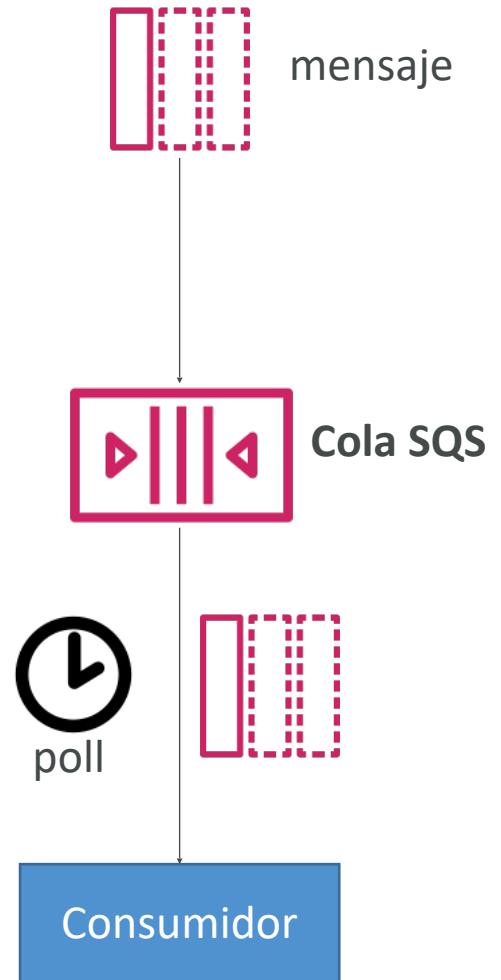
Amazon SQS – Cola de espera

- Retrasa un mensaje (los consumidores no lo ven inmediatamente) hasta 15 minutos
- El valor por defecto es 0 segundos (el mensaje está disponible inmediatamente)
- Puedes establecer un valor por defecto a nivel de cola
- Puedes anular el valor por defecto en el envío utilizando el parámetro **DelaySeconds**



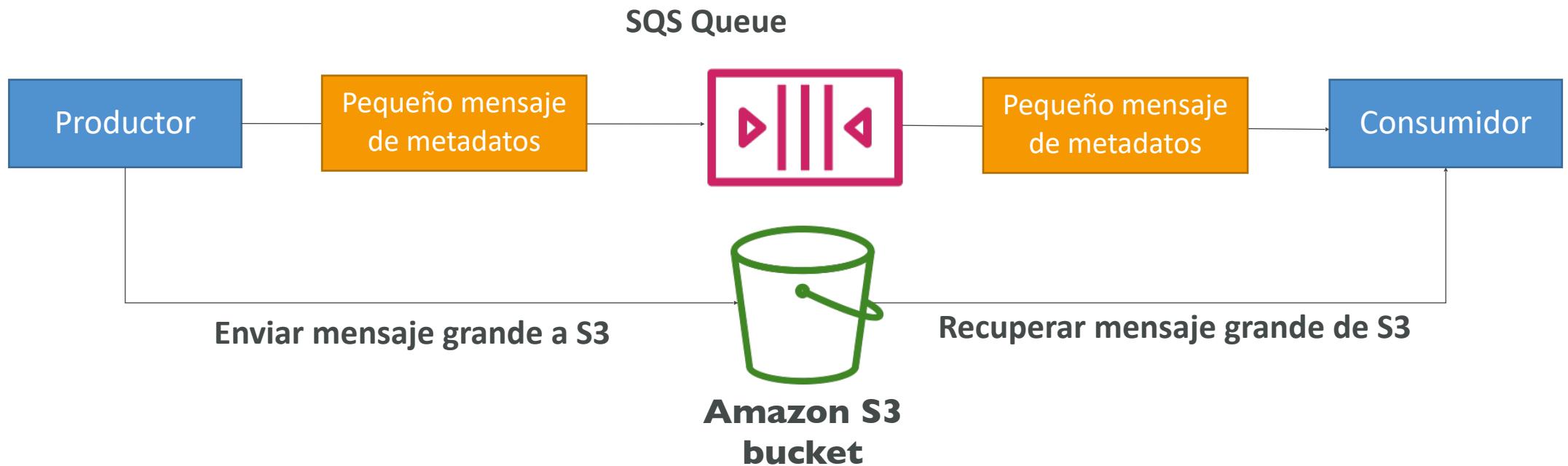
Amazon SQS - Sondeo largo (Long Polling)

- Cuando un consumidor solicita mensajes de la cola, puede opcionalmente "esperar" a que lleguen los mensajes si no hay ninguno en la cola
- Esto se llama Sondeo Largo
- **LongPolling disminuye el número de llamadas API realizadas a SQS, al tiempo que aumenta la eficiencia y reduce la latencia de tu aplicación**
- El tiempo de espera puede oscilar entre 1 y 20 segundos (preferiblemente 20 segundos)
- El sondeo largo es preferible al sondeo corto
- El sondeo largo puede activarse a nivel de cola o a nivel de API utilizando **WaitTimeSeconds**



Cliente extendido SQS

- El límite de tamaño de los mensajes es de 256 KB, ¿cómo enviar mensajes grandes, por ejemplo de 1 GB?
- Utilizando el Cliente Extendido SQS (Biblioteca Java)

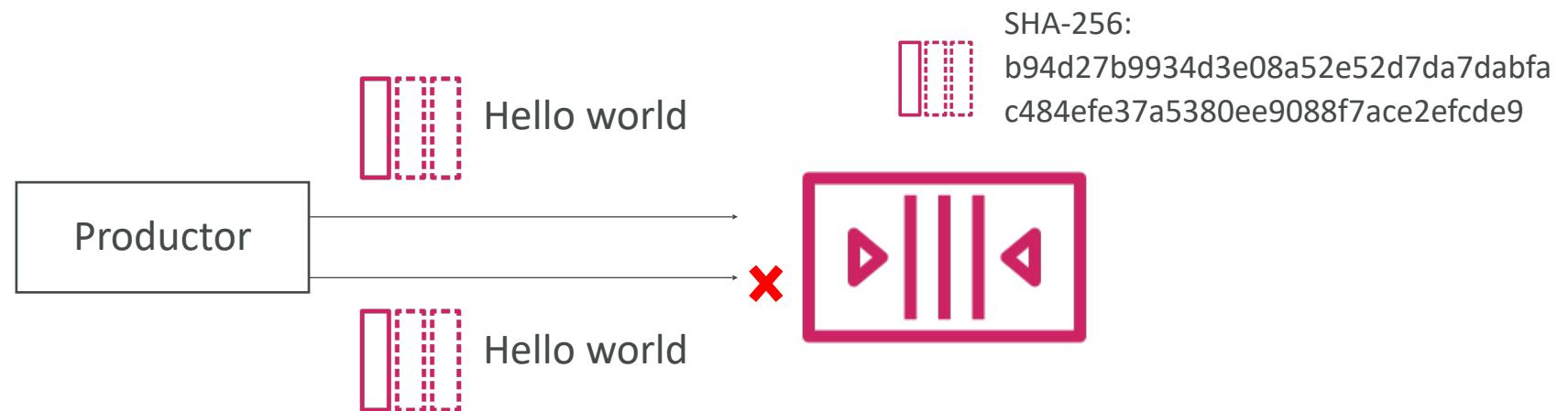


SQS - Conocimiento imprescindible de la API

- **CreateQueue (MessageRetentionPeriod), DeleteQueue**
- **PurgeQueue**: borra todos los mensajes de la cola
- **SendMessage (DelaySeconds), ReceiveMessage, DeleteMessage**
- **MaxNumberOfMessages**: por defecto 1, máximo 10 (para la API ReceiveMessage)
- **ReceiveMessageWaitTimeSeconds**: Sondeo largo
- **ChangeMessageVisibility**: cambiar el tiempo de espera del mensaje
- Las API por lotes para **SendMessage, DeleteMessage, ChangeMessageVisibility** te ayudan a reducir costes

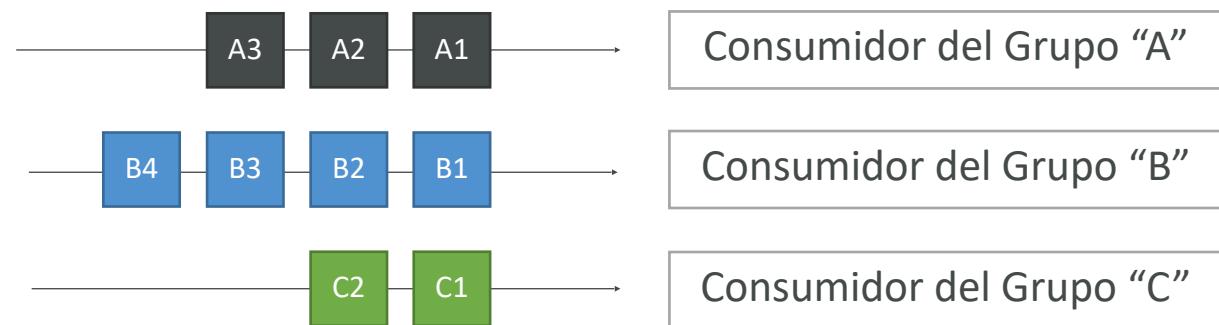
SQS FIFO - Deduplicación

- El intervalo de deduplicación es de 5 minutos
- Dos métodos de deduplicación:
 - Deduplicación basada en el contenido: hará un hash SHA-256 del cuerpo del mensaje
 - Proporciona explícitamente un ID de deduplicación del mensaje



SQS FIFO - Agrupación de mensajes

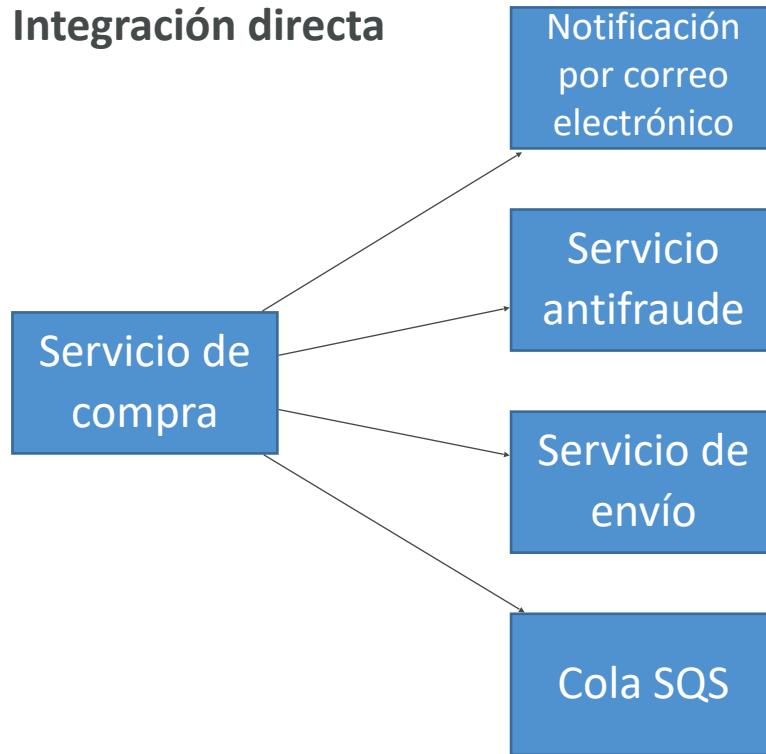
- Si especificas el mismo valor de **MessageGroupId** en una cola SQS FIFO, sólo puedes tener un consumidor, y todos los mensajes estarán ordenados
- Para obtener el orden a nivel de un subconjunto de mensajes, especifica valores diferentes para **MessageGroupId**
 - Los mensajes que comparten un ID de Grupo de mensajes común estarán en orden dentro del grupo
 - Cada ID de Grupo puede tener un consumidor diferente (¡procesamiento paralelo!)
 - No se garantiza el orden entre grupos



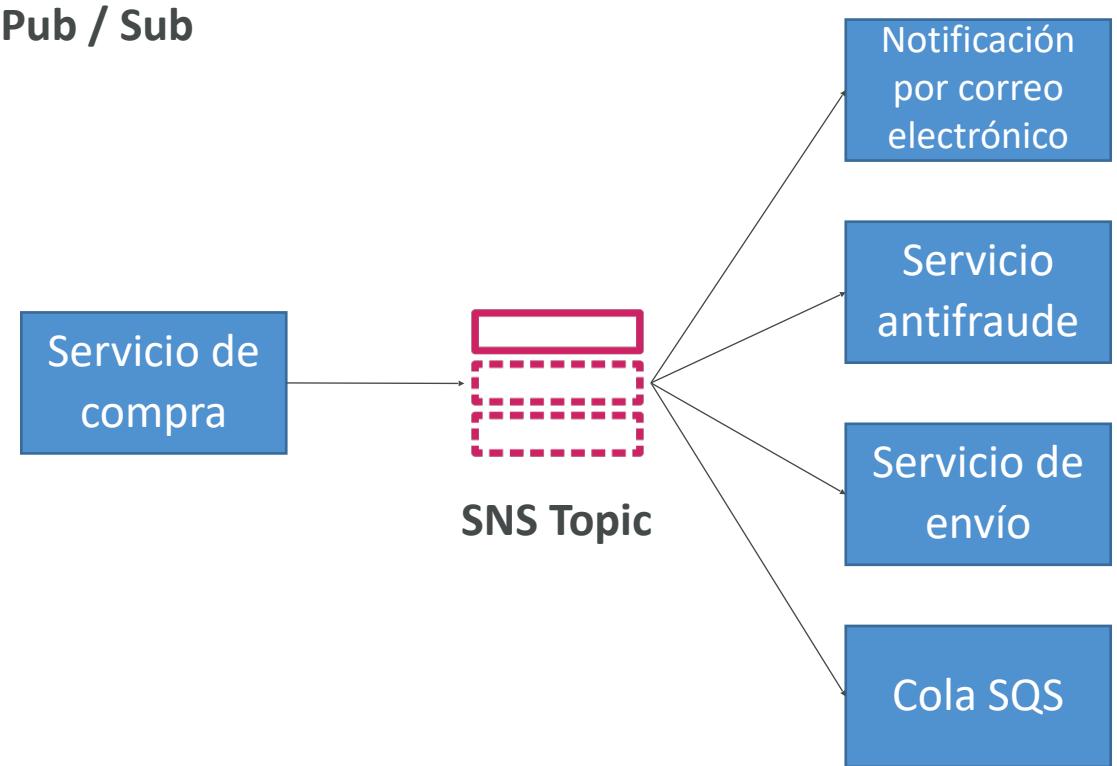
Amazon SNS

- ¿Y si quieres enviar un mensaje a muchos destinatarios?

Integración directa



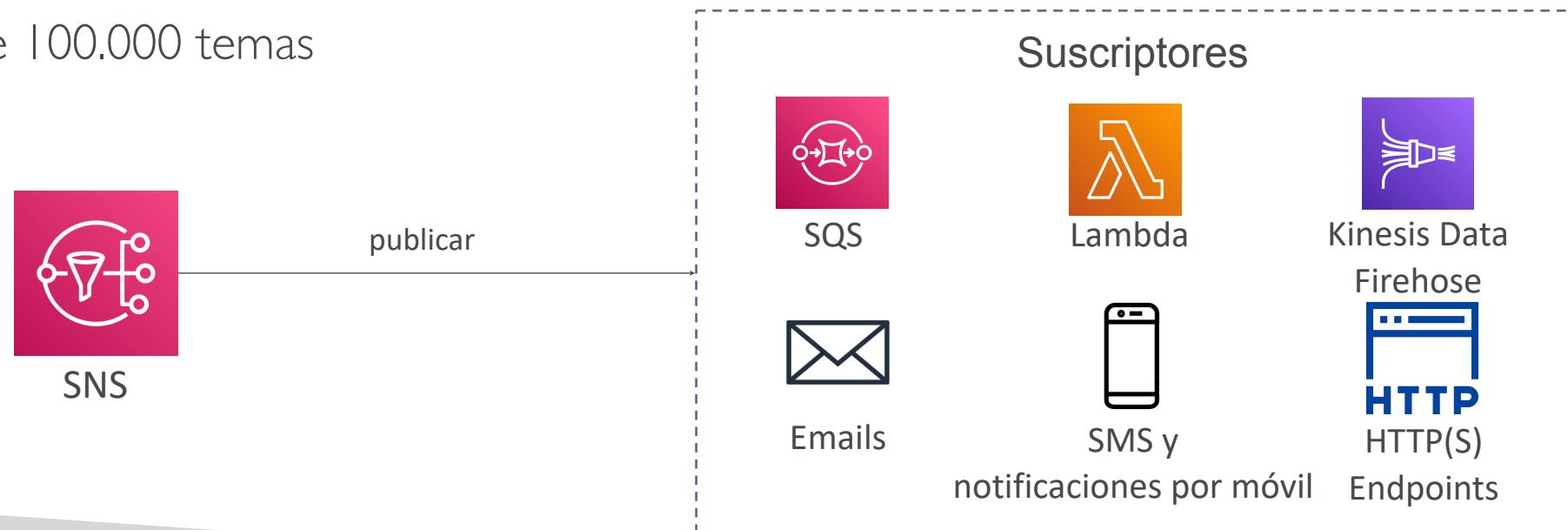
Pub / Sub



Amazon SNS

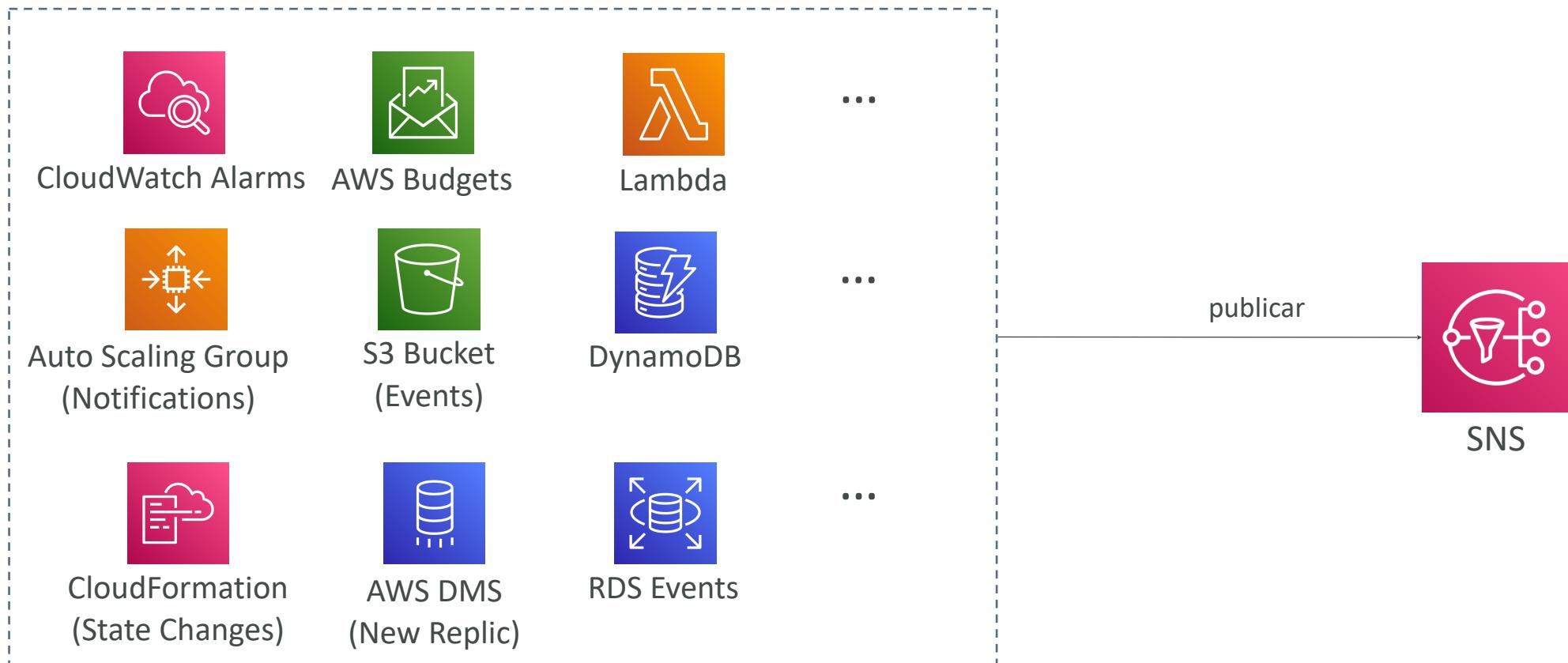


- El "productor de eventos" sólo envía mensajes a un tema SNS
- Tantos "receptores de eventos" (suscriptores) como queramos para escuchar las notificaciones del tema SNS
- Cada suscriptor al tema recibirá todos los mensajes (nota: nueva función para filtrar mensajes)
- Hasta 12.500.000 suscripciones por tema
- Límite de 100.000 temas



SNS se integra con muchos servicios de AWS

- Muchos servicios de AWS pueden enviar datos directamente a SNS para notificaciones



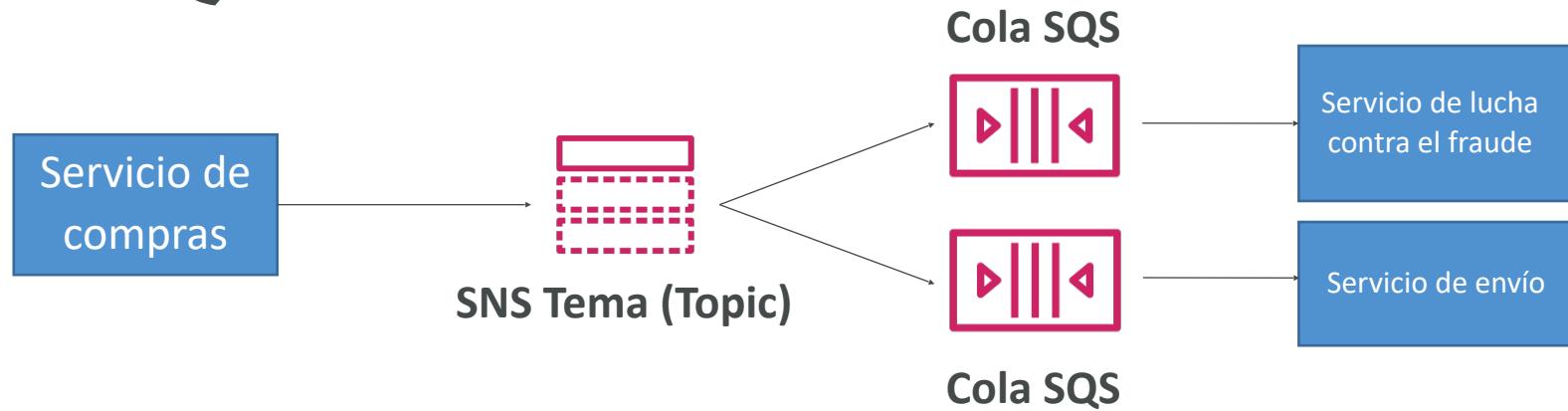
Amazon SNS - Cómo publicar

- Publicación de temas (mediante el SDK)
 - Crear un tema
 - Crea una suscripción (o varias)
 - Publicar en el tema
- Publicación directa (para aplicaciones móviles SDK)
 - Crear una aplicación de plataforma
 - Crear un punto final de plataforma
 - Publicar en el punto final de la plataforma
 - Funciona con Google GCM, Apple APNS, Amazon ADM...

Amazon SNS – Seguridad

- **Cifrado:**
 - Cifrado en vuelo mediante API HTTPS
 - Cifrado en reposo mediante claves KMS
 - Cifrado del lado del cliente si el cliente desea realizar el cifrado/descifrado por sí mismo
- **Controles de acceso:** Políticas IAM para regular el acceso a la API SNS
- **Políticas de acceso SNS** (similares a las políticas de bucket S3)
 - Útil para el acceso entre cuentas a temas SNS
 - Útil para permitir que otros servicios (S3...) escriban en un tema SNS

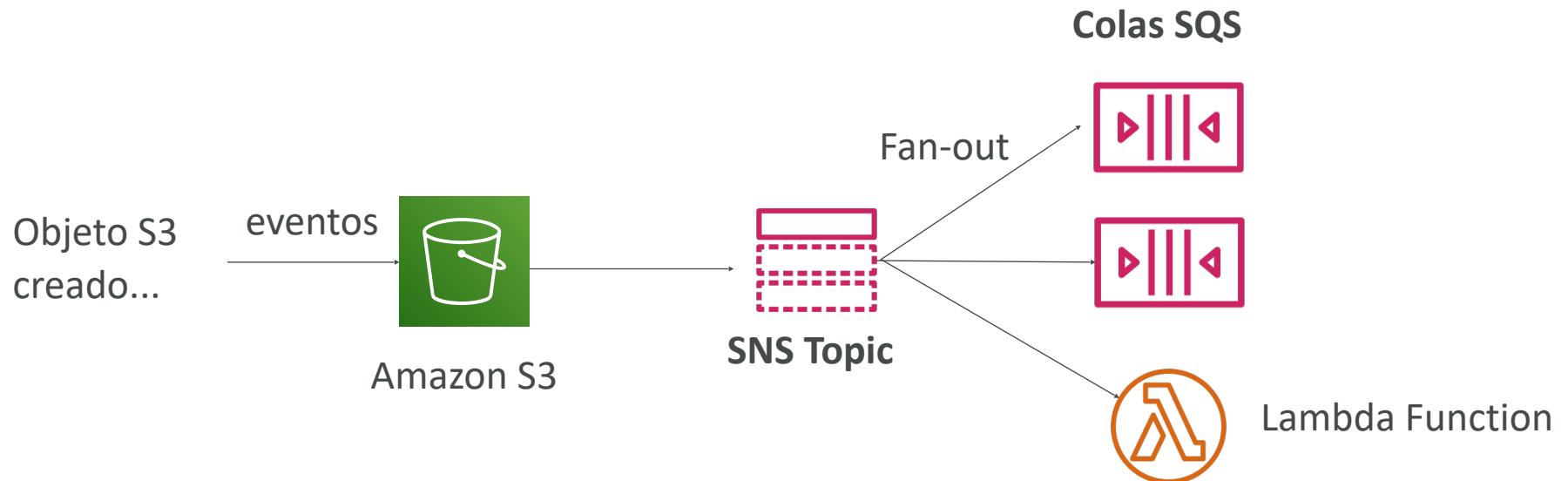
SNS + SQS: Fan Out



- Haz el push una vez en SNS, recibe en todas las colas SQS que son suscriptores
- Totalmente desacoplado, sin pérdida de datos
- SQS permite: persistencia de datos, procesamiento diferido y reintentos de trabajo
- Posibilidad de añadir más suscriptores SQS con el tiempo
- Asegúrate de que la **política de acceso** a la cola SQS permite que SNS pueda escribir

Aplicación: Eventos S3 a múltiples colas

- Para la misma combinación de: **tipo de evento** (p.e. creación de objeto) y **prefijo** (p.e. imágenes/) sólo puedes tener una regla de Evento S3
- Si quieres enviar el mismo evento S3 a muchas colas SQS, utiliza fan-out



Aplicación: SNS a Amazon S3 a través de Kinesis Data Firehose

- SNS puede enviar a Kinesis y por lo tanto podemos tener la siguiente arquitectura de soluciones:



Amazon SNS - Tema (topic) FIFO

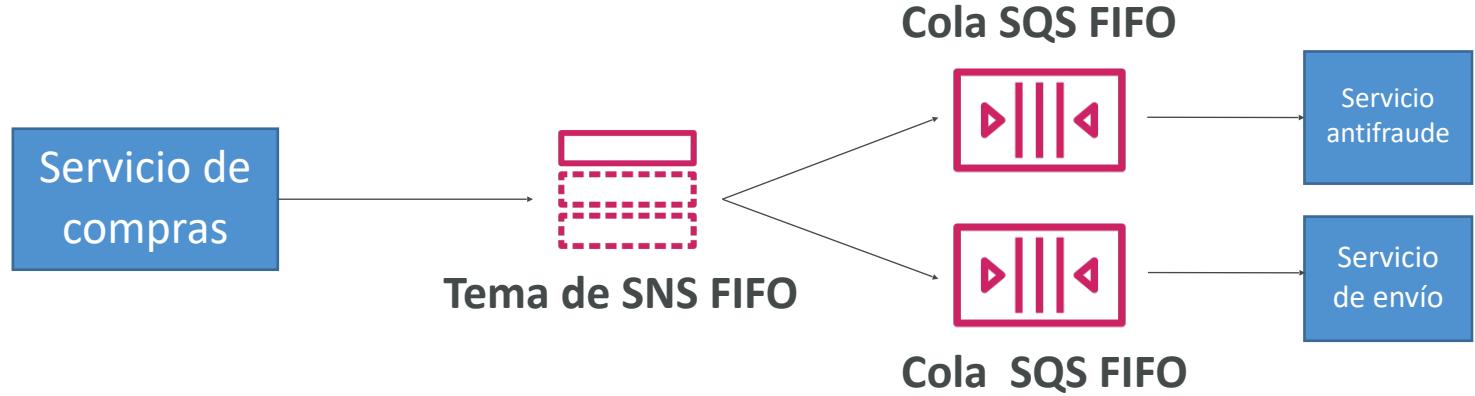
- FIFO = First In First Out (orden de los mensajes en el tema)



- Características similares a SQS FIFO:
 - **Ordenación** por ID de grupo de mensajes (se ordenan todos los mensajes del mismo grupo)
 - **Deduplicación** mediante ID de deduplicación o deduplicación basada en contenido
- Sólo puede tener colas SQS FIFO como suscriptores
- Rendimiento limitado (el mismo que SQS FIFO)

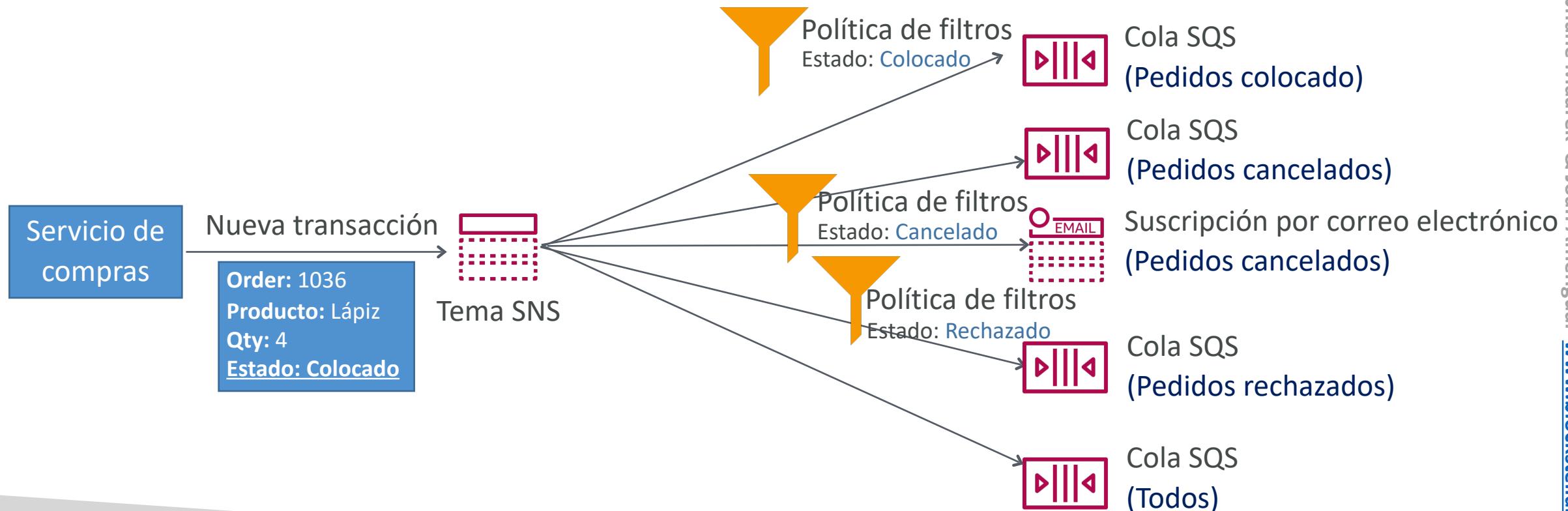
SNS FIFO + SQS FIFO: Fan Out

- En caso de que necesites fan-out + ordenación + deduplicación

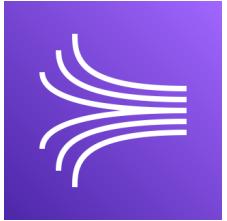


SNS - Filtrado de mensajes

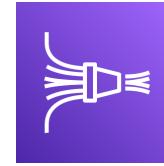
- Política JSON utilizada para filtrar los mensajes enviados a las suscripciones del tema SNS
- Si una suscripción no tiene una política de filtrado, recibe todos los mensajes



Visión general de Kinesis

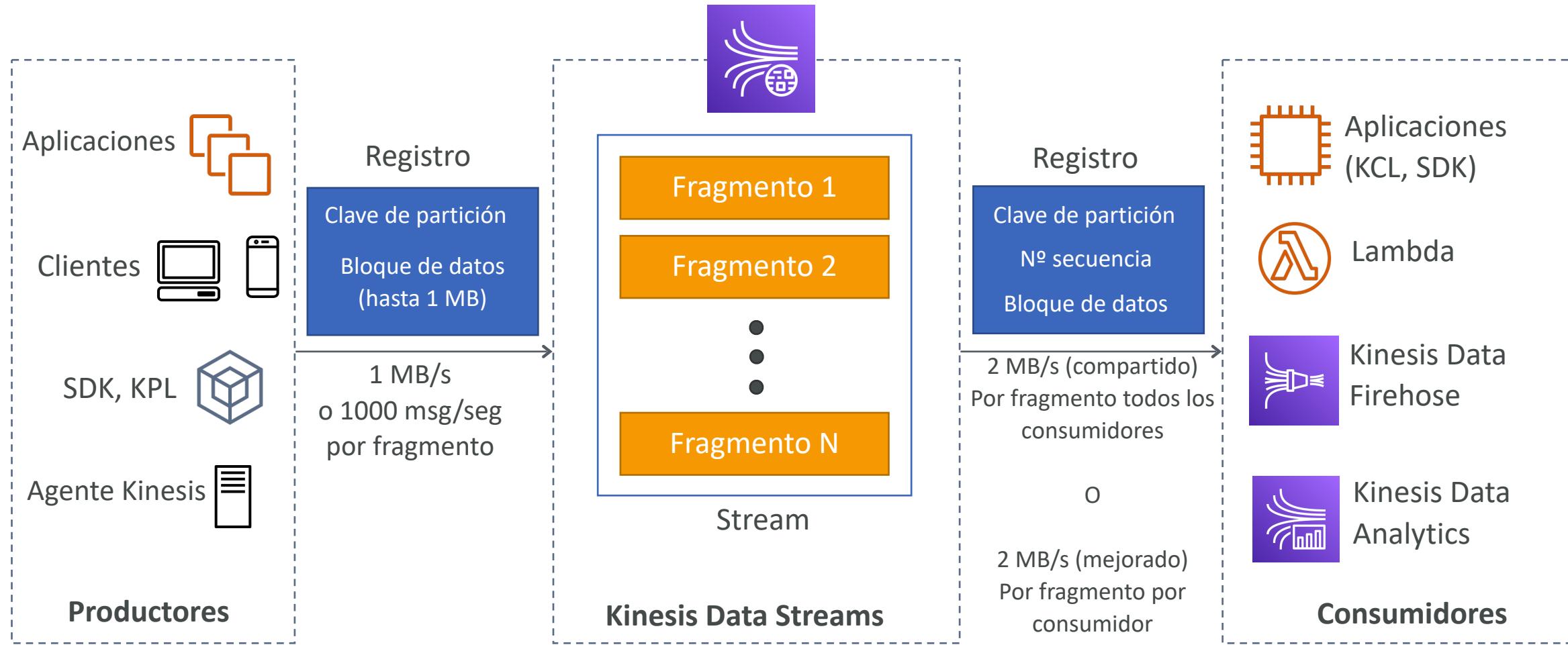


- Facilita la **recopilación**, el **procesamiento** y el **análisis** de datos de flujo continuo en tiempo real
- Ingesta de datos en tiempo real como: Registros de aplicaciones, métricas, secuencias de clics de sitios web, datos telemétricos de IoT...



- **Kinesis Data Streams**: captura, procesa y almacena flujos de datos
- **Kinesis Data Firehose**: carga flujos de datos en almacenes de datos de AWS
- **Kinesis Data Analytics**: analiza flujos de datos con SQL o Apache Flink
- **Kinesis Video Streams**: captura, procesa y almacena transmisiones de vídeo

Kinesis Data Streams





Kinesis Data Streams

- Retención entre 1 día y 365 días
- Posibilidad de volver a procesar (reproducir) los datos
- Una vez que los datos se insertan en Kinesis, no pueden borrarse (inmutabilidad)
- Los datos que comparten la misma partición van al mismo fragmento (ordenación)
- Productores: SDK de AWS, biblioteca de productores de Kinesis (KPL), agente de Kinesis
- Consumidores:
 - Escriba el suyo propio: Kinesis Client Library (KCL), AWS SDK
 - Administrados: AWS Lambda, Kinesis Data Firehose, Kinesis Data Analytics,

Kinesis Data Streams - Modos de capacidad

- **Modo aprovisionado:**

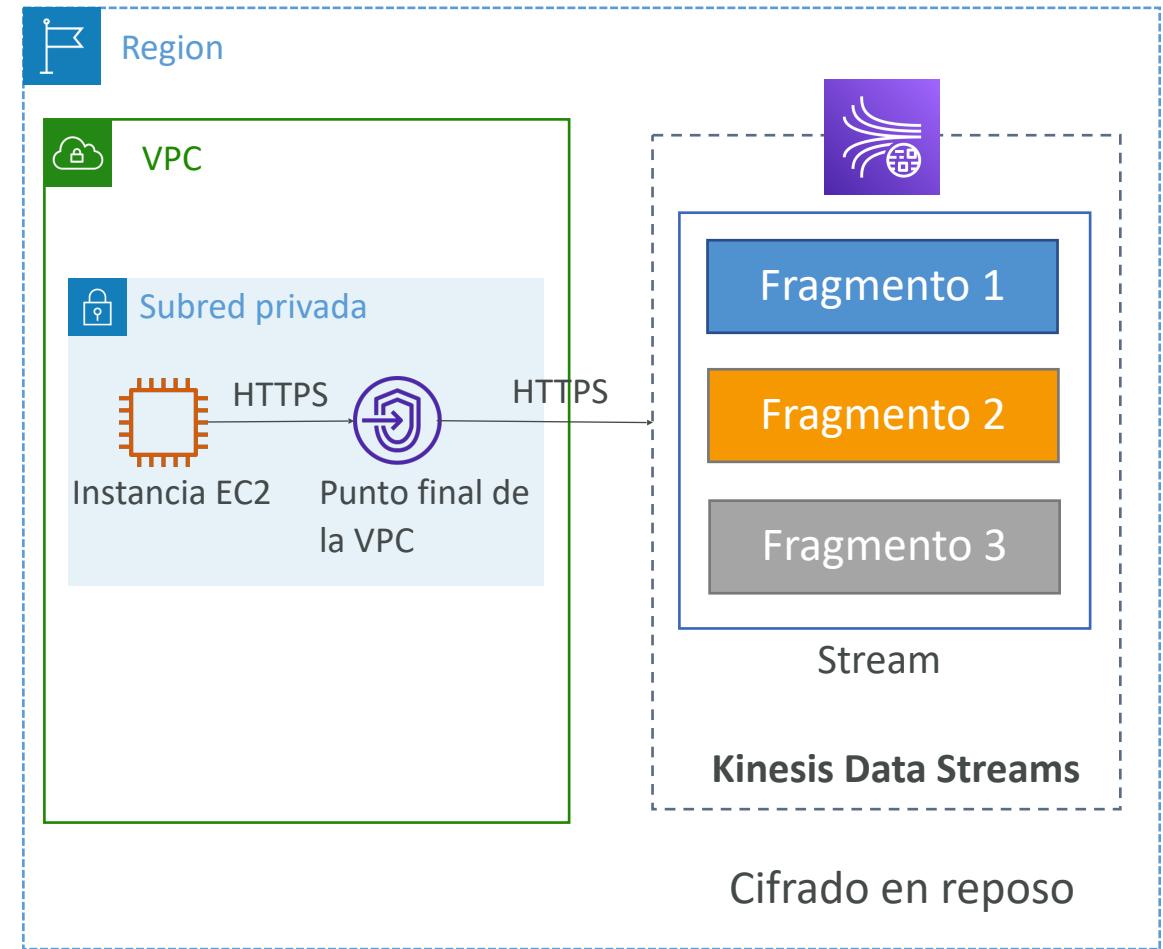
- Tú eliges el número de shards aprovisionados, escala manualmente o usando API
- Cada fragmento recibe 1 MB/s (o 1000 registros por segundo)
- Cada fragmento recibe 2 MB/s de salida (consumo en fan-out clásico o mejorado)
- Se paga por cada fragmento aprovisionado por hora

- **Modo bajo demanda:**

- No es necesario aprovisionar ni gestionar la capacidad
- Capacidad provisionada por defecto (4 MB/s de entrada o 4000 registros por segundo)
- Escala automáticamente en función del pico de rendimiento observado durante los últimos 30 días
- Pago por flujo por hora y entrada/salida de datos por GB

Seguridad de Kinesis Data Streams

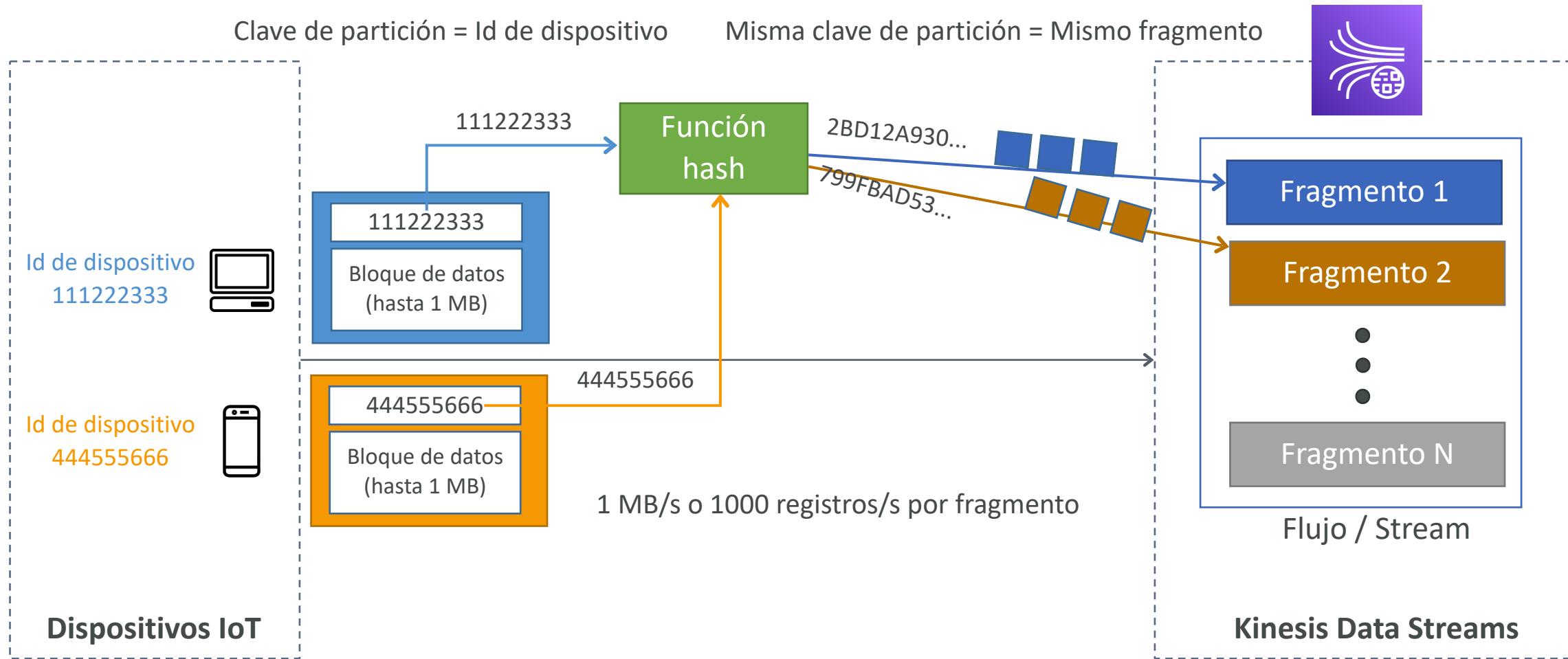
- Control de acceso / autorización mediante políticas IAM
- Cifrado en vuelo mediante puntos finales HTTPS
- Cifrado en reposo mediante KMS
- Puede implementar el cifrado/ descifrado de datos en el lado del cliente (más difícil)
- Puntos finales VPC disponibles para que Kinesis acceda dentro de VPC
- Supervisión de las llamadas a la API mediante CloudTrail



Productores Kinesis

- Coloca registros de datos en streams de datos
- El registro de datos consta de:
 - Número de secuencia (único por clave de partición dentro del fragmento)
 - Clave de partición (debe especificarse al poner los registros en el stream)
 - Bloque de datos (hasta 1 MB)
- Productores:
 - **AWS SDK**: productor simple
 - **Biblioteca de productores Kinesis (KPL)**: C++, Java, por lotes, compresión, reintentos
 - **Agente Kinesis**: monitoriza los logs
- Velocidad de escritura: 1 MB/seg o 1000 registros/seg por fragmento
- API PutRecord
- Utiliza el procesamiento por lotes con la API PutRecords para reducir costes y aumentar el rendimiento

Productores Kinesis



Kinesis - ProvisionedThroughputExceeded

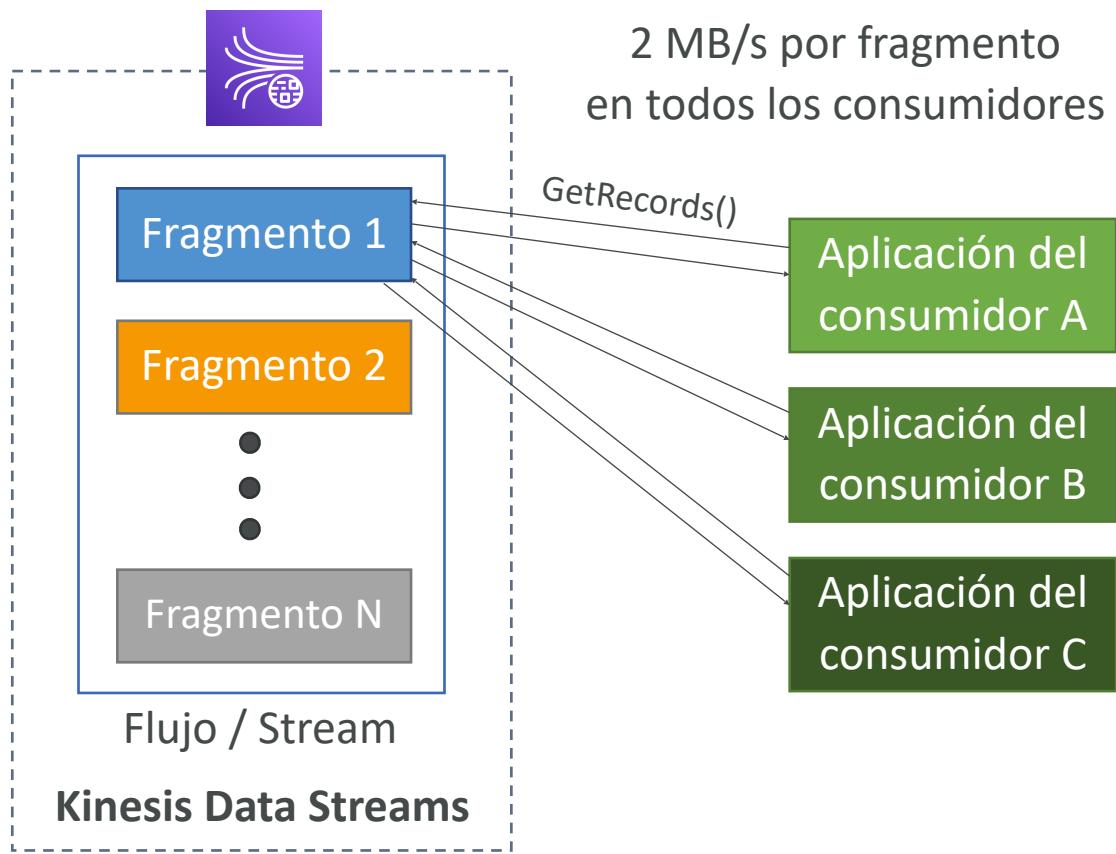


Consumidores de Kinesis Data Streams

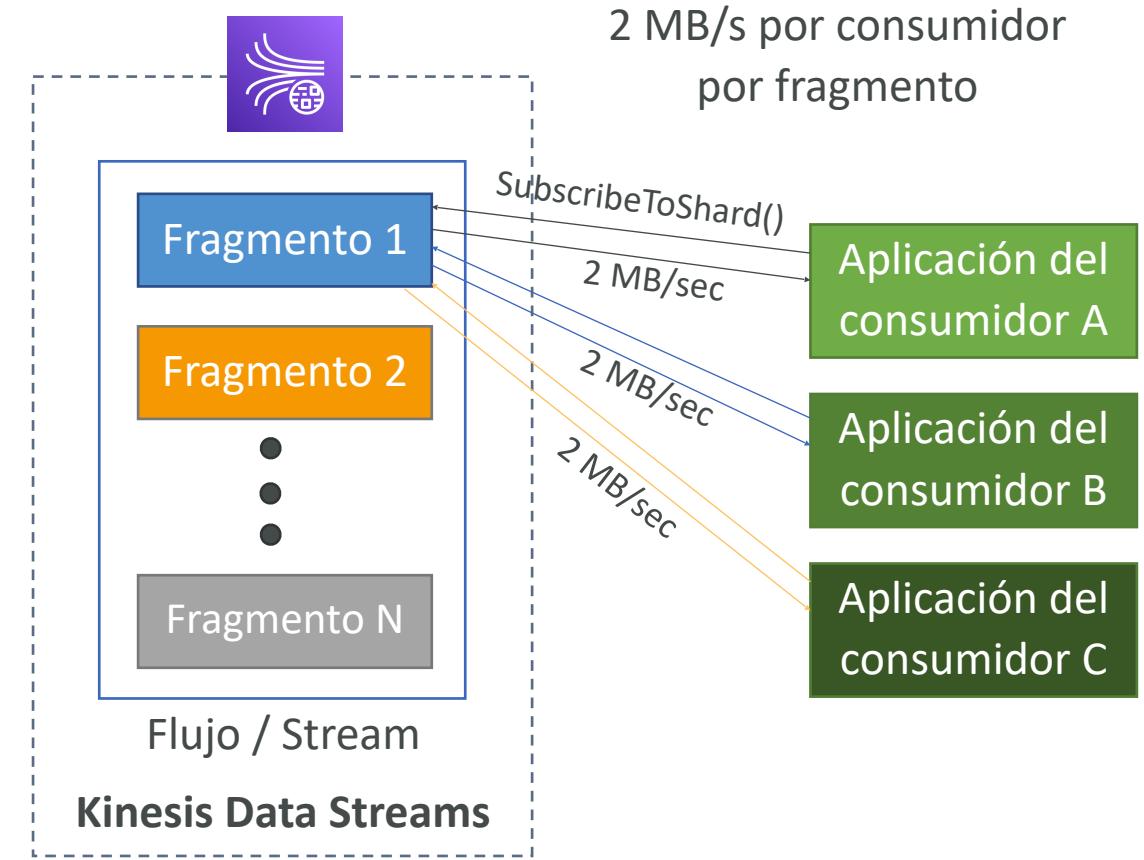
- Obtener registros de datos de streams de datos y procesarlos
- AWS Lambda
- Kinesis Data Analytics
- Kinesis Data Firehose
- Consumidor personalizado (AWS SDK) - Clásico o Fan-Out mejorado
- Kinesis Client Library (KCL): biblioteca para simplificar la lectura del stream de datos

Consumidores Kinesis - Consumidor personalizado

Consumidor en Fan-out compartido (clásico)



Consumidor Fan-out mejorado



Tipos de consumidores Kinesis

Consumidor en Fan-out compartido (clásico) - pull

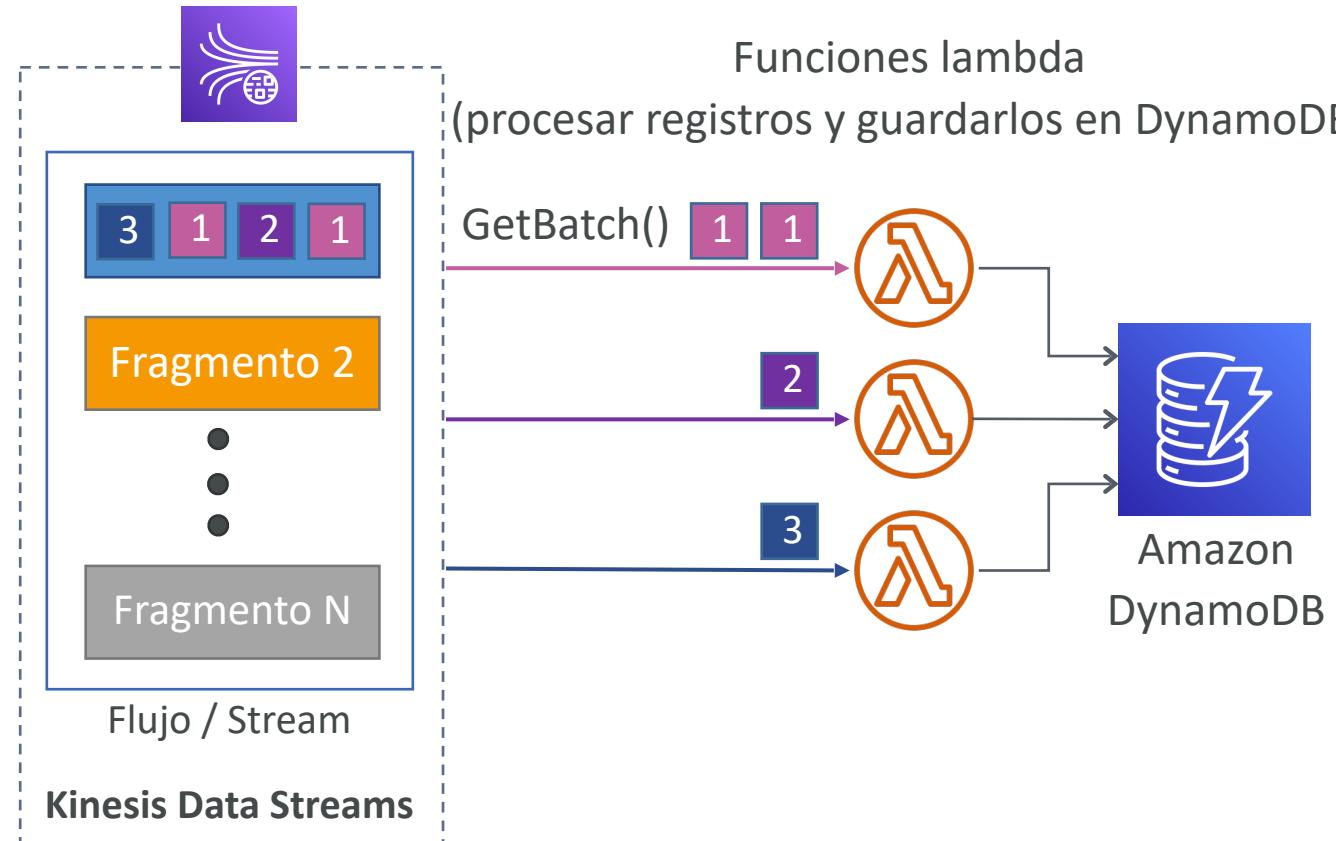
- Bajo número de aplicaciones consumidoras
- Rendimiento de lectura: 2 MB/s por fragmento en todos los consumidores
- Máx. 5 llamadas a la API GetRecords/seg
- Latencia ~200 ms
- Minimizar el coste (\$)
- Los consumidores consultan los datos de Kinesis mediante una llamada a la API GetRecords
- Devuelve hasta 10 MB (luego se ralentiza durante 5 segundos) o hasta 10000 registros

Consumidor en Fan-out mejorado - push

- Múltiples aplicaciones consumidoras del mismo stream
- 2 MB/seg por consumidor por fragmento
- Latencia ~70 ms
- Mayores costes (\$\$\$)
- Kinesis envía datos a los consumidores a través de HTTP/2 (API SubscribeToShard)
- Límite suave de 5 aplicaciones consumidoras (KCL) por stream de datos (por defecto)

Consumidores Kinesis - AWS Lambda

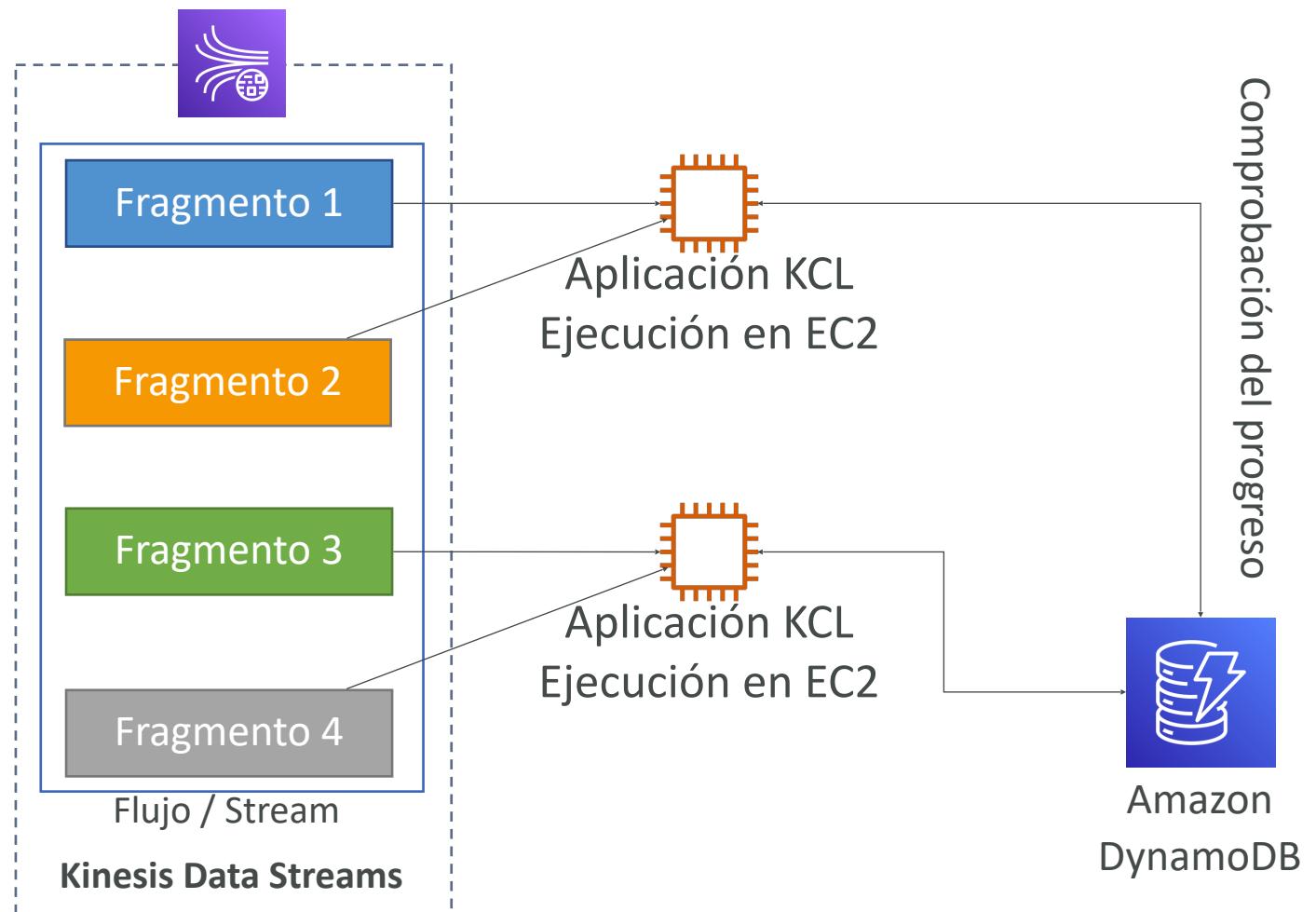
- Soporta consumidores en Fan-Out clásico y mejorado
- Lee registros por lotes (batches)
- Puedes configurar el **tamaño del lote** y la **ventana del lote**
- Si se produce un error, Lambda reintenta hasta que tenga éxito o los datos caduquen
- Puede procesar hasta 10 lotes por fragmento simultáneamente



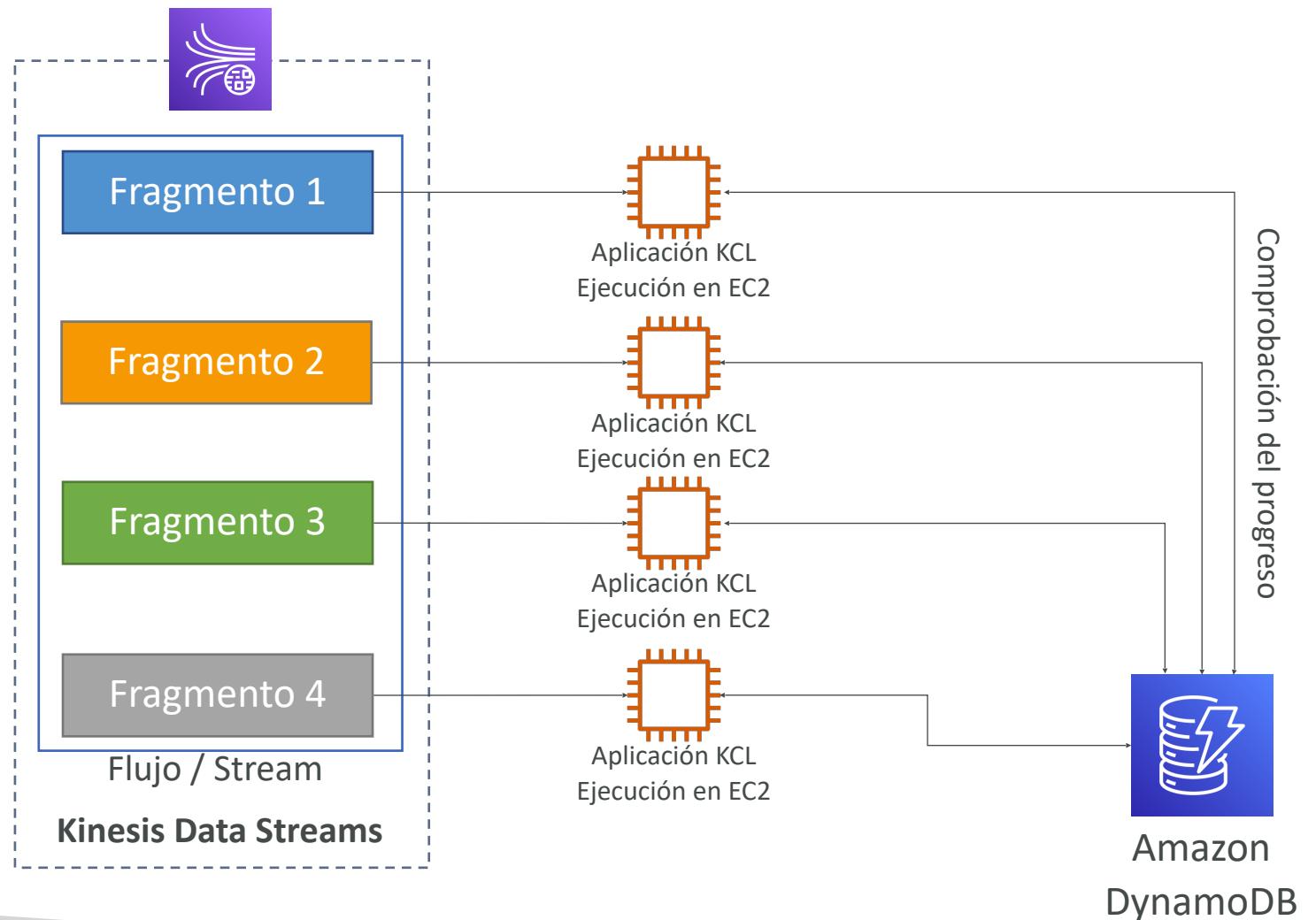
Kinesis Client Library (KCL)

- Una biblioteca Java que ayuda a leer registros de un stream de Kinesis Data con aplicaciones distribuidas que comparten la carga de trabajo de lectura
- Cada fragmento debe ser leído por una sola instancia de KCL
 - 4 fragmentos = máx. 4 instancias KCL
 - 6 fragmentos = máx. 6 instancias KCL
- El progreso se comprueba en DynamoDB (necesita acceso IAM)
- Realiza un seguimiento de otros trabajadores y comparte el trabajo entre fragmentos utilizando DynamoDB
- KCL puede ejecutarse en EC2, Elastic Beanstalk y en las propias instalaciones
- Los registros se leen en orden a nivel de fragmento
- Versiones:
 - KCL 1.x (soporta consumidor compartido)
 - KCL 2.x (soporta consumidor compartido y fan-out mejorado)

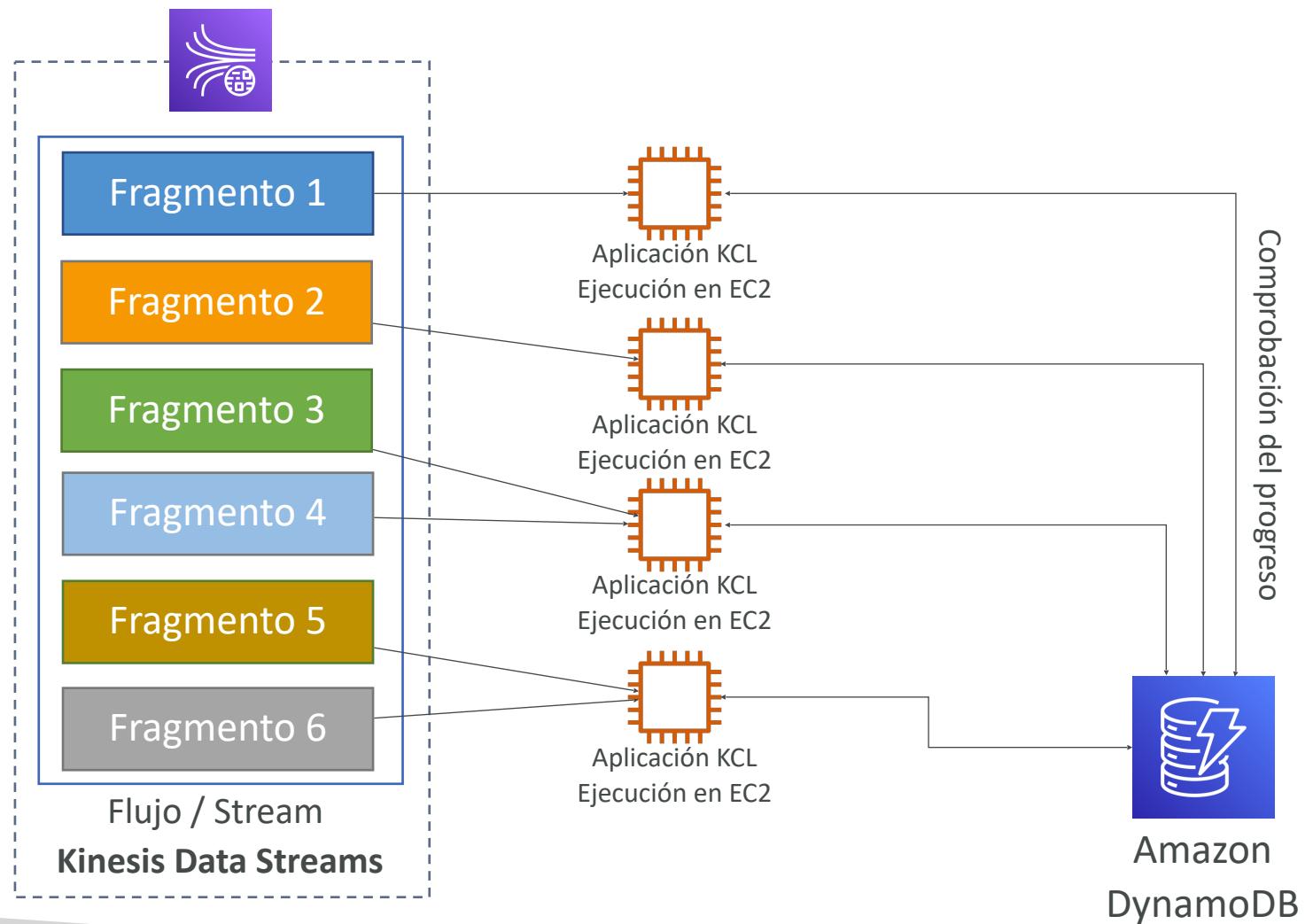
Ejemplo de KCL: 4 fragmentos



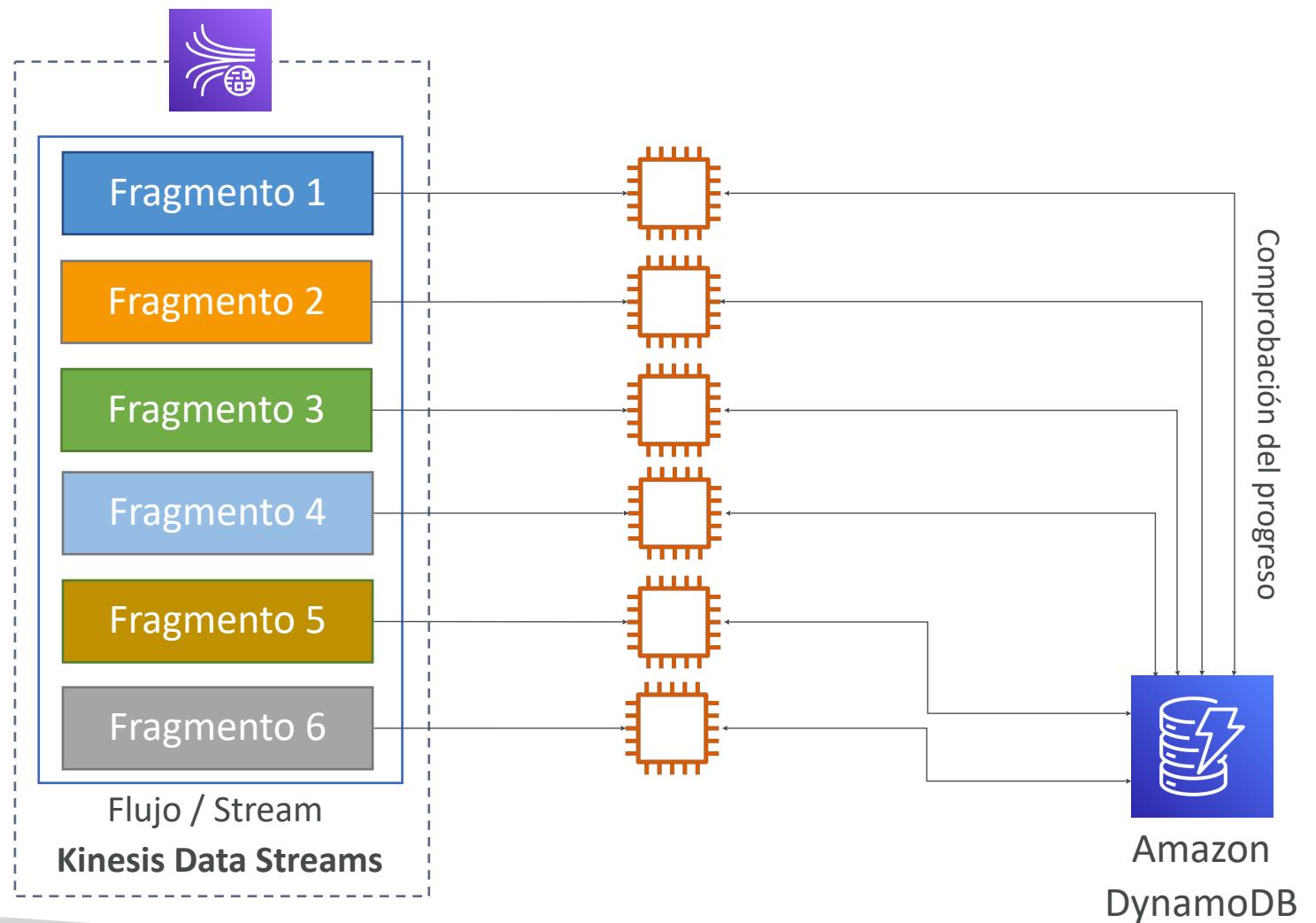
Ejemplo de KCL: 4 fragmentos, aplicación KCL a escala



Ejemplo de KCL: 6 fragmentos, Kinesis escalable

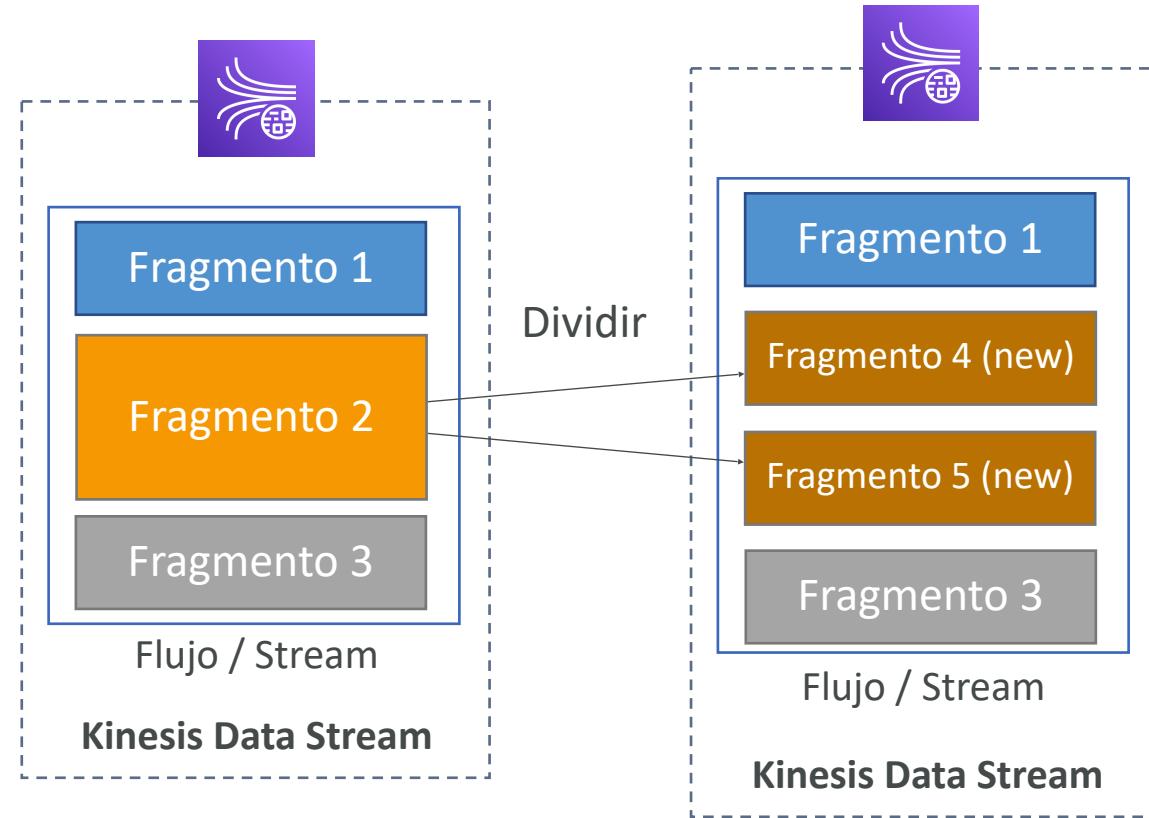


Ejemplo de KCL: 6 fragmentos, escalado de la aplicación KCL



Operación Kinesis - Dividir fragmentos

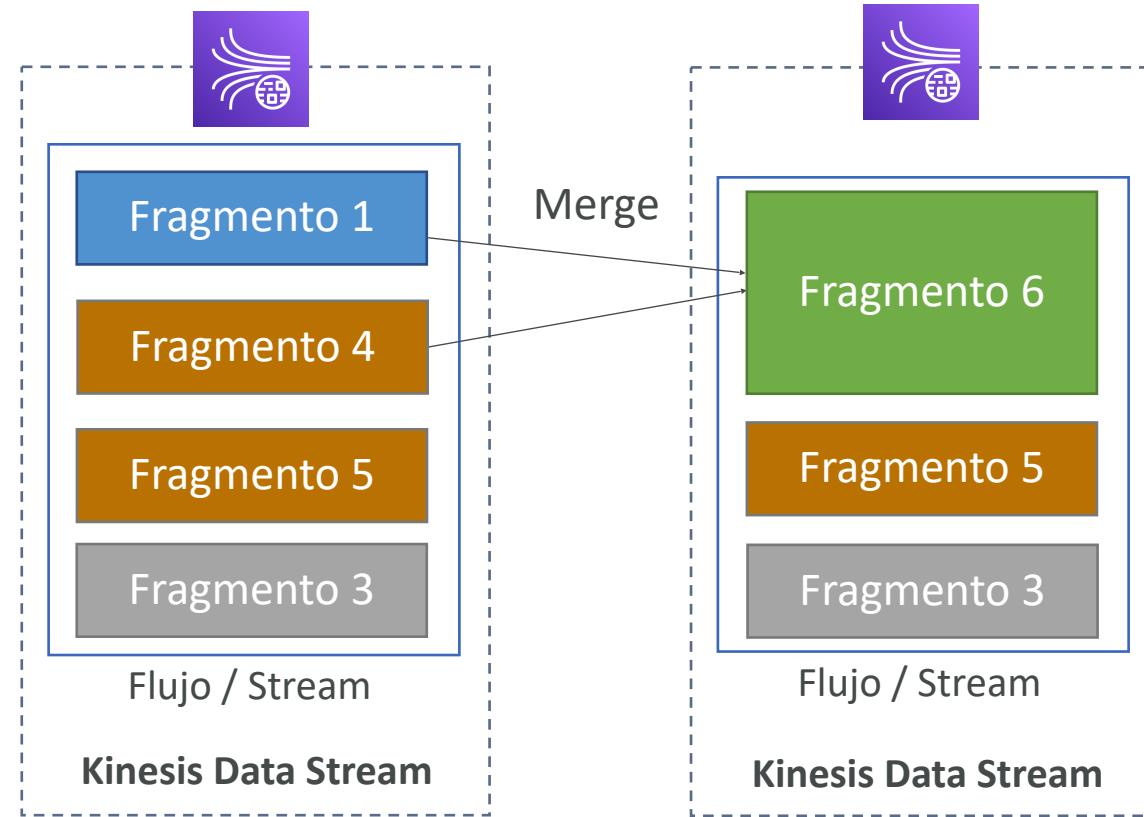
- Se utiliza para aumentar la capacidad del stream (1 MB/s de datos por fragmento)
- Se utiliza para dividir un “fragmento caliente”
- El antiguo fragmento se cierra y se elimina cuando caducan los datos
- No hay escalado automático (aumenta/diminuye la capacidad manualmente)
- No se puede dividir en más de dos shards en una sola operación



Aumentar la capacidad y el coste

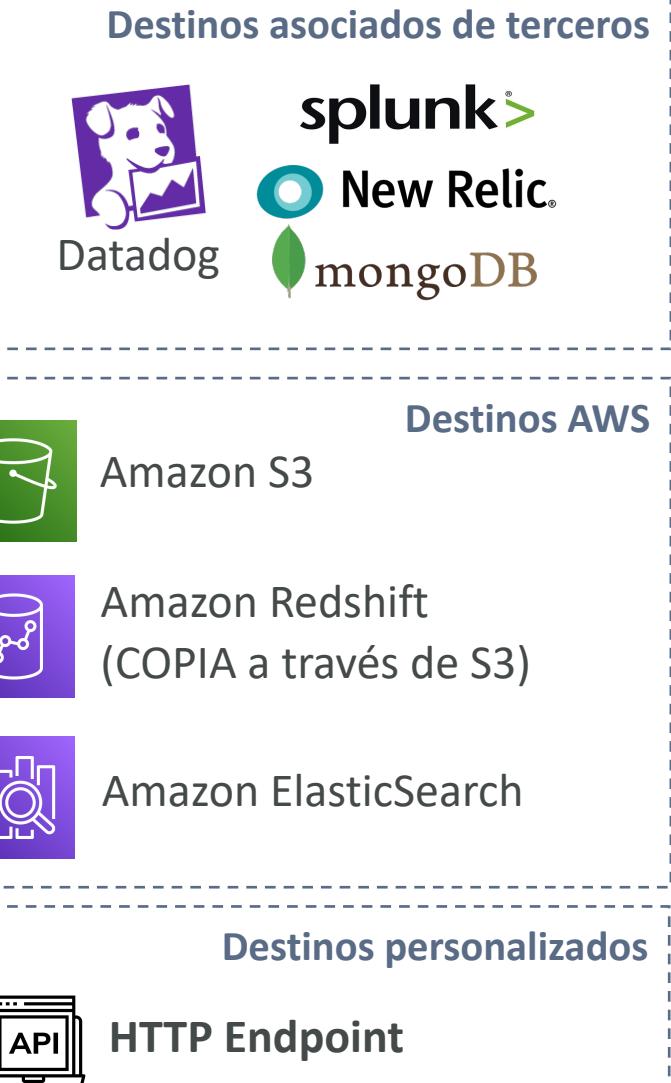
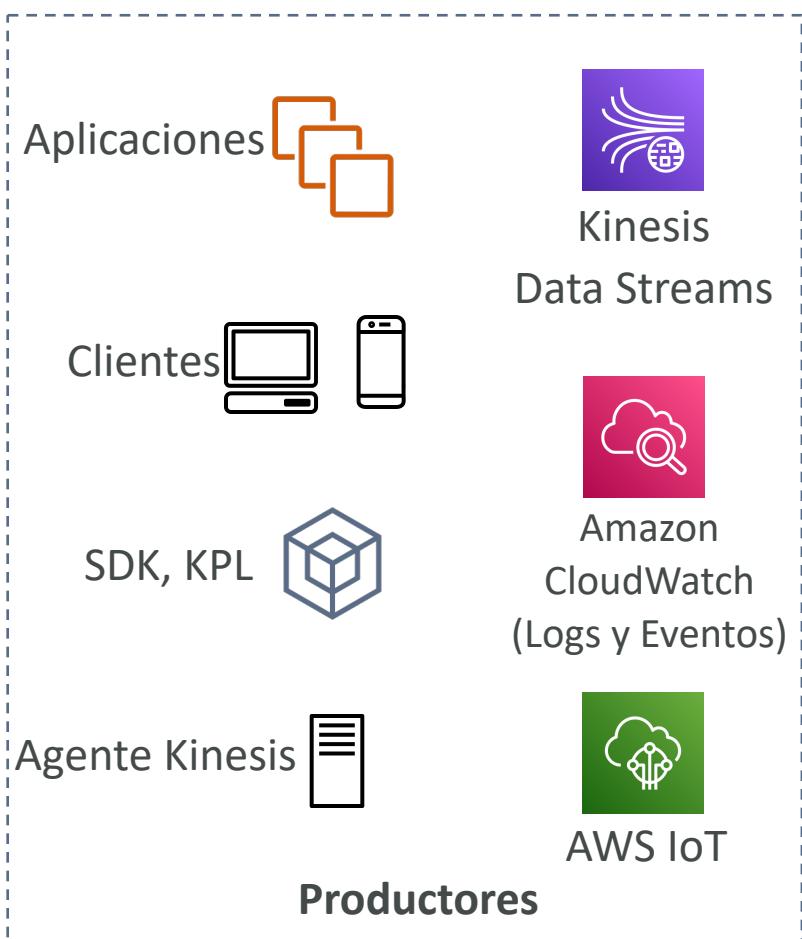
Operación Kinesis - Merge (fusión) de fragmentos

- Reduce la capacidad del stream y ahorra costes
- Puede utilizarse para agrupar dos shards con poco tráfico (shards fríos)
- Los shards antiguos se cierran y se borrarán cuando caduquen los datos
- No se pueden fusionar más de dos shards en una sola operación

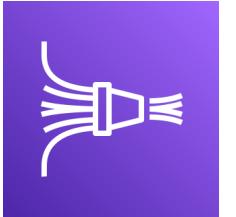


Disminuir la capacidad y el coste

Kinesis Data Firehose



Kinesis Data Firehose



- Servicio totalmente administrado, sin administración, escalado automático, sin servidor
 - AWS: Redshift / Amazon S3 / ElasticSearch
 - Socio de terceros: Splunk / MongoDB / DataDog / NewRelic / ...
 - Personalizado: enviar a cualquier punto final HTTP
- Pague por los datos que pasan por Firehose
- **Casi en tiempo real**
 - Latencia mínima de 60 segundos para lotes no completos
 - Un mínimo de 1 MB de datos a la vez
- Admite muchos formatos de datos, conversiones, transformaciones y compresión
- Admite transformaciones de datos personalizadas mediante AWS Lambda
- Puede enviar datos fallidos o todos los datos a un bucket de S3 de backup

Kinesis Data Streams vs Kinesis Data Firehose



Kinesis Data Streams

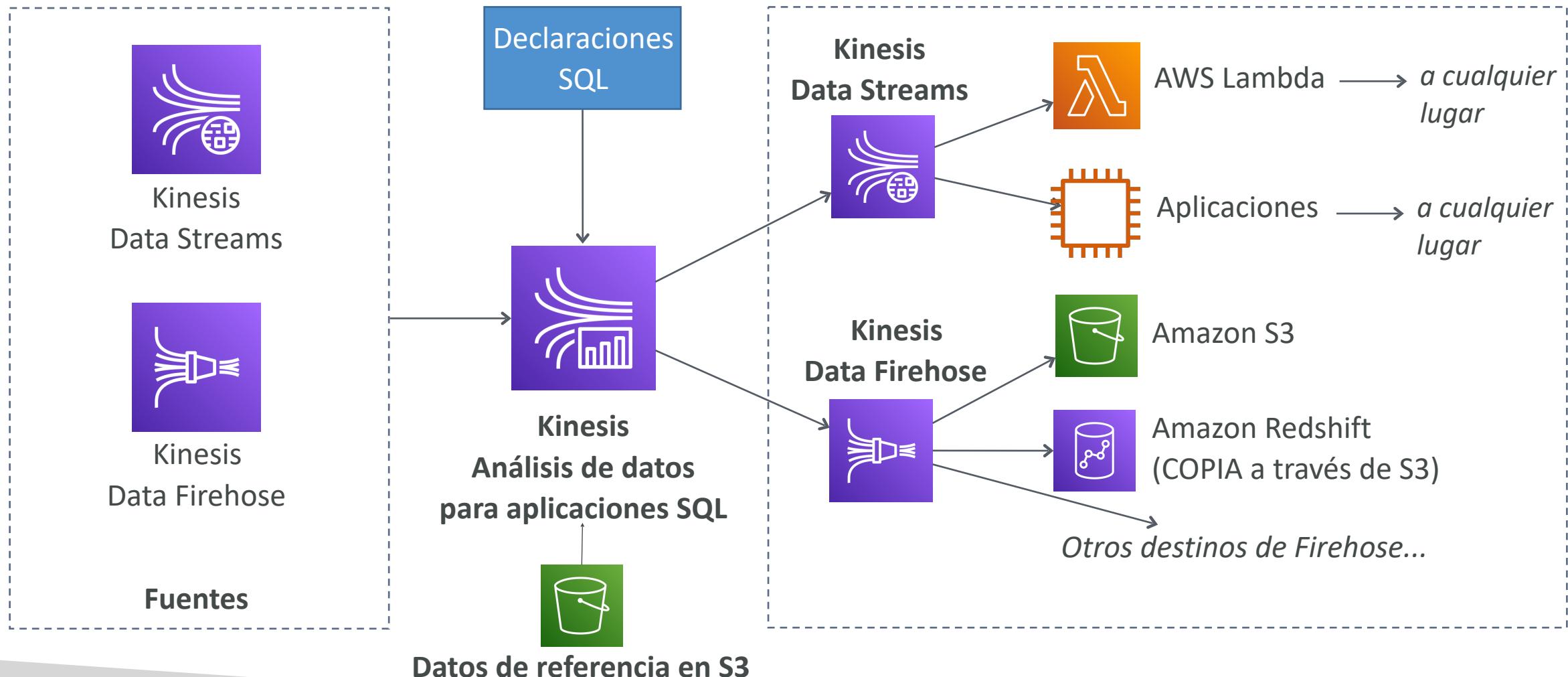
- Servicio de streaming para la ingesta a escala
- Escribir código personalizado (productor / consumidor)
- En tiempo real (~200 ms)
- Gestión del escalado (división/fusión de fragmentos)
- Almacenamiento de datos de 1 a 365 días
- Capacidad de reproducción



Kinesis Data Firehose

- Carga de datos de streaming en S3 / Redshift / ES / terceros / HTTP personalizado
- Totalmente gestionado
- Casi en tiempo real (tiempo de buffer min. 60 seg)
- Escalado automático
- Sin almacenamiento de datos
- No soporta capacidad de repetición

Kinesis Data Analytics para aplicaciones SQL



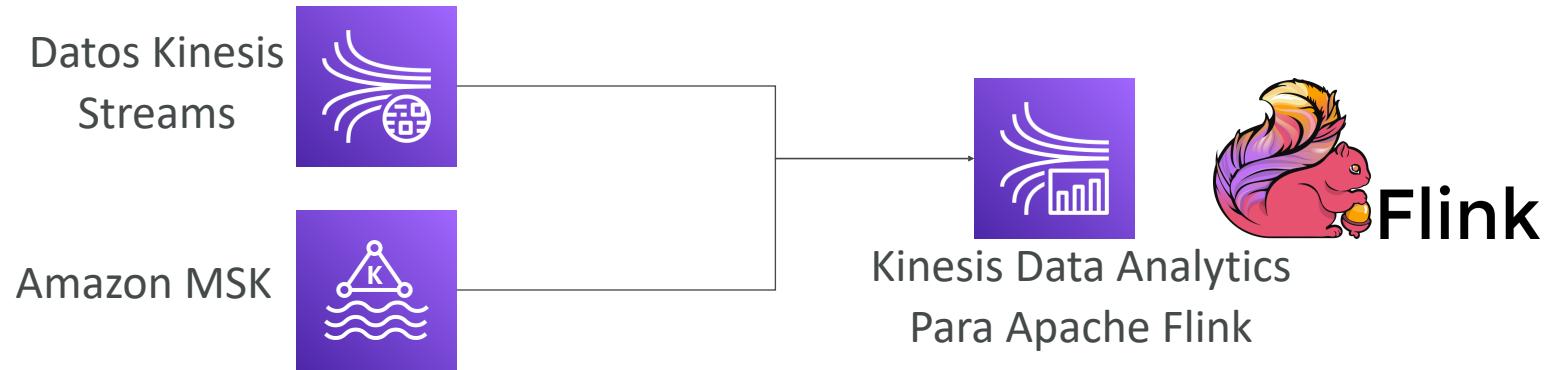


Kinesis Data Analytics (aplicación SQL)

- Análisis en tiempo real en **Kinesis Data Streams & Firehose** mediante SQL
- Añadir datos de referencia de Amazon S3 para enriquecer los datos de streaming
- Totalmente administrado, sin servidores que aprovisionar
- Escalado automático
- Paga por la tasa de consumo real
- Salida:
 - Kinesis Data Streams: crea flujos a partir de las consultas analíticas en tiempo real
 - Kinesis Data Firehose: envío de resultados de consultas analíticas a destinos
- Casos de uso:
 - Análisis de series temporales
 - Dashboards en tiempo real
 - Métricas en tiempo real

Kinesis Data Analytics para Apache Flink

- Utilización de Flink (Java, Scala o SQL) para procesar y analizar datos en streaming

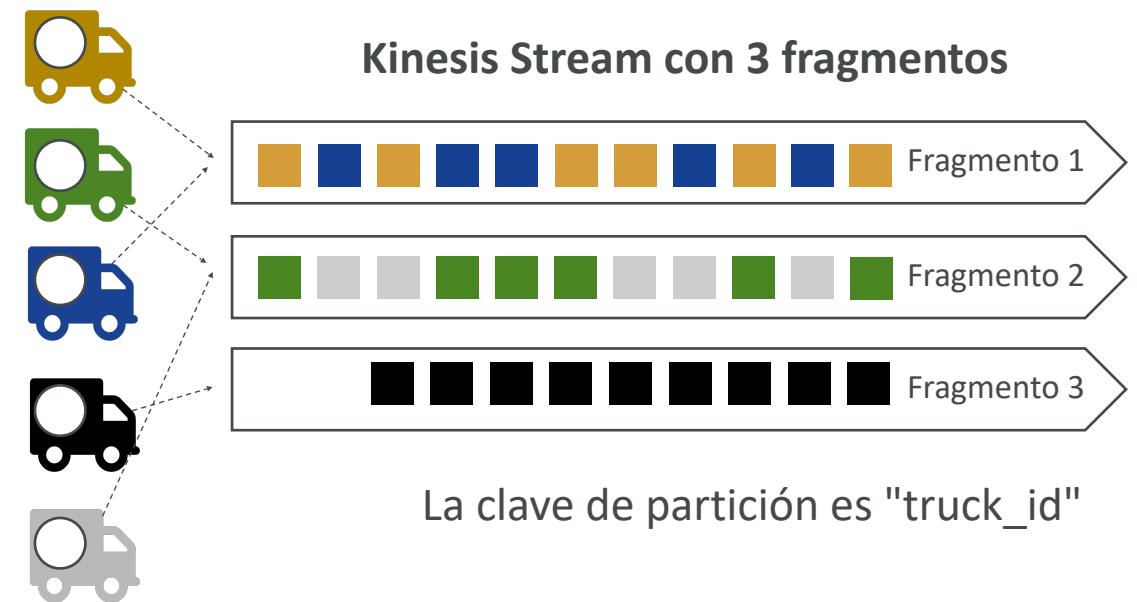


- Ejecuta cualquier aplicación Apache Flink en un clúster administrado en AWS
 - aprovisionamiento de recursos informáticos, computación paralela, escalado automático
 - backups de aplicaciones (implementados como puntos de control e instantáneas)
 - Utiliza cualquier característica de programación de Apache Flink
 - Flink no lee de Firehose (utiliza Kinesis Analytics para SQL en su lugar)

Ordenar datos en Kinesis

- Imagina que tienes 100 camiones (camión_1, camión_2, ... camión_100) en la carretera enviando sus posiciones GPS regularmente a AWS.
- Se desea consumir los datos en orden para cada camión, de modo que pueda seguir su movimiento con precisión.
- ¿Cómo debe enviar esos datos a Kinesis?

- **Respuesta:** enviar utilizando un valor de "Partition Key" del "truck_id" ("camion_id").
- **La misma clave irá siempre al mismo fragmento**

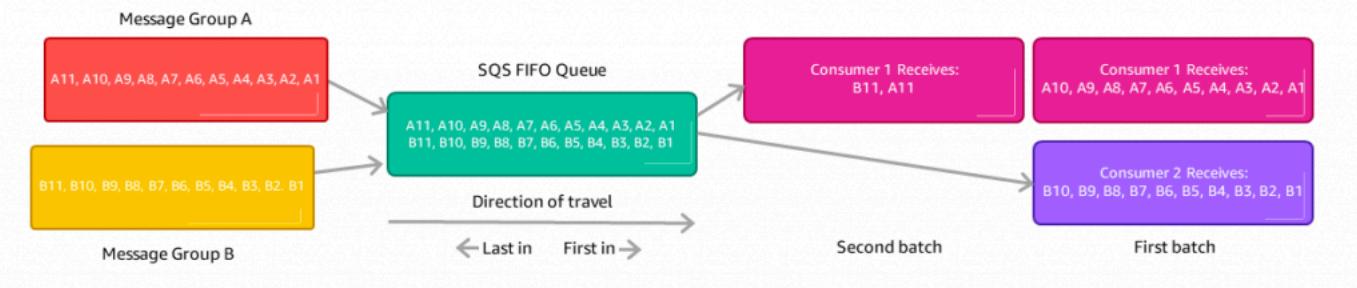


Ordenar datos en SQS

- Para SQS estándar, no hay ordenación.
- Para SQS FIFO, si no se utiliza un ID de grupo, los mensajes se consumen en el orden en que se envían, **con un solo consumidor**



- Deseas escalar el número de consumidores, pero quieres que los mensajes estén "agrupados" cuando se relacionan entre sí
- En ese caso, puedes utilizar un ID de grupo (similar a la clave de partición en Kinesis).



Ordenación Kinesis vs SQS

- **Supongamos 100 camiones, 5 fragmentos kinesis, 1 SQS FIFO**
- Kinesis Data Streams:
 - Por término medio, tendrás 20 camiones por fragmento.
 - Los camiones tendrán sus datos ordenados dentro de cada fragmento
 - La cantidad máxima de consumidores en paralelo que podemos tener es 5
 - Puede recibir hasta 5 MB/s de datos
- SQS FIFO
 - Sólo tienes una cola SQS FIFO
 - Tendrás 100 ID de grupo
 - Podrás tener hasta 100 Consumidores (debido al 100 Group ID)
 - Tendrás hasta 300 mensajes por segundo (o 3000 si utilizamos batching)

SQS vs SNS vs Kinesis

SQS:

- Los consumidores "tiran de los datos"
- Los datos se borran después de ser consumidos
- Podemos tener tantos trabajadores (consumidores) como queramos
- No es necesario aprovisionar rendimiento
- Garantías de ordenación sólo en colas FIFO
- Capacidad de retardo de mensajes individuales



SNS:

- Envío de datos a muchos suscriptores
- Hasta 12.500.000 suscriptores
- Los datos no se conservan (se pierden si no se entregan)
- Pub/Sub
- Hasta 100.000 temas (topics)
- No es necesario aprovisionar caudal
- Se integra con SQS para un patrón de arquitectura en fan-out
- Capacidad FIFO para SQS FIFO



Kinesis:

- Estándar: extraer datos
 - 2 MB por fragmento
- Fan-out reforzado: datos push
 - 2 MB por fragmento y consumidor
- Posibilidad de reproducir los datos
- Pensado para big data en tiempo real, análisis y ETL
- Ordenación a nivel de fragmento
- Los datos caducan a los X días
- Modo aprovisionado o modo de capacidad bajo demanda



AWS Lambda

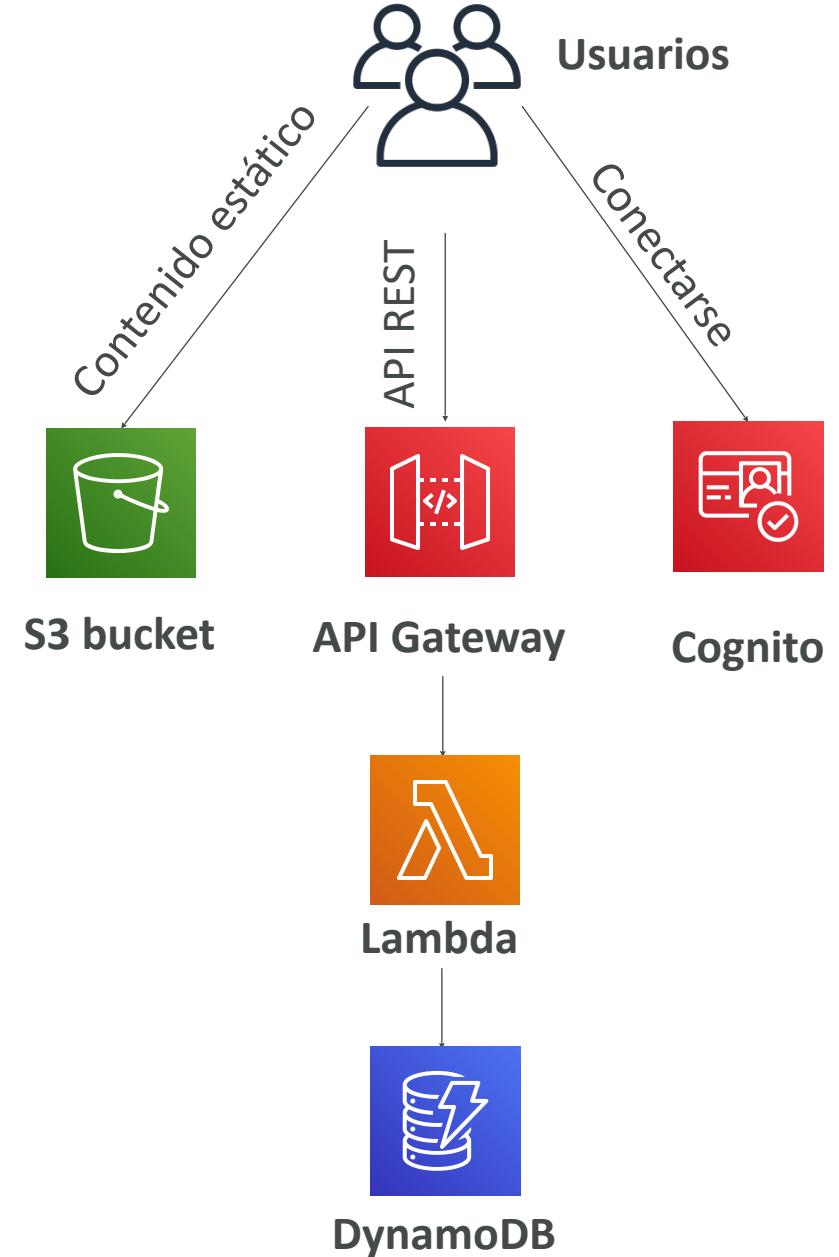
Un mundo sin servidores

¿Qué es serverless?

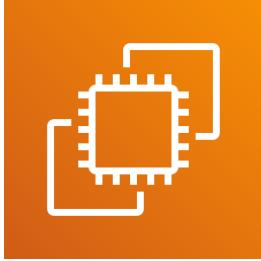
- Serverless es un nuevo paradigma en el que los desarrolladores ya no tienen que gestionar servidores...
- Sólo despliegan código
- Sólo despliegan... ¡funciones!
- Inicialmente... Serverless == FaaS (Función como Servicio)
- Serverless fue pionero en AWS Lambda pero ahora también incluye cualquier cosa que se gestione: "bases de datos, mensajería, almacenamiento, etc".
- **Serverless no significa que no haya servidores...** significa que simplemente no los gestionas / aprovisionas / ves

Sin servidor en AWS

- AWS Lambda
- DynamoDB
- AWS Cognito
- AWS API Gateway
- Amazon S3
- AWS SNS y SQS
- Kinesis Data Firehose
- Aurora Serverless
- Funciones por pasos (Step Functions)
- Fargate



Por qué AWS Lambda



Amazon EC2

- Servidores virtuales en la nube
- Limitado por RAM y CPU
- Funcionamiento continuo
- Escalado significa intervención para añadir/eliminar servidores



Amazon Lambda

- **Funciones** virtuales: sin servidores que gestionar
- Limitado por el tiempo - **ejecuciones cortas**
- Ejecución **bajo demanda**
- **Escalado automatizado**

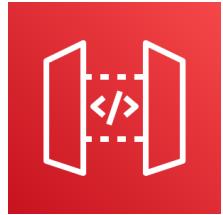
Beneficios de AWS Lambda

- Precios sencillos:
 - Pago por solicitud y tiempo de cómputo
 - Capa gratuita de 1.000.000 de solicitudes de AWS Lambda y 400.000 GB de tiempo de cómputo
- Integrado con todo el conjunto de servicios de AWS
- **Dirigido por eventos:** las funciones son invocadas por AWS cuando se necesitan
- Integrado con muchos lenguajes de programación
- Fácil monitorización a través de AWS CloudWatch
- Fácil de obtener más recursos por funciones (¡hasta 10 GB de RAM!)
- ¡El aumento de la RAM también mejorará la CPU y la red!

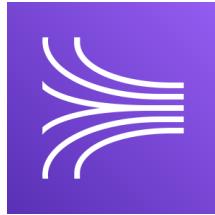
Soporte del lenguaje AWS Lambda

- Node.js (JavaScript)
- Python
- Java (compatible con Java 8)
- C# (.NET Core)
- Golang
- C# / Powershell
- Ruby
- API de tiempo de ejecución personalizado (compatible con la comunidad, ejemplo Rust)
- Imagen de contenedor Lambda
 - La imagen del contenedor debe implementar la API de tiempo de ejecución Lambda
 - Se prefiere ECS / Fargate para ejecutar imágenes Docker arbitrarias

Integraciones de AWS Lambda Principales



API Gateway



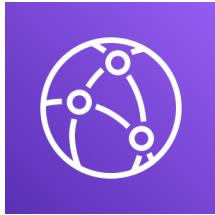
Kinesis



DynamoDB



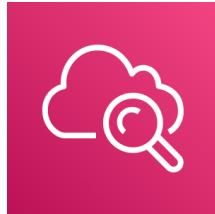
S3



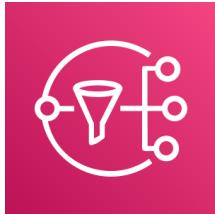
CloudFront



CloudWatch Events
EventBridge



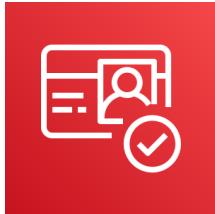
CloudWatch Logs



SNS



SQS



Cognito

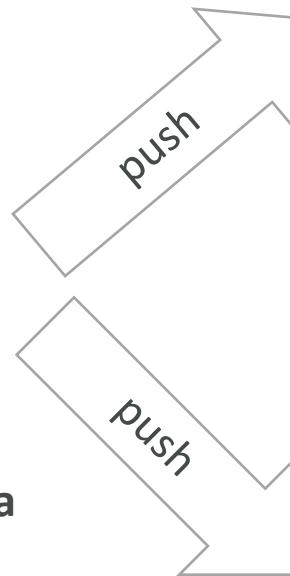
Ejemplo: Creación de miniaturas Serverless



Nueva imagen en S3



La función AWS Lambda
crea una miniatura



Nueva miniatura en S3



Nombre de la imagen
Tamaño de la imagen
Fecha de creación
etc.



Metadata en DynamoDB

Ejemplo: Trabajo CRON Serverless



CloudWatch Events
EventBridge



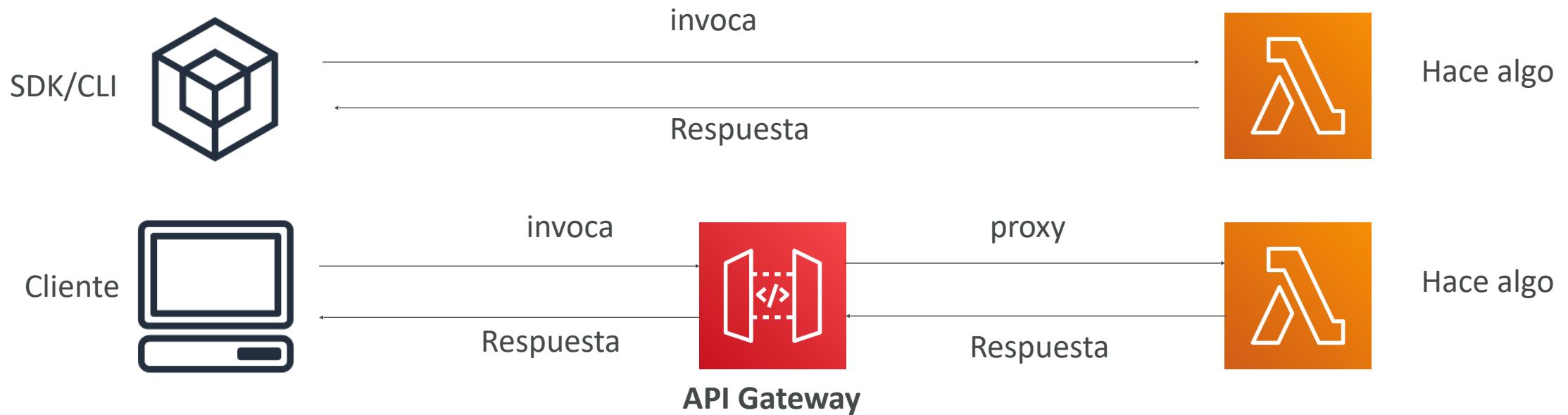
Función AWS Lambda
realiza una tarea

Precios de AWS Lambda: ejemplo

- Encontrará información general sobre precios en el enlace:
 - <https://aws.amazon.com/lambda/pricing/>
- Pago por **llamadas**:
 - Las primeras 1.000.000 de solicitudes son gratuitas
 - 0,20 \$ por cada millón de solicitudes (0,0000002 \$ por solicitud)
- Pago por **duración**: (en incrementos de 1 ms)
 - 400.000 GB-segundos de tiempo de cálculo al mes GRATIS
 - == 400.000 segundos si la función es de 1GB RAM
 - == 3.200.000 segundos si la función es de 128 MB RAM
 - Después, 1 dólar por 600.000 GB-segundos
- Suele ser muy barato ejecutar AWS Lambda, por lo que es muy popular

Lambda - Invocaciones síncrona / sincrónicas

- Sincrónico: CLI, SDK, API Gateway, Application Load Balancer
 - Los resultados se devuelven inmediatamente
 - La gestión de errores debe realizarse en el lado del cliente (reintentos, retroceso exponencial, etc.)

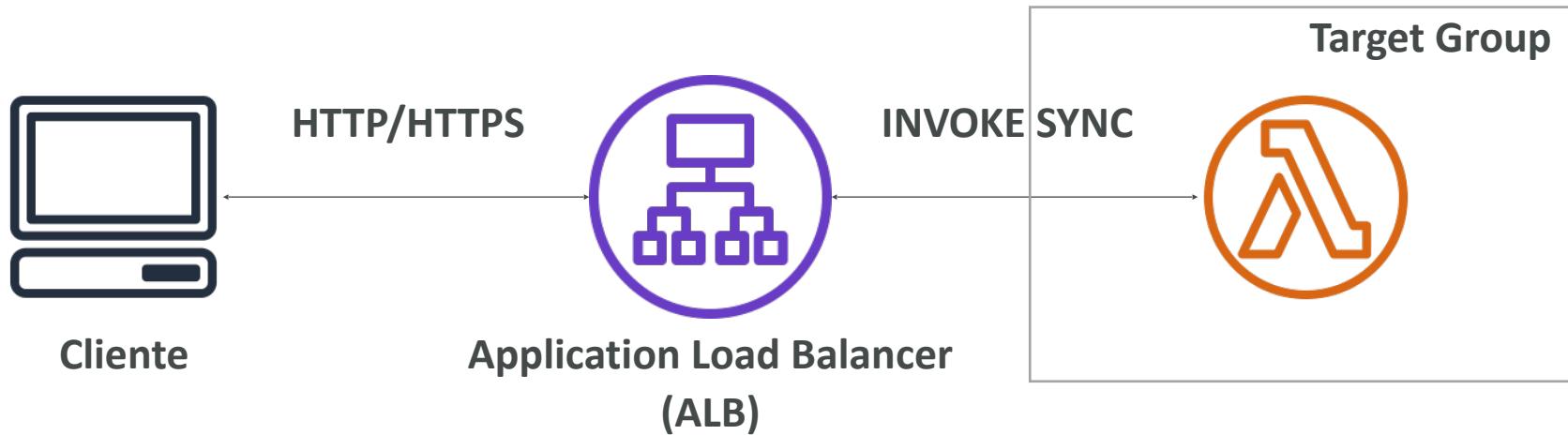


Lambda - Invocaciones sincrónicas - Servicios

- **Invocado por usuario:**
 - **Elastic Load Balancing (Application Load Balancer)**
 - **Amazon API Gateway**
 - **Amazon CloudFront (Lambda@Edge)**
 - Amazon S3 Batch
- **Invocado por servicio:**
 - **Amazon Cognito**
 - **AWS Step Functions**
- Otros servicios:
 - Amazon Lex
 - Amazon Alexa
 - Amazon Kinesis Data Firehose

Integración de Lambda con ALB

- Para exponer una función Lambda como un endpoint HTTP(S)...
- Puedes utilizar el **Application Load Balancer** (o un API Gateway)
- La función Lambda debe estar registrada en un **grupo de destino (target group)**



De ALB a Lambda: De HTTP a JSON

Carga útil (Payload) de la petición para la función lambda

```
{  
  "requestContext": {  
    "elb": {  
      "targetGroupArn": "arn:aws:elasticloadbalancing:us-east-2:12345678901234567890:targetgroup/49e9d65c45c6791a"  
    }  
  },  
  "httpMethod": "GET",  
  "path": "/lambda",  
  "queryStringParameters": {  
    "query": "1234ABCD"  
  },  
  "headers": {  
    "connection": "keep-alive",  
    "host": "lambda-alb-123578498.us-east-2.elb.amazonaws.com",  
    "user-agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/64.0.3282.140 Safari/537.36",  
    "x-amzn-trace-id": "Root=1-5c536348-3d683b8b04734faae651f476",  
    "x-forwarded-for": "72.12.164.125",  
    "x-forwarded-port": "80",  
    "x-forwarded-proto": "http",  
  },  
  "body": "",  
  "isBase64Encoded": false  
}
```

Información sobre el ELB

Método y ruta HTTP

Parámetros de cadena de consulta
como pares clave/valor

Cabeceras como pares Clave/Valor

Cuerpo (para POST, PUT...) & isBase64Encoded

Conversiones de Lambda a ALB: JSON a HTTP

Respuesta de la función lambda

```
{  
  "statusCode": 200,  
  "statusDescription": "200 OK",  
  "headers": {  
    "Content-Type": "text/html; charset=utf-8"  
  },  
  "body": "<h1>Hello world!</h1>",  
  "isBase64Encoded": false  
}
```

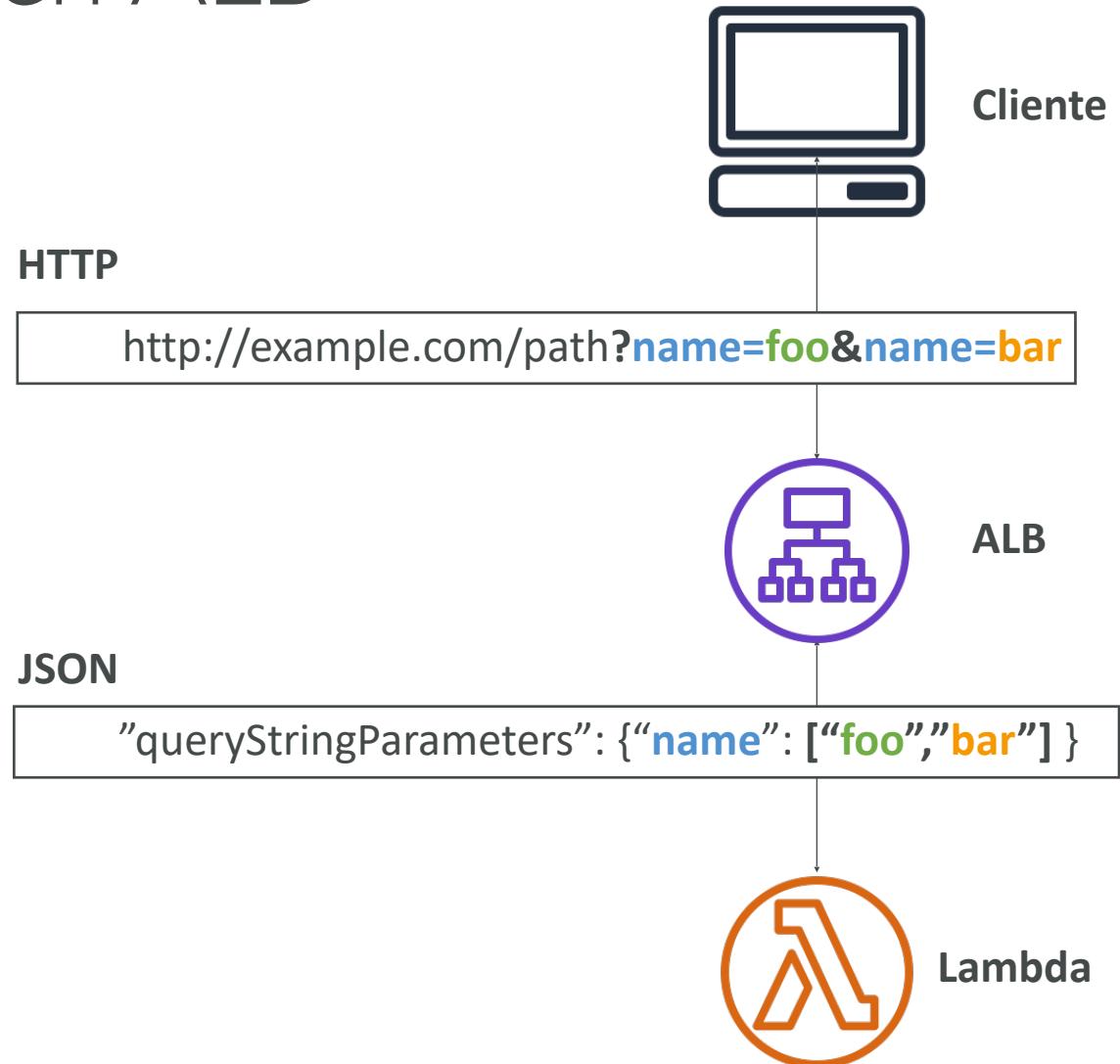
Código de estado y descripción

Cabeceras como pares Clave/Valor

Cuerpo & isBase64Encoded

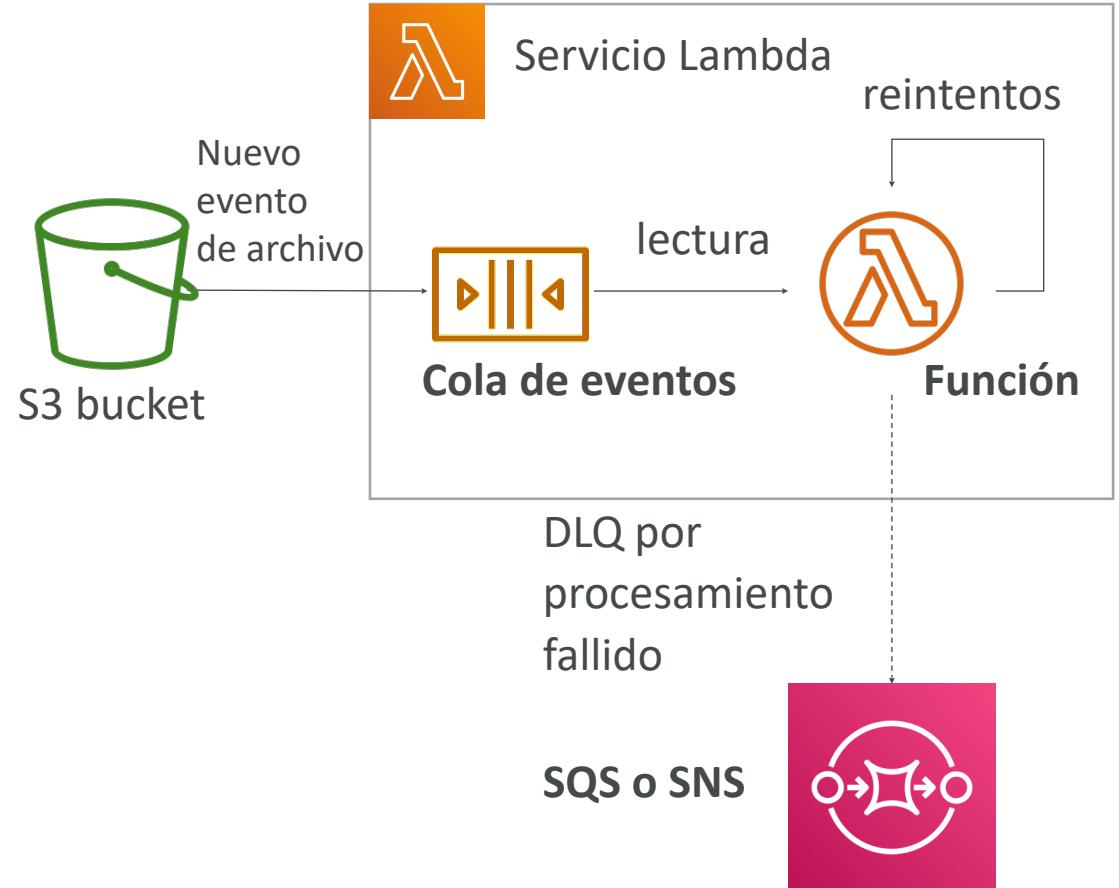
Encabezados multivalores en ALB

- ALB puede soportar multivalores de cabecera (configuración de ALB)
- Cuando habilitas los encabezados multivalores, los **encabezados HTTP y los parámetros de cadena de consulta** que se envían con **varios valores** se muestran **como arrays** dentro de los objetos de evento y respuesta de AWS Lambda.



Lambda - Invocaciones asíncronas

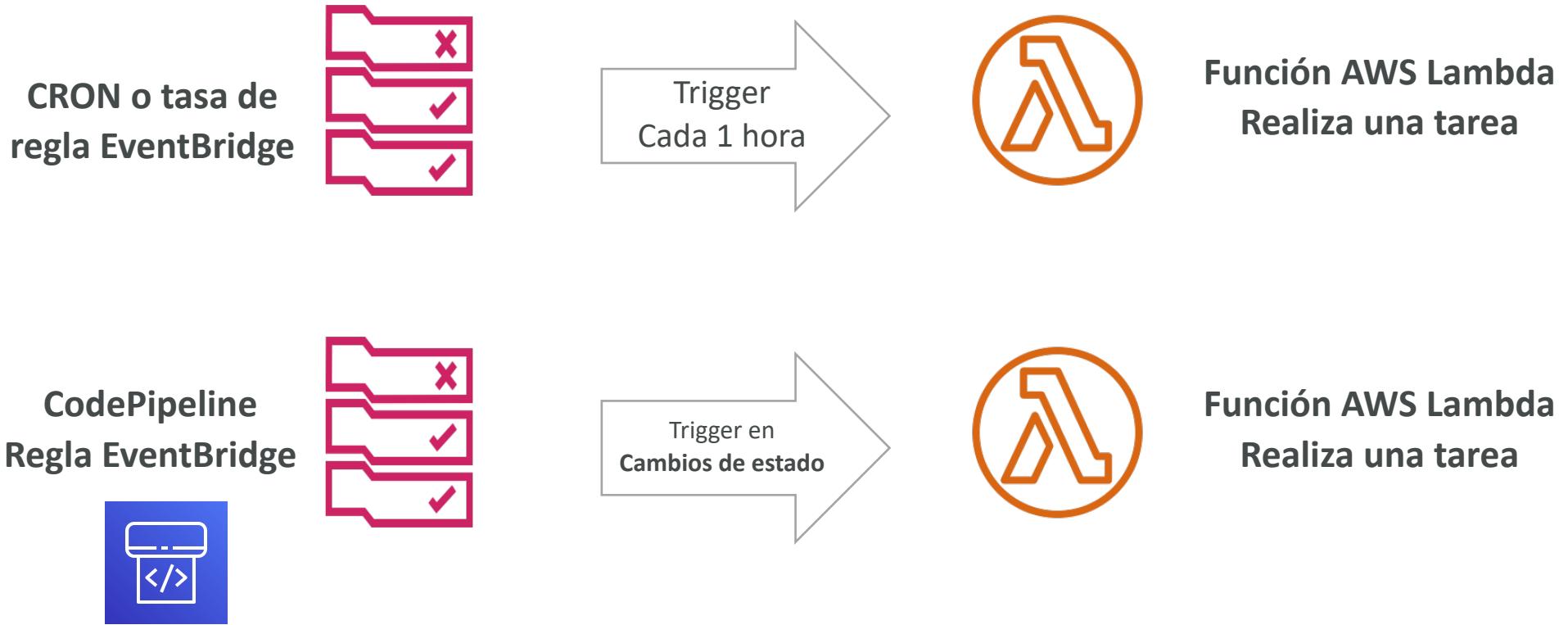
- S3, SNS, CloudWatch Events...
- Los eventos se colocan en una **cola de eventos**
- Lambda intenta reintentar en caso de error
 - 3 intentos en total
 - 1 minuto de espera después del primero, luego 2 minutos de espera
- Asegúrate de que el procesamiento es **idempotente** (en caso de reintentos)
- Si se reintenta la función, **verás entradas de logs duplicadas en CloudWatch Logs**
- Puedes definir una DLQ (cola de mensajes fallidos)
 - **SNS o SQS** - para el procesamiento fallido (necesitas permisos IAM correctos)
- Las invocaciones asíncronas te permiten acelerar el procesamiento si no necesitas esperar el resultado (ej: necesitas 1000 archivos procesados)



Lambda - Invocaciones asíncronas - Servicios

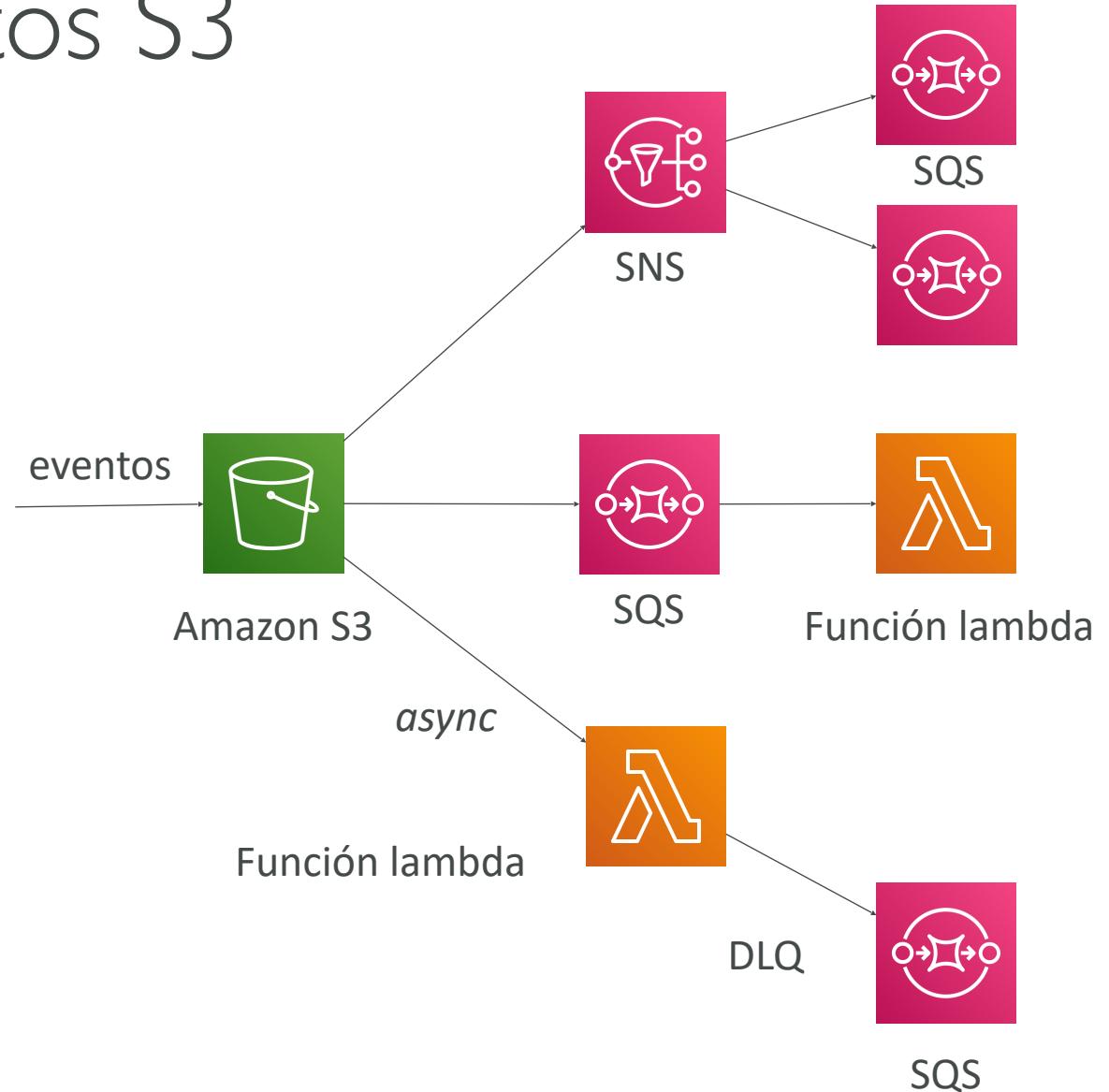
- **Amazon Simple Storage Service (S3)**
- **Amazon Simple Notification Service (SNS)**
- **Amazon CloudWatch Events / EventBridge**
- AWS CodeCommit (CodeCommit Trigger: nueva rama, nueva etiqueta, nuevo push)
- AWS CodePipeline (invocar una función Lambda durante la canalización, Lambda debe hacer callback)
----- otros -----
 - Amazon CloudWatch Logs (procesamiento de logs)
 - Amazon Simple Email Service
 - AWS CloudFormation
 - AWS Config
 - AWS IoT
 - AWS IoT Events

CloudWatch Events / EventBridge



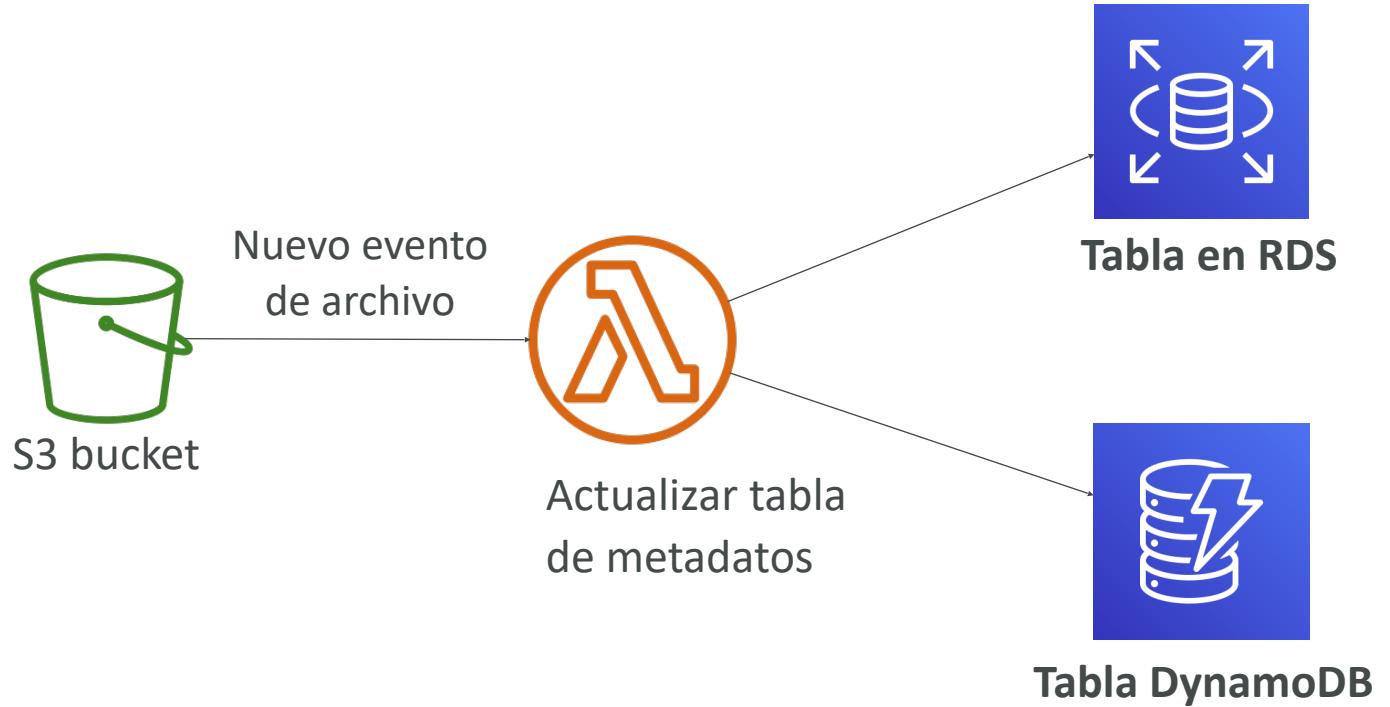
Notificaciones de eventos S3

- S3:ObjectCreated, S3:ObjectRemoved, S3:ObjectRestore, S3:Replication...
- Posibilidad de filtrar el nombre del objeto (*.jpg)
- Caso de uso: generar miniaturas de imágenes subidas a S3
- Las notificaciones de eventos de S3 suelen entregar los eventos en segundos, pero a veces pueden tardar un minuto o más
- Si se realizan dos escrituras en un mismo objeto no versionado al mismo tiempo, es posible que sólo se envíe una única notificación de evento
- Si quieres asegurarte de que se envía una notificación de evento por cada escritura correcta, puedes activar el versionado en tu bucket.



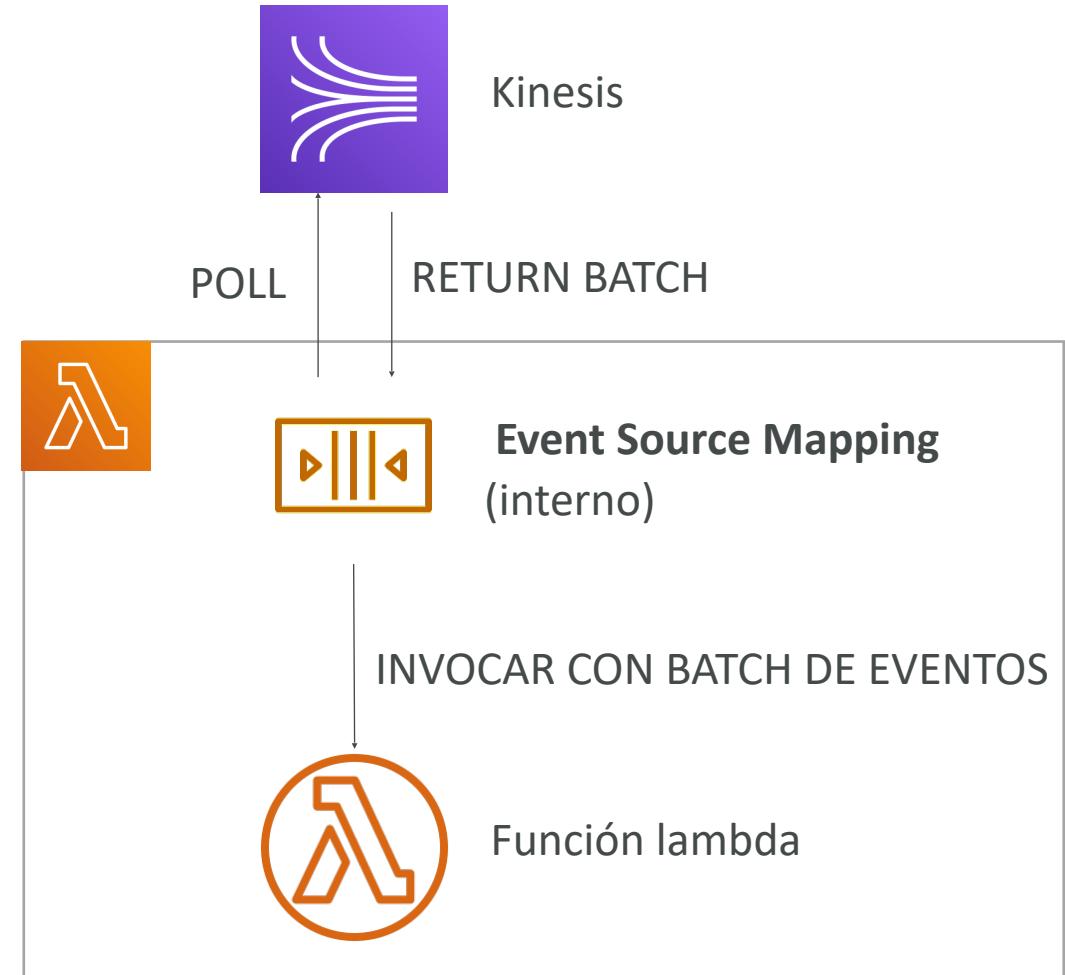
Patrón de evento S3 simple

Sincronización de metadatos



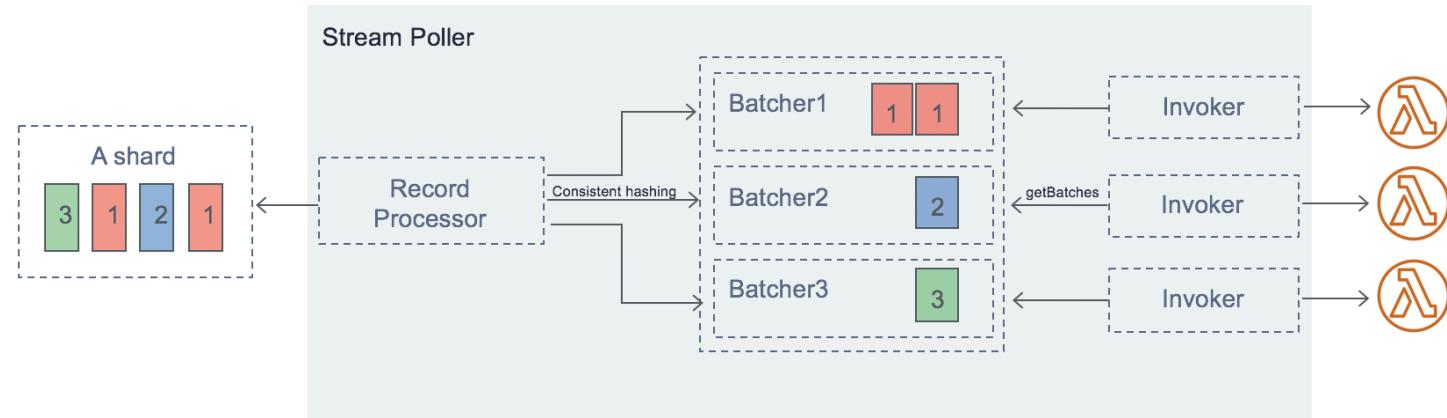
Lambda - Mapeo de fuentes de eventos

- Kinesis Data Streams
 - Cola SQS y SQS FIFO
 - DynamoDB Streams
-
- Denominador común: hay que sondar los registros desde el origen
 - **Tu función Lambda se invoca de forma sincrónica**



Streams y Lambda (Kinesis y DynamoDB)

- Un mapeo de origen de eventos crea un iterador para cada fragmento, procesa los elementos en orden
- Comienza con nuevos elementos, desde el principio o desde la marca de tiempo
- Los elementos procesados no se eliminan del stream (otros consumidores pueden leerlos)
- Poco tráfico: utiliza la ventana de lotes para acumular registros antes de procesarlos
- Puedes procesar varios lotes en paralelo
 - Hasta 10 lotes por fragmento
 - El procesamiento en orden sigue estando garantizado para cada clave de partición



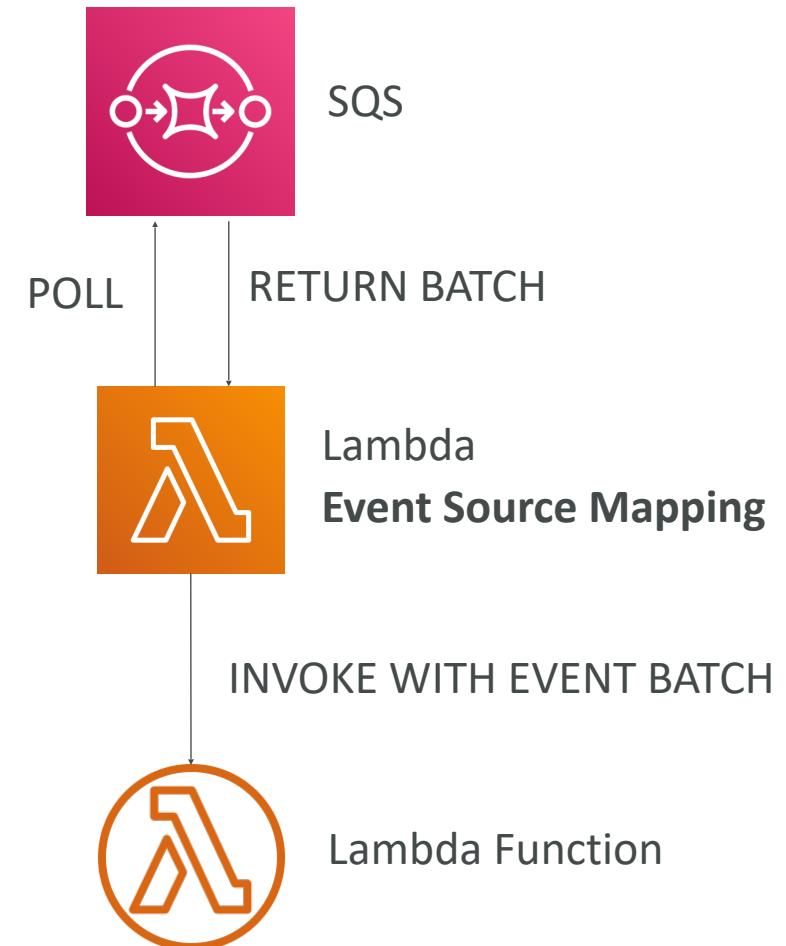
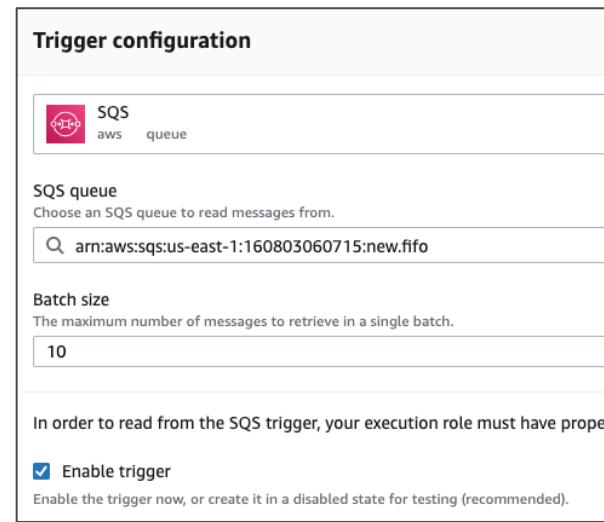
<https://aws.amazon.com/blogs/compute/new-aws-lambda-scaling-controls-for-kinesis-and-dynamodb-event-sources/>

Streams y Lambda - Tratamiento de errores

- **Por defecto, si tu función devuelve un error, todo el lote se vuelve a procesar hasta que la función tenga éxito, o los elementos del lote caduquen.**
- Para garantizar un procesamiento en orden, el procesamiento del fragmento afectado se pausa hasta que se resuelva el error.
- Puedes configurar el mapeo de la fuente de eventos para que
 - descartar eventos antiguos
 - restringir el número de reintentos
 - dividir el lote en caso de error (para solucionar problemas de tiempo de espera de Lambda)
- Los eventos descartados pueden ir a un **destino**

Lambda – Event Source Mapping SQS y SQS FIFO

- El mapeo de fuentes de eventos sondeará SQS (**sondeo largo**)
- Especifica el **tamaño del lote/batch** (1-10 mensajes)
- Recomendado: Establece el tiempo de espera de visibilidad de la cola en 6 veces el tiempo de espera de tu función Lambda
- **Para utilizar una DLQ**
 - **Configúralo en la cola SQS, no en Lambda** (DLQ para Lambda es sólo para invocaciones asíncronas)
 - O utilizar un destino Lambda para los fallos



Colas y Lambda

- Lambda también soporta el procesamiento en orden para colas FIFO (primero en entrar, primero en salir), **escalando hasta el número de grupos de mensajes activos.**
- En las colas estándar, los elementos no se procesan necesariamente en orden.
- Lambda escala para procesar una cola estándar lo más rápidamente posible.

- Cuando se produce un error, los lotes se devuelven a la cola como elementos individuales y podrían procesarse en una agrupación diferente a la del lote original.
- Ocasionalmente, el mapeo de la fuente de eventos podría recibir el mismo elemento de la cola dos veces, aunque no se haya producido ningún error de función.
- Lambda elimina los elementos de la cola después de que se hayan procesado correctamente.
- Puedes configurar la cola de origen para que envíe los ítems a una cola de mensajes fallidos si no se pueden procesar.

Escalado del Lambda Event Mapper

- **Kinesis Data Streams y DynamoDB Streams:**

- Una invocación Lambda por fragmento de stream
- Si utilizas la paralelización, hasta 10 lotes procesados por shard simultáneamente

- **SQS Standard:**

- Lambda añade 60 instancias más por minuto para escalar
- Hasta 1000 lotes de mensajes procesados simultáneamente

- **SQS FIFO:**

- Los mensajes con el mismo GroupID se procesarán en orden
- La función Lambda se adapta al número de grupos de mensajes activos

Lambda - Objetos de evento y contexto



Lambda - Objetos de evento y contexto

- **Objeto de evento**

- Documento con formato JSON que contiene datos para que la función los procese
- Contiene información del servicio invocador (por ejemplo, EventBridge, personalizado, ...)
- El tiempo de ejecución de Lambda convierte el evento en un objeto (por ejemplo, de tipo dict en Python)
- Ejemplo: argumentos de entrada, argumentos del servicio invocador, ...

- **Objeto de contexto**

- Proporciona métodos y propiedades que ofrecen información sobre la invocación, la función y el entorno de ejecución
- Pasados a tu función por Lambda en tiempo de ejecución
- Ejemplo: aws_request_id, function_name, memory_limit_in_mb, ...

Lambda - Objetos de evento y contexto

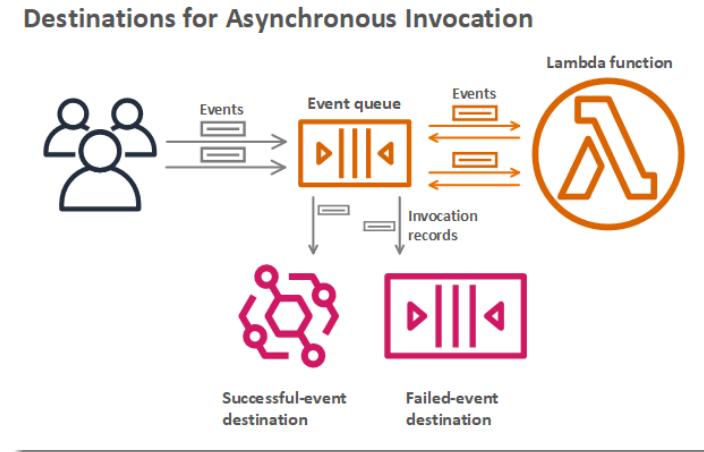
Acceder a Objetos de evento y contexto con Python

```
def lambda_handler(event, context):
    print("Event Source:", event.source)
    print("Event Region:", event.region)

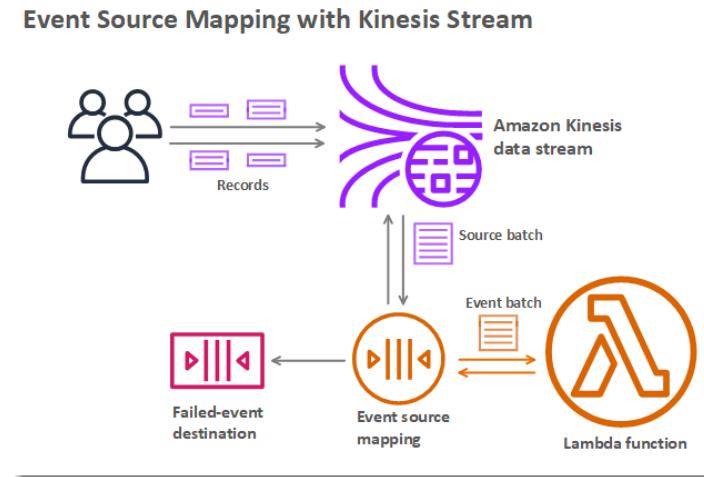
    print("Lambda Request ID:", context.aws_request_id)
    print("Lambda function ARN:", context.function_name)
    print("Lambda function ARN:", context.invoked_function_arn)
    print("Lambda function memory limits in MB:", context.memory_limit_in_mb)
    print("CloudWatch log stream name:", context.log_stream_name)
    print("CloudWatch log group name:", context.log_group_name)
```

Lambda - Destinos

- Nov 2019: Puedes configurar el envío del resultado a un **destino**
- **Invocaciones asíncronas**: se pueden definir destinos para eventos exitosos y fallidos:
 - Amazon SQS
 - Amazon SNS
 - AWS Lambda
 - Bus de Amazon EventBridge
- Nota: AWS recomienda utilizar destinos en lugar de DLQ ahora (pero ambos pueden utilizarse al mismo tiempo)
- **Event Source mapping (Mapping de fuente de eventos)**: para lotes de eventos descartados
 - Amazon SQS
 - Amazon SNS
- Nota: puedes enviar eventos a un DLQ directamente desde SQS



<https://docs.aws.amazon.com/lambda/latest/dg/invocation-async.html>



<https://docs.aws.amazon.com/lambda/latest/dg/invocation-eventsourcemapping.html>

Rol de ejecución Lambda (Rol IAM)



- Concede a la función Lambda permisos para los servicios / recursos de AWS
- Ejemplos de políticas gestionadas para Lambda:
 - AWSLambdaBasicExecutionRole - Subir logs a CloudWatch
 - AWSLambdaKinesisExecutionRole - Leer de Kinesis
 - AWSLambdaDynamoDBExecutionRole - Leer desde DynamoDB Streams
 - AWSLambdaSQSQueueExecutionRole - Lectura desde SQS
 - AWSLambdaVPCAccessExecutionRole - Implementar función Lambda en VPC
 - AWSXRayDaemonWriteAccess - Cargar datos de rastreo en X-Ray.
- **Cuando utilizas un mapping de fuente de eventos (Event Source Mapping) para invocar tu función, Lambda utiliza el rol de ejecución para leer los datos de eventos.**
- Mejor práctica: crea un rol de ejecución de Lambda por función

Políticas basadas en recursos Lambda

- Utiliza políticas basadas en recursos para dar permiso a otras cuentas y servicios de AWS para utilizar tus recursos Lambda
- Similar a las políticas de bucket S3 para bucket S3
- Un principal IAM puede acceder a Lambda
 - si la política IAM adjunta al principal lo autoriza (por ejemplo, acceso de usuario)
 - O si la política basada en recursos lo autoriza (por ejemplo, acceso a servicios)
- Cuando un servicio de AWS como Amazon S3 llama a tu función Lambda, la política basada en recursos le da acceso.

Variables de entorno Lambda

- Variable de entorno = par clave / valor en forma de “Strings”
- Ajusta el comportamiento de la función sin actualizar el código
- Las variables de entorno están disponibles para tu código
- El servicio Lambda también añade sus propias variables de entorno del sistema
- Útil para almacenar secretos (cifrados por KMS)
- Los secretos pueden estar cifrados por la clave del servicio Lambda, o por tu propia CMK

Logs y monitorización de Lambda

- Logs de CloudWatch:
 - Los logs de ejecución de AWS Lambda se almacenan en AWS CloudWatch Logs
 - **Asegúrate de que tu función de AWS Lambda tiene un rol de ejecución con una política IAM que autoriza las escrituras en CloudWatch Logs**
- Métricas de CloudWatch:
 - Las métricas de AWS Lambda se muestran en AWS CloudWatch Metrics
 - Invocaciones, duraciones, ejecuciones concurrentes
 - Recuento de errores, tasas de éxito
 - Fallos de entrega asíncrona

Rastreo Lambda con X-Ray

- Activar en la configuración de Lambda (**Rastreo activo**)
- Ejecuta el demonio X-Ray por ti
- Utiliza el SDK de AWS X-Ray en el código
- Asegúrate de que la función Lambda tiene un rol de ejecución IAM correcto
 - La política administrada se llama AWSXRayDaemonWriteAccess
- Variables de entorno para comunicarse con X-Ray
 - **_X_AMZN_TRACE_ID**: contiene la cabecera de rastreo
 - **AWS_XRAY_CONTEXT_MISSING**: por defecto, LOG_ERROR
 - **AWS_XRAY_DAEMON_ADDRESS**: la IP_ADDRESS:PORT del demonio de X-Ray



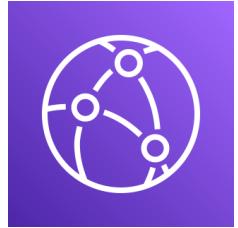
Personalización en el Edge

- Muchas aplicaciones modernas ejecutan alguna forma de la lógica en el borde
- **Función de borde (Edge Function):**
 - Un código que escribes y adjuntas a las distribuciones de CloudFront
 - Se ejecuta cerca de tus usuarios para minimizar la latencia
- CloudFront proporciona dos tipos: **Funciones CloudFront y Lambda@Edge**
- No tienes que gestionar ningún servidor; desplegado globalmente

- Caso de uso: personalizar el contenido de la CDN
- Paga sólo por lo que utilices
- Totalmente sin servidor

Funciones CloudFront y Lambda@Edge

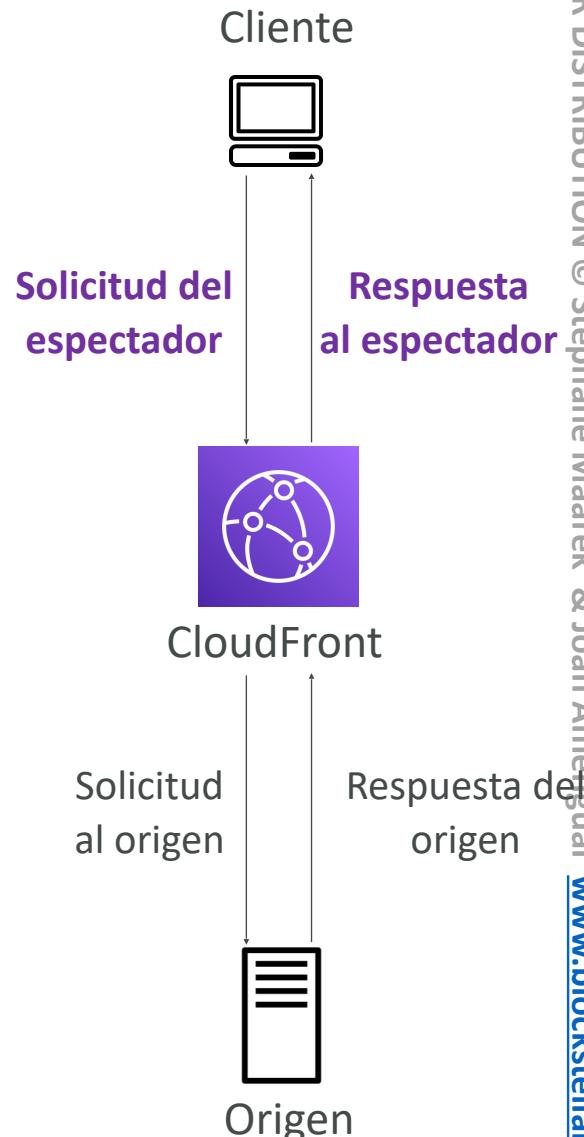
Casos de uso



- Seguridad y privacidad del sitio web
- Aplicación Web Dinámica en el Edge
- Optimización para motores de búsqueda (SEO)
- Enrutamiento Inteligente entre Orígenes y Centros de Datos
- Mitigación de bot en el Edge
- Transformación de imágenes en tiempo real
- Test A/B
- Autenticación y autorización de usuarios
- Priorización de usuarios
- Seguimiento y análisis de usuarios

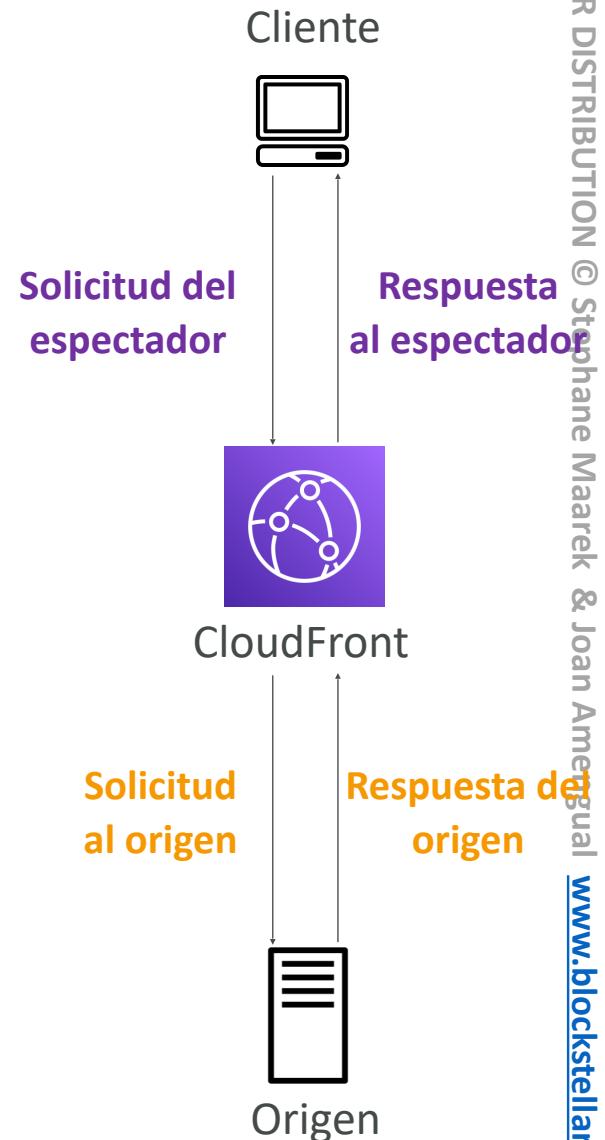
Funciones de CloudFront

- Funciones ligeras escritas en JavaScript
- Para personalizaciones de CDN a gran escala y sensibles a la latencia
- Tiempos de arranque inferiores a milisegundos, **millones de solicitudes/segundo**
- Se utiliza para modificar las solicitudes y respuestas de los espectadores:
 - **Solicitud del espectador:** después de que CloudFront reciba una solicitud de un espectador
 - **Respuesta al espectador:** antes de que CloudFront envíe la respuesta al espectador
- Característica nativa de CloudFront (gestiona el código completamente dentro de CloudFront)



Lambda@Edge

- Funciones lambda escritas en NodeJS o Python
- Escala a **1000s de peticiones/segundo**
- Se utiliza para modificar las solicitudes y respuestas de CloudFront:
 - **Solicitud del espectador**: después de que CloudFront reciba una solicitud de un espectador.
 - **Solicitud al origen**: antes de que CloudFront reenvíe la solicitud al origen.
 - **Respuesta del origen**: después de que CloudFront reciba la respuesta del origen
 - **Respuesta al espectador**: antes de que CloudFront reenvíe la respuesta al espectador.
- Crea tus funciones en una región de AWS (us-east-1), luego CloudFront replica a tus ubicaciones



Funciones de CloudFront vs Lambda@Edge

	Funciones CloudFront	Lambda@Edge
Soporte en tiempo de ejecución	JavaScript	Node.js, Python
Número de solicitudes	Millones de solicitudes por segundo	Miles de solicitudes por segundo
Triggers de CloudFront	- Solicitud/Respuesta del espectador	- Solicitud/Respuesta del espectador - Origen Solicitud/Respuesta
Máx. Tiempo de ejecución	< 1 ms	5 – 10 segundos
Max. Memoria	2 MB	128 MB hasta 10 GB
Tamaño total del paquete	10 KB	1 MB – 50 MB
Acceso a la red, acceso al sistema de archivos	No	Si
Acceso al cuerpo de la solicitud	No	Si
Precios	Nivel gratuito disponible, 1/6 del precio de @Edge	No hay nivel gratuito, se cobra por solicitud y duración

Funciones de CloudFront vs Lambda@Edge

Casos de uso

Funciones de CloudFront

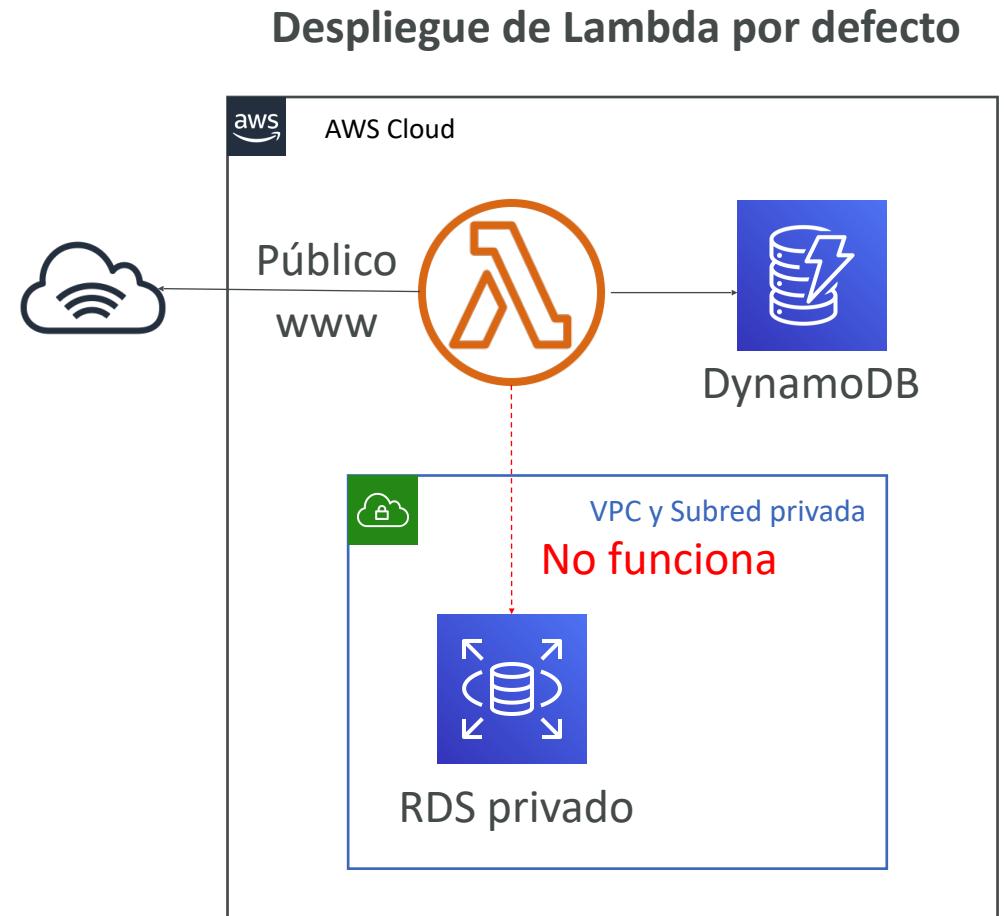
- Normalización de la clave de caché
 - Transformación de los atributos de la solicitud (cabeceras, cookies, cadenas de consulta, URL) para crear una clave de caché óptima
- Manipulación de cabeceras
 - Inserción/modificación/eliminación de cabeceras HTTP en la solicitud o la respuesta
- Reescritura o redireccionamiento de URL
- Autenticación y autorización de solicitudes
 - Creación y validación de tokens generados por el usuario (por ejemplo, JWT) para permitir/denegar solicitudes

Lambda@Edge

- Mayor tiempo de ejecución (varios ms)
- CPU o memoria ajustables
- El código depende de una tercera biblioteca (p. ej., AWS SDK para acceder a otros servicios de AWS)
- Acceso a la red para utilizar servicios externos para el procesamiento
- Acceso al sistema de archivos o acceso al cuerpo de las solicitudes HTTP

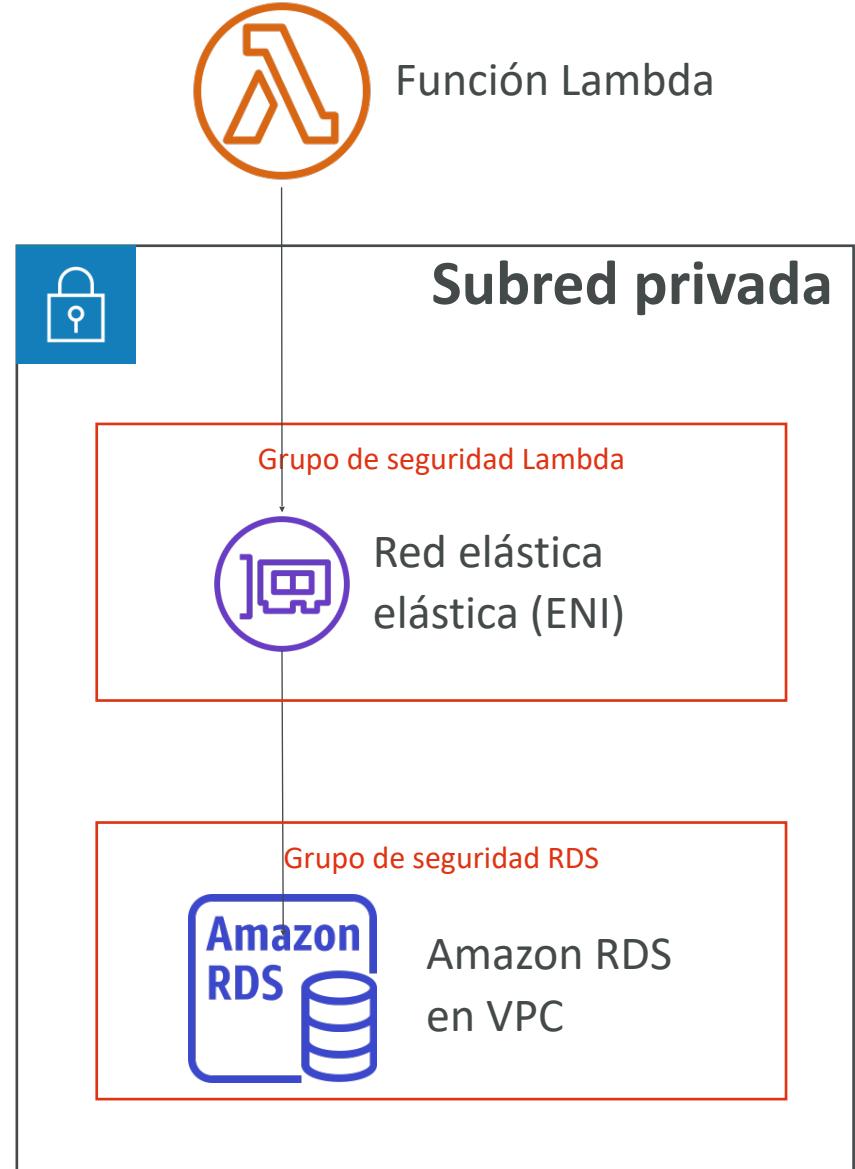
Lambda por defecto

- Por defecto, la función Lambda se lanza fuera de la propia VPC (en una VPC propiedad de AWS).
- Por lo tanto, no puedes acceder a los recursos de la VPC (RDS, ElastiCache, ELB interno...)



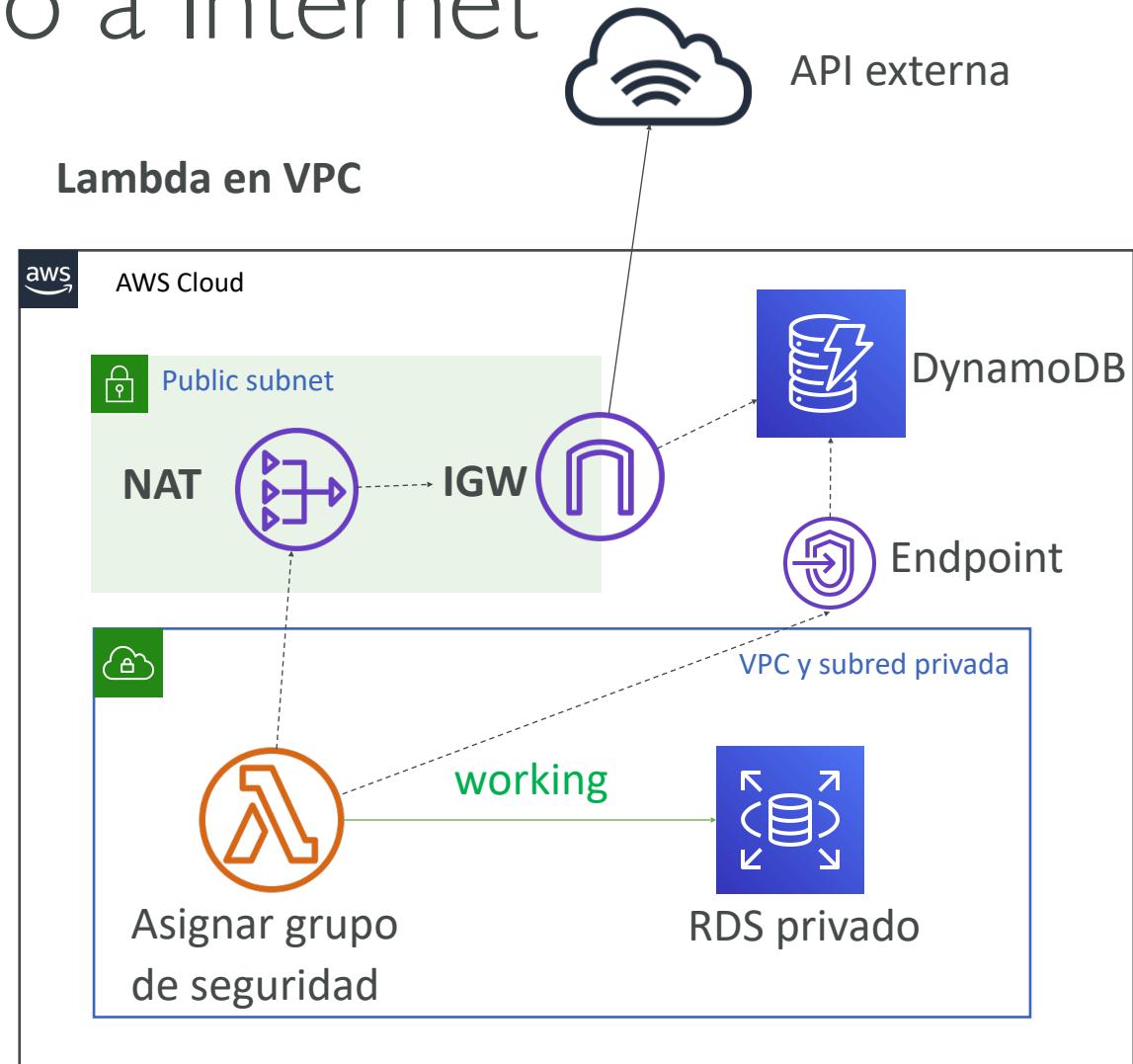
Lambda en VPC

- Debes definir el ID de la VPC, las subredes y los grupos de seguridad
- Lambda creará una ENI (Elastic Network Interface) en tus subredes
- **AWSLambdaVPCAccessExecutionRole**



Lambda en VPC - Acceso a Internet

- Una función Lambda en tu VPC no tiene acceso a Internet
- **Desplegar una función Lambda en una subred pública no le da acceso a Internet ni una IP pública**
- Implementar una función Lambda en una subred privada le da acceso a Internet si tienes una **instancia / Gateway NAT**
- Puedes utilizar **endpoints de VPC** para acceder de forma privada a los servicios de AWS sin NAT



Nota: Lambda - CloudWatch Logs funciona incluso sin endpoint o Gateways NAT

Configuración de la función lambda

- **RAM:**
 - De 128MB a 10GB en incrementos de 1MB
 - Cuanta más RAM añadas, más créditos de vCPU obtendrás
 - A 1.792 MB, una función tiene el equivalente a una vCPU completa
 - Después de 1.792 MB, obtienes más de una CPU, y necesitas utilizar multihilo en tu código para beneficiarte de ello (hasta 6 vCPU)
- **Si tu aplicación está ligada a la CPU (computación pesada), aumenta la RAM**
- **Timeout:** por defecto 3 segundos, el máximo es 900 segundos (15 minutos)

Contexto de ejecución lambda

- El contexto de ejecución es un entorno temporal de tiempo de ejecución que inicializa cualquier dependencia externa de tu código lambda
- Ideal para conexiones a bases de datos, clientes HTTP, clientes SDK...
- El contexto de ejecución se mantiene durante algún tiempo en previsión de otra invocación a la función Lambda
- La siguiente invocación a la función puede "reutilizar" el contexto a tiempo de ejecución y ahorrar tiempo en la inicialización de objetos de conexión
- El contexto de ejecución incluye el directorio /tmp

Inicializar fuera del controlador

¡MAL!

```
import os

def get_user_handler(event, context):
    DB_URL = os.getenv("DB_URL")
    db_client = db.connect(DB_URL)
    user = db_client.get(user_id = event["user_id"])

    return user
```

Se establece la conexión con la BD
En cada invocación de función

¡BIEN!

```
import os

DB_URL = os.getenv("DB_URL")
db_client = db.connect(DB_URL)

def get_user_handler(event, context):
    user = db_client.get(user_id = event["user_id"])

    return user
```

La conexión a la BD se establece una vez
Y se reutiliza entre invocaciones

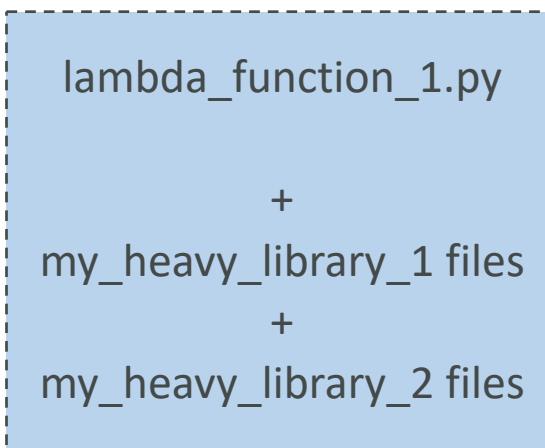
Funciones lambda en el espacio /tmp

- Si tu función Lambda necesita descargar un archivo grande para trabajar...
- Si tu función Lambda necesita espacio en disco para realizar operaciones...
- Puedes utilizar el directorio /tmp
- El tamaño máximo es de 10 GB
- El contenido del directorio permanece cuando se congela el contexto de ejecución, proporcionando una caché transitoria que puede utilizarse para múltiples invocaciones (útil para comprobar tu trabajo)
- Para la persistencia permanente del objeto (no temporal), utiliza S3

Capas de Lambda

- Tiempos de ejecución personalizados
 - Ej: C++ <https://github.com/awslabs/aws-lambda-cpp>
 - Ej: Rust <https://github.com/awslabs/aws-lambda-rust-runtime>
- Externaliza las dependencias para reutilizarlas:

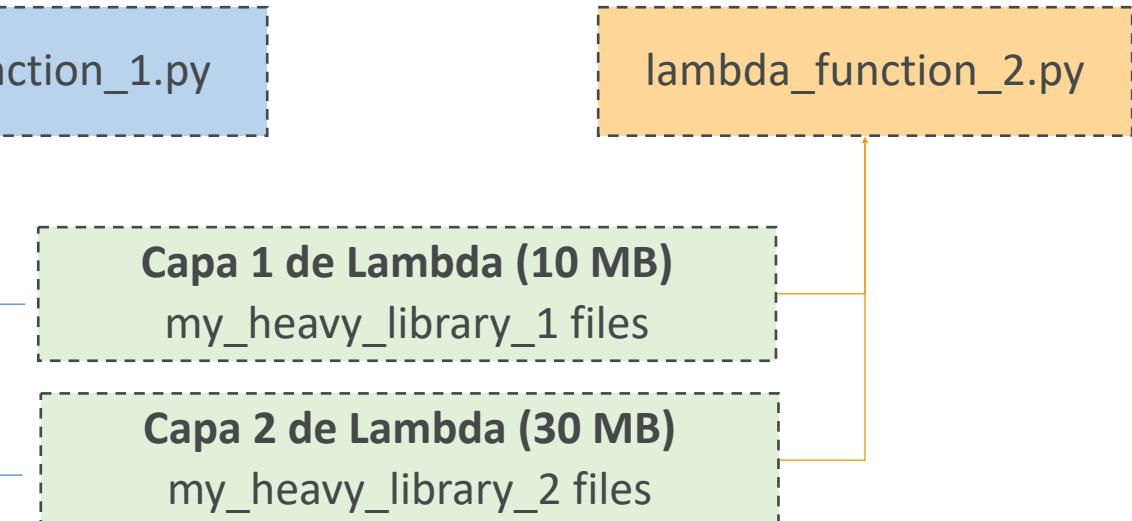
Paquete de aplicación 1 (30.02MB)



Paquete de aplicación 1 (20KB)

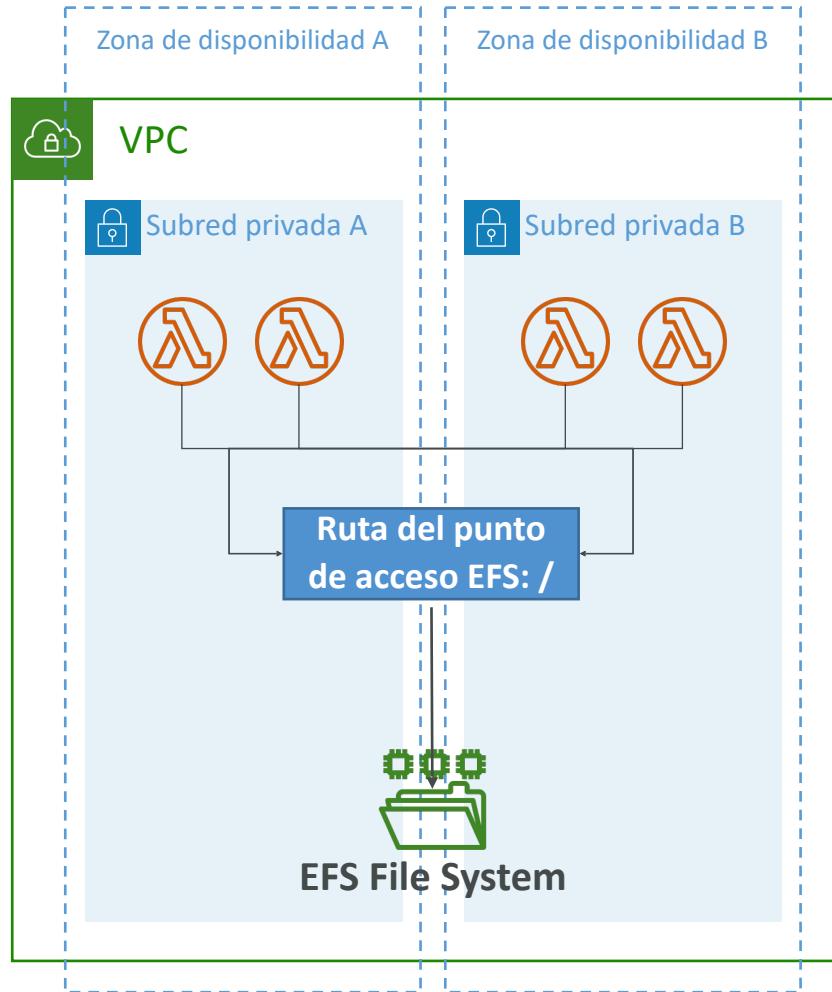


Paquete de aplicación 1 (60KB)



Lambda - Montaje de sistemas de archivos

- Las funciones Lambda pueden acceder a los sistemas de archivos EFS si se ejecutan en una VPC
- Configura Lambda para montar sistemas de archivos EFS en el directorio local durante la inicialización
- Debe aprovechar los puntos de acceso EFS
- Limitaciones: ten cuidado con los límites de conexión EFS (una instancia de función = una conexión) y los límites de ráfaga de conexión

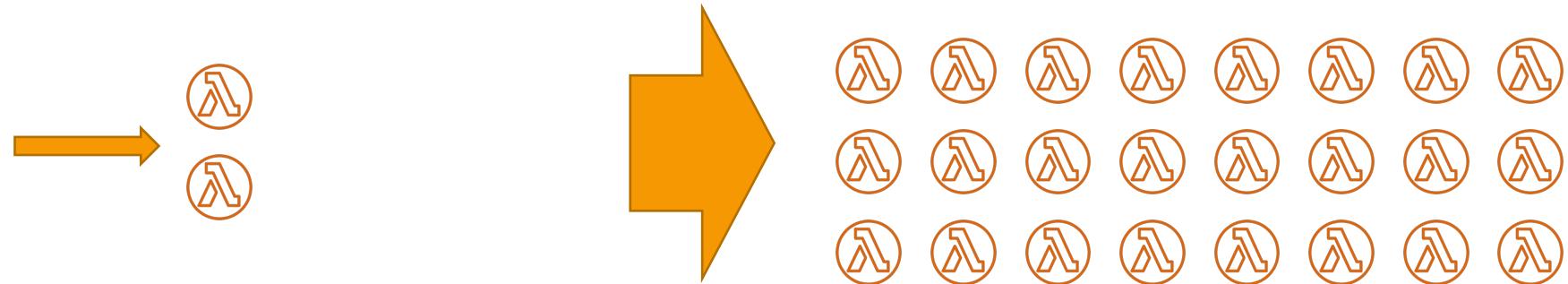


Lambda - Opciones de almacenamiento

	Almacenamiento efímero /tmp	Capas de lambda	Amazon S3	Amazon EFS
Tamaño máximo	10,240 MB	5 capas por función hasta 250 MB en total	Elástico	Elástico
Persistencia	Efímero	Duradero	Duradero	Duradero
Contenido	Dinámico	Estático	Dinámico	Dinámico
Tipo de almacenamiento	File System	Archive	Objeto	File System
Operaciones soportadas	cualquier operación del File System	Inmutable	Atómica con versionado	cualquier operación del File System
Precios	Incluido en Lambda	Incluido en Lambda	Almacenamiento + Peticiones + Transferencia de datos	Almacenamiento + Transferencia de datos + Rendimiento
Uso compartido/Permisos	Sólo para función lambda	IAM	IAM	IAM + NFS
Velocidad relativa de acceso a los datos desde Lambda	Más rápido	Más rápido	Rápido	Muy rápido
Compartido en todas las invocaciones	No	Sí	Sí	Sí

Concurrencia y estrangulamiento de Lambda

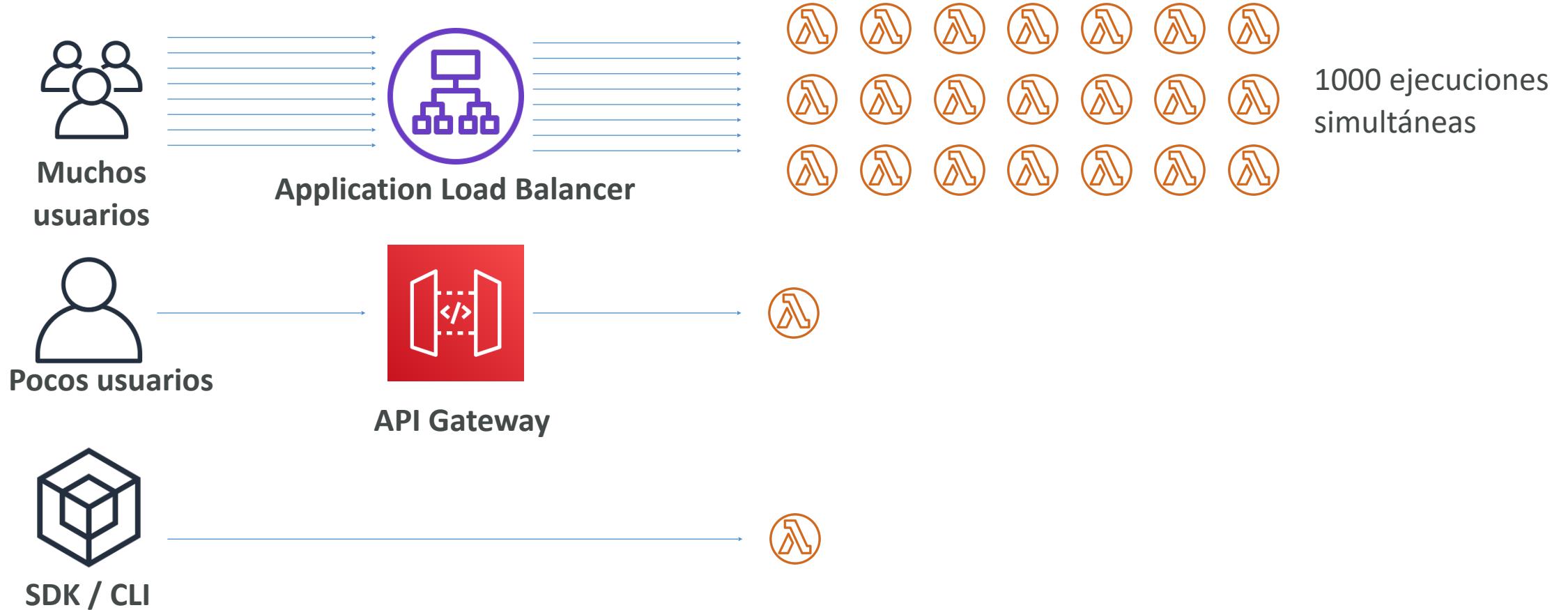
- Límite de concurrencia: hasta 1000 ejecuciones concurrentes



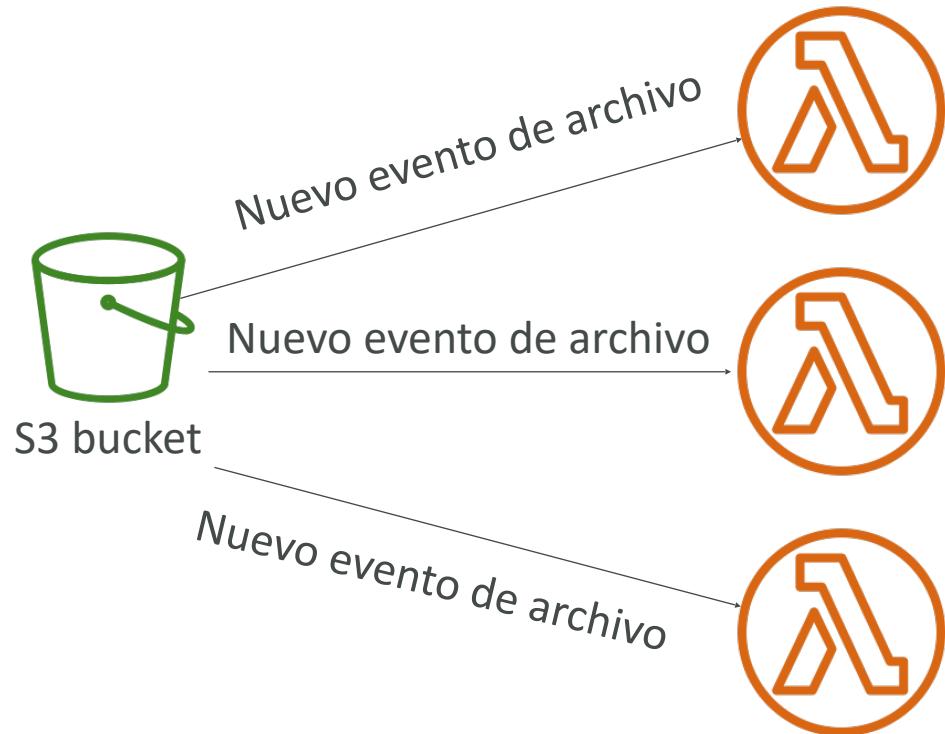
- Puedes establecer una "**concurrencia reservada**" a nivel de función (=límite)
- Cada invocación que supere el límite de concurrencia provocará un "Estrangulamiento"
- Comportamiento del "estrangulamiento":
 - Si la invocación es síncrona => devuelve ThrottleError - 429
 - Si la invocación es asíncrona => reintentar automáticamente y luego pasar a DLQ
- Si necesitas un límite mayor, abre un ticket de soporte

Problema de concurrencia de Lambda

- Si no reservas (=limitas) la concurrencia, puede ocurrir lo siguiente:



Concurrencia e invocaciones asíncronas



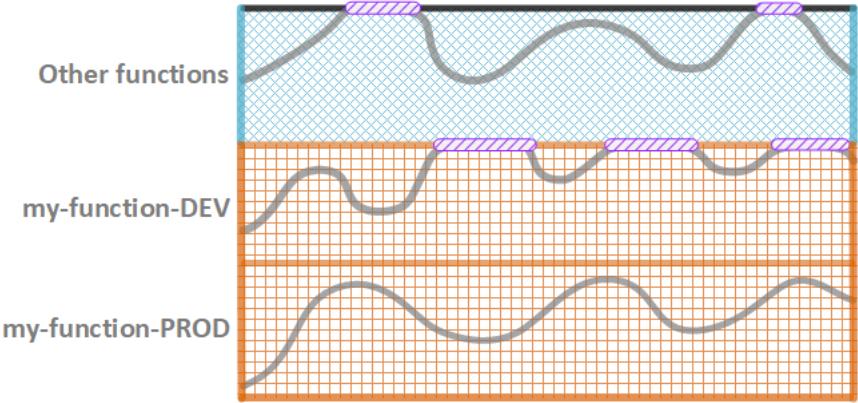
- Si la función no tiene suficiente concurrencia disponible para procesar todos los eventos, se estrangulan las peticiones adicionales.
- Para los errores de estrangulamiento (429) y los errores del sistema (serie 500), Lambda devuelve el evento a la cola e intenta volver a ejecutar la función durante un máximo de 6 horas.
- El intervalo de reintento aumenta exponencialmente desde 1 segundo después del primer intento hasta un máximo de 5 minutos.

Arranques en frío y concurrencia aprovisionada

- **Arranque en frío:**
 - Nueva instancia => se carga el código y se ejecuta el código fuera del manejador (init)
 - Si el init es grande (código, dependencias, SDK...) este proceso puede llevar algún tiempo.
 - La primera petición servida por las nuevas instancias tiene mayor latencia que el resto
- **Concurrencia aprovisionada:**
 - La capacidad de respuesta se asigna antes de invocar la función (por adelantado)
 - Así nunca se produce el arranque en frío y todas las invocaciones tienen baja latencia
 - El autoescalado de aplicaciones puede gestionar la concurrencia (programación o utilización objetivo)
- **Nota:**
 - Nota: los arranques en frío en VPC se han reducido drásticamente en octubre y noviembre de 2019
 - <https://aws.amazon.com/blogs/compute/announcing-improved-vpc-networking-for-aws-lambda-functions/>

Concurrencia reservada y provisionada

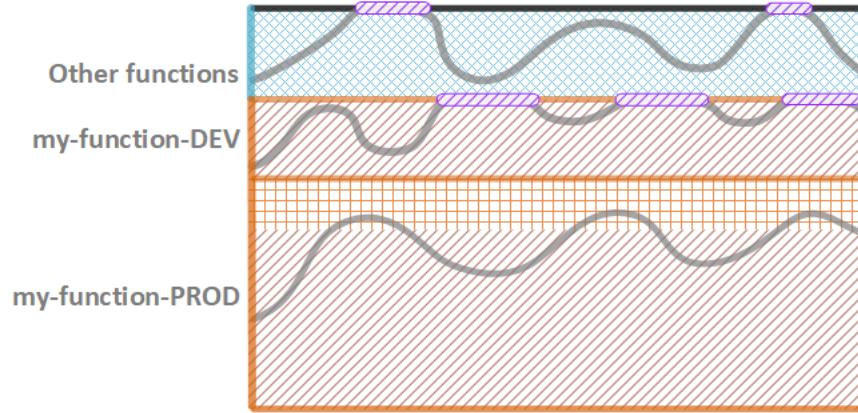
Reserved Concurrency



Legend

- Function concurrency
- Reserved concurrency
- Unreserved concurrency
- Throttling

Provisioned Concurrency with Reserved Concurrency



Legend

- Function concurrency
- Reserved concurrency
- Provisioned concurrency
- Unreserved concurrency
- Throttling

<https://docs.aws.amazon.com/lambda/latest/dg/configuration-concurrency.html>

Dependencias de la función lambda

- Si tu función Lambda depende de bibliotecas externas: por ejemplo, AWS X-Ray SDK, clientes de base de datos, etc...
- **Necesitas instalar los paquetes junto a tu código y comprimirlos juntos**
 - Para Node.js, utiliza npm y el directorio “node_modules”
 - Para Python, utiliza las opciones pip --target
 - Para Java, incluye los archivos .jar correspondientes
- Sube el **zip** directamente a Lambda si pesa menos de 50MB, si no, primero a S3
- Las bibliotecas nativas funcionan: deben compilarse en Amazon Linux
- AWS SDK viene por defecto con cada función Lambda

Lambda y CloudFormation - inline

```
AWSTemplateFormatVersion: '2010-09-09'
Description: Lambda function inline
Resources:
  primer:
    Type: AWS::Lambda::Function
    Properties:
      Runtime: python3.x
      Role: arn:aws:iam::123456789012:role/lambda-role
      Handler: index.handler
    Code:
      ZipFile: |
        import os

        DB_URL = os.getenv("DB_URL")
        db_client = db.connect(DB_URL)
        def handler(event, context):
          user = db_client.get(user_id = event["user_id"])
          return user
```

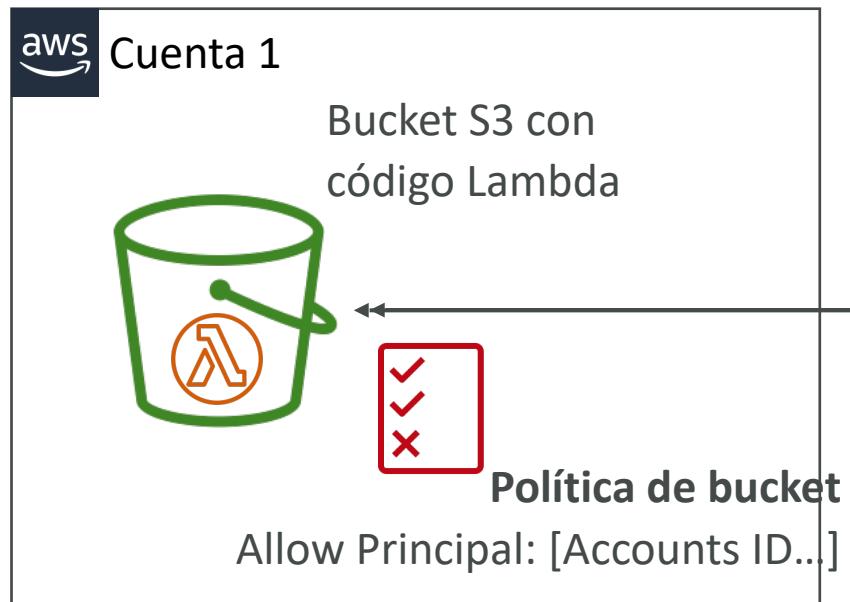
- Las funciones inline son muy sencillas
- Utiliza la propiedad **Code.ZipFile**
- No puedes incluir dependencias de funciones con funciones inline

Lambda y CloudFormation - a través de S3

```
AWSTemplateFormatVersion: '2010-09-09'
Description: Lambda from S3
Resources:
  Function:
    Type: AWS::Lambda::Function
    Properties:
      Handler: index.handler
      Role: arn:aws:iam::123456789012:role/lambda-role
      Code:
        S3Bucket: my-bucket
        S3Key: function.zip
        S3ObjectVersion: String
      Runtime: nodejs12.x
```

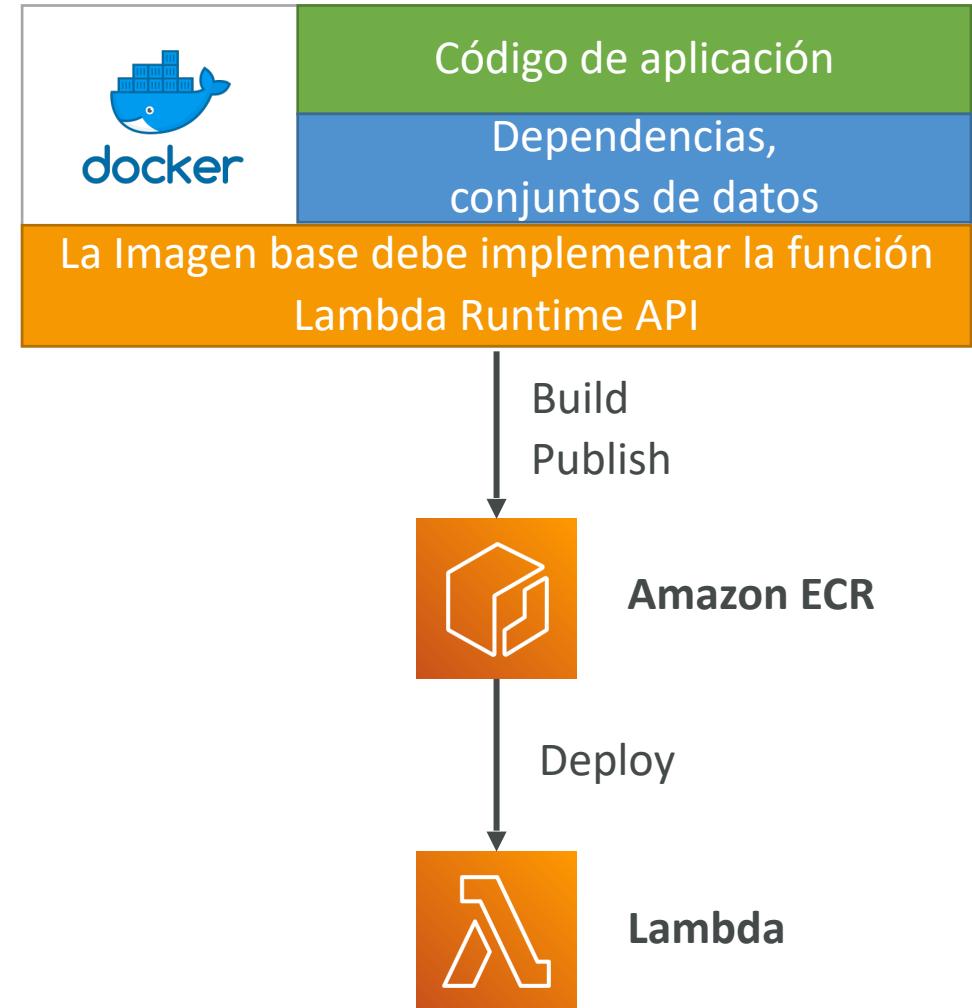
- Debes almacenar el zip de Lambda en S3
- Debes hacer referencia a la ubicación del zip de S3 en el código de CloudFormation
 - S3Bucket
 - S3Key: ruta completa al zip
 - S3ObjectVersion: si se trata de un bucket versionado
- **Si actualizas el código en S3, pero no actualizas S3Bucket, S3Key o S3ObjectVersion, CloudFormation no actualizará tu función**

Lambda y CloudFormation - a través de S3 Múltiples cuentas



Imágenes del contenedor Lambda

- Despliega la función Lambda como imágenes de contenedor de hasta 10 GB desde ECR
- Empaquea dependencias complejas, dependencias grandes en un contenedor
- Hay imágenes base disponibles para Python, Node.js, Java, .NET, Go, Ruby
- Puedes crear tu propia imagen siempre que implemente **la API Lambda Runtime**
- Testea los contenedores localmente utilizando el Emulador de la Interfaz de Tiempo de Ejecución Lambda
- Flujo de trabajo unificado para crear aplicaciones



Imágenes del contenedor Lambda

- Ejemplo: construir a partir de las imágenes base proporcionadas por AWS

```
# Utiliza una imagen que implemente la API Lambda Runtime
FROM amazon/aws-lambda-nodejs:12

# Copia el código y los archivos de tu aplicación
COPY app.js package*.json ./

# Instala las dependencias en el contenedor
RUN npm install

# Función a ejecutar cuando se invoque a la función Lambda
CMD [ "app.lambdaHandler" ]
```

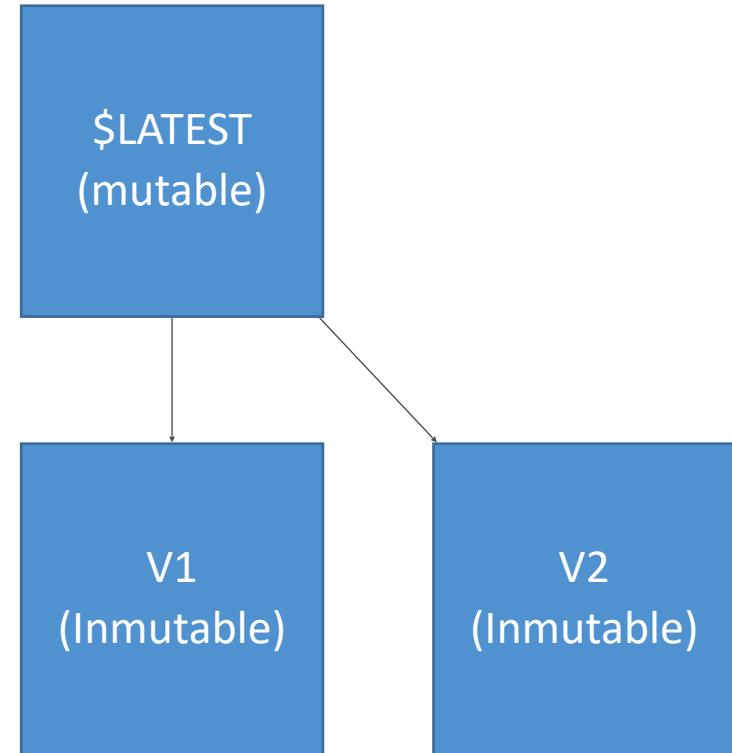
Imágenes de contenedores Lambda

Buenas prácticas

- Estrategias para optimizar las imágenes de contenedor:
 - **Utilizar imágenes base proporcionadas por AWS**
 - Estables, construidas en Amazon Linux 2, almacenadas en caché por el servicio Lambda
 - **Utiliza compilaciones multietapa**
 - Construye tu código en imágenes preliminares más grandes, copia sólo los artefactos que necesites en tu imagen de contenedor final, descarta los pasos preliminares
 - **Construye desde lo estable a los cambios frecuentes**
 - Realiza los cambios más frecuentes lo más tarde posible en tu Dockerfile
 - **Utiliza un único repositorio para funciones con grandes capas**
 - ECR compara cada capa de una imagen de contenedor cuando se envía para evitar subir y almacenar duplicados.
- Utilízalas para subir Funciones Lambda grandes (hasta 10 GB)

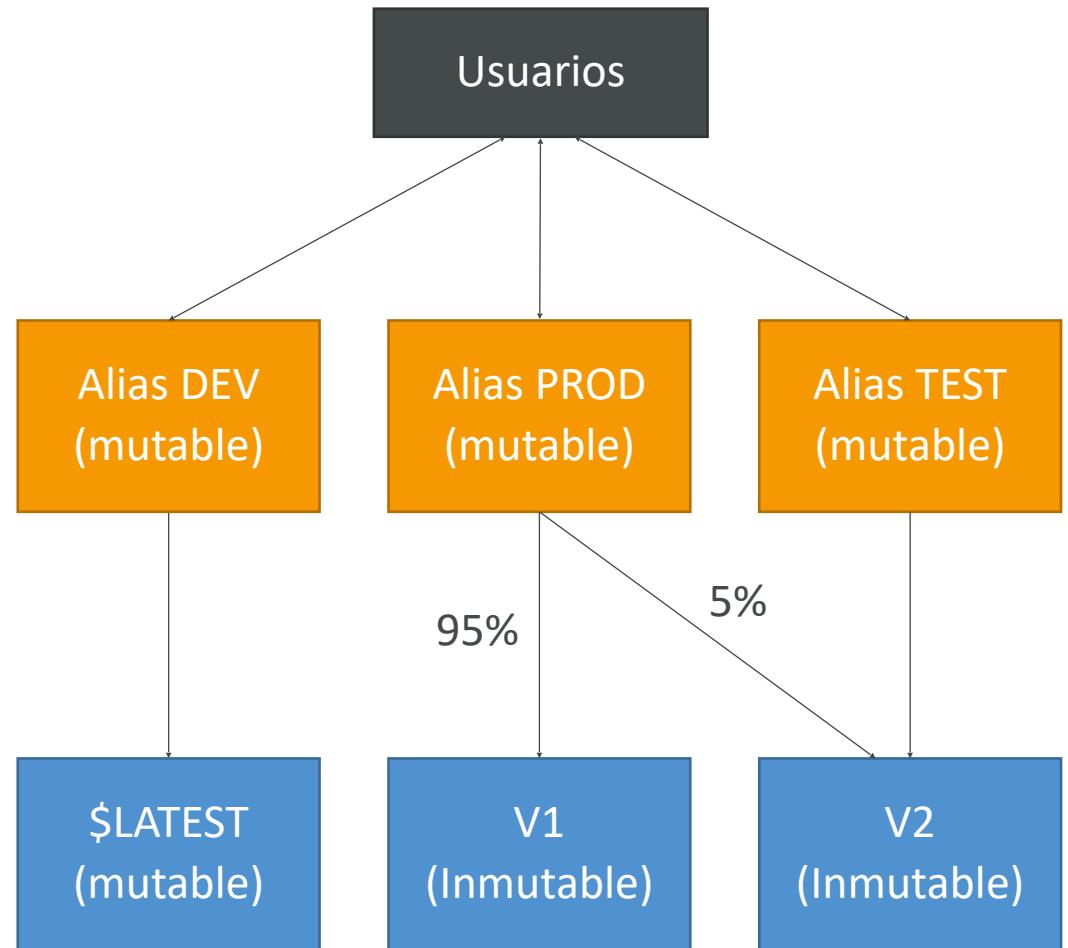
Versiones de AWS Lambda

- Cuando trabajas en una función Lambda, nosotros trabajamos en **\$LATEST**
- Cuando estamos listos para publicar una función Lambda, creamos una versión
- Las versiones son inmutables
- Las versiones tienen números de versión crecientes
- Las versiones tienen su propio ARN (Amazon Resource Name)
- Versión = código + configuración (no se puede cambiar nada - inmutable)
- Se puede acceder a cada versión de la función Lambda



Alias de AWS Lambda

- Los alias son "punteros" a versiones de funciones lambda
- Podemos definir alias "dev", "test", "prod" y hacer que apunten a distintas versiones de lambda
- Los alias son mutables
- Los alias permiten el despliegue de Canary asignando pesos a las funciones lambda
- Los alias permiten una configuración estable de nuestros desencadenantes / destinos de eventos
- Los alias tienen sus propios ARN
- **Los alias no pueden hacer referencia a alias**



Lambda y CodeDeploy

- **CodeDeploy** puede ayudarte a automatizar el cambio de tráfico para los alias de Lambda
- La función está integrada en el framework SAM
- **Lineal**: crece el tráfico cada N minutos hasta el 100%
 - Linear|0PercentEvery3Minutes
 - Linear|0PercentEvery10Minutes
- **Canary**: prueba X por ciento y luego 100%
 - Canary|0Percent5Minutes
 - Canary|0Percent30Minutes
- **AllAtOnce**: inmediato
- Puedes crear hooks pre y post tráfico para comprobar la salud de la función Lambda



Lambda y CodeDeploy – AppSpec.yml

```
version: 0.0
```

```
Resources:
```

```
- myLambdaFunction:
```

```
  Type: AWS::Lambda::Function
```

```
Properties:
```

```
  Name: myLambdaFunction
```

```
  Alias: myLambdaFunctionAlias
```

```
  CurrentVersion: 1
```

```
  TargetVersion: 2
```

- **Name (obligatorio)** - el nombre de la función Lambda a desplegar
- **Alias (obligatorio)** - el nombre del alias de la función Lambda
- **CurrentVersion (obligatorio)** - la versión de la función Lambda a la que apunta actualmente el tráfico
- **TargetVersion (obligatorio)** - la versión de la función Lambda a la que se desplaza el tráfico

Lambda - URL de la función

- Endpoint HTTP(S) dedicado para tu función Lambda
- Se genera un endpoint URL único para ti (nunca cambia)
 - **`https://<url-id>.lambda-url.<region>.on.aws (dual-stack IPv4 & IPv6)`**
- Invoca a través de un navegador web, curl, Postman o cualquier cliente HTTP
- Accede a la URL de tu función sólo a través del Internet público
 - No soporta PrivateLink (las funciones Lambda sí lo soportan)
- Soporta políticas basadas en recursos y configuraciones CORS
- Puede aplicarse a cualquier alias de función o a \$LATEST (no puede aplicarse a otras versiones de función)
- Crear y configurar mediante la consola de AWS o la API de AWS
- Acelera tu función utilizando la concurrencia reservada



Función lambda

[https://yj4xbxeirvacv3xdjp5uyt3j7y0ltzqa.
lambda-url.us-east-1.on.aws/](https://yj4xbxeirvacv3xdjp5uyt3j7y0ltzqa.lambda-url.us-east-1.on.aws/)

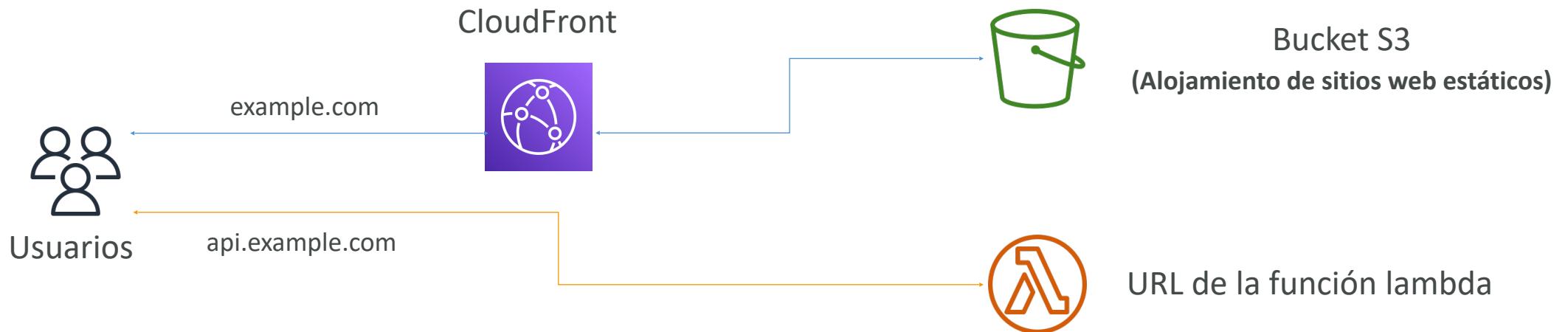
Lambda - Seguridad de la URL de la función

- **Política basada en recursos**

- Autorizar otras cuentas / CIDR específicos / IAM principales

- **Compartir recursos entre orígenes (CORS)**

- Si llamas a la URL de tu función Lambda desde un dominio diferente

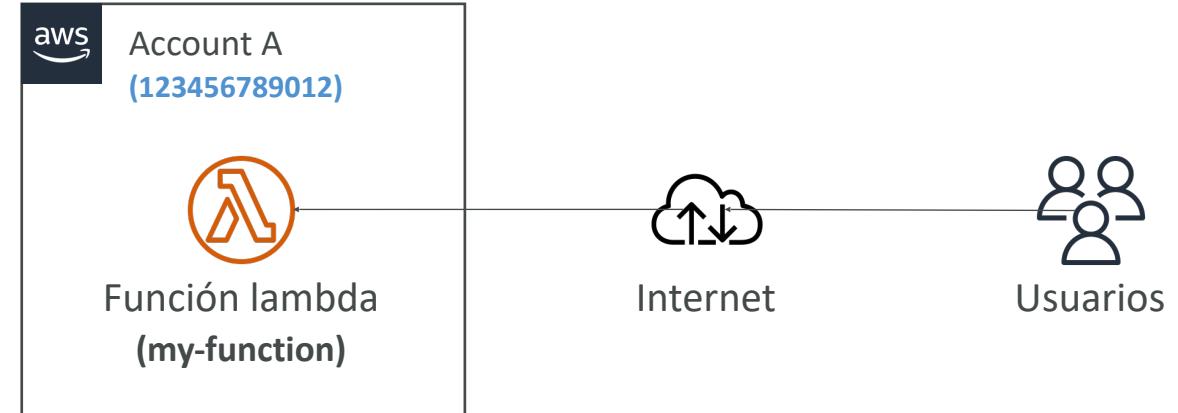


Lambda - Seguridad de la URL de la función

- **AuthType NONE** - permite el acceso público y no autenticado
 - La política basada en recursos está siempre en Effect (debe permitir el acceso público)

```
{  
  "Version": "2012-10-17",  
  "Statement": [  
    {  
      "Effect": "Allow",  
      "Principal": "*",  
      "Action": "lambda:InvokeFunctionUrl",  
      "Resource": "arn:aws:lambda:us-east-1:123456789012:  
function:my-function",  
      "Condition": {  
        "StringEquals": {  
          "lambda:FunctionUrlAuthType": "NONE"  
        }  
      }  
    }  
  ]  
}
```

Política basada en los recursos



Lambda - Seguridad de la URL de la función

- **AuthType AWS_IAM** - IAM se utiliza para autenticar y autorizar peticiones
 - Se evalúan tanto la Política basada en Identidad como la Política basada en recursos del Principal
 - El Principal debe tener permisos **lambda:InvokeFunctionUrl**
 - **Misma cuenta** - Política basada en la identidad  Política basada en recursos como ALLOW
 - **Cuenta cruzada** - Política basada en la identidad  Política basada en recursos como ALLOW

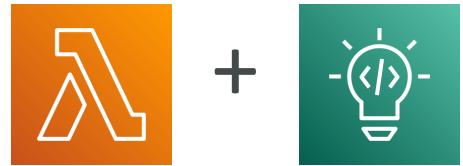
```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::444455556666:role/my-role"
      },
      "Action": "lambda:InvokeFunctionUrl",
      "Resource": "arn:aws:lambda:us-east-1:123456789012:function:my-function",
      "Condition": {
        "StringEquals": {
          "lambda:FunctionUrlAuthType": "AWS_IAM"
        }
      }
    }
  ]
}
```

Política basada en los recursos



Política basada en la identidad

Lambda y CodeGuru Profiling



- Obtén información sobre el rendimiento en tiempo de ejecución de tus funciones Lambda utilizando CodeGuru Profiler
- CodeGuru crea un grupo de perfiles para tu función Lambda
- Soporta tiempos de ejecución Java y Python
- Activar desde la consola de AWS Lambda
- Cuando se activa, Lambda añade:
 - Capa CodeGuru Profiler a tu función
 - Variables de entorno a tu función
 - Política **AmazonCodeGuruProfilerAgentAccess** a tu función

Función lambda
(MyFunction)



observaciones
sobre el
rendimiento en
tiempo de ejecución



CodeGuru Profiler
(aws-lambda-MyFunction)

Límites de AWS Lambda que debes conocer - por región

- **Ejecución:**

- Asignación de memoria: 128 MB - 10GB (incrementos de 1 MB)
- Tiempo máximo de ejecución: 900 segundos (15 minutos)
- Variables de entorno (4 KB)
- Capacidad del disco en el "contenedor de funciones" (en /tmp): 512 MB a 10GB
- Conurrencia de ejecuciones: 1000 (puede aumentarse)

- **Despliegue:**

- Tamaño del despliegue de la función Lambda (.zip comprimido): 50 MB
- Tamaño del despliegue sin comprimir (código + dependencias): 250 MB
- Puede utilizar el directorio /tmp para cargar otros archivos al inicio
- Tamaño de las variables de entorno: 4 KB

Mejores prácticas de AWS Lambda

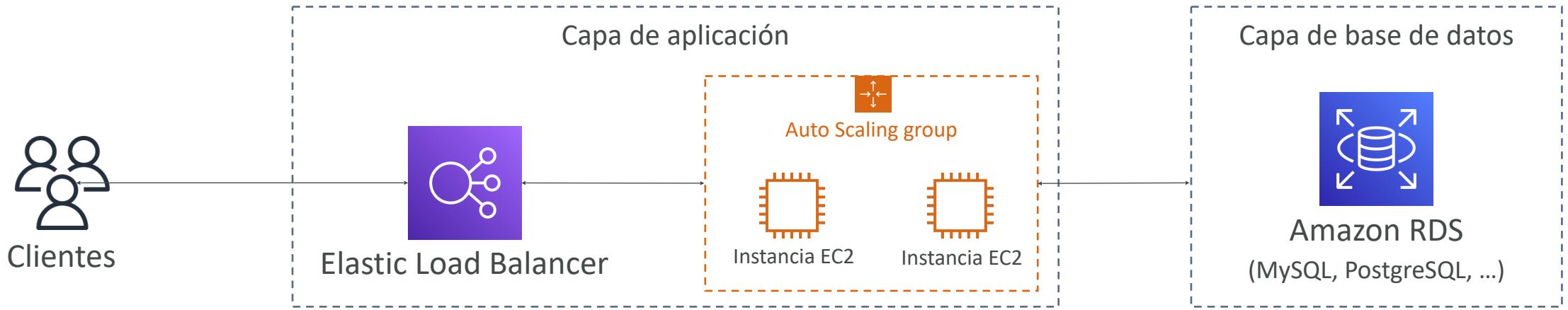


- **Realiza trabajos pesados fuera de tu controlador (handler) de funciones**
 - Conéctate a bases de datos fuera de tu controlador de funciones
 - Inicializar el SDK de AWS fuera de tu controlador de funciones
 - Introducir dependencias o conjuntos de datos fuera de tu controlador de funciones
- **Utiliza variables de entorno para:**
 - Strings de conexión a base de datos, S3 bucket, etc... no pongas estos valores en tu código
 - Contraseñas, valores sensibles... se pueden cifrar utilizando KMS
- **Minimiza el tamaño de tu paquete de despliegue a sus necesidades de tiempo de ejecución.**
 - Descompón la función si es necesario
 - Recuerda los límites de AWS Lambda
 - Utiliza capas cuando sea necesario
- **Evita utilizar código recursivo, nunca hagas que una función Lambda se llame a sí misma**

DynamoDB

Base de datos NoSQL sin servidor

Arquitectura tradicional



- Las aplicaciones tradicionales aprovechan las bases de datos RDBMS
- Estas bases de datos disponen del lenguaje de consulta SQL
- Requisitos estrictos sobre cómo deben modelarse los datos
- Capacidad para realizar consultas “JOIN”, agregaciones, cálculos complejos
- Escalado vertical (conseguir una CPU / RAM / IO más potente)
- Escalado horizontal (aumentar la capacidad de lectura añadiendo réplicas de lectura EC2 / RDS)

Bases de datos NoSQL

- Las bases de datos NoSQL son bases de datos no relacionales y están distribuidas
- Las bases de datos NoSQL incluyen MongoDB, DynamoDB, ...
- Las bases de datos NoSQL no soportan consultas "JOIN"
- Todos los datos necesarios para una consulta están presentes en una fila
- Las bases de datos NoSQL no realizan agregaciones como "SUM", "AVG", ...
- **Las bases de datos NoSQL escalan horizontalmente**
- No hay "correcto o incorrecto" para NoSQL frente a SQL, sólo requieren modelar los datos de forma diferente y pensar en las consultas de los usuarios de forma diferente



Amazon DynamoDB

- Totalmente gestionada, de alta disponibilidad con replicación en varias AZ
- Base de datos NoSQL - no una base de datos relacional
- Escala a cargas de trabajo masivas, base de datos distribuida
- Millones de peticiones por segundo, billones de filas, cientos de TB de almacenamiento
- Rendimiento rápido y constante (baja latencia en la recuperación)
- Integrada con IAM para seguridad, autorización y administración
- Permite la programación basada en eventos con DynamoDB Streams
- Bajo coste y capacidad de autoescalado
- Clase de tabla de acceso estándar e infrecuente (IA)

DynamoDB - Conceptos básicos

- DynamoDB está formado por **Tablas**
- Cada tabla tiene una **Clave Primaria** (debe decidirse en el momento de la creación)
- Cada tabla puede tener un número infinito de elementos (= filas)
- Cada elemento tiene **atributos** (pueden añadirse con el tiempo - pueden ser nulos)
- El tamaño máximo de un elemento es de **400 KB**
- Los tipos de datos soportados son:
 - **Tipos escalares** - String, Number, Binary, Boolean, Null
 - **Tipos de documento** - List, Map
 - **Tipos de conjuntos** - String Set, Number Set, Binary Set

DynamoDB - Claves primarias

- **Opción I: Clave de partición (HASH)**

- La clave de partición debe ser única para cada elemento
- La clave de partición debe ser "diversa" para que los datos estén distribuidos
- Ejemplo: "User_ID" para una tabla de usuarios

Clave primaria		Atributos		
Clave de partición				
User_ID		First_Name	Last_Name	Age
7791a3d6...		John	William	46
873e0634...		Oliver		24
a80f73a1...		Katie	Lucas	31

DynamoDB - Claves primarias

- **Opción 2: Clave de partición + Clave de ordenación / clasificación (HASH + RANGO)**
 - La combinación debe ser única para cada elemento
 - Los datos se agrupan por clave de partición
 - Ejemplo: tabla usuarios-juegos, “User_ID” como clave de partición e “Game_ID” como clave de clasificación / ordenación

Clave primaria		Atributos	
Clave de partición	Clave de clasificación	Score	Result
User_ID	Game_ID	92	Win
7791a3d6...	4421		
873e0634...	1894	14	Lose
873e0634...	4521	77	Win

Misma clave de partición
Diferente clave de ordenación

DynamoDB - Claves de partición (Ejercicio)

- Estamos construyendo una base de datos de películas
- ¿Cuál es la mejor Clave de Partición para maximizar la distribución de los datos?
 - movie_id
 - producer_name
 - leader_actor_name
 - movie_language
- “movie_id” tiene la cardinalidad más alta, por lo que es un buen candidato
- “movie_language” no admite muchos valores y puede estar sesgado hacia el inglés, por lo que no es una buena opción para la clave de partición

DynamoDB

Modos de capacidad de lectura/escritura

- Controla cómo gestionar la capacidad de tu tabla (rendimiento de lectura/escritura)
- **Modo aprovisionado (por defecto)**
 - Especificas el número de lecturas/escrituras por segundo
 - Es necesario planificar la capacidad de antemano
 - Pagas por unidades de capacidad de lectura (RCU) y unidades de capacidad de escritura (WCU) provisionadas
 - Posibilidad de añadir el modo de autoescalado para RCU y WCU
- **Modo bajo demanda**
 - Las lecturas/escrituras aumentan/disminuyen automáticamente con tus cargas de trabajo
 - No es necesario planificar la capacidad
 - Pagas por lo que utilizas, más caro (\$\$\$)
 - Ideal para cargas de trabajo impredecibles, picos repentinos pronunciados
- Puedes cambiar entre diferentes modos una vez cada 24 horas

Modos de capacidad R/W - Provisionado

- La tabla debe tener aprovisionadas unidades de capacidad de lectura y escritura
- **Unidades de Capacidad de Lectura (RCU)** - rendimiento para lecturas
- **Unidades de capacidad de escritura (WCU)** - rendimiento de escritura
- Opción para configurar el **escalado automático** del rendimiento para satisfacer la demanda
- El rendimiento puede superarse temporalmente utilizando la "**Capacidad de ráfaga**" (**Burst Capacity**)
- Si se ha consumido la capacidad de ráfaga, obtendrás una "**ProvisionedThroughputExceededException**"
- En ese caso, se aconseja hacer un reintento de **retroceso exponencial**

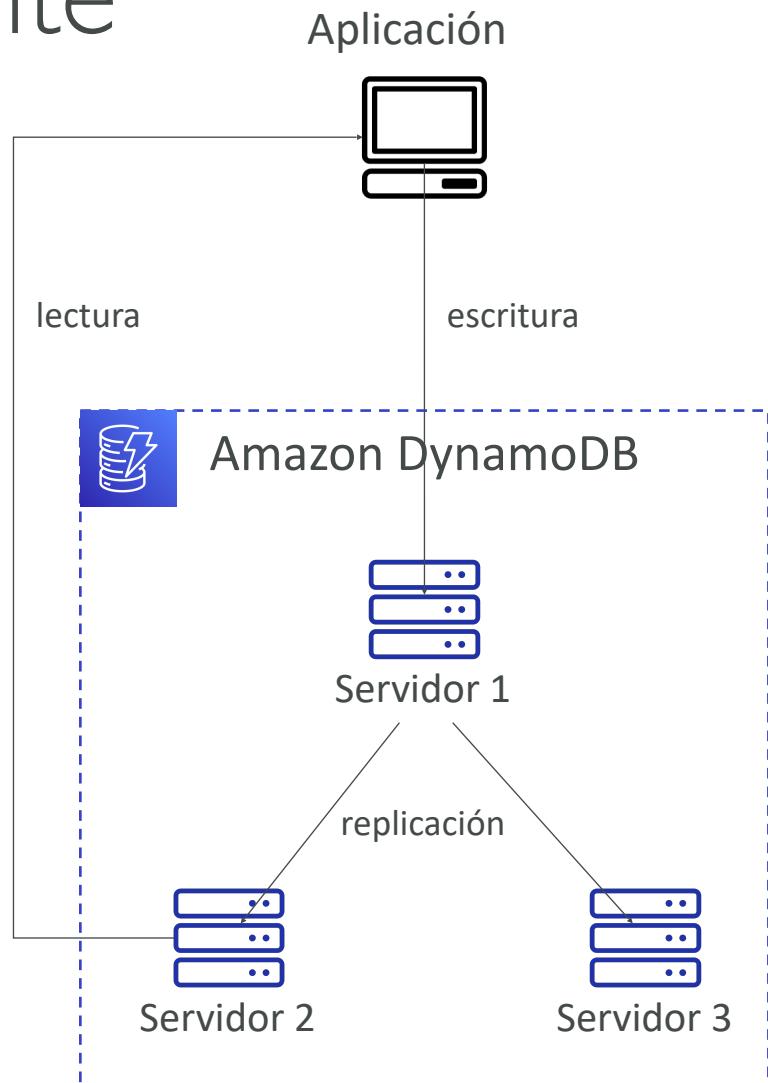
DynamoDB – Unidades de capacidad de escritura (WCU)

- Una Write Capacity Unit / Unidad de Capacidad de Escritura (WCU) representa una escritura por segundo para un elemento de hasta 1 KB de tamaño
- Si los elementos son mayores de 1 KB, se consumen más WCUs
- **Ejemplo 1:** escribimos 10 elementos por segundo, con un tamaño de elemento de 2 KB
 - Necesitamos $10 * \left(\frac{2 \text{ KB}}{1 \text{ KB}}\right) = 20 \text{ WCUs}$
- **Ejemplo 2:** escribimos 6 elementos por segundo, con un tamaño de elemento de 4,5 KB
 - Necesitamos $6 * \left(\frac{5 \text{ KB}}{1 \text{ KB}}\right) = 30 \text{ WCUs}$ (4,5 se redondea al KB superior)
- **Ejemplo 3:** escribimos 120 elementos por minuto, con un tamaño de elemento de 2 KB
 - Necesitamos $\left(\frac{120}{60}\right) * \left(\frac{2 \text{ KB}}{1 \text{ KB}}\right) = 4 \text{ WCUs}$

Lectura fuertemente consistente vs. Lectura eventualmente consistente

- **Lectura eventualmente consistente (por defecto)**

- Si leemos justo después de una escritura, es posible que obtengamos algunos datos obsoletos debido a la replicación



- **Lectura fuertemente consistente**

- Si leemos justo después de una escritura, obtendremos los datos correctos
- Pon el parámetro "**ConsistentRead**" a **True** en las llamadas a la API (GetItem, BatchGetItem, Query, Scan)
- Consume el doble de RCU

DynamoDB – Unidades de capacidad de lectura (RCUs)

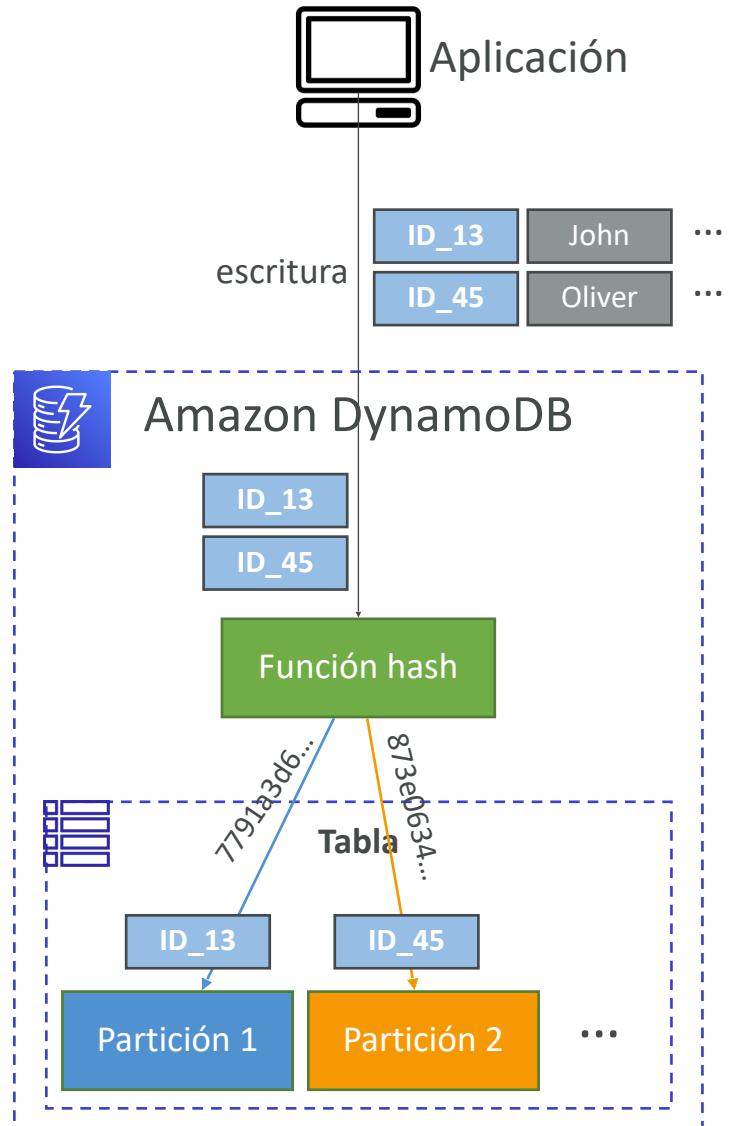
- Una *Read Capacity Unit / Unidad de Capacidad de Lectura (RCU)* representa **una lectura fuertemente consistente** por segundo, o **dos lecturas eventualmente consistentes** por segundo, para un elemento de hasta **4 KB** de tamaño
- Si los elementos son mayores de 4 KB, se consumen más RCUs
- **Ejemplo 1:** 10 lecturas fuertemente consistentes por segundo, con un tamaño de elemento de 4 KB
 - Necesitamos $10 * \left(\frac{4 \text{ KB}}{4 \text{ KB}} \right) = 10 \text{ RCUs}$
- **Ejemplo 2:** 16 Lecturas eventualmente consistentes por segundo, con un tamaño de elemento de 12 KB
 - Necesitamos $\left(\frac{16}{2} \right) * \left(\frac{12 \text{ KB}}{4 \text{ KB}} \right) = 24 \text{ RCUs}$
- **Ejemplo 3:** 10 lecturas fuertemente consistentes por segundo, con un tamaño de elemento de 6 KB
 - Necesitamos $10 * \left(\frac{8 \text{ KB}}{4 \text{ KB}} \right) = 20 \text{ RCUs}$ (debemos redondear 6 KB a 8 KB)

DynamoDB - Particiones internas

- Los datos se almacenan en particiones
- Las claves de las particiones pasan por un algoritmo hash para saber a qué partición van
- Para calcular el número de particiones:

- $\# \text{ of partitions}_{\text{by capacity}} = \left(\frac{\text{RCUs}_{\text{Total}}}{3000} \right) + \left(\frac{\text{WCUs}_{\text{Total}}}{1000} \right)$
- $\# \text{ of partitions}_{\text{by size}} = \frac{\text{Total Size}}{10 \text{ GB}}$
- $\# \text{ of partitions} = \text{ceil}(\max(\# \text{ of partitions}_{\text{by capacity}}, \# \text{ of partitions}_{\text{by size}}))$

• **Las WCUs y RCUs se reparten uniformemente por las particiones**



DynamoDB – Throttling / Estrangulamiento

- Si superamos las RCUs o WCUs provisionadas, obtenemos “**ProvisionedThroughputExceededException**”
- Motivos:
 - **Claves calientes** - una clave de partición se lee demasiadas veces (por ejemplo, elemento popular)
 - **Particiones calientes**
 - **Elementos muy grandes**, recuerda que RCU y WCU dependen del tamaño de los elementos
- Solución:
 - **Retroceso exponencial** cuando se encuentra una excepción (disponible en el SDK)
 - **Distribuye las claves de partición** tanto como sea posible
 - Si hay problemas de RCU, podemos **utilizar el Acelerador de DynamoDB (DAX)**

Modos de capacidad R/W - Bajo demanda

- Las lecturas/escrituras aumentan/disminuyen automáticamente con tus cargas de trabajo
- No necesitas planificar la capacidad (WCU / RCU)
- WCU y RCU ilimitadas, sin acelerador, más caras
- Te cobran por las lecturas/escrituras que utilizas en términos de RRU y WRU
- **Unidades de petición de lectura (RRU)** - rendimiento de las lecturas (igual que RCU)
- **Unidades de petición de escritura (WRU)** - rendimiento de las escrituras (igual que WCU)
- 2,5 veces más caro que la capacidad provisionada (¡con cuidado!)
- Casos de uso: cargas de trabajo desconocidas, tráfico de aplicación impredecible, ...

DynamoDB - Escritura de datos

- **PutItem**

- Crea un nuevo elemento o sustituye totalmente un elemento antiguo (misma Clave Primaria)
- Consume WCUs

- **UpdateItem**

- Edita los atributos de un elemento existente o añade uno nuevo si no existe
- Puede utilizarse para implementar **contadores atómicos** - un atributo numérico que se incrementa incondicionalmente

- **Conditional Writes**

- Acepta una escritura/actualización/borrado sólo si se cumplen las condiciones, de lo contrario devuelve un error
- No afecta al rendimiento

DynamoDB - Lectura de datos

- **GetItem**

- Lectura basada en clave primaria
- La clave primaria puede ser **HASH** o **HASH+RANGE**
- Lectura Eventualmente Consistente (por defecto)
- Opción de utilizar Lecturas Fuertemente Consistentes (más RCU - puede tardar más)
- Se puede especificar una **ProjectionExpression** para recuperar sólo determinados atributos

DynamoDB - Lectura de datos (Consulta)

- La consulta devuelve elementos basados en:
 - **KeyConditionExpression**
 - Valor de la Clave de Partición (**debe ser = el operador**) - obligatorio
 - Valor de la Clave de Ordenación (=, <, <=, >, >=, Entre, Empieza por) - opcional
 - **FilterExpression**
 - Filtrado adicional después de la operación de Consulta (antes de que te devuelvan los datos)
 - Utilízalo sólo con atributos que no sean clave (no permite atributos HASH o RANGE)
- Devuelve:
 - El número de elementos especificados en **Limit**
 - O hasta 1 MB de datos
- Posibilidad de hacer paginación en los resultados
- Puedes hacer consultas a la tabla, un Índice Secundario Local o un Índice Secundario Global

DynamoDB - Lectura de datos (Escaneado)

- **Escanea** toda la tabla y luego filtra los datos (ineficiente)
- Devuelve hasta 1 MB de datos - utiliza la paginación para seguir leyendo
- Consume mucha RCU
- Limita el impacto utilizando **Limit** o reduce el tamaño del resultado
- Para un rendimiento más rápido, utiliza **escaneado paralelo**
 - Varios “workers” escanean varios segmentos de datos al mismo tiempo
 - Aumenta el rendimiento y la RCU consumida
 - Limita el impacto de los escaneos paralelos igual que harías con los escaneos
- Puedes utilizar **ProjectionExpression** y **FilterExpression** (sin cambios en la RCU)

DynamoDB - Eliminación de datos

- **DeleteItem**

- Borrar un elemento individual
- Posibilidad de realizar una eliminación condicional

- **DeleteTable**

- Borrar una tabla entera y todos sus elementos
- Eliminación mucho más rápida que llamar a **DeleteItem** en todos los ítems

DynamoDB - Operaciones por lotes

- Te permite ahorrar en latencia reduciendo el número de llamadas a la API
- Las operaciones se realizan en paralelo para una mayor eficiencia
- Parte de un lote puede fallar, en cuyo caso hay que volver a intentarlo con los elementos fallidos
- **BatchWriteItem**
 - Hasta 25 **PutItem** y/o **DeleteItem** en una llamada
 - Hasta 16 MB de datos escritos, hasta 400 KB de datos por elemento
 - No se pueden actualizar los elementos (utiliza **UpdateItem**)
 - UnprocessedItems para operaciones de escritura fallidas (backoff / retroceso exponencial o añadir WCU)
- **BatchGetItem**
 - Devuelve elementos de una o varias tablas
 - Hasta 100 elementos, hasta 16 MB de datos
 - Los elementos se recuperan en paralelo para minimizar la latencia
 - UnprocessedKeys para operaciones de lectura fallidas (backoff / retroceso exponencial o añadir RCU)

DynamoDB – PartiQL

- Lenguaje de consulta compatible con SQL para DynamoDB
- Te permite seleccionar, insertar, actualizar y eliminar datos en DynamoDB mediante SQL
- Ejecuta consultas en varias tablas de DynamoDB
- Ejecuta consultas PartiQL desde:
 - Consola de administración de AWS
 - NoSQL Workbench para DynamoDB
 - API de DynamoDB
 - AWS CLI
 - AWS SDK

```
SELECT OrderID, Total  
FROM Orders  
WHERE OrderID IN [1, 2, 3]  
ORDER BY OrderID DESC
```

DynamoDB - Escrituras condicionales

- Para **PutItem**, **UpdateItem**, **DeleteItem**, y **BatchWriteItem**
- Puedes especificar una expresión de condición para determinar qué elementos deben modificarse:
 - **attribute_exists**
 - **attribute_not_exists**
 - **attribute_type**
 - **contains** (para string)
 - **begins_with** (para string)
 - ProductCategory **IN** (:cat1, :cat2) y precio entre :low and :high
 - **size** (longitud del string)
- **Nota:** Las expresiones de filtro filtran los resultados de las consultas de lectura, mientras que las expresiones de condición son para las operaciones de escritura

Escrituras condicionales - Ejemplo en actualizar elemento

```
aws dynamodb update-item \  
  --table-name ProductCatalog \  
  --key '{ "Id": { "N": "456" } }' \  
  --update-expression "SET Price = Price - :discount" \  
  --condition-expression "Price > :limit" \  
  --expression-attribute-values file://values.json
```

```
{  
  ":discount": {  
    "N": "150"  
  },  
  ":limit": {  
    "N": "500"  
  }  
}
```

values.json

```
{  
  "Id": {  
    "N": "456"  
  },  
  "Price": {  
    "N": "650"  
  },  
  "ProductCategory": {  
    "S": "Sporting Goods"  
  }  
}
```



```
{  
  "Id": {  
    "N": "456"  
  },  
  "Price": {  
    "N": "500"  
  },  
  "ProductCategory": {  
    "S": "Sporting Goods"  
  }  
}
```

Escrituras condicionales - Ejemplo en eliminar elemento

- **attribute_not_exists**

- Sólo tiene éxito si el atributo aún no existe (sin valor)

```
aws dynamodb delete-item \  
  --table-name ProductCatalog \  
  --key '{ "Id": { "N": "456" } }' \  
  --condition-expression "attribute_not_exists(Price)"
```

- **attribute_exists**

- Opuesto a attribute_not_exists

```
aws dynamodb delete-item \  
  --table-name ProductCatalog \  
  --key '{ "Id": { "N": "456" } }' \  
  --condition-expression "attribute_exists(ProductReviews.OneStar)"
```

Escrituras condicionales - No sobrescribir elementos

- **attribute_not_exists(partition_key)**
 - Asegúrate de que el elemento no se sobrescribe
- **attribute_not_exists(partition_key) y attribute_not_exists(sort_key)**
 - Asegúrate de que no se sobrescribe la combinación de clave partición / ordenación

Escrituras condicionales - Ejemplo de condición compleja

```
aws dynamodb delete-item \
--table-name ProductCatalog \
--key '{ "Id": { "N": "456" } }' \
--condition-expression "(ProductCategory IN (:cat1, :cat2)) and (Price between :lo and :hi)" \
--expression-attribute-values file://values.json
```

```
{
    ":cat1": {
        "S": "Sporting Goods"
    },
    ":cat2": {
        "S": "Gardening Supplies"
    },
    ":lo": {
        "N": "500"
    },
    ":hi": {
        "N": "600"
    }
}
```

values.json

```
{
    "Id": {
        "N": "456"
    },
    "Price": {
        "N": "650"
    },
    "ProductCategory": {
        "S": "Sporting Goods"
    }
}
```

Escrituras condicionales - Ejemplo de comparación de Strings

- **begins_with** – comprobar si el prefijo coincide
- **contains** – comprobar si una string está contenida en otra string

```
aws dynamodb delete-item \
  --table-name ProductCatalog \
  --key '{ "Id": { "N": "456" } }' \
  --condition-expression "begins_with(Pictures.FrontView, :v_sub)" \
  --expression-attribute-values file://values.json
```



```
{
    ":v_sub": {
        "S": "http://"
    }
}
```

values.json

DynamoDB - Índice Secundario Local (LSI)

- **Clave de ordenación alternativa** para tu tabla (la misma **clave de partición** que la de la tabla base)
- La clave de ordenación consiste en un atributo escalar (String, Number o Binary)
- Hasta 5 Índices Secundarios Locales (LSI) por tabla
- **Deben definirse al crear la tabla**
- **Proyecciones de atributos** - pueden contener algunos o todos los atributos de la tabla base (**KEYS_ONLY**, **INCLUDE**, **ALL**)

Clave primaria		Atributos		
Clave de partición	Clave de ordenación	LSI		
User_ID	Game_ID	Game_TS	Score	Result
7791a3d6...	4421	"2021-03-15T17:43:08"	92	Win
873e0634...	4521	"2021-06-20T19:02:32"		Lose
a80f73a1...	1894	"2021-02-11T04:11:31"	77	Win

DynamoDB - Índice Secundario Global (GSI)

- **Clave primaria alternativa (HASH o HASH+RANGE)** de la tabla base
- Acelera las consultas sobre atributos no clave
- La clave índice consiste en atributos escalares
- **Proyecciones de atributos** - algunos o todos los atributos de la tabla base (**ALL, KEYS_ONLY, INCLUDE**)
- Debe proporcionar RCU's y WCU's para el índice
- **Pueden añadirse/modificarse tras la creación de la tabla**

Clave de partición	Clave de ordenación	Atributos
User_ID	Game_ID	Game_TS
7791a3d6-...	4421	"2021-03-15T17:43:08"
873e0634-...	4521	"2021-06-20T19:02:32"
a80f73a1-...	1894	"2021-02-11T04:11:31"

TABLE (consulta por "User_ID")

Clave de partición	Clave de ordenación	Atributos
Game_ID	Game_TS	User_ID
4421	"2021-03-15T17:43:08"	7791a3d6-...
4521	"2021-06-20T19:02:32"	873e0634-...
1894	"2021-02-11T04:11:31"	a80f73a1-...

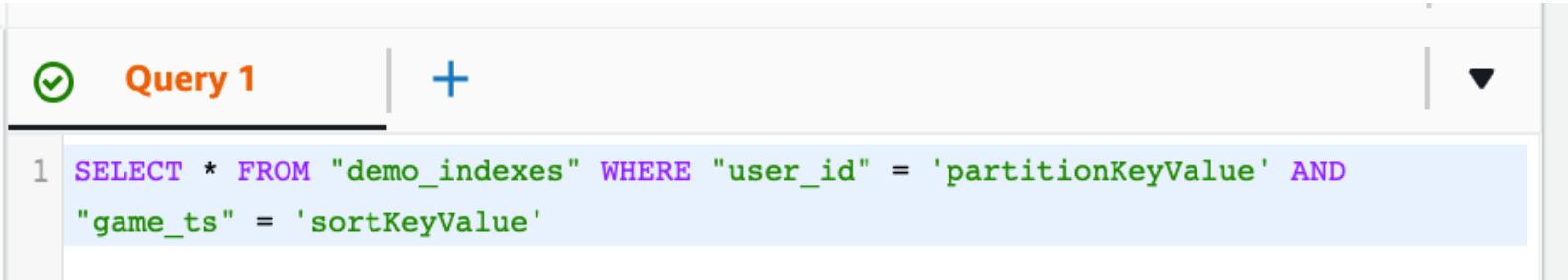
INDEX GSI (consulta por "Game_ID")

DynamoDB - Índices y estrangulamiento

- Índice Secundario Global (GSI):
 - **Si se estrangulan las escrituras en el GSI, ¡se estrangulará la tabla principal!**
 - Aunque las WCU de las tablas principales estén bien
 - ¡Elige con cuidado tu clave de partición GSI!
 - ¡Asigna cuidadosamente la capacidad de tu WCU!
- Índice Secundario Local (LSI):
 - Utiliza las WCUs y RCUs de la tabla principal
 - Sin consideraciones especiales de estrangulamiento

DynamoDB - PartiQL

- Utiliza una sintaxis similar a SQL para manipular las tablas de DynamoDB

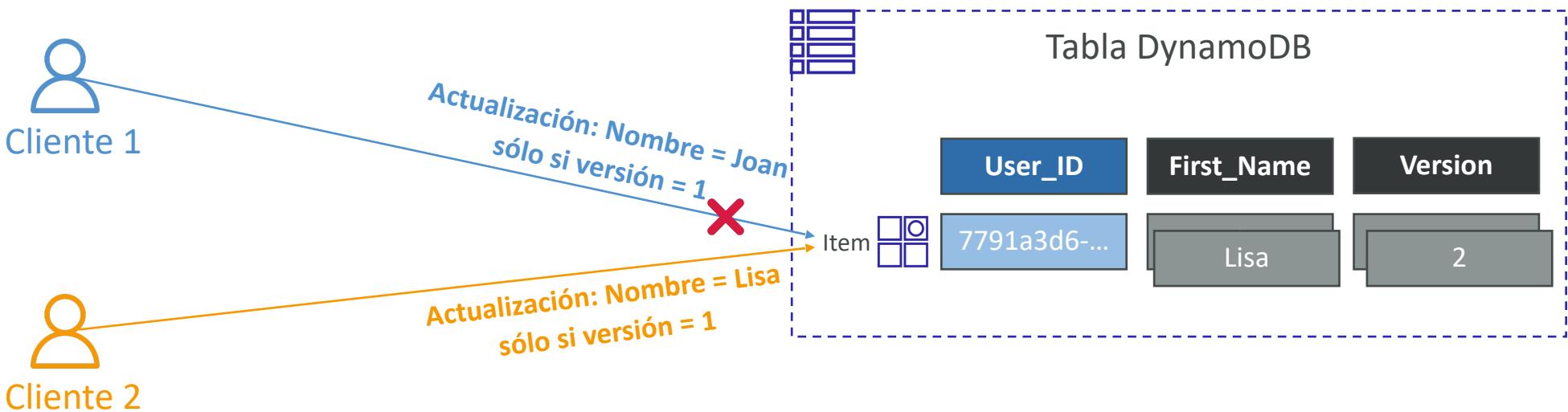


```
Query 1 + ▾  
1 SELECT * FROM "demo_indexes" WHERE "user_id" = 'partitionKeyValue' AND  
    "game_ts" = 'sortKeyValue'
```

- Soporta algunas declaraciones (pero no todas):
 - INSERT
 - UPDATE
 - SELECT
 - DELETE
- Soporta operaciones por lotes

DynamoDB - Bloqueo optimista

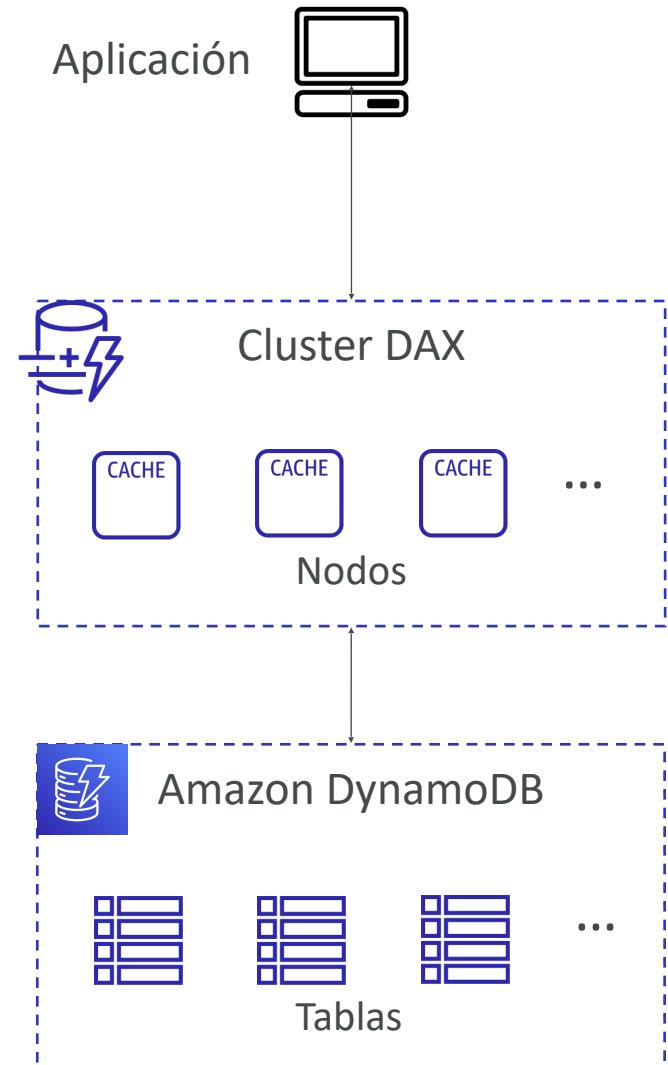
- DynamoDB tiene una función llamada “**Conditional Writes**”
- Una estrategia para asegurarte de que un elemento no ha cambiado antes de actualizarlo/eliminarlo
- Cada elemento tiene un atributo que actúa como número de versión



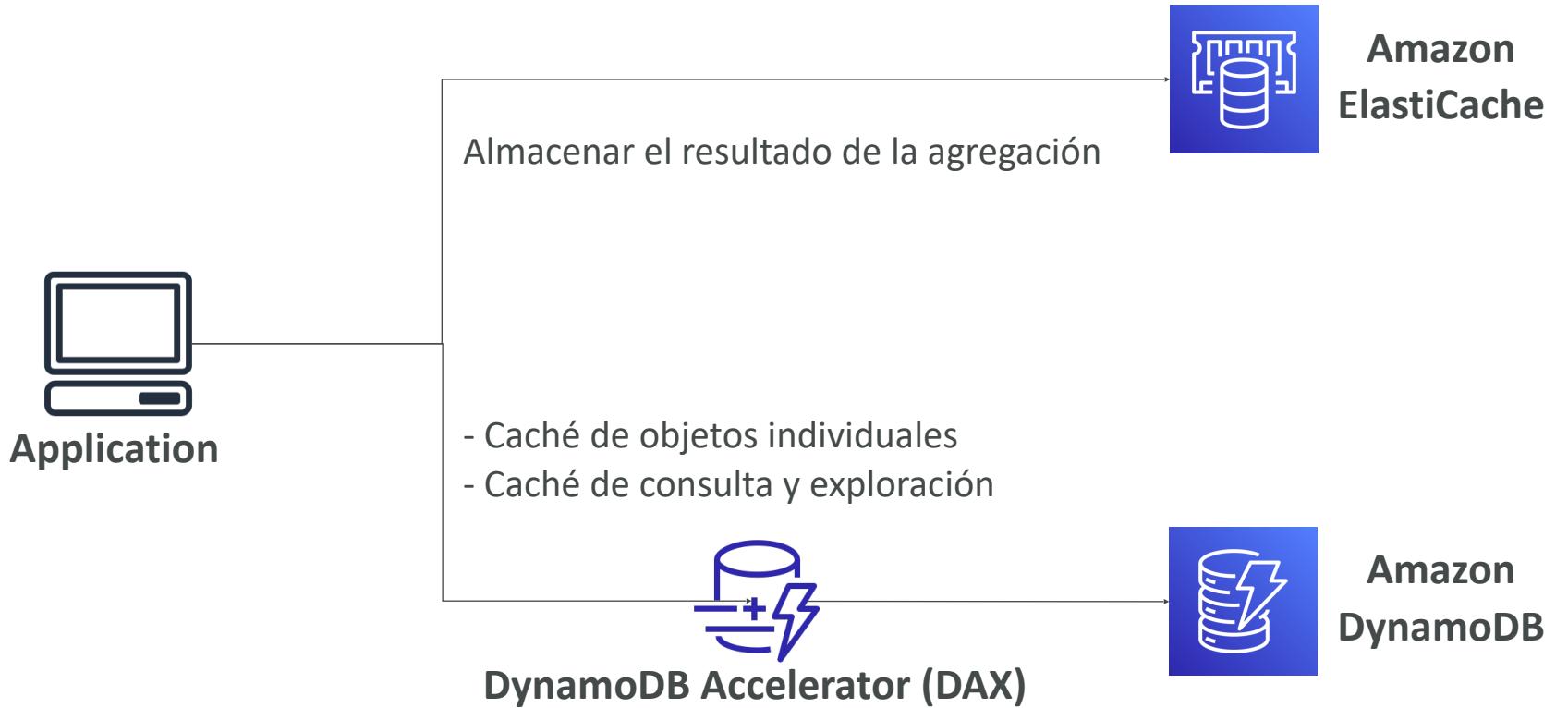
DynamoDB Accelerator (DAX)



- Caché en memoria totalmente gestionada, de alta disponibilidad y sin fisuras para DynamoDB
- Latencia de microsegundos para lecturas y consultas en caché
- No requiere modificar la lógica de la aplicación (compatible con las API de DynamoDB existentes)
- Resuelve el problema de la “clave caliente” (demasiadas lecturas)
- 5 minutos de TTL para la caché (por defecto)
- Hasta 11 nodos en el cluster
- Multi-AZ (3 nodos como mínimo recomendados para producción)
- Seguro (cifrado en reposo con KMS, VPC, IAM, CloudTrail, ...)



DynamoDB Accelerator (DAX) vs. ElastiCache

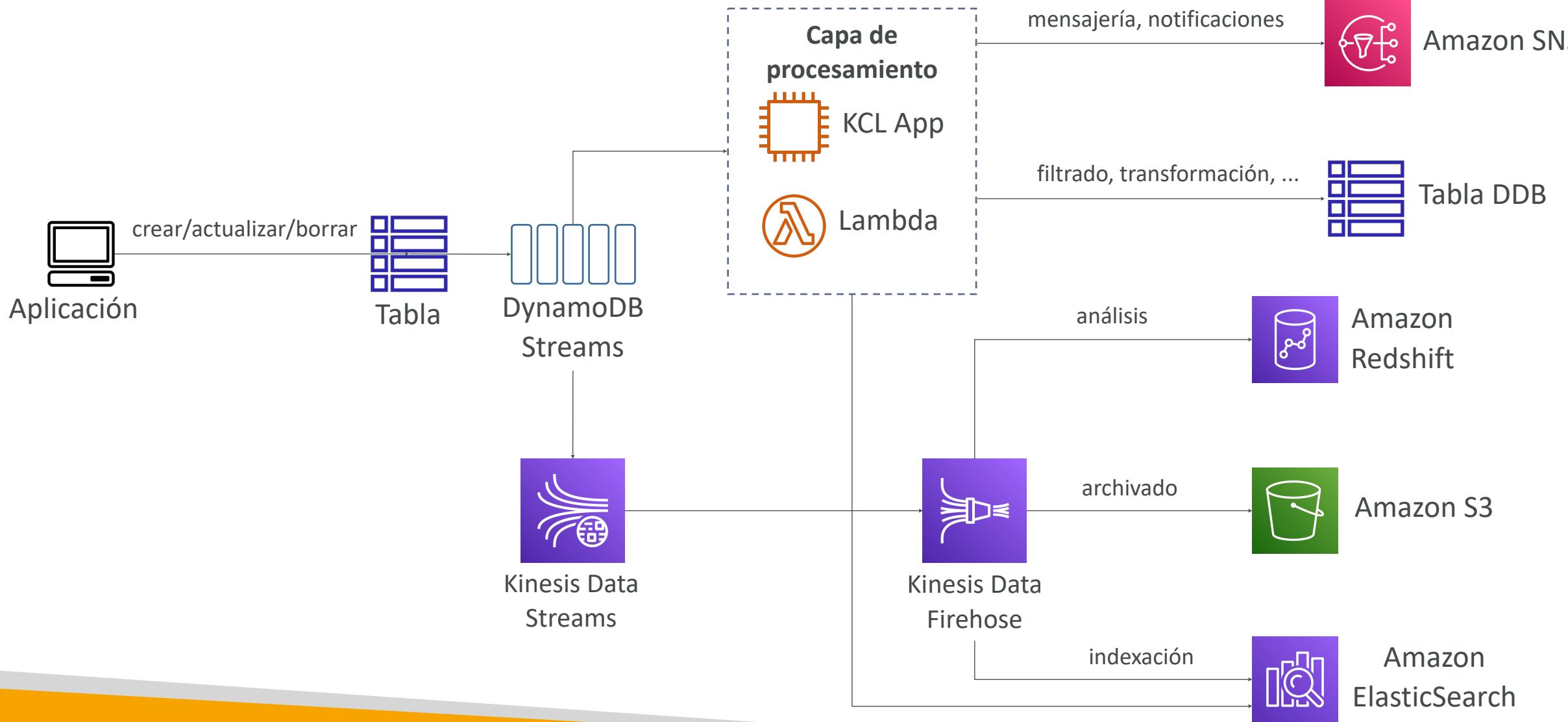




DynamoDB Streams

- Stream ordenado de modificaciones a nivel de elemento (crear/actualizar/borrar) en una tabla
- Los registros del stream pueden ser:
 - Enviados a **Kinesis Data Streams**
 - Leídos por **AWS Lambda**
 - Leídos por **aplicaciones de la biblioteca de clientes de Kinesis**
- Retención de datos hasta 24 horas
- Casos de uso:
 - Reaccionar a los cambios en tiempo real (correo electrónico de bienvenida a los usuarios)
 - Analíticas
 - Insertar en tablas derivadas
 - Inserción en ElasticSearch
 - Implementar la replicación entre regiones

DynamoDB Streams

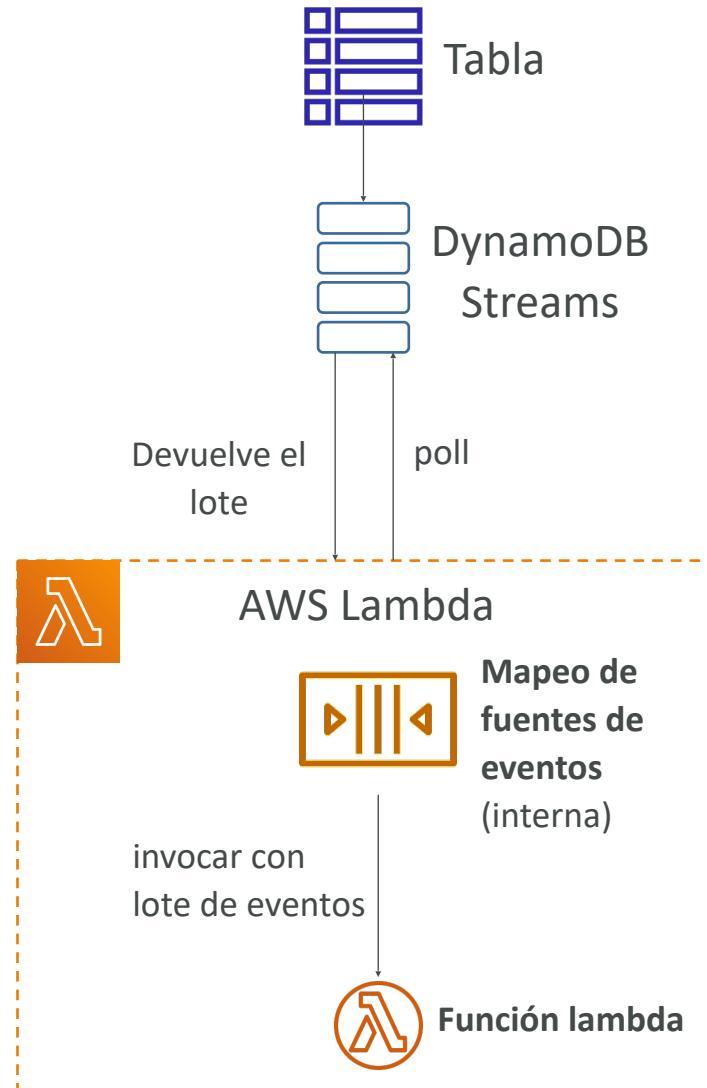


DynamoDB Streams

- Posibilidad de elegir la información que se escribirá en el stream:
 - **KEYS_ONLY** - sólo los atributos clave del elemento modificado
 - **NEW_IMAGE** - el elemento completo, tal y como aparece después de ser modificado
 - **OLD_IMAGE** - el elemento completo, tal y como aparecía antes de ser modificado
 - **NEW_AND_OLD_IMAGES** - tanto la imagen nueva como la antigua del elemento
- DynamoDB Streams se compone de fragmentos, al igual que Kinesis Data Streams
- Tú no aprovisionas los fragmentos, esto lo automatiza AWS
- **Los registros no se rellenan retroactivamente en un stream después de habilitarlo**

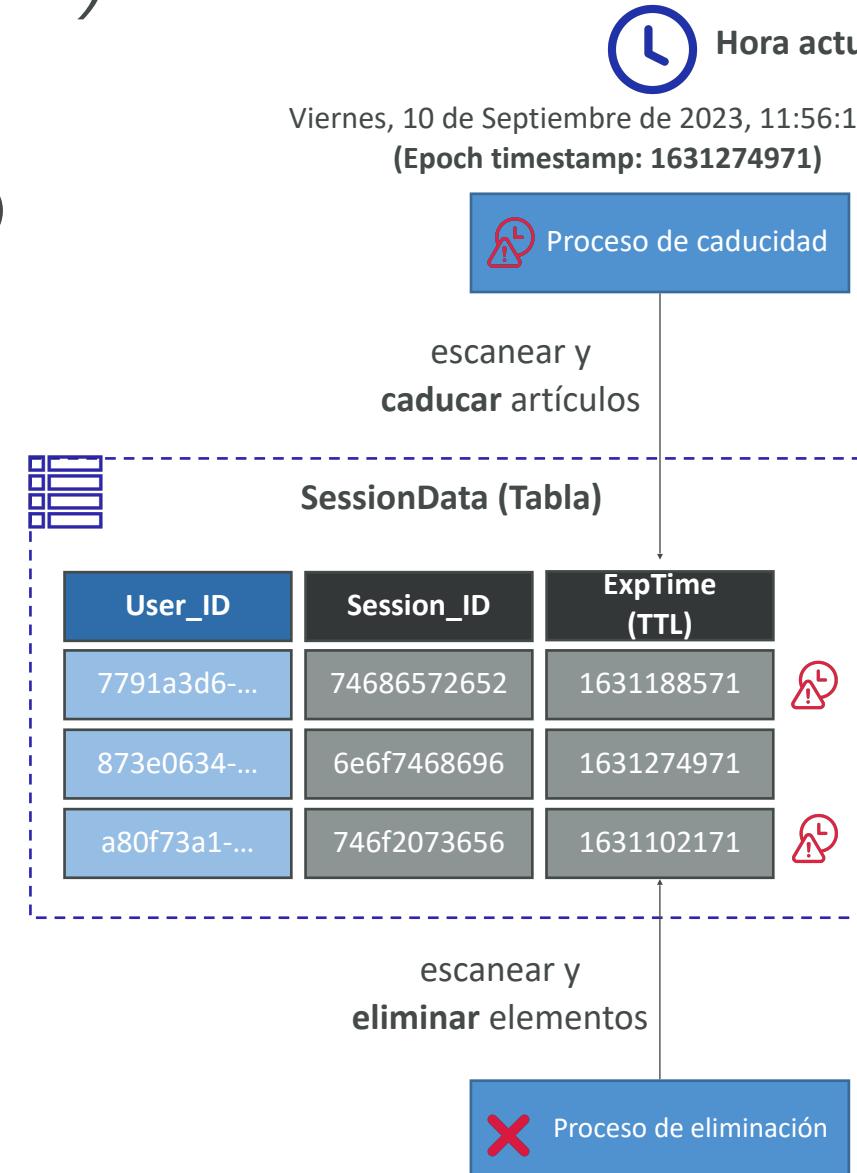
DynamoDB Streams y AWS Lambda

- Necesitas definir un **mapeo de fuente de eventos** para leer de un DynamoDB Streams
- Necesitas asegurarte de que la función Lambda tiene los **permisos adecuados**
- **Tu función Lambda se invoca de forma sincrónica**



DynamoDB – Time To Live (TTL)

- Elimina automáticamente los elementos después de una fecha de caducidad
- No consume ninguna WCU (es decir, no tiene coste adicional)
- El atributo TTL debe ser un tipo de dato **“Number”** con valor **“Unix Epoch timestamp”**
- Los artículos caducados se eliminan en las 48 horas siguientes a su caducidad
- Los elementos caducados, que no se han borrado, aparecen en las lecturas/consultas/escaneos (si no los quieres, filtralos)
- Los artículos caducados se borran tanto de los LSI como de los GSI
- Una operación de eliminación por cada elemento caducado entra en los DynamoDB Streams (puede ayudar a recuperar los elementos caducados)
- Casos de uso: reducir los datos almacenados conservando sólo los elementos actuales, cumplir las obligaciones normativas, ...



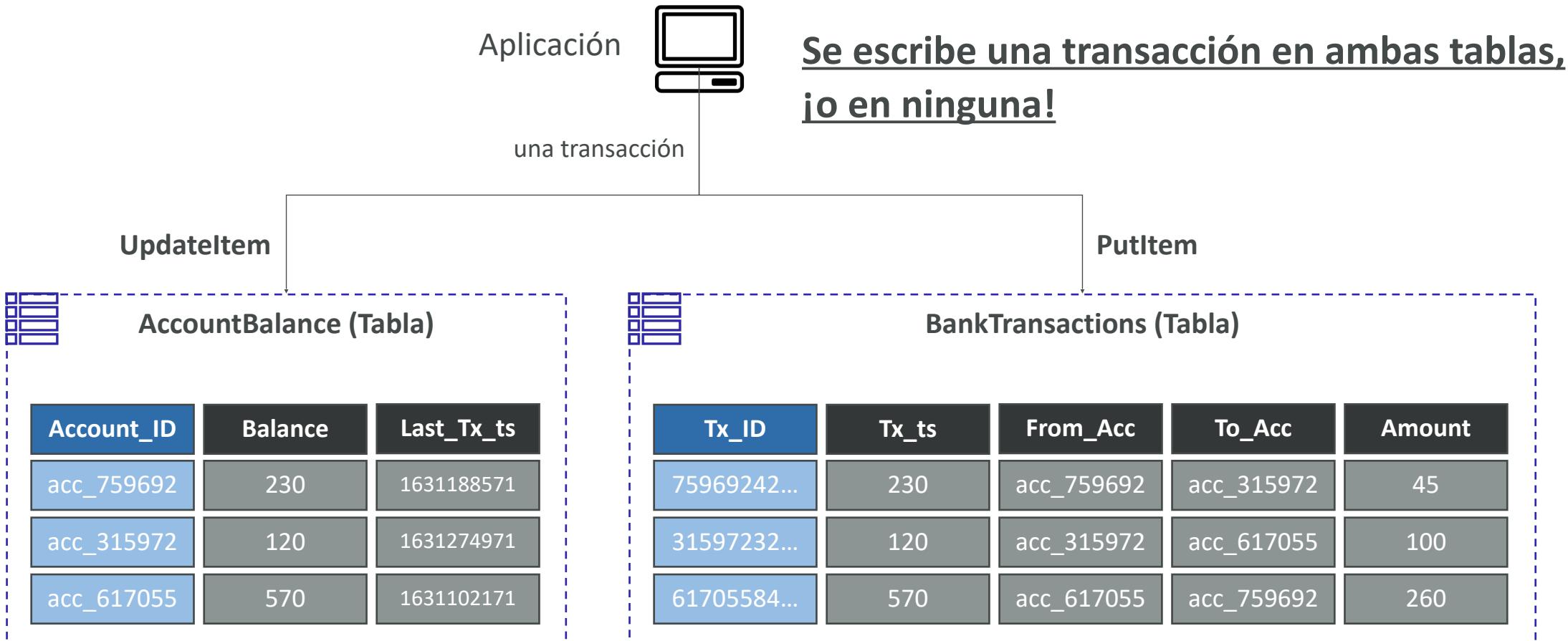
CLI de DynamoDB - Conviene saber

- **--projection-expression**: uno o más atributos a recuperar
- **--filter-expression**: filtra los elementos antes de devolvértelos
- Opciones generales de paginación de la CLI de AWS (por ejemplo, DynamoDB, S3, ...)
 - **--page-size**: especifica que la CLI de AWS recupere la lista completa de elementos, pero con un mayor número de llamadas a la API en lugar de una sola (por defecto: 1000 elementos)
 - **--max-items**: número máximo de elementos a mostrar en la CLI (devuelve **NextToken**)
 - **--starting-token**: especifica el último **NextToken** para recuperar el siguiente conjunto de elementos

Transacciones DynamoDB

- Operaciones coordinadas de todo o nada (añadir/actualizar/eliminar) en varios elementos de una o varias tablas
- Proporciona atomicidad, consistencia, aislamiento y durabilidad (ACID)
- **Modos de lectura** - Consistencia eventual, consistencia fuerte, transaccional
- **Modos de escritura** - Estándar, transaccional
- **Consumo 2x WCUs y RCUs**
 - DynamoDB realiza 2 operaciones por cada elemento (preparar y confirmar)
- Dos operaciones:
 - **TransactGetItems** - una o más operaciones **GetItem**
 - **TransactWriteItems** - una o más operaciones **PutItem**, **UpdateItem** y **DeleteItem**
- Casos de uso: transacciones financieras, gestión de pedidos, juegos multijugador, ...

Transacciones DynamoDB



Transacciones DynamoDB

Cálculos de capacidad

- **¡MUY IMPORTANTE para el examen!**
- **Ejemplo 1:** 3 escrituras transaccionales por segundo, con un tamaño de elemento de 5 KB
 - Necesitamos $3 * \left(\frac{5 \text{ KB}}{1 \text{ KB}} \right) * 2$ (*transactional cost*) = 30 WCUs
- **Ejemplo 2:** 5 lecturas de transacciones por segundo, con un tamaño de elemento de 5 KB
 - Necesitamos $5 * \left(\frac{8 \text{ KB}}{4 \text{ KB}} \right) * 2$ (*transactional cost*) = 20 RCUs
 - (5 se redondea al 4 KB superior)

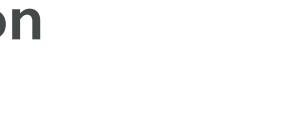
DynamoDB como caché de estado de sesión

- Es habitual utilizar DynamoDB para almacenar estados de sesión
- **vs ElastiCache**
 - ElastiCache es en memoria, pero DynamoDB es sin servidor
 - Ambos son almacenes de claves/valores
- **vs EFS**
 - EFS debe conectarse a las instancias EC2 como una unidad de red
- **vs EBS y Instance Store**
 - EBS e Instance Store sólo pueden utilizarse para el almacenamiento en caché local, no para el almacenamiento en caché compartido
- **vs S3**
 - S3 tiene mayor latencia y no está pensado para objetos pequeños

Fragmentación de escritura en DynamoDB

- Imagina que tenemos una aplicación de votación con dos candidatos, el **candidato A** y el **candidato B**
- Si la **clave de partición** es “**Candidate_ID**”, esto da lugar a dos particiones, lo que generará problemas (por ejemplo, partición en caliente)
- Una estrategia que permite distribuir mejor los elementos uniformemente entre las particiones
- **Añade un sufijo al valor de la clave de partición**
- Dos métodos:
 - Fragmentación mediante sufijo aleatorio
 - Fragmentación mediante sufijo calculado

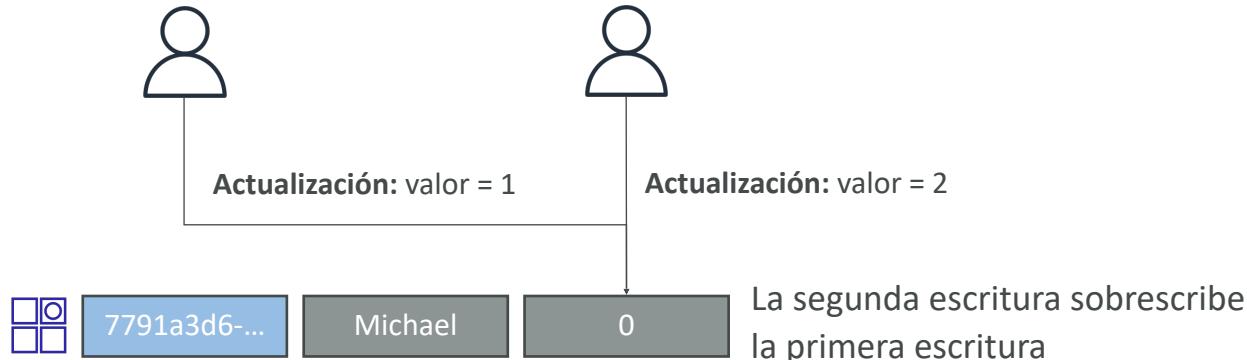
Clave de partición	Clave de ordenación	Atributos
Candidate_ID	Vote_ts	Voter_ID
Candidate_A-11	1631188571	7791
Candidate_B-17	1631274971	8301
Candidate_B-80	1631102171	6750
Candidate_A-20	1631102171	2404



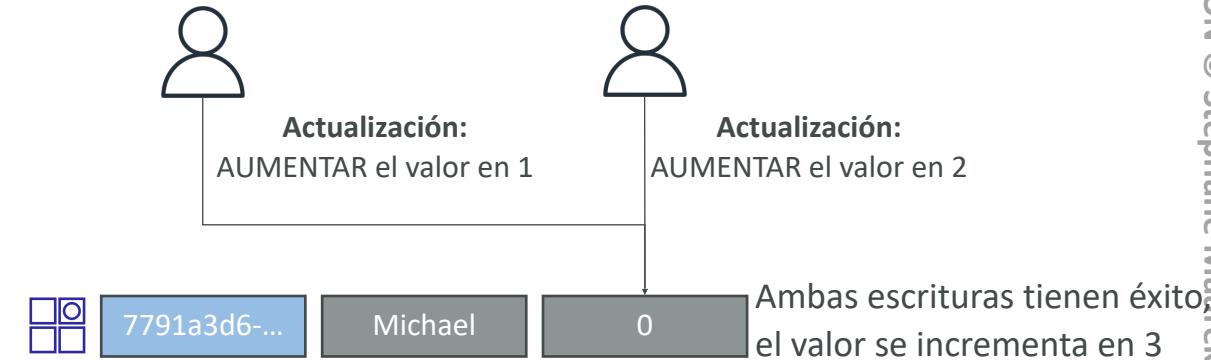
Candidate_ID + Sufijo aleatorio

DynamoDB - Tipos de escritura

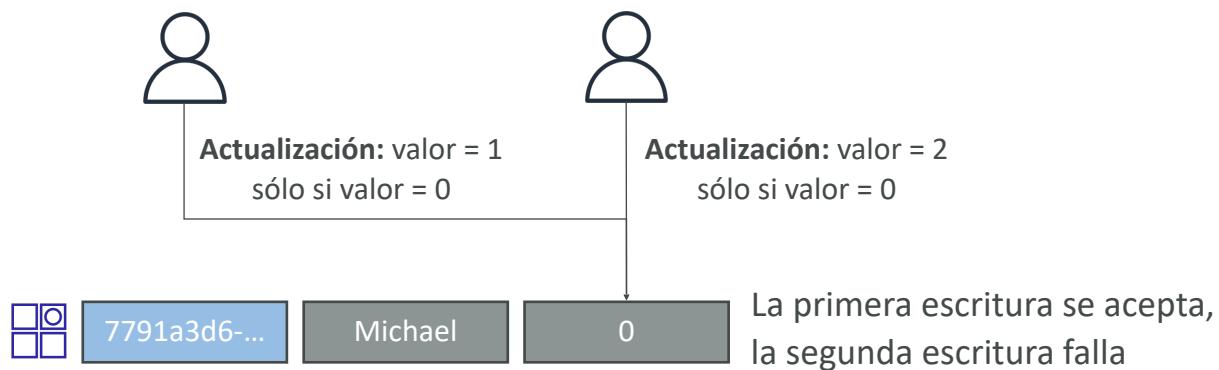
Escrituras concurrentes



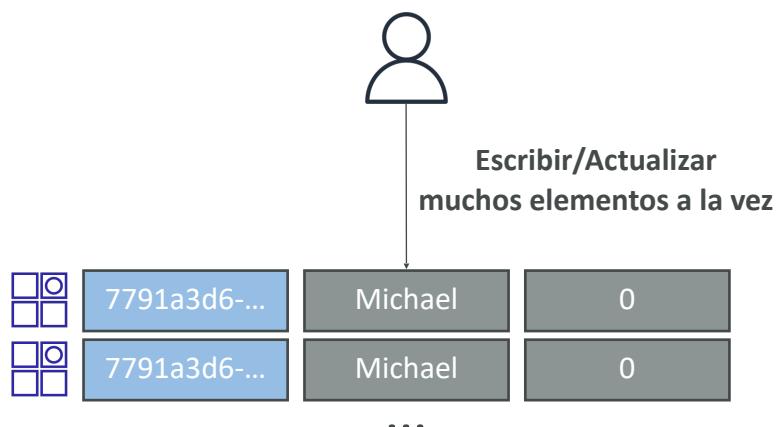
Escrituras atómicas



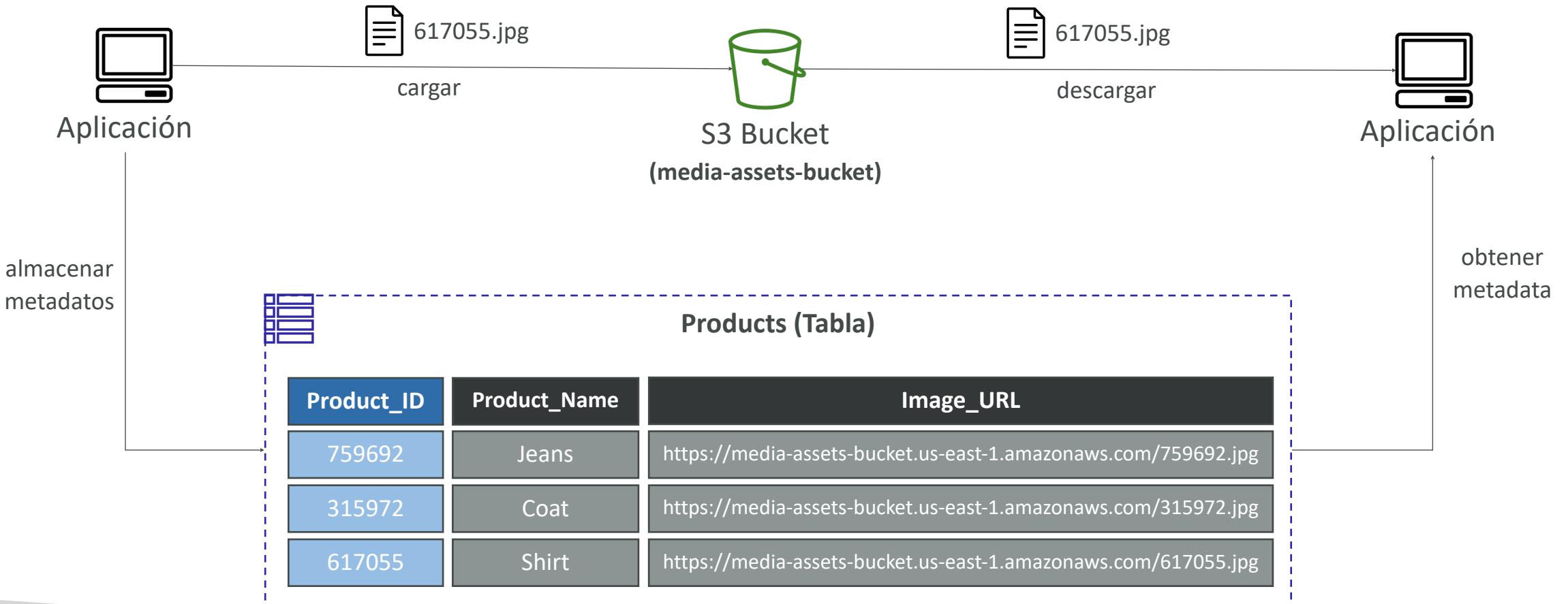
Escrituras condicionales



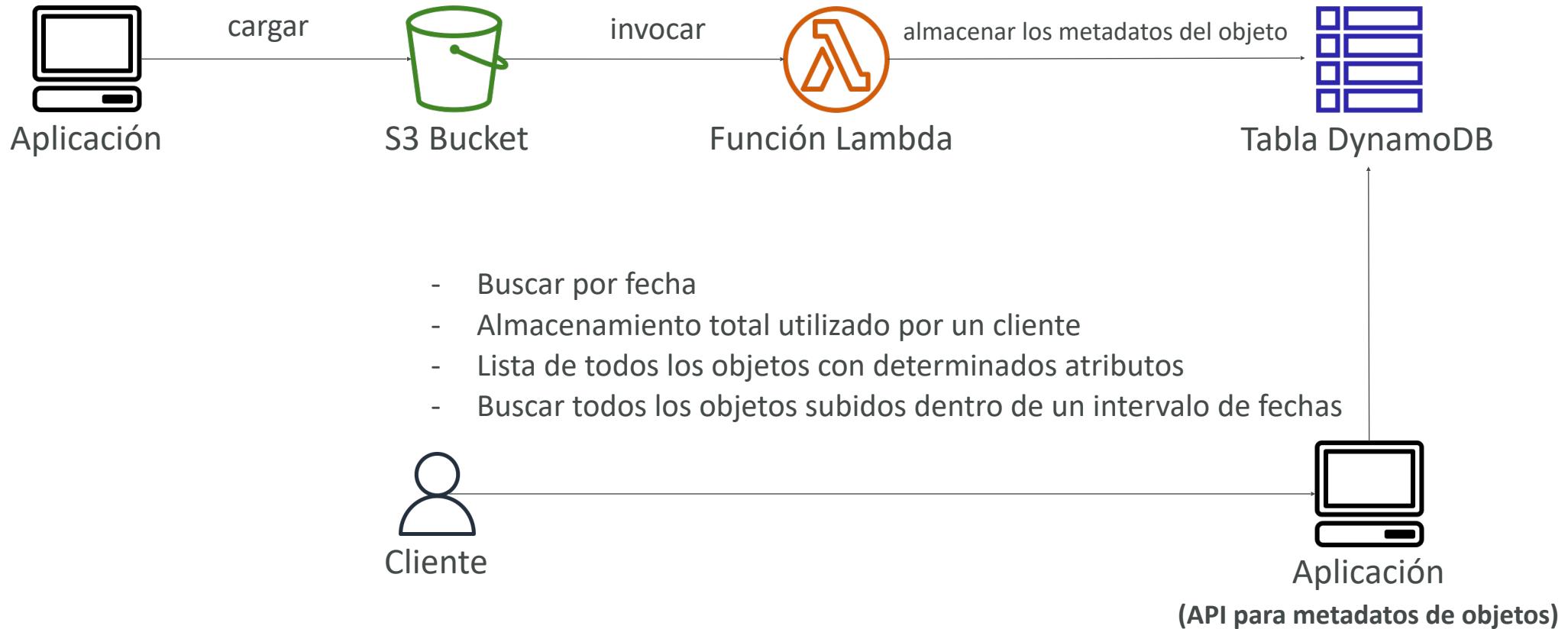
Escrituras por lotes



DynamoDB - Patrón de objetos grandes

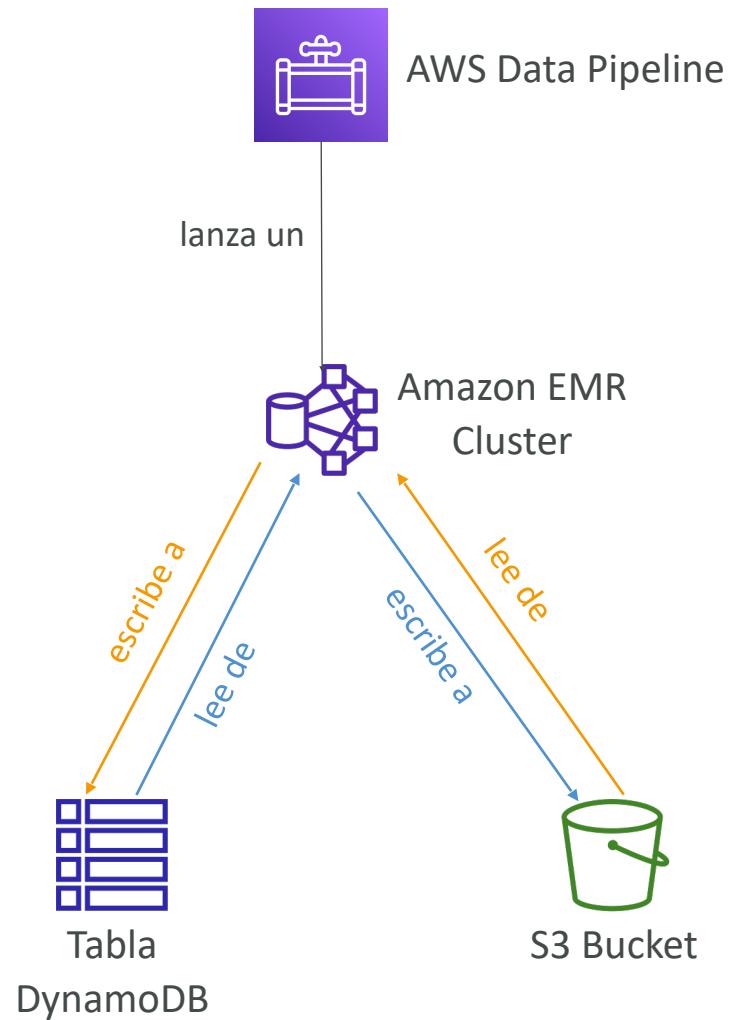


DynamoDB - Indexación de metadatos de objetos S3



Operaciones DynamoDB

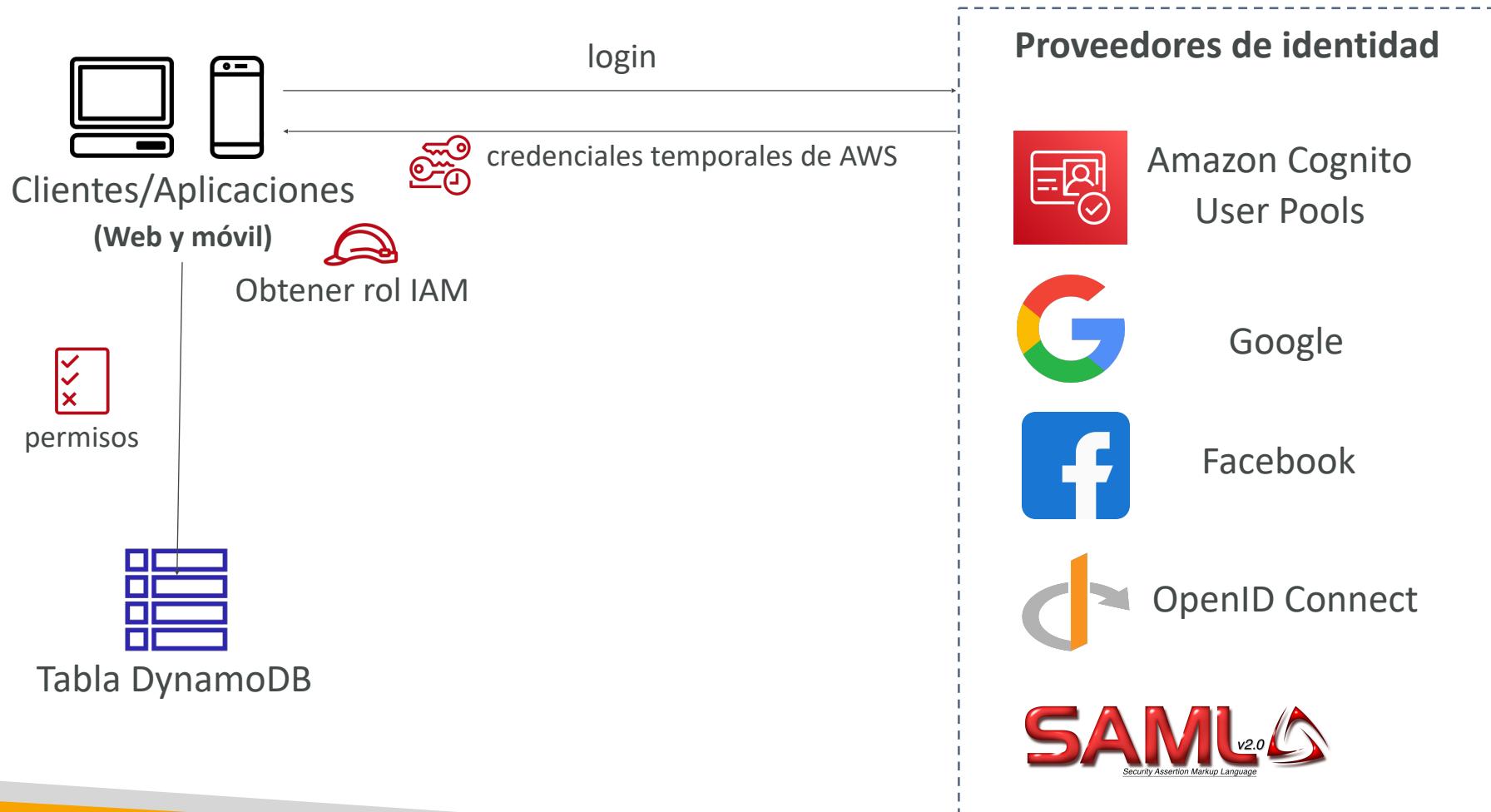
- **Limpieza de tablas**
 - **Opción 1:** Escanear + DeleteItem
 - Muy lento, consume RCU y WCU, caro
 - **Opción 2:** Eliminar Tabla + Recrear Tabla
 - Rápido, eficiente, barato
- **Copiar una tabla DynamoDB**
 - **Opción 1:** Utilizar AWS Data Pipeline
 - **Opción 2:** Copiar y restaurar en una nueva tabla
 - Tarda un tiempo
 - **Opción 3:** Escanear + PutItem o BatchWriteItem
 - Escribe tu propio código



DynamoDB - Seguridad y otras características

- **Seguridad**
 - Endpoints VPC disponibles para acceder a DynamoDB sin utilizar Internet
 - Acceso totalmente controlado por IAM
 - Cifrado en reposo mediante AWS KMS y en tránsito mediante SSL/TLS
- **Función de copia de seguridad y restauración disponible**
 - Recuperación puntual (PITR) como RDS
 - Sin impacto en el rendimiento
- **Tablas globales**
 - Multiregión, multiactiva, totalmente replicada, alto rendimiento
- **DynamoDB Local**
 - Desarrolla y testea aplicaciones localmente sin acceder al servicio web DynamoDB (sin Internet)
 - AWS Database Migration Service (AWS DMS) puede utilizarse para migrar a DynamoDB (desde MongoDB, Oracle, MySQL, S3, ...)

DynamoDB - Los usuarios interactúan directamente con DynamoDB



DynamoDB - Control de acceso detallado

- Utilizando **Web Identity Federation** o **Cognito Identity Pools**, cada usuario obtiene credenciales de AWS
- Puedes asignar un rol IAM a estos usuarios con una **condición** para limitar su acceso API a DynamoDB
- **LeadingKeys** - limitar el acceso a nivel de fila para los usuarios en la clave primaria
- **Atributos** - limita los atributos específicos que el usuario puede ver

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Effect": "Allow",  
            "Action": [  
                "dynamodb:GetItem", "dynamodb:BatchGetItem", "dynamodb:Query",  
                "dynamodb:PutItem", "dynamodb:UpdateItem", "dynamodb:DeleteItem",  
                "dynamodb:BatchWriteItem"  
            ],  
            "Resource": "arn:aws:dynamodb:us-west-2:123456789012:table/MyTable",  
            "Condition": {  
                "ForAllValues:StringEquals": {  
                    "dynamodb:LeadingKeys": ["${cognito-identity.amazonaws.com:sub}"]  
                }  
            }  
        }  
    ]  
}
```

Más información en: <https://docs.aws.amazon.com/amazondynamodb/latest/developerguide/specifying-conditions.html>

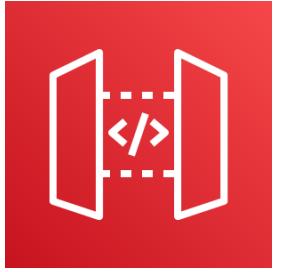
API Gateway

Construir, desplegar y gestionar APIs

Ejemplo: Creación de una API sin servidor



AWS API Gateway



- AWS Lambda + API Gateway: Sin infraestructura que administrar
- Compatibilidad con el protocolo WebSocket
- Gestión de versiones de API (v1, v2...)
- Gestión de diferentes entornos (desarrollo, test, producción...)
- Gestión de la seguridad (autenticación y autorización)
- Creación de claves de API, gestión de la limitación de solicitudes
- Importación de Swagger / Open API para definir la API rápidamente
- Transformación y validación de solicitudes y respuestas
- Generación de SDK y especificaciones de API
- Almacenamiento en caché de respuestas de API

API Gateway - Integraciones de alto nivel

- **Función Lambda**

- Invocar función Lambda
- Manera sencilla de exponer la API REST respaldada por AWS Lambda

- **HTTP**

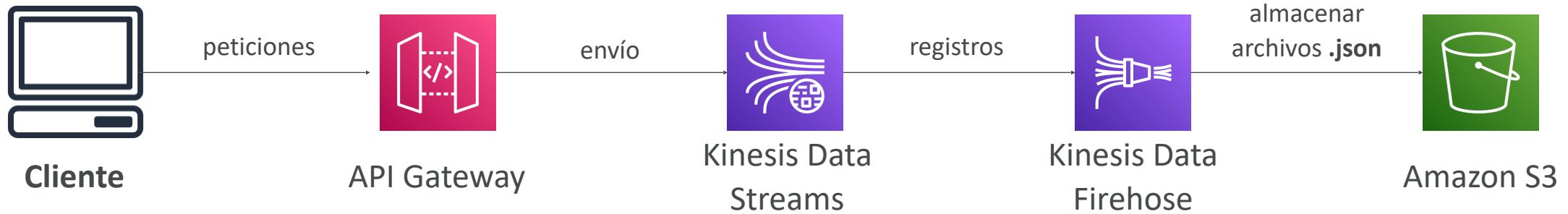
- Exponer puntos de enlace HTTP en el backend
- ¿Por qué? Añadir limitación de velocidad, almacenamiento en caché, autenticaciones de usuario, claves API, etc...

- **Servicio AWS**

- Exponer cualquier API de AWS a través de API Gateway
- Ejemplo: iniciar un flujo de trabajo AWS Step Function, enviar un mensaje a SQS
- ¿Por qué? Añadir autenticación, desplegar públicamente, ...

Ejemplo de API Gateway

Integración de servicios de AWS Kinesis Data Streams



API Gateway - Tipos de endpoints

- **Edge-Optimized (por defecto):** Para clientes globales
 - Las solicitudes se enrutan a través de las ubicaciones de CloudFront Edge (mejora la latencia).
 - API Gateway sigue viviendo en una sola región
- **Regional:**
 - Para clientes dentro de la misma región
 - Podría combinarse manualmente con CloudFront (más control sobre las estrategias de almacenamiento en caché y la distribución)
- **Privada:**
 - Solo se puede acceder desde tu VPC utilizando un punto final de VPC de interfaz (ENI)
 - Utiliza una política de recursos para definir el acceso

API Gateway – Seguridad

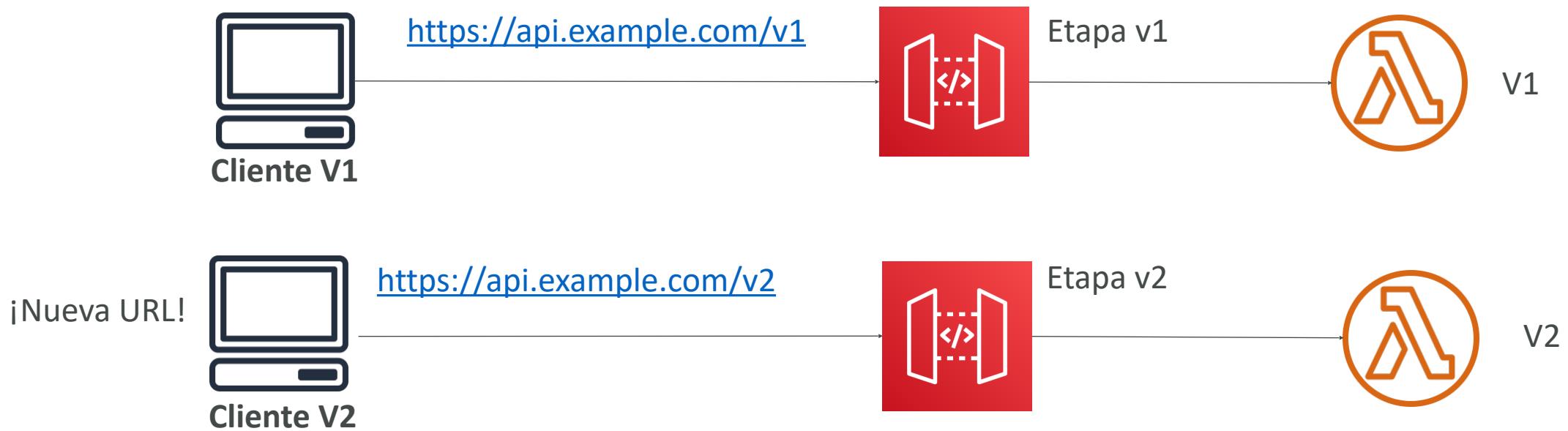
- **Autenticación de usuarios mediante**
 - Roles IAM (útil para aplicaciones internas)
 - Cognito (identidad para usuarios externos - ejemplo usuarios móviles)
 - Autorizador personalizado (tu propia lógica)
- **Seguridad HTTPS de nombre de dominio** personalizado a través de la integración con AWS Certificate Manager (ACM)
 - Si utilizas el punto de enlace Edge-Optimized, el certificado debe estar en **us-east-1**
 - Si utilizas el punto de enlace regional, el certificado debe estar en la región de API Gateway
 - Debes configurar el registro CNAME o A-alias en Route 53

API Gateway – Etapas de despliegue

- Hacer cambios en la API Gateway no significa que sean efectivos
- Necesitas hacer un "despliegue" para que tengan efecto
- Es una fuente habitual de confusión
- Los cambios se despliegan en "Etapas" (tantas como quieras)
- Utiliza la nomenclatura que quieras para las etapas (dev, test, prod)
- Cada etapa tiene sus propios parámetros de configuración
- Las etapas se pueden volver atrás, ya que se guarda un historial de despliegues

API Gateway – Etapas v1 y v2

Cambio importante en la API

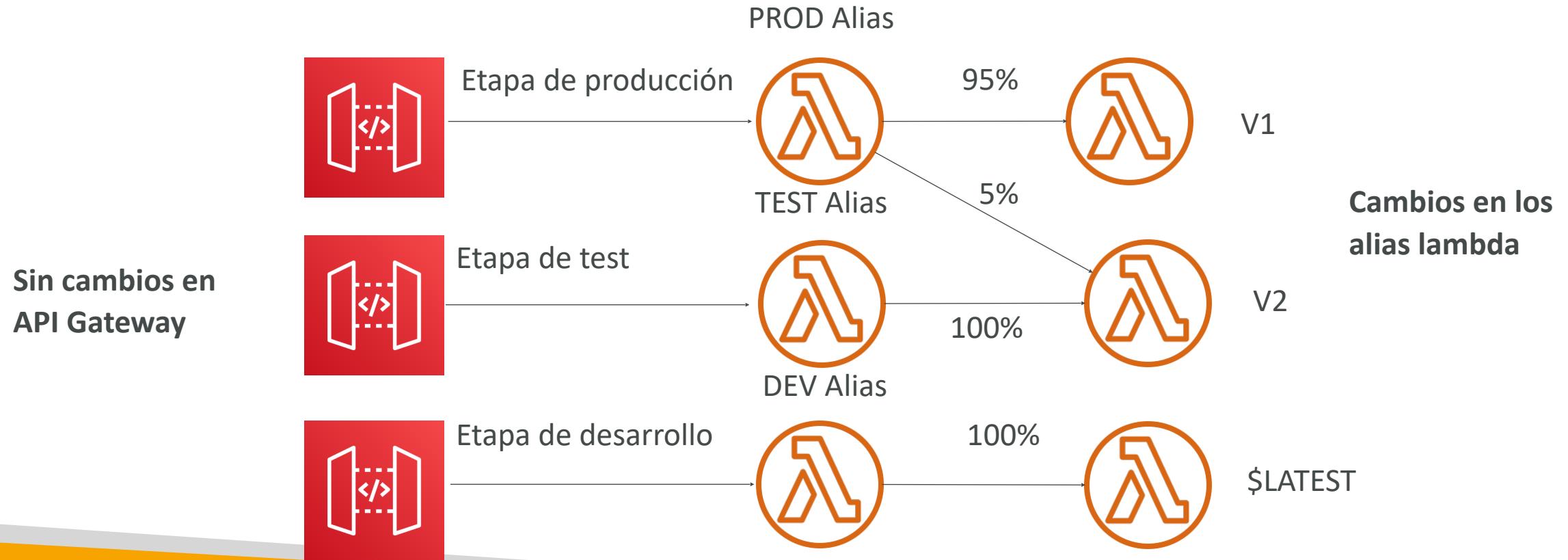


API Gateway – Variables de etapa

- Las variables de etapa son como variables de entorno para API Gateway
- Utilízalas para modificar valores de configuración que cambian a menudo
- Se pueden utilizar en:
 - ARN de función Lambda
 - Endpoint HTTP
 - Plantillas de mapeo de parámetros
- Casos de uso:
 - Configurar endpoints HTTP con los que hablan tus etapas (dev, test, prod...)
 - Pasa parámetros de configuración a AWS Lambda mediante plantillas de mapeo
- Las variables de etapa se pasan al objeto "contexto" en AWS Lambda
- Formato: \${stageVariables.variableName}

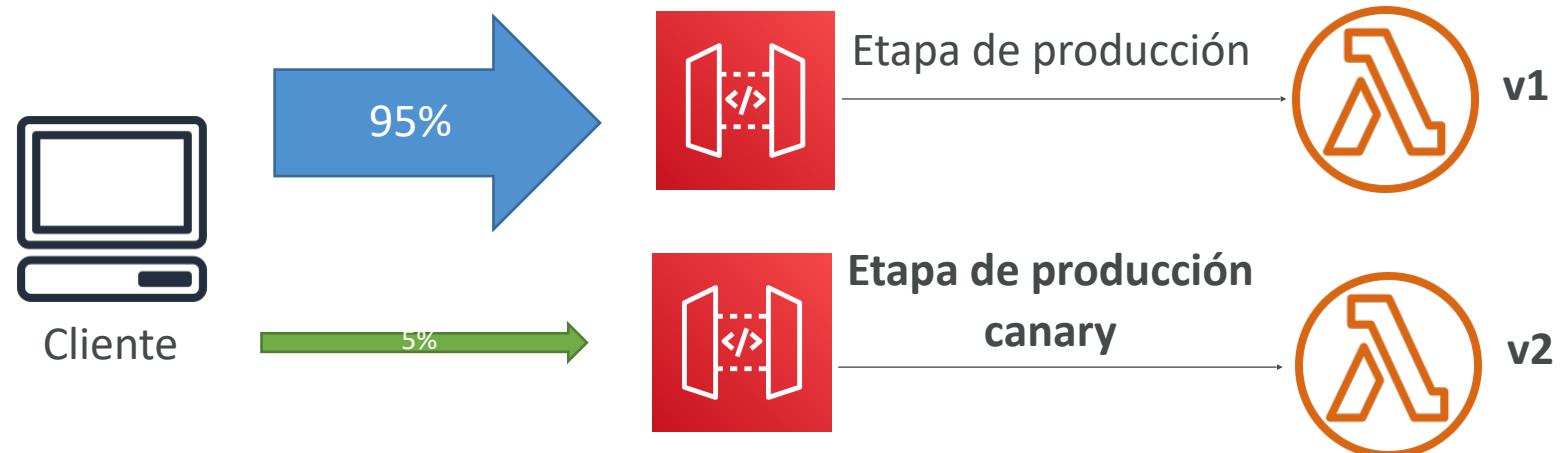
Variables de etapa y alias Lambda de API Gateway

- Creamos una **variable de etapa** para indicar el alias Lambda correspondiente
- ¡Nuestra API Gateway invocará automáticamente la función Lambda correcta!



API Gateway - Despliegue Canary

- Posibilidad de habilitar despliegues canary para cualquier etapa (normalmente prod)
- Elige el % de tráfico que recibe el canal canary



- Las métricas y los logs están separados (para una mejor monitorización)
- Posibilidad de anular variables de etapa para el canary
- Esta es una implementación blue / green con AWS Lambda y API Gateway

API Gateway - Tipos de integración

- Tipo de integración **MOCK**
 - API Gateway devuelve una respuesta sin enviar la petición al backend
- Tipo de integración **HTTP / AWS (servicios Lambda y AWS)**
 - Debes configurar tanto la petición como la respuesta de integración
 - Configura el mapeo de datos utilizando **plantillas de mapeo** para la petición y la respuesta



API Gateway - Tipos de integración

- Tipo de integración **AWS_PROXY (Proxy Lambda)**:
 - La petición entrante del cliente es la entrada a Lambda
 - La función es responsable de la lógica de petición / respuesta
 - **No se pasan como argumentos plantillas de mapeo, cabeceras, parámetros de cadenas de consulta...**

```
{  
  "resource": "Resource path",  
  "path": "Path parameter",  
  "httpMethod": "Incoming request's method name",  
  "headers": "String containing incoming request headers",  
  "multiValueHeaders": "List of strings containing incomin  
  "queryStringParameters": "query string parameters ",  
  "multiValueQueryStringParameters": "List of query string  
  "pathParameters": "path parameters",  
  "stageVariables": "Applicable stage variables",  
  "requestContext": "Request context, including authorizer  
  "body": "A JSON string of the request payload.",  
  "isBase64Encoded": "A boolean flag"  
}
```

Payload de la invocación a la función lambda

```
{  
  "isBase64Encoded": "true|false",  
  "statusCode": "httpStatusCode",  
  "headers": { "headerName": "headerValue", ... },  
  "multiValueHeaders": { "headerName": ["headerValue", "headerValue"] },  
  "body": "..."  
}
```

Respuesta esperada de la función lambda

API Gateway - Tipos de integración

- Tipo de integración **HTTP_PROXY**
 - Sin plantilla de mapeo
 - La petición HTTP se pasa al backend
 - La respuesta HTTP del backend es reenviada por API Gateway
 - Posibilidad de añadir cabeceras HTTP si es necesario (ej: clave API)



Plantillas de mapeo (integración AWS y HTTP)

- Las plantillas de mapeo se pueden utilizar para modificar peticiones / respuestas
- Renombrar / Modificar los **parámetros de la cadena de consulta**
- Modificar el **contenido del cuerpo**
- **Añadir cabeceras**
- Utiliza el Velocity Template Language (VTL): bucle for, if etc...
- Filtra los resultados de salida (elimina datos innecesarios)
- El tipo de contenido se puede establecer en **application/json** o **application/xml**

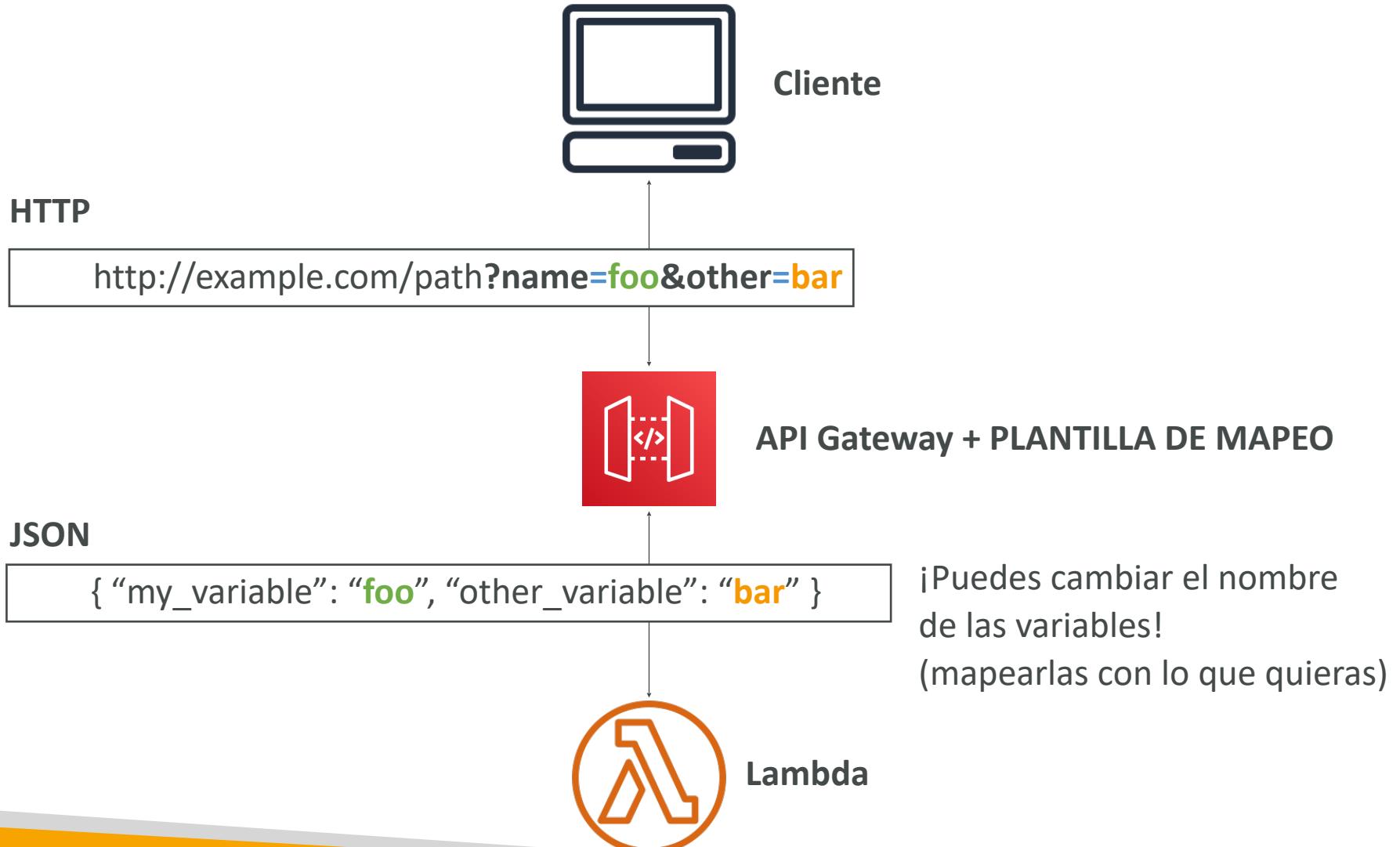
Ejemplo de mapeo: JSON a XML con SOAP

- Las API SOAP se basan en XML, mientras que las API REST se basan en JSON



- En este caso, API Gateway debería:
 - Extraer datos de la petición: ya sea la ruta, el payload o la cabecera
 - Construir un mensaje SOAP basado en los datos de la petición (plantilla de mapeo)
 - Llamar al servicio SOAP y recibir la respuesta XML
 - Transformar la respuesta XML al formato deseado (como JSON), y responder al usuario

Ejemplo de mapeo Parámetros de la cadena de consulta



API Gateway - Open API spec

- Forma común de definir las API REST, utilizando la definición de la API como código
- Importar la especificación OpenAPI 3.0 existente a API Gateway
 - Método
 - Petición de método
 - Petición de integración
 - Método de respuesta
 - + Extensiones de AWS para API Gateway y configurar cada una de las opciones
- Puedes exportar la API actual como OpenAPI spec
- Las OpenAPI spec se pueden escribir en YAML o JSON
- Usando OpenAPI podemos generar SDK para nuestras aplicaciones



API REST - Validación de peticiones

- Puedes configurar API Gateway para que realice una validación básica de una petición API antes de proceder con la petición de integración
- Cuando falla la validación, API Gateway falla inmediatamente la petición
 - Devuelve una respuesta de error 400
- Esto reduce las llamadas innecesarias al backend
- Comprueba:
 - Los parámetros de petición requeridos en el URI, la cadena de consulta y las cabeceras de una petición entrante están incluidos y no están en blanco
 - El Payload de la petición aplicable se vincula al modelo de petición JSON Schema configurado del método

REST API – RequestValidation – OpenAPI

- Configurar la validación de peticiones importando el archivo de definiciones OpenAPI

```
{
  "openapi": "3.0.0",
  "info": {
    "title": "ReqValidation Sample",
    "version": "1.0.0"
  },
  "servers": [ ... ],
  "x-amazon-apigateway-request-validation": {
    "all": {
      "validateRequestBody": true,
      "validateRequestParameters": true
    },
    "params-only": {
      "validateRequestBody": false,
      "validateRequestParameters": true
    }
  }
}
```

Define los validadores (validators)

```
{
  "openapi": "3.0.0",
  "info": {
    "title": "ReqValidation Sample",
    "version": "1.0.0"
  },
  "servers": [ ... ],
  "x-amazon-apigateway-request-validator": "params-only",
  ...
}

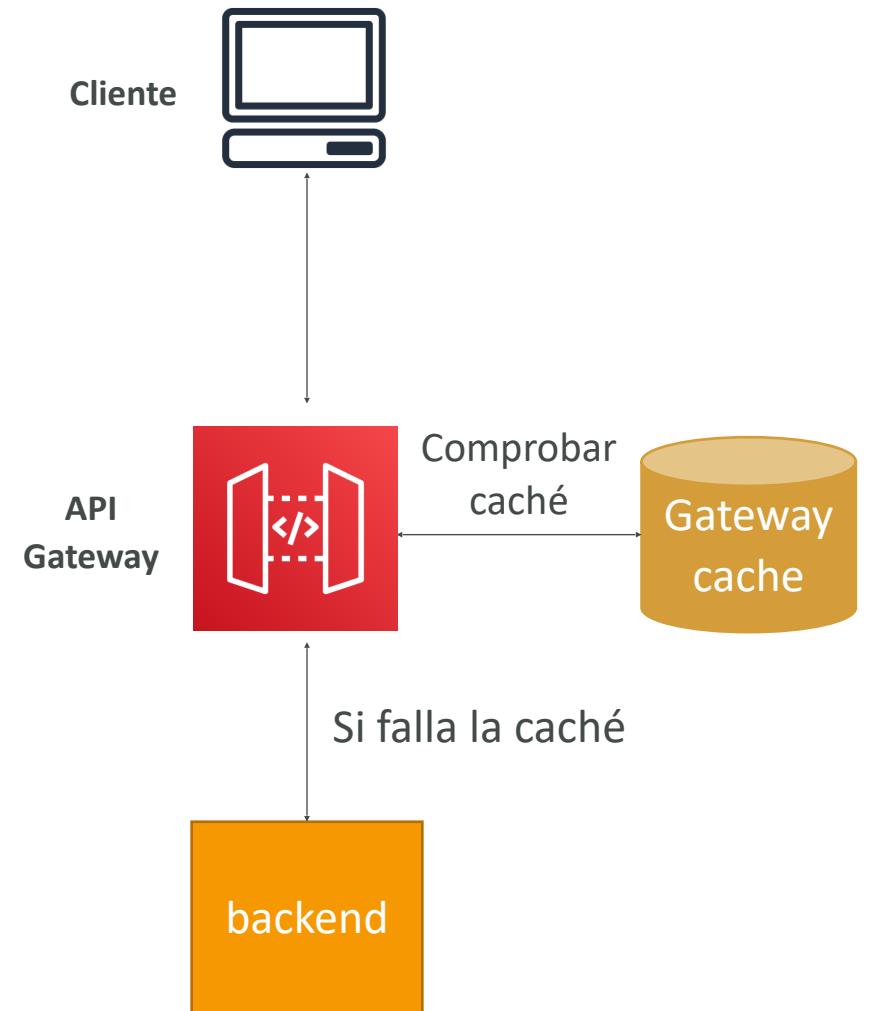
{
  "openapi": "3.0.0",
  "info": {
    "title": "ReqValidation Sample",
    "version": "1.0.0"
  },
  "servers": [ ... ],
  "paths": {
    "/validation": {
      "post": {
        "x-amazon-apigateway-request-validator": "all"
      }
    }
  },
  ...
}
```

Activar validador de **params-only en todos los métodos de la API**

Activar todos (all) los validadores en el método POST /validation (anula el validador de **params-only heredado de la API)**

Almacenamiento en caché de las respuestas de la API

- El almacenamiento en caché reduce el número de llamadas realizadas al backend
- El **TTL** (tiempo de vida) por defecto es de 300 segundos (min: 0s, max: 3600s)
- **Las cachés se definen por etapa**
- Es posible anular la configuración de la caché **por método**
- Opción de cifrado de la caché
- Capacidad de caché entre 0,5GB y 237GB
- La caché es cara, tiene sentido en prod, puede no tenerlo en dev / test



Invalidación de la caché de la API Gateway

- Capaz de vaciar toda la caché (invalidarla) inmediatamente
- Los clientes pueden invalidar la caché con la **cabecera: Cache-Control: max-age=0** (con la debida autorización de IAM)
- Si no impones una política InvalidateCache (o eliges la casilla de requerir autorización en la consola), cualquier cliente puede invalidar la caché de la API

```
{  
  "Version": "2012-10-17",  
  "Statement": [  
    {  
      "Effect": "Allow",  
      "Action": [  
        "execute-api:InvalidateCache"  
      ],  
      "Resource": [  
        "arn:...:api-id/stage-name/GET/resource-path-specifier"  
      ]  
    }  
  ]  
}
```

API Gateway - Planes de uso y claves API

- Si quieres poner una API a disposición de tus clientes como oferta (\$)
- **Plan de uso:**
 - Quién puede acceder a una o más etapas y métodos de la API desplegada
 - Cuánto y a qué velocidad pueden acceder
 - Utiliza claves API para identificar a los clientes de la API y medir el acceso
 - Configura los límites de estrangulamiento y de cuota que se aplican a cada cliente individual
- **Claves API / API Keys:**
 - Valores de strings alfanuméricos para distribuir entre tus clientes
 - Ej: WBjHxNtoAb4WPKBC7cGm64CBibIb24b4jt8jJHo9
 - Se puede utilizar con planes de uso para controlar el acceso
 - Los límites de estrangulamiento se aplican a las claves API
 - El límite de cuotas es el número total de peticiones máximas

API Gateway - Orden correcto para las claves API

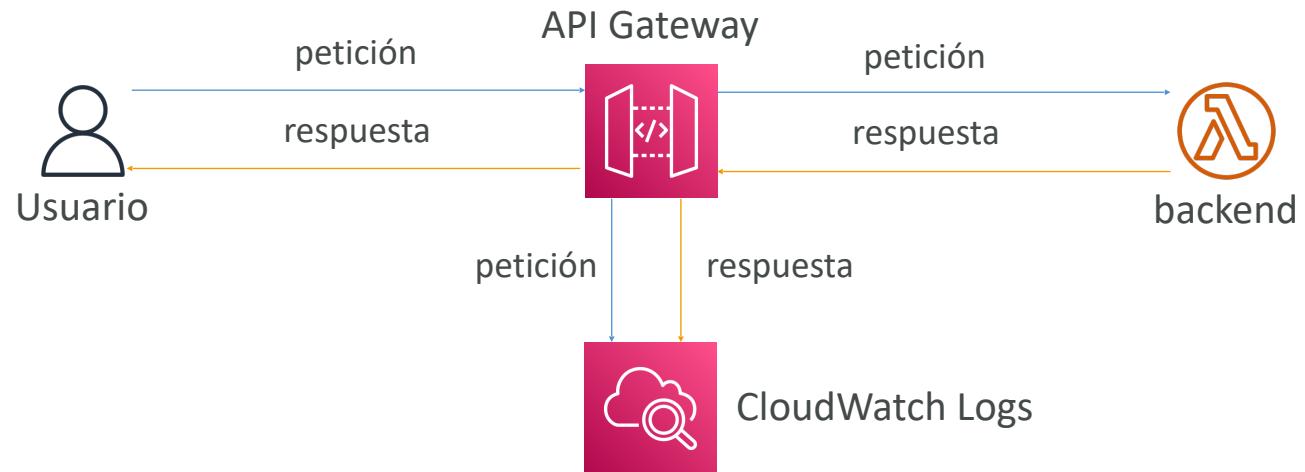
- **Para configurar un plan de uso**

1. Crea una o varias API, configura los métodos para que requieran una clave de API y despliega las API en las etapas.
 2. Genera o importa claves API para distribuirlas a los desarrolladores de aplicaciones (tus clientes) que vayan a utilizar tu API.
 3. Crea el plan de uso con los límites de aceleración y cuota deseados.
 4. Asocia las etapas API y las claves API al plan de uso.
-
- Quienes llamen a la API deben proporcionar una clave de API asignada en la cabecera x-api-key en las peticiones a la API.

API Gateway - Logs y rastreo

- **Logs de CloudWatch**

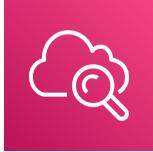
- Los logs contienen información sobre el cuerpo de la petición/respuesta
- Habilita los logs de CloudWatch en el nivel de etapa (con Log Level - ERROR, DEBUG, INFO)
- Puedes anular la configuración por API



- **X-Ray**

- Habilita el rastreo para obtener información adicional sobre las peticiones en API Gateway
- X-Ray API Gateway + AWS Lambda te ofrece la imagen completa

API Gateway - Métricas de CloudWatch



- Las métricas son por etapa, posibilidad de activar métricas detalladas
- **CacheHitCount** y **CacheMissCount**: eficacia de la caché
- **Count**: Número total de peticiones a la API en un periodo determinado.
- **IntegrationLatency**: El tiempo transcurrido entre el momento en que API Gateway transmite una petición al backend y el momento en que recibe una respuesta del backend.
- **Latency**: El tiempo que transcurre entre que API Gateway recibe una petición de un cliente y le devuelve una respuesta. La latencia incluye la latencia de integración y otros gastos generales de la API Gateway.
- **4XXError** (del lado del cliente) y **5XXError** (del lado del servidor)

Estrangulamiento / Limitación de API Gateway

- **Límite de la cuenta**
 - API Gateway limita las peticiones a 10000 peticiones en todas las API
 - Límite suave que puede aumentarse previa petición
- En caso de “estrangulamiento / limitación” => **429 demasiadas peticiones**
- Puedes establecer **límites de etapa y de método** para mejorar el rendimiento
- O puedes definir **planes de uso** para limitar por cliente
- **Al igual que la concurrencia Lambda, una API que esté sobrecargada, si no se limita, puede hacer que las demás API se “estrangulen”**

API Gateway - Errores

- **4xx significa errores del cliente**
 - 400: Petición errónea
 - 403: Acceso denegado, WAF filtrado
 - 429: Cuota superada, Throttle (estrangulamiento)
- **5xx significa errores del servidor**
 - 502: Excepción de Gateway incorrecto, normalmente por una salida incompatible devuelta desde un backend de integración proxy Lambda y ocasionalmente por invocaciones fuera de orden debido a cargas pesadas.
 - 503: Excepción de servicio no disponible
 - 504: Fallo de integración - ej: Endpoint Request Timed-out Exception
 - Las peticiones de **API Gateway** **expiran después de un máximo de 29 segundos**

AWS API Gateway - CORS

- CORS debe estar activado cuando recibas llamadas a la API desde otro dominio.
- La petición previa OPTIONS debe contener las siguientes cabeceras:
 - Access-Control-Allow-Methods
 - Access-Control-Allow-Headers
 - Access-Control-Allow-Origin
- CORS puede activarse a través de la consola

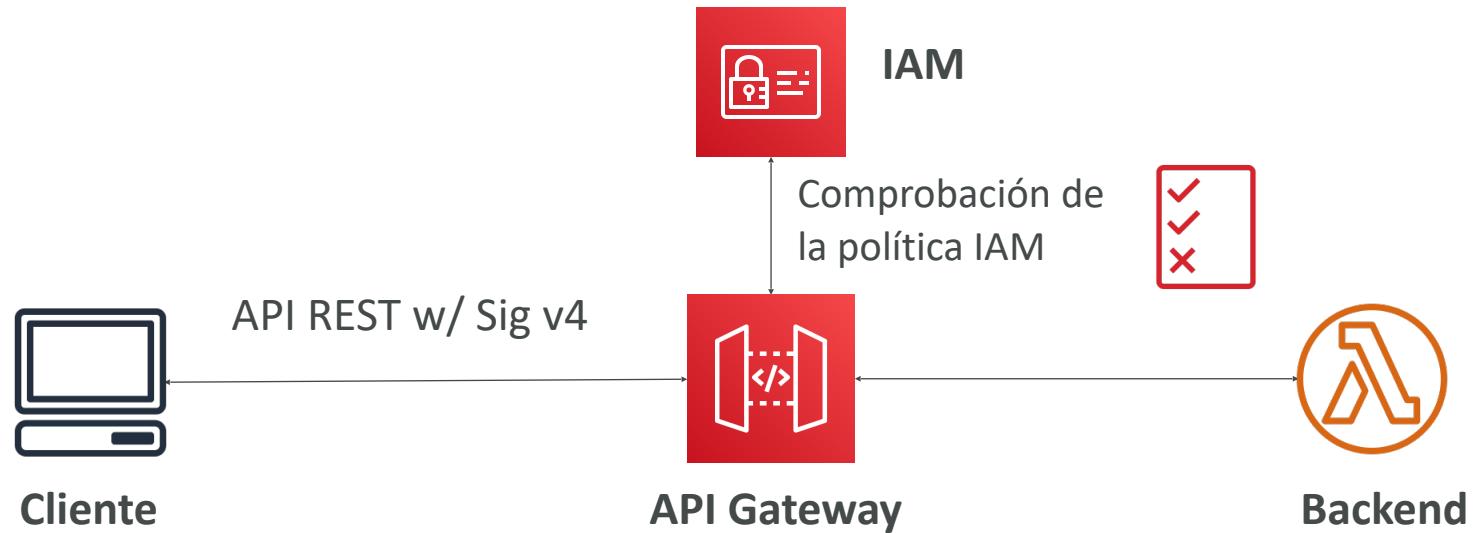
CORS - Activado en la API Gateway



API Gateway - Seguridad

Permisos IAM

- Crea una autorización de política IAM y adjúntala a un usuario / rol
- **Autenticación = IAM** | **Autorización = Política IAM**
- Es bueno para proporcionar acceso dentro de AWS (EC2, Lambda, usuarios IAM ...)
- Aprovecha la capacidad "Sig v4" donde las credenciales IAM están en las cabeceras



API Gateway - Políticas de recursos

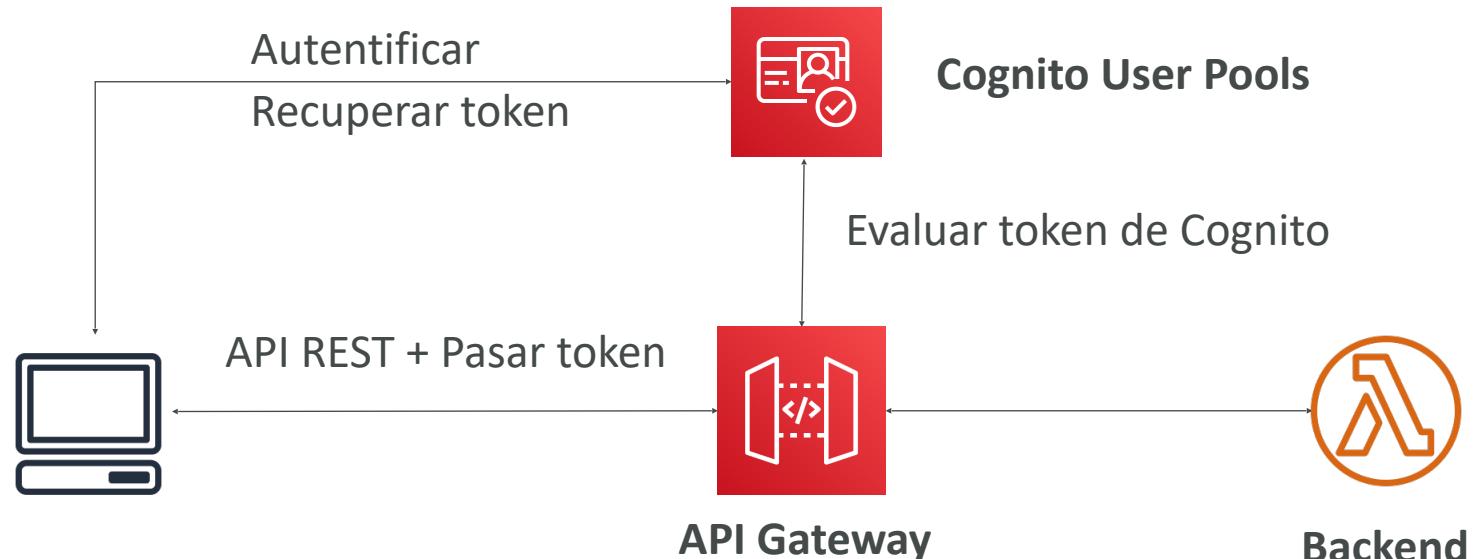
- **Políticas de recursos**
(similares a la política de recursos Lambda)
- **Permitir el acceso entre cuentas (combinado con la seguridad IAM)**
- Permitir una dirección IP de origen específica
- Permitir un endpoint VPC

```
{  
  "Version": "2012-10-17",  
  "Statement": [  
    {  
      "Effect": "Allow",  
      "Principal": {  
        "AWS": [  
          "arn:aws:iam::account-id-2:user/Alice",  
          "account-id-2"  
        ]  
      },  
      "Action": "execute-api:Invoke",  
      "Resource": [  
        "arn:aws:execute-api:region:account-id-1:api-id/stage/GET/pets"  
      ]  
    }  
  ]  
}
```

API Gateway - Seguridad

Cognito User Pools

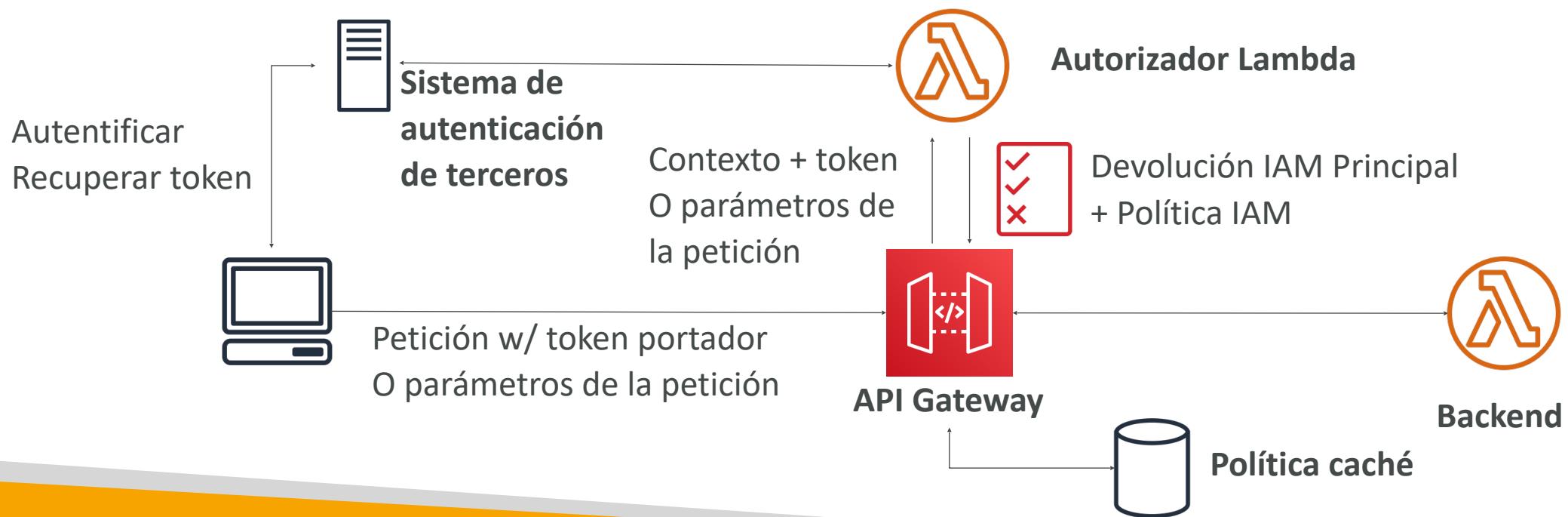
- Cognito gestiona completamente el ciclo de vida del usuario, el token caduca automáticamente
- API Gateway verifica la identidad automáticamente desde AWS Cognito
- No requiere implementación personalizada
- **Autenticación = Cognito User Pools** | **Autorización = Métodos API Gateway**



API Gateway – Seguridad

Autorizador Lambda (antes autorizadores personalizados)

- **Autorizador basado en token** (token portador / bearer) - ej JWT (JSON Web Token) u Oauth
- Un autorizador Lambda **basado en parámetros de petición** (cabeceras, query string, stage var)
- Lambda debe devolver una política IAM para el usuario, la política resultante se almacena en caché
- **Autenticación = Externa** | **Autorización = Función Lambda**



API Gateway - Seguridad - Resumen

- **IAM:**

- Ideal para usuarios / roles que ya están dentro de tu cuenta AWS, + **política de recursos para cuentas cruzadas**
- Gestiona la autenticación + autorización
- Aprovecha la Firma v4

- **Cognito User Pools:**

- Gestiona tu propio pool de usuarios (puede estar respaldado por Facebook, Google login, etc...)
- No necesitas escribir código personalizado
- Debes implementar la autorización en el backend

- **Autorizador Lambda (autorizador personalizado):**

- Ideal para tokens de terceros
- Muy flexible en cuanto a qué política IAM se devuelve
- Gestiona la verificación de autenticación + autorización en la función Lambda
- Paga por invocación Lambda, los resultados se almacenan en caché

API Gateway – HTTP API vs REST API

• APIs HTTP

- Proxy AWS Lambda de baja latencia y rentable, API de proxy HTTP e integración privada (sin mapeo de datos)
- Soporte para autorización OIDC y OAuth 2.0, y soporte integrado para CORS
- Sin planes de uso ni claves API

• API REST

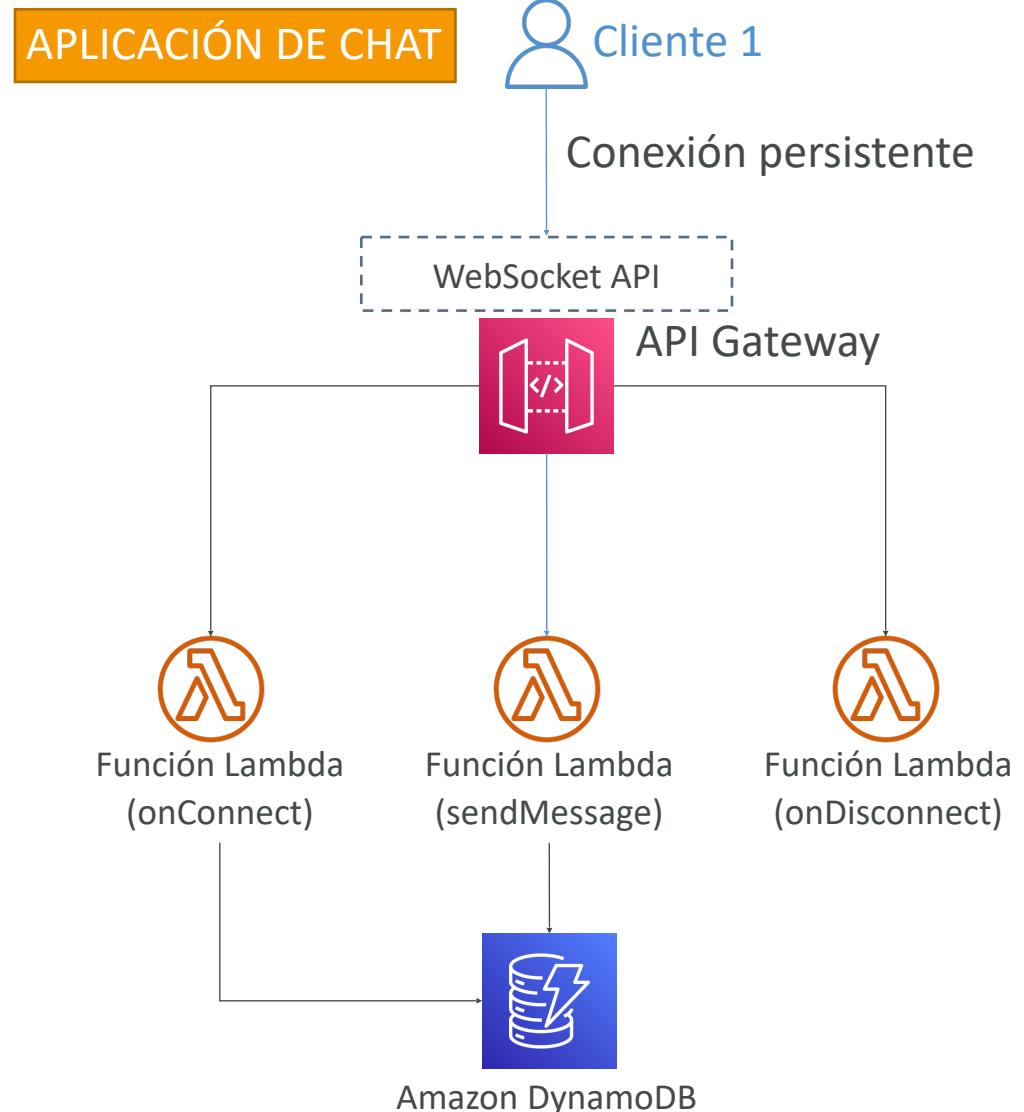
- Todas las funciones (excepto OpenID Connect / OAuth 2.0 nativos)

Autorizadores	HTTP API	REST API
AWS Lambda	✓	✓
IAM	✓	✓
Políticas de recursos		✓
Amazon Cognito	✓ *	✓
OpenID Connect nativo / OAuth 2.0 / JWT	✓	

Lista completa aquí: <https://docs.aws.amazon.com/apigateway/latest/developerguide/http-api-vs-rest.html>

API Gateway - WebSocket API - Visión general

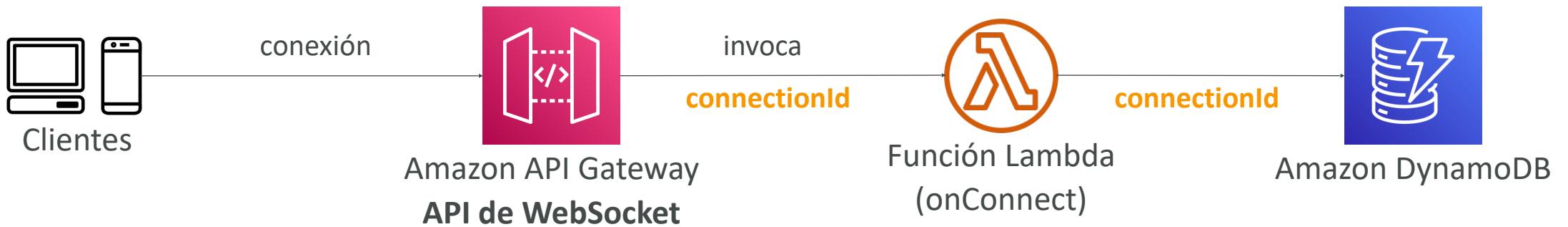
- ¿Qué es WebSocket?
 - Comunicación interactiva bidireccional entre el navegador de un usuario y un servidor
 - El servidor puede enviar información al cliente
 - Esto permite casos de uso de aplicaciones **con estado**
- Las API de WebSocket se utilizan a menudo en **aplicaciones en tiempo real**, como aplicaciones de chat, plataformas de colaboración, juegos multijugador y plataformas de comercio financiero.
- Funciona con servicios de AWS (Lambda, DynamoDB) o endpoints HTTP



Conexión a la API

URL del WebSocket

wss://[some-uniqueid].execute-api.[region].amazonaws.com/[stage-name]

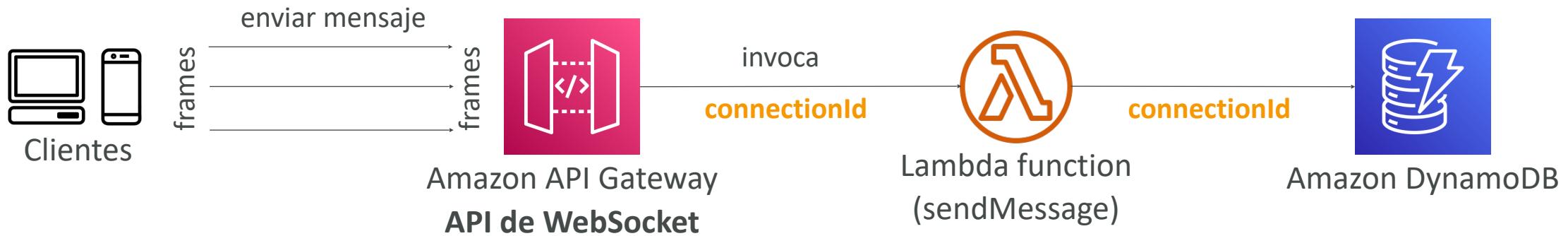


Mensajería Cliente-Servidor

Se reutiliza el ConnectionID

URL del WebSocket

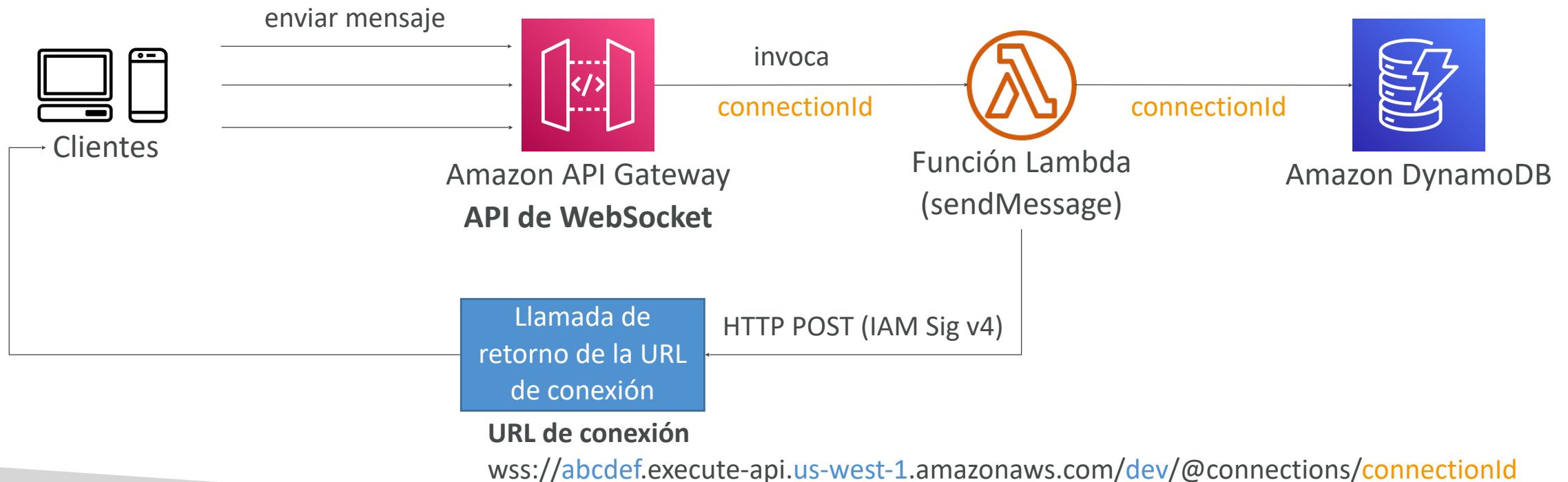
wss://abcdef.execute-api.us-west-1.amazonaws.com/dev



Mensajería de servidor a cliente

URL del WebSocket

wss://abcdef.execute-api.us-west-1.amazonaws.com/dev



Operaciones de URL de conexión

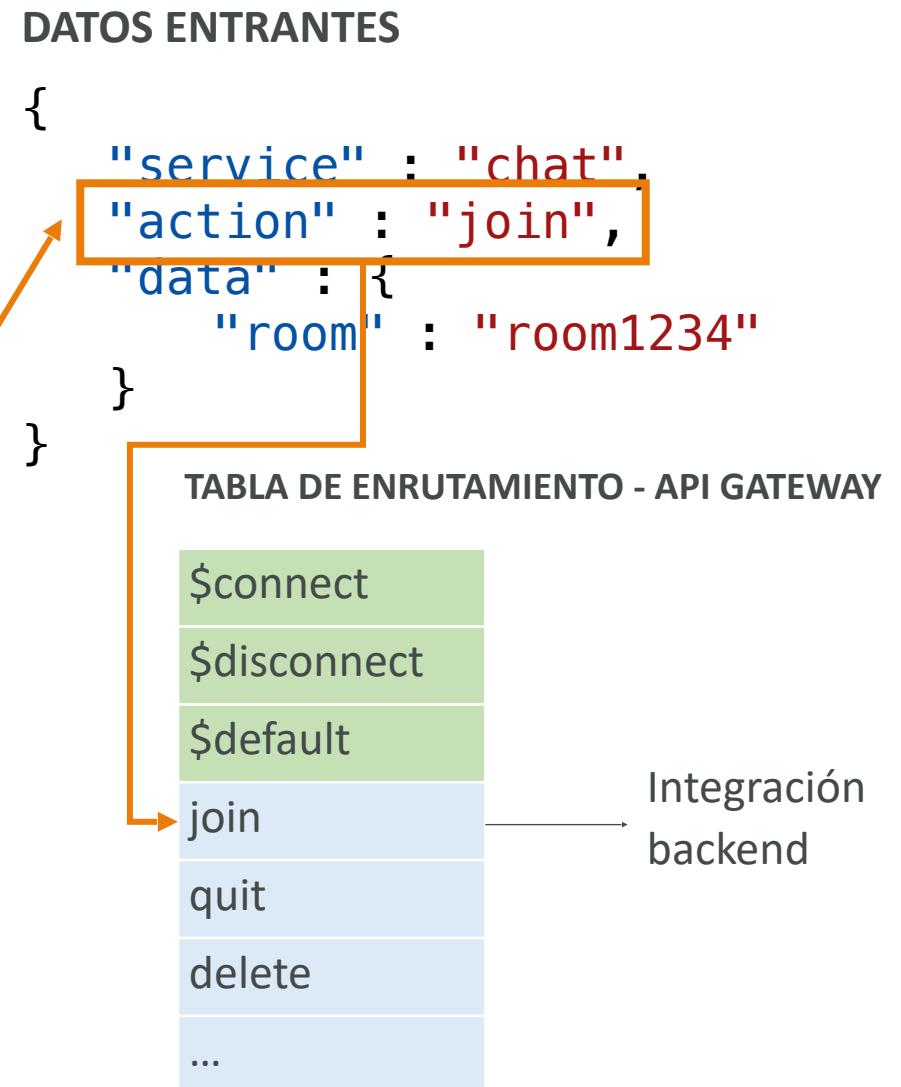
URL de conexión

wss://[@connections/](https://abcdef.execute-api.us-west-1.amazonaws.com/dev)[connectionId](#)

Operación	Acción
POST	Envía un mensaje del servidor al cliente WS conectado
GET	Obtiene el último estado de conexión del cliente WS conectado
DELETE	Desconectar el cliente conectado de la conexión WS

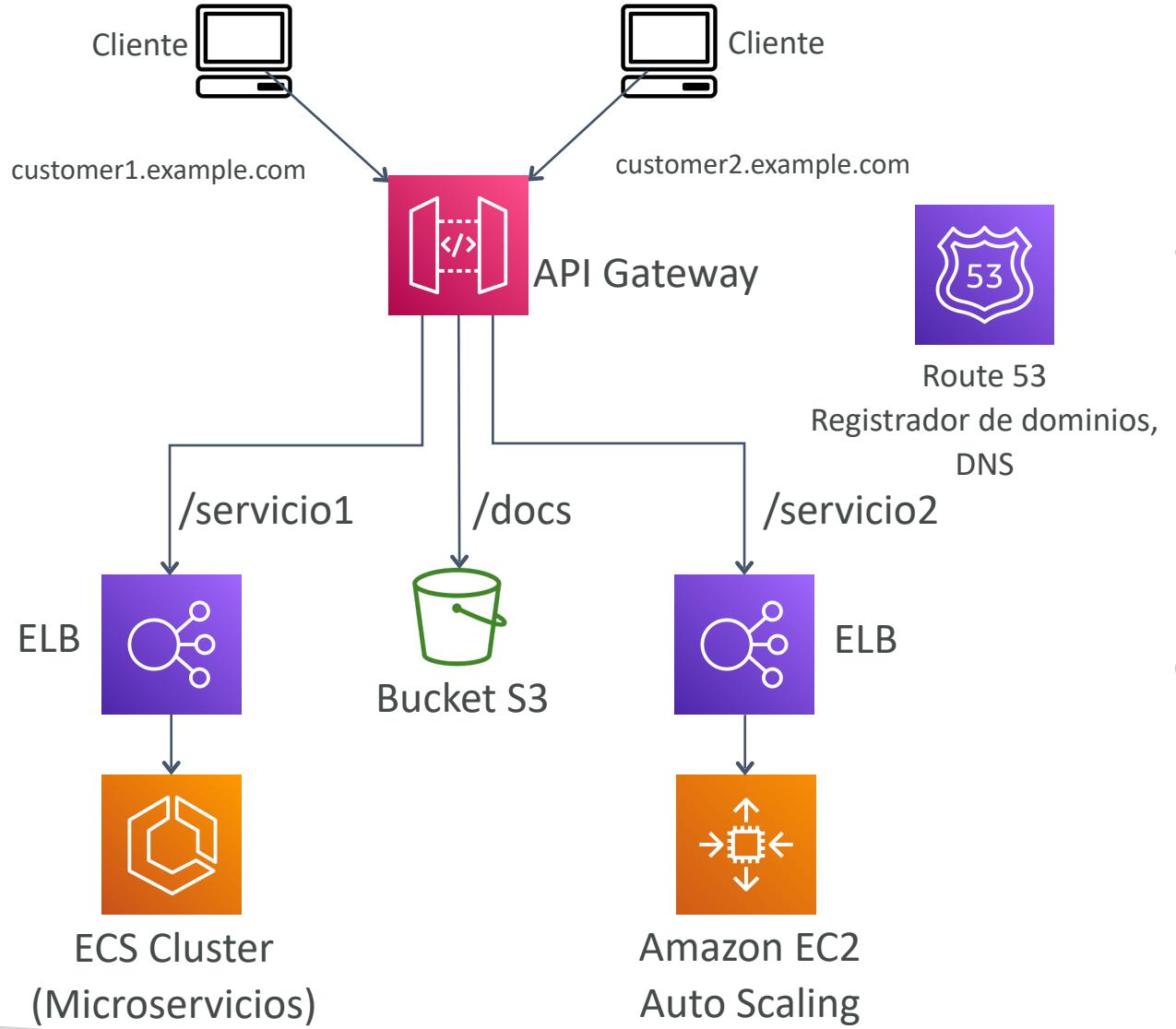
API Gateway – WebSocket API – Enrutamiento

- Los mensajes JSON entrantes se enrutan a un backend diferente
- Si no hay rutas => se envían a \$default
- Sigue una expresión de selección de ruta para seleccionar el campo en JSON desde el que enrutar
- Expresión de ejemplo: **\$request.body.action**
- El resultado se evalúa con las claves de ruta disponibles en tu API Gateway
- La ruta se conecta entonces al backend que has configurado a través de API Gateway



API Gateway - Arquitectura

- Crea una interfaz única para todos los microservicios de tu empresa
- Utiliza API endpoints con varios recursos
- Aplica un nombre de dominio sencillo y certificados SSL
- Puedes aplicar reglas de reenvío y transformación a nivel de API Gateway



AWS CICD

CodeCommit, CodePipeline, CodeBuild, CodeDeploy, ...

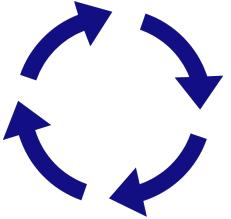
CICD - Introducción

- Hemos aprendido a:
 - Crear recursos de AWS, manualmente (fundamentos)
 - Interactuar con AWS mediante programación (AWS CLI)
 - Implementar código en AWS utilizando Elastic Beanstalk
- ¡Todos estos pasos manuales hacen muy probable que cometamos errores!
- Nos gustaría tener nuestro código "en un repositorio" y que se desplegara en AWS
 - Automáticamente
 - De la forma correcta
 - Asegurándonos de que se prueba antes de desplegarlo
 - Con posibilidad de pasar a diferentes etapas (dev, test, staging, prod)
 - Con aprobación manual cuando sea necesario
- Para ser un verdadero desarrollador de AWS... necesitamos aprender AWS CICD

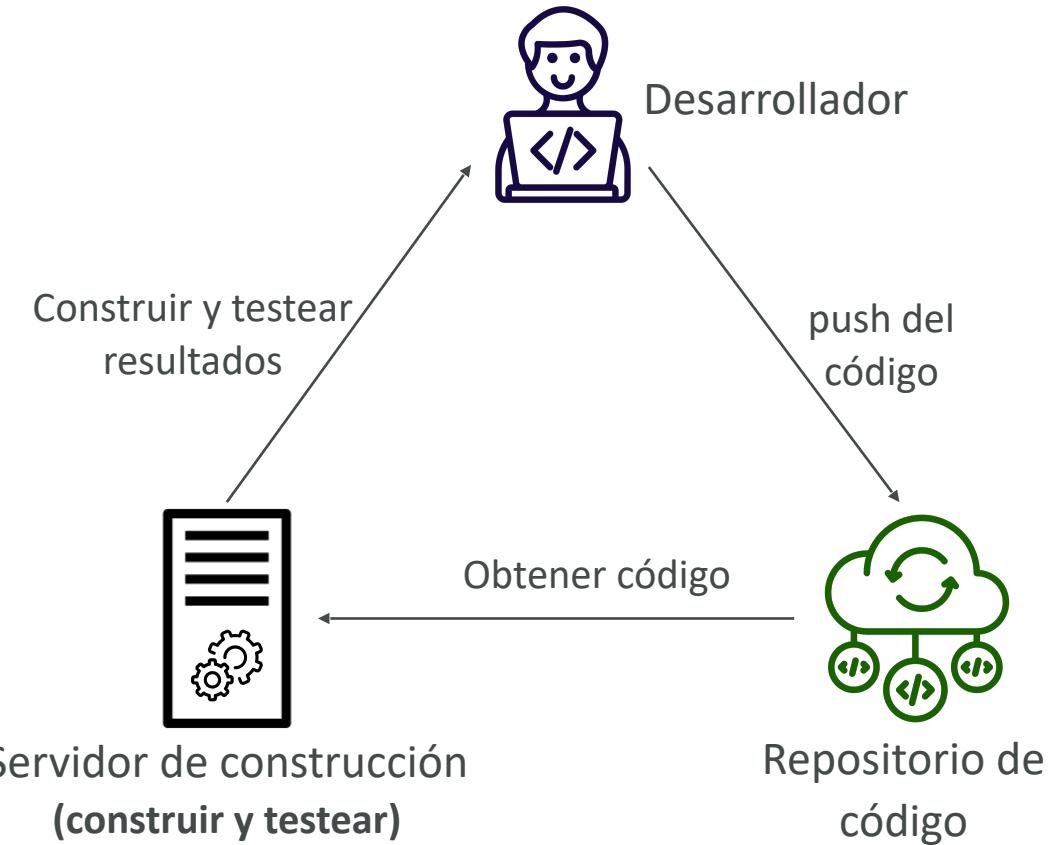
CICD - Introducción

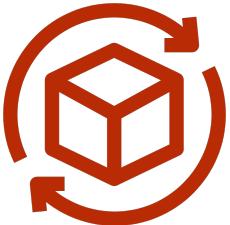
- Esta sección trata sobre la automatización del despliegue que hemos hecho hasta ahora, añadiendo al mismo tiempo una mayor seguridad.
- Aprenderemos sobre:
 - **AWS CodeCommit** - almacenar nuestro código
 - **AWS CodePipeline** - automatizar nuestra canalización desde el código hasta Elastic Beanstalk
 - **AWS CodeBuild** - crear y probar nuestro código
 - **AWS CodeDeploy** - despliegue del código en instancias EC2 (no Elastic Beanstalk)
 - **AWS CodeStar** - gestionar las actividades de desarrollo de software en un solo lugar
 - **AWS CodeArtifact** - almacenar, publicar y compartir paquetes de software
 - **AWS CodeGuru** - revisiones de código automatizadas mediante Machine Learning

Integración continua (CI)



- Los desarrolladores envían el código a un repositorio de código con frecuencia (por ejemplo, GitHub, CodeCommit, Bitbucket...)
- Un servidor de tests/construcción comprueba el código en cuanto se envía (CodeBuild, Jenkins CI, ...)
- El desarrollador recibe información sobre los tests y comprobaciones que se han superado o fallado.
- Encuentra errores pronto, y corrígelos
- Entrega más rápido a medida que se prueba el código
- Despliega a menudo
- Desarrolladores más felices, ya que están desbloqueados



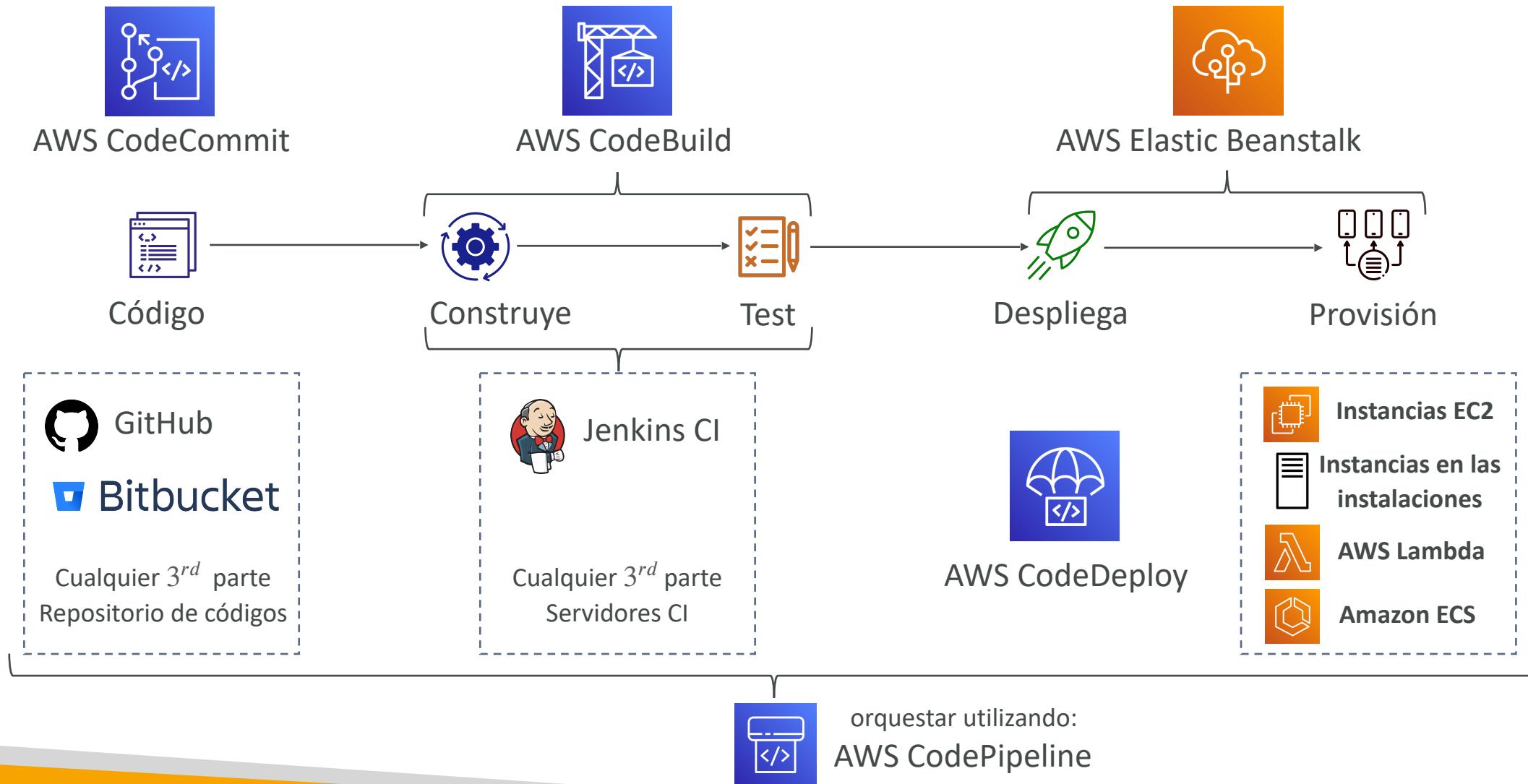


Entrega continua (CD)

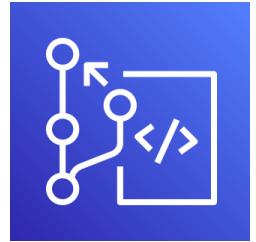
- Garantiza que el software pueda liberarse de forma fiable siempre que sea necesario
- Garantiza que los despliegues sean frecuentes y rápidos
- Pasa de "una versión cada 3 meses" a "5 versiones al día".
- Eso suele significar un despliegue automatizado (por ejemplo, CodeDeploy, Jenkins CD, Spinnaker, ...)



Stack tecnológico para CICD



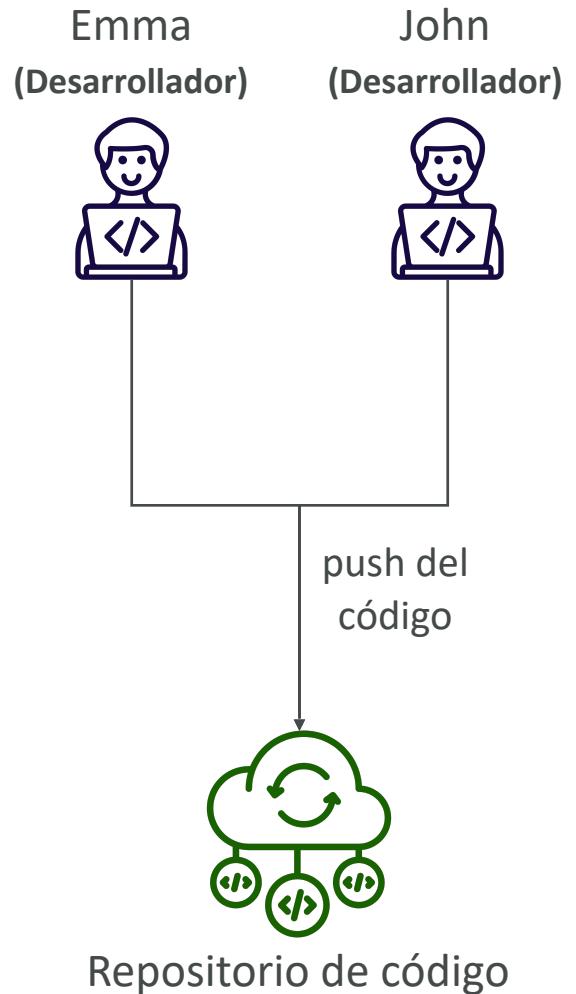
AWS CodeCommit



- El **control de versiones** es la capacidad de comprender los distintos cambios que se han producido en el código a lo largo del tiempo (y posiblemente de hacerlos retroceder).
- Todo esto se consigue utilizando un sistema de control de versiones como Git
- Un repositorio Git puede sincronizarse en tu ordenador, pero normalmente se carga en un repositorio central en línea
- Las ventajas son:
 - Colaborar con otros desarrolladores
 - Asegurarte de que el código tiene una copia de seguridad en algún sitio
 - Asegúrate de que es totalmente visible y auditible

AWS CodeCommit

- Los repositorios Git pueden ser caros
- El sector incluye GitHub, GitLab, Bitbucket, ...
- Y **AWS CodeCommit**:
 - Repositorios Git privados
 - Sin límite de tamaño en los repositorios (escalan sin problemas)
 - Totalmente gestionado, altamente disponible
 - Código sólo en la cuenta de AWS Cloud => mayor seguridad y normativa
 - Seguridad (cifrado, control de acceso, ...)
 - Integrado con Jenkins, AWS CodeBuild y otras herramientas CI



CodeCommit – Seguridad

- Las interacciones se realizan mediante Git (estándar)
- **Autenticación**
 - **Claves SSH** - Los usuarios de AWS pueden configurar claves SSH en su consola IAM
 - **HTTPS** - con el ayudante de credenciales CLI de AWS o credenciales Git para el usuario IAM
- **Autorización**
 - Políticas IAM para gestionar los permisos de usuarios/roles a los repositorios
- **Cifrado**
 - Los repositorios se cifran automáticamente en reposo utilizando AWS KMS
 - Cifrado en tránsito (sólo se puede utilizar HTTPS o SSH, ambos seguros)
- **Acceso entre cuentas**
 - NO compartas tus claves SSH ni tus credenciales de AWS
 - Utiliza un rol IAM en tu cuenta de AWS y usa AWS STS (API **AssumeRole**)

CodeCommit vs. GitHub

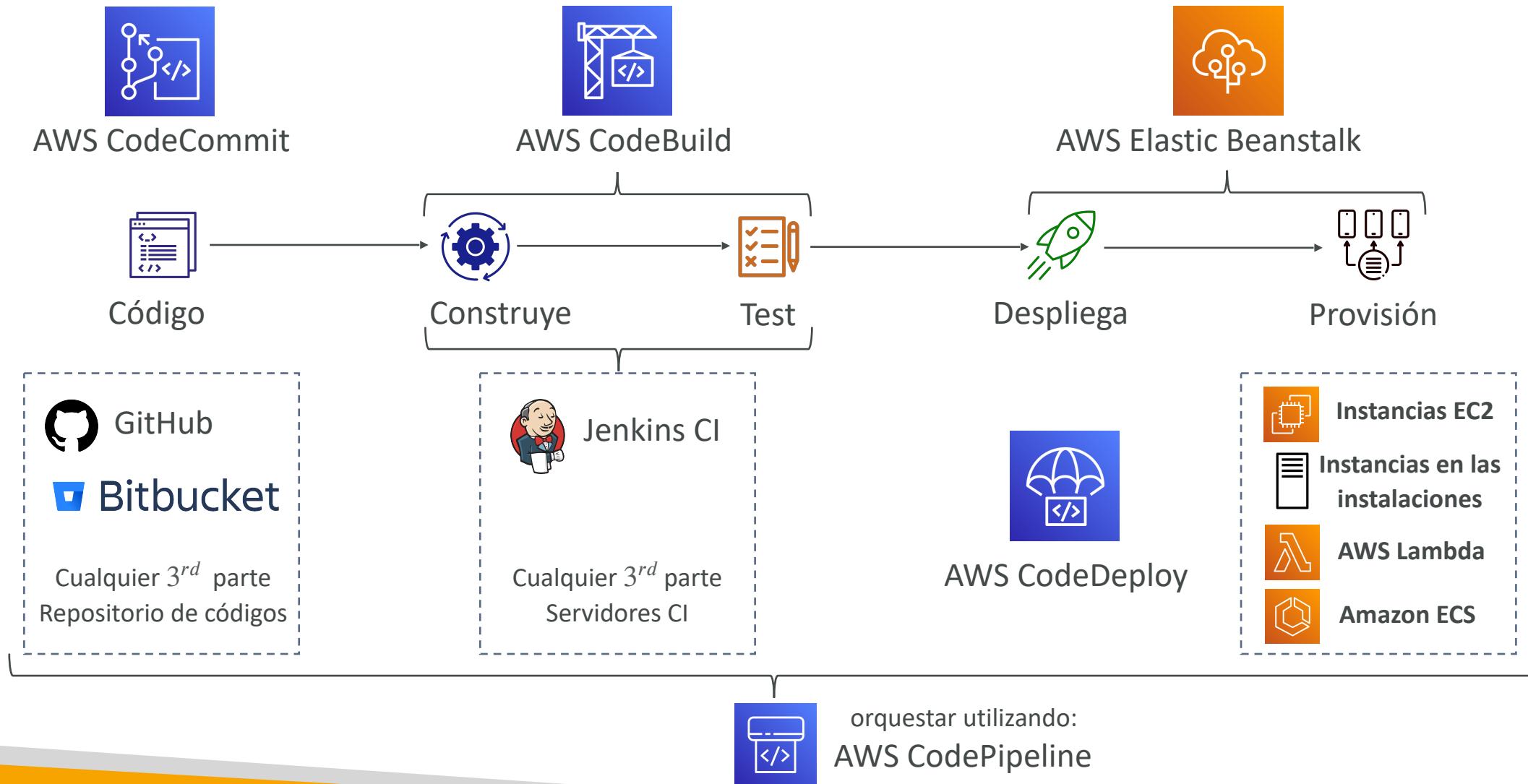
	CodeCommit	GitHub
Revisión del código de soporte (Pull Requests)	✓	✓
Integración con AWS CodeBuild	✓	✓
Autenticación (SSH y HTTPS)	✓	✓
Seguridad	Usuarios y roles de IAM	Usuarios de GitHub
Hosting (Alojamiento)	Gestionado y alojado por AWS	<ul style="list-style-type: none">- Alojado en GitHub- GitHub Enterprise: autoalojado en tus servidores
Interfaz de usuario	Mínimo	Totalmente destacado

AWS CodePipeline



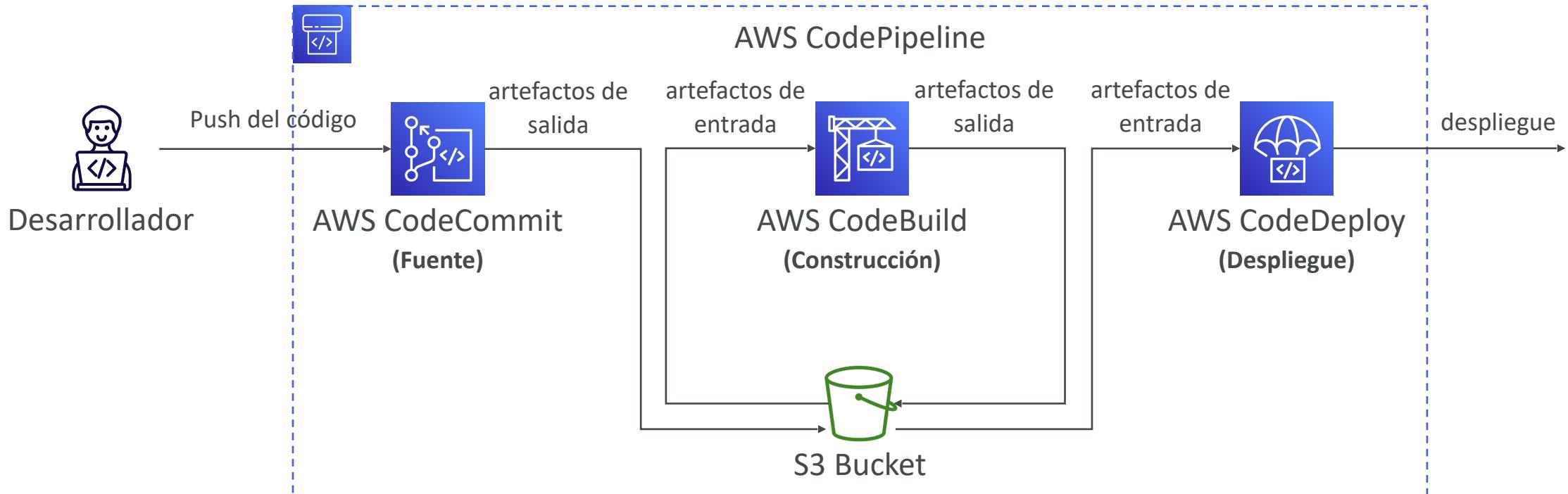
- Flujo de trabajo visual para orquestar tu CICD
- **Fuente** - CodeCommit, ECR, S3, Bitbucket, GitHub
- **Construcción** - CodeBuild, Jenkins, CloudBees, TeamCity
- **Test** - CodeBuild, AWS Device Farm, herramientas de terceros, ...
- **Implementación** - CodeDeploy, Elastic Beanstalk, CloudFormation, ECS, S3, ...
- Consta de etapas:
 - Cada etapa puede tener acciones secuenciales y/o acciones paralelas
 - Ejemplo: Construcción → Test → Deploy → Test de carga → ...
 - La aprobación manual puede definirse en cualquier etapa

Stack tecnológico para CICD



CodePipeline - Artefactos

- Cada etapa del pipeline puede crear **artefactos**
- Los artefactos se almacenan en un bucket de S3 y se pasan a la siguiente etapa

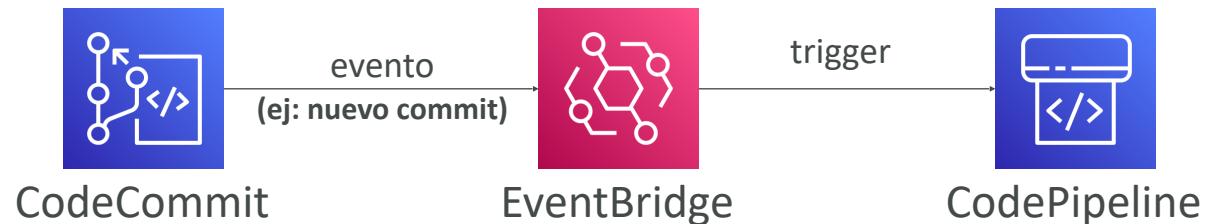


CodePipeline – Solución de problemas

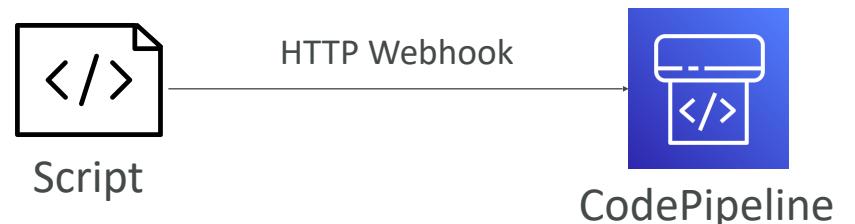
- Utiliza **CloudWatch Events (Amazon EventBridge)**. Ejemplo:
 - Puedes crear eventos para pipelines fallidos
 - Puedes crear eventos para etapas canceladas
- Si CodePipeline falla una etapa, tu pipeline se detiene, y puedes obtener información en la consola
- Si el pipeline no puede realizar una acción, asegúrate de que el "rol de servicio IAM" adjunto tiene suficientes permisos IAM (política IAM)
- AWS CloudTrail puede utilizarse para auditar las llamadas a la API de AWS

CodePipeline - Eventos vs. Webhooks vs. Polling

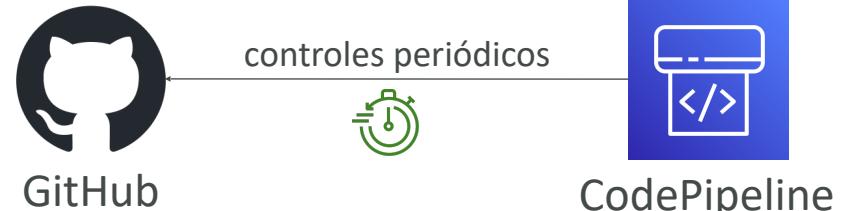
Eventos



Webhooks



Polling



Nota: Los eventos son los predeterminados y recomendados

CodePipeline

Tipos de acción

Restricciones para artefactos

- **Propietario**

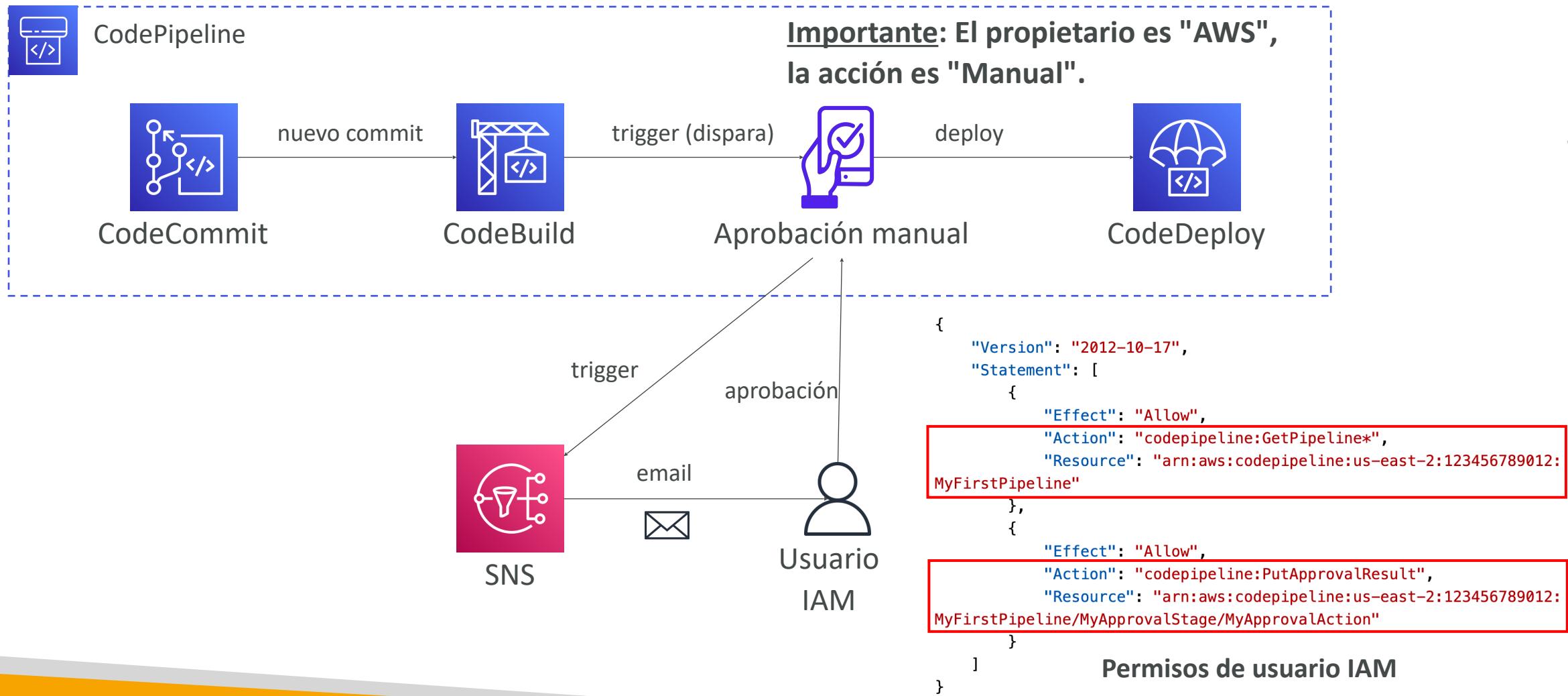
- **AWS** - para servicios AWS
- **3^a Parte** - GitHub o Alexa Skills Kit
- **Personalizado** - Jenkins

- **Tipo de acción**

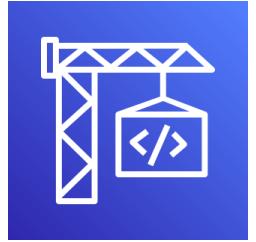
- **Fuente** - S3, ECR, GitHub, ...
- **Build / Construir** - CodeBuild, Jenkins
- **Test** - CodeBuild, Device Farm, Jenkins
- **Aprobación** - Manual
- **Invocar** - Lambda
- **Deploy / Despliegue** - S3, CloudFormation, CodeDeploy, Elastic Beanstalk, OpsWorks, ECS, Service Catalog, ...

Propietario	Tipo de acción	Proveedor	Número válido de artefactos de entrada	Número válido de artefactos de salida
AWS	Source	S3	0	1
AWS	Source	CodeCommit	0	1
AWS	Source	ECR	0	1
3 ^a Parte	Source	GitHub	0	1
AWS	Build	Codebuild	1 a 5	0 a 5
AWS	Test	CodeBuild	1 a 5	0 a 5
AWS	Test	Device Farm	1	0
AWS	Approval	Manual	0	0
AWS	Deploy	S3	1	0
AWS	Deploy	CloudFormation	0 a 10	0 a 1
AWS	Deploy	CodeDeploy	1	0
AWS	Deploy	Elastic Beanstalk	1	0
AWS	Deploy	OpsWorks Stacks	1	0
AWS	Deploy	ECS	1	0
AWS	Deploy	Service Catalog	1	0
AWS	Invoke	Lambda	0 a 5	0 a 5
3 ^a Parte	Deploy	Alexa Skills Kit	1 a 2	0
Personalizado	Build	Jenkins	0 a 5	0 a 5
Personalizado	Test	Jenkins	0 a 5	0 a 5
Personalizado	Cualquier categoría sugerida	Especificado en la acción personalizada	0 a 5	

CodePipeline - Fase de aprobación manual



AWS CodeBuild



- Un servicio de integración continua (CI) totalmente gestionado
- Escalado continuo (sin servidores que gestionar o aprovisionar, ni cola de compilación)
- Compila código fuente, ejecuta tests, produce paquetes de software, ...
- Alternativa a otras herramientas de compilación (por ejemplo, Jenkins)
- Se cobra por minuto de recursos informáticos (tiempo que se tarda en completar las compilaciones)
- Aprovecha Docker para realizar compilaciones reproducibles
- Utiliza imágenes Docker preempaquetadas o crea tu propia imagen Docker personalizada
- Seguridad:
 - Integración con KMS para el cifrado de artefactos de construcción
 - IAM para los permisos de CodeBuild, y VPC para la seguridad de la red
 - AWS CloudTrail para logs de llamadas a la API

AWS CodeBuild

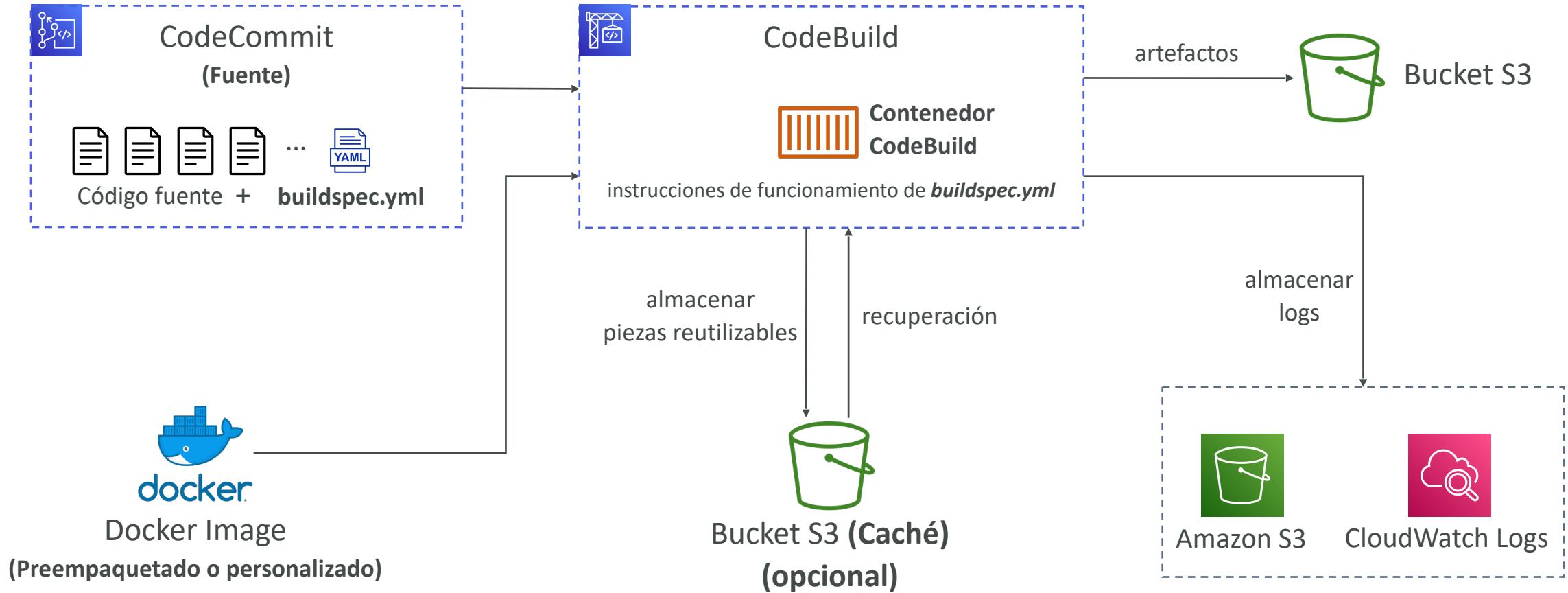


- **Fuente** - CodeCommit, S3, Bitbucket, GitHub
- **Instrucciones de compilación:** Archivo de código **buildspec.yml** o insértalo manualmente en la Consola
- Los **logs de salida** se pueden almacenar en Amazon S3 y CloudWatch Logs
- Utiliza las métricas de CloudWatch para supervisar las estadísticas de compilación
- Utiliza CloudWatch Events para detectar compilaciones fallidas y activar notificaciones
- Utiliza las Alarmas de CloudWatch para notificar si necesitas "umbrales" de fallos
- **Los proyectos de construcción pueden definirse dentro de CodePipeline o CodeBuild**

CodeBuild - Entornos soportados

- Java
- Ruby
- Python
- Go
- Node.js
- Android
- .NET Core
- PHP
- Docker – amplía el entorno que quieras

CodeBuild – ¿Cómo funciona?



CodeBuild – buildspec.yml

- El archivo **buildspec.yml** debe estar en la **raíz** de tu código
- **env** - definir variables de entorno
 - **variables** - variables de texto plano
 - **parameter-store** - variables almacenadas en el Almacén de Parámetros SSM
 - **secrets-manager** - variables almacenadas en AWS Secrets Manager
- **fases** - especifica los comandos a ejecutar:
 - **install** - instala las dependencias que puedas necesitar para tu compilación
 - **pre_build** - comandos finales para ejecutar antes de construir
 - **Build - comandos de construcción actual**
 - **post_build** - toques finales (por ejemplo, salida zip)
- **artefactos** - qué subir a S3 (cifrado con KMS)
- **caché** - archivos que almacenar en caché (normalmente dependencias) en S3 para acelerar la construcción en el futuro

```
version: 0.2

env:
  variables:
    JAVA_HOME: "/usr/lib/jvm/java-8-openjdk-amd64"
  parameter-store:
    LOGIN_PASSWORD: /CodeBuild/dockerLoginPassword

phases:
  install:
    commands:
      - echo "Entered the install phase..."
      - apt-get update -y
      - apt-get install -y maven
  pre_build:
    commands:
      - echo "Entered the pre_build phase..."
      - docker login -u User -p $LOGIN_PASSWORD
  build:
    commands:
      - echo "Entered the build phase..."
      - echo "Build started on `date`"
      - mvn install
  post_build:
    commands:
      - echo "Entered the post_build phase..."
      - echo "Build completed on `date`"

artifacts:
  files:
    - target/messageUtil-1.0.jar

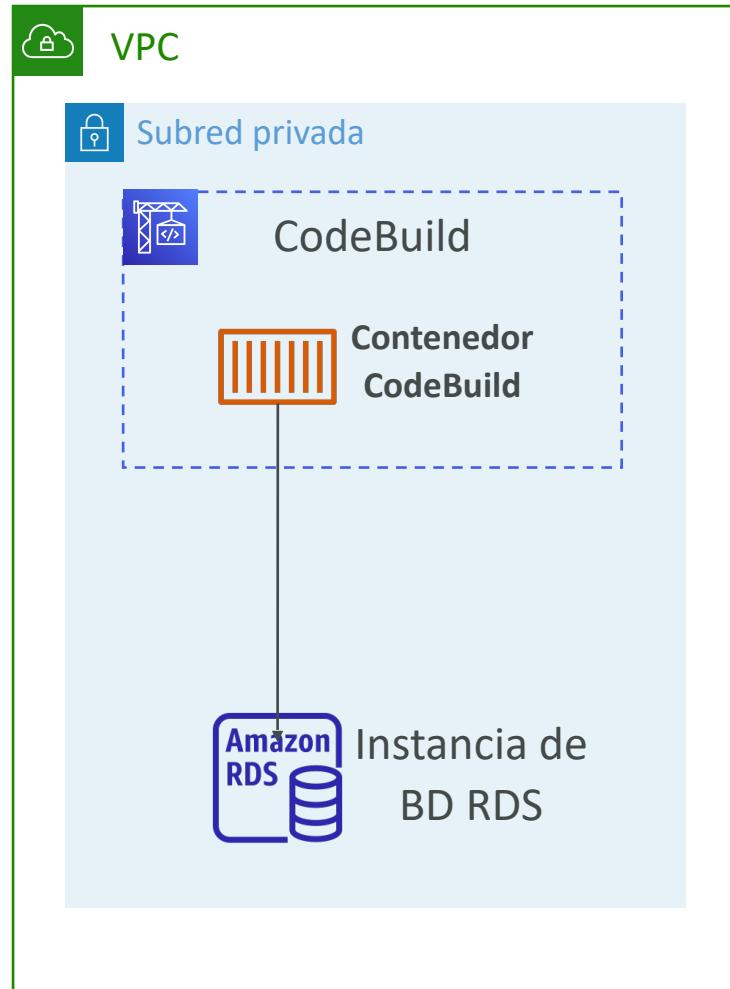
cache:
  paths:
    - "/root/.m2/**/*"
```

CodeBuild - Construcción local

- En caso de necesitar una solución de problemas profunda más allá de los logs...
- Puedes ejecutar CodeBuild localmente en tu escritorio (después de instalar Docker)
- Para ello, aprovecha el Agente CodeBuild
- <https://docs.aws.amazon.com/codebuild/latest/userguide/use-codebuild-agent.html>

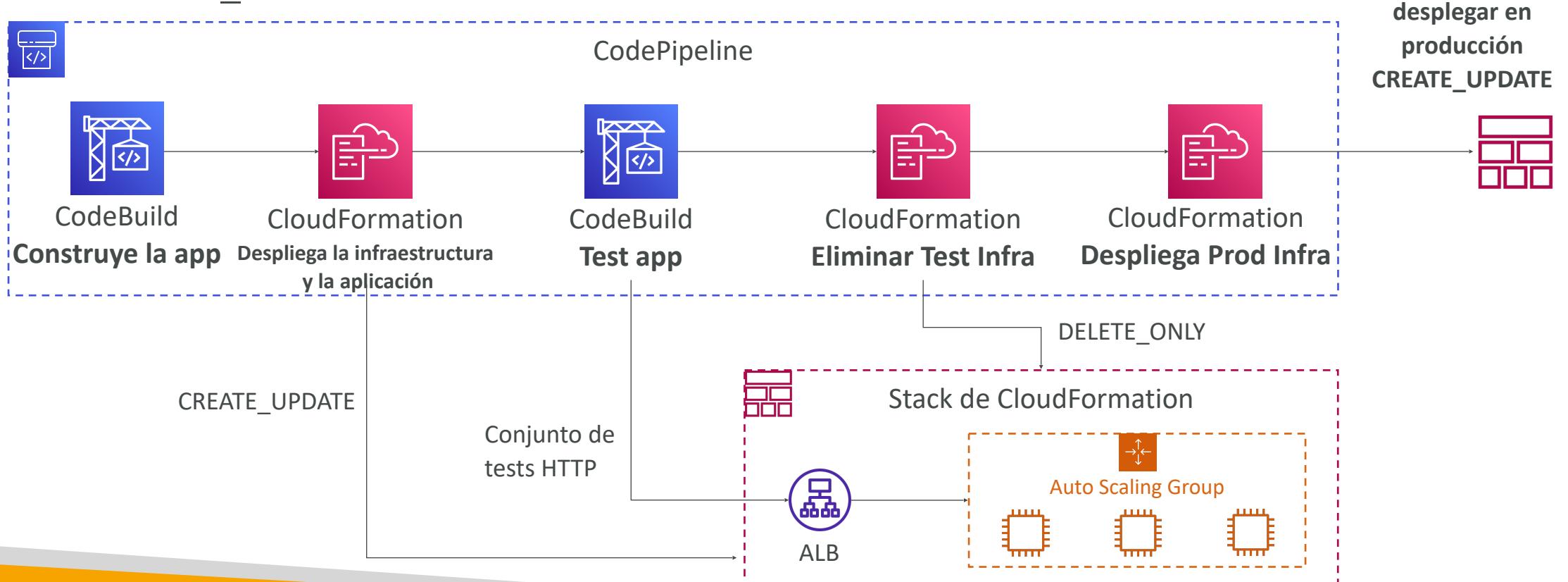
CodeBuild - Dentro de VPC

- Por defecto, tus contenedores CodeBuild se lanzan fuera de tu VPC
 - No puede acceder a los recursos de una VPC
- Puedes especificar una configuración de VPC:
 - VPC ID
 - ID de subred
 - ID de grupos de seguridad
- Entonces tu compilación podrá acceder a los recursos de tu VPC (por ejemplo, RDS, ElastiCache, EC2, ALB, ...)
- Casos de uso: Test de integración, consulta de datos, balanceadores de carga internos, ...



CodePipeline – Integración de CloudFormation

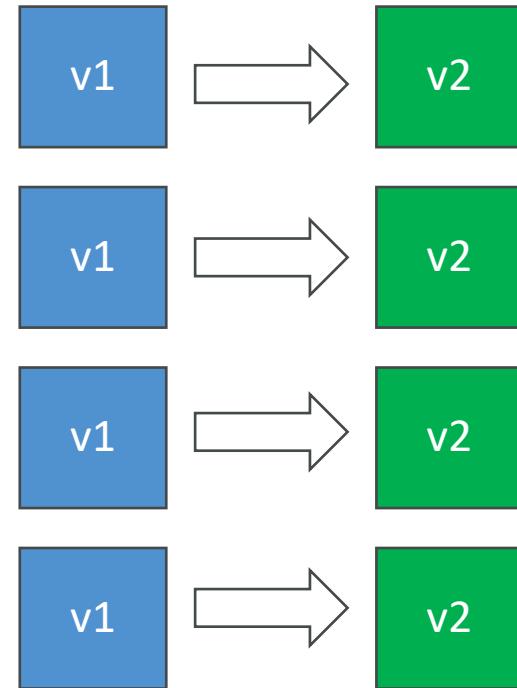
- CloudFormation se utiliza para desplegar infraestructuras complejas mediante una API
 - **CREATE_UPDATE** – crear o actualizar un stack existente
 - **DELETE_ONLY** – eliminar un stack si existe



AWS CodeDeploy



- Queremos desplegar nuestra aplicación automáticamente en muchas instancias EC2
- Estas instancias EC2 no están gestionadas por Elastic Beanstalk
- Hay varias formas de gestionar los despliegues utilizando herramientas de código abierto (Ansible, Terraform, Chef, Puppet, ...)
- Podemos utilizar el servicio gestionado AWS CodeDeploy



CodeDeploy – Pasos para que funcione

- Cada instancia EC2/servidor local debe estar ejecutando el Agente CodeDeploy
- El agente está continuamente sondeando a AWS CodeDeploy en busca de trabajo por hacer
- La aplicación + **appspec.yml** se extrae de GitHub o S3
- Las instancias EC2 ejecutarán las instrucciones de despliegue en **appspec.yml**
- El Agente de CodeDeploy informará del éxito/fracaso del despliegue



CodeDeploy - Componentes principales

- **Aplicación** - un nombre único funciona como contenedor (revisión, configuración de despliegue, ...)
- **Plataforma informática** - EC2/On-Premises, AWS Lambda o Amazon ECS
- **Configuración de despliegue** - un conjunto de reglas de despliegue para el éxito/fracaso
 - **EC2/On-premises** - especifica el número mínimo de instancias sanas para el despliegue
 - **AWS Lambda o Amazon ECS** - especifica cómo se enruta el tráfico a tus versiones actualizadas
- **Grupo de despliegue** - grupo de instancias EC2 etiquetadas (permite desplegar gradualmente, o dev, test, prod...)
- **Tipo de despliegue** - método utilizado para desplegar la aplicación en un grupo de despliegue
 - **Despliegue in situ** - soporta EC2/en local
 - **Despliegue Blue/Green** - sólo soporta instancias EC2, AWS Lambda y Amazon ECS
- **Perfil de Instancia IAM** - dar a las instancias EC2 los permisos para acceder tanto a S3 / GitHub
- **Revisión de la aplicación** - código de la aplicación + archivo **appspec.yml**
- **Rol de servicio** - un rol IAM para que CodeDeploy realice operaciones en instancias EC2, ASGs, ELBs...
- **Revisión de destino** - la revisión más reciente que quieres desplegar en un grupo de despliegue

CodeDeploy – appspec.yml

- **files** - cómo crear y copiar desde S3 / GitHub al sistema de archivos
 - **fuente**
 - **destino**
- **hooks** - conjunto de instrucciones a realizar para desplegar la nueva versión (los hooks pueden tener timeouts), el orden es:
 - **ApplicationStop**
 - **DownloadBundle**
 - **BeforeInstall**
 - **Install**
 - **AfterInstall**
 - **ApplicationStart**
 - **ValidateService** (**¡¡importante!!**)

```
version: 0.0
os: linux

files:
  - source: Config/config.txt
    destination: /webapps/Config
  - source: source
    destination: /webapps/myApp

hooks:
  BeforeInstall:
    - location: Scripts/UnzipResourceBundle.zip
    - location: Scripts/UnzipDataBundle.zip
  AfterInstall:
    - location: Scripts/RunResourceTests.sh
      timeout: 180
  ApplicationStart:
    - location: Scripts/RunFunctionalTests.sh
      timeout: 3600
  ValidateService:
    - location: Scripts/MonitorService.sh
      timeout: 3600
      runas: codedeployuser
```

Orden de despliegue y hooks de AWS CodeDeploy

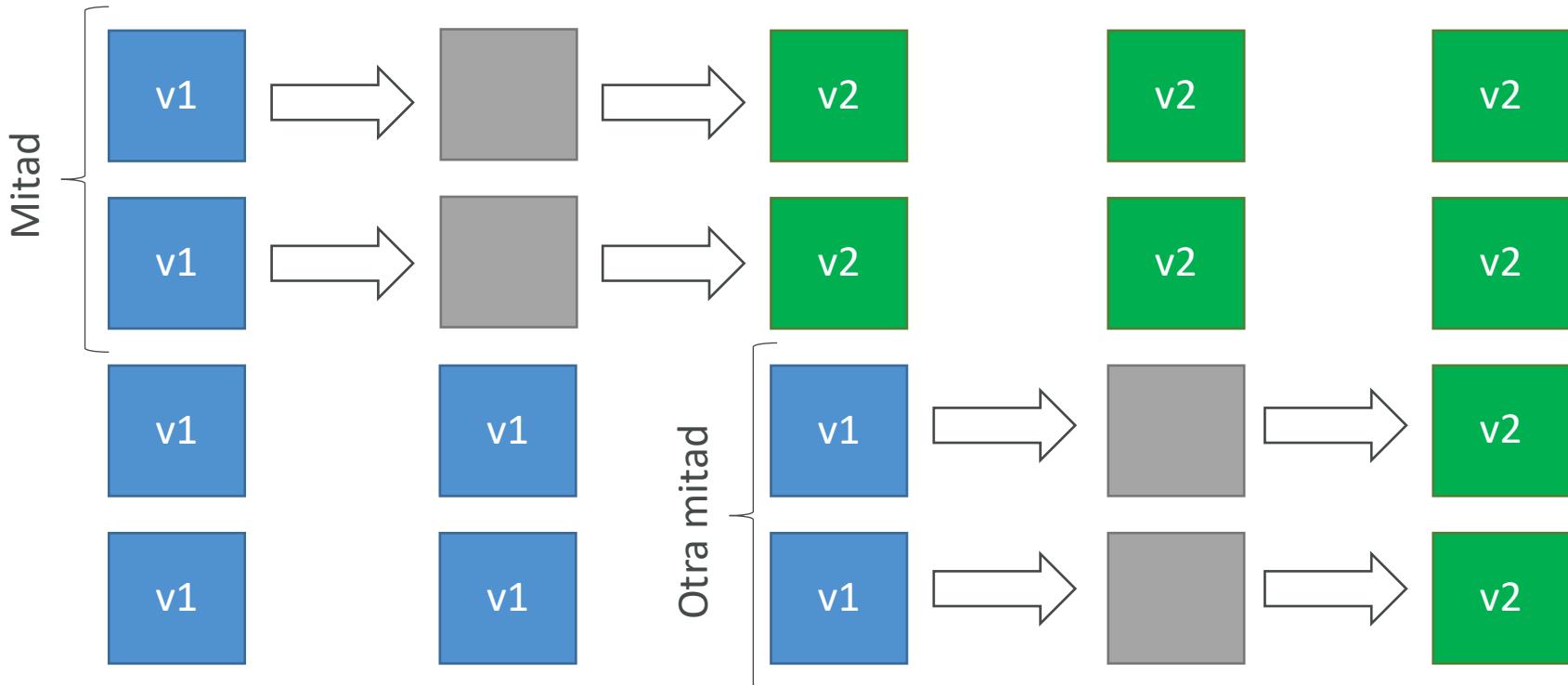
Event	Start time	End time	Duration	Status
ApplicationStop	Sept 26, 2018 7:51:29 AM UTC	Sept 26, 2018 7:51:29 AM UTC	less than one second	Succeeded
DownloadBundle	Sept 26, 2018 7:51:30 AM UTC	Sept 26, 2018 7:51:30 AM UTC	less than one second	Succeeded
BeforeInstall	Sept 26, 2018 7:51:31 AM UTC	Sept 26, 2018 7:51:32 AM UTC	2 secs	Succeeded
Install	Sept 26, 2018 7:51:33 AM UTC	Sept 26, 2018 7:51:33 AM UTC	less than one second	Succeeded
AfterInstall	Sept 26, 2018 7:51:34 AM UTC	Sept 26, 2018 7:51:34 AM UTC	less than one second	Succeeded
ApplicationStart	Sept 26, 2018 7:51:35 AM UTC	Sept 26, 2018 7:51:35 AM UTC	less than one second	Succeeded
ValidateService	Sept 26, 2018 7:51:36 AM UTC	Sept 26, 2018 7:51:36 AM UTC	less than one second	Succeeded
BeforeAllowTraffic	Sept 26, 2018 7:51:49 AM UTC	Sept 26, 2018 7:51:49 AM UTC	less than one second	Succeeded
AllowTraffic	Sept 26, 2018 7:51:50 AM UTC	Sept 26, 2018 7:52:11 AM UTC	21 secs	Succeeded
AfterAllowTraffic	Sept 26, 2018 7:52:12 AM UTC	Sept 26, 2018 7:52:12 AM UTC	less than one second	Succeeded

CodeDeploy - Configuración del despliegue

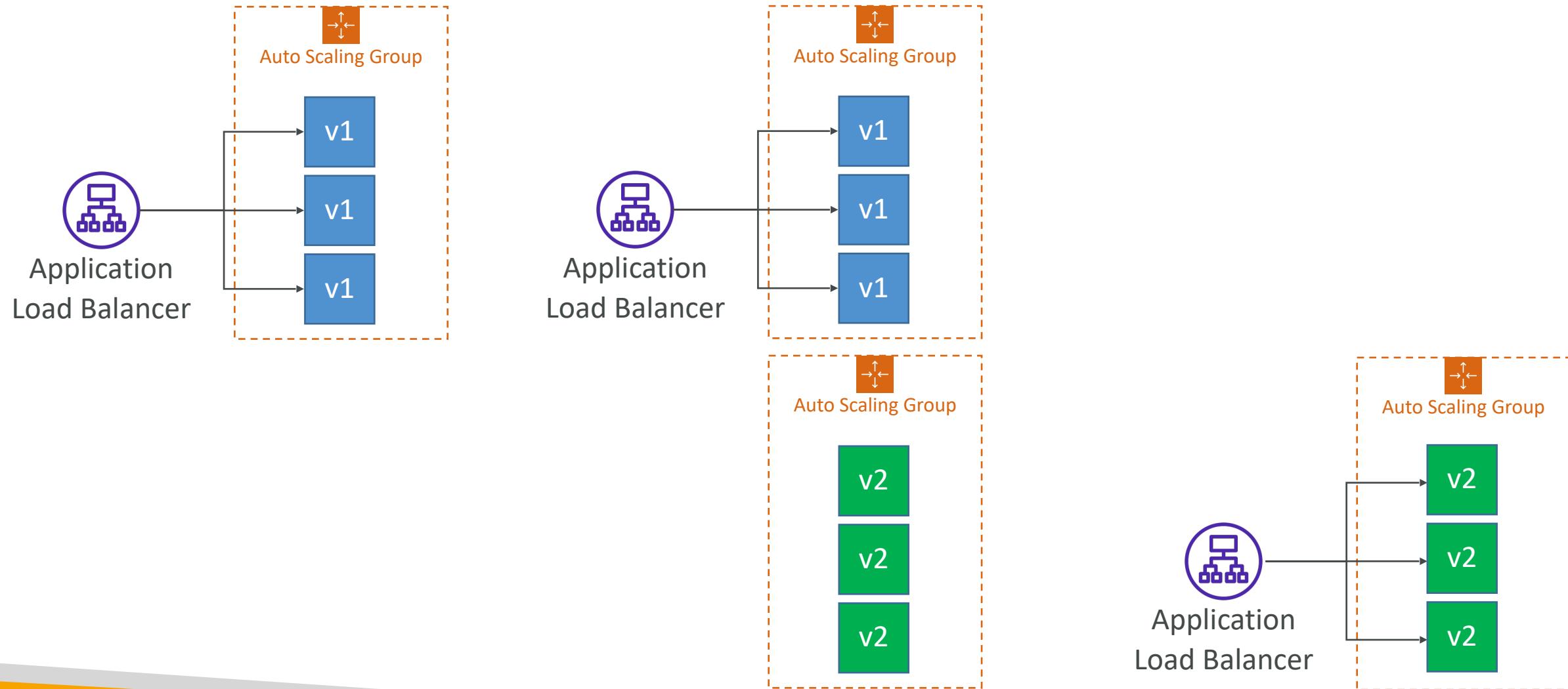
- Configuraciones:
 - **Uno a la vez** - una instancia EC2 a la vez, si falla una instancia se detiene el despliegue
 - **Medio a la vez** - 50%
 - **Todos a la vez** - rápido pero sin host “sano” = tiempo de inactividad. Es bueno para desarrollo
 - **Personalizada**: host “sano” mínimo = 75%.
- Fallos:
 - Las instancias EC2 permanecen en estado “Fallido”
 - Los nuevos despliegues se desplegarán primero en las instancias fallidas
 - **Para revertir, vuelve a desplegar el despliegue anterior o activa la reversión automática de fallos**
- Grupos de despliegue:
 - Un conjunto de instancias EC2 **etiquetadas**
 - Directamente a un ASG
 - Mezcla de ASG / Etiquetas para que puedas construir segmentos de despliegue
 - Personalización en scripts con variables de entorno DEPLOYMENT_GROUP_NAME

CodeDeploy - Despliegue in situ

Mitad a mitad

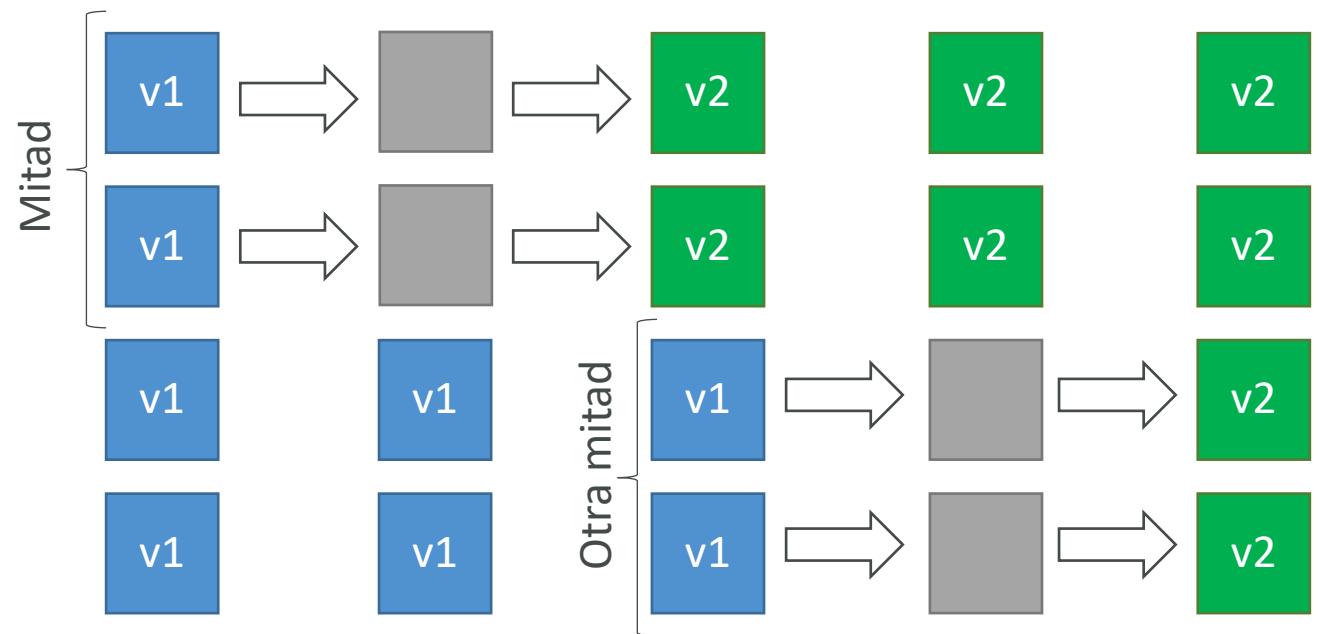


CodeDeploy - Despliegue Blue/Green



CodeDeploy - Despliegue en EC2

- Define cómo desplegar la aplicación mediante **appspec.yml** + Estrategia de despliegue
- Realizará la actualización in situ en tu flota de instancias EC2
- Puede utilizar hooks para verificar el despliegue después de cada fase de despliegue



CodeDeploy - Despliega en un ASG

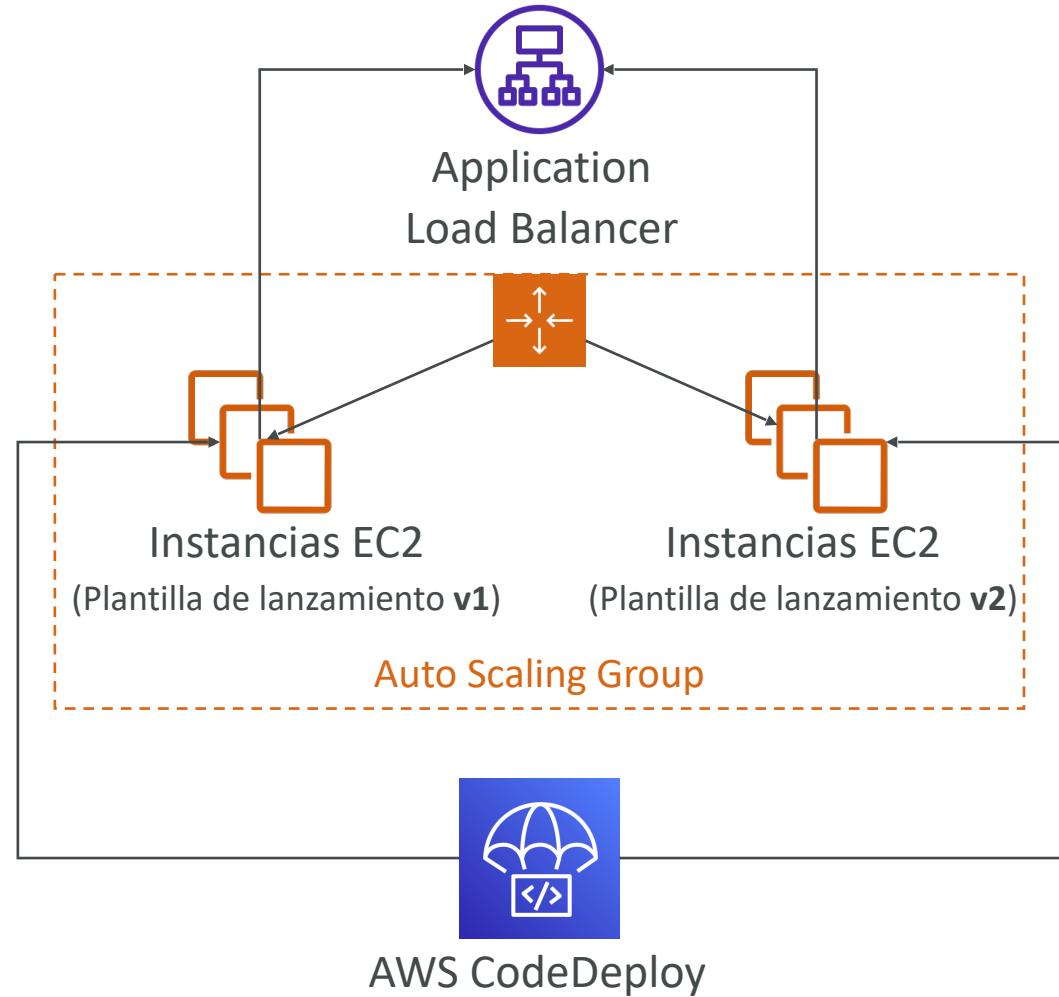
- **Despliegue in situ**

- Actualiza las instancias EC2 existentes
- Las instancias EC2 recién creadas por un ASG también recibirán despliegues automatizados

- **Despliegue Blue/Green**

- Se crea un nuevo Auto Scaling Group (se copia la configuración)
- Elige cuánto tiempo se conservarán las antiguas instancias EC2 (antiguo ASG)
- Debe utilizar un ELB

Despliegue Blue/Green



CodeDeploy - Redistribución y retrocesos

- **Retroceso / Rollback** = volver a desplegar una revisión de tu aplicación previamente desplegada
- Los despliegues se pueden deshacer (Rolling back):
 - **Automáticamente** - retroceder cuando falla un despliegue o retroceder cuando se alcanzan los umbrales de alarma de CloudWatch
 - **Manualmente**
- Desactivar reversiones - no realizar reversiones para este despliegue
- **Si se produce un rollback, CodeDeploy vuelve a desplegar la última revisión buena conocida como un nuevo despliegue (no una versión restaurada)**

CodeDeploy - Resolución de problemas

28/01/2023 @ 5:01pm

- **Error de despliegue: “`InvalidSignatureException – Signature expired: [time] is now earlier than [time]`”**
 - Para que CodeDeploy pueda realizar sus operaciones, necesita referencias temporales precisas
 - Si la fecha y la hora de tu instancia EC2 no están configuradas correctamente, podrían no coincidir con la fecha de firma de tu petición de despliegue, lo que CodeDeploy rechaza
- Comprueba los logs para entender los problemas de despliegue
 - Para Amazon Linux, Ubuntu y RHEL los logs se almacenan en `/opt/codedeploy-agent/deployment-root/deployment-logs/codedeploy-agent-deployments.log`



14/06/2020 @ 4:30am

AWS CodeStar



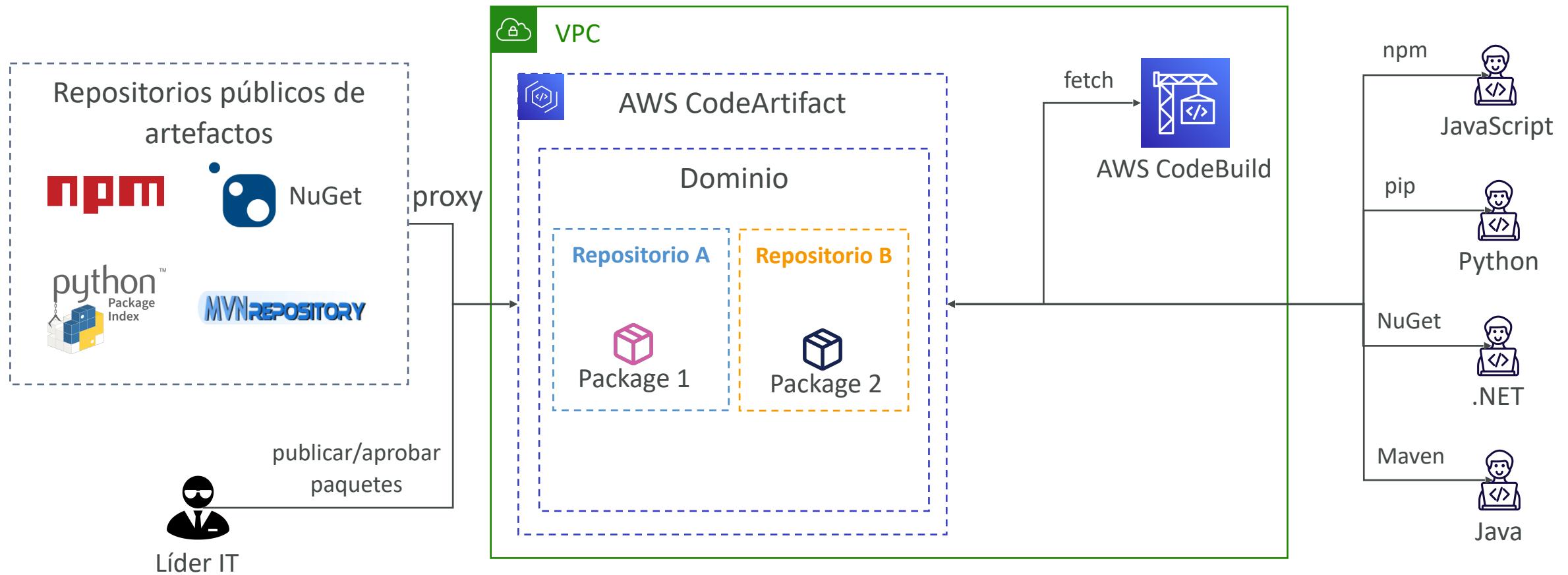
- Una solución integrada que agrupa: GitHub, CodeCommit, CodeBuild, CodeDeploy, CloudFormation, CodePipeline, CloudWatch, ...
- Crea rápidamente proyectos "listos para CICD" para EC2, Lambda, Elastic Beanstalk
- Lenguajes soportados: C#, Go, HTML 5, Java, Node.js, PHP, Python, Ruby
- Integración de seguimiento de incidencias con JIRA / GitHub Issues
- Capacidad de integración con Cloud9 para obtener un IDE web (no en todas las regiones)
- Un dashboard para ver todos tus componentes
- Servicio gratuito, paga sólo por el uso subyacente de otros servicios
- Personalización limitada

AWS CodeArtifact



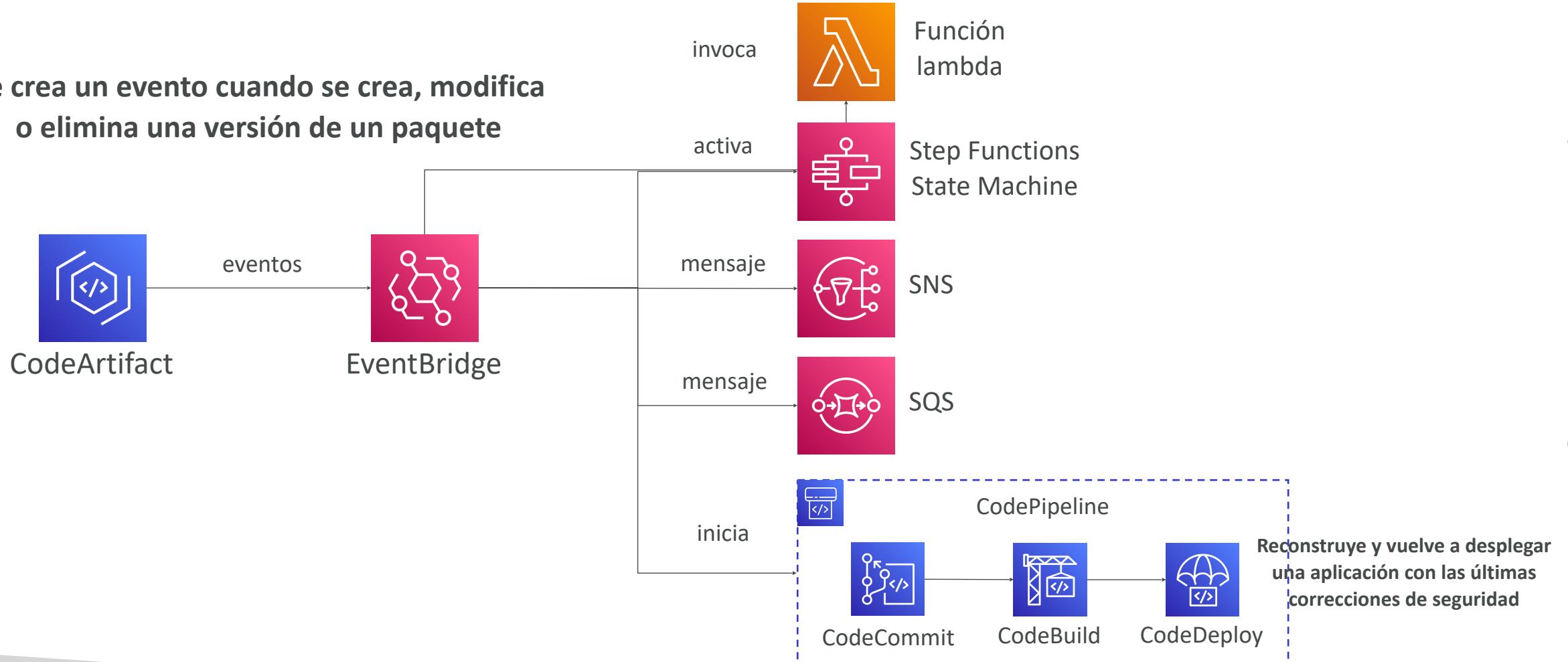
- Los paquetes de software dependen unos de otros para construirse (también llamadas dependencias de código), y se crean otros nuevos
- Almacenar y recuperar estas dependencias se denomina **gestión de artefactos**
- Tradicionalmente tienes que configurar tu propio sistema de gestión de artefactos
- **CodeArtifact** es una **gestión de artefactos** segura, escalable y rentable para el desarrollo de software
- Funciona con herramientas comunes de gestión de dependencias como Maven, Gradle, npm, yarn, twine, pip y NuGet
- **Los desarrolladores y CodeBuild pueden recuperar las dependencias directamente desde CodeArtifact**

AWS CodeArtifact



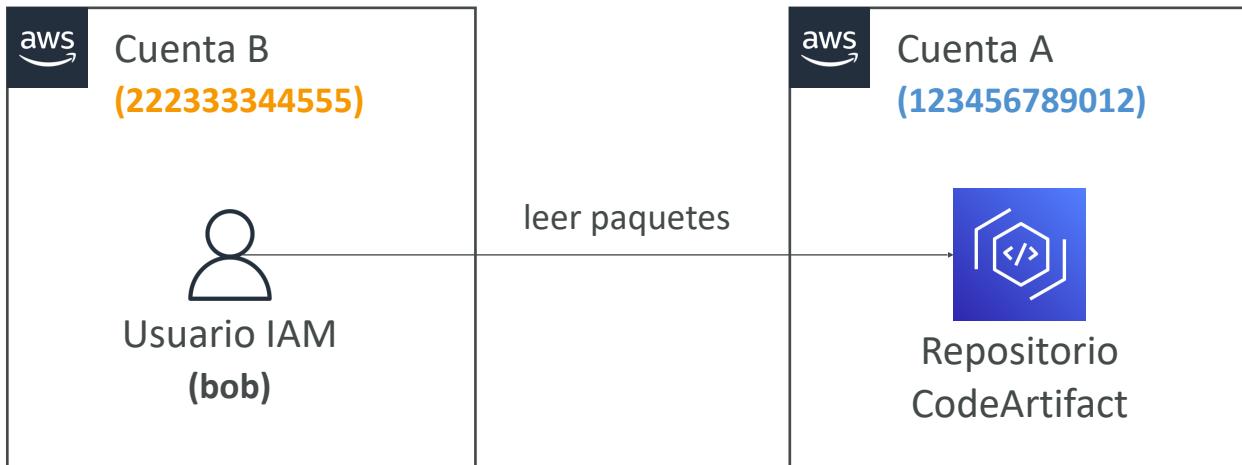
CodeArtifact - Integración con EventBridge

Se crea un evento cuando se crea, modifica o elimina una versión de un paquete



CodeArtifact - Política de recursos

- Puede utilizarse para autorizar a otra cuenta a acceder a CodeArtifact
- Una entidad de seguridad determinada puede leer todos los paquetes de un repositorio o ninguno de ellos

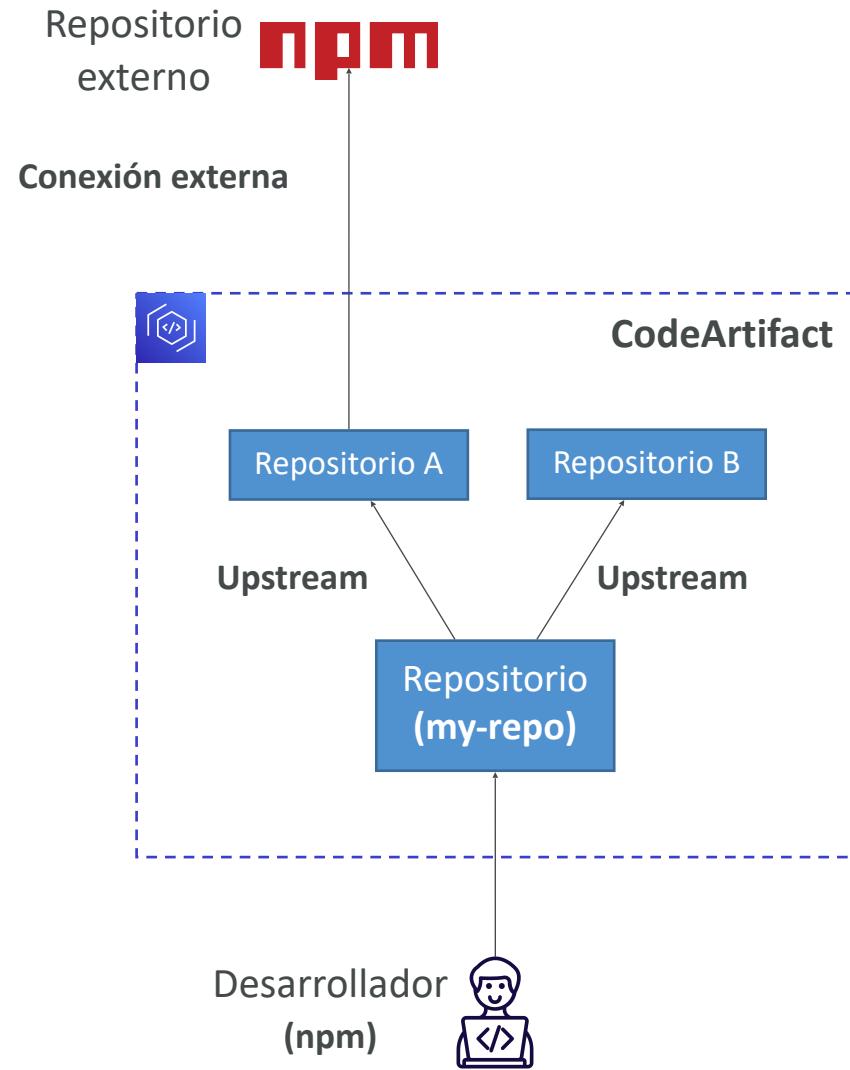


```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "codeartifact:DescribePackageVersion",
        "codeartifact:DescribeRepository",
        "codeartifact:GetPackageVersionReadme",
        "codeartifact:GetRepositoryEndpoint",
        "codeartifact>ListPackages",
        "codeartifact>ListPackageVersions",
        "codeartifact>ListPackageVersionAssets",
        "codeartifact>ListPackageVersionDependencies",
        "codeartifact:ReadFromRepository"
      ],
      "Principal": {
        "AWS": [
          "arn:aws:iam::123456789012:root",
          "arn:aws:iam::222333344555:user/bob"
        ]
      },
      "Resource": "*"
    }
  ]
}
```

Política de recursos del repositorio

CodeArtifact - Repositorios Upstream

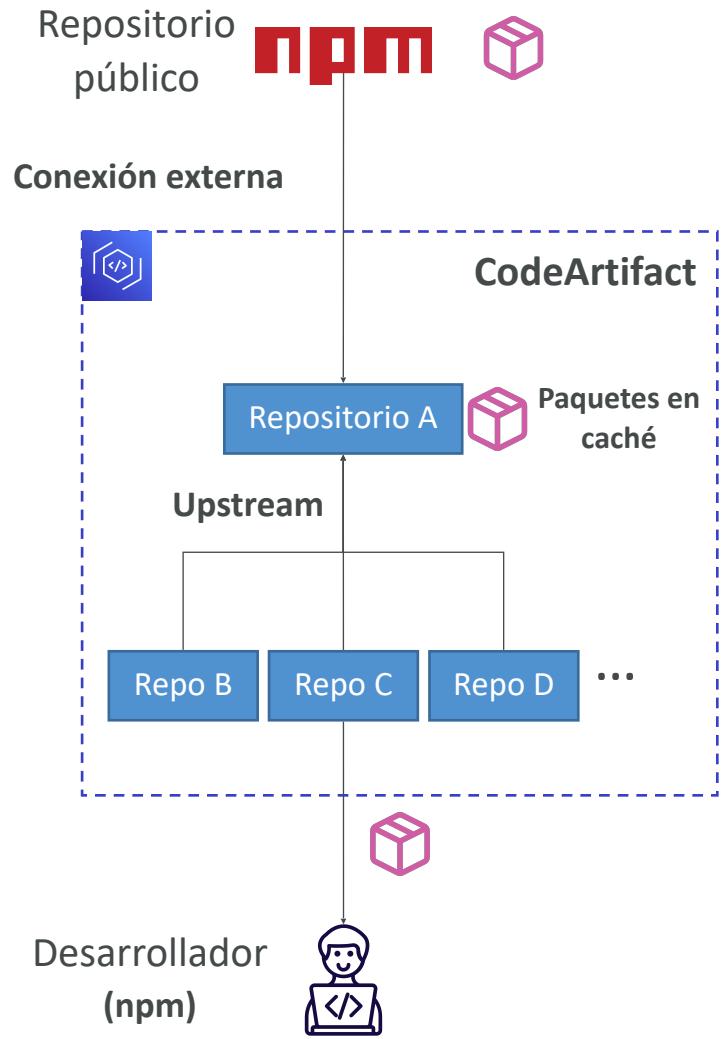
- Un repositorio CodeArtifact puede tener otros repositorios CodeArtifact como **repositorios ascendentes / upstream**
- Permite a un cliente gestor de paquetes acceder a los paquetes contenidos en más de un repositorio utilizando un único endpoint de repositorio
- Hasta 10 repositorios ascendentes
- Sólo una conexión externa



CodeArtifact - Conexión externa

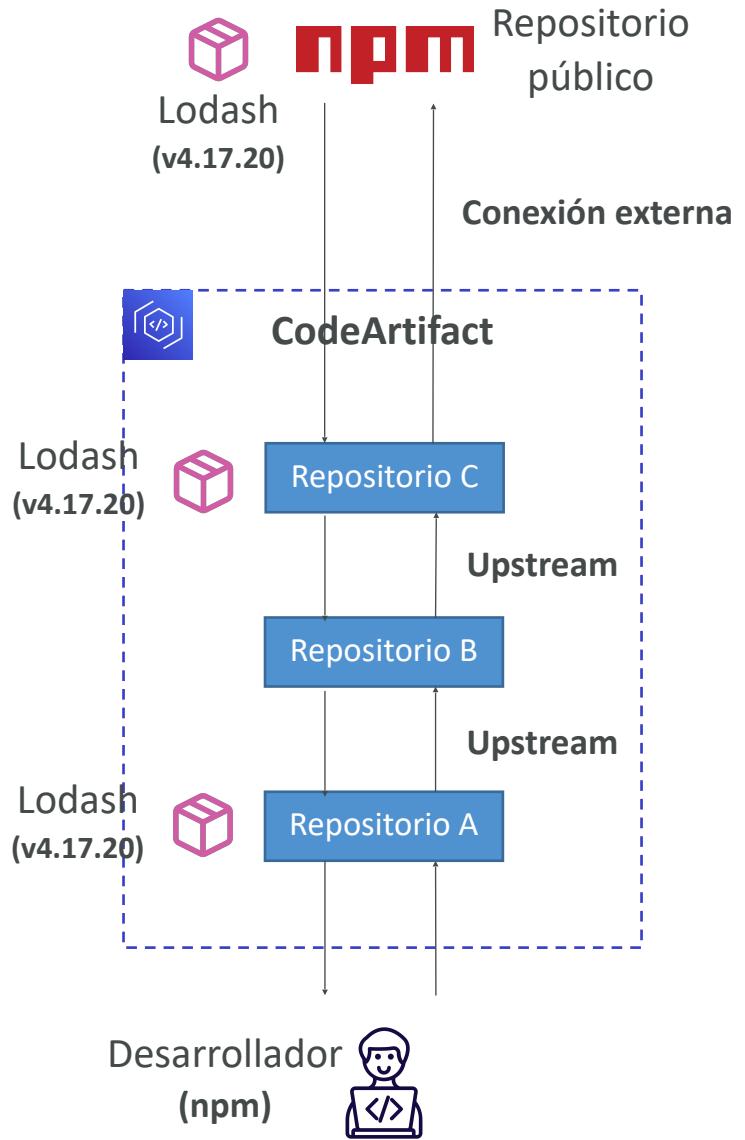
- Una conexión externa es una conexión entre un repositorio CodeArtifact y un repositorio externo/público (por ejemplo, Maven, npm, PyPI, NuGet...)
- Te permite obtener paquetes que aún no están presentes en tu repositorio CodeArtifact.
- Un repositorio tiene como máximo 1 conexión externa
- Crea muchos repositorios para muchas conexiones externas

- **Ejemplo - Conexión a npmjs.com**
 - Configura un repositorio CodeArtifact en tu dominio con una conexión externa a npmjs.com
 - Configura todos los demás repositorios con un upstream hacia él
 - Los paquetes obtenidos de npmjs.com se almacenan en caché en el repositorio Upstream, en lugar de obtenerlos y almacenarlos en cada repositorio.



CodeArtifact - Retención

- Si una versión de paquete solicitada se encuentra en un repositorio upstream, se conserva una referencia a ella y siempre está disponible en el repositorio downstream
- La versión del paquete conservada no se ve afectada por cambios en el repositorio Upstream (borrado, actualización del paquete, ...)
- Los repositorios intermedios no conservan el paquete
- **Ejemplo - Obtener un paquete de npmjs.com**
 - El gestor de paquetes conectado al repositorio A solicita el paquete **Lodash v4.17.20**
 - La versión del paquete no está presente en ninguno de los tres repositorios
 - La versión del paquete se obtendrá de npmjs.com
 - Cuando se obtenga **Lodash 4.17.20**, se conservará en:
 - **Repositorio A** - el repositorio más descendente
 - **Repositorio C** - tiene la conexión externa a npmjs.com
 - La versión del paquete no se conservará en el repositorio B, ya que es un repositorio intermedio



CodeArtifact – Dominios

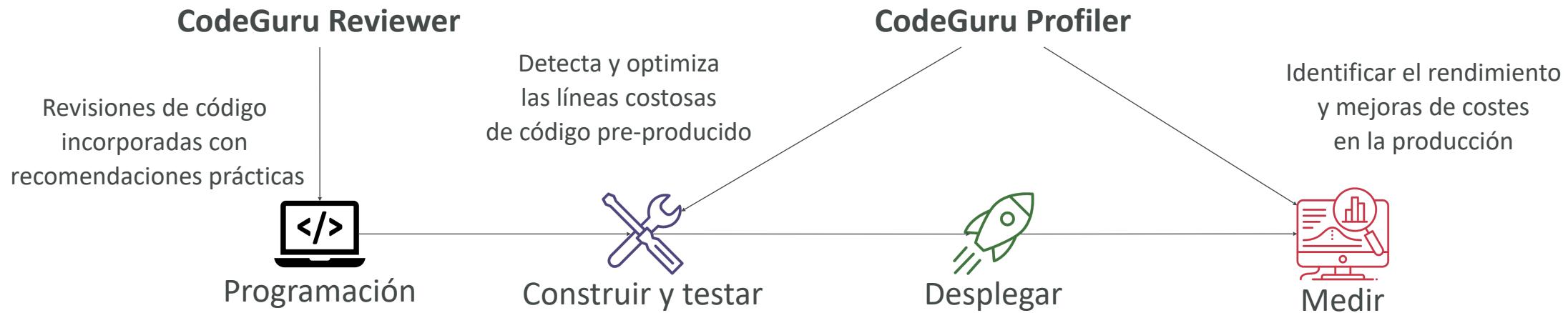
- **Almacenamiento deduplicado** - el activo sólo debe almacenarse una vez en un dominio, aunque esté disponible en muchos repositorios (sólo se paga una vez por el almacenamiento)
- **Copia rápida** - sólo se actualiza el registro de metadatos cuando se extraen paquetes de un repositorio Upstream CodeArtifact a un repositorio Downstream
- **Compartir fácilmente entre repositorios y equipos** - todos los activos y metadatos de un dominio se cifran con una única clave AWS KMS
- **Aplicar políticas en múltiples repositorios** - el administrador del dominio puede aplicar políticas en todo el dominio, como por ejemplo:
 - Restringir qué cuentas tienen acceso a los repositorios del dominio
 - Quién puede configurar conexiones a repositorios públicos para utilizarlos como fuentes de paquetes



Amazon CodeGuru



- Un servicio con tecnología ML para **revisiones de código automatizadas y recomendaciones sobre el rendimiento de las aplicaciones**
- Ofrece dos funcionalidades
 - **CodeGuru Reviewer**: revisiones de código automatizadas para el análisis estático del código (desarrollo)
 - **CodeGuru Profiler**: visibilidad/recomendaciones sobre el rendimiento de la aplicación durante el tiempo de ejecución (producción)



Amazon CodeGuru Reviewer

- Identifica problemas críticos, vulnerabilidades de seguridad y fallos difíciles de encontrar
- Ejemplo: mejores prácticas de codificación comunes, fugas de recursos, detección de seguridad, validación de entradas
- Utiliza el Machine Learning y el razonamiento automatizado
- Lecciones aprendidas a través de millones de revisiones de código en miles de repositorios de código abierto y de Amazon
- Soporta Java y Python
- Se integra con GitHub, Bitbucket y AWS CodeCommit

The screenshot shows the Amazon CodeGuru Reviewer interface. At the top, there's a navigation bar with 'CodeGuru' and 'Code reviews'. Below it, the repository name 'RepositoryAnalysis-amazon-codeguru-reviewer-sample-app-master-mw2tsa56o0000000' is displayed. The main section is titled 'Details' and includes information such as:

- Status: Completed
- Recommendations: 4
- Metered lines of code: 80
- Time created: 10 Nov 2020 08:08:47 AM GMT-0800
- Last updated: 10 Nov 2020 08:11:44 AM GMT-0800
- Type: RepositoryAnalysis
- Provider: GitHub
- Repository: amazon-codeguru-reviewer-sample-app
- Branch name: master

Below the details, there's a section for 'Recommendations (4)'. The first recommendation is for 'EventHandler.java Line: 79' with a note: 'This code appears to be waiting for a resource before it runs. You could use the waiters feature to help improve efficiency. Consider using ObjectExists or ObjectNotExists. For more information, see <https://aws.amazon.com/blogs/developer/waiters-in-the-aws-sdk-for-java/>'. There are 'Was this helpful?' upvote and downvote buttons.

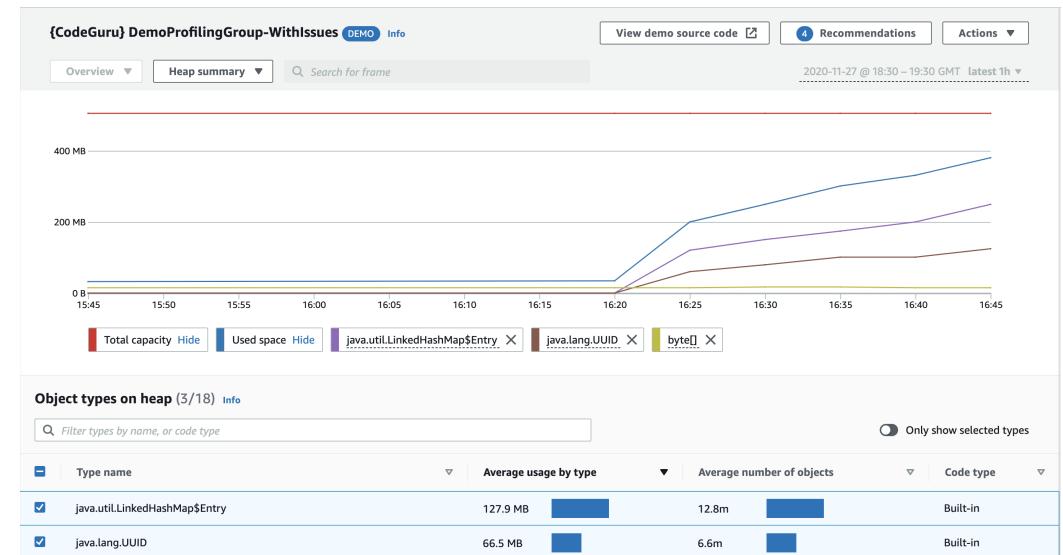
The second recommendation is for 'EventHandler.java Line: 100' with a note: 'This code might not produce accurate results if the operation returns paginated results instead of all results. Consider adding another call to check for additional results.' There are also 'Was this helpful?' upvote and downvote buttons.

The third recommendation is for 'EventHandler.java Line: 100' with a note: 'This code uses an outdated API. ListObjectsV2 is the revised List Objects API, and we recommend you use this revised API for new application developments.' There are 'Was this helpful?' upvote and downvote buttons.

<https://aws.amazon.com/codeguru/features/>

Amazon CodeGuru Profiler

- Ayuda a comprender el comportamiento en tiempo de ejecución de tu aplicación
- Ejemplo: identificar si tu aplicación está consumiendo una capacidad de CPU excesiva en una rutina de logs
- Funciones:
 - Identificar y eliminar las ineficiencias del código
 - Mejorar el rendimiento de la aplicación (por ejemplo, reducir la utilización de la CPU)
 - Disminuye los costes de computación
 - Proporciona un resumen de la pila (identifica los objetos que consumen memoria)
 - Detección de anomalías
- Admite aplicaciones que se ejecutan en AWS o en las instalaciones



<https://aws.amazon.com/codeguru/features/>

Amazon CodeGuru - Configuración del agente

- **MaxStackDepth** - profundidad máxima del CodeGuru Profiler.
 - Ejemplo: si CodeGuru Profiler encuentra un método A, que llama al método B, que llama al método C, que llama al método D, entonces la profundidad es 4
 - Si MaxStackDepth se establece en 2, entonces el perfilador evalúa A y B
- **MemoryUsageLimitPercent** - porcentaje máximo de la memoria disponible que el CodeGuru Profiler puede utilizar durante la ejecución
- **MinimumTimeForReportingInMilliseconds** - parámetro de configuración que establece el tiempo mínimo entre envíos de informes generados por el Profiler (milisegundos)
- **ReportingIntervalInMilliseconds** - intervalo de tiempo utilizado para reportar perfiles en CodeGuru (milisegundos)
- **SamplingIntervalInMilliseconds** - parámetro de configuración que determina el intervalo de muestreo utilizado para el perfilado de muestras en el CodeGuru Profiler (milisegundos)
 - Reducir para tener una frecuencia de muestreo más alta

AWS Cloud9



- Entorno de desarrollo integrado (IDE) basado en el Cloud
- Editor de código, depurador, terminal en un navegador
- Trabaja en tus proyectos desde cualquier lugar con conexión a Internet
- Preempaquetado con herramientas esenciales para los lenguajes de programación más populares (JavaScript, Python, PHP, ...)
- Comparte tu entorno de desarrollo con tu equipo (programación en parejas)
- Totalmente integrado con AWS SAM & Lambda para crear fácilmente aplicaciones sin servidor

AWS Cloud9 File Edit Find View Goto Run Tools Window Support Preview Run

```

claire:~/environment $ ls
Lambda Code  MyCodeCommitRepo2  NodeJS  python.1  Ruby  Untitled.1
MyCodeCommitRepo  MyCodeCommitRepo3  PHP  README.md  Untitled
claire:~/environment $ aws s3 mb s3://cloud9sample3
make_bucket: cloud9sample3
claire:~/environment $ aws s3 rb s3://cloud9sample3
remove_bucket: cloud9sample3
claire:~/environment $ git clone https://git-codecommit.us-west-2.amazonaws.com/v1/repos/MyCodeCommitRepo4
Cloning into 'MyCodeCommitRepo4'...
Username for 'https://git-codecommit.us-west-2.amazonaws.com': claire-at-56388640512
Password for 'https://clare-at-56388640512@git-codecommit.us-west-2.amazonaws.com':
warning: You appear to have cloned an empty repository.
claire:~/environment $ cd MyDemoCloud9Repo
bash: cd: MyDemoCloud9Repo: No such file or directory
claire:~/environment $ cd MyCodeCommitRepo
claire:~/environment/MyCodeCommitRepo (master) $ git status
On branch master

Initial commit

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)

    new file:  bird.txt
    new file:  insects.txt
    new file:  reptile.txt

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

    modified:   bird.txt
    modified:   reptile.txt

claire:~/environment/MyCodeCommitRepo (master) $ 
```

lambda_function.py

```

44 from base64 import b64decode
45 from urlparse import parse_qs
46
47
48
49
49 ENCRYPTED_EXPECTED_TOKEN = os.environ['']
50 kms = boto3.client('kms')
51 expected_token = kms.decrypt(CiphertextBlob=)
52
53
54 def respond(err, res=None):
55     return {
56         'statusCode': '400' if err else
57         'body': err.message if err else
58         'headers': {
59             'Content-Type': 'application/json'
60         },
61     }
62
63
64 def lambda_handler(event, context):
65     params = parse_qs(event['body'])
66     token = params['token'][0]
67     if token != expected_token:
68         logger.error("Request token (%s) does not match expected token (%s)" % (token, expected_token))
69     return respond(Exception("Invalid token"))
70
71 user = params['user_name'][0]
72 command = params['command'][0]
73 channel = params['channel_name'][0]
74 command_text = params['text'][0]
75
76 return respond(None, "%s invoked %s" % (user, command))

77

```

<https://aws.amazon.com/cloud9/>

AWS Serverless Application Model (SAM)

Lleva tu desarrollo sin servidor al siguiente nivel



AWS SAM



- **SAM = Modelo de aplicación sin servidor**
- Framework para desarrollar y desplegar aplicaciones sin servidor
- Toda la configuración es código YAML
- Genera CloudFormation complejo a partir de un sencillo archivo SAM YAML
- Soporta cualquier cosa de CloudFormation: Salidas, Mapeos, Parámetros, Recursos...
- Sólo dos comandos para desplegar en AWS
- SAM puede utilizar CodeDeploy para desplegar funciones Lambda
- SAM puede ayudarte a ejecutar Lambda, API Gateway, DynamoDB localmente

AWS SAM – Receta

- **Transform en la cabecera indica que es la plantilla SAM:**
 - Transform: 'AWS::Serverless-2016-10-31'
- **Escribe el código**
 - AWS::Serverless::Function
 - AWS::Serverless::Api
 - AWS::Serverless::SimpleTable
- **Empaquetar y desplegar (Package & Deploy):**
 - aws cloudformation package / sam package
 - aws cloudformation deploy / sam deploy

Inmersión profunda en la implantación de SAM



SAM - Depuración CLI

- Construye, testea y depura localmente tus aplicaciones sin servidor definidas mediante plantillas SAM de AWS
- Proporciona localmente un entorno de ejecución tipo lambda
- SAM CLI + AWS Toolkits => paso a paso y depura tu código
- Soporta IDEs: AWS Cloud9, Visual Studio Code, JetBrains, PyCharm, IntelliJ, ...
- **Kits de herramientas de AWS:**
Complementos IDE que te permiten construir, testar, depurar, desplegar e invocar funciones Lambda construidas con AWS SAM



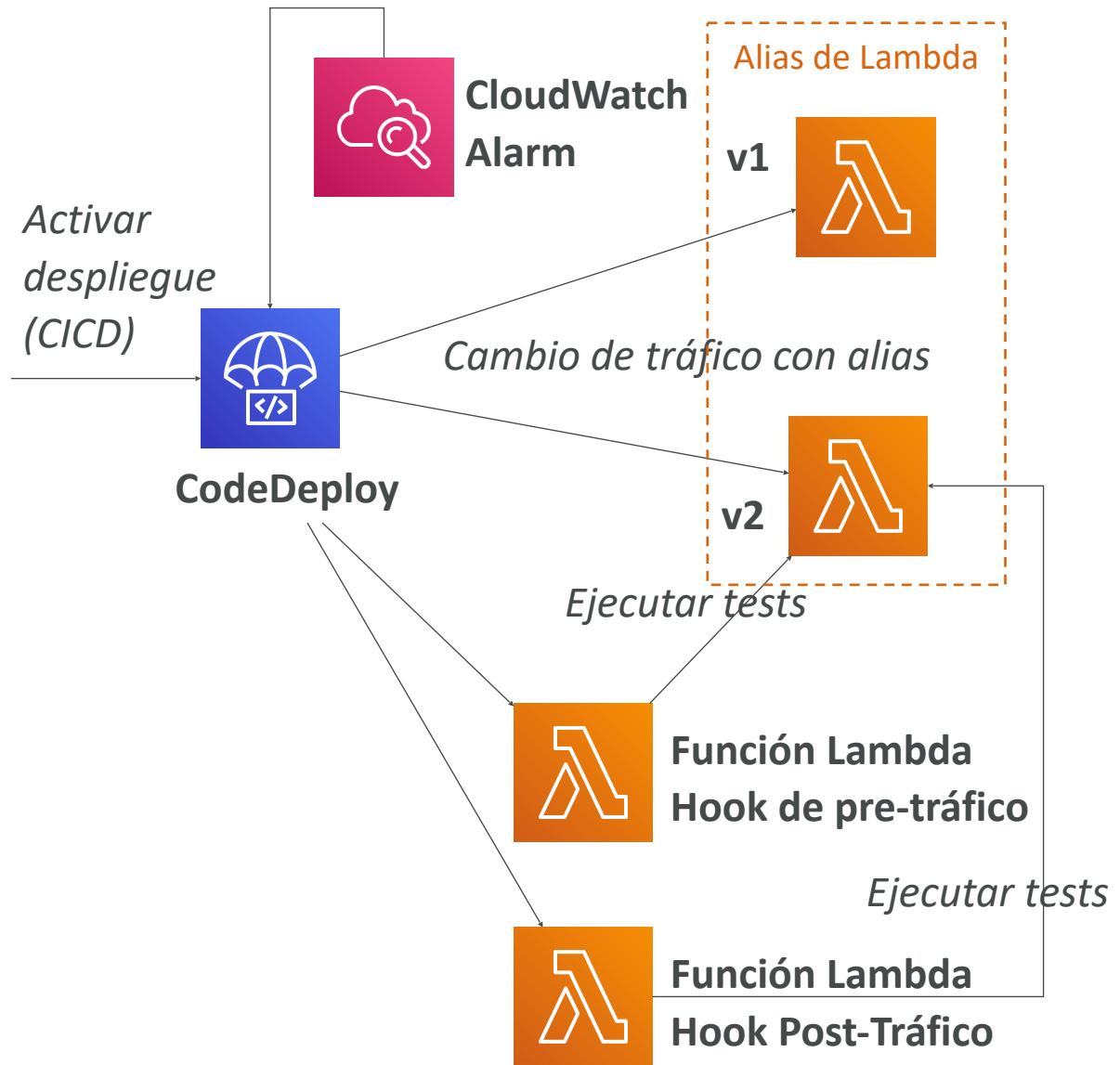
Plantillas de políticas SAM

- Lista de plantillas para aplicar permisos a tus funciones Lambda
- Lista completa disponible aquí: <https://docs.aws.amazon.com/serverless-application-model/latest/developerguide/serverless-policy-templates.html#serverless-policy-template-table>
- Ejemplos importantes:
 - **S3ReadPolicy**: Da permisos de sólo lectura a objetos en S3
 - **SQSPollerPolicy**: Permite sondear una cola SQS
 - **DynamoDBCrudPolicy**: CRUD = crear leer actualizar eliminar

```
MyFunction:
  Type: 'AWS::Serverless::Function'
  Properties:
    CodeUri: ${codeuri}
    Handler: hello.handler
    Runtime: python2.7
  Policies:
    - SQSPollerPolicy:
        QueueName:
          !GetAtt MyQueue.QueueName
```

SAM y CodeDeploy

- El framework SAM utiliza de forma nativa CodeDeploy para actualizar las funciones Lambda
- Función de cambio de tráfico
- Funciones Pre y Post traffic hooks para validar el despliegue (antes de que comience el desplazamiento de tráfico y después de que termine)
- Reversión fácil y automatizada mediante alarmas de CloudWatch



SAM - Capacidades locales

- **Iniciar localmente AWS Lambda**

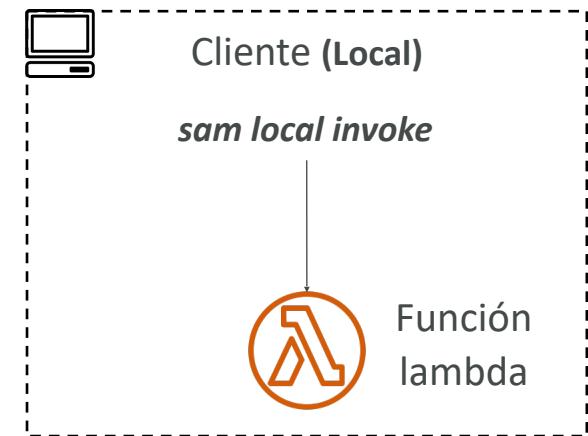
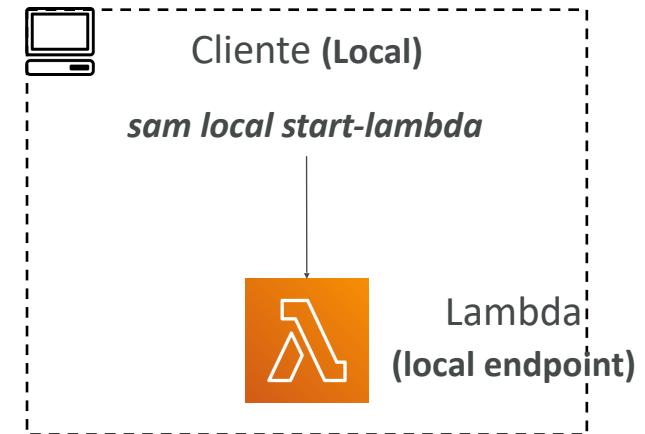
- ***sam local start-lambda***

- Inicia un endpoint local que emula AWS Lambda
 - Puedes ejecutar pruebas automatizadas contra este endpoint local

- **Invocar localmente una función Lambda**

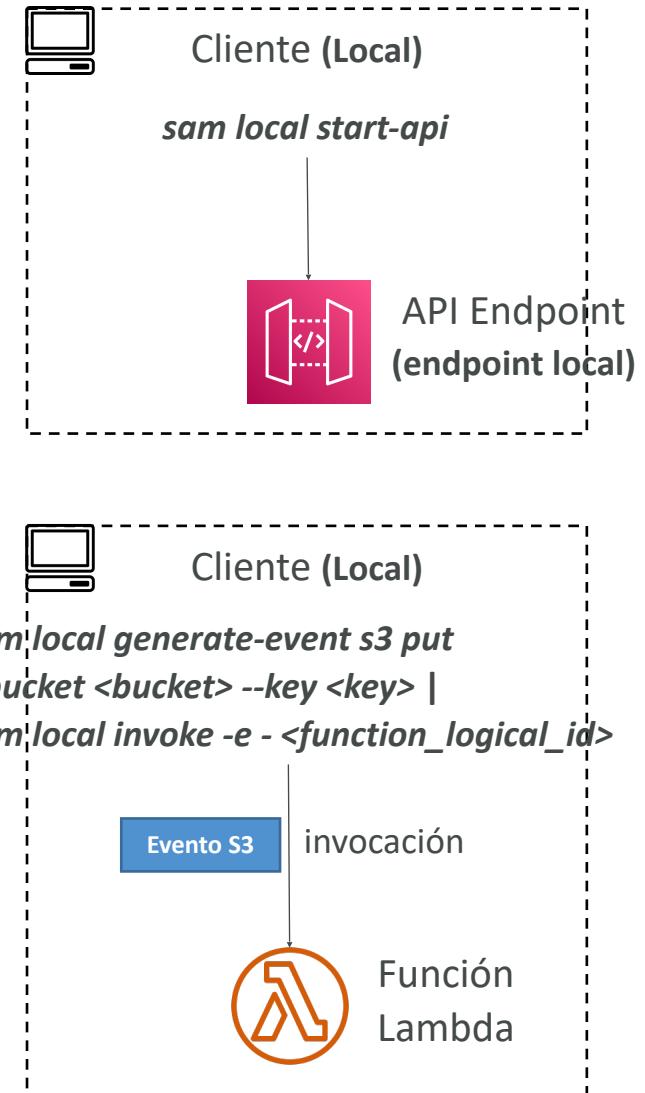
- ***sam local invoke***

- Invoca la función Lambda con la carga útil una vez y abandona después de completar la invocación
 - Útil para generar casos de test
 - Si la función realiza llamadas API a AWS, asegúrate de que estás utilizando la opción --profile correcta



SAM - Capacidades locales

- Iniciar localmente un endpoint de API Gateway
 - ***sam local start-api***
 - Inicia un servidor HTTP local que aloja todas tus funciones
 - Los cambios en las funciones se recargan automáticamente
- Generar eventos de AWS para funciones Lambda
 - ***sam local generate-event***
 - Genera cargas útiles de muestra para fuentes de eventos
 - S3, API Gateway, SNS, Kinesis, DynamoDB...



SAM - Resumen del examen



- SAM se basa en CloudFormation
- SAM requiere las secciones **Transform** y **Resources**
- Comandos que debes conocer
 - sam build: obtener dependencias y crear artefactos de despliegue locales
 - sam package: empaquetar y subir a Amazon S3, generar plantilla CF
 - sam deploy: despliega en CloudFormation
- Plantillas de política SAM para definir fácilmente la política IAM
- SAM se integra con CodeDeploy para desplegar en alias Lambda

Serverless Application Repository (SAR)



- Repositorio gestionado para aplicaciones sin servidor
- **Las aplicaciones se empaquetan utilizando SAM**
- Construye y publica aplicaciones que puedan ser reutilizadas por las organizaciones
 - Se pueden compartir públicamente
 - Pueden compartirse con cuentas específicas de AWS
- Esto evita duplicar el trabajo, y pasar directamente a la publicación
- La configuración y el comportamiento de la aplicación pueden personalizarse mediante **variables de entorno**



AWS Cloud Development Kit (CDK)

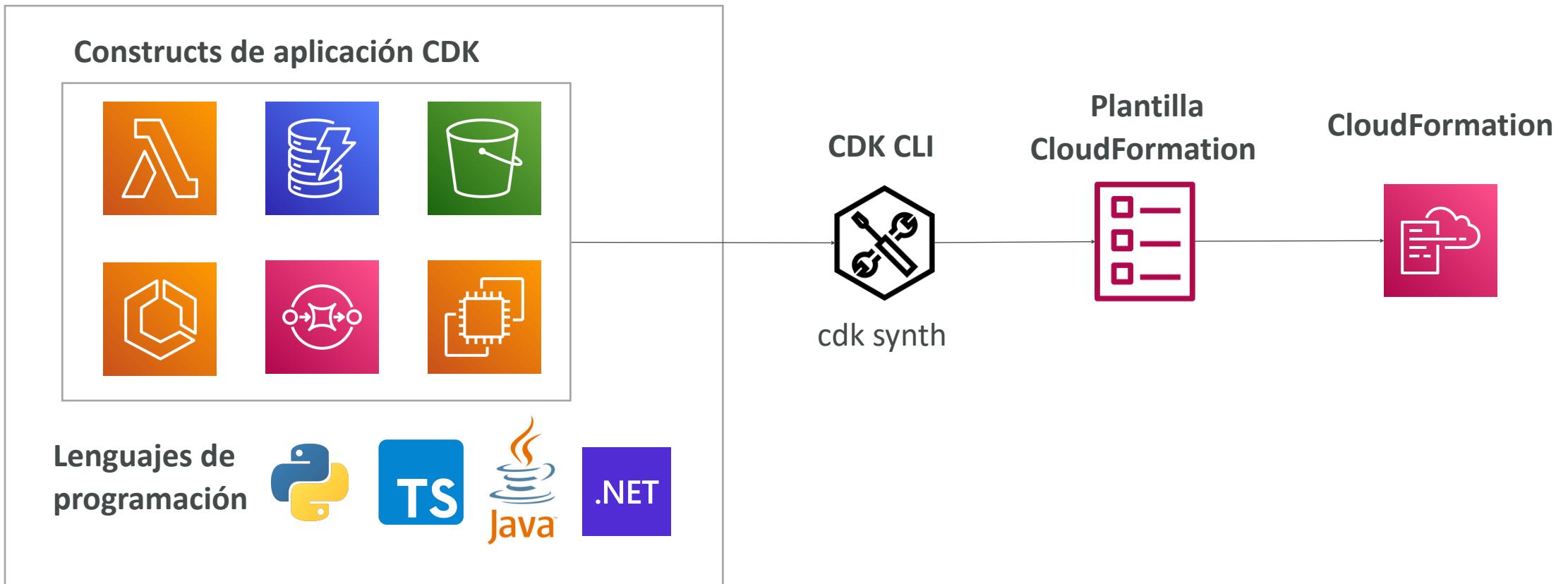
AWS Cloud Development Kit (CDK)



- Define tu infraestructura Cloud utilizando un lenguaje familiar:
 - JavaScript/TypeScript, Python, Java y .NET
- Contiene componentes de alto nivel llamados **constructs**
- El código se "compila" en una plantilla de CloudFormation (JSON/YAML)
- **Por tanto, puedes desplegar juntos la infraestructura y el código de tiempo de ejecución de la aplicación**
 - Ideal para funciones Lambda
 - Genial para contenedores Docker en ECS / EKS

```
export class MyEcsConstructStack extends core.Stack {  
  constructor(scope: core.App, id: string, props?: core.StackProps)  
    super(scope, id, props);  
  
  const vpc = new ec2.Vpc(this, "MyVpc", {  
    maxAzs: 3 // Default is all AZs in region  
  });  
  
  const cluster = new ecs.Cluster(this, "MyCluster", {  
    vpc: vpc  
  });  
  
  // Create a Load-balanced Fargate service and make it public  
  new ecs_patterns.ApplicationLoadBalancedFargateService(this, "My  
    cluster: cluster, // Required  
    cpu: 512, // Default is 256  
    desiredCount: 6, // Default is 1  
    taskImageOptions: { image: ecs.ContainerImage.fromRegistry("an  
      memoryLimitMiB: 2048, // Default is 512  
      publicLoadBalancer: true // Default is false  
    };  
  }  
}
```

CDK en un diagrama

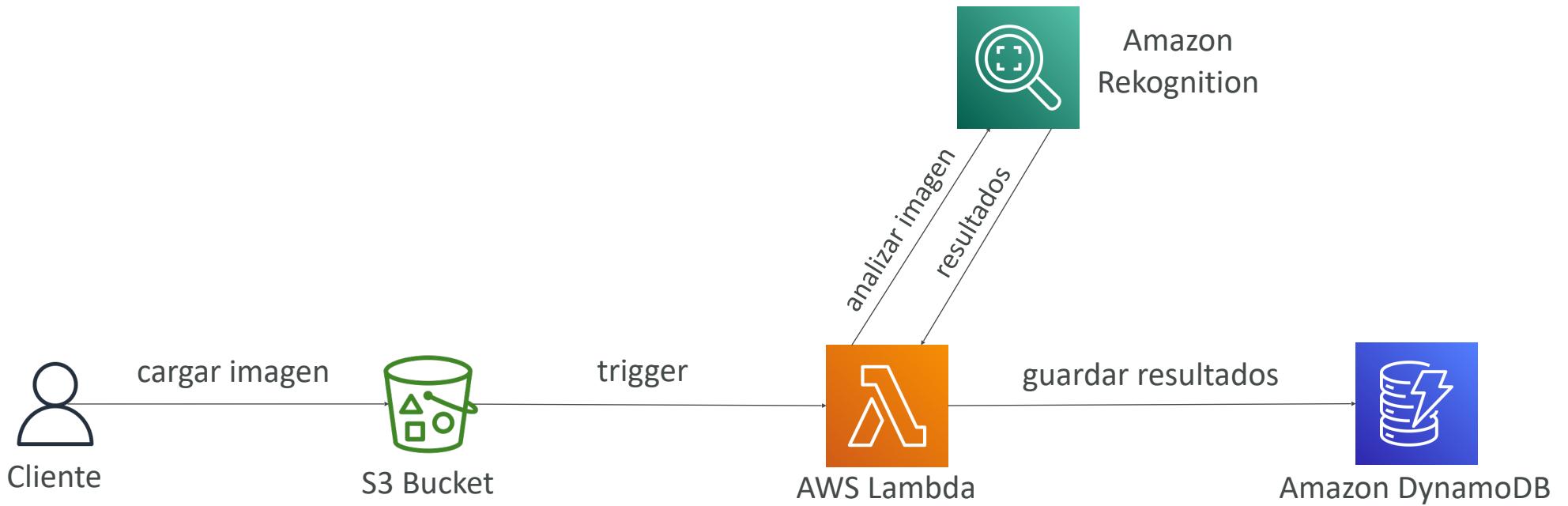


CDK vs SAM



- SAM:
 - Enfocado sin servidor
 - Escribe tu plantilla de forma declarativa en JSON o YAML
 - Ideal para empezar rápidamente con Lambda
 - Aprovecha CloudFormation
- CDK:
 - Todos los servicios de AWS
 - Escribe infra en un lenguaje de programación JavaScript/TypeScript, Python, Java y .NET
 - Aprovecha CloudFormation

CDK - Práctica



Constructs CDK

- Un Construct de CDK es un componente que encapsula todo lo que CDK necesita para crear el stack final de CloudFormation
- Puede representar un único recurso de AWS (por ejemplo, un bucket de S3) o varios recursos relacionados (por ejemplo, una cola de workers con computación)
- **Biblioteca de constructs de AWS**
 - Una colección de constructs incluida en AWS CDK que contiene constructs para cada recurso de AWS
 - Contiene 3 niveles diferentes de constructs disponibles (L1, L2, L3)
- **Construct Hub** - contiene constructs adicionales de AWS, de terceros y de la comunidad CDK de código abierto

Constructs CDK - Constructs de Capa I (LI)

- Puede llamarse **Recursos CFN** que representa todos los recursos directamente disponibles en CloudFormation
- Los constructs se generan periódicamente a partir de la **especificación de recursos de CloudFormation**
- Los nombres de los constructs empiezan por **Cfn** (por ejemplo, **CfnBucket**)
- **Debes configurar explícitamente todas las propiedades de los recursos**

```
const bucket = new s3.CfnBucket(this, "MyBucket", {  
    bucketName: "MyBucket"  
});
```

Constructs CDK - Constructs de Capa 2 (L2)

- Representa los recursos de AWS pero con un nivel superior
- Funcionalidad similar a L1 pero con cómodos valores por defecto
 - No necesitas conocer todos los detalles sobre las propiedades del recurso
- Proporcionan métodos que simplifican el trabajo con el recurso (por ejemplo, **bucket.addLifeCycleRule()**)

```
const s3 = require('aws-cdk-lib/aws-s3');

const bucket = new s3.Bucket(this, 'MyBucket', {
    versioned: true,
    encryption: s3.BucketEncryption.KMS
});

// Returns the HTTPS URL of an S3 Object
const objectUrl = bucket.urlForObject('MyBucket/MyObject');
```

Constructs CDK – Constructs de Capa 3 (L3)

- Puede llamarse **Patrones**, que representan múltiples recursos relacionados
- Te ayuda a completar tareas comunes en AWS
- Ejemplos:
 - **aws-apigateway.LambdaRestApi** representa una API Gateway respaldada por una función Lambda
 - **aws-ecs-patterns.ApplicationLoadBalancerFargateService** que representa una arquitectura que incluye un Cluster Fargate con Application Load Balancer

```
const api = new apigateway.LambdaRestApi(this, 'myapi', {
  handler: backend,
  proxy: false
});

const items = api.root.addResource('items');
items.addMethod('GET'); // GET /items
items.addMethod('POST'); // POST /items

const item = items.addResource('{item}');
item.addMethod('GET'); // GET /items/{item}

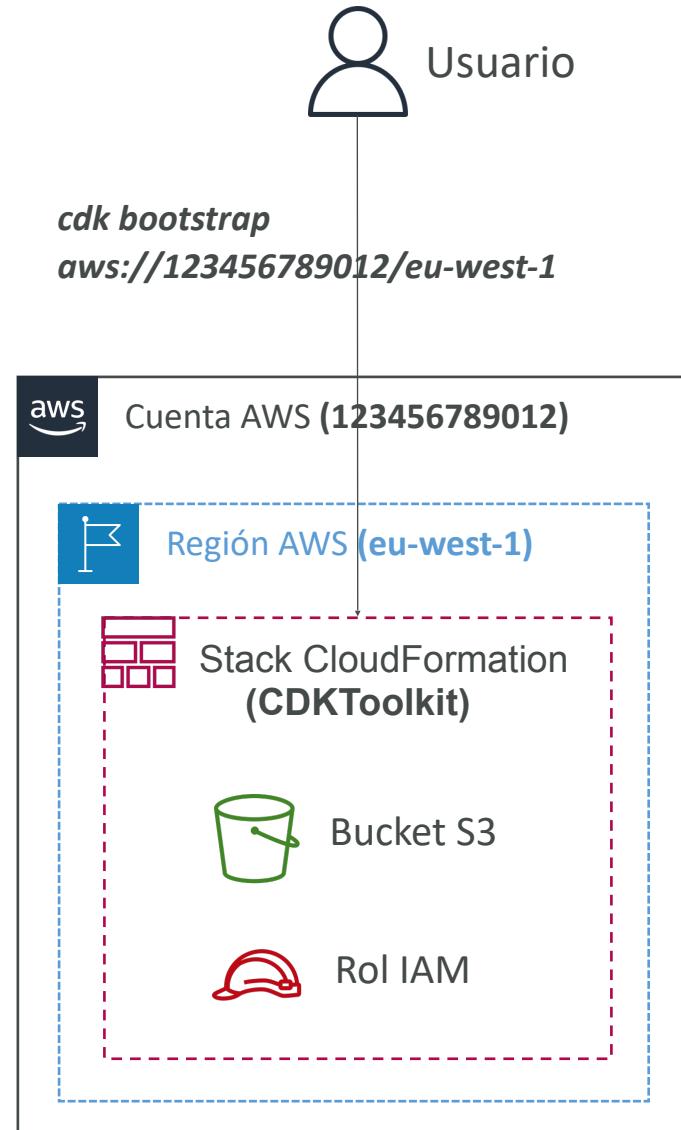
item.addMethod('DELETE', new apigateway.HttpIntegration('http://amazon.com'));
```

CDK - Comandos importantes que debes conocer

Comando	Descripción
npm install -g aws-cdk-lib	Instala la CLI y las bibliotecas CDK
cdk init app	Crea un nuevo proyecto CDK a partir de una plantilla especificada
cdk synth	Sintetiza e imprime la plantilla de CloudFormation
cdk bootstrap	Despliega el stack del kit de herramientas CDK
cdk deploy	Despliega el/los stack(s)
cdk diff	Ver las diferencias entre el CDK local y el stack desplegado
cdk destroy	Destruye el/los stack(s)

CDK – Bootstrapping

- El proceso de aprovisionamiento de recursos para CDK antes de que puedas implementar aplicaciones CDK en un entorno AWS
- **Entorno AWS = cuenta y región**
- Se crea el stack de CloudFormation llamado CDKToolkit y contiene:
 - **S3 Bucket** - para almacenar archivos
 - **Roles IAM** - para conceder permisos para realizar despliegues
- Debes ejecutar el siguiente comando para cada nuevo entorno:
 - `cdk bootstrap aws://<aws_account>/<aws_region>`
 - De lo contrario, obtendrás un **error "La política contiene una declaración con uno o más principales no válidos"**



CDK – Testing

- Para testear aplicaciones CDK, utiliza el **Módulo de Aserciones CDK** combinado con populares frameworks de test como Jest (JavaScript) o Pytest (Python)
- Comprueba que tenemos recursos específicos, reglas, condiciones, parámetros...
- Dos tipos de test:
 - **Aserciones de grano fino (comunes)** - testean aspectos específicos de la plantilla de CloudFormation (por ejemplo, comprobar si un recurso tiene esta propiedad con este valor)
 - **Tests de Snapshot** - testea la plantilla CloudFormation sintetizada frente a una plantilla de referencia almacenada previamente
- Para importar una plantilla
 - `Template.fromStack(MyStack)` : stack construido en CDK
 - `Template.fromString(mystring)` : stack construido fuera de CDK

```

describe("StateMachineStack", () => {
  test("synthesizes the way we expect", () => {
    ...
    // Prepare the stack for assertions
    const template = Template.fromStack(MyStack);

    // Assert it creates Lambda with correct properties...
    template.hasResourceProperties("AWS::Lambda::Function", {
      Handler: "handler",
      Runtime: "nodejs14.x",
    });
    // Fine-grained Assertions

    // Assert it creates the SNS subscription...
    template.resourceCountIs("AWS::SNS::Subscription", 1);

    // Assert the synthesized CloudFormation template
    // against a previously stored baseline template
    expect(template.toJSON()).toMatchSnapshot();
  });
});
```

Test de Snapshot

Amazon Cognito

Amazon Cognito

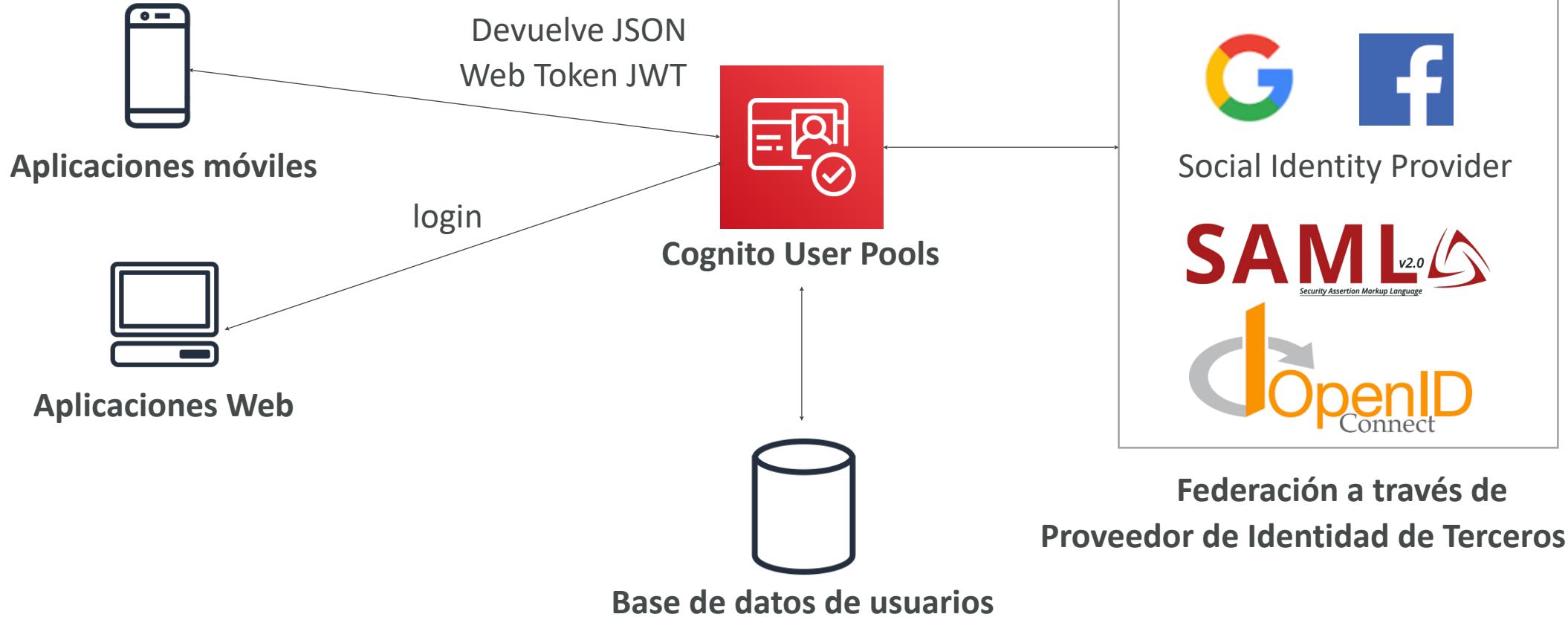


- Dar a los usuarios una identidad para interactuar con nuestra aplicación web o móvil
- **Grupos de usuarios Cognito (Cognito User Pools):**
 - Funcionalidad de inicio de sesión para usuarios de aplicaciones
 - Integración con API Gateway y Application Load Balancer
- **Cognito Identity Pools (Identidad Federada):**
 - Proporciona credenciales de AWS a los usuarios para que puedan acceder directamente a los recursos de AWS
 - Integrar con Cognito User Pools como proveedor de identidades
- **Cognito vs IAM:** "cientos de usuarios", "usuarios móviles", "autenticar con SAML"

Cognito User Pools (CUP) – Funciones de usuario

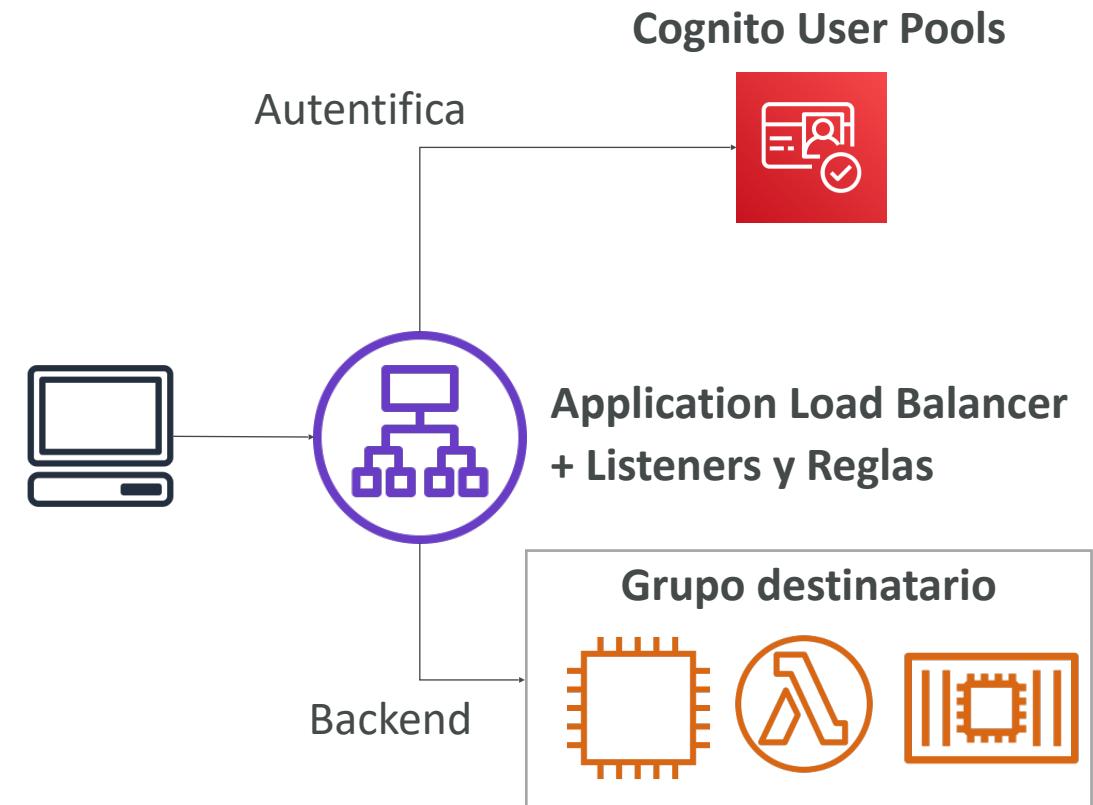
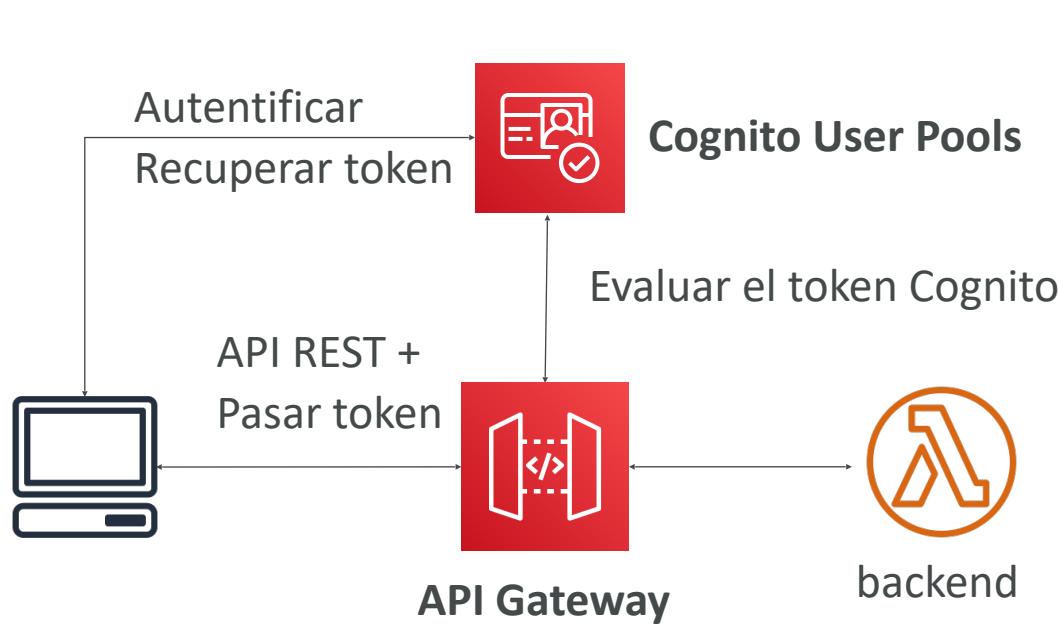
- **Crea una base de datos de usuarios sin servidor para tus aplicaciones web y móviles**
- Inicio de sesión sencillo: Combinación de nombre de usuario (o correo electrónico) / contraseña
- Restablecimiento de contraseña
- Verificación de correo electrónico y número de teléfono
- Autenticación multifactor (MFA)
- Identidades federadas: usuarios de Facebook, Google, SAML...
- Característica: bloquear usuarios si sus credenciales están comprometidas en otro lugar
- El inicio de sesión devuelve un Token Web JSON (JWT)

Cognito User Pools (CUP) – Diagrama



Cognito User Pools (CUP) - Integraciones

- CUP se integra con **API Gateway** y **Application Load Balancer (ALB)**



Cognito User Pools – Triggers de Lambda

- CUP puede invocar una función Lambda de forma sincrónica sobre estos activadores:

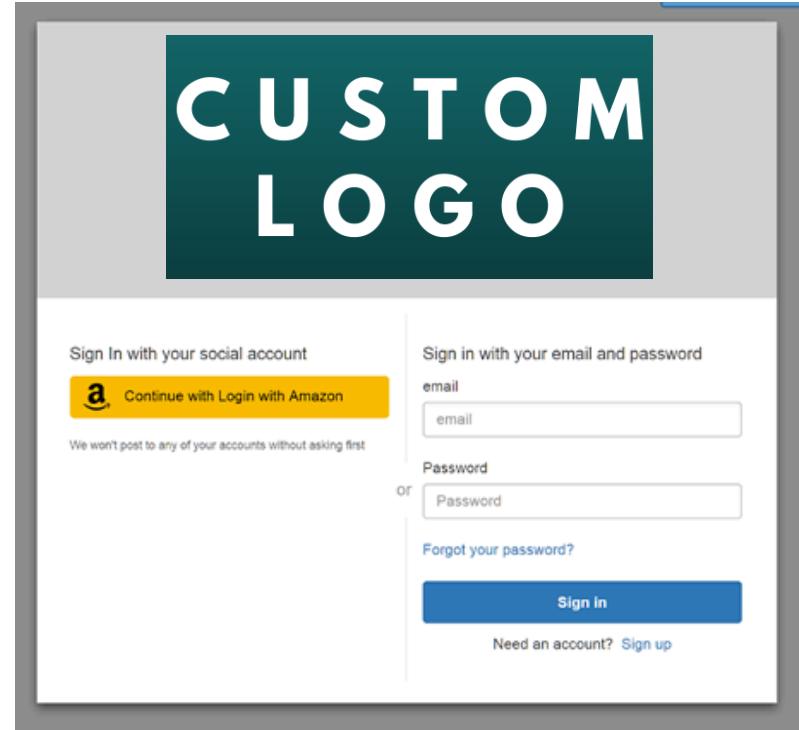
Flujo de User Pool	Operación	Descripción
Eventos de autenticación	Trigger Lambda de preautenticación	Validación personalizada para aceptar o denegar la petición de inicio de sesión
	Trigger Lambda Post autenticación	Logs de eventos para análisis personalizados
	Trigger Lambda previo a la generación del token	Aumentar o suprimir las reclamaciones de token
Sign-Up / Inscripción	Trigger Lambda previo a la inscripción	Validación personalizada para aceptar o denegar la petición de inscripción
	Trigger lambda postconfirmación	Mensajes de bienvenida personalizados o logs de eventos para análisis personalizados
	Migrar el trigger Lambda de usuario	Migrar un usuario de un directorio de usuarios existente a grupos de usuarios
Mensajes	Mensaje personalizado Trigger Lambda	Personalización avanzada y localización de mensajes
Creación de token	Pre Generación de token Trigger Lambda	Añadir o eliminar atributos en tokens de Id

<https://docs.aws.amazon.com/cognito/latest/developerguide/cognito-user-identity-pools-working-with-aws-lambda-triggers.html>

Cognito User Pools

Interfaz de usuario de autenticación alojada

- Cognito tiene una **interfaz de usuario de autenticación alojada** que puedes añadir a tu aplicación para gestionar los flujos de trabajo de registro e inicio de sesión.
- Utilizando la IU alojada, tienes una base para la integración con inicios de sesión sociales, OIDC o SAML
- Se puede **personalizar mediante un logotipo y un CSS**



<https://aws.amazon.com/blogs/aws/launch-amazon-cognito-user-pools-general-availability-app-integration-and-federation/>

CUP - Dominio personalizado de IU alojada

- Para los dominios personalizados, debes crear un certificado ACM en us-east-1
- El dominio personalizado debe definirse en la sección "Integración de aplicaciones" (App Integration en inglés)

The screenshot shows the AWS Cognito User Pool configuration interface. The top navigation bar includes tabs for Users, Groups, Sign-in experience, Sign-up experience, Messaging, App integration (which is underlined in blue), and User pool properties. Below the navigation, a section titled 'Configuration for all app clients' is displayed. It contains a 'Domain Info' card with a description of configuring domains for Hosted UI and OAuth 2.0 endpoints. The 'Cognito domain' section shows a 'Domain' field with the value 'https://demo-alb.auth.eu-central-1.amazoncognito.com'. The 'Custom domain' section shows a 'Domain' field with a single dash '-' listed. An 'Actions' dropdown menu is visible on the right side of the domain info card.

CUP - Autenticación adaptativa

- **Bloquea los inicios de sesión o exige MFA si el inicio de sesión parece sospechoso**
- Cognito examina cada intento de inicio de sesión y genera una puntuación de riesgo (bajo, medio, alto) según la probabilidad de que la petición de inicio de sesión proceda de un atacante malintencionado
- Sólo se solicita a los usuarios una segunda MFA cuando se detecta un riesgo.
- La puntuación de riesgo se basa en distintos factores, como si el usuario ha utilizado el mismo dispositivo, ubicación o dirección IP
- Comprueba si hay credenciales comprometidas, protección contra la toma de control de cuentas y verificación por teléfono y correo electrónico
- Integración con logs de CloudWatch (intentos de inicio de sesión, puntuación de riesgo, desafíos fallidos...)



Descodificación de un token de identificación; JWT - JSON Web Token

- CUP emite tokens JWT (codificados en Base64):
 - **Header / Cabecera**
 - **Payload / Carga útil**
 - **Signature / Firma**
- **La firma debe verificarse para garantizar que se puede confiar en el JWT**
- Las bibliotecas pueden ayudarte a verificar la validez de los tokens JWT emitidos por Cognito User Pools
- El Payload contendrá la información del usuario (sub UUID, given_name, email, phone_number, atributos...)
- A partir del sub UUID, puedes recuperar todos los detalles de los usuarios de Cognito / OIDC

```
<header>.  
{  
    "sub": "aaaaaaaa-bbbb-cccc-dddd-eeeeeeeeeee", User ID en Cognito DB  
    "email": "my-test-user@example.com",  
    "email_verified": true,  
    "middle_name": "Jane",  
    "cognito:username": "my-test-user",  
    "cognito:groups": [  
        "my-test-group"  
    ],  
    "cognito:roles": [  
        "arn:aws:iam::111122223333:role/my-test-role"  
    ],  
    "cognito:preferred_role": "arn:aws:iam::111122223333:role/my-test-role",  
    "iss": "https://cognito-idp.us-west-2.amazonaws.com/us-west-2_example",  
    "nonce": "abcdefg",  
    "origin_jti": "aaaaaaaa-bbbb-cccc-dddd-eeeeeeeeeee",  
    "aud": "xxxxxxxxxxxxexample",  
    "event_id": "64f513be-32db-42b0-b78e-b02127b4f463",  
    "token_use": "id",  
    "auth_time": 1676312777, Caducidad y fecha de emisión  
    "exp": 1676316377,  
    "iat": 1676312777,  
    "jti": "aaaaaaaa-bbbb-cccc-dddd-eeeeeeeeeee",  
}  
.<token signature>
```

ID JWT Token Payload

Application Load Balancer

Autenticación de usuarios

- Tu Application Load Balancer puede autenticar de forma segura a los usuarios
 - Descarga el trabajo de autenticar usuarios a tu Load Balancer
 - Tus aplicaciones pueden centrarse en su lógica de negocio
- Autentica a los usuarios a través de:
 - Proveedor de identidades (IdP): Compatible con OpenID Connect (OIDC)
 - Cognito User Pools:
 - IdP sociales, como Amazon, Facebook o Google
 - Identidades corporativas mediante SAML, LDAP o Microsoft AD
- **Debes utilizar una escucha HTTPS para establecer las reglas authenticate-oidc & authenticate-cognito**
- **OnUnauthenticatedRequest** - autenticar (por defecto), denegar, permitir

Listener details

A listener is a process that checks for connection determine how the load balancer routes request

Protocol	Port
HTTPS ▾	: 443 1-65535

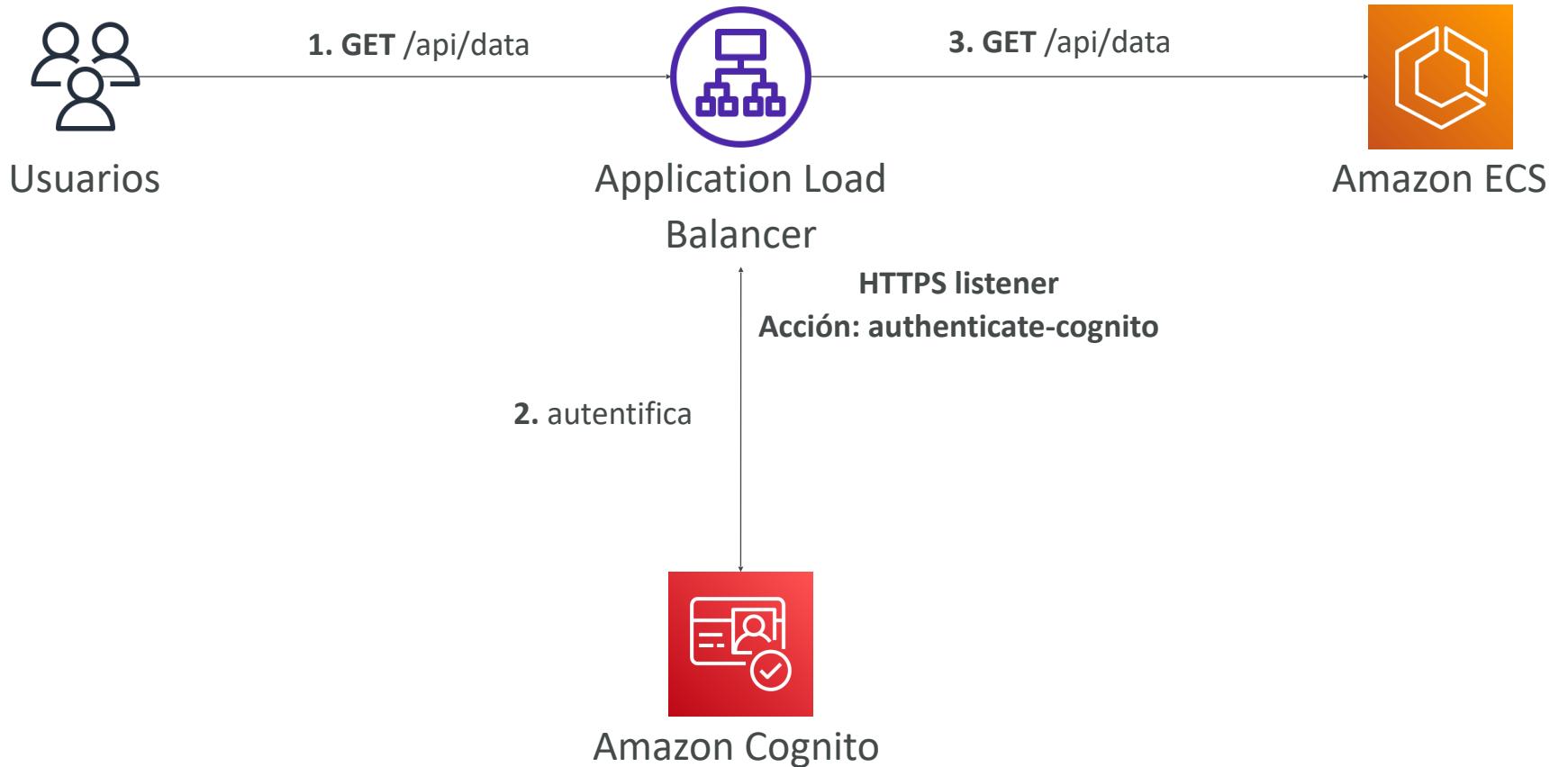
Default actions [Info](#)

Specify the default actions for traffic on this listener. Rules can be configured after the listener is created.

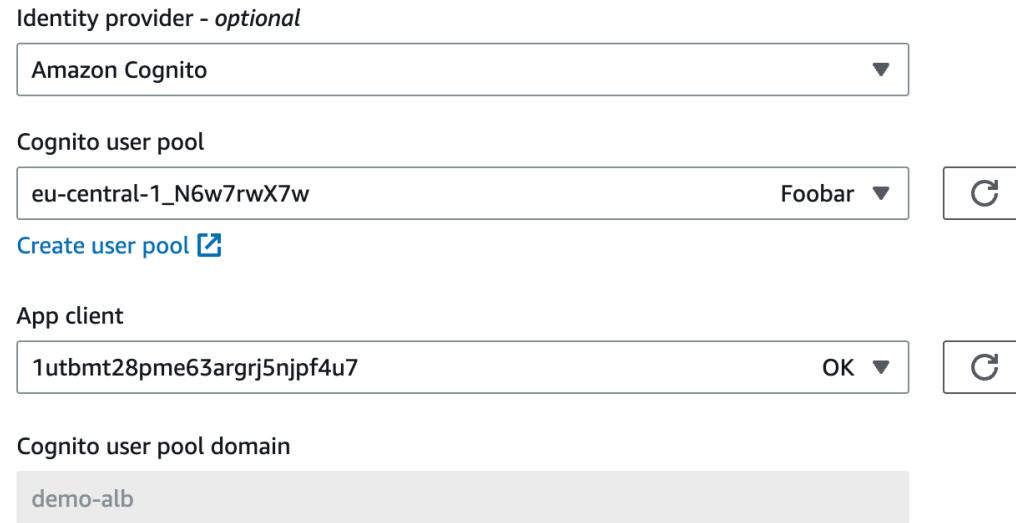
► 1. Authenticate [Info](#)

► 2. Forward to [Info](#)

Application Load Balancer – Auth. de Cognito

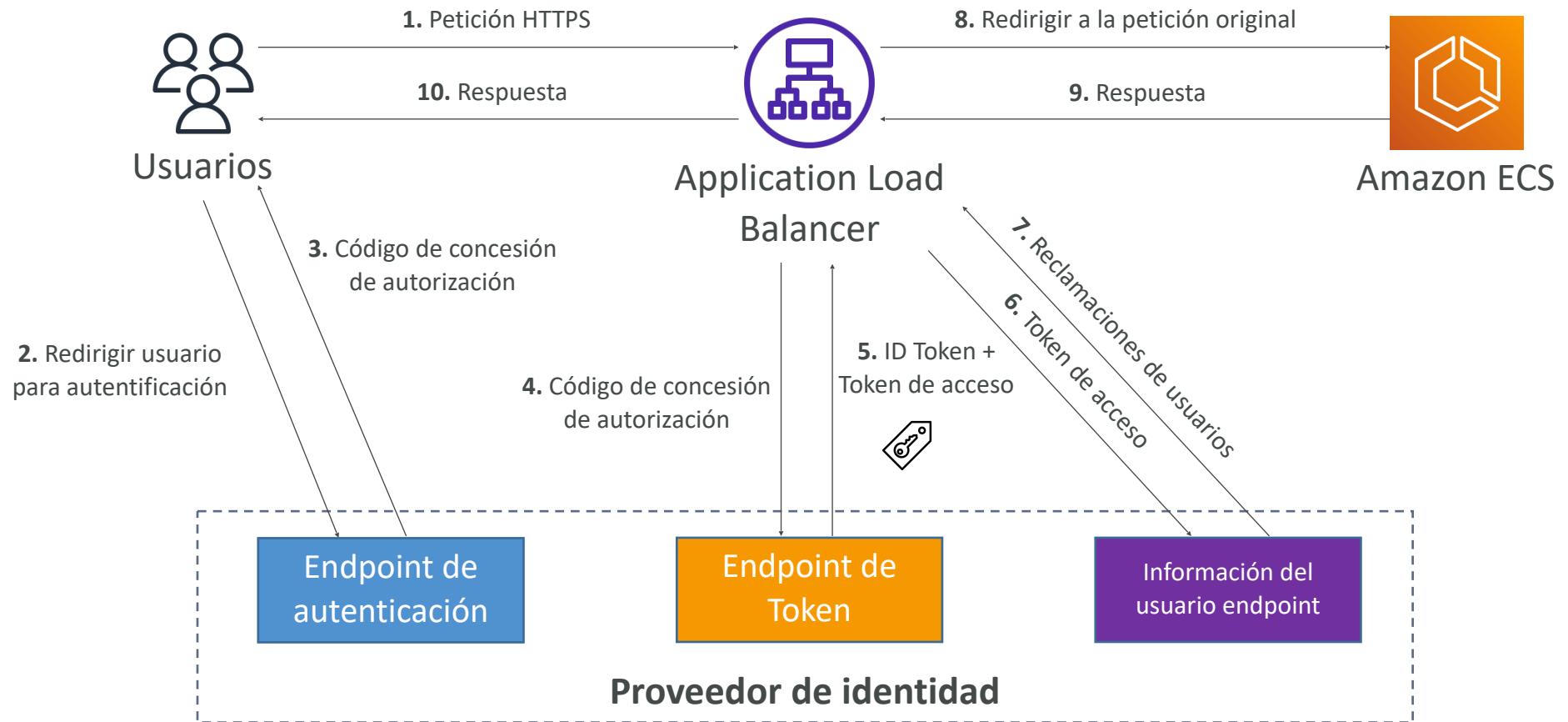


ALB - Autenticación a través de Cognito User Pools



- Crear Cognito User Pools, Cliente y Dominio
- Asegúrate de que se devuelve un token de identificación
- Añade el IdP social o corporativo si es necesario
- Son necesarias varias redirecciones de URL
- Permite el dominio de tu Cognito User Pool en la URL de devolución de llamada de tu aplicación IdP. Por ejemplo:
 - **https://domain-prefix.auth.region.amazoncognito.com/saml2/idpresponse**
 - **https://user-pool-domain/oauth2/idpresponse**

Application Load Balancer – Autenticación OIDC



ALB - Autenticación a través de un proveedor de identidad (IdP) compatible con OpenID Connect (OIDC)

Identity provider - optional

OIDC

Issuer
Enter the OpenID provider.

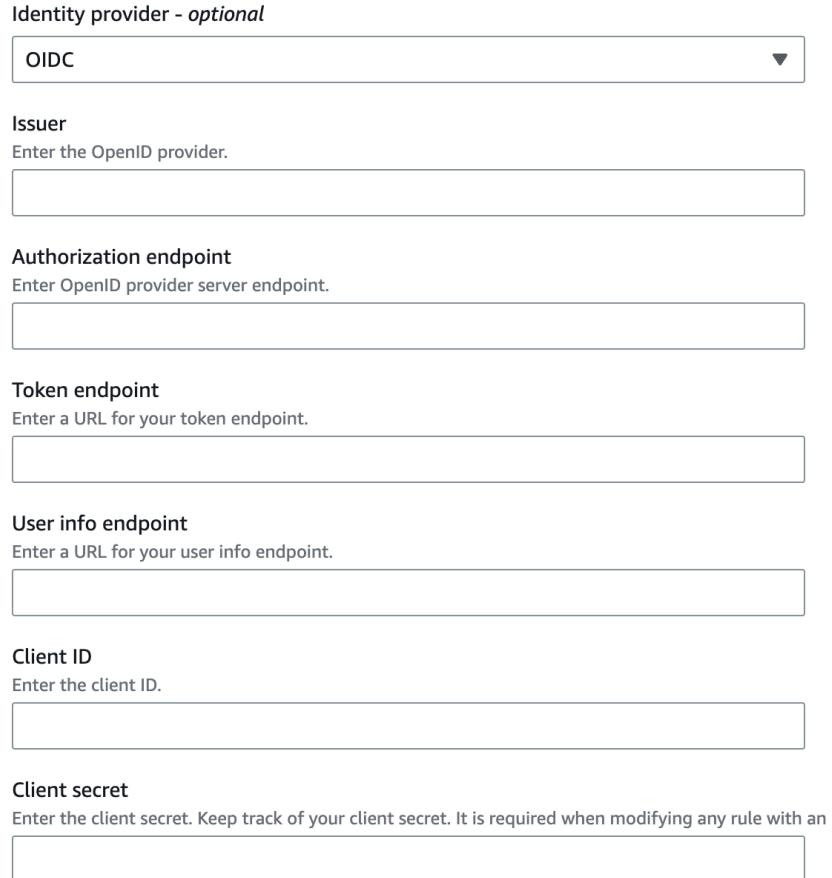
Authorization endpoint
Enter OpenID provider server endpoint.

Token endpoint
Enter a URL for your token endpoint.

User info endpoint
Enter a URL for your user info endpoint.

Client ID
Enter the client ID.

Client secret
Enter the client secret. Keep track of your client secret. It is required when modifying any rule with an

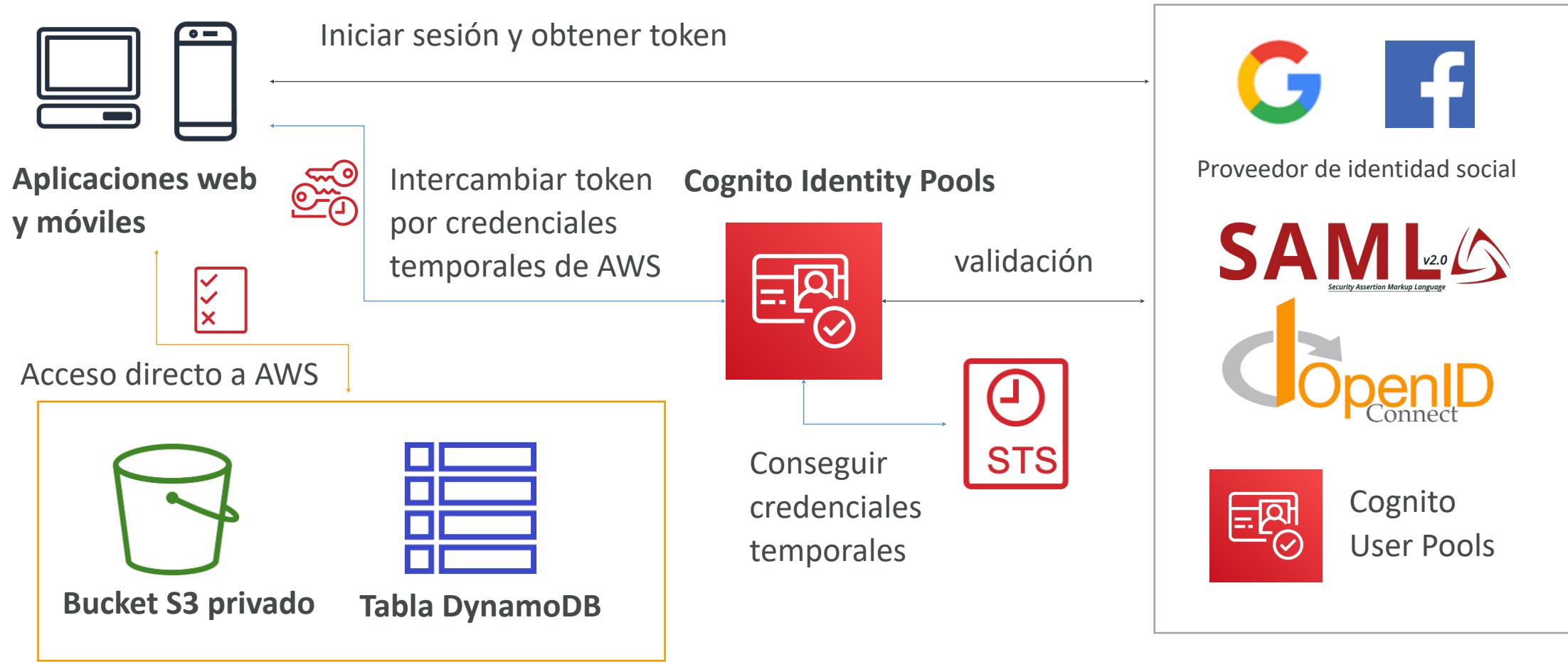


- Configura un ID de cliente y un secreto de cliente
- Permitir la redirección desde OIDC a tu Application Load Balancer mediante el nombre DNS (proporcionado por AWS) y CNAME (alias DNS de tu aplicación)
 - **<https://DNS/oauth2/idpresponse>**
 - **<https://CNAME/oauth2/idpresponse>**

Cognito Identity Pools (Identidades Federadas)

- **Obtener identidades para "usuarios" para que obtengan credenciales temporales de AWS**
- Tu grupo de identidades (por ejemplo, fuente de identidades) puede incluir
 - Proveedores públicos (inicio de sesión con Amazon, Facebook, Google, Apple)
 - Usuarios de un pool de usuarios de Amazon Cognito
 - Proveedores OpenID Connect y Proveedores de identidad SAML
 - Identidades autenticadas por desarrolladores (servidor de inicio de sesión personalizado)
 - Los Cognito Identity Pools permiten el acceso no autenticado (invitados)
- **Los usuarios pueden acceder a los servicios de AWS directamente o a través de API Gateway**
 - Las políticas IAM aplicadas a las credenciales se definen en Cognito
 - Se pueden personalizar en función del user_id para un control más preciso

Cognito Identity Pools – Diagrama



Cognito Identity Pools – Diagrama con CUP



Cognito Identity Pools – Roles IAM

- Roles IAM por defecto para usuarios autenticados e invitados
 - Define reglas para elegir el rol de cada usuario en función de su ID
 - Puedes particionar el acceso de tus usuarios utilizando **variables de política**
-
- Las credenciales IAM las obtiene Cognito Identity Pools a través de STS
 - Los roles deben tener una política de "confianza" de Cognito Identity Pools

Cognito Identity Pools

Ejemplo de usuario invitado

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Action": [  
                "s3:GetObject"  
            ],  
            "Effect": "Allow",  
            "Resource": [  
                "arn:aws:s3:::mybucket/assets/my_picture.jpg"  
            ]  
        }  
    ]  
}
```

Cognito Identity Pools – Variable de política de S3

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Action": ["s3>ListBucket"],  
            "Effect": "Allow",  
            "Resource": ["arn:aws:s3:::mybucket"],  
            "Condition": {"StringLike": {"s3:prefix": ["${cognito-identity.amazonaws.com:sub}/*"]}}  
        },  
        {  
            "Action": [  
                "s3:GetObject",  
                "s3:PutObject"  
            ],  
            "Effect": "Allow",  
            "Resource": ["arn:aws:s3:::mybucket,${cognito-identity.amazonaws.com:sub}/*"]  
        }  
    ]  
}
```

Cognito Identity Pools – DynamoDB

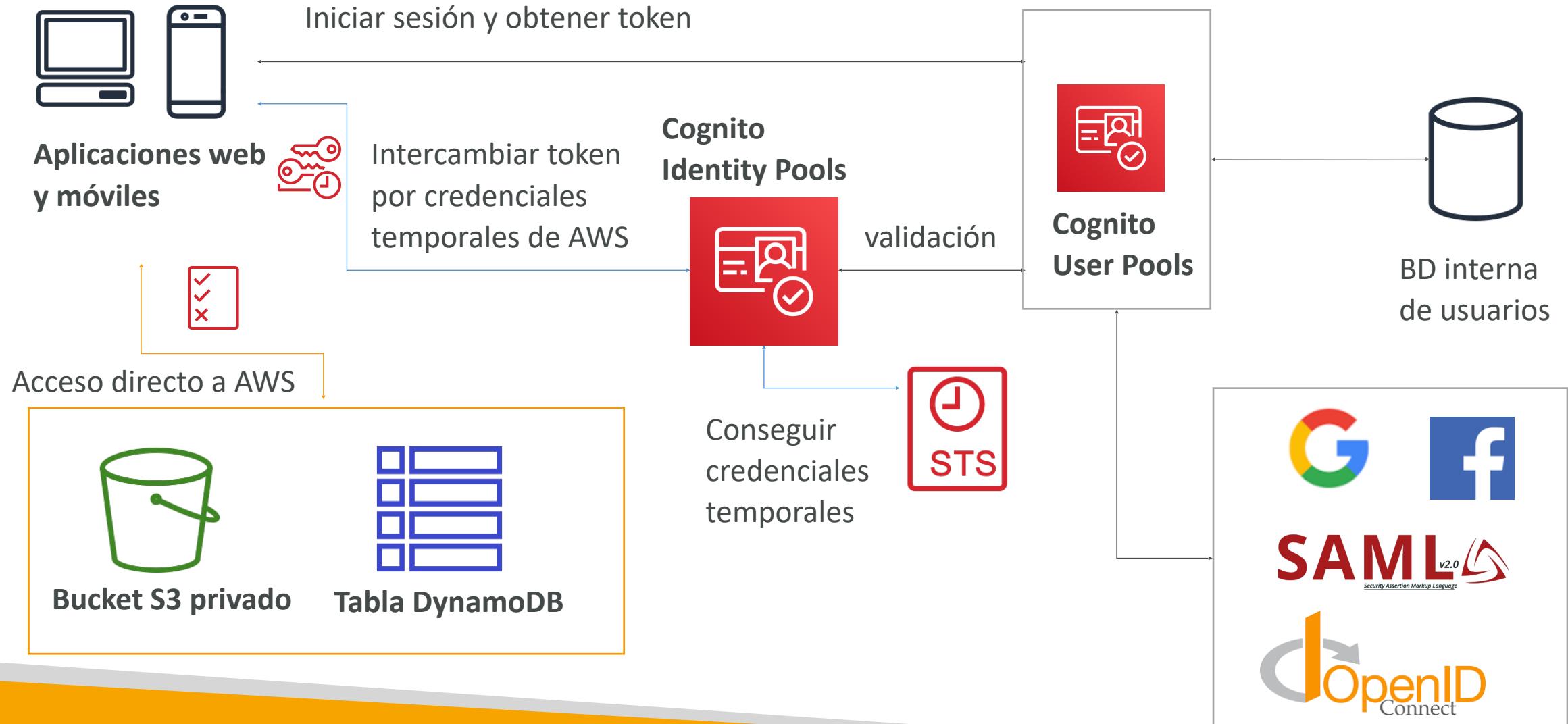
```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Effect": "Allow",  
            "Action": [  
                "dynamodb:GetItem", "dynamodb:BatchGetItem", "dynamodb:Query",  
                "dynamodb:PutItem", "dynamodb:UpdateItem", "dynamodb:DeleteItem",  
                "dynamodb:BatchWriteItem"  
            ],  
            "Resource": [  
                "arn:aws:dynamodb:us-west-2:123456789012:table/MyTable"  
            ],  
            "Condition": {  
                "ForAllValues:StringEquals": {  
                    "dynamodb:LeadingKeys": [  
                        "${cognito-identity.amazonaws.com:sub}"  
                    ]  
                }  
            }  
        }  
    ]  
}
```

Cognito User Pools vs Identity Pools



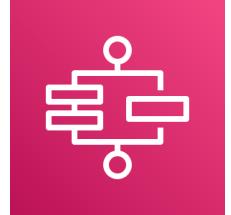
- **Cognito User Pools (para autenticación = verificación de identidad)**
 - Base de datos de usuarios para tu aplicación web y móvil
 - Permite federar inicios de sesión a través de Public Social, OIDC, SAML...
 - Puede personalizar la interfaz de usuario alojada para la autenticación (incluido el logotipo)
 - Tiene disparadores con AWS Lambda durante el flujo de autenticación
 - Adapta la experiencia de inicio de sesión a distintos niveles de riesgo (MFA, autenticación adaptativa, etc...)
- **Cognito Identity Pools (para autorización = control de acceso)**
 - Obtén credenciales de AWS para tus usuarios
 - Los usuarios pueden iniciar sesión a través de Public Social, OIDC, SAML y Cognito User Pools
 - Los usuarios pueden ser no autenticados (invitados)
 - Los usuarios están asignados a roles y políticas IAM, pueden aprovechar las variables de política
- **CUP + CIP = autenticación + autorización**

Cognito Identity Pools – Diagrama con CUP

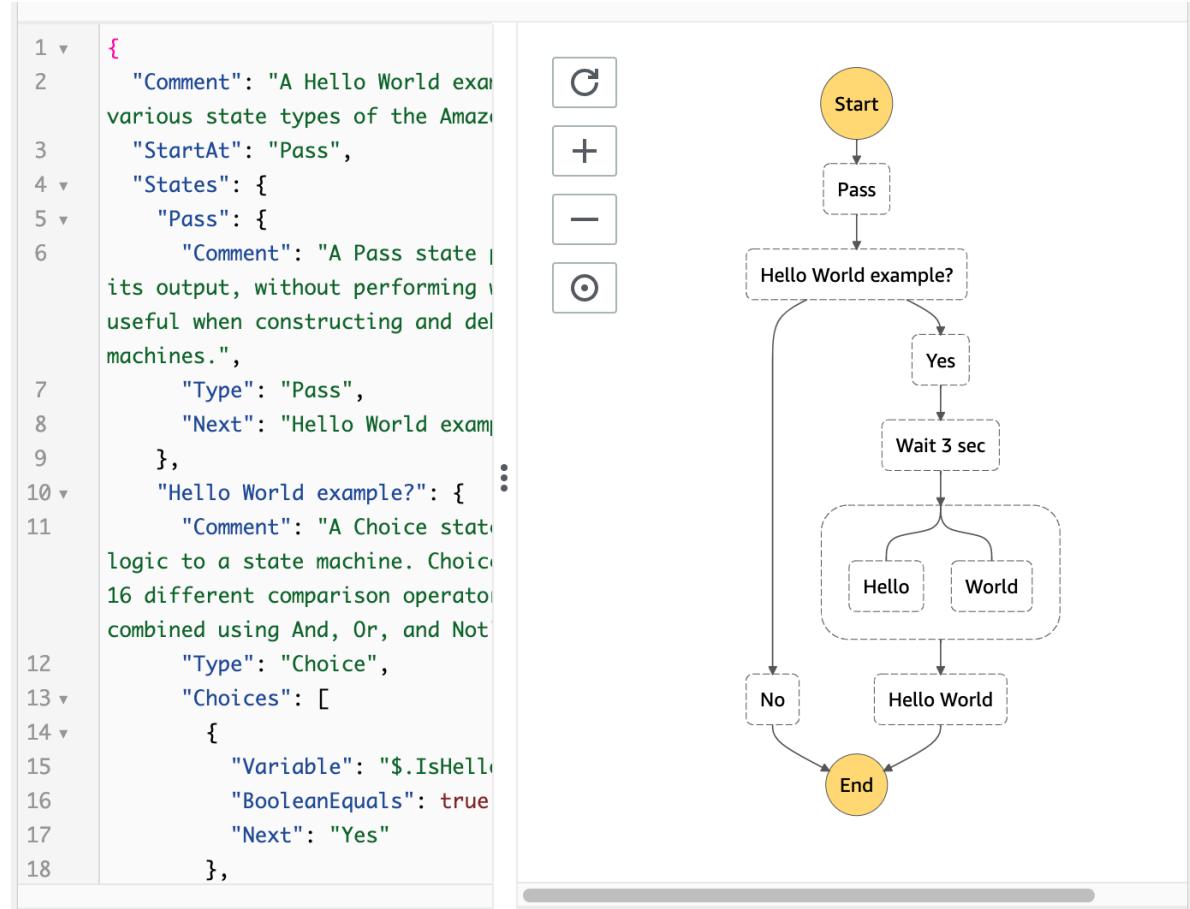


Otra sección sin servidor

AWS Step Functions



- Modela tus **flujos de trabajo (workflows) como máquinas de estado (una por flujo de trabajo)**
 - Cumplimiento de pedidos, procesamiento de datos
 - Aplicaciones web, cualquier flujo de trabajo
- Escrito en JSON
- Visualización del flujo de trabajo y de su ejecución, así como del historial
- Inicia el flujo de trabajo con una llamada al SDK, API Gateway, Event Bridge (CloudWatch Event)



Step Function – Estado de tareas

- Haz algún trabajo en tu máquina de estados
- Invoca un servicio de AWS
 - Puede invocar una **función Lambda**
 - Ejecutar un trabajo por lotes de AWS
 - Ejecutar una tarea ECS y esperar a que se complete
 - Insertar un elemento desde DynamoDB
 - Publicar un mensaje en SNS, SQS
 - Lanzar otro flujo de trabajo de Step Functions...
- Ejecutar una actividad
 - EC2, Amazon ECS, en las instalaciones
 - Las actividades preguntan a las Step Functions si están trabajando
 - Las actividades devuelven los resultados a las Step Functions



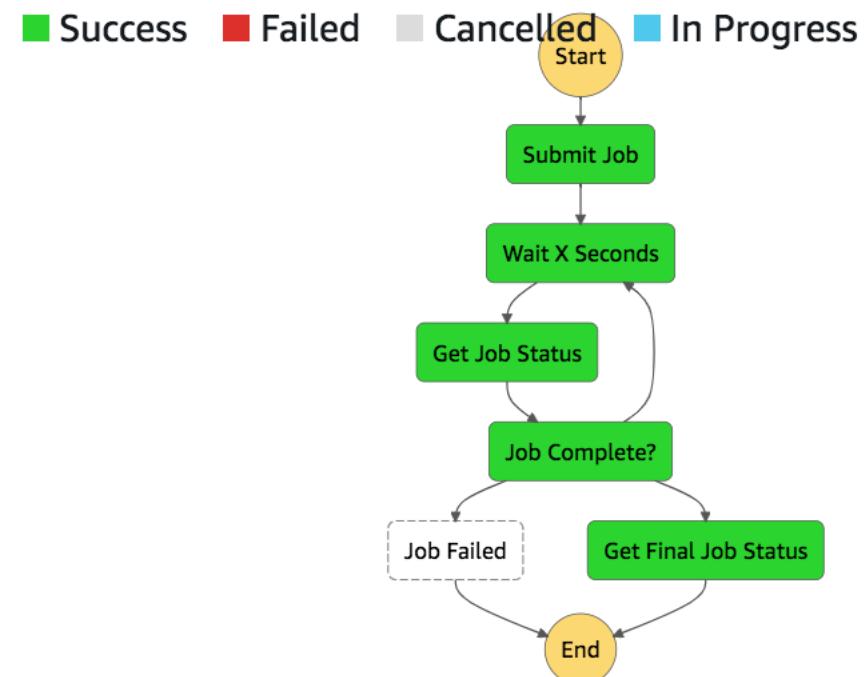
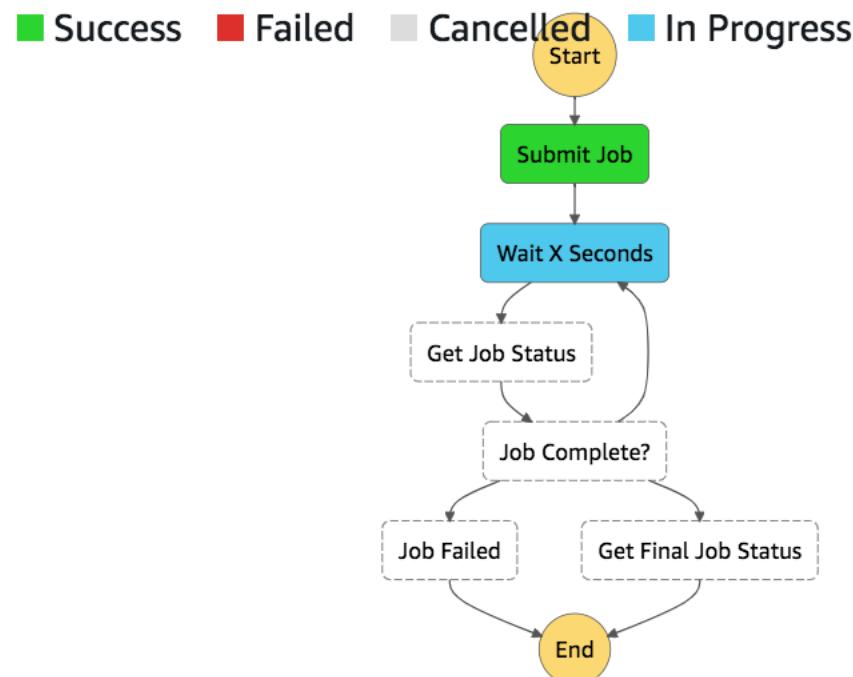
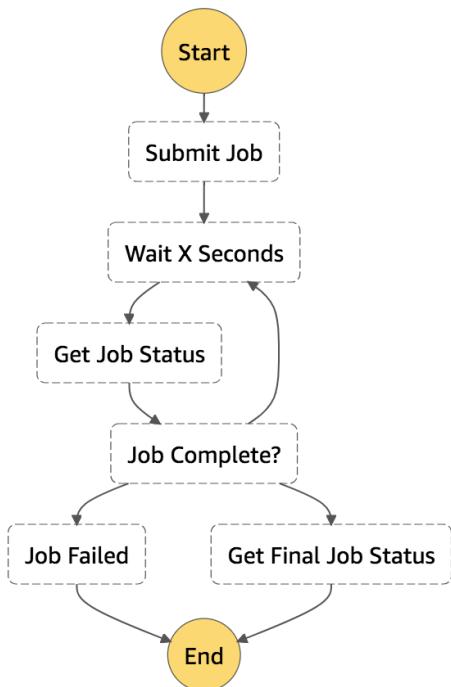
Ejemplo - Invocar función lambda

```
"Invoke Lambda function": {  
    "Type": "Task",  
    "Resource": "arn:aws:states:::lambda:invoke",  
    "Parameters": {  
        "FunctionName":  
            "arn:aws:lambda:REGION:ACCOUNT_ID:function:FUNCTION_NAME",  
        "Payload": {  
            "Input.$": "$"  
        }  
    },  
    "Next": "NEXT_STATE",  
    "TimeoutSeconds": 300  
}
```

Step Function - Estados

- **Estado Choice** - Testea una condición para enviarla a una rama (o rama por defecto)
- **Estado Fail o Succeed** - Detener la ejecución con fallo o éxito
- **Estado Pass** - Simplemente pasa su entrada a su salida o inyecta algún dato fijo, sin realizar trabajo
- **Estado Wait** - Proporcionar un retraso durante un cierto tiempo o hasta una hora/fecha especificada
- **Estado Map** - Iterar pasos dinámicamente'
- **Estado Parallel** - *Iniciar ramas paralelas de ejecución*

Flujo de trabajo visual en Step Functions



Tratamiento de errores en Step Functions

- Cualquier estado puede encontrar errores en tiempo de ejecución por varias razones:
 - Problemas de definición de la máquina de estados (por ejemplo, ninguna regla coincidente en un estado *choice*)
 - Fallos de tareas (por ejemplo, una excepción en una función Lambda)
 - Problemas transitorios (por ejemplo, eventos de partición de red)
- Utiliza **Retry** (para reintentar el estado *fallido*) y **Catch** (transición a la ruta de fallo) en la máquina de estado para manejar los errores en lugar de dentro del código de aplicación
- Códigos de error predefinidos:
 - States.ALL : coincide con cualquier nombre de error
 - States.Timeout: la tarea se ejecutó más de TimeoutSeconds o no se recibió ningún heartbeat (señal de estar vivo)
 - States.TaskFailed: fallo de ejecución
 - States.Permissions: privilegios insuficientes para ejecutar código
- El estado puede informar de sus propios errores

Step Functions – Retry (Tarea o Estado Parallel)

```
"HelloWorld": {
    "Type": "Task",
    "Resource": "arn:aws:lambda:REGION:ACCOUNT_ID:function:FUNCTION_NAME",
    "Retry": [
        {
            "ErrorEquals": ["CustomError"],
            "IntervalSeconds": 1,
            "MaxAttempts": 2,
            "BackoffRate": 2.0
        },
        {
            "ErrorEquals": ["States.TaskFailed"],
            "IntervalSeconds": 30,
            "MaxAttempts": 2,
            "BackoffRate": 2.0
        },
        {
            "ErrorEquals": ["States.ALL"],
            "IntervalSeconds": 5,
            "MaxAttempts": 5,
            "BackoffRate": 2.0
        }
    ],
    "End": true
}
```

- Se evalúa de arriba abajo
- **ErrorEquals**: coincide con un tipo específico de error
- **IntervalSeconds**: retardo inicial antes de reintentar
- **BackoffRate**: multiplica el retardo después de cada reinicio
- **MaxAttempts**: por defecto 3, o establecer en 0 para no reintentar nunca
- Cuando se alcanza el máximo de intentos, se activa el **Catch**

Step Functions – Catch (Task o Estado Parallel)

```
"HelloWorld": {
    "Type": "Task",
    "Resource": "arn:aws:lambda:....",
    "Catch": [
        {
            "ErrorEquals": ["CustomError"],
            "Next": "CustomErrorFallback"
        },
        {
            "ErrorEquals": ["States.TaskFailed"],
            "Next": "ReservedTypeFallback"
        },
        {
            "ErrorEquals": ["States.ALL"],
            "Next": "NextTask",
            "ResultPath": "$.error"
        }
    ],
    "End": true
},
"CustomErrorFallback": {
    "Type": "Pass",
    "Result": "This is a fallback from a custom lambda function exception"
    "End": true
},
```

- Se evalúa de arriba abajo
- **ErrorEquals**: coincide con un tipo específico de error
- **Next**: estado al que enviar
- **ResultPath**: ruta que determina qué entrada se envía al estado especificado en el campo Next.

Step Function – ResultPath

- Incluir el error en la entrada

```
"HelloWorld": {  
    "Type": "Task",  
    "Resource": "arn:aws:lambda:....",  
    "Catch": [{  
        "ErrorEquals": ["States.ALL"],  
        "Next": "NextTask",  
        "ResultPath": "$.error"  
    }],  
    "End": true  
},  
"NextTask": {  
    "Type": "Pass",  
    "Result": "This is a fallback from a reserved error code",  
    "End": true  
}
```

{"foo": "bar"}

ENTRADA

{
 "foo": "bar",
 "error": {
 "Error": "Error here"
 }
}

SALIDA CON ERROR
MEDIANTE
RESULTPATH

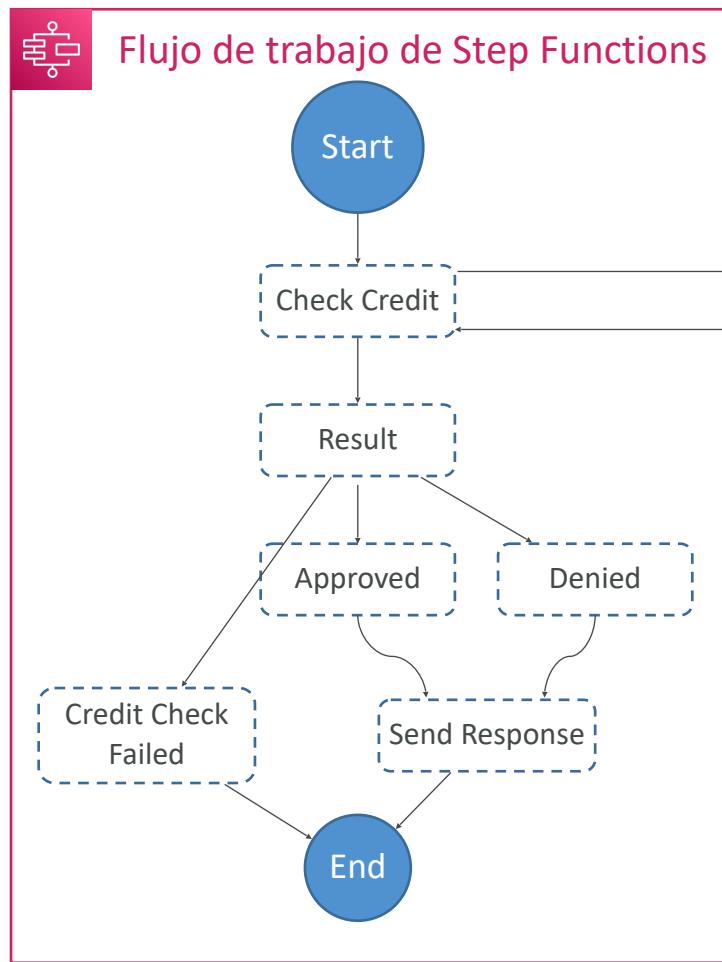
Step Functions – Espera del token de tarea

- Te permite pausar Step Functions durante una tarea hasta que se devuelva un token de tarea (Task Token)
- La tarea puede esperar a otros servicios de AWS, aprobación humana, integración de terceros, llamada a sistemas heredados...
- Añade **.waitForTaskToken** al campo **Resource** para indicar a Step Functions que espere a que se devuelva el token de tarea

`"Resource": "arn:aws:states:::sns:sendMessage.waitForTaskToken"`

- La tarea se detendrá hasta que reciba ese token de tarea de vuelta con una llamada a la API **SendTaskSuccess** o **SendTaskFailure**

Step Functions – Espera del token de tarea



```
{
  "Resource": "arn:aws:states:::sns:publish",
  "Parameters": {
    "TopicArn": "arn:aws:sns:eu-west-1:123456789012:MyTopic"
  }
}
```

Ilustrar la ejecución de la función Lambda que se encarga de enviar el resultado de la verificación de crédito a través de SNS.

Llamar a SQS con el token de tarea (task token)

Cola SQS
(MyQueue)



mensajes de sondeo

```
{
  "output": "CreditAccepted",
  "taskToken": "83p1E0dMccoxlzFhglsdkzpK9mBVKZsp7d9yrT1W"
}
```

Tarea completada

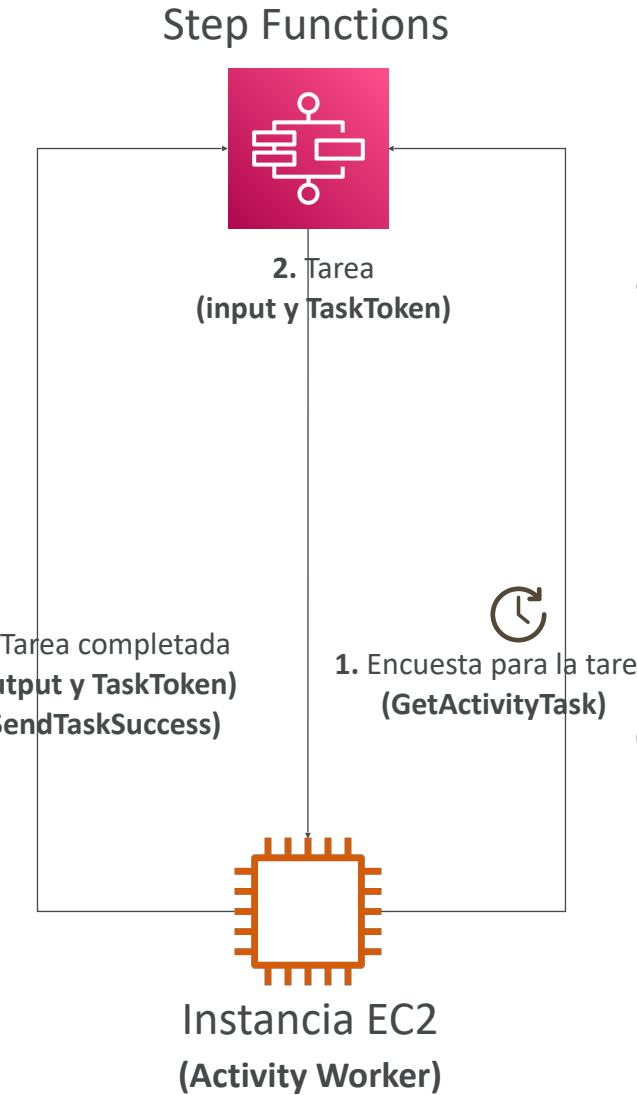
Llamada API **SendTaskSuccess**



Mensajes de proceso

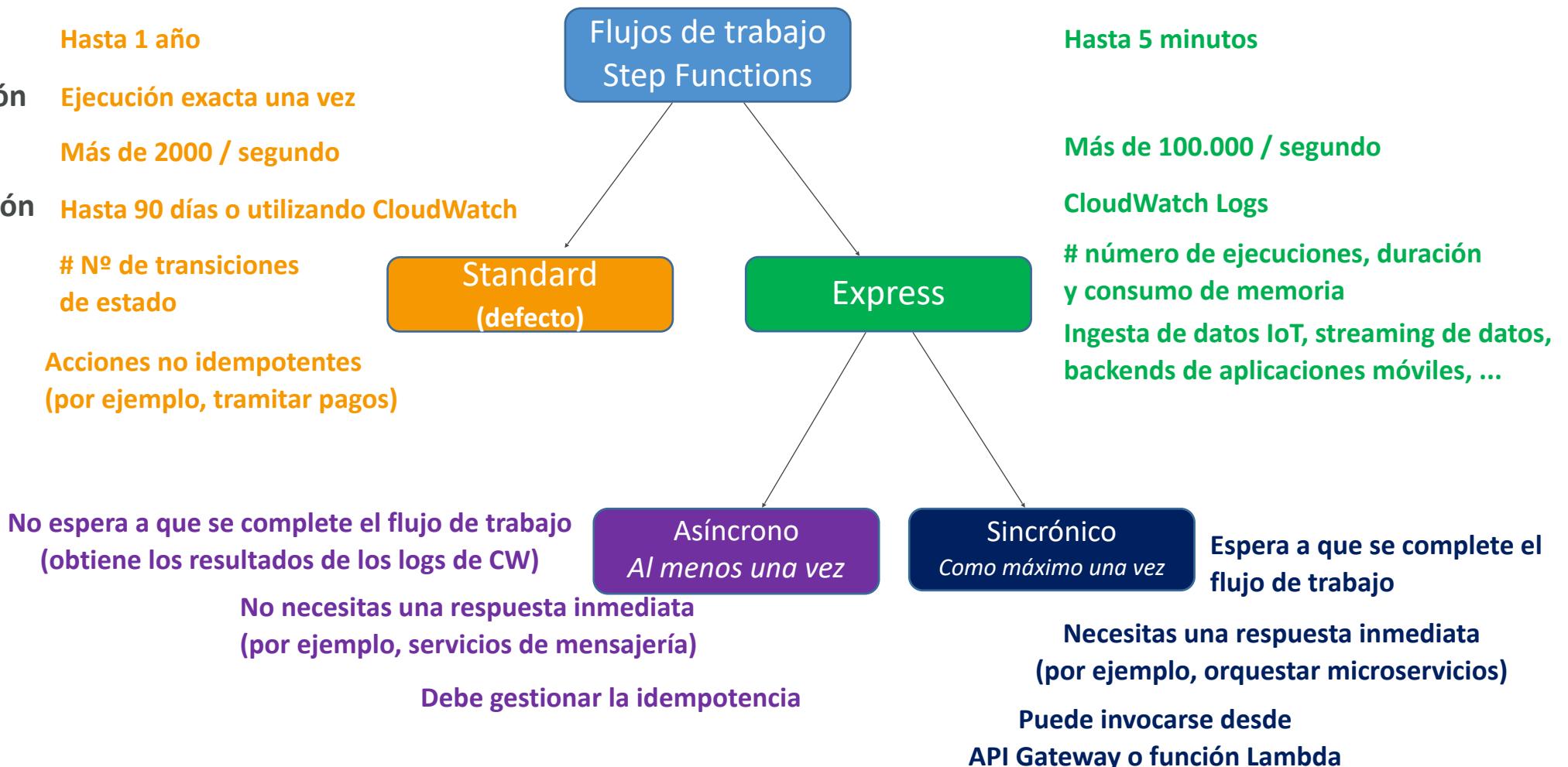
Step Functions – Tareas de actividad

- Te permite que el trabajo de la tarea lo realice un **trabajador de actividad (Activity Worker)**
- Las aplicaciones **Activity Worker** pueden ejecutarse en EC2, Lambda, dispositivo móvil...
- El Activity Worker sondea una tarea utilizando la API **GetActivityTask**
- Después de que el Activity Worker complete su trabajo, envía una respuesta de su éxito/fracaso utilizando **SendTaskSuccess** o **SendTaskFailure**
- Mantener activa la tarea
 - Configura cuánto tiempo puede esperar una tarea estableciendo **TimeoutSeconds**
 - Envía periódicamente un latido desde tu Activity Worker utilizando **SendTaskHeartBeat** dentro del tiempo que establezcas en HeartBeatSeconds
- Configurando un **TimeoutSeconds** largo y enviando activamente un heartbeat, la tarea puede esperar hasta 1 año



Step Functions – Standard vs. Express

Máx. Duración	Hasta 1 año
Modelo de ejecución	Ejecución exacta una vez
Tasa de ejecución	Más de 2000 / segundo
Historial de ejecución	Hasta 90 días o utilizando CloudWatch
Precios	# Nº de transiciones de estado
Casos de uso	Acciones no idempotentes (por ejemplo, tramitar pagos)



AWS AppSync - Visión general



- **AppSync** es un servicio gestionado que utiliza **GraphQL**
- **GraphQL** facilita que las aplicaciones obtengan exactamente los datos que necesitan
- Esto incluye combinar datos de **una o varias fuentes**
 - Almacenes de datos NoSQL, bases de datos relacionales, APIs HTTP...
 - Se integra con DynamoDB, Aurora, OpenSearch y otros
 - Fuentes personalizadas con AWS Lambda
- Recupera datos en **tiempo real con WebSocket o MQTT en WebSocket**
- Para aplicaciones móviles: acceso a datos locales y sincronización de datos
- Todo comienza con la carga de un **GraphQL schema**

Ejemplo de GraphQL

```

type Query {
  human(id: ID!): Human
}

type Human {
  name: String
  appearsIn: [Episode]
  starships: [Starship]
}

enum Episode {
  NEWHOPE
  EMPIRE
  JEDI
}

type Starship {
  name: String
}
  
```

Esquema GraphQL en AppSync

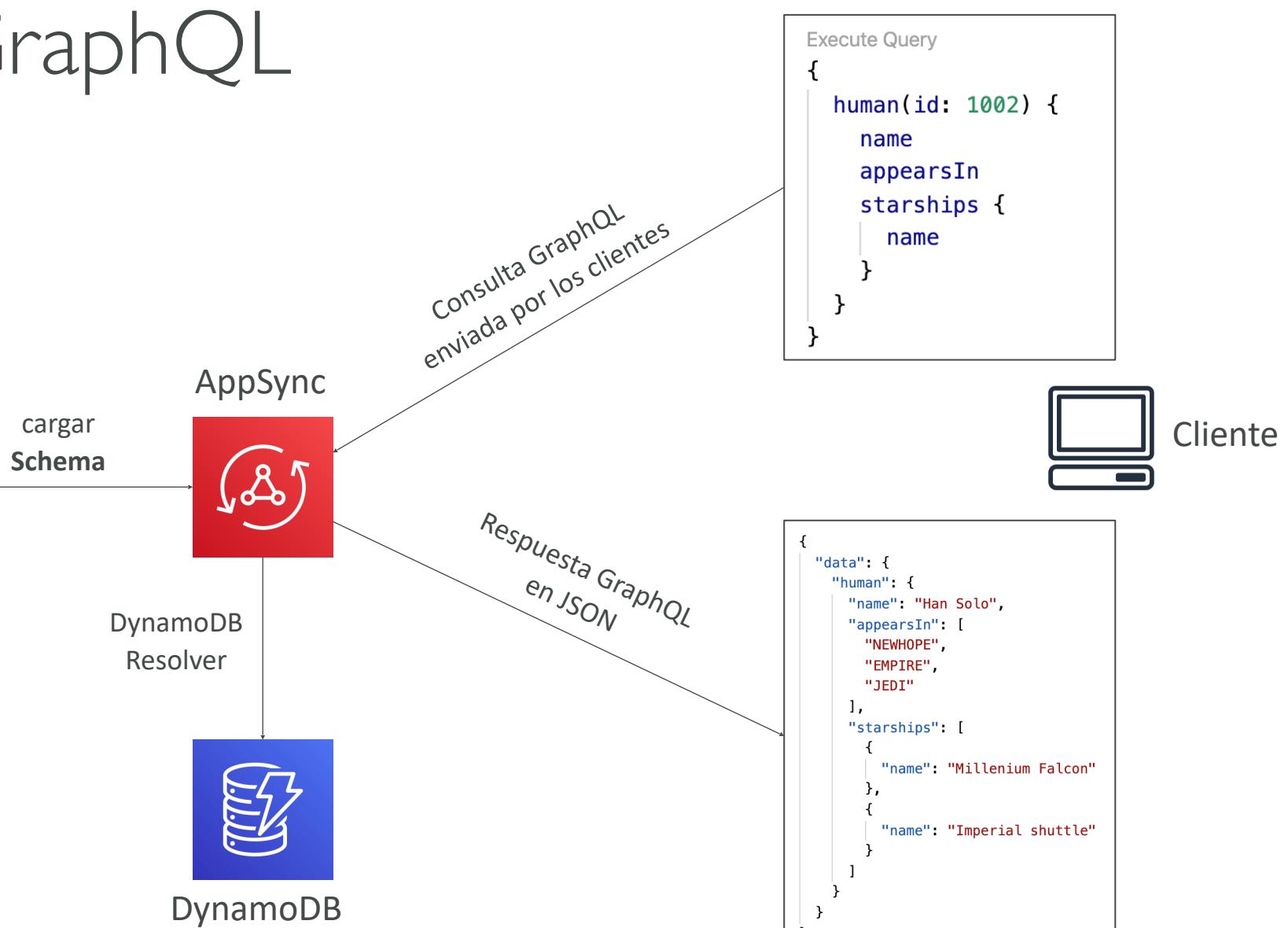
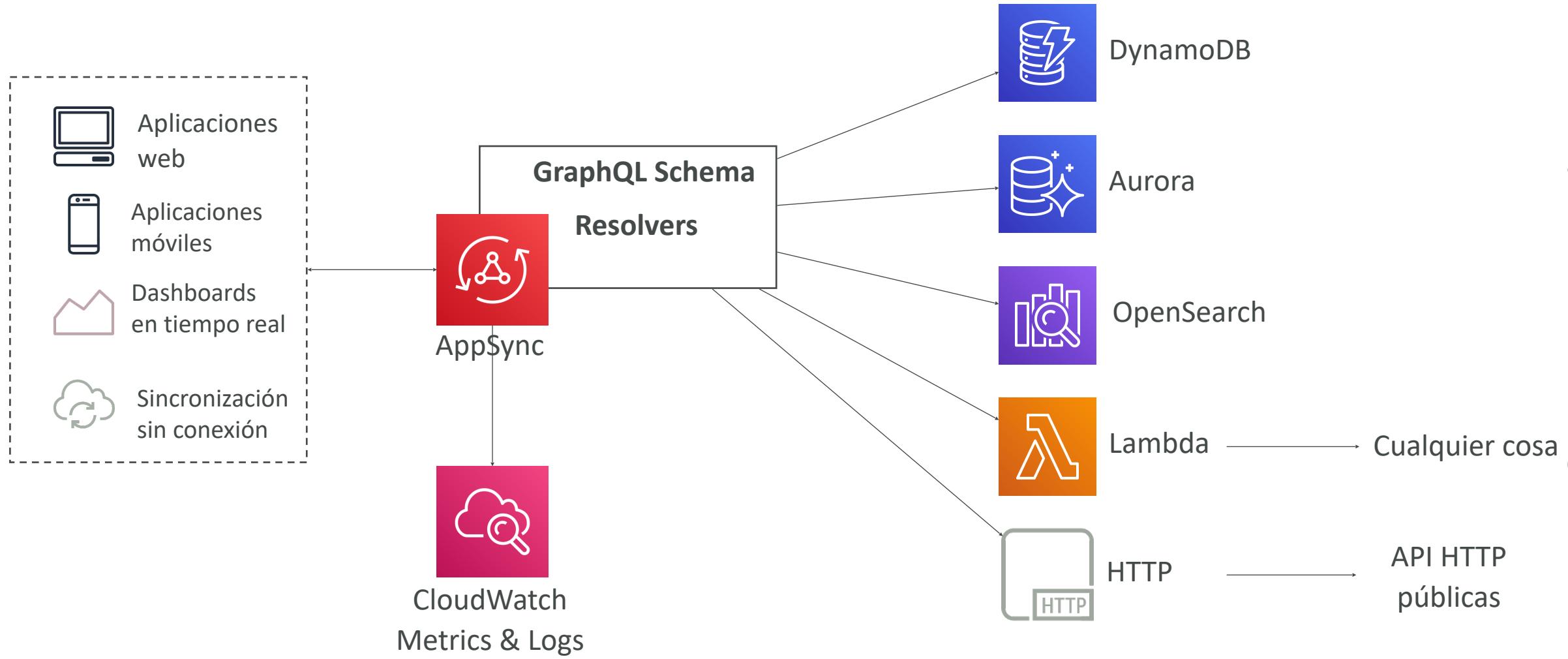


Diagrama AppSync



AppSync - Seguridad

- Hay cuatro formas de autorizar aplicaciones para que interactúen con tu API GraphQL de AWS AppSync:
- **API_KEY**
- **AWS_IAM**: usuarios IAM / roles / acceso a cuentas cruzadas
- **OPENID_CONNECT**: Proveedor de OpenID Connect / Token Web JSON
- **AMAZON_COGNITO_USER_POOLS**
- Para dominio personalizado y HTTPS, utiliza CloudFront delante de AppSync

AWS Amplify

Crear aplicaciones móviles y web



Amplify Studio

Construye visualmente una aplicación full stack, tanto la interfaz de usuario front-end como el back-end.



Amplify CLI

Configurar un backend de Amplify Con un flujo de trabajo CLI guiado



Amplify Libraries

Conecta tu aplicación a los servicios AWS existentes (Cognito, S3 y más)

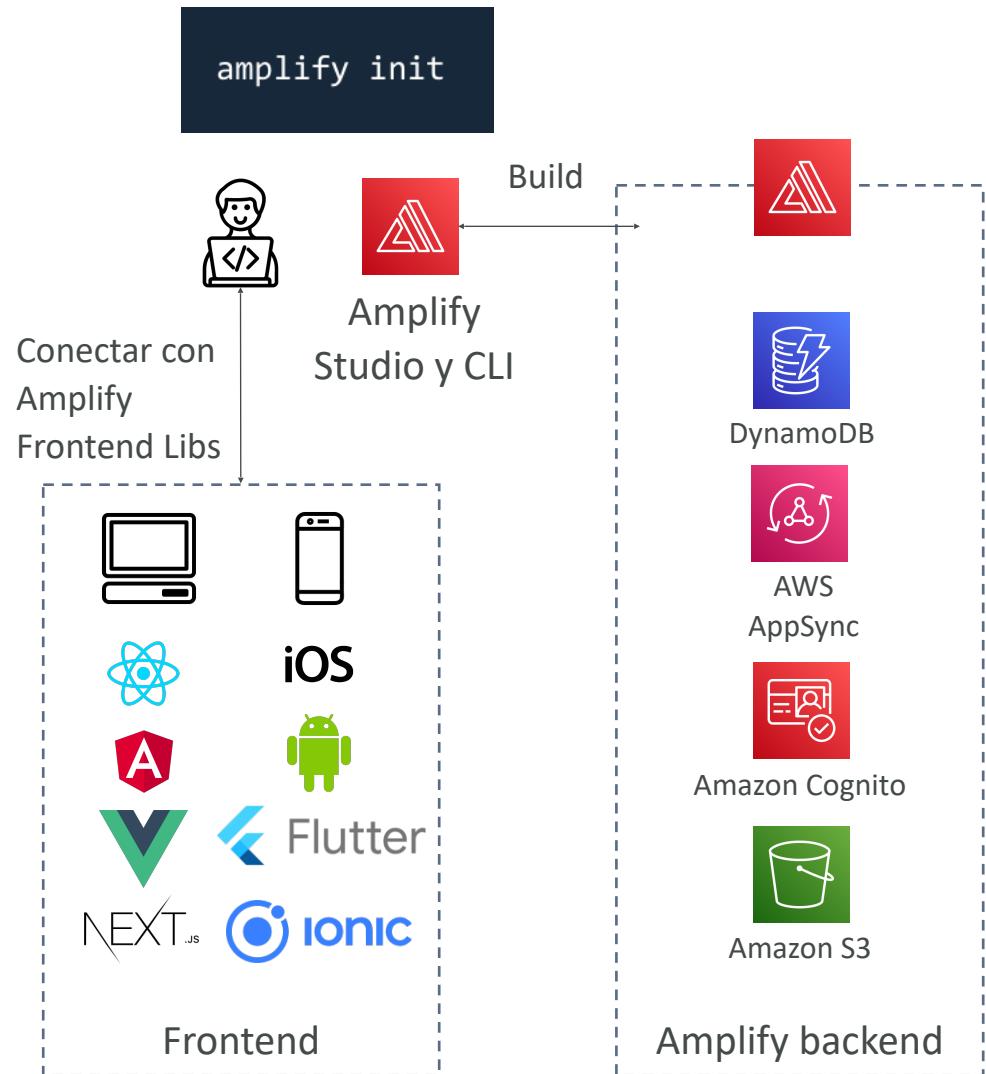


Amplify Hosting

Aloja aplicaciones web o sitios web seguros, fiables y rápidos a través de la red de entrega de contenido de AWS.

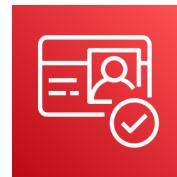
AWS Amplify

- Conjunto de herramientas para iniciarse en la creación de **aplicaciones móviles y web**
- "Elastic Beanstalk para aplicaciones móviles y web"
- Funciones imprescindibles como el **almacenamiento de datos, la autenticación, el almacenamiento y el machine learning**, todas ellas impulsadas por los servicios de AWS
- **Bibliotecas front-end** con componentes listos para usar para React.js, Vue, Javascript, iOS, Android, Flutter, etc.
- Incorpora las mejores prácticas de AWS para la fiabilidad, seguridad y escalabilidad
- Construye y despliega con **Amplify CLI o Amplify Studio**



AWS Amplify – Características importantes

```
amplify add auth
```



```
amplify add api
```



AUTENTICACIÓN

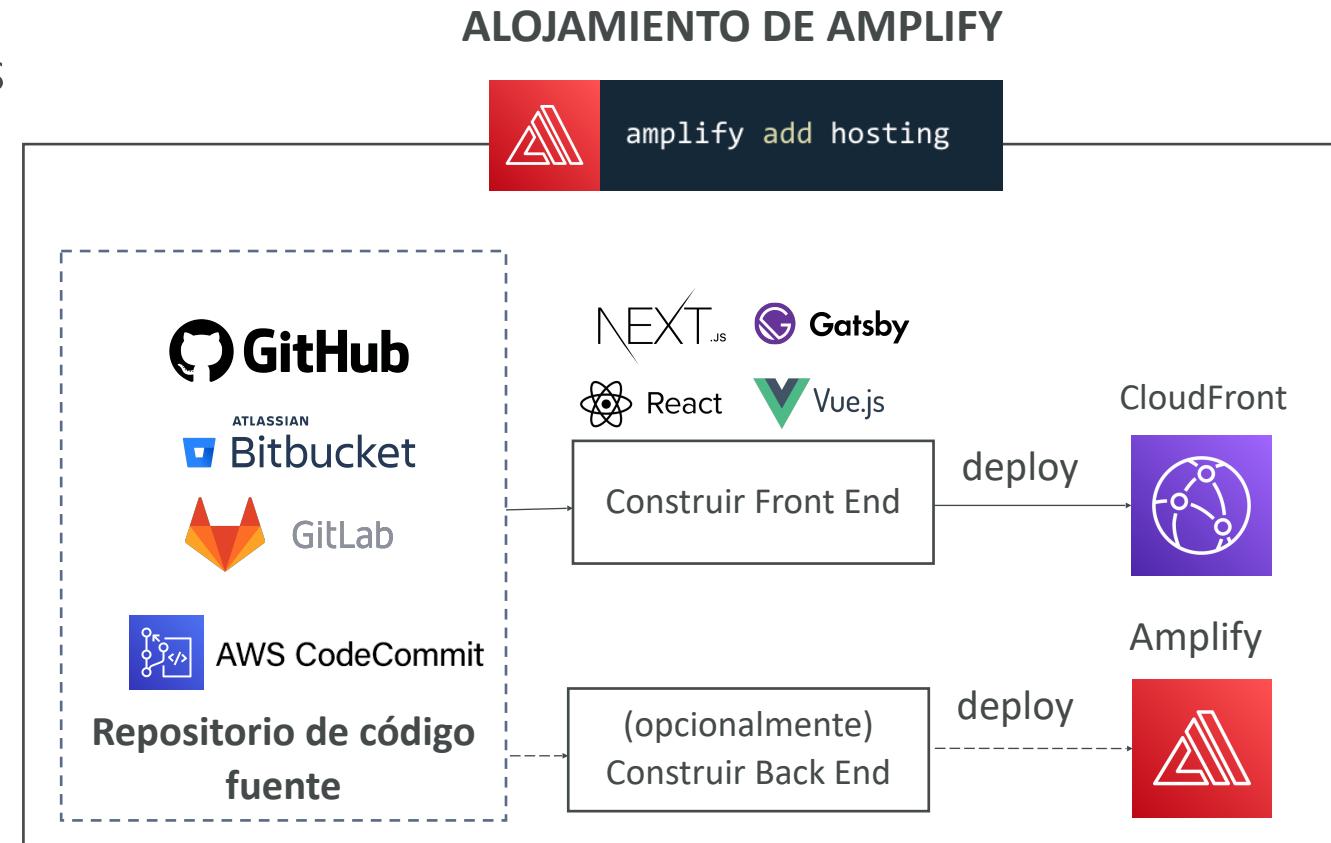
- Aprovecha Amazon Cognito
- Registro de usuarios, autenticación, recuperación de cuentas y otras operaciones
- Soporta MFA, Social Sign-in, etc...
- Componentes de interfaz de usuario prediseñados
- Autorización detallada

ALMACÉN DE DATOS

- Aprovecha Amazon AppSync y Amazon DynamoDB
- Trabaja con datos locales y **sincronízalos automáticamente en el Cloud** sin código complejo
- Potenciado por GraphQL
- Capacidades offline y en tiempo real
- Modelado visual de datos con Amplify Studio

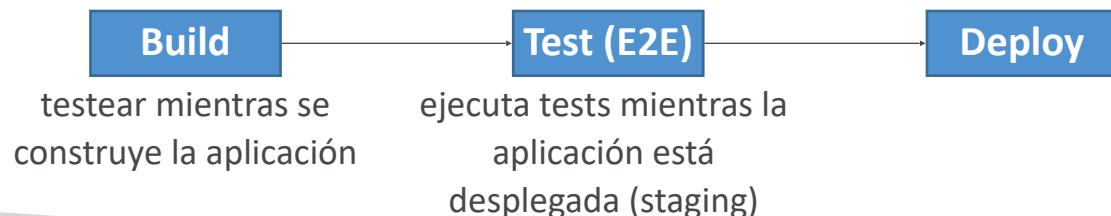
Alojamiento de AWS Amplify

- Crea y aloja aplicaciones web modernas
- CICD (build, test, deploy)
- Previsualización de peticiones
- Dominios personalizados
- Supervisión
- Redirección y cabeceras personalizadas
- Protección con contraseña



AWS Amplify – Testing End-to-End (E2E)

- Ejecuta tests de extremo a extremo (E2E) en la **fase de test** en Amplify
- Detecta regresiones antes de pasar el código a producción
- Utiliza el paso test para ejecutar cualquier comando de prueba en tiempo de compilación (**amplify.yml**)
- **Integrado con el framework de test Cypress**
 - Te permite generar informes de interfaz de usuario para tus tests



```

test:
phases:
  preTest:
    commands:
      - npm ci
      - npm install -g pm2
      - npm install -g wait-on
      - npm install mocha mochawesome ...
      - pm2 start npm -- start
      - wait-on http://localhost:3000
  test:
    commands:
      - 'npx cypress run --reporter ...'
  postTest:
    commands:
      - npx mochawesome-merge cypress/...
      - pm2 kill
artifacts:
  baseDirectory: cypress
  configFilePath: "**/mochawesome.json"
  files:
    - "**/*.png"
    - "**/*.mp4"
  
```

amplify.yml

Identidad Avanzada

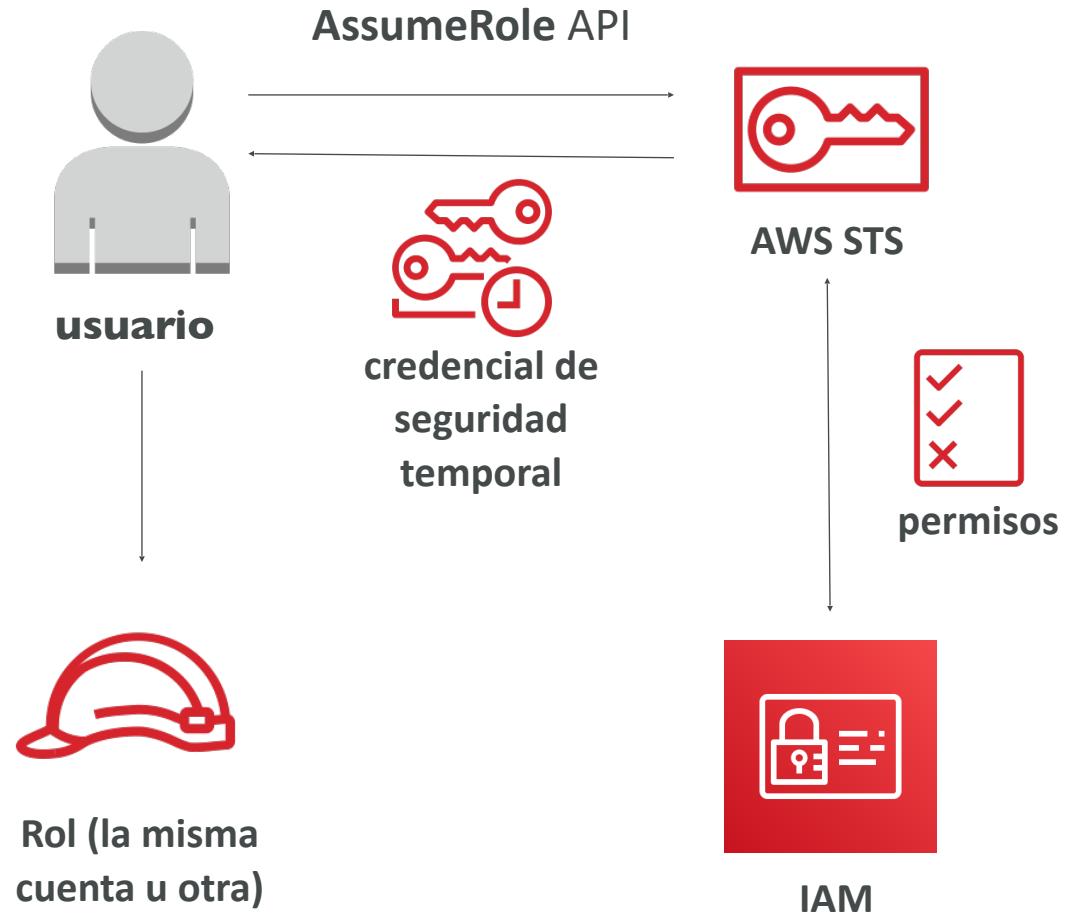


AWS STS – Security Token Service

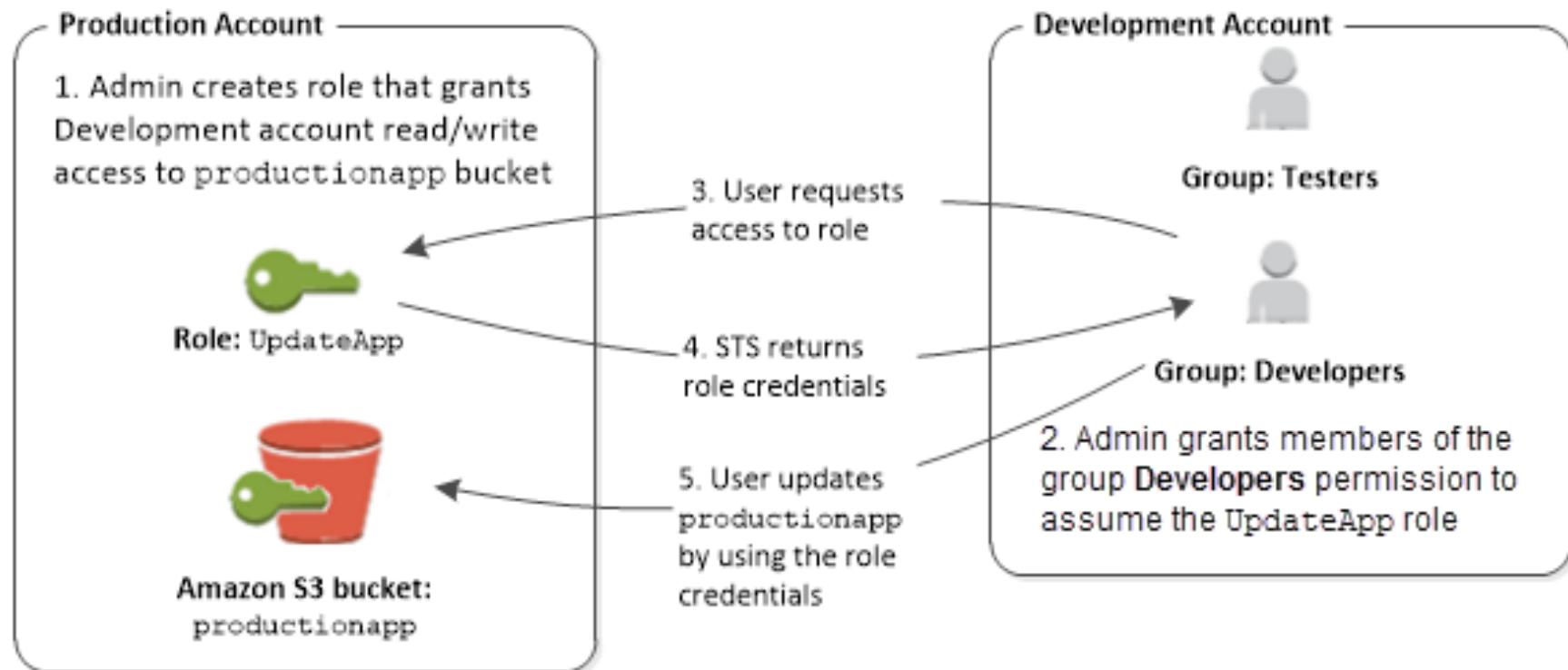
- Permite conceder acceso limitado y temporal a los recursos de AWS (hasta 1 hora).
- **AssumeRole**: Asume roles dentro de tu cuenta o entre cuentas
- **AssumeRoleWithSAML**: devuelve las credenciales de los usuarios logs con SAML
- **AssumeRoleWithWebIdentity**
 - devuelve credenciales para usuarios registrados con un IdP (Facebook Login, Google Login, compatible con OIDC...)
 - AWS recomienda no utilizar esta opción, y utilizar en su lugar Cognito Identity Pools
- **GetSessionToken**: para MFA, de un usuario o usuario root de la cuenta AWS
- **GetFederationToken**: para obtener creds temporales de un usuario federado
- **GetCallerIdentity**: devuelve detalles sobre el usuario o rol IAM utilizado en la llamada a la API
- **DecodeAuthorizationMessage**: descodifica el mensaje de error cuando se deniega una API de AWS

Utilizar el STS para asumir un rol

- Define un rol IAM dentro de tu cuenta o cuenta cruzada
- Define qué principales pueden acceder a este rol IAM
- Utiliza AWS STS (Security Token Service) para recuperar credenciales y suplantar el rol IAM al que tienes acceso (**API AssumeRole**)
- Las credenciales temporales pueden ser válidas entre 15 minutos y 1 hora



Acceso cruzado de cuentas con STS



https://docs.aws.amazon.com/IAM/latest/UserGuide/id_roles_common-scenarios_aws-accounts.html

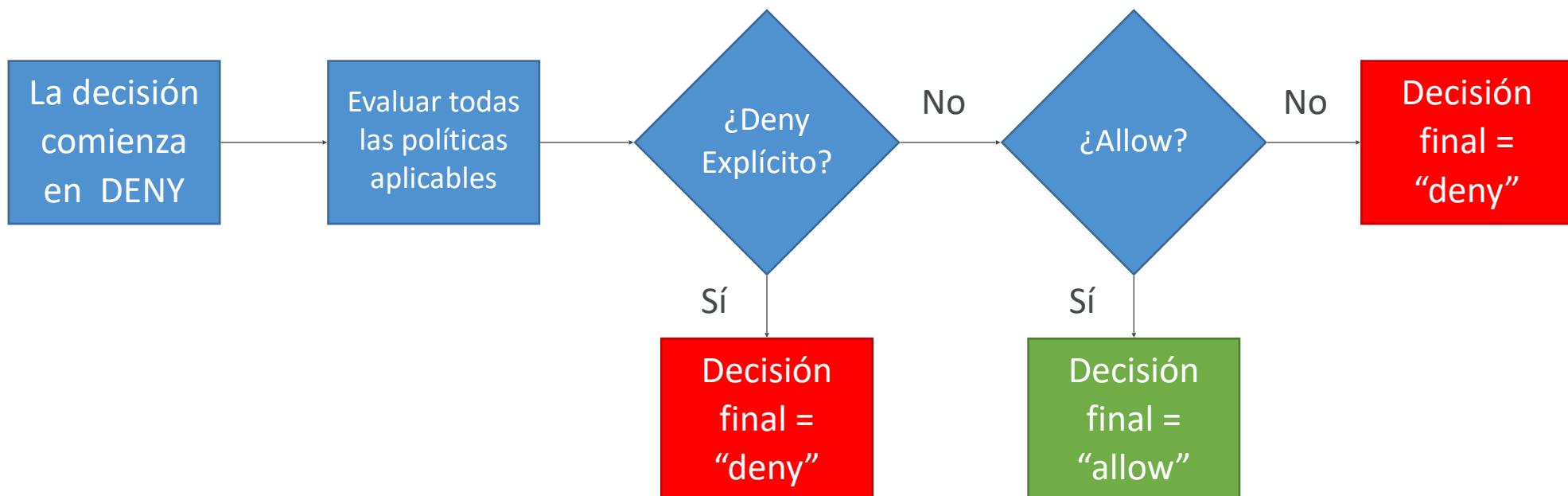
STS con MFA

- Utiliza **GetSessionToken** desde STS
- Política IAM adecuada utilizando las condiciones IAM
- **aws:MultiFactorAuthPresent:true**
- Recordatorio, GetSessionToken devuelve:
 - Access ID
 - Secret Key
 - Session Token
 - Expiration date

```
{  
  "Version": "2012-10-17",  
  "Statement": [  
    {  
      "Effect": "Allow",  
      "Action": [  
        "ec2:StopInstances",  
        "ec2:TerminateInstances"  
      ],  
      "Resource": [  
        "*"  
      ],  
      "Condition": {  
        "Bool": {  
          "aws:MultiFactorAuthPresent": "true"  
        }  
      }  
    }  
  ]  
}
```

IAM avanzado - Modelo de autorización evaluación de políticas, simplificado

1. Si hay un DENY explícito, termina la decisión con DENY
2. Si hay un ALLOW, termina la decisión con ALLOW
3. Si no, DENY



Políticas IAM y Políticas de bucket S3

- Las políticas IAM se adjuntan a usuarios, roles y grupos
- Las políticas de bucket S3 se adjuntan a los buckets
- Al evaluar si un Principal IAM puede realizar una operación X en un bucket, se evaluará la **unión** de sus Políticas IAM y Políticas de bucket S3 asignadas



Ejemplo I

- Rol IAM adjunto a instancia EC2, autoriza RW a "mi_bucket"
 - No se adjunta Política de bucket S3
-
- => La instancia EC2 puede leer y escribir en "mi_bucket"

Ejemplo 2

- Rol IAM adjunto a la instancia EC2, autoriza RW a "mi_bucket"
 - Política de bucket S3 adjunta, denegación explícita al rol IAM
-
- => La instancia EC2 **no puede** leer ni escribir en "mi_bucket"

Ejemplo 3

- Rol IAM adjunto a la instancia EC2, sin permisos de bucket S3
 - Política de bucket S3 adjunta, permiso RW explícito al rol IAM
-
- => la instancia EC2 **puede** leer y escribir en “mi_bucket”

Ejemplo 4

- Rol IAM adjunto a la instancia EC2, denegación explícita de permisos de bucket S3
- Política de bucket S3 adjunta, permiso RW explícito al rol IAM
- => la instancia EC2 **no puede** leer ni escribir en “mi_bucket”

Políticas dinámicas con IAM

- ¿Cómo se asigna a cada usuario una carpeta /home/<usuario> en un bucket S3?
- Opción 1:
 - Crea una política IAM que permita a George tener acceso a /home/george
 - Crea una política IAM que permita a Sara acceder a /home/sara
 - Crea una política IAM que permita a Mateo acceder a /home/mateo
 - ... ¡Una política por usuario!
 - Esto no es escalable
- Opción 2:
 - Crear una política dinámica con IAM
 - Aprovecha la variable de política especial \${aws:nombredeusuario}

Ejemplo de política dinámica

```
{  
    "Sid": "AllowAllS3ActionsInUserFolder",  
    "Action": ["s3:*"],  
    "Effect": "Allow",  
    "Resource": ["arn:aws:s3:::my-company/home/${aws:username}/*"]  
}
```

Políticas en línea vs políticas gestionadas

- Política administrada por AWS
 - Mantenida por AWS
 - Buena para usuarios avanzados y administradores
 - Actualizada en caso de nuevos servicios / nuevas API
- Política gestionada por el cliente
 - Mejor práctica, reutilizable, puede aplicarse a muchos principales
 - Versión controlada + rollback, gestión central de cambios
- En línea
 - Estricta relación uno a uno entre política y entidad de seguridad
 - La política se elimina si eliminas el principal IAM

Conceder permisos a un usuario para pasar un rol a un servicio de AWS

- Para configurar muchos servicios de AWS, debes **pasar** un rol IAM al servicio (esto ocurre sólo una vez durante la configuración)
- Posteriormente, el servicio asumirá el rol y realizará acciones
- Ejemplo de paso de un rol:
 - A una instancia EC2
 - A una función Lambda
 - A una tarea ECS
 - A CodePipeline para permitirle invocar otros servicios
- Para ello, necesitas el permiso IAM **iam:PassRole**
- Suele ir acompañado de iam:GetRole para ver el rol que se pasa

Ejemplo de IAM PassRole

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Effect": "Allow",  
            "Action": [  
                "ec2:*"  
            ],  
            "Resource": "*"  
        },  
        {  
            "Effect": "Allow",  
            "Action": "iam:PassRole",  
            "Resource": "arn:aws:iam::123456789012:role/S3Access"  
        }  
    ]  
}
```

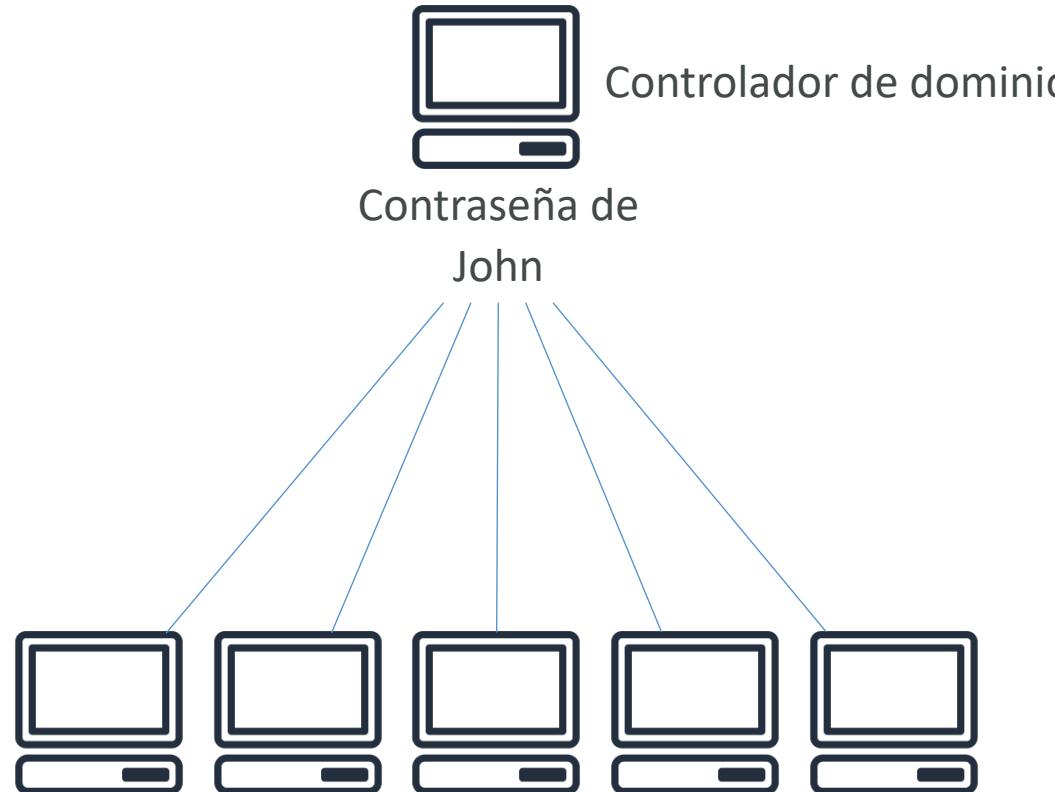
¿Se puede pasar un rol a cualquier servicio?

- **No: los roles sólo pueden pasar a lo que su confianza permite**
- Una política de confianza para el rol que permita al servicio asumir el rol

```
{  
    "Version": "2012-10-17",  
    "Statement": {  
        "Sid": "TrustPolicyStatementThatAllowsEC2ServiceToAssumeTheAttachedRole",  
        "Effect": "Allow",  
        "Principal": {  
            "Service": "ec2.amazonaws.com"  
        },  
        "Action": "sts:AssumeRole"  
    }  
}
```

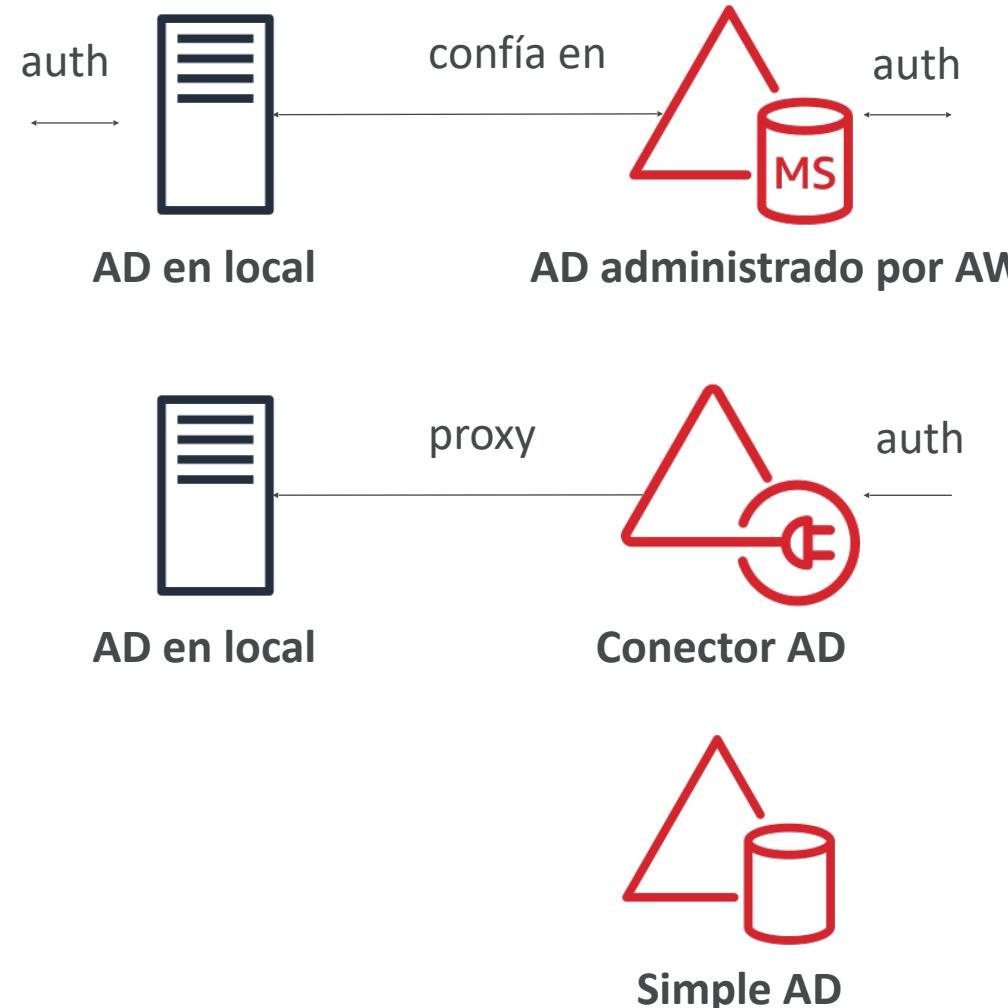
¿Qué es Microsoft Active Directory (AD)?

- Se encuentra en cualquier Servidor Windows con Servicios de Dominio AD
- Base de datos de **objetos**: Cuentas de usuario, ordenadores, impresoras, archivos compartidos, grupos de seguridad
- Gestión centralizada de la seguridad, crear cuenta, asignar permisos
- Los objetos se organizan en **árboles**
- Un grupo de árboles es un **bosque**



Servicios de directorio de AWS

- **Microsoft AD administrado por AWS**
 - Crea tu propio AD en AWS, administra usuarios localmente, soporta MFA
 - Establece conexiones de "confianza" con tu AD local
- **Conektor AD**
 - Directory Gateway (proxy) para redirigir al AD local, soporta MFA
 - Los usuarios se gestionan en el AD local
- **AD simple**
 - Directorio gestionado compatible con AD en AWS
 - No se puede unir con AD local



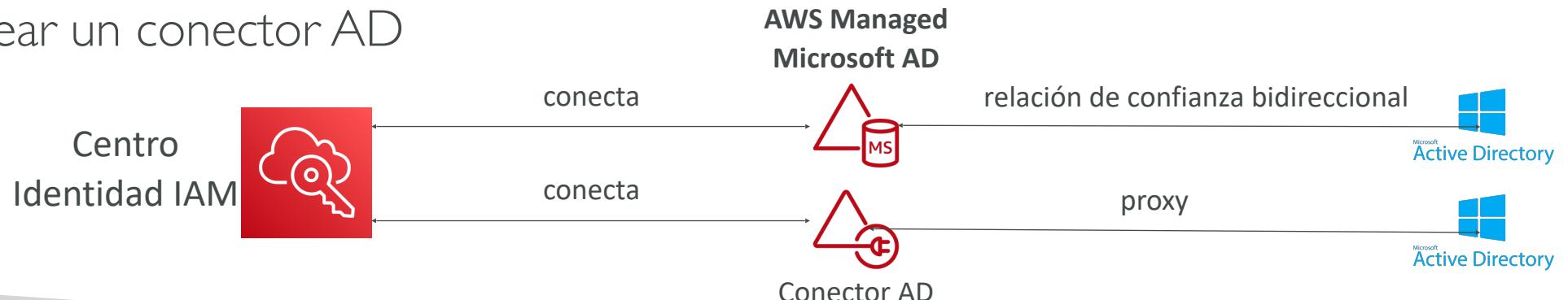
Centro de Identidades IAM - Configuración del Directorio Activo

- **Conectarse a un Microsoft AD (Servicio de directorio) administrado por AWS**
 - La integración está fuera de la caja



- **Conectarse a un directorio autogestionado**

- Crear una relación de confianza bidireccional utilizando Microsoft AD administrado por AWS
- Crear un conector AD

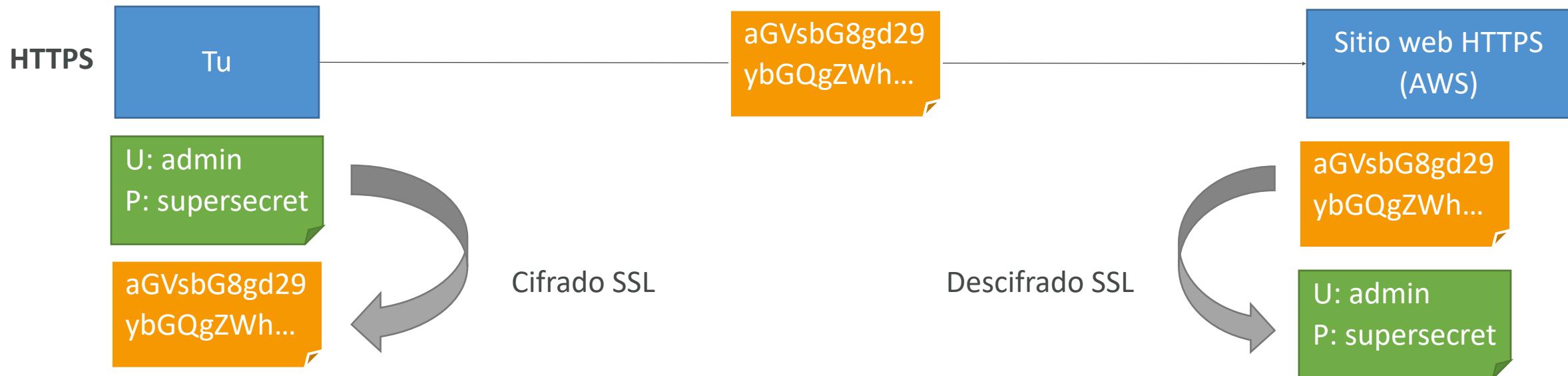


Seguridad y cifrado de AWS

KMS, Encryption SDK, SSM Parameter Store

¿Por qué cifrado? Cifrado en vuelo (SSL)

- Los datos se cifran antes de enviarlos y se descifran después de recibirlos
- Los certificados SSL ayudan al cifrado (HTTPS)
- El cifrado en vuelo garantiza que no pueda producirse un ataque MITM (man in the middle attack)



¿Por qué cifrado?

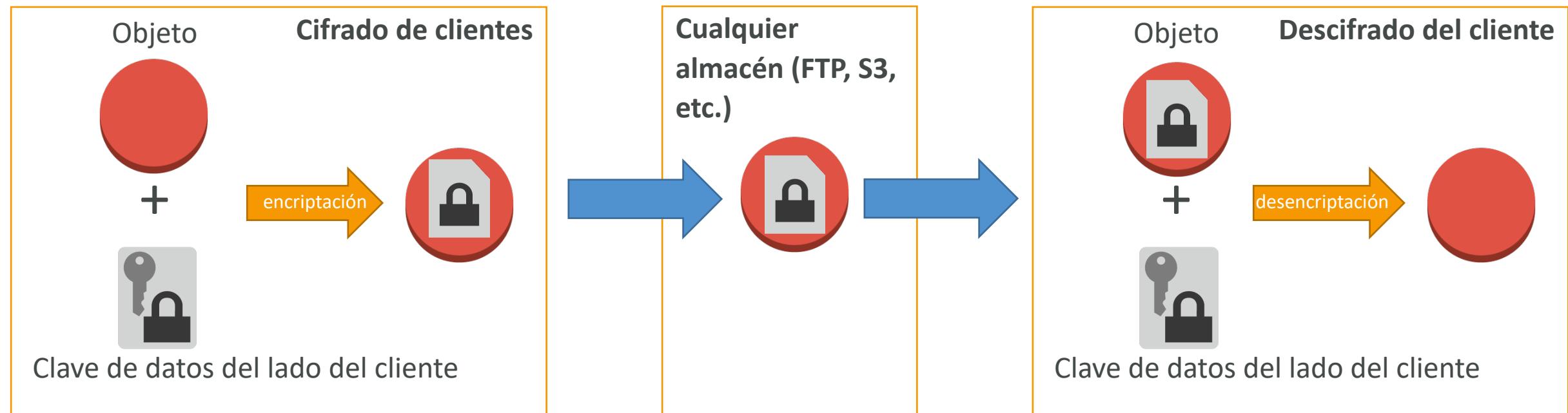
Cifrado del lado del servidor en reposo

- Los datos se cifran después de ser recibidos por el servidor
- Los datos se descifran antes de ser enviados
- Se almacenan cifrados gracias a una clave (normalmente una clave de datos)
- Las claves de cifrado/descifrado deben gestionarse en algún lugar y el servidor debe tener acceso a ellas



¿Por qué cifrado? Cifrado del lado del cliente

- Los datos son cifrados por el cliente y nunca descifrados por el servidor
- Los datos serán descifrados por un cliente receptor
- El servidor no debería poder descifrar los datos



AWS KMS (Servicio de administración de claves)



- Cada vez que oigas "cifrado" para un servicio de AWS, lo más probable es que se trate de KMS
- AWS gestiona las claves de cifrado por nosotros
- Totalmente integrado con IAM para la autorización
- Forma sencilla de controlar el acceso a tus datos
- Capaz de auditar el uso de claves KMS mediante CloudTrail
- Perfectamente integrado en la mayoría de los servicios de AWS (EBS, S3, RDS, SSM...)
- **Nunca jamás almacenes tus secretos en texto plano, ¡especialmente en tu código!**
 - El cifrado de claves KMS también está disponible a través de llamadas a la API (SDK, CLI)
 - Los secretos cifrados pueden almacenarse en el código / variables de Entorno

Tipos de claves KMS

- **Claves KMS es el nuevo nombre de Clave Maestra de Cliente KMS**
- **Simétrica (claves AES-256)**
 - Clave de cifrado única que se utiliza para cifrar y descifrar
 - Los servicios de AWS integrados con KMS utilizan CMK Simétricas
 - Nunca tienes acceso a la clave KMS sin cifrar (debes llamar a la API KMS para utilizarla)
- **Asimétrica (pares de claves RSA y ECC)**
 - Par de claves pública (cifrar) y privada (descifrar)
 - Se utiliza para operaciones de Cifrar/Descifrar o Firmar/Verificar
 - La clave pública se puede descargar, pero no puedes acceder a la Clave Privada sin cifrar

AWS KMS (Key Management Service)



- Tipos de claves KMS:
 - Claves propias de AWS (gratis): SSE-S3, SSE-SQS, SSE-DDB (clave por defecto)
 - Clave administrada por AWS: **gratis** (aws/nombre-servicio, ejemplo: aws/rds o aws/ebs)
 - Claves gestionadas por el cliente creadas en KMS: **I \$ / mes**
 - Claves gestionadas por el cliente importadas (deben ser claves simétricas): **I \$ / mes**
 - + pagar por llamada API a KMS (0,03 \$ / 10000 llamadas)
- Rotación automática de la clave:
 - Clave KMS gestionada por AWS: automática cada 1 año
 - Clave KMS gestionada por el cliente: (debe estar activada) automática cada 1 año
 - Clave KMS importada: sólo es posible la rotación manual mediante alias

Encryption key management

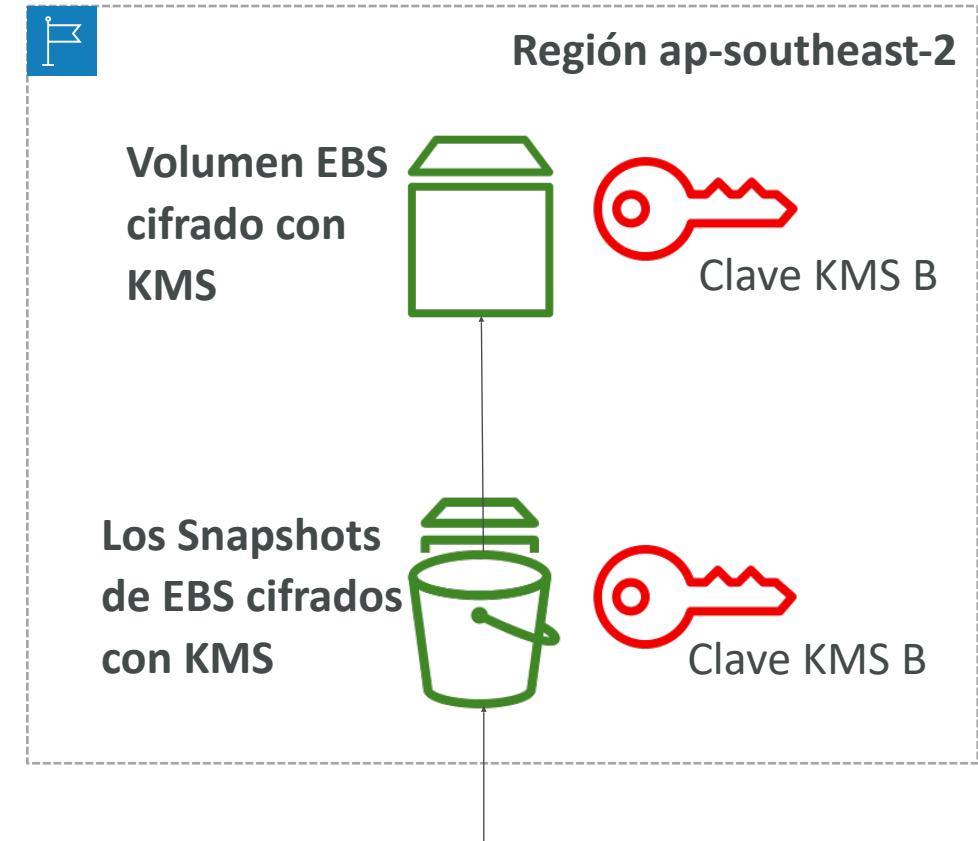
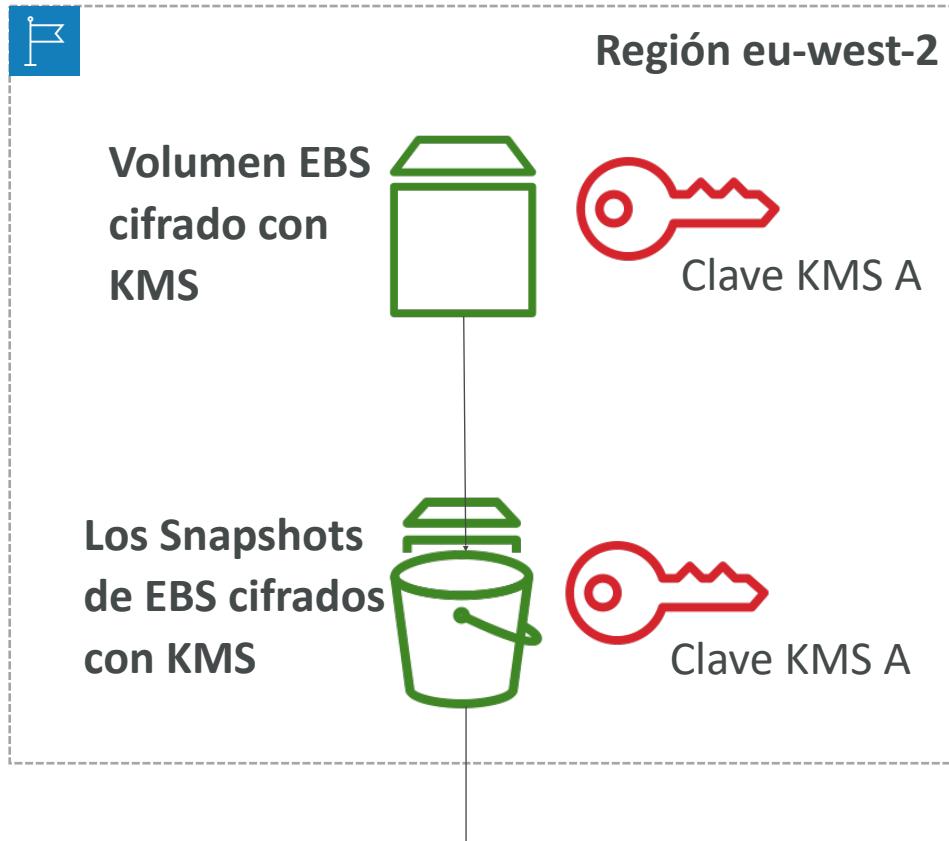
Owned by Amazon DynamoDB

AWS managed key [Lea](#)
Key alias: aws/dynamodb.

Stored in your account,
and owned and managed by you

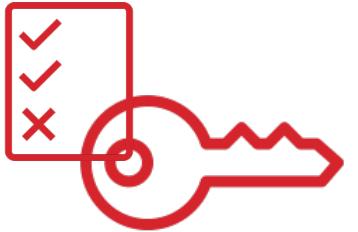


Copiar Snapshots entre regiones



Reencryptación KMS con clave B KMS

Políticas clave KMS



- Controlar el acceso a las claves KMS, "similar" a las políticas de bucket S3
- Diferencia: no puedes controlar el acceso sin ellas
- **Política de claves KMS por defecto:**
 - Se crea si no proporcionas una Política de Claves KMS específica
 - Acceso completo a la clave para el usuario root = toda la cuenta de AWS
- **Política de claves KMS personalizada:**
 - Define los usuarios y roles que pueden acceder a la clave KMS
 - Define quién puede administrar la clave
 - Útil para el acceso entre cuentas de tu clave KMS

Copiar Snapshots entre cuentas

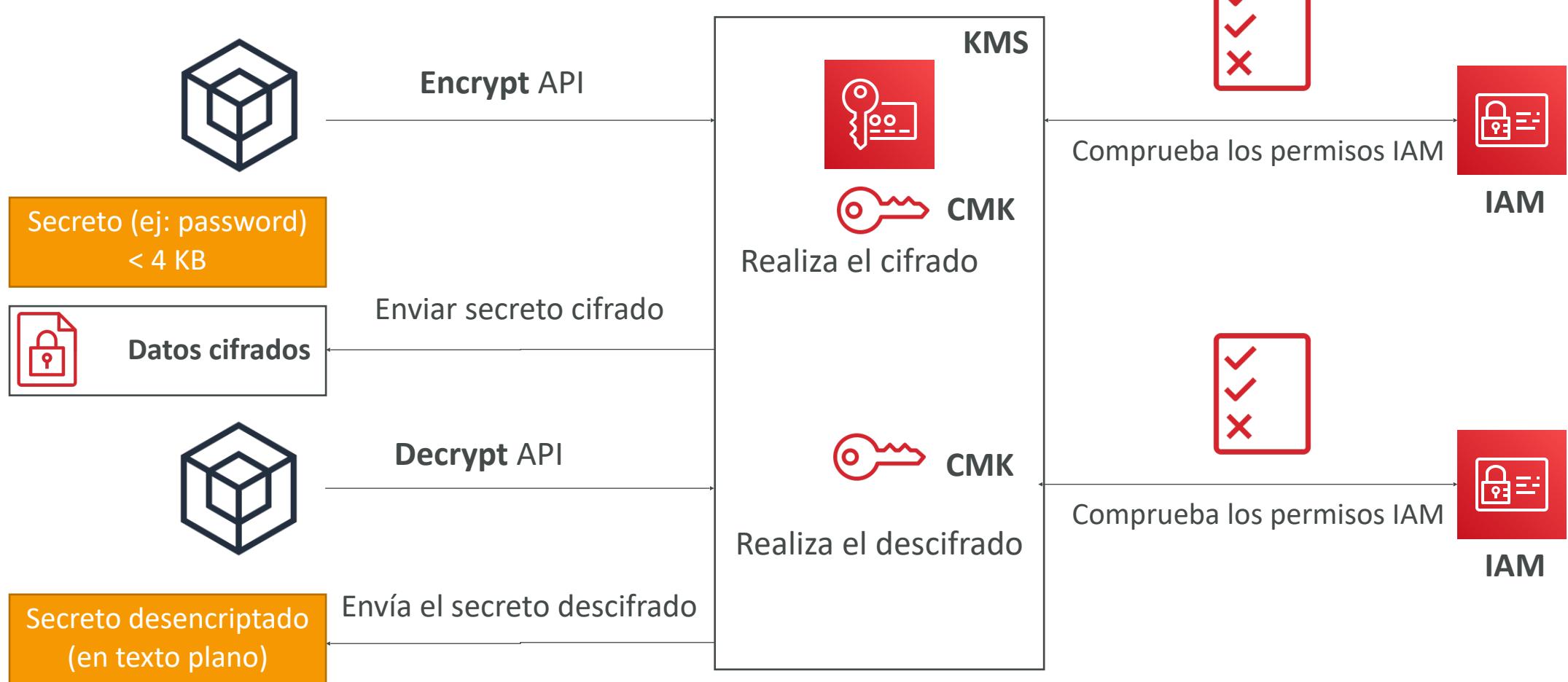
1. Crea una Snapshot, cifrada con tu propia Clave KMS (Clave Gestionada por el Cliente)
2. **Adjunta una Política de Clave KMS para autorizar el acceso entre cuentas**
3. Comparte la Snapshot cifrada
4. (en destino) Crea una copia de la Snapshot, cífrala con una CMK en tu cuenta
5. Crea un volumen a partir de la Snapshot

```
{  
  "Sid": "Allow use of the key with destination account",  
  "Effect": "Allow",  
  "Principal": {  
    "AWS": "arn:aws:iam::TARGET-ACCOUNT-ID:role/ROLENAMESPACE"  
  },  
  "Action": [  
    "kms:Decrypt",  
    "kms>CreateGrant"  
  ],  
  "Resource": "*",  
  "Condition": {  
    "StringEquals": {  
      "kms:ViaService": "ec2.REGION.amazonaws.com",  
      "kms:CallerAccount": "TARGET-ACCOUNT-ID"  
    }  
  }  
}
```

Política de claves KMS

¿Cómo funciona KMS?

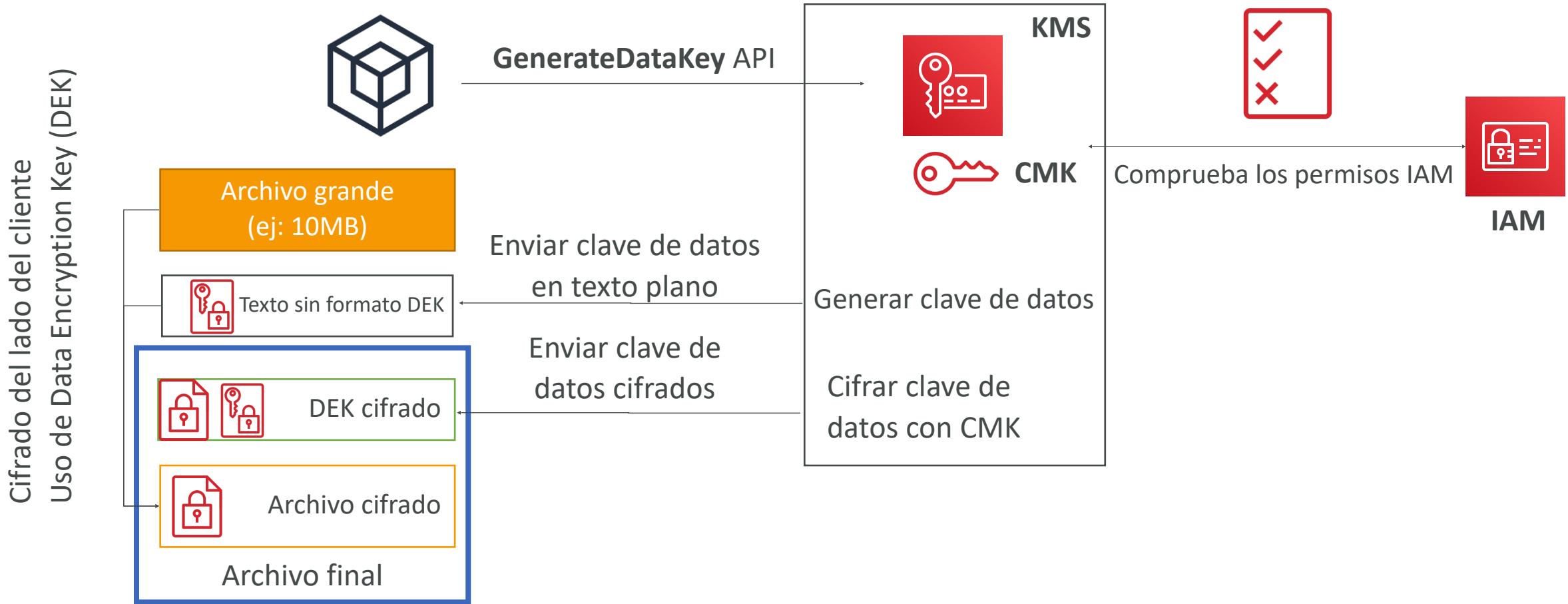
API - Cifrado y descifrado



Cifrado de sobre / Envelope Encryption

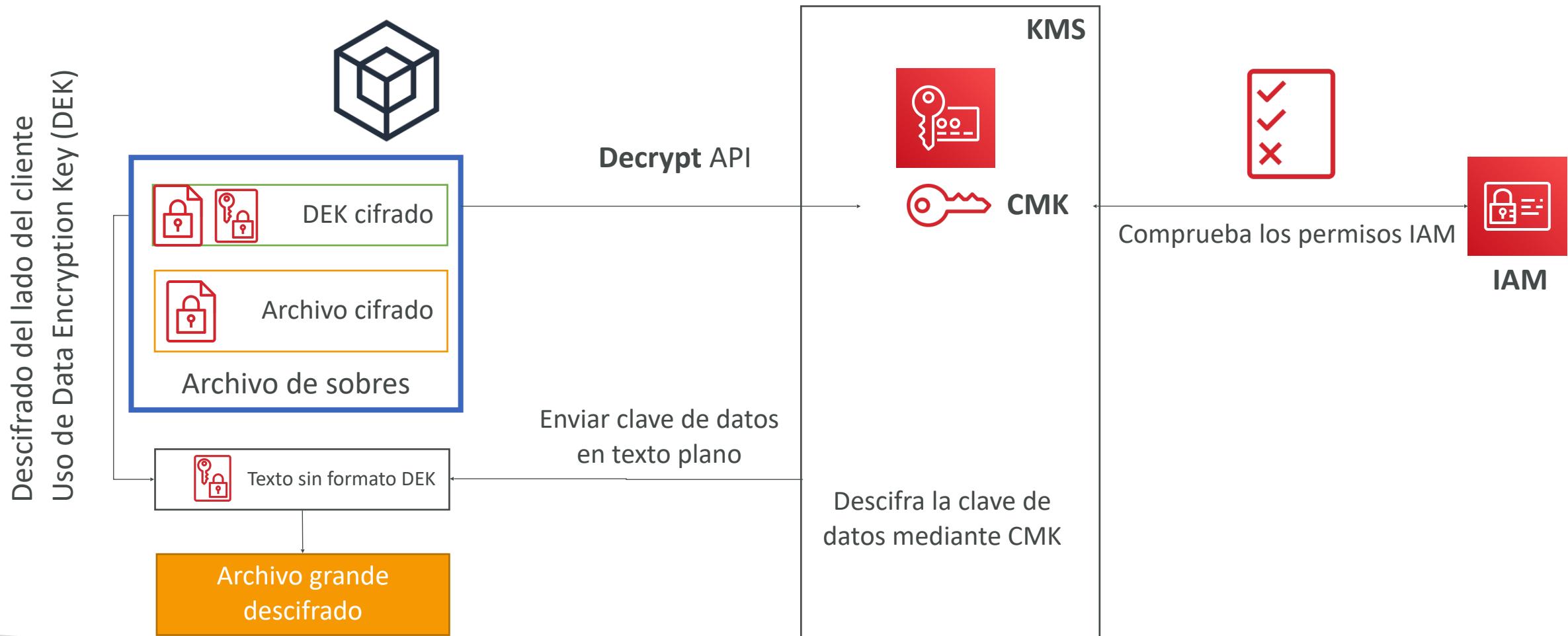
- La llamada a la API de cifrado de KMS tiene un límite de 4 KB
- Si quieras cifrar >4 KB, tenemos que usar el **Cifrado de sobre / Envelope Encryption**
- La principal API que nos ayudará es la API **GenerateDataKey**
- Para el examen: todo lo que supere los 4 KB de datos que deban cifrarse debe utilizar la API Envelope Encryption == GenerateDataKey

Profundización en el cifrado de sobres API **GenerateDataKey**



Profundizar en el cifrado de sobres

Descifrar datos de sobres





SDK de cifrado

- El SDK de cifrado de AWS implementó el cifrado de sobres por nosotros
 - El SDK de cifrado también existe como herramienta CLI que podemos instalar
 - Implementaciones para Java, Python, C, JavaScript
-
- **Característica - Almacenamiento en caché de claves de datos:**
 - Reutiliza las claves de datos en lugar de crear otras nuevas para cada cifrado.
 - Ayuda a reducir el número de llamadas al KMS con una contrapartida de seguridad
 - Utiliza LocalCryptoMaterialsCache (edad máxima, bytes máximos, número máximo de mensajes)

KMS Simétrica - Resumen de la API



- **Encrypt:** cifra hasta 4 KB de datos a través de KMS
- **GenerateDataKey:** genera una única clave simétrica de datos (DEK)
 - Devuelve una copia en texto plano de la clave de datos
 - Y una copia cifrada con la CMK que especifiques
- **GenerateDataKeyWithoutPlaintext:**
 - Genera una DEK para utilizarla en algún momento (no inmediatamente)
 - DEK cifrado con la CMK que especifiques (debes utilizar Descifrar después)
- **Decrypt:** descifra hasta 4 KB de datos (incluidas las Claves de Cifrado de Datos)
- **GenerateRandom:** Devuelve una cadena de bytes aleatoria

Cuotas de solicitud KMS

- Cuando superas la cuota de peticiones, recibes una **ThrottlingException**:

```
You have exceeded the rate at which you may call KMS. Reduce the frequency of your calls.  
(Service: AWSKMS; Status Code: 400; Error Code: ThrottlingException; Request ID: <ID>)
```

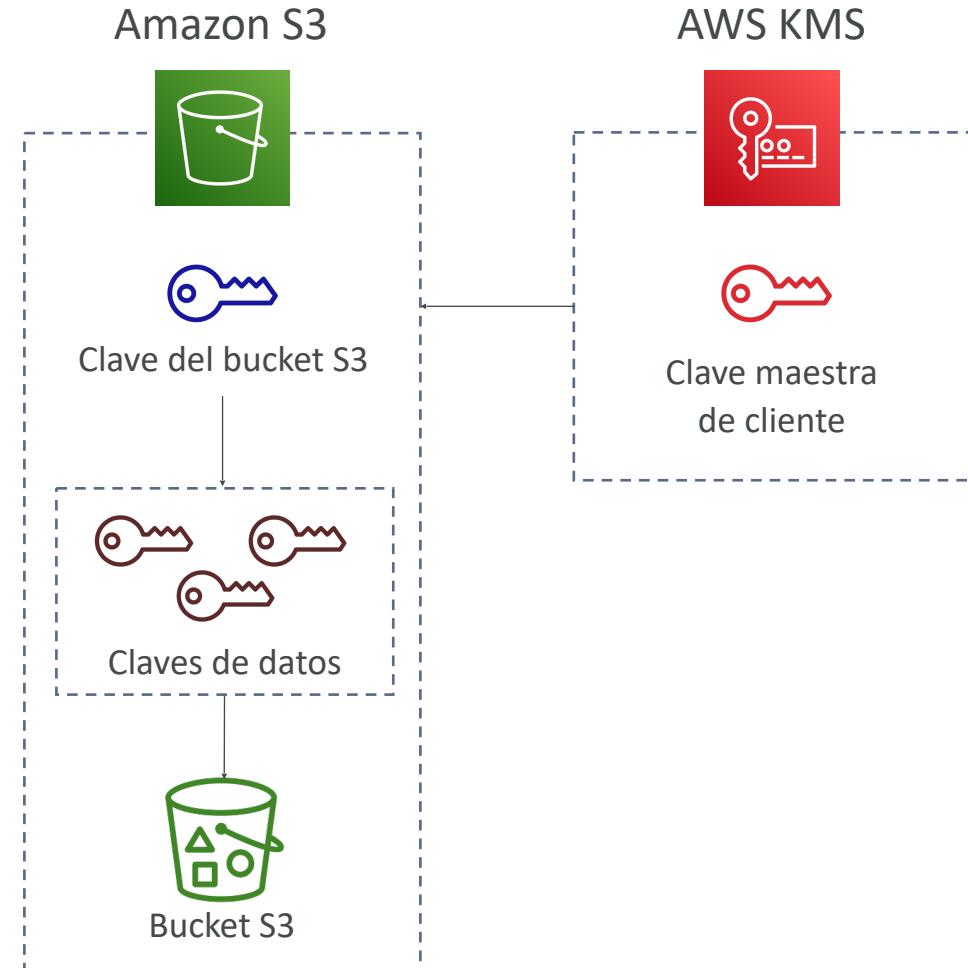
- Para responder, utilizan el backoff exponencial (backoff y reintento)
- Para las operaciones criptográficas, comparten una cuota
- Esto incluye las peticiones realizadas por AWS en tu nombre (ej: SSE-KMS)
- Para GenerateDataKey, considera el uso de la caché DEK del SDK de cifrado
- **Puedes solicitar un aumento de las cuotas de petición a través de la API o de AWS support**

Cuotas de solicitud KMS

Funcionamiento de la API	Cuotas de petición (por segundo)
Decrypt Encrypt GenerateDataKey (simétrica) GenerateDataKeyWithoutPlaintext (simétrica) GenerateRandom ReEncrypt Sign (asimétrica) Verify (asimétrica)	<p>Estas cuotas compartidas varían según la región de AWS y el tipo de CMK utilizado en la petición. Cada cuota se calcula por separado.</p> <p>Cuota CMK simétrica:</p> <ul style="list-style-type: none">• 5,500 (compartidos)• 10,000 (compartidos) en las siguientes Regiones:<ul style="list-style-type: none">• us-east-2, ap-southeast-1, ap-southeast-2, ap-northeast-1, eu-central-1, eu-west-2• 30,000 (compartidos) en las siguientes regiones:<ul style="list-style-type: none">• us-east-1, us-west-2, eu-west-1 <p>Cuota CMK asimétrica:</p> <ul style="list-style-type: none">• 500 (compartido) para CMKs RSA• 300 (compartidos) para CMK de curva elíptica (ECC)

Clave de bucket S3 para cifrado SSE-KMS

- Nuevo ajuste para disminuir...
 - El número de llamadas API realizadas a KMS desde S3 en un 99%
 - Coste total del cifrado de KMS con Amazon S3 en un 99%
- Esto aprovecha las claves de datos
 - Se genera una "clave de bucket de S3"
 - Esa clave se utiliza para cifrar objetos KMS con nuevas claves de datos
- Verás **menos eventos KMS en CloudTrail**



Políticas de clave - Ejemplos



Política de claves KMS por defecto

```
{  
  "Effect": "Allow",  
  "Action": "kms:*",  
  "Principal": {  
    "AWS": "arn:aws:iam::123456789012:root"  
  },  
  "Resource": "*"  
}
```

Permitir usuario federado

```
{  
  "Effect": "Allow",  
  "Action": [  
    "kms:Encrypt",  
    "kms:Decrypt",  
    "kms:ReEncrypt*",  
    "kms:GenerateDataKey*",  
    "kms:DescribeKey"  
  ],  
  "Principal": {  
    "AWS": "arn:aws:sts::123456789012:federated-user/user-name"  
  },  
  "Resource": "*"  
}
```

Principales opciones de las políticas IAM

- Cuenta AWS y usuario root

```
"Principal": { "AWS": "123456789012" }
```

```
"Principal": { "AWS": "arn:aws:iam::123456789012:root" }
```

- Roles IAM

```
"Principal": { "AWS": "arn:aws:iam::123456789012:role/role-name" }
```

- Sesiones de rol IAM

```
"Principal": { "AWS": "arn:aws:sts::123456789012:assumed-role/role-name/role-session-name" }
```

```
"Principal": { "Federated": "cognito-identity.amazonaws.com" }
```

```
"Principal": { "Federated": "arn:aws:iam::123456789012:saml-provider/provider-name" }
```

Principales opciones de las políticas IAM

- Usuarios IAM

```
"Principal": { "AWS": "arn:aws:iam::123456789012:user/user-name" }
```

- Sesiones federadas de Usuario

```
"Principal": { "AWS": "arn:aws:sts::123456789012:federated-user/user-name" }
```

- Servicios AWS

```
"Principal": {  
    "Service": [  
        "ecs.amazonaws.com",  
        "elasticloadbalancing.amazonaws.com"  
    ]  
}
```

- Todos los principales

```
"Principal": "*"  
"Principal": { "AWS": "*" }
```

CloudHSM



- KMS => AWS administra el software de cifrado
- CloudHSM => AWS proporciona **hardware** de cifrado
- Hardware dedicado (HSM = Hardware Security Module)
- Gestionas por completo tus propias claves de cifrado (no AWS)
- El dispositivo HSM es a prueba de manipulaciones, cumple la normativa:
 - FIPS 140-2 Level 3
- Soporta cifrado simétrico y **asimétrico** (claves SSL/TLS)
- No hay ningún nivel gratuito disponible
- Debes utilizar el software cliente CloudHSM
- Redshift soporta CloudHSM para el cifrado de bases de datos y la gestión de claves
- **Buena opción para utilizar con cifrado SSE-C**

Diagrama de CloudHSM



Permisos IAM:

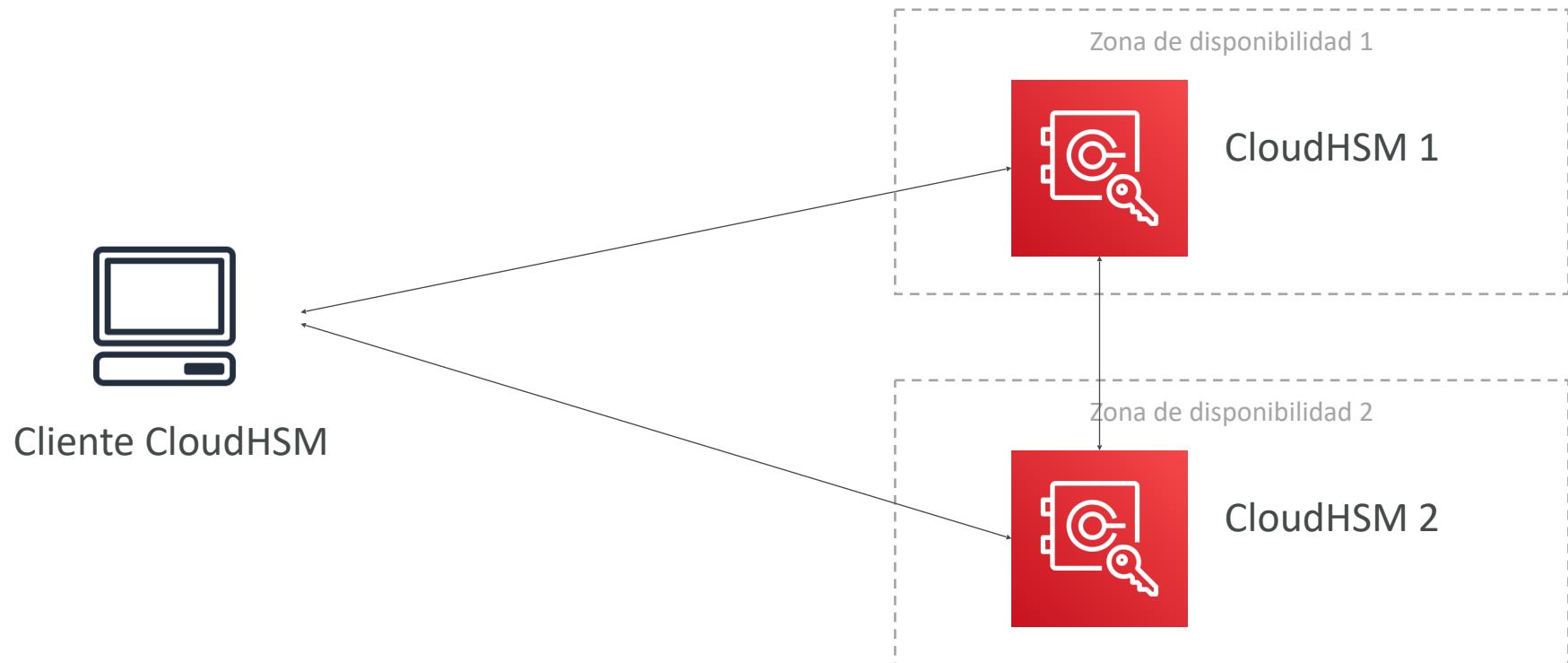
- CRUD un Cluster HSM

Software CloudHSM:

- Gestionar las claves
- Gestionar los usuarios

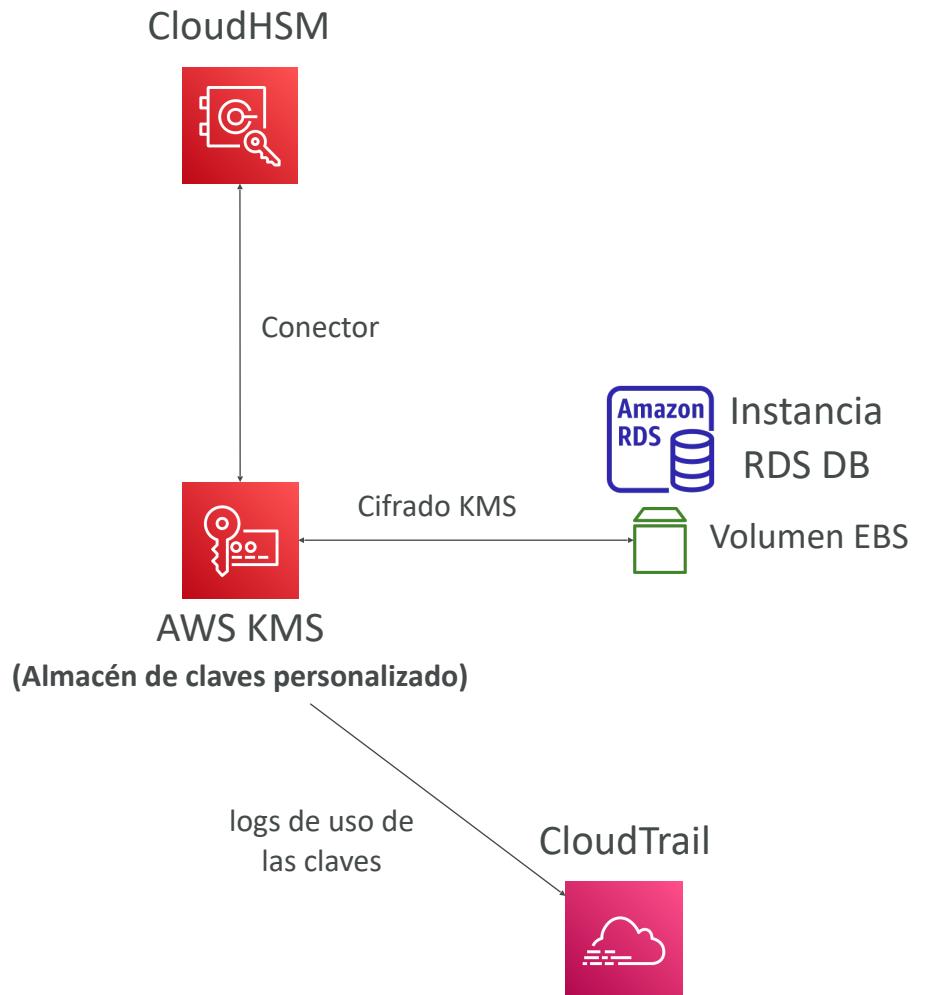
CloudHSM - Alta disponibilidad

- Los Clusters CloudHSM están repartidos en Multi AZ (Alta disponibilidad)
- Gran disponibilidad y durabilidad



CloudHSM - Integración con los servicios de AWS

- Mediante la integración con AWS KMS
- Configurar el almacén de claves personalizadas KMS con CloudHSM
- Ejemplo: EBS, S3, RDS ...



CloudHSM vs. KMS

Función	AWS KMS	AWS CloudHSM
Alquiler	Multiinquilino	Un único inquilino
Estándar	FIPS 140-2 Level 2	FIPS 140-2 Level 3
Claves maestras	<ul style="list-style-type: none"> CMK propia de AWS CMK administrada por AWS CMK administrada por el cliente 	CMK gestionada por el cliente
Tipos de claves	<ul style="list-style-type: none"> Simétrica Asimétrica Firma digital 	<ul style="list-style-type: none"> Simétrica Asimétrica Firma digital y Hashing
Accesibilidad de clave	Accesible en varias regiones de AWS (no se puede acceder a las claves fuera de la región en la que se crea)	<ul style="list-style-type: none"> Desplegada y gestionada en una VPC Puede compartirse entre VPCs (VPC Peering)
Aceleración criptográfica	Ninguna	<ul style="list-style-type: none"> Aceleración SSL/TLS Aceleración Oracle TDE
Acceso y autenticación	AWS IAM	Creas usuarios y gestionas sus permisos

CloudHSM vs. KMS

Función	AWS KMS	AWS CloudHSM
Alta disponibilidad	Servicio gestionado de AWS	Añade varios HSM en diferentes AZ
Capacidad de auditoría	<ul style="list-style-type: none">• CloudTrail• CloudWatch	<ul style="list-style-type: none">• CloudTrail• CloudWatch• Soporte MFA
Nivel gratuito	Sí	No

SSM Parameter Store

Almacén de parámetros SSM

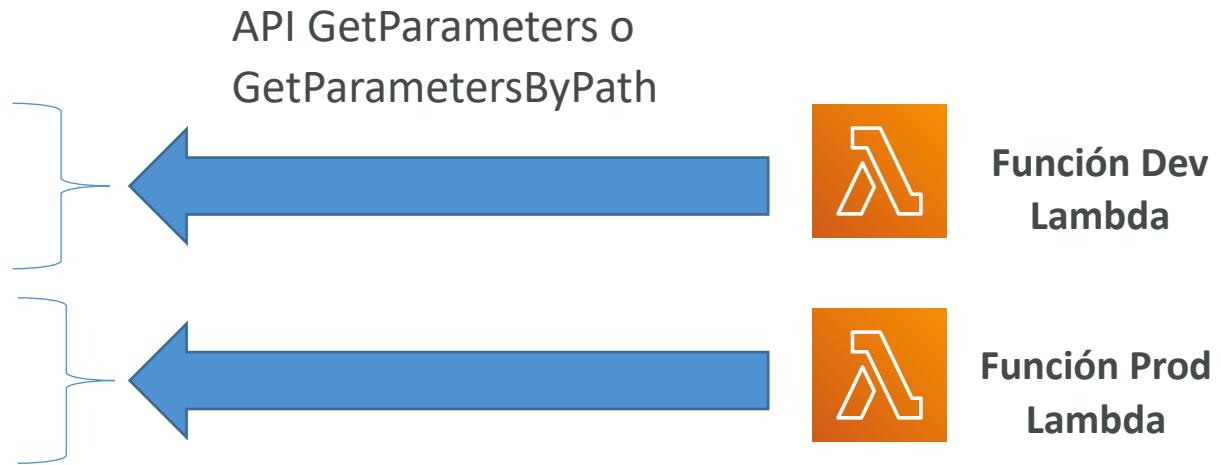


- Almacenamiento seguro de la configuración y los secretos
- Cifrado sin fisuras opcional mediante KMS
- SDK sin servidor, escalable, duradero y sencillo
- Seguimiento de versiones de configuraciones / secretos
- Seguridad mediante IAM
- Notificaciones con Amazon EventBridge
- Integración con CloudFormation



Jerarquía del almacén de parámetros SSM

- /mi-departamento/
 - mi-app/
 - dev/
 - db-url
 - db-contraseña
 - prod/
 - db-url
 - db-contraseña
 - otra-app/
 - /otro-departamento/
 - /aws/reference/secretsmanager/secret_ID_in_Secrets_Manager
 - /aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-x86_64-gp2 (público)



Niveles de parámetros estándar y avanzado

	Estándar	Avanzado
Número total de parámetros permitidos (por cuenta AWS y Región)	10,000	100,000
Tamaño máximo del valor de un parámetro	4 KB	8 KB
Políticas de parámetros disponibles	No	Si
Coste	Sin coste adicional	Se aplican cargos
Precios de almacenamiento	Gratis	0,05 \$ por parámetro avanzado al mes

Políticas de parámetros (para parámetros avanzados)

- Permite asignar un TTL a un parámetro (fecha de caducidad) para forzar la actualización o eliminación de datos sensibles como contraseñas
- Puede asignar varias políticas a la vez

Expiration (para eliminar un parámetro)

```
{  
  "Type": "Expiration",  
  "Version": "1.0",  
  "Attributes": {  
    "Timestamp": "2020-12-02T21:34:33.000Z"  
  }  
}
```

ExpirationNotification (EventBridge)

```
{  
  "Type": "ExpirationNotification",  
  "Version": "1.0",  
  "Attributes": {  
    "Before": "15",  
    "Unit": "Days"  
  }  
}
```

NoChangeNotification (EventBridge)

```
{  
  "Type": "NoChangeNotification",  
  "Version": "1.0",  
  "Attributes": {  
    "After": "20",  
    "Unit": "Days"  
  }  
}
```

AWS Secrets Manager

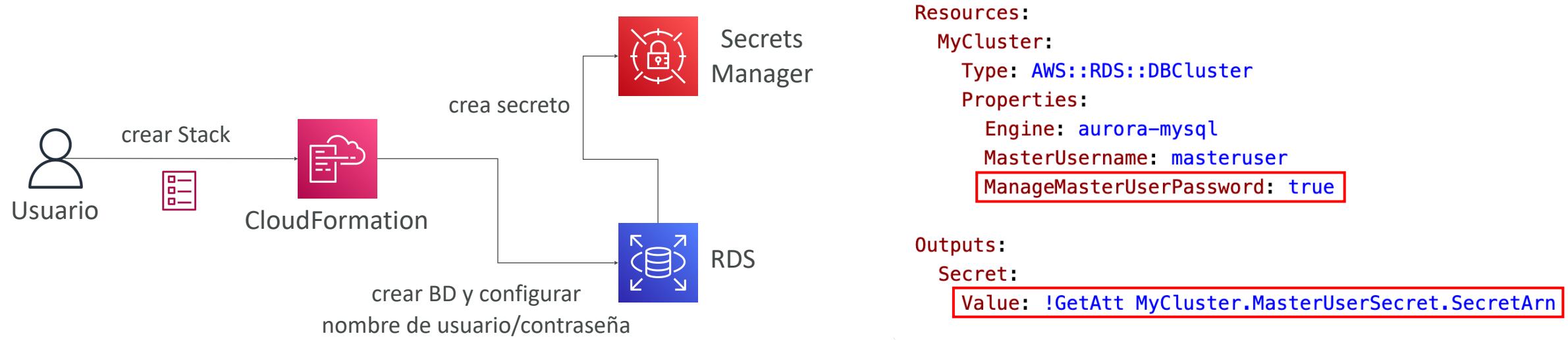


- Servicio más nuevo, pensado para almacenar secretos
- Capacidad para forzar la **rotación de secretos** cada X días
- Automatizar la generación de secretos en la rotación (utiliza Lambda)
- Integración con **Amazon RDS** (MySQL, PostgreSQL, Aurora)
- Los secretos se cifran mediante KMS
- Pensado principalmente para la integración con RDS

Secrets Manager

Integración de CloudFormation RDS y Aurora

- **ManageMasterUserPassword** - crea el secreto de administrador implícitamente
- RDS, Aurora gestionará el secreto en el Secrets Manager y su rotación



Secrets Manager CloudFormation - Referencia dinámica

```
Resources:  
  # Secret resource with a randomly generated password in its SecureString JSON  
  MyRDSDBInstanceRotationSecret:  
    Type: AWS::SecretsManager::Secret  
    Properties:  
      GenerateSecretString:  
        SecretStringTemplate: '{"username": "admin"}'  
        GenerateStringKey: password  
        PasswordLength: 16  
        ExcludeCharacters: "\"@/\\""
```

se genera el secreto

```
# RDS Instance resource. Its master username and password use dynamic references  
# to resolve values from Secrets Manager
```

```
MyRDSDBInstance:  
  Type: AWS::RDS::DBInstance  
  Properties:  
    DBInstanceClass: db.t2.micro  
    Engine: mysql  
    MasterUsername: !Sub "{{resolve:secretsmanager:${MyRDSDBInstanceRotationSecret}:username}}"  
    MasterUserPassword: !Sub "{{resolve:secretsmanager:${MyRDSDBInstanceRotationSecret}:password}}"
```

secreto de referencia en
la instancia RDS DB

```
# SecretTargetAttachment resource which updates the referenced Secret with properties  
# about the referenced RDS instance
```

```
SecretRDSDBInstanceAttachment:  
  Type: AWS::SecretsManager::SecretTargetAttachment  
  Properties:  
    TargetType: AWS::RDS::DBInstance  
    SecretId: !Ref MyRDSDBInstanceRotationSecret  
    TargetId: !Ref MyRDSDBInstance
```

vincular el secreto a la
instancia RDS DB

SSM Parameter Store vs Secrets Manager

- **Secrets Manager (\$\$\$):**

- Rotación automática de secretos con AWS Lambda
- Se proporciona la función Lambda para RDS, Redshift, DocumentDB
- El cifrado KMS es obligatorio
- Puede integrarse con CloudFormation

- **SSM Parameter Store (\$):**

- API simple
- Sin rotación secreta (se puede activar la rotación mediante Lambda activada por Eventos CW)
- El cifrado KMS es opcional
- Puede integrarse con CloudFormation
- Puedes extraer un secreto del Secrets Manager utilizando la API del SSM Parameter Store

SSM Parameter Store vs. Secrets Manager Rotación

AWS Secrets Manager



SSM Parameter Store

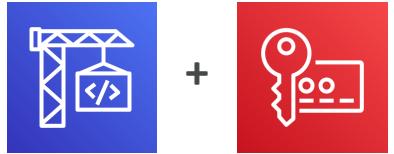




CloudWatch Logs - Cifrado

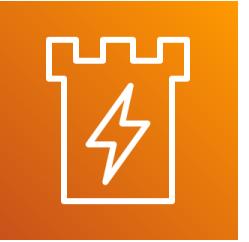
- Puedes cifrar logs de CloudWatch con claves KMS
- El cifrado se activa a nivel de grupo de logs, asociando una CMK a un grupo de logs, ya sea cuando creas el grupo de logs o después de que exista
- No puedes asociar una CMK a un grupo de logs utilizando la consola de CloudWatch
- Debes utilizar la API de logs de CloudWatch:
 - **associate-kms-key** : si el grupo de logs ya existe
 - **create-log-group**: si el grupo de logs aún no existe

Seguridad CodeBuild



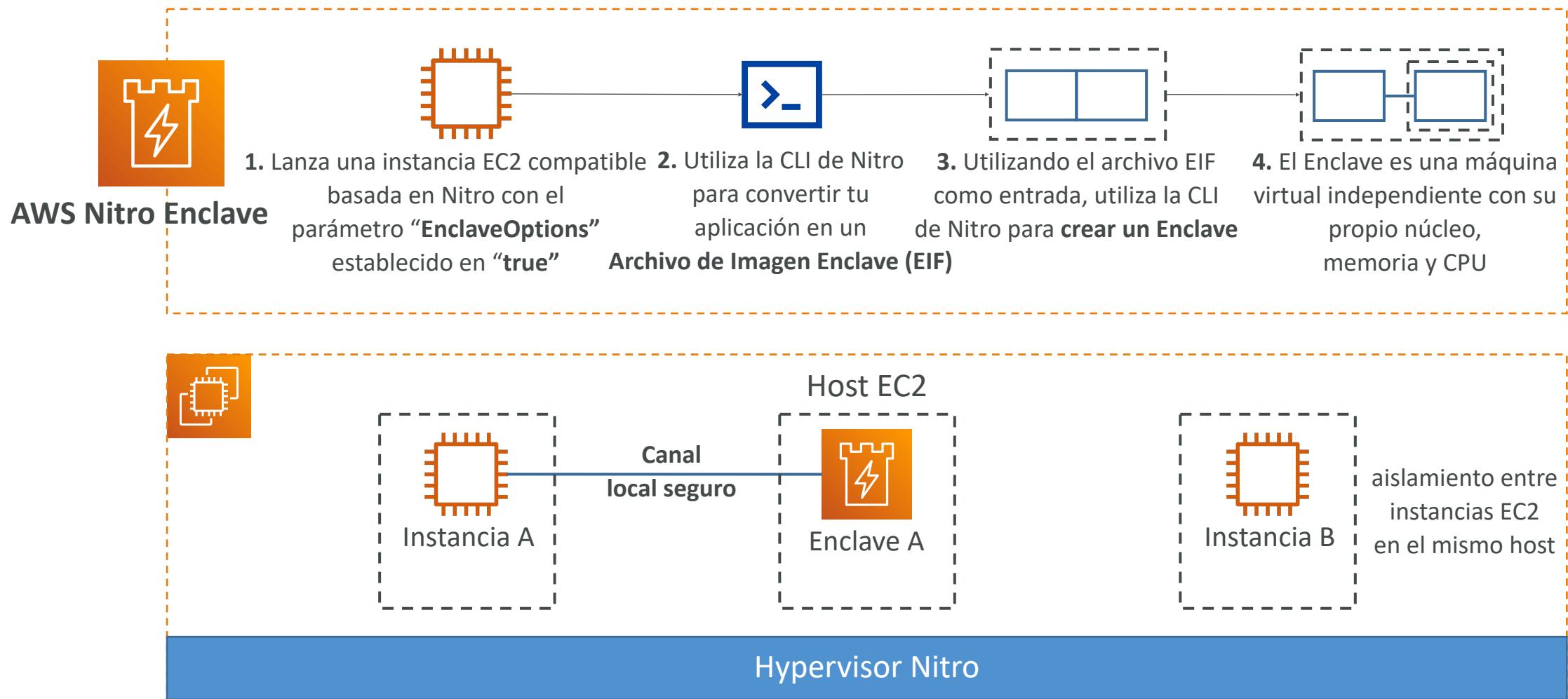
- Para acceder a los recursos de tu VPC, asegúrate de especificar una configuración de VPC para tu CodeBuild
- Secretos en CodeBuild:
- No los almacenes como texto plano en variables de entorno
- En cambio...
 - Las variables de entorno pueden hacer referencia a los parámetros de Parameter Store
 - Las variables de entorno pueden hacer referencia a los secretos de Secrets Manager

AWS Nitro Enclaves



- Procesa datos altamente sensibles en un entorno informático aislado
 - Información de identificación personal (PII), sanitaria, financiera, ...
- Máquinas virtuales totalmente aisladas, reforzadas y altamente restringidas
 - No es un contenedor, ni almacenamiento persistente, ni acceso interactivo, ni red externa
- Ayuda a reducir la superficie de ataque de las aplicaciones de procesamiento de datos sensibles
 - **Declaración criptográfica:** sólo el código autorizado puede ejecutarse en tu Enclave
 - Sólo los Enclaves pueden acceder a datos sensibles (integración con KMS)
- Casos de uso: protección de claves privadas, procesamiento de tarjetas de crédito, cálculo seguro entre múltiples partes...

AWS Nitro Enclaves



Otros servicios

Visión general de los Servicios que podrían surgir en algunas preguntas

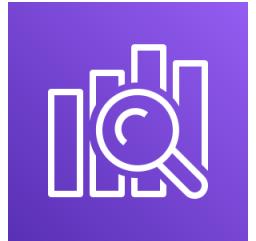


Amazon Simple Email Service (Amazon SES)

- **Servicio totalmente gestionado para enviar correos electrónicos de forma segura, global y a escala**
- Permite correos electrónicos entrantes/salientes
- Dashboards de reputación, perspectivas de rendimiento, información antispam
- Proporciona estadísticas como entregas de correos electrónicos, rebotes, resultados del bucle de retroalimentación, correos electrónicos abiertos
- Soporta DomainKeys Identified Mail (DKIM) y Sender Policy Framework (SPF)
- Despliegue de IP flexible: IP compartida, dedicada y propiedad del cliente
- Envía correos electrónicos con tu aplicación utilizando la consola de AWS, las API o SMTP
- Casos de uso: comunicaciones transaccionales, de marketing y masivas por correo electrónico

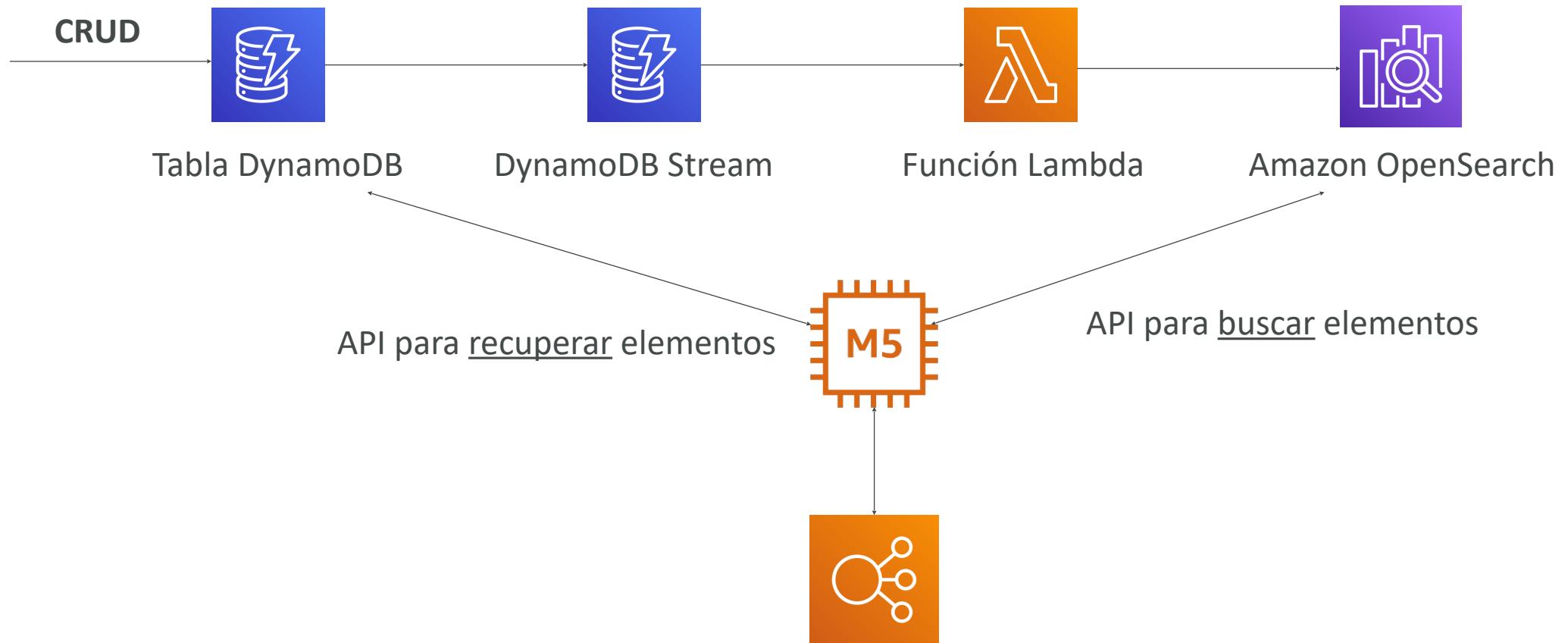


Amazon OpenSearch



- **Amazon OpenSearch es el sucesor de Amazon Elasticsearch**
- En DynamoDB, las consultas sólo existen por clave primaria o índices...
- **Con OpenSearch, puedes buscar en cualquier campo, incluso coincidencias parciales**
- Es habitual utilizar OpenSearch como complemento de otra base de datos
- OpenSearch requiere un Cluster de instancias (no sin servidor)
- No soporta SQL (tiene su propio lenguaje de consulta)
- Ingestión desde Kinesis Data Firehose, AWS IoT y CloudWatch Logs
- Seguridad mediante Cognito & IAM, cifrado KMS, TLS
- Viene con OpenSearch Dashboards (visualización)

Patrones de OpenSearch DynamoDB

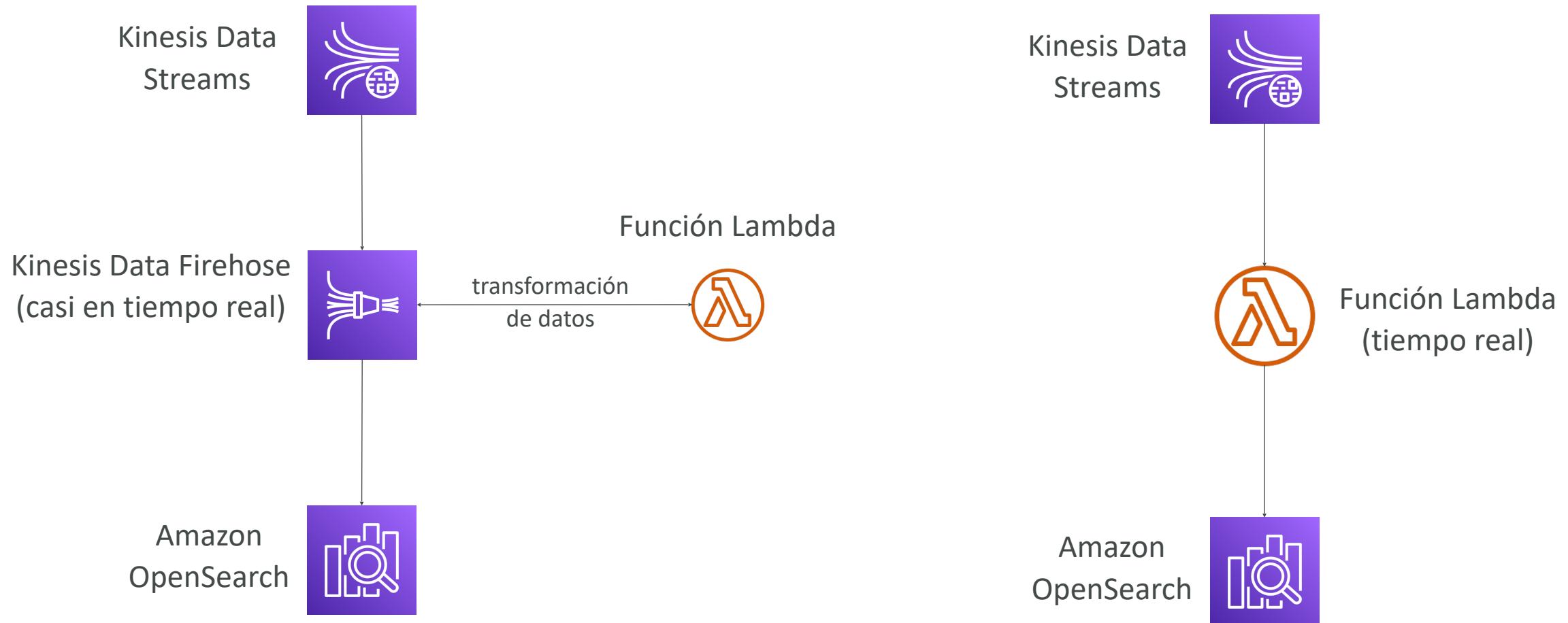


Patrones de OpenSearch CloudWatch Logs



Patrones de OpenSearch

Kinesis Data Streams y Kinesis Data Firehose



Amazon Athena



- Servicio de consulta **sin servidor** para analizar datos almacenados en Amazon S3.
 - Utiliza el lenguaje SQL estándar para consultar los archivos
 - Admite CSV, JSON, ORC, Avro y Parquet
 - Precio: 5 dólares por TB de datos analizados
 - Se utiliza habitualmente con Amazon Quicksight para la elaboración de informes y Dashboards
-
- **Casos de uso:** Inteligencia empresarial / análisis / informes, analizar y consultar registros de flujo de VPC, registros de ELB, **CloudTrail trails**, etc....
 - **Sugerencia de examen:** analizar datos en S3 utilizando SQL sin servidor, utilizar Athena

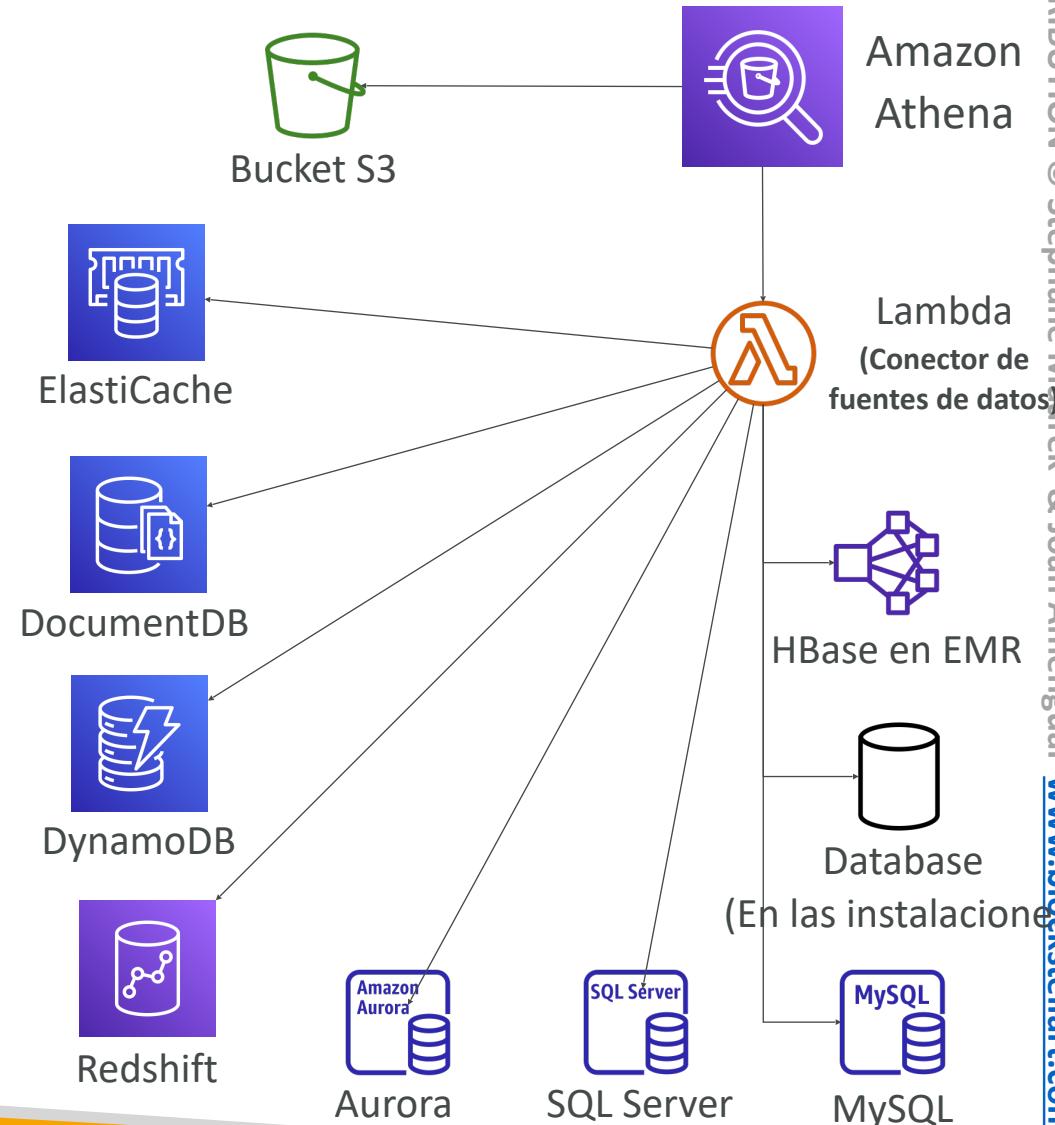


Amazon Athena – Mejora del rendimiento

- Utiliza **datos en columnas** para ahorrar costes
 - Se recomienda Apache Parquet u ORC
 - Enorme mejora del rendimiento
 - Utiliza Glue para convertir tus datos en Parquet u ORC
- **Comprime los datos** para recuperaciones más pequeñas (bzip2, gzip, lz4, snappy, zlip, zstd...)
- **Particionar** conjuntos de datos en S3 para facilitar la consulta en columnas virtuales
 - s3://yourBucket/pathToTable
 /<PARTITION_COLUMN_NAME>=<VALUE>
 /<PARTITION_COLUMN_NAME>=<VALUE>
 /<PARTITION_COLUMN_NAME>=<VALUE>
 /etc...
 - Ejemplo: s3://athena-examples/flight/parquet/year=1991/month=1/day=1/
- **Utiliza archivos de mayor tamaño (> 128 MB)** para minimizar la sobrecarga

Amazon Athena – Consulta federada

- Permite ejecutar consultas SQL en datos almacenados en fuentes de datos relacionales, no relacionales, de objetos y personalizadas (AWS o en las instalaciones).
- Utiliza conectores de fuentes de datos que se ejecutan en AWS Lambda para ejecutar consultas federadas (por ejemplo, CloudWatch Logs, DynamoDB, RDS, ...)
- Almacena los resultados de nuevo en Amazon S3

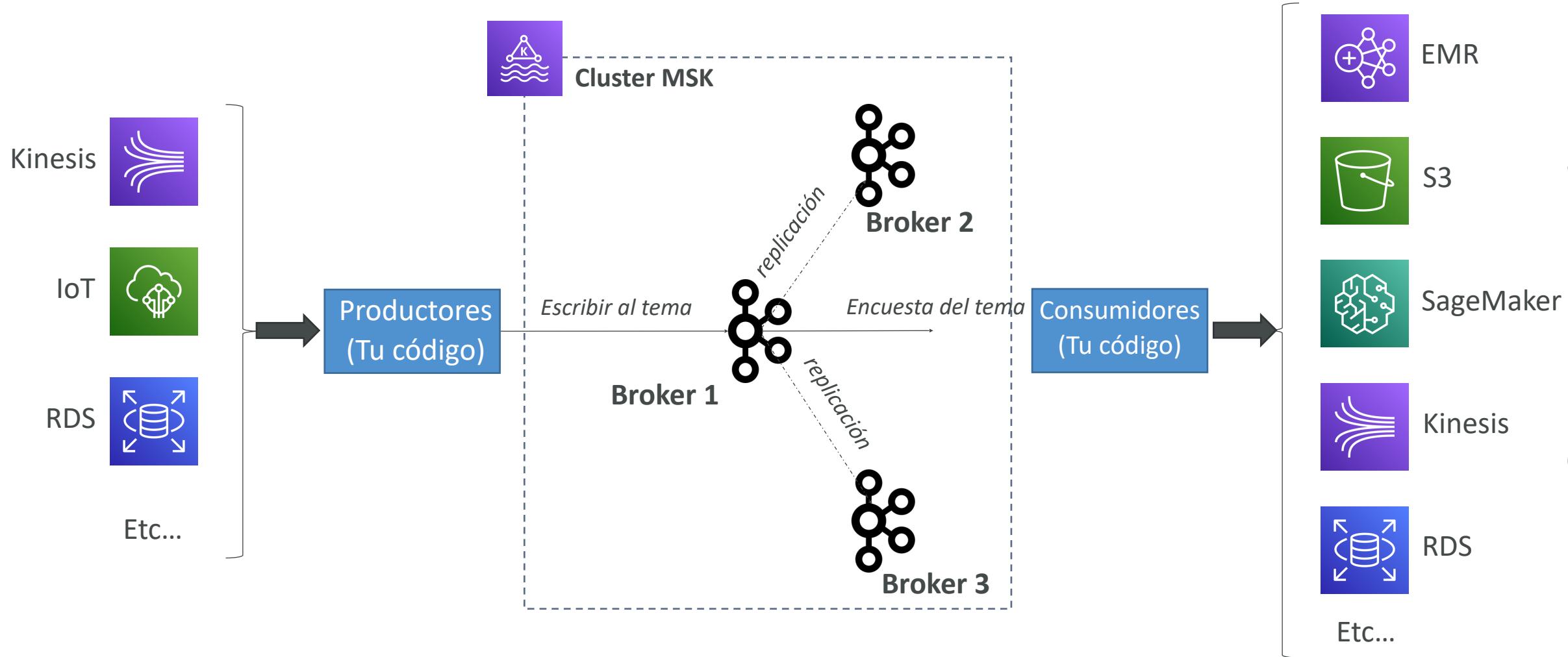


Amazon Managed Streaming para Apache Kafka (Amazon MSK)



- Alternativa a Amazon Kinesis
- Apache Kafka totalmente administrado en AWS
 - Permite crear, actualizar y eliminar clústeres
 - MSK crea y administra nodos brokers de Kafka y nodos Zookeeper por ti
 - Implementa el clúster MSK en tu VPC, multi-AZ (hasta 3 para Alta Disponibilidad)
 - Recuperación automática de fallos comunes de Apache Kafka
 - Los datos se almacenan en volúmenes EBS **durante todo el tiempo que deseas**
- **MSK sin servidor**
 - Ejecuta Apache Kafka en MSK sin gestionar la capacidad
 - MSK aprovisiona automáticamente los recursos y escala la computación y el almacenamiento

Apache Kafka a alto nivel

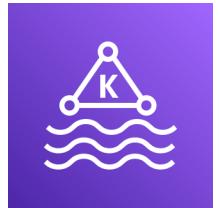


Kinesis Data Streams frente a Amazon MSK



Kinesis Data Streams

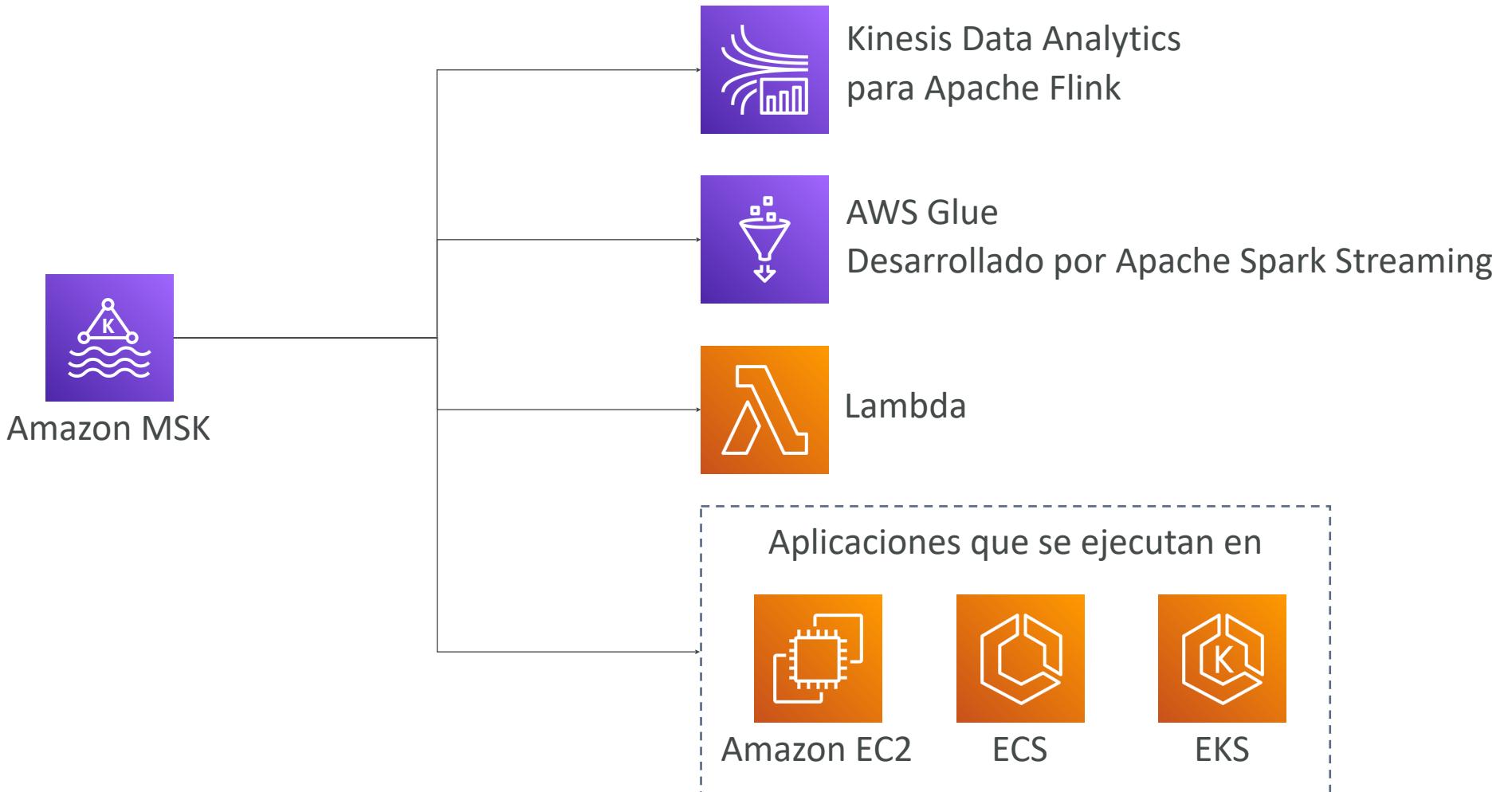
- Límite de tamaño de los mensajes: 1 MB
- Flujos de datos con shards
- División y fusión de fragmentos
- Cifrado TLS en vuelo
- Cifrado KMS en reposo



Amazon MSK

- 1MB por defecto, configurar para mayor (ej: 10MB)
- Temas Kafka con particiones
- Sólo se pueden añadir particiones a un tema
- Cifrado en vuelo PLAINTEXT o TLS
- Cifrado KMS en reposo

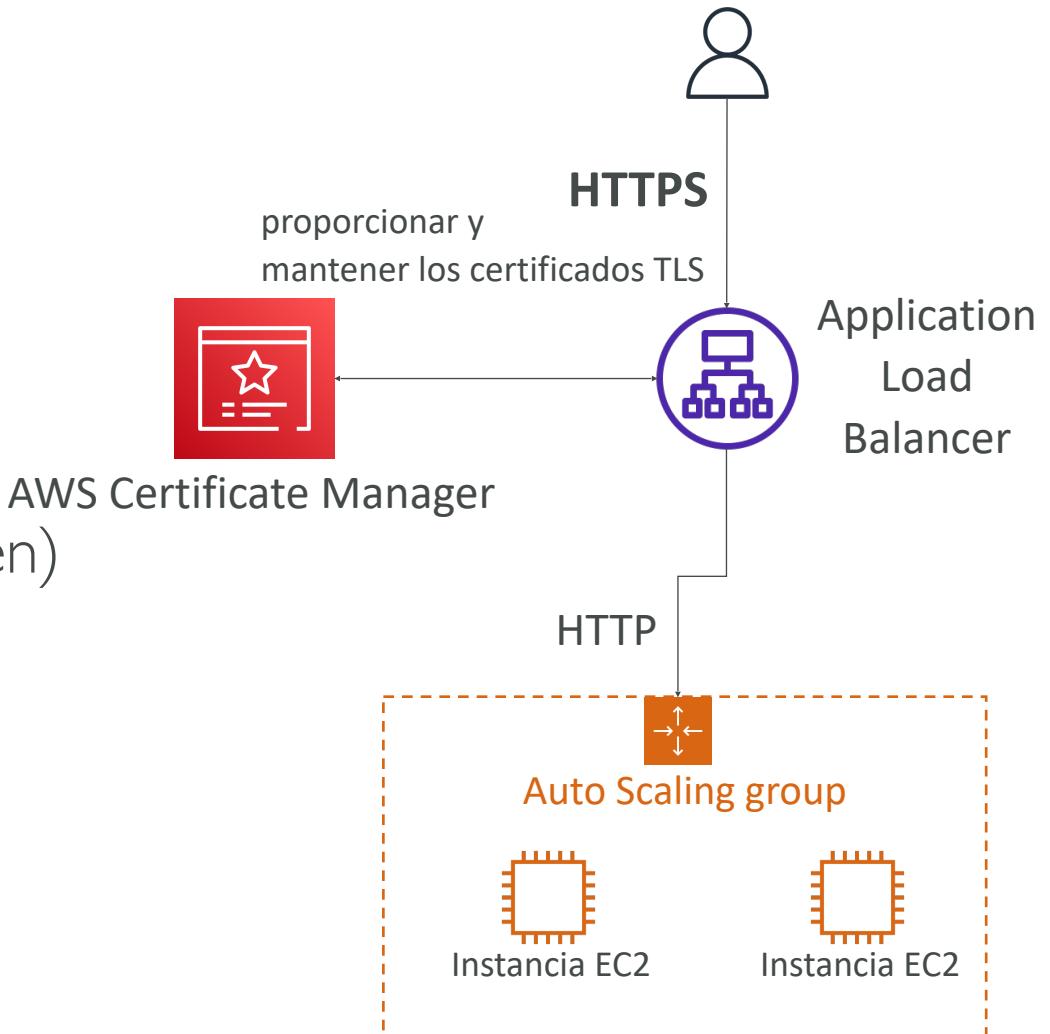
Consumidores Amazon MSK



AWS Certificate Manager (ACM)



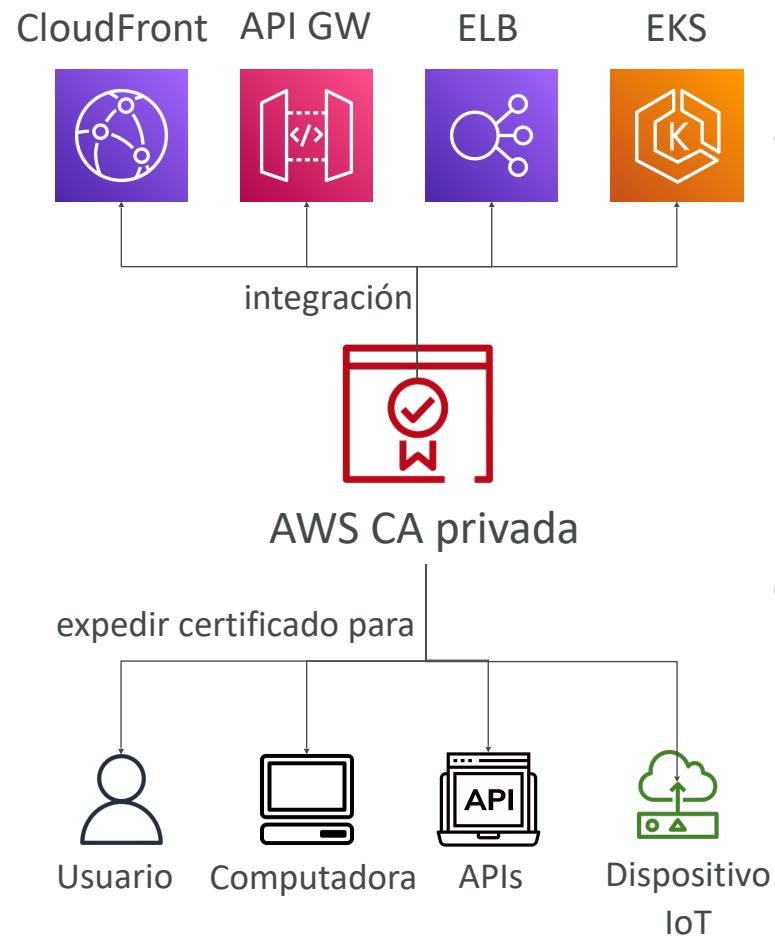
- Te permite aprovisionar, gestionar y desplegar fácilmente los **certificados SSL/TLS**
- Se utilizan para proporcionar encriptación en vuelo para los sitios web (HTTPS)
- Admite certificados TLS públicos y privados
- Gratuito para los certificados TLS públicos
- Renovación automática de certificados TLS
- Integraciones con (carga de certificados TLS en)
 - Elastic Load Balancers
 - Distribuciones de CloudFront
 - APIs en API Gateway



Autoridad de certificación (CA) privada de AWS



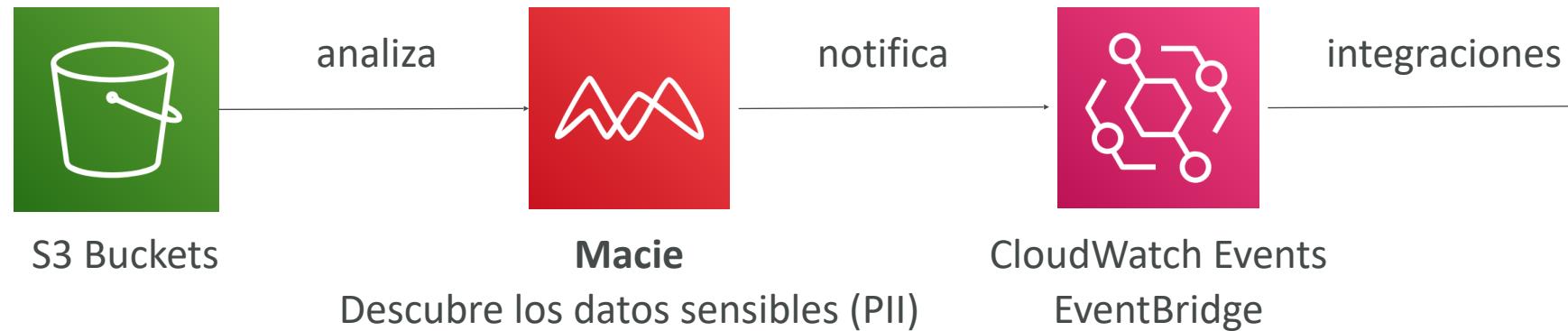
- El servicio gestionado te permite crear Autoridades de Certificación (CA) privadas, incluidas CA raíz y subordinadas
- Puede emitir y desplegar certificados X.509 de entidad final
- En los certificados sólo confía tu Organización (no la Internet pública)
- Funciona para los servicios de AWS integrados con ACM
- Casos de uso:
 - Comunicación TLS cifrada, Código de firma criptográfica
 - Autenticar usuarios, ordenadores, endpoints API y dispositivos IoT
 - Clientes empresariales que crean una Infraestructura de Clave Pública (PKI)



Amazon Macie



- Amazon Macie es un servicio de seguridad y privacidad de datos totalmente gestionado que utiliza el **machine learning y la concordancia de patrones para descubrir y proteger tus datos sensibles en AWS**.
- Macie te ayuda a identificar y alertar sobre los **datos sensibles, como la información personal identifiable (PII)**



AWS AppConfig

- Configura, valida y despliega configuraciones dinámicas
- Despliega cambios de configuración dinámica en tus aplicaciones independientemente de cualquier despliegue de código
- No necesitas reiniciar la aplicación
- Banderas de características, ajuste de aplicaciones, lista de permitidos/bloqueados...
- Utilízalo con aplicaciones en instancias EC2, Lambda, ECS, EKS...
- Despliega gradualmente los cambios de configuración y revierte si surgen problemas
- Valida los cambios de configuración antes del despliegue utilizando
 - **Esquema JSON** (comprobación sintáctica) o
 - **Función Lambda** - ejecuta código para realizar la validación (comprobación semántica)



Repaso del examen y consejos

Punto de control del estado de aprendizaje

- Veamos hasta dónde hemos llegado en nuestro viaje de aprendizaje
- <https://aws.amazon.com/certification/certified-developer-associate/>

La práctica hace al maestro

- Si eres nuevo en AWS, practica un poco gracias a este curso antes de lanzarte al examen
 - El examen te recomienda tener uno o más años de experiencia práctica en el desarrollo y mantenimiento de aplicaciones basadas en AWS
 - ¡La práctica hace al maestro!
-
- Si te sientes abrumado por la cantidad de conocimientos que acabas de aprender, repásalos una vez más

¡Ideas para practicar...!

- Toma una de tus aplicaciones existentes
- Intenta desplegarla manualmente en EC2
- Prueba a desplegarla en Elastic Beanstalk y haz que escale
- Prueba a crear un pipeline CI/CD para ella
- Prueba a desacoplar componentes utilizando SQS / SNS
- Si es posible, intenta ejecutarlo en AWS Lambda & friends
- Escribe scripts de automatización utilizando la CLI / SDK
 - Idea 1: Apaga las instancias EC2 por la noche / inícialas por la mañana
 - Idea 2: Automatizar Snapshots de volúmenes EBS por la noche
 - Idea 3: Listar todas las instancias EC2 infroutilizadas (Utilización CPU < 10%)

Proceder por eliminación

- La mayoría de las preguntas van a estar basadas en situaciones hipotéticas
 - Para todas las preguntas, descarta las respuestas que sepas con certeza que son incorrectas
 - Para las respuestas restantes, entiende cuál tiene más sentido
-
- Hay muy pocas preguntas trampa
 - No le des demasiadas vueltas
 - Si una solución parece factible pero muy complicada, probablemente sea errónea

Hojea los Whitepapers de AWS

- Puedes leer algunos White Papers de AWS aquí:
 - Mejores prácticas de seguridad de AWS
 - AWS Well-Architected Framework (en inglés)
 - Arquitectura para el Cloud y mejores prácticas de AWS
 - Practicar la integración continua y la entrega continua en AWS Acelerar la entrega de software con DevOps
 - Microservicios en AWS
 - Arquitecturas sin servidor con AWS Lambda
 - Optimización de la economía empresarial con arquitecturas sin servidor
 - Ejecución de microservicios en contenedores en AWS
 - Implementaciones Blue/Green en AWS
- En general hemos explorado todos los conceptos más importantes del curso
- ¡Nunca está de más echar un vistazo a los Whitepapers que te parezcan interesantes!

Lee las FAQ de cada servicio

- FAQ = Preguntas más frecuentes
- Ejemplo: <https://aws.amazon.com/lambda/faqs/>
- Las FAQ cubren muchas de las preguntas que se hacen en el examen
- Ayudan a confirmar tu comprensión de un servicio

Entra en la Comunidad AWS

- Ayudar y debatir con otras personas en las preguntas y respuestas del curso
- Repasa las preguntas formuladas por otras personas en las Preguntas y Respuestas
- Haz el test práctico de esta sección

- Lee foros en línea
- Lee blogs online
- Asiste a reuniones locales y debate con otros ingenieros de AWS

¿Cómo funcionará el examen?

- Tendrás que inscribirte en línea en <https://www.aws.training/>
- La tasa del examen es de 150 USD
- Proporciona un documento de identidad (DNI, Pasaporte, los detalles están en los correos electrónicos que te enviamos...)
- No se permiten notas, ni bolígrafo, ni hablar
- Se harán 65 preguntas en 130 minutos
- Utiliza la función "Marcar" para marcar las preguntas que quieras volver a revisar
- Al final puedes revisar opcionalmente todas las preguntas / respuestas

- Para aprobar necesitas una puntuación mínima de 720 sobre 1000
- Sabrás en un plazo de 5 días si has aprobado / suspendido los exámenes (la mayoría de las veces menos)
- Conocerás la puntuación global unos días después (notificación por correo electrónico)
- No sabrás qué respuestas eran correctas / incorrectas
- Si suspendes, puedes volver a hacer el examen 14 días después

¡Enhорabuena y próximos pasos!

¡Enhorabuena!

- ¡Enhorabuena por haber terminado el curso!
- Espero que apruebes el examen sin problemas ☺.
- Si apruebas, estaré más que feliz de saber que te he ayudado
 - Publícalo en Preguntas y Respuestas para ayudar y motivar a otros estudiantes. ¡Comparte tus consejos!
 - ¡Publícalo en LinkedIn y etiquétame!
- En general, espero que hayas aprendido a utilizar AWS y que seas un desarrollador AWS tremadamente bueno

Próximos pasos

- Sigue con el programa de certificaciones de AWS
 - FOUNDATIONAL
 - ASSOCIATE
 - PROFESIONAL
 - SPECIALTY
- Te recomiendo que te certifiques en alguna especialidad en el campo que deseas

FOUNDATIONAL

Certificación basada en conocimientos para obtener conocimiento básico de la nube de AWS.

No se necesita experiencia previa.



ASSOCIATE

Certificaciones basadas en roles que demuestran su conocimiento y habilidades de AWS y que construyen su credibilidad como profesional de la nube de AWS.

Se recomienda tener experiencia previa sólida en TI local o en la nube.



PROFESIONAL

Certificaciones basadas en roles que validan habilidades y conocimientos avanzados necesarios para diseñar aplicaciones seguras, optimizadas y modernas, y automatizar procesos en AWS. **Se recomienda tener 2 años de experiencia previa en la nube de AWS**



SPECIALTY

Aprenda a profundidad y posíóngase como un asesor de confianza para las partes interesadas o clientes de estas áreas estratégicas. **Consulte las guías de examen en las páginas de exámenes para saber la experiencia recomendada.**

