



AWS DEVELOPER

QUIEN SOY

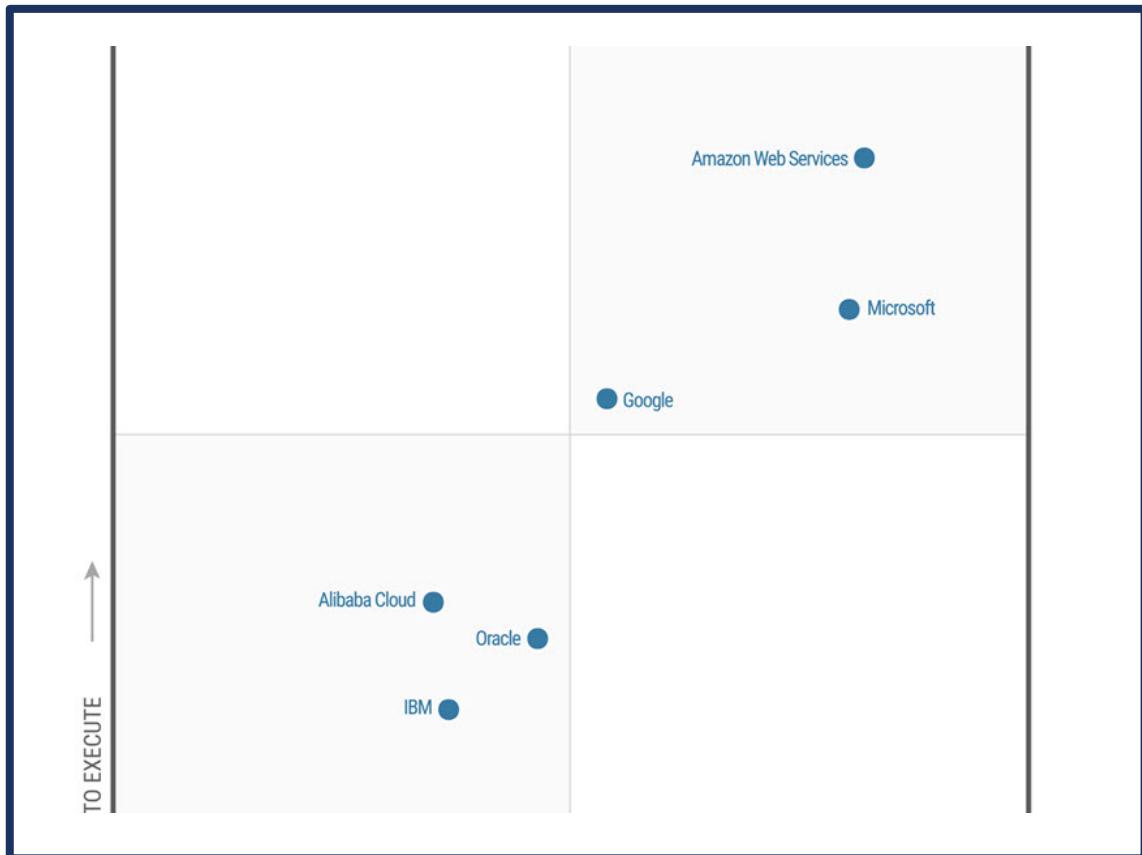


Martinez Miguez Luisa
AWS Solution Architect at Linke IT



Contact Info
Your Profile : linkedin.com/in/martinez-miguez-luisa-1951369
Email : luisa.martinezm@gmail.com

INTRODUCION



- AWS lleva 9 años consecutivos liderando Gartner's Magic Quadrant for Cloud Infrastructure as a Service (IaaS)
- Esto define a AWS como el líder en madurez en la cloud pública

Certificaciones disponibles de AWS Certifications

Professional

Dos años de amplia experiencia en el diseño, la operación y la solución de problemas con la nube de AWS



Associate

Un año de experiencia solucionando problemas e implementando soluciones con la nube de AWS



Arquitecto

Operaciones

Desarrollador

Foundational

Seis meses de conocimiento básico sobre la nube de AWS y el sector

Profesional de la
nube



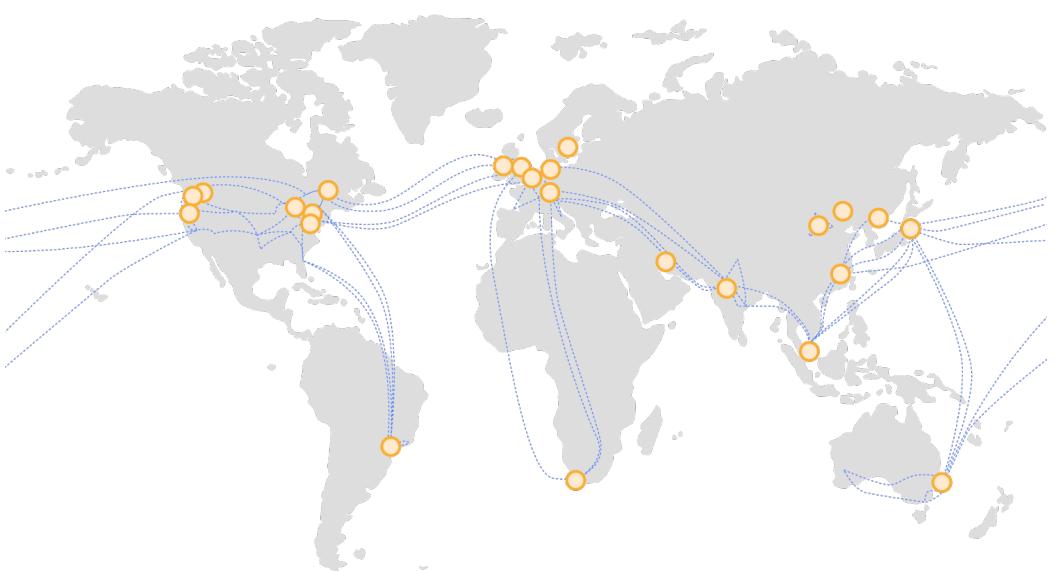
Specialty

Experiencia técnica en la nube de AWS de nivel Specialty según lo especificado en la **guía del examen**



<https://aws.amazon.com/es/certification/>

DISEÑO DE RED



Red global

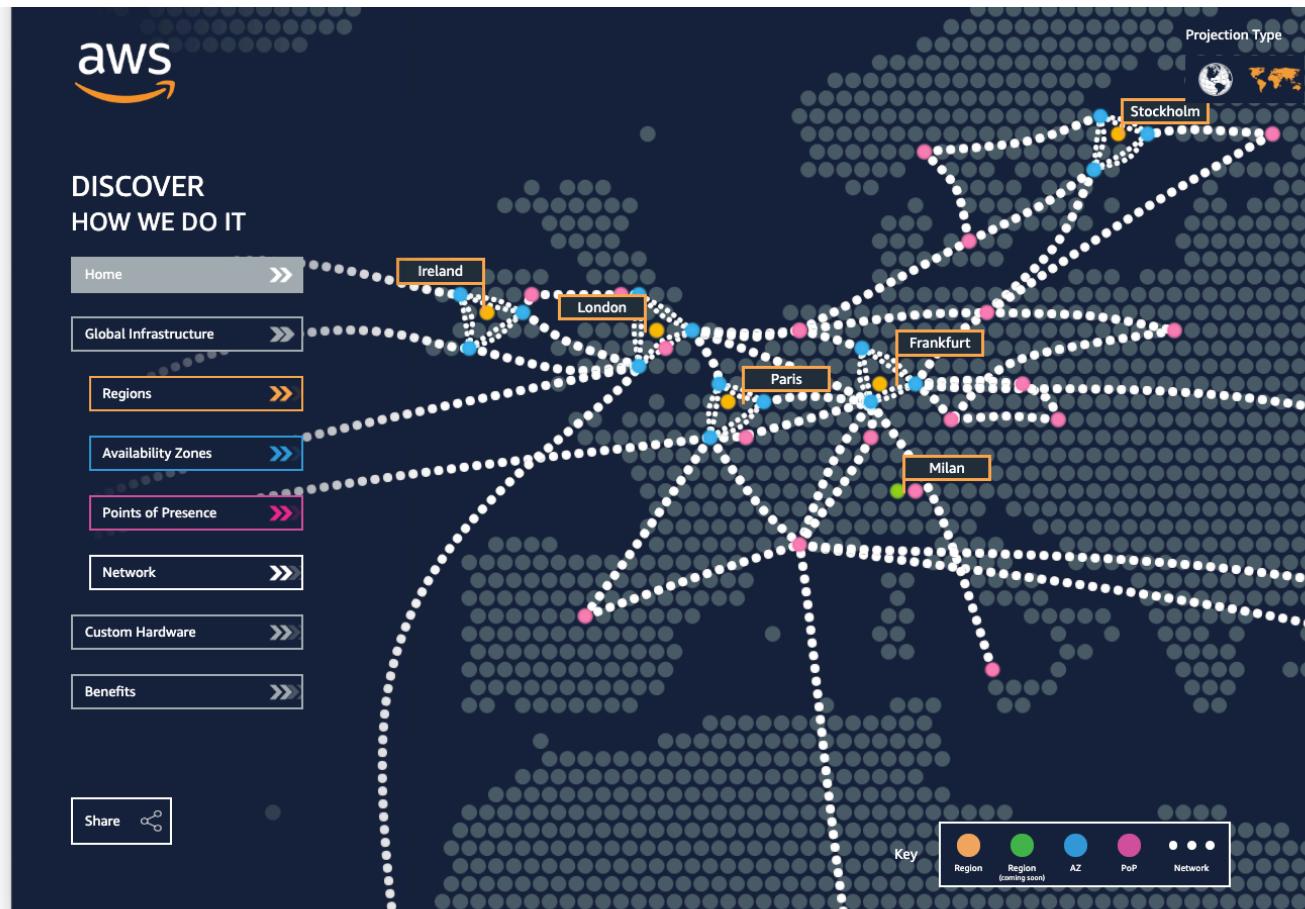
■ AWS tiene el espacio de infraestructura global más grande de todos los proveedores de la nube, y este espacio se está expandiendo continuamente para ayudar a los clientes a ofrecer mejores experiencias a los usuarios finales, expandir rápidamente las operaciones a prácticamente cualquier región o país y cumplir con los requisitos de soberanía y localidad de datos. Si una empresa quiere prestar servicios a sus clientes en Europa, el cliente debería poder elegir entre una amplia opción de regiones o centros de datos en Europa, como París, Londres, Fráncfort o Dublín. La nube de AWS incluye 69 zonas de disponibilidad en 22 regiones geográficas de todo el mundo. Además, se anunciaron planes para incorporar 9 zonas de disponibilidad y tres regiones adicionales en Ciudad del Cabo, Yakarta y Milán.

Infraestructura global

■ Todos los centros de datos, AZ y la región de AWS están interconectados a través de una infraestructura de red global privada especialmente diseñada, altamente disponible y de baja latencia. La red se basa en una red de fibra metropolitana de 100 GbE paralela global y totalmente redundante que se conecta mediante cables transoceánicos a través de los océanos Atlántico, Pacífico e Índico, así como del Mar Mediterráneo, el Mar Rojo y los mares del sur de China.

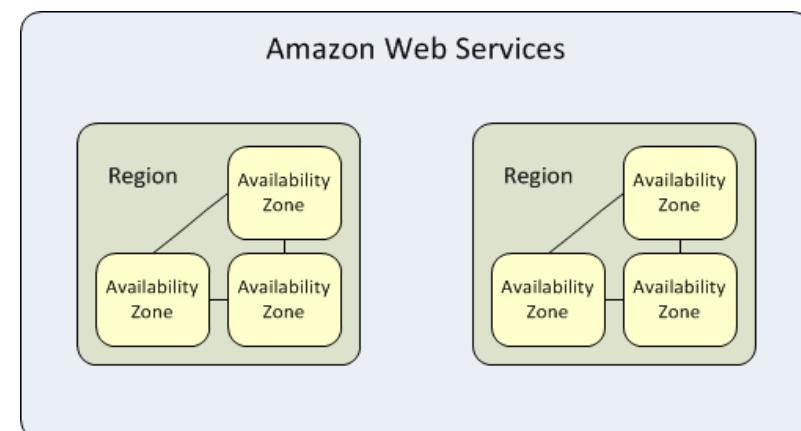
<https://www.infrastructure.aws/>

INTRODUCCIÓN RED

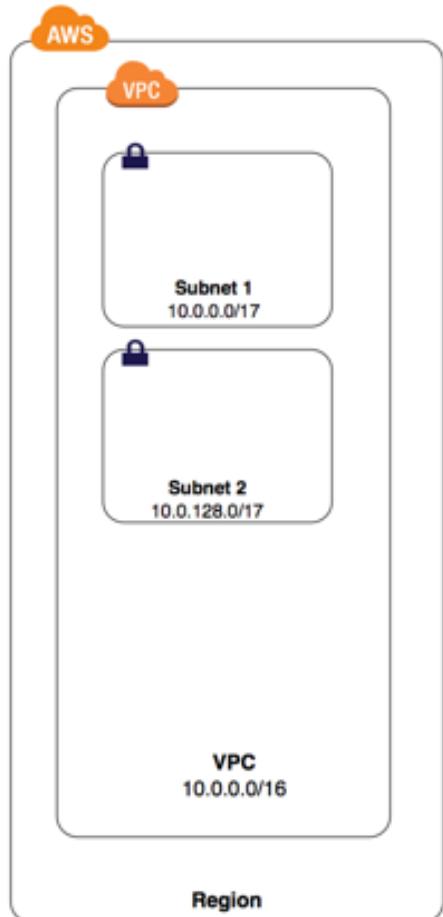


Conceptos de región y zona de disponibilidad

Cada región es totalmente independiente. Cada zona de disponibilidad está aislada, pero las zonas de disponibilidad de una región están conectadas a través de conexiones de baja latencia. En el siguiente diagrama se ilustra la relación entre las regiones y las zonas de disponibilidad.



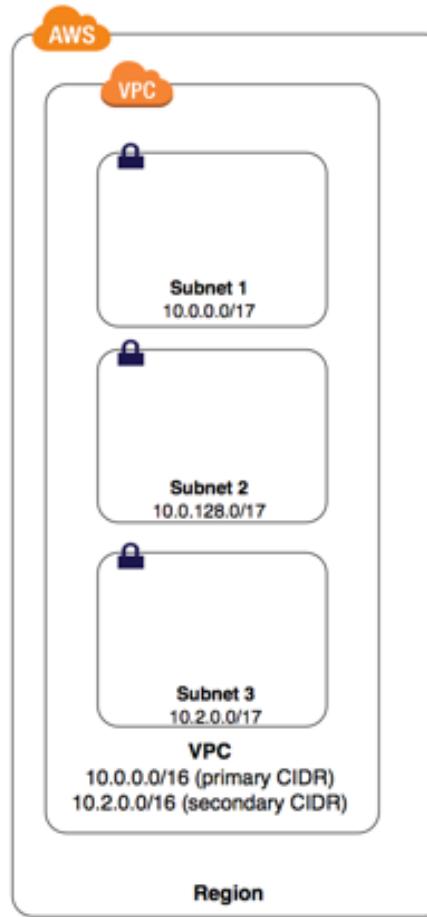
VPC with 1 CIDR block



Main route table

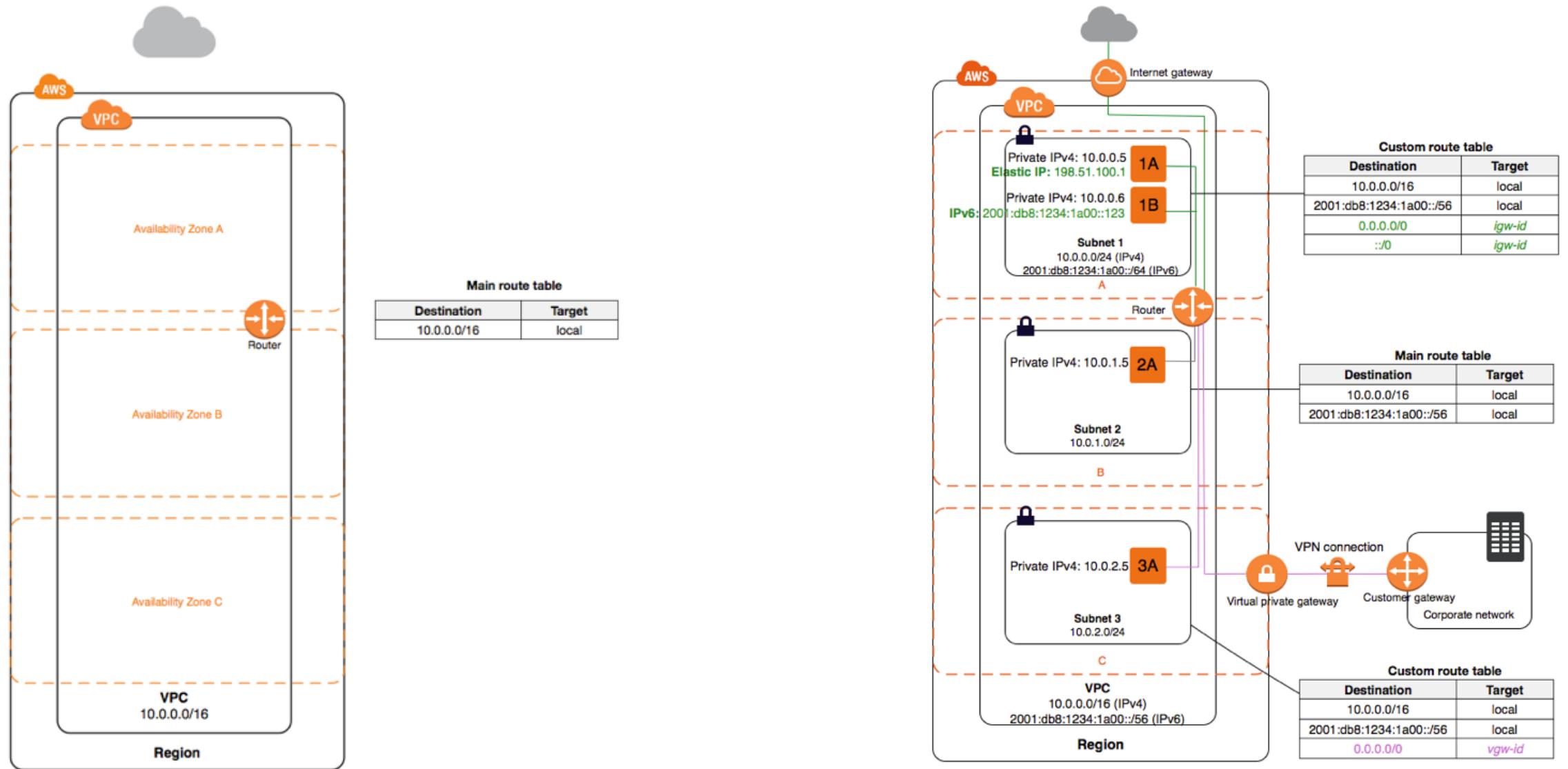
Destination	Target
10.0.0.0/16	local

VPC with 2 CIDR blocks



Main route table

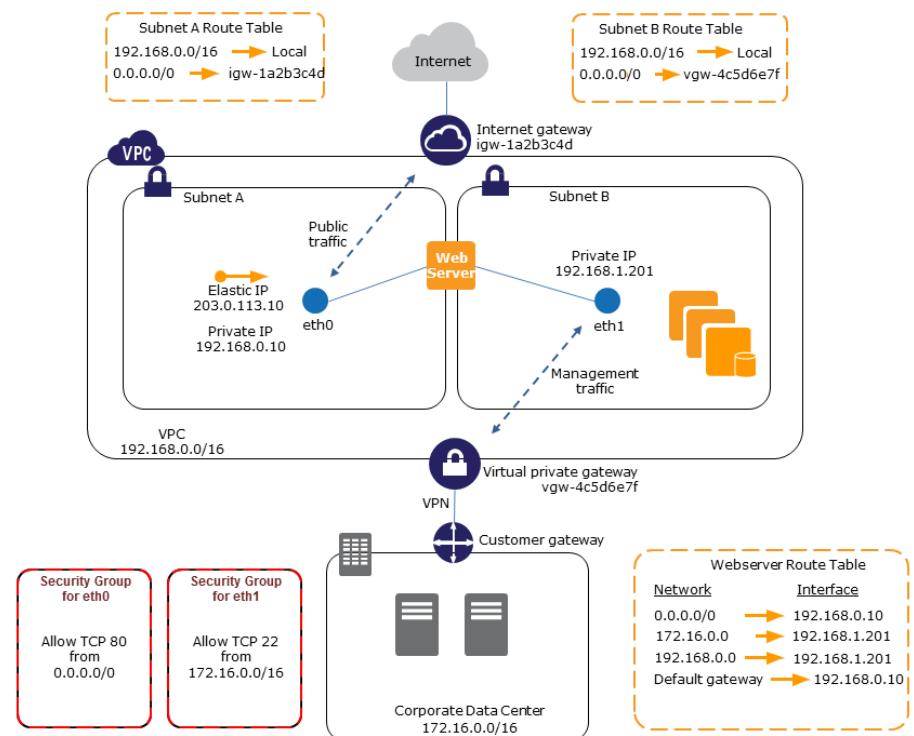
Destination	Target
10.0.0.0/16	local
10.2.0.0/16	local



COMPONENTES DE RED DE UNA VPC

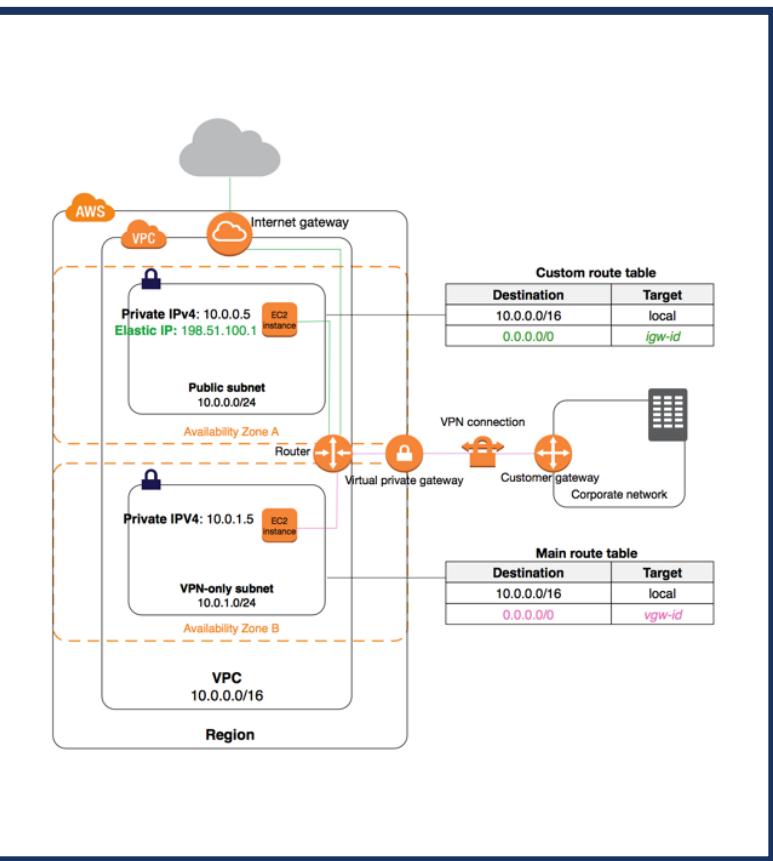
- [Interfaces de red](#)
- [Tablas de ruteo](#)
- [Gateways de Internet](#)
- [Gateways de Internet de solo salida](#)
- [Conjuntos de opciones de DHCP](#)
- [DNS](#)
- [Direcciones IP elásticas](#)
- [Puntos de conexión de la VPC](#)
- [NAT](#)
- [Interconexión de VPC](#)
- [ClassicLink](#)
- [Network Interfaces](#)
- [Route Tables](#)
- [Internet Gateways](#)
- [Egress-Only Internet Gateways](#)
- [DHCP Options Sets](#)
- [DNS](#)
- [Elastic IP Addresses](#)
- [VPC Endpoints](#)
- [NAT](#)
- [VPC Peering](#)
- [ClassicLink](#)

INTERFACES DE REDES ELÁSTICAS

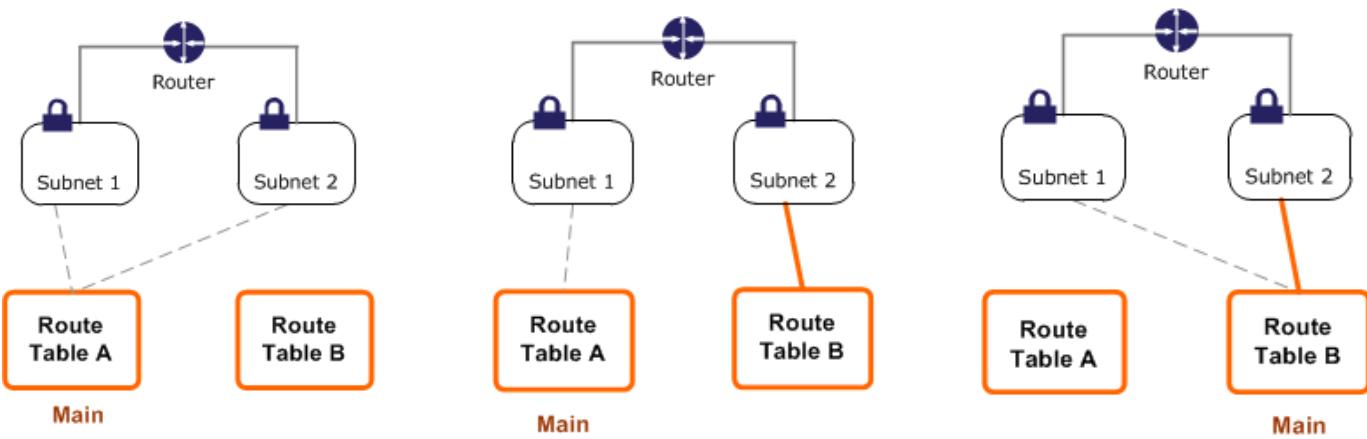


- Una interfaz de red elástica (es un componente de red lógico en una VPC que representa una tarjeta de red virtual).
 - Una dirección IPv4 privada principal del intervalo de direcciones IPv4 de la VPC
 - Una o más direcciones IPv4 privada secundaria del intervalo de direcciones IPv4 de la VPC
 - Una dirección IP elástica (IPv4) por dirección IPv4 privada
 - Una dirección IPv4 pública
 - Una o varias direcciones IPv6
 - Uno o varios grupos de seguridad
 - Una dirección MAC
 - Una marca de comprobación de origen/destino
 - Una descripción

ROUTING TABLE



- Contienen conjuntos de reglas, denominadas rutas, que se usan para determinar adónde se dirige el tráfico de red.
- Se asocian a nivel de subred.
 - 1 subred asociado solo a 1 router
 - 1 router asociado a N subredes
 - En una VPC pude haber X routers
 - Si una subred no se asocia, esta se asocia al Main route
 - Una subred se puede asociar a otro router cuando se quiera.



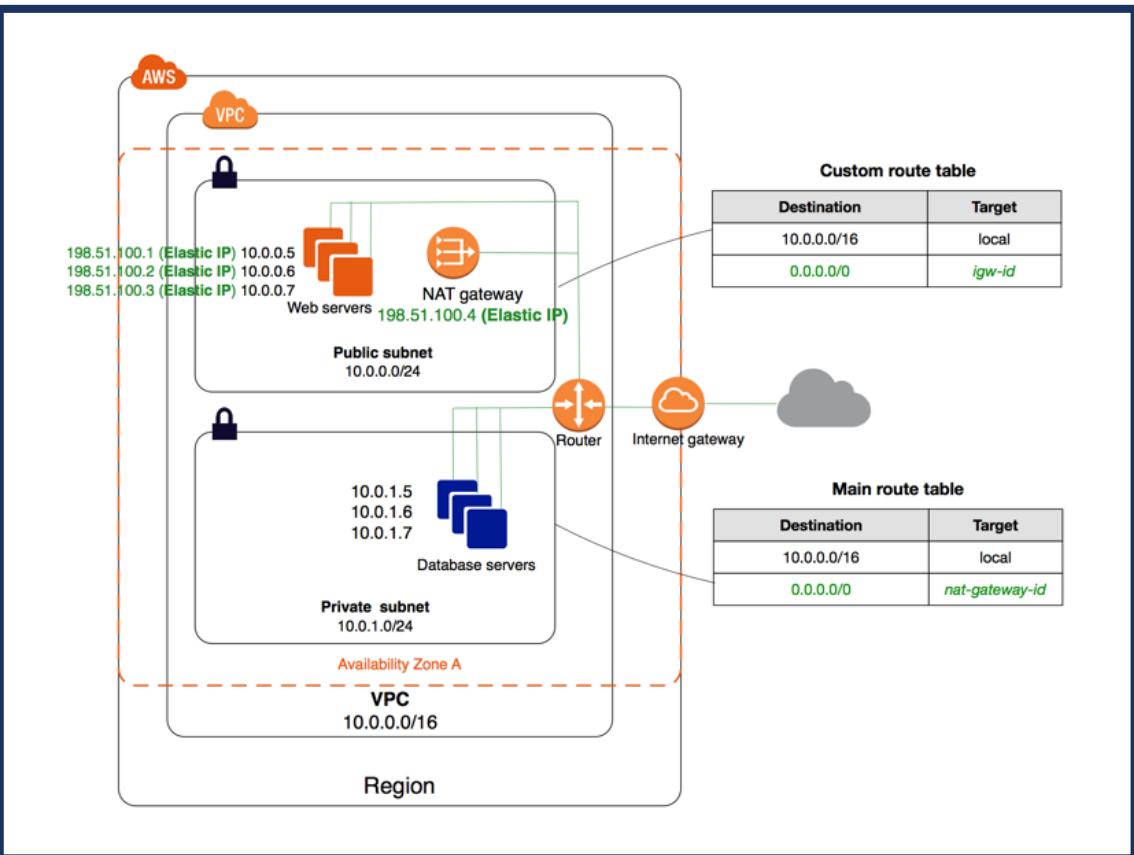
ROUTING TABLE(PRIORIDAD DE LA RUTA)

Destino	Objetivo
10.0.0.0/16	Local
172.31.0.0/16	pcx-1a2b3c4d
0.0.0.0/0	igw-11aa22bb

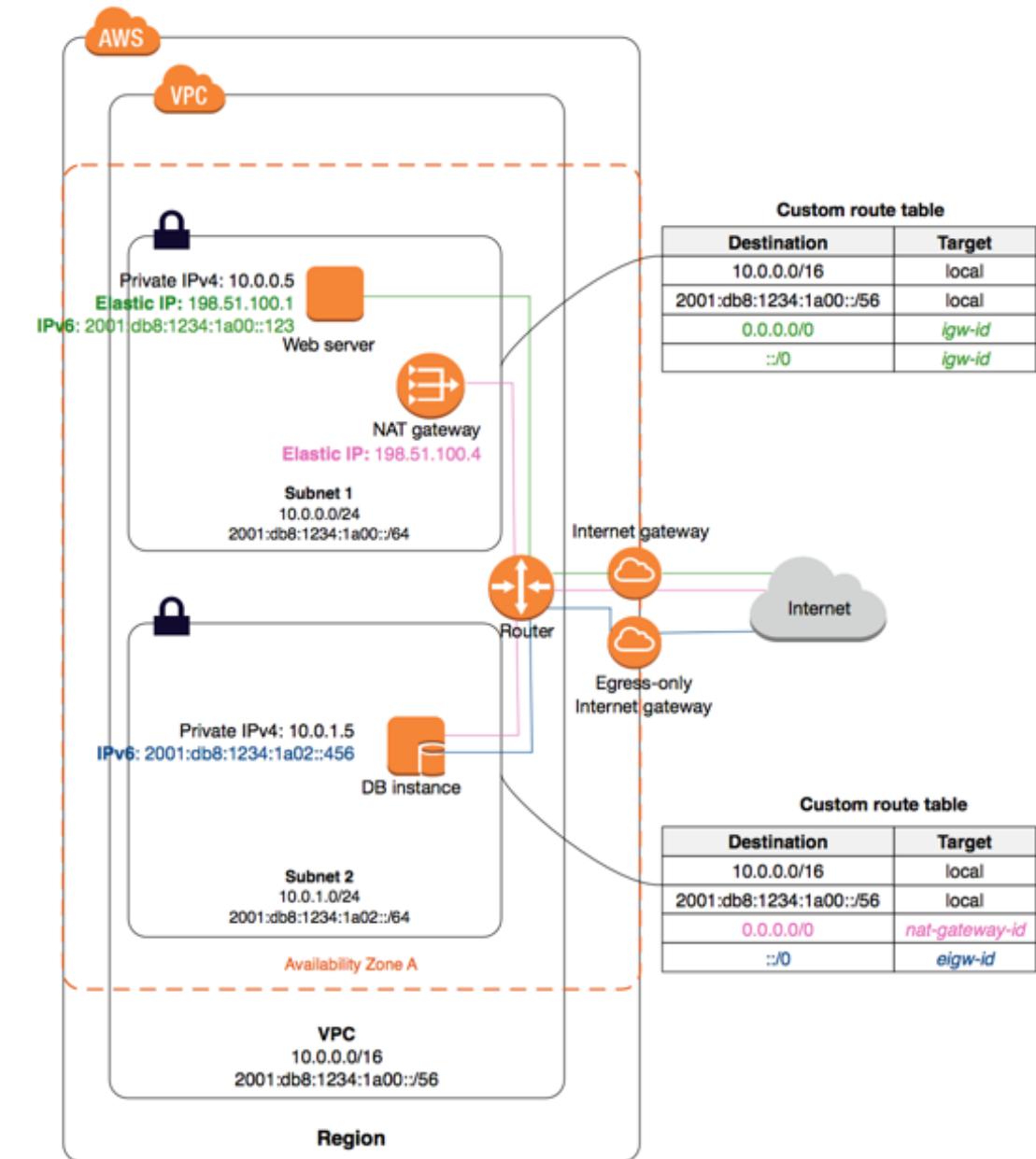
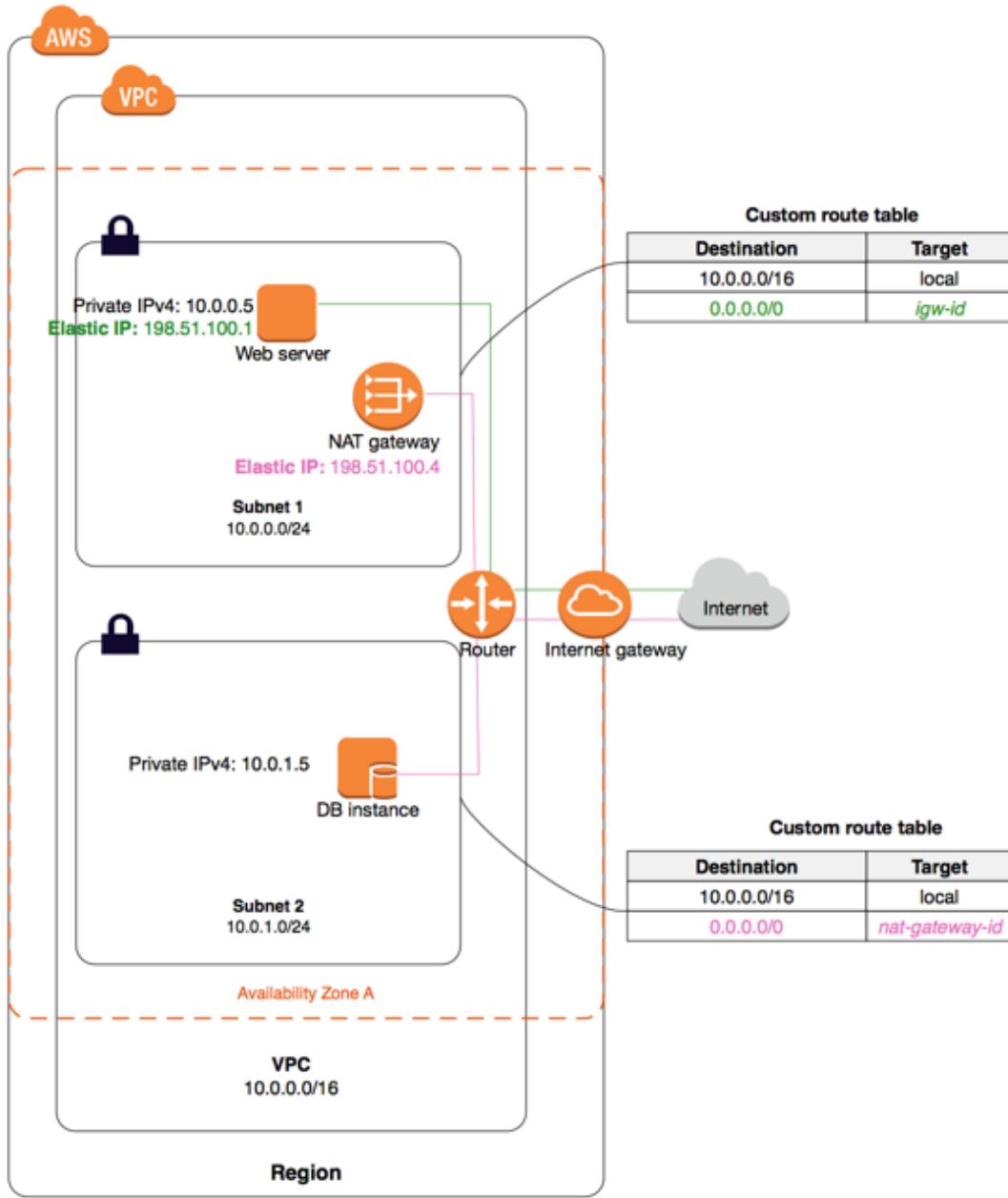
Destino	Objetivo
10.0.0.0/16	Local
172.31.0.0/24	vgw-1a2b3c4d (propagado)
172.31.0.0/24	igw-11aa22bb

- Para determinar cómo dirigir tráfico, se utiliza la ruta más específica de su tabla de ruteo que coincide con el tráfico en cuestión (coincidencia del prefijo más largo). (máscara con valor más alto).
- Las rutas fijas tienen preferencia sobre las propagadas
- Las rutas de Direc connect tienen preferencia sobre las de la VPN
- En este ejemplo, el router tiene una ruta estática a una gateway de Internet (añadida manualmente) y una ruta propagada a una gateway privada virtual. Ambas rutas tienen el destino 172.31.0.0/24. En este caso, todo el tráfico con destino 172.31.0.0/24 se dirige a la gateway de Internet, ya que se trata de una ruta estática con prioridad sobre la ruta propagada.

NAT

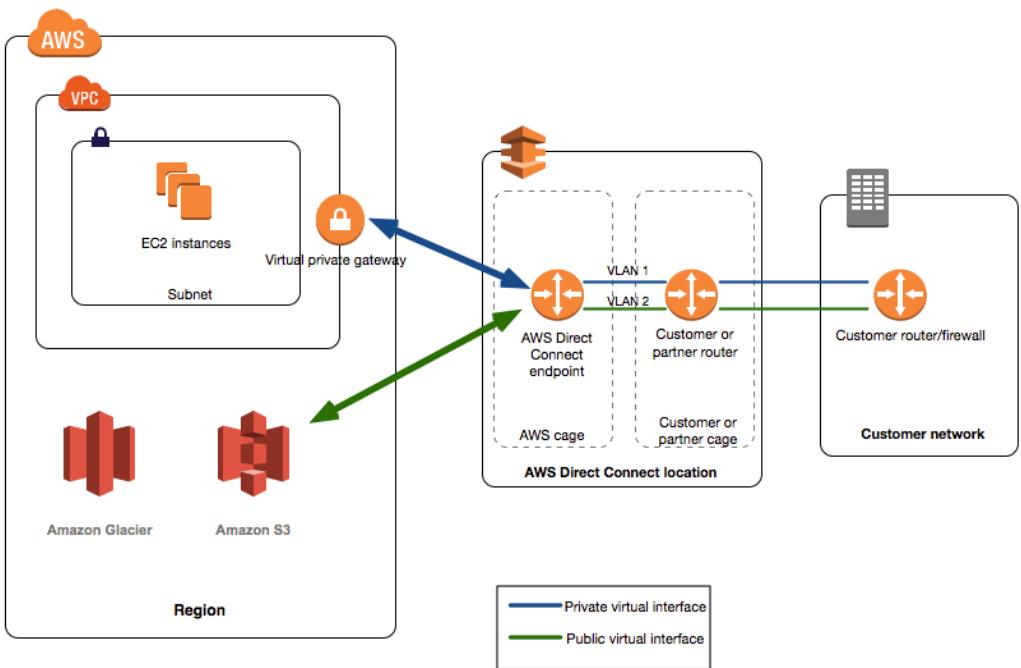


- Permite a las instancias de la subred privada conectarse a Internet o a otros servicios de AWS a la vez que se impide a Internet iniciar una conexión a esas mismas instancias.
- En IPv6 esta función la hace Egress-Only gateway



Atributo	gateway NAT	Instancia NAT
Disponibilidad	Altamente disponibles. Las gateways NAT de cada zona de disponibilidad se implementan con redundancia. Cree una gateway NAT en cada zona de disponibilidad para garantizar una arquitectura independiente de zonas.	Utilice un script para administrar la comutación por error entre instancias.
Ancho de banda	Se puede escalar hasta 45 Gbps.	Dependen del ancho de banda del tipo de instancia.
Mantenimiento	Administrado por AWS. No necesita realizar ningún mantenimiento.	Administradas por usted. Por ejemplo, al instalar actualizaciones de software o parches de sistema operativo en la instancia.
Desempeño	El software está optimizado para la gestión del tráfico de NAT.	Una AMI de Amazon Linux configurada para realizar la NAT.
Costo	Se cobra en función del número de gateways NAT que utilice, la duración del uso y la cantidad de datos que envíe mediante las gateways NAT.	Se cobra en función del número de instancias NAT que utilice, la duración del uso y el tamaño y el tipo de instancia.
Tipo y tamaño	Oferta uniforme; no necesita decidir el tamaño ni el tipo.	Elija un tipo de instancia y un tamaño adecuados, acordes con la estimación de su carga de trabajo.
Direcciones IP públicas	Elija la dirección IP elástica para asociar a la gateway NAT en el momento de la creación.	Utilice una dirección IP elástica o una dirección IP pública con una instancia NAT. Puede cambiar la dirección IP pública en el momento de asociar una nueva dirección IP elástica a la instancia.
Direcciones IP privadas	Se seleccionan automáticamente del rango de direcciones IP de la subred al crear la gateway.	Al lanzar la instancia, asigne una dirección IP privada específica del rango de direcciones IP de la subred.
Grupos de seguridad	No se pueden asociar a una gateway NAT. Puede asociar grupos de seguridad a sus recursos detrás de la gateway NAT para controlar el tráfico entrante y saliente.	Asocie su instancia NAT y los recursos detrás de su instancia NAT para controlar el tráfico entrante y saliente.
ACL de red	Utilice una ACL de red para controlar el tráfico hacia la subred y procedente de esta en la que se encuentra su gateway NAT.	Utilice una ACL de red para controlar el tráfico hacia la subred y procedente de esta en la que se encuentra su instancia NAT.
Logs de flujo	Utilice los logs de flujo para capturar el tráfico.	Utilice los logs de flujo para capturar el tráfico.
Enrutamiento de puertos	No es compatible.	Personalice manualmente la configuración para que admita el reenvío de puertos.
Servidores bastión	No es compatible.	Se pueden utilizar como servidor bastión.
Métricas de tráfico	Consulte las métricas de CloudWatch para la gateway NAT .	Consulte las métricas de CloudWatch para la instancia.
Comportamiento de los tiempos de espera	Cuando el tiempo de espera de una conexión finaliza, una gateway NAT devuelve un paquete RST a los recursos situados detrás de la gateway NAT que intenten continuar la conexión (no envía un paquete FIN).	Cuando el tiempo de espera de una conexión finaliza, una instancia NAT envía un paquete FIN a los recursos situados detrás de la instancia NAT para cerrar la conexión.
Fragmentación de IP	Admiten el reenvío de paquetes IP fragmentados para el protocolo UDP. No admiten la fragmentación para los protocolos ICMP y TCP. Los paquetes fragmentados para estos protocolos se retirarán.	Admiten el reensamblado de paquetes IP fragmentados para los protocolos ICMP, UDP y TCP.

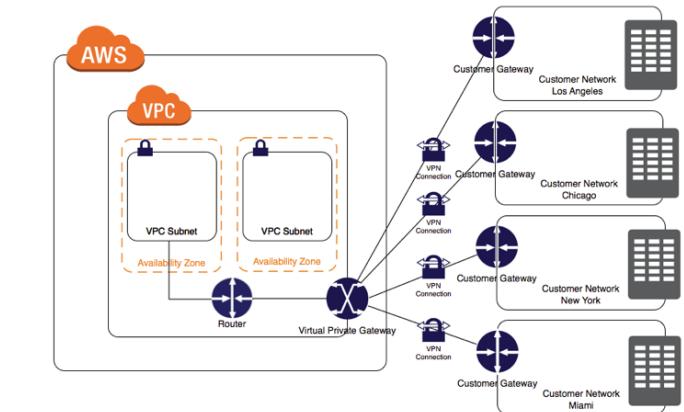
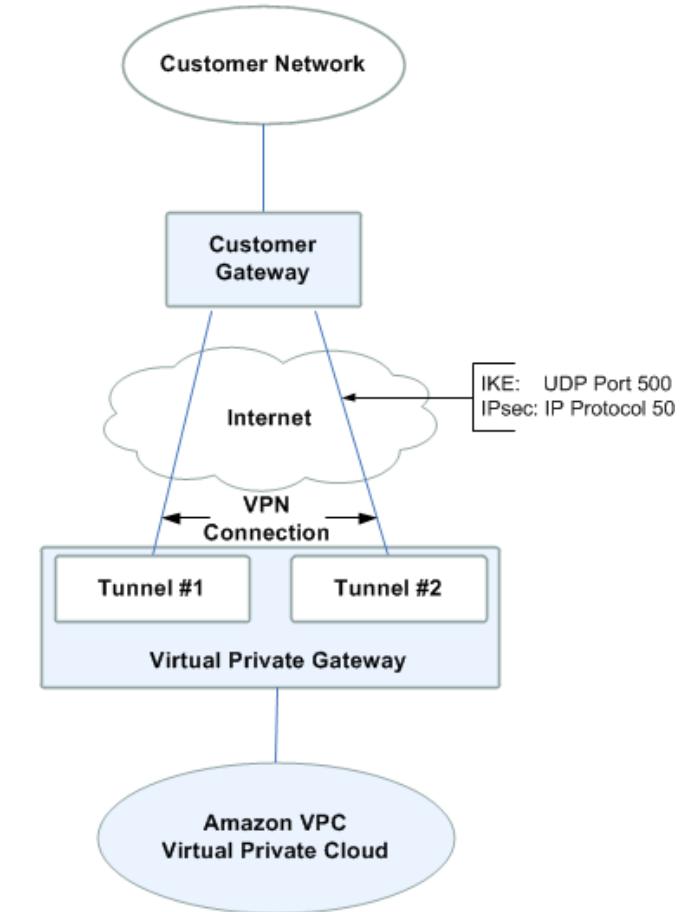
DIRECT CONNECT



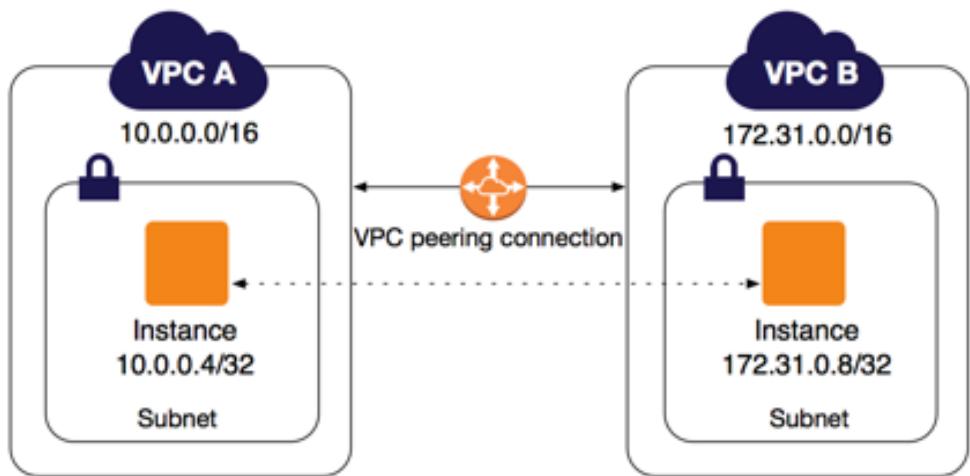
- Direct Connect vincula su red on prem con una ubicación de AWS Direct Connect a través de una fibra óptica. Un extremo dse conecta a su router y el otro al router de AWS Direct Connect.
- Sobre esta conexión, puede crear *interfaces virtuales*
- **públicos** a servicios en AWS con Ip publica(por ejemplo, en Amazon S3)
- **Privados** a Amazon VPC

VPN

- AWS Site-to-Site VPN extends your data center or branch office to the cloud. It uses cryptography to protect communications over Internet Protocol (IP) networks and supports connecting to both a virtual private gateway and an AWS Transit Gateway.

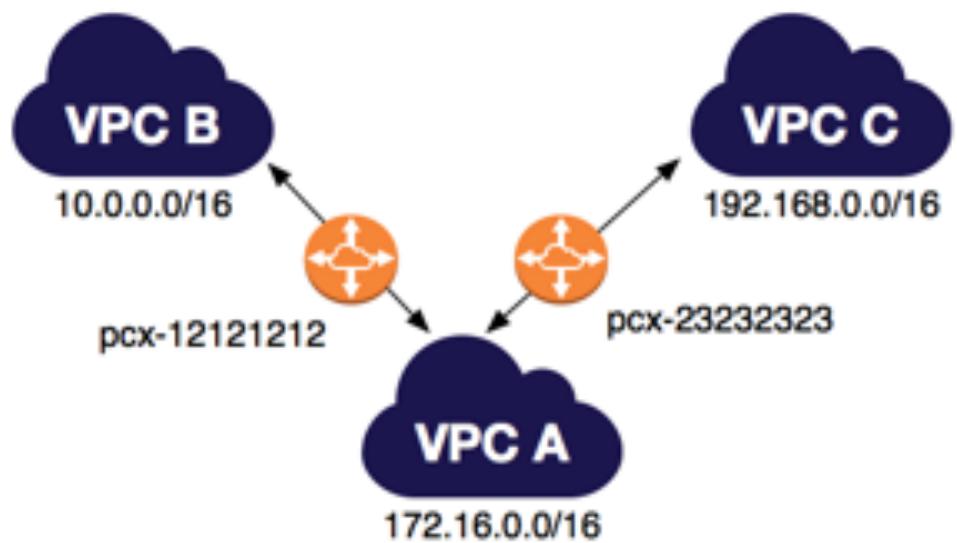


VPC PEERING



- Una interconexión de VPC es una conexión de redes entre dos VPC que permite direccionar tráfico entre ellas mediante direcciones IPv6 o direcciones IPv4 privadas. Las instancias de ambas VPC se pueden comunicar entre sí siempre que se encuentren en la misma red. Puede crear una interconexión de VPC entre sus propias VPC o con una VPC de otra cuenta de AWS. Las VPC pueden encontrarse en regiones distintas (lo que se conoce como interconexión de VPC entre regiones).

VPC PEERING



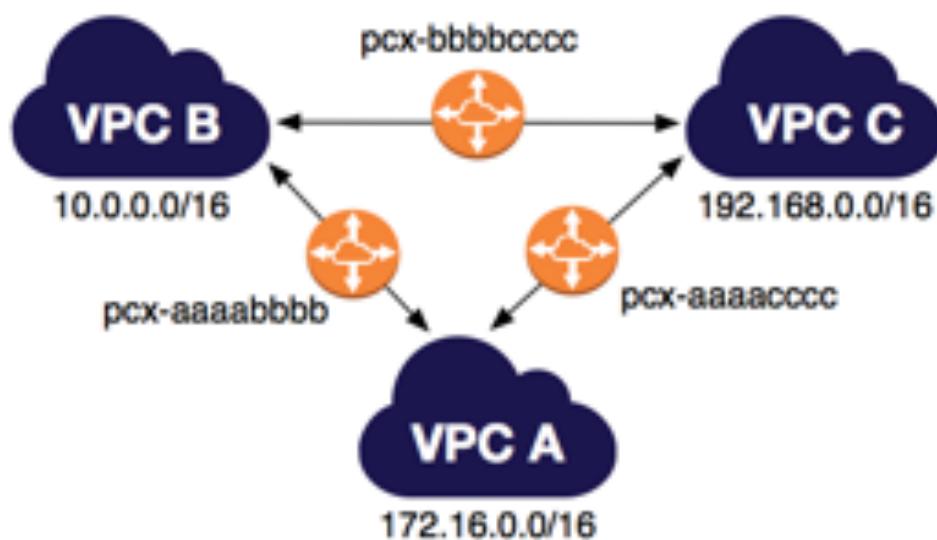
■ Supongamos que tiene una VPC central (VPC A) y una interconexión de VPC entre la VPC A y la VPC B (pcx-12121212), así como entre la VPC A y la VPC C (pcx-23232323). Las VPC se encuentran en la misma cuenta de AWS y no tienen bloques de CIDR solapados.

■ Puede utilizar esta configuración "triangular" cuando tiene recursos en una VPC central como, por ejemplo un repositorio de servicios, que necesita que estén disponibles para las otras VPC. Las otras VPC no necesitan tener acceso a los recursos de cada una; solo necesitan tener acceso a los recursos de la VPC central.

■ **Nota:** La VPC B y la VPC C no se pueden enviar tráfico directamente entre sí a través de la VPC A.

Tabla de enrutamiento	Destino	Objetivo
VPC A	172.16.0.0/16	Local
	10.0.0.0/16	pcx-12121212
	192.168.0.0/16	pcx-23232323
VPC B	10.0.0.0/16	Local
	172.16.0.0/16	pcx-12121212
VPC C	192.168.0.0/16	Local
	172.16.0.0/16	pcx-23232323

VPC PEERING



Tablas de ruteo	Destino	Objetivo
VPC A	172.16.0.0/16	Local
	10.0.0.0/16	pcx-aaaabbbb
	192.168.0.0/16	pcx-aaaaacccc
VPC B	10.0.0.0/16	Local
	172.16.0.0/16	pcx-aaaabbbb
	192.168.0.0/16	pcx-bbbbcccc
VPC C	192.168.0.0/16	Local
	172.16.0.0/16	pcx-aaaaacccc
	10.0.0.0/16	pcx-bbbbcccc

Bloques de CIDR solapados

No puede crear una interconexión de VPC entre VPC que tengan los mismos bloques de CIDR IPv4 o con bloques solapados.

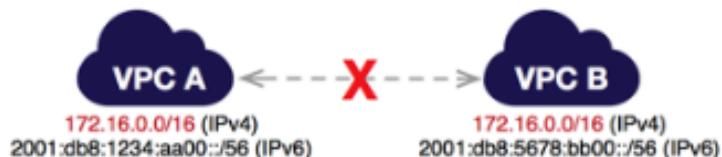


Si las VPC tienen varios bloques de CIDR IPv4, no podrá crear una interconexión de VPC si se superpone cualquiera de los bloques de CIDR (independientemente de que tenga la intención de usar la interconexión de VPC para la comunicación únicamente entre bloques de CIDR que no se superpongan).



Esta limitación también se aplica a VPC que tienen bloques de CIDR IPv6 que no se solapan. Incluso si pretende utilizar la interconexión de VPC solo para la comunicación IPv6, no podrá crear una interconexión de VPC si las VPC tienen los mismos bloques de CIDR IPv4 o si los bloques se solapan.

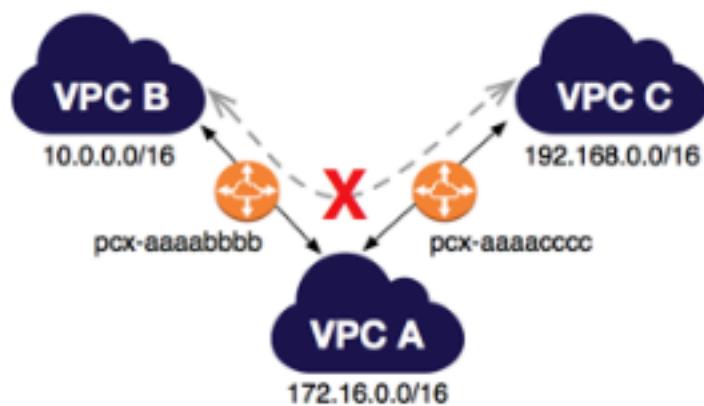
No se admite la comunicación a través de IPv6 para las interconexiones de VPC entre regiones.



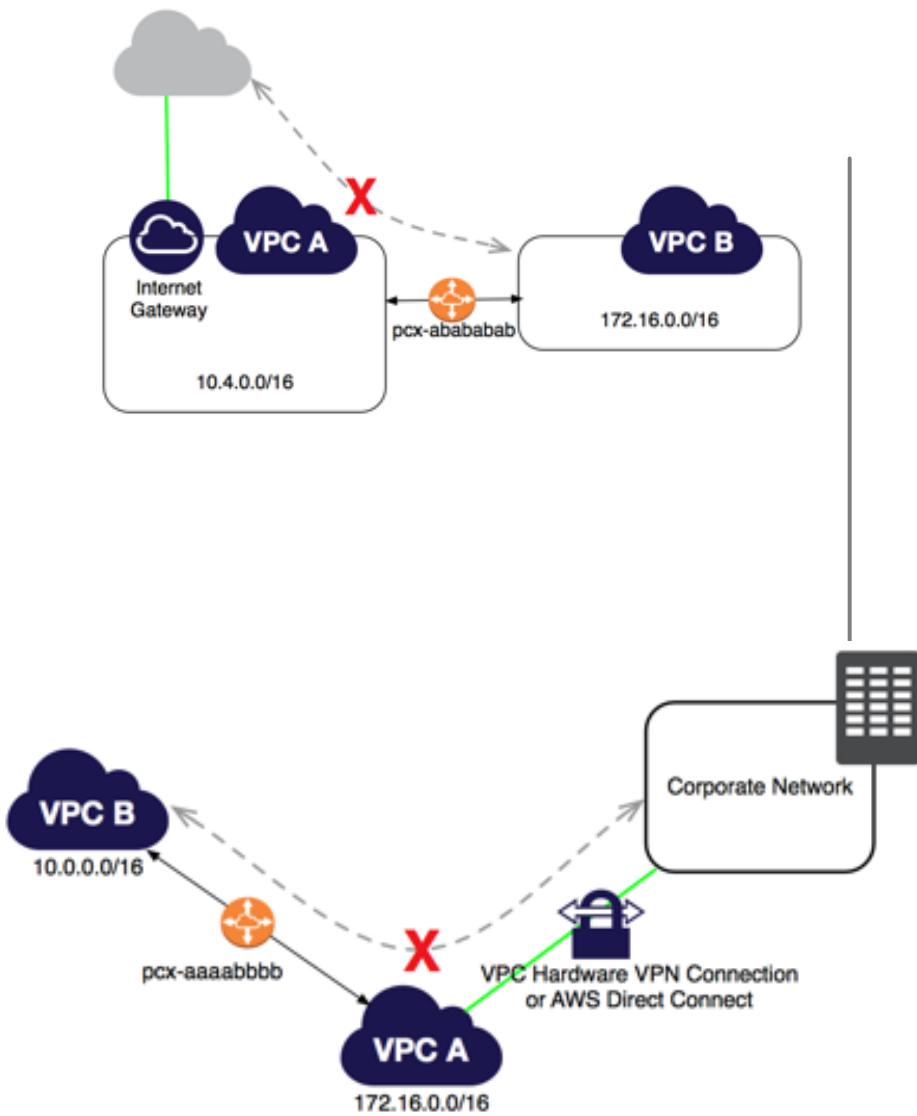
Interconexión transitiva

En lugar de usar la interconexión de VPC, pude usar una AWS Transit Gateway que actúe como un centro de tránsito de red para interconectar las redes de las VPC y locales. Para obtener más información sobre las transit gateways, consulte [¿Qué es una gateway de tránsito?](#) en la *Gateways de tránsito de Amazon VPC*.

Supongamos que tiene una interconexión de VPC entre la VPC A y la VPC B (pcx-aaaabbbb) y entre la VPC A y la VPC C (pcx-aaaacccc). No hay interconexión de VPC entre la VPC B y la VPC C ni puede direccionar paquetes directamente desde la VPC B a la VPC C a través de la VPC A.

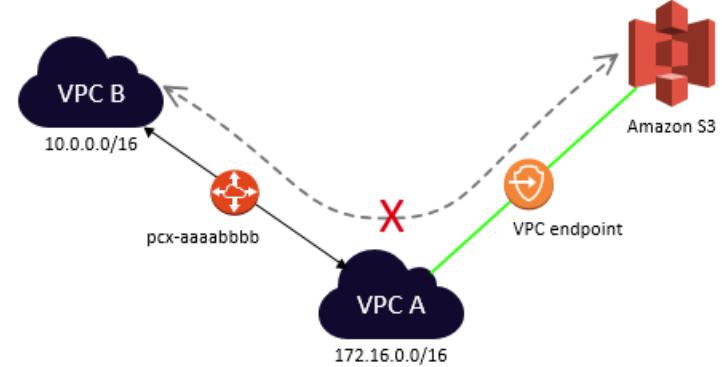


Para direccionar paquetes directamente entre la VPC B y la VPC C, puede crear una interconexión de VPC distinta entre ellas (siempre que no tengan bloques de CIDR solapados). Para obtener más información, consulte [Tres VPC interconectadas](#).



Si ambas VPC, no podrá extender la relación :

- Conexión de VPN o conexión de AWS Direct Connect a una red corporativa
- Conexión a Internet mediante GatewayInternet
- Conexión a Internet de una subred privada mediante un dispositivo NAT
- Punto de enlace de la VPC a un servicio de AWS como, por ejemplo, un punto de enlace a Amazon S3.
- (IPv6) Conexión de ClassicLink. Puede habilitar la comunicación IPv4



IDS vs. IPS

IDS are detection and monitoring tools.

These tools do not take action on their own.

IDS requires a human or another system to look at the results.

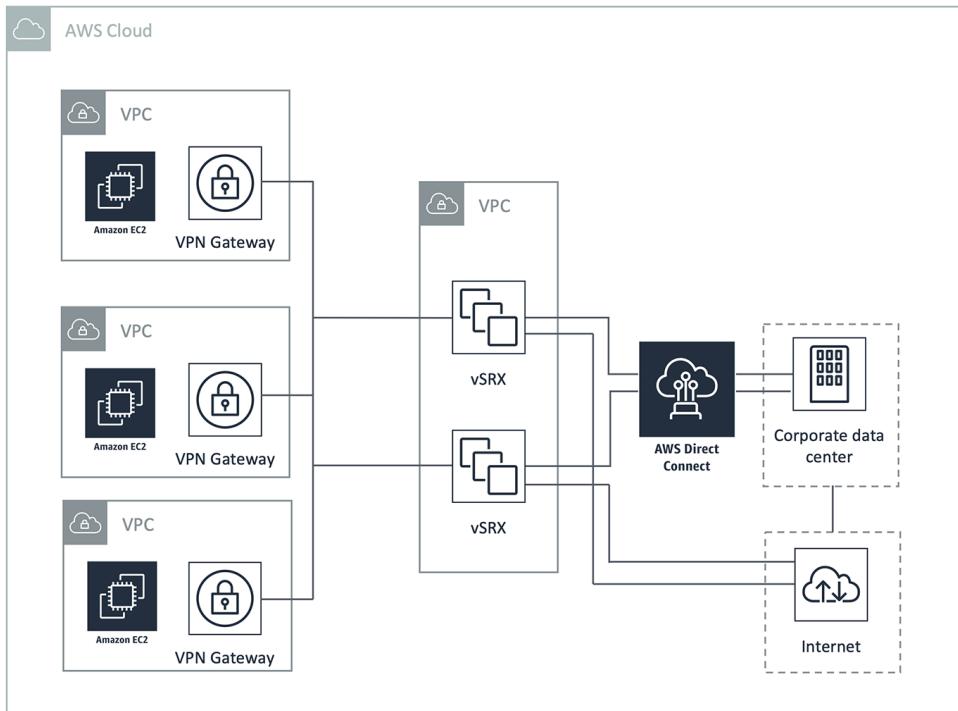
Both read network packets and compare the contents to a database of known threats.

IPS is a control system.

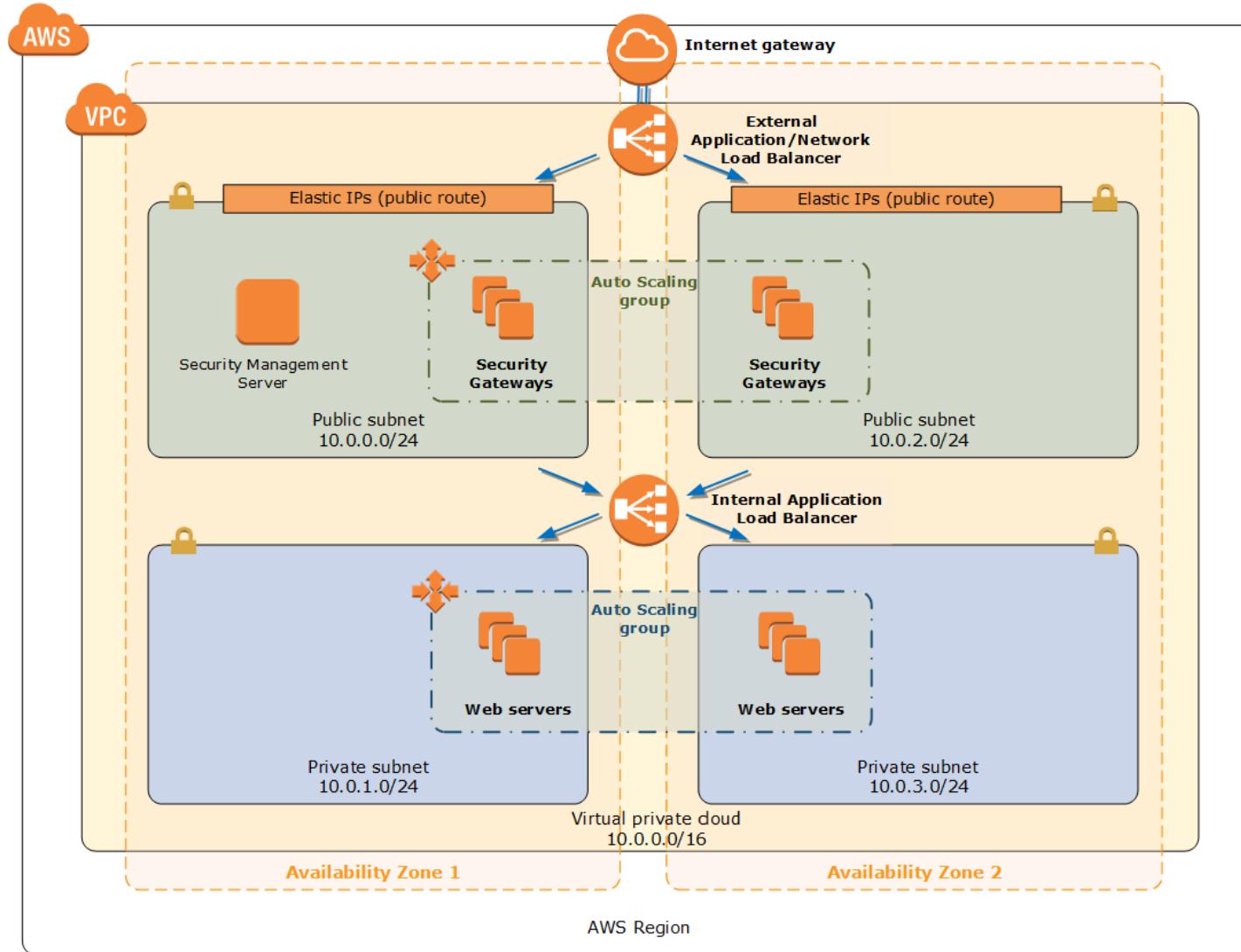
The control system accepts and rejects a packet based on the ruleset.

IPS requires that the database gets regularly updated with new threat data.

IPS / IDS



Sirve para ampliar el nivel de seguridad empresarial integral, mediante la incorporación a la nube de AWS de protección frente a amenazas de día cero, inspección profunda de HTTPS de paquetes, sistema de prevención de intrusiones (IPS) y un completo conocimiento de las identidades y aplicaciones. Se protegen los recursos en la nube de ataques y, simultáneamente, se pone a disposición una conectividad segura. Además, es posible aplicar políticas de seguridad uniformes en toda su organización.
Se debe implementar una VPC de transito para integrarlo



VPC Flow Logs

Within your VPC you could potentially have hundreds or even thousands of resources all communicating together



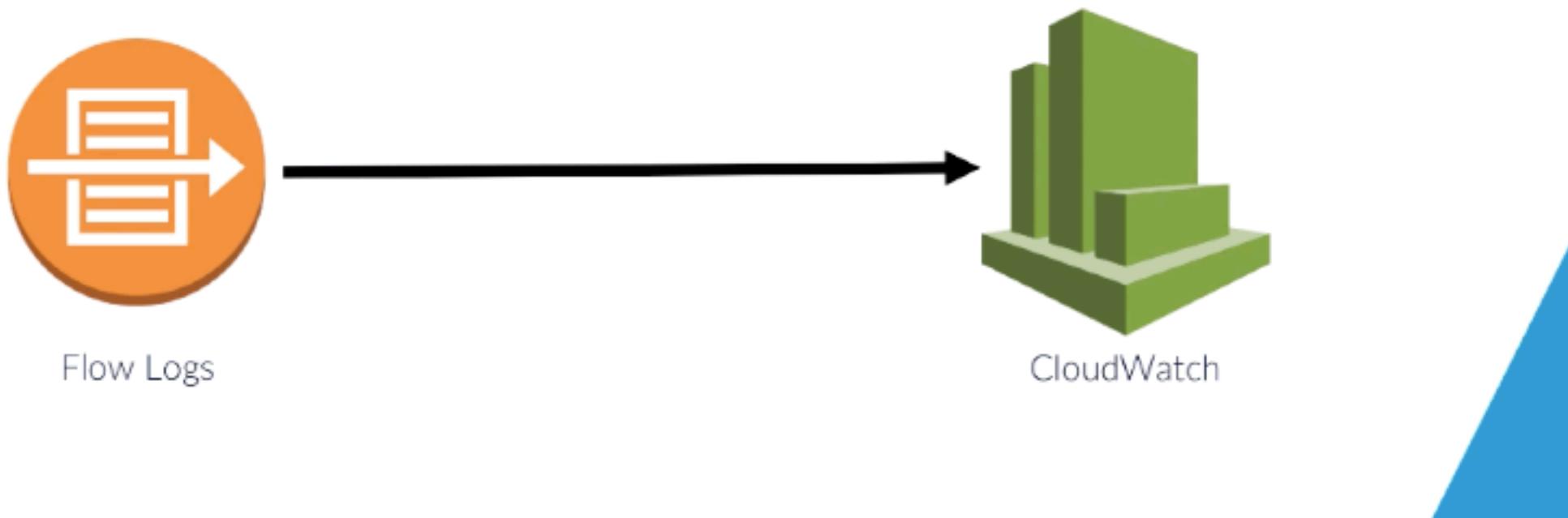
Capture IP traffic information that flows within your VPC

Resolve incidents with network communication and traffic flow

Help spot traffic reaching a destination that should be prohibited

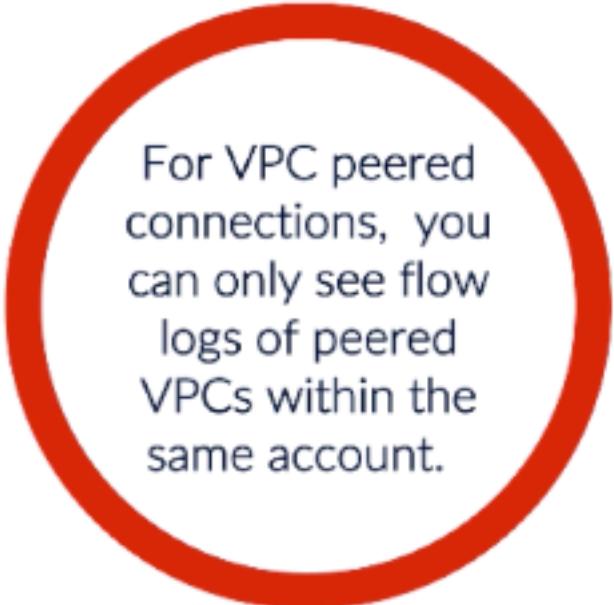
VPC Flow Logs Destination

The log data generated by VPC flow logs is sent to CloudWatch Logs

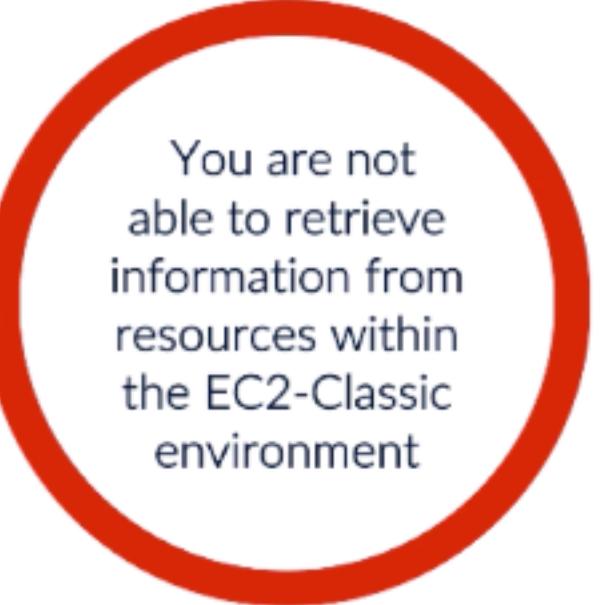


Limitations

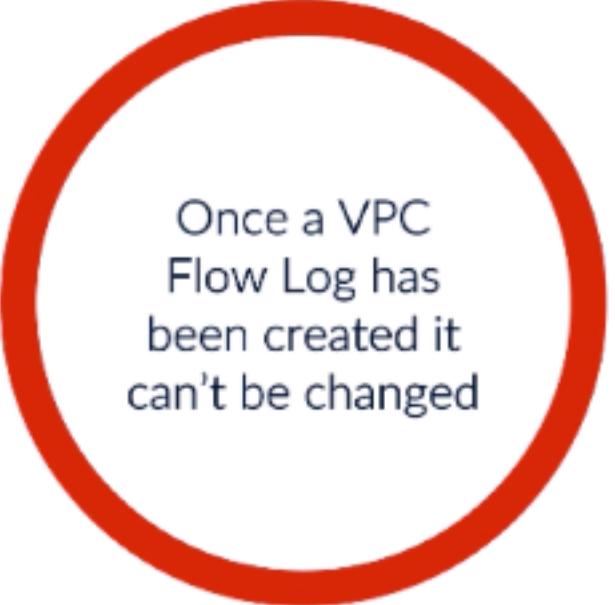
Be aware of the limitations of VPC Flow Logs



For VPC peered connections, you can only see flow logs of peered VPCs within the same account.



You are not able to retrieve information from resources within the EC2-Classic environment



Once a VPC Flow Log has been created it can't be changed

Limitations

The following traffic is not captured by the logs:

- DHCP Traffic within the VPC
- Traffic from instances destined for Amazon DNS Servers
- Traffic destined to the IP addresses for the VPC default router
- Traffic to and from the following:
 - 169.254.169.254 (instance metadata)
 - 169.254.169.123 (Time Sync Service)
- Traffic relating to an Amazon Windows activation license from a Window instance
- Traffic between a Network Load Balancer Network Interface and an Endpoint Network Interface

VPC Flow Logs

You can set up and create a Flow Log against these resources:

A Network interface on one of your instances

A Subnet within your VPC

Your VPC itself



Data is captured for all network interfaces

Publishing Data to CloudWatch

Every network interface that publishes data to the CloudWatch Log Group uses a different log stream



Within each stream there is the flow log event data that shows the content of the log entries



Each of these logs captures data during a windows of approximately 10-15 minutes.

Permissions

To push Flow log data to a CloudWatch Log Group, an IAM Role is required with the relevant permissions:

- This role is selected during the setup and configuration of the VPC Flow Log
- You also need to allow VPC Flow Log to assume the IAM role to deliver logs to CloudWatch
- To review and assess VPC Flow Logs:

```
{  
  "Version": "2012-10-17",  
  "Statement": [  
    {  
      "Sid": "",  
      "Effect": "Allow",  
      "Principal": {  
        "Service": "vpc-flow-logs.amazonaws.com"  
      },  
      "Action": "sts:AssumeRole"  
    }  
  ]  
}
```

Permissions

To push Flow log data to a CloudWatch Log Group, an IAM Role is required for permissions to do so:

- The `logs:GetLogData` permissions enables you to list log events from a data stream
- to Create Flow Logs, you need to also grant the user the IAM permission of `iam:passrole`, which allows the service to assume the role

```
ec2:CreateFlowLogs  
ec2:DeleteFlowLogs  
ec2:DescribeFlowLogs  
logs:GetLogData  
iam:passrole
```

```
version account-id interface-id srcaddr dstaddr srcport dstport  
protocol packets bytes start end action log-status
```

Version: The version of the Flow Log itself

Account-id: This is your AWS account ID

Interface-id: This is the interface ID of which the log stream data applies to

Srcaddr: This is the IP source address

Dstaddr: This is the IP destination address

Srcport: This is the source port being used for the traffic

Dstport: This is the destination port being used for the traffic

Flow Log Record

```
version account-id interface-id srcaddr dstaddr srcport dstport  
protocol packets bytes start end action log-status
```

Protocol: The protocol number being used for the traffic

Packets: The total number of packets sent during the capture

Bytes: the total number of bytes sent during the capture

Start & End: Shows the timestamp of when the capture window started and finished

Action: Shows if the traffic was accepted or rejected by security groups and NACLs

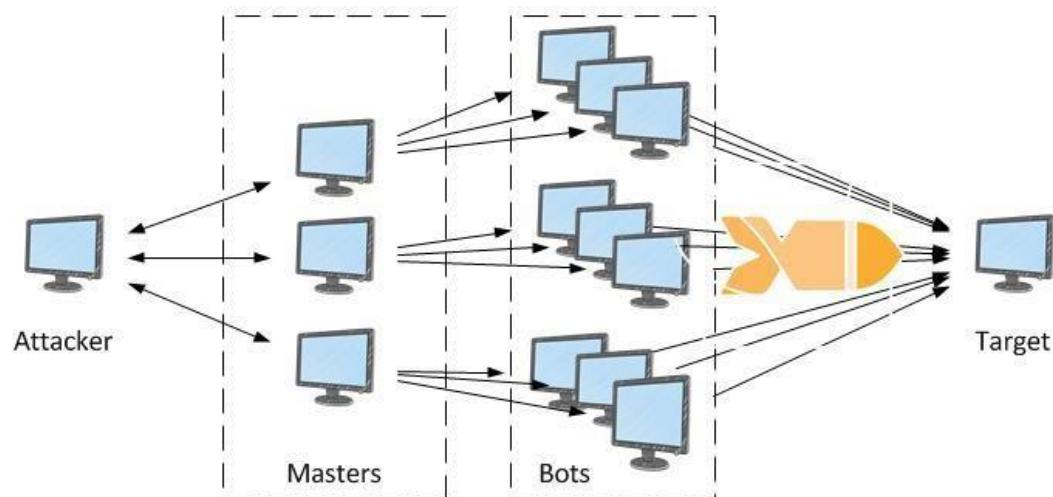
Log-status: shows the status of the logging through 3 different codes:

OK - Data is being received by CloudWatch Logs

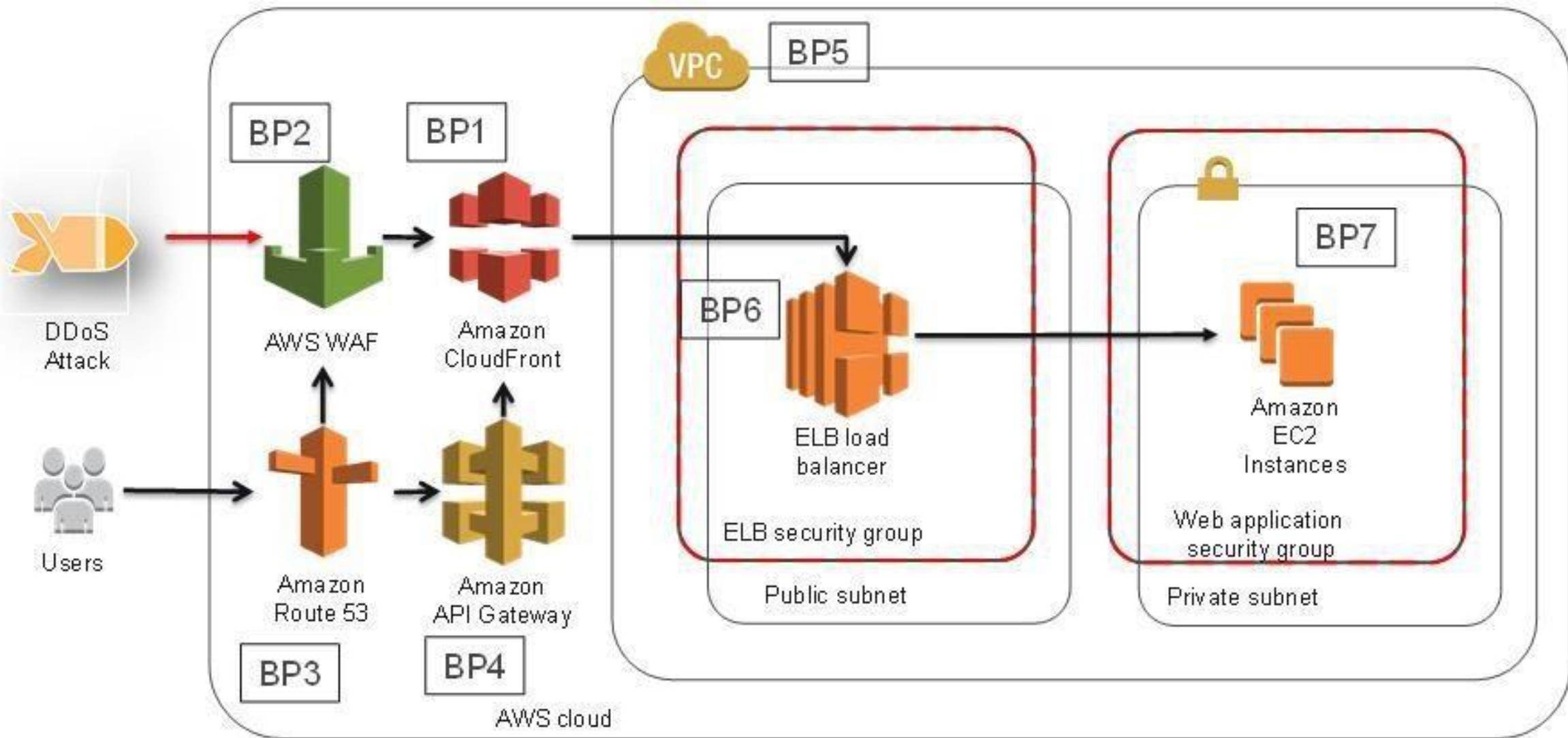
NoData – No traffic to capture during the capture window

SkipData - Some data within the log was captured

Distributed Denial of Service (DDoS) attack, an attacker uses multiple sources – which may be distributed groups of malware infected computers, routers, IoT devices, and other endpoints– to orchestrate an attack against a target. As illustrated in the following DDoS attack diagram (Figure 2), a network of compromised hosts participates in the attack, generating a flood of packets or requests to overwhelm the target.



#	Layer	Unit	Description	Vector Examples
7	Application	Data	Network process to application	HTTP floods, DNS query floods
6	Presentation	Data	Data representation and encryption	TLS abuse
5	Session	Data	Interhost communication	N/A
4	Transport	Segments	End-to-end connections and reliability	SYN floods
3	Network	Packets	Path determination and logical addressing	UDP reflection attacks
2	Data Link	Frames	Physical addressing	N/A
1	Physical	Bits	Media, signal, and binary transmission	N/A



AWS Shield Standard

Para protección contra los ataques de DDoS más comunes y acceso a las herramientas y prácticas recomendadas a los fines de crear una arquitectura resistente a DDoS.

Automáticamente disponible en todos los servicios de AWS.

AWS Shield Advanced

Para protección adicional ante ataques de gran escala y más sofisticados, visibilidad de los ataques y acceso ininterrumpido a los expertos de DDoS para casos complejos. Consulte el [Contrato de nivel de servicio de AWS Shield Advanced](#).

Disponible en:

- Amazon Route 53
- Amazon CloudFront
- Elastic Load Balancing
- AWS Global Accelerator
- IP elástica (Amazon Elastic Compute Cloud y balanceador de carga de red)

AWS Edge Locations	AWS Regions					
	Amazon CloudFront (BP1) with AWS WAF (BP2)	Amazon Route 53 (BP3)	Elastic Load Balancing (BP6)	Amazon API Gateway (BP4)	Amazon VPC (BP5)	Amazon EC2 with Auto Scaling (BP7)
Layer 3 (for example, UDP reflection) attack mitigation	✓	✓	✓	✓	✓	✓
Layer 4 (for example, SYN flood) attack mitigation	✓	✓	✓	✓		
Layer 6 (for example, TLS) attack mitigation	✓		✓	✓		
Reduce attack surface	✓	✓	✓	✓	✓	
Scale to absorb application layer traffic	✓	✓	✓	✓	✓	
Layer 7 (application layer) attack mitigation	✓	✓	✓ (if used with AWS WAF)			
Geographic isolation and dispersion of excess traffic, and larger DDoS attacks	✓	✓				



IAM

AWS IAM OVERVIEW

- AWS Identity and Access Management (IAM) is a web service that helps you securely control access to AWS resources for your users.
- IAM is used to control
 - **Identity** – who can use your AWS resources (authentication)
 - **Access** – what resources they can use and in what ways (authorization)
- IAM can also keep your account credentials private.
- With IAM, multiple IAM users can be created under the umbrella of the AWS account or temporary access can be enabled through identity federation with corporate directory or third party providers
- IAM also enables access to resources across AWS accounts.

IAM FEATURES

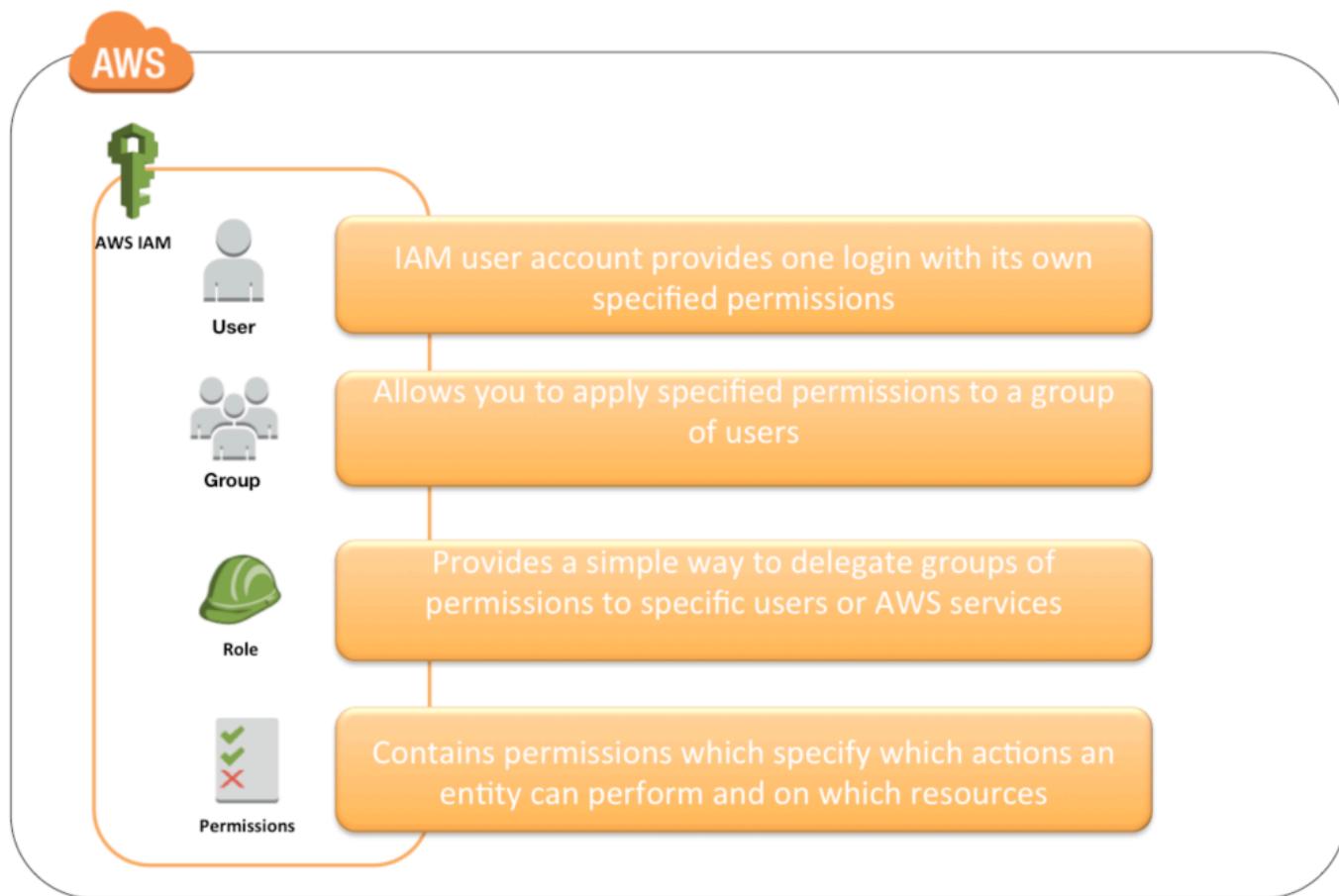
- **Shared access to your AWS account**
 - Grant other people permission to administrate and use resources in your AWS account without having to share your password or access key.
- **Granular permissions**
 - Each user can be granted with different set granular permissions as required to perform their job
- **Secure access to AWS resources for applications that run on EC2**
 - IAM can help provide applications running on EC2 instance temporary credentials that they need in order to access other AWS resources
- **Identity federation**
 - IAM allows users to access AWS resources, without requiring the user to have accounts with AWS, by providing temporary credentials for e.g. through corporate network or Google or Amazon authentication
- **Identity information for assurance**
 - CloudTrail can be used to receive log records that include information about those who made requests for resources in the account.

IAM FEATURES

- **PCI DSS Compliance**
 - IAM supports the processing, storage, and transmission of credit card data by a merchant or service provider, and has been validated as being Payment Card Industry Data Security Standard (PCI DSS) compliant
- **Integrated with many AWS services**
 - IAM integrates with almost all the AWS services
- **Eventually Consistent**
 - IAM, like many other AWS services, is eventually consistent and achieves high availability by replicating data across multiple servers within Amazon's data centers around the world.
 - Changes made to IAM would be eventually consistent and hence would take some time to reflect
- **Free to use**
 - IAM is offered at no additional charge and charges are applied only for use of other AWS products.
- **AWS Security Token Service**
 - IAM provide STS which is an included feature of the AWS account offered at no additional charge.
 - AWS charges only for the use of other AWS services accessed by the AWS STS temporary security credentials.

IDENTITIES

AWS IAM Identities



ACCOUNT ROOT USER

- Root Account Credentials are the email address and password with which you sign-in into the AWS account
- Root Credentials has full unrestricted access to AWS account including the account security credentials which include sensitive information
- IAM Best Practice – Do not use or share the Root account once the AWS account is created, instead create a separate user with admin privilege
- An Administrator account can be created for all the activities which too has full access to the AWS account except the accounts security credentials, billing information and ability to change password
- **Only use to create account.**

ROOT ACCOUNT



Inicio de sesión de usuarios de cuentas raíces

Correo electrónico

luisa.martinezm@gmail.com

Contraseña

[Iniciar sesión](#)

[Iniciar sesión con una cuenta diferente](#)

[¿Has olvidado tu contraseña?](#)



Las cuentas de AWS incluyen 12 meses de acceso a capas gratuitas

Incluye el uso de Amazon EC2,
Amazon S3 y Amazon RDS

Visite aws.amazon.com/free para leer las condiciones completas de la oferta.

IAM USERS

- IAM user represents the person or service who uses the access to interact with AWS.
- **IAM Best Practice – Create Individual Users**
- User credentials can consist of the following
 - Password to access AWS services through **AWS Management Console**
 - **Access Key/Secret Access Key** to access AWS services through API, CLI or SDK
- IAM user starts with no permissions and is not authorized to perform any AWS actions on any AWS resources and should be granted permissions as per the job function requirement
- **IAM Best Practice – Grant least Privilege**
- **Each IAM user is associated with one and only one AWS account.**
- IAM User cannot be renamed from AWS management console and has to be done from CLI or SDK tools.
- IAM handles the renaming of user w.r.t unique id, groups, policies where the user was mentioned as a principal. However, you need to handle the renaming in the policies where the user was mentioned as a resource

USER ACCOUNT

Identity and Access Management (IAM)				
AWS Account (181901723754)				
Panel				 Buscar por nombre de usuario o clave de acceso
Grupos				
Usuarios	Nombre de usuario	Grupos	Antigüedad de la clave de acceso	Antigüedad de la contraseña
rgarcia	developers		80 días	80 días
user1	developers		79 días	80 días
user10	developers		80 días	80 días
user2	developers		80 días	80 días
user3	developers		80 días	80 días
user4	developers		80 días	80 días
user5	developers		80 días	80 días
user6	developers		80 días	80 días
user7	developers		80 días	80 días
user8	developers		80 días	80 días
user9	developers		80 días	80 días

USER ACCOUNT

Añadir usuario(s)

1 2 3 4 5

Establecer los detalles del usuario

Puede añadir varios usuarios a la vez con los mismos permisos y el mismo tipo de acceso. [Más información](#)

Nombre de usuario*

[+ Añadir otro usuario](#)

Seleccionar el tipo de acceso de AWS

Seleccione la forma en que estos usuarios accederán a AWS. Las claves de acceso y las contraseñas generadas automáticamente se proporcionan en último paso. [Más información](#)

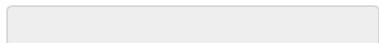
Tipo de acceso* Acceso mediante programación

Habilita una **ID de clave de acceso** y una **clave de acceso secreta** para el SDK, la CLI y la API de AWS, además de otras herramientas de desarrollo.

Acceso a la consola de administración de AWS

Habilita una **contraseña** que permite a los usuarios iniciar sesión en la consola de administración de AWS.

Contraseña de la consola* Contraseña generada automáticamente
 Contraseña personalizada



USER ACCOUNT

Identity and Access Management (IAM)

▼ AWS Account (181901723754)

- Panel
- Grupos
- Usuarios**
- Roles
- Políticas
- Proveedores de identidad
- Configuración de cuenta
- Informe de credenciales

Buscar en IAM

▼ AWS Organizations

- Organization activity
- Service control policies (SCPs)

Credenciales de inicio de sesión

Resumen	• Enlace de inicio de sesión de la consola: https://altia.signin.aws.amazon.com/console
Contraseña de la consola	Habilitada (nunca ha iniciado sesión) Administración
Dispositivo MFA asignado	Sin asignar Administración
Certificados de firma	Ninguna

Claves de acceso

Utilice las claves de acceso para realizar solicitudes seguras de protocolos Query HTTP o REST a las API de servicio de AWS. Para su protección, no comparta nunca las claves secretas. Se recomienda una rotación de claves frecuente. [Más información](#)

Crear una clave de acceso

ID de clave de acceso	Creado	Último uso	Estado	
AKIASUWRQBBVHIDPKA27	2019-08-31 09:00 UTC+0200	N/D	Activa	

Claves de SSH para AWS CodeCommit

Utilice las claves públicas de SSH para autenticar el acceso a los repositorios de AWS CodeCommit. [Más información](#)

USER ACCOUNT (CONSOLA)



Cuenta:

altia

Nombre de usuario:

luisa

Contraseña:

.....

Iniciar sesión

[Iniciar sesión utilizando credenciales de cuenta raíz](#)

[¿Olvidó la contraseña?](#)

AWS re:Invent

Almacenamiento más inteligente con Amazon S3

Automaticice los ahorros en costos con la nueva clase de almacenamiento S3 Intelligent-Tiering, o elija de la cartera de tipos de almacenamiento más amplia del sector



USER ACCOUNT (CLI)

User name	Password	Access key ID	Secret access key
luisa	SBLAHK^sx4A!	AKIASUWRQBBVHIDPKA27	aDysmG8/m2FeFwAlSLifeCchgjdUx86jEB/DPAe1

- Instalar CLI

```
## usar perfiles  
aws configure --profile curso
```

```
MacBook-Pro-de-Luisa:~ luisamartinezmiguez$ aws configure --  
profile curso  
AWS Access Key ID [None]: SBLAHK^sx4A!  
AWS Secret Access Key [None]:AKIASUWRQBBVHIDPKA27  
Default region name [None]: eu-west-1  
Default output format [None]: json
```

IAM GROUPS

- IAM group is a collection of IAM users
- IAM groups can be used to specify permissions for a collection of users sharing the same job function making it easier to manage
- **IAM Best Practice – Use groups to assign permissions to IAM Users**
- A group is not truly an identity because it cannot be identified as a Principal in an access policy. It is only a way to attach policies to multiple users at one time
- A group can have multiple users, while a user can belong to multiple groups (10 max)
- Groups cannot be nested and can only have users within it
- AWS does not provide any default group to hold all users in it and if one is required it should be created with all users assigned to it.
- Renaming of a group name or path, IAM handles the renaming w.r.t to policies attached to the group, unique ids, users within the group. However, IAM does not update the policies where the group is mentioned as a resource and must be handled manually
- Deletion of the groups requires you to detach users and managed policies and delete any inline policies before deleting the group. With AWS management console, the deletion and detachment is taken care of.

MULTIFACTOR AUTHENTICATION (MFA)

- For increased security and to help protect the AWS resources, Multi-Factor authentication can be configured
- **IAM Best Practice – Enable MFA on Root accounts and privilege users**
- Multi-Factor Authentication can be configured using
 - Security token-based
 - AWS Root user or IAM user can be assigned a hardware/virtual MFA device
 - Device generates a six digit numeric code based upon a time-synchronized one-time password algorithm which needs to be provided during authentication
 - SMS text message-based (Preview Mode)
 - IAM user can be configured with the phone number of the user's SMS-compatible mobile device which would receive a 6 digit code from AWS
 - SMS-based MFA is available only for IAM users and does not work for AWS root account
- MFA needs to be enabled on the Root user and IAM user separately as they are distinct entities. Enabling MFA on Root does not enable it for all other users
- MFA device can be associated with only one AWS account or IAM user and vice versa
- If the MFA device stops working or is lost, you won't be able to login into the AWS console and would need to reach out to AWS support to deactivate MFA
- MFA protection can be enabled for service API's calls using “Condition”: {“Bool”: {“aws:MultiFactorAuthPresent”: “true”}} and is available only if the service supports temporary security credentials.

AWS IAM ROLE

- IAM role is very similar to a user, in that it is an identity with permission policies that determine what the identity can and cannot do in AWS.
- IAM role is not intended to be uniquely associated with a particular user, group or service and is intended to be assumable by anyone who needs it.
- Role does not have any credentials (password or access keys) associated with it and whoever assumes the role is provided with a dynamic temporary credentials
- Role helps in access delegation to grant permissions to someone that allows access to resources that you control
- Roles can help to prevent accidental access to or modification of sensitive resources
- Modification of a Role can be done anytime and the changes are reflected across all the entities associated with the Role immediately

AWS IAM ROLE

- AM Role plays a very important role in the following scenarios
 - Services like EC2 instance running an application that needs to access other AWS services
 - Allowing users from different AWS accounts have access to AWS resources in different account, instead of having to create users
 - Company uses a Corporate Authentication mechanism and don't want the User to authenticate twice or create duplicate users in AWS
 - Applications allowing login through external authentication mechanism e.g. Amazon, Facebook, Google etc
- Role can be assumed by
 - IAM user within the same AWS account
 - IAM user from a different AWS account
 - AWS service such as EC2, EMR to interact with other services
 - An external user authenticated by an external identity provider (IdP) service that is compatible with SAML 2.0 or OpenID Connect (OIDC), or a custom-built identity broker.

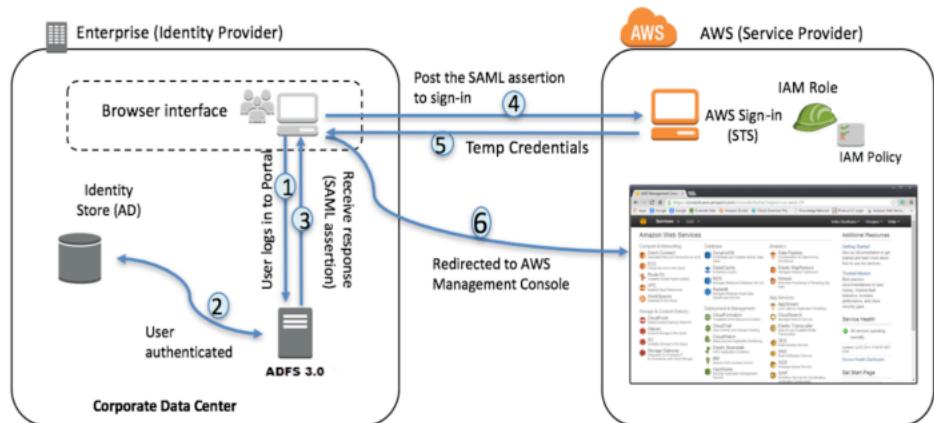
AWS IAM ROLE

- Role involves defining two policies
 - **Trust policy**
 - Trust policy defines – **who can assume the role**
 - Trust policy involves setting up a trust between the account that owns the resource (trusting account) and the account who owns the user that needs access to the resources (trusted account)
 - **Permissions policy**
 - Permissions policy defines – **what they can access**
 - Permissions policy determines authorization, which grants the user of the role the needed permissions to carry out the desired tasks on the resource

AWS IAM ROLE

- **Federation** is creating a trust relationship between an external Identity Provider (IdP) and AWS
 - Users can also sign in to an enterprise identity system that is compatible with SAML
 - Users can sign in to a web identity provider, such as Login with Amazon, Facebook, Google, or any IdP that is compatible with OpenID connect (OIDC).
 - When using OIDC and SAML 2.0 to configure a trust relationship between these external identity providers and AWS, the user is assigned to an IAM role and receives temporary credentials that enables the user to access AWS resources

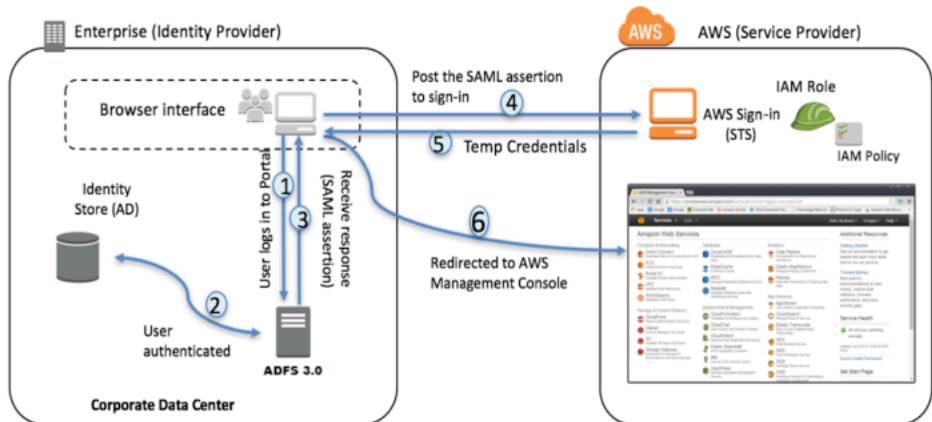
AWS FEDERATED AUTHENTICATION WITH ACTIVE DIRECTORY FEDERATION SERVICES (AD FS) SSO



1. Corporate user accesses the corporate Active Directory Federation Services portal sign-in page and provides Active Directory authentication credentials.
2. AD FS authenticates the user against Active Directory.
3. Active Directory returns the user's information, including AD group membership information.
4. AD FS dynamically builds ARNs by using Active Directory group memberships for the IAM roles and user attributes for the AWS account IDs, and sends a signed assertion to the user's browser with a redirect to post the assertion to AWS STS.
5. Temporary credentials are returned using STS AssumeRoleWithSAML.
6. The user is authenticated and provided access to the AWS management console.

When you use an IAM identity provider, you don't have to create custom sign-in code or manage your own user identities. The IdP provides that for you. Your external users sign in through a well-known IdP, such as Login with Amazon, Facebook, or Google. You can give those external identities permissions to use AWS resources in your account. IAM identity providers help keep your AWS account secure because you don't have to distribute or embed long-term security credentials, such as access keys, in your application.

SSO CON AWS



1. El user hace log contra su AS en local (<https://Fully.Qualified.Domain.Name.Here/adfs/ls/IdpInitiatedSignOn.aspx>)
2. Se autentifica el usuario contra el AD (user + pass) .Autentifica y busca que perfil tiene ese usuario.
3. El usuario recibe una SAML assertion con la autenticación. También tiene la información del rol que usará en AWS.
4. El usuario usa la SAML assertion con el AWS sign-in endpoint for SAML (<https://signin.aws.amazon.com/saml>)
5. Se le devuelven las credenciales temporales para esa cuenta y rol.
6. Se le redirige a AWS y se accede con las credenciales temporales.

AWS STS & TEMPORARY CREDENTIALS

- AWS Security Token Service (STS) helps create and provide trusted users with temporary security credentials that can control access to AWS resources
- AWS STS is a global service with a single endpoint <https://sts.amazonaws.com>
- AWS STS API calls can be made either to a global endpoint or to one of the regional endpoints. Regional endpoint can help reduce latency and improve the performance of your API calls
- Temporary Credentials are similar to Long Term Credentials except for
 - are short term and are regularly rotated
 - can be configured to last from few minutes to several hours
 - do not have to be embedded or distributed
 - are not stored or attached with the User but are generated dynamically and provided to the user as and when requested

STS

- The AWS Security Token Service (STS) is a web service that enables you to request temporary, limited-privilege credentials for AWS Identity and Access Management (IAM) users or for users that you authenticate (federated users).
- **Endpoints**
 - By default, AWS Security Token Service (STS) is available as a global service, and all AWS STS requests go to a single endpoint at <https://sts.amazonaws.com>. Global requests map to the US East (N. Virginia) region. AWS recommends using Regional AWS STS endpoints instead of the global endpoint to reduce latency, build in redundancy, and increase session token validity.
 - Most AWS Regions are enabled for operations in all AWS services by default. Those Regions are automatically activated for use with AWS STS. Some Regions, such as Asia Pacific (Hong Kong), must be manually enabled. To learn more about enabling and disabling AWS Regions, see [Managing AWS Regions](#) in the AWS General Reference. When you enable these AWS Regions, they are automatically activated for use with AWS STS. You cannot activate the STS endpoint for a Region that is disabled. Tokens that are valid in all AWS Regions are longer than tokens that are valid in Regions that are enabled by default. Changing this setting might affect existing systems where you temporarily store tokens.

Identity & Access Management (IAM):

X

IAM Security Token Service (STS):

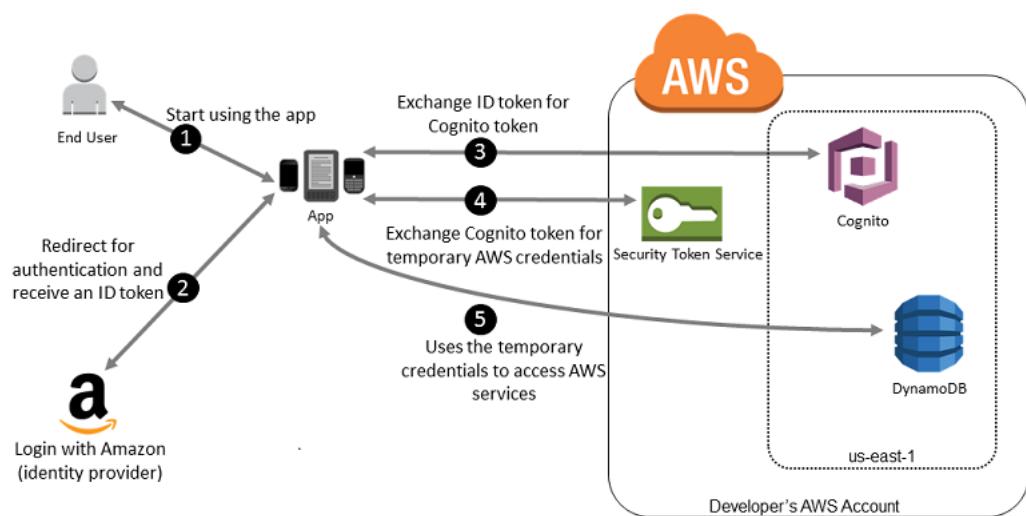
- STS allows you to create temporary security credentials that grant trusted users access to your AWS resources.
- These temporary credentials are for short-term use, with a configurable session duration between 15 minutes and 12 or 36 hours.
- Once expired, they can no longer be used to access your AWS resources.
- When requested through an STS API call, a credential object is returned containing:
 - ***Session Token***
 - ***An Access Key ID***
 - ***A Secret Access key***
 - ***Expiration Timestamp***

When to use STS:

- Identity Federation:
 - Enterprise identity federation (authenticate through your companies network)
 - STS supports Security Assertion Markup Language (SAML), which allows for use of Microsoft Active Directory (of your own solutions).
 - Web identity federation (3rd party identity providers, i.e. Facebook, Google, Amazon)
- Roles for Cross-Account Access
 - Used for organizations that have more than one AWS account.
- Roles for Amazon EC2 (and other AWS services)
 - Grant access an to application running on an EC2 instance to access other AWS services without having to imbed credentials.

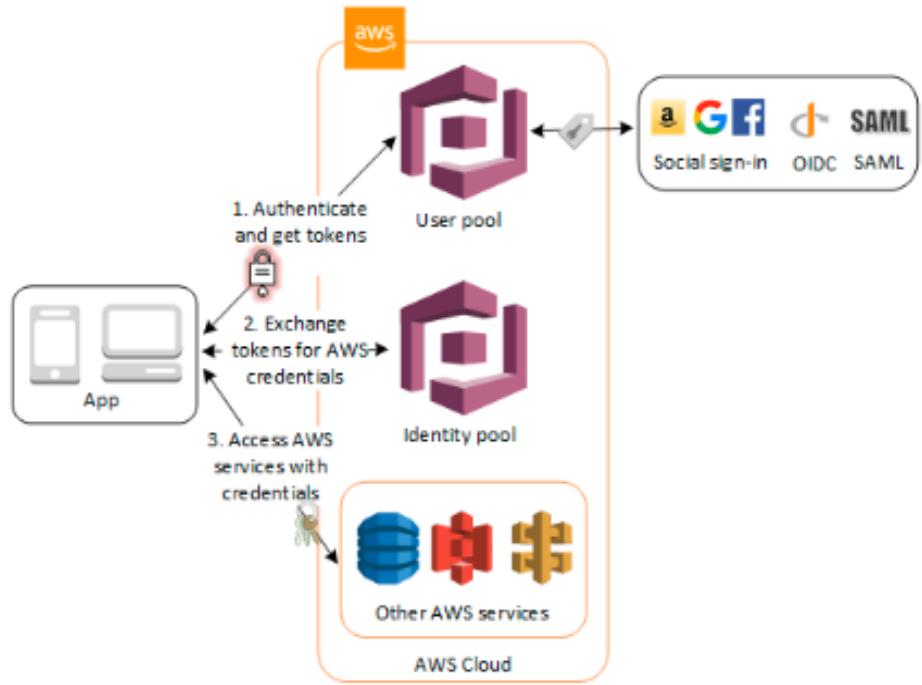
For mobile applications AWS recommend using Cognito rather than STS directly. Cognito provides additional mobile specific functionality which makes the flow easier to manage.

MOBILE OR WEB IDENTITY FEDERATION WITH COGNITO



- Social and enterprise identity federation:
With Amazon Cognito, your users can sign-in through social identity providers such as Google, Facebook, and Amazon, and through enterprise identity providers such as Microsoft Active Directory using SAML.
- Access control for AWS resources
Amazon Cognito provides solutions to control access to AWS resources from your app. You can define roles and map users to different roles so your app can access only the resources that are authorized for each user.
- Cognito is an Identity Broker which handles interaction between your applications and the Web ID provider
- Recommended for all mobile applications

MOBILE OR WEB IDENTITY FEDERATION WITH COGNITO



- The two main components of Amazon Cognito are **user pools** and **identity pools**.
 - A user pool is a user directory in Amazon Cognito. With a user pool, your users can sign in to your web or mobile app through Amazon Cognito. Your users can also sign in through social identity providers like Facebook or Amazon, and through SAML identity providers. Whether your users sign in directly or through a third party, all members of the user pool have a directory profile that you can access through an SDK.
 - After successfully authenticating a user, Amazon Cognito issues JSON web token (JWT) that you can use to secure and authorize access to your own APIs, or exchange for AWS credentials.

ROLE TYPE

- AWS Service Roles
- Instance Profile
- Cross-Account access Roles

SERVICE ROLES

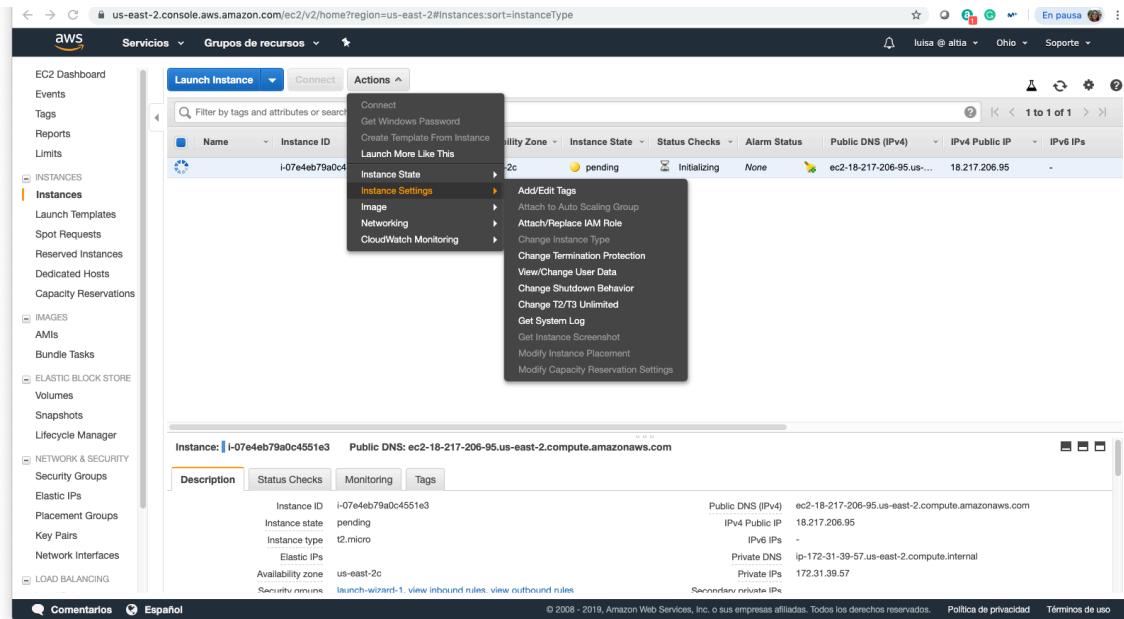
- Some AWS services need to interact with other AWS services for e.g. *EC2 interacting with S3, SQS etc*
- Best practice is to assign these services with IAM roles instead of embedding or passing IAM user credentials directly into an instance, because distributing and rotating long-term credentials to multiple instances is challenging to manage and a potential security risk.
- AWS automatically provides temporary security credentials for these services e.g. *Amazon EC2 instance* to use on behalf of its applications
- Deleting a role or instance profile that is associated with a running EC2 instance will break any applications running on the instance

```
{  
  "Version": "2012-10-17",  
  "Statement": [  
    {"Effect": "Allow",  
     "Action": "s3>List*",  
     "Resource": [  
       "arn:aws:s3:::mybucket",  
       "arn:aws:s3:::mybucket/*"  
     ]  
   }  
}
```

SERVICE ROLES

- Create a IAM role with services who would use it for e.g. EC2 as trusted entity and define permission policies with the access the service needs
- Associated a Role (actually an Instance profile) with the EC2 service when the instance is launched
- Temporary security credentials are available on the instance and are automatically rotated before they expire so that a valid set is always available
- Application can retrieve the temporary credentials either using the Instance metadata directly or through AWS SDK
- Applications running on the EC2 instance can now use the permissions defined in the Role to access other AWS resources
- Application, if caching the credentials, needs to make sure it uses the correct credentials before they expire

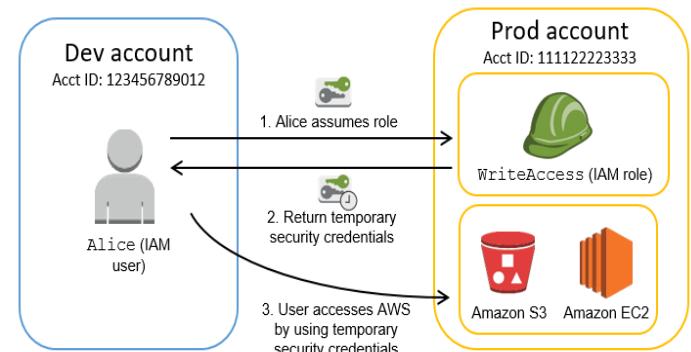
INSTANCE PROFILE



- An instance profile is a container for an IAM role that you can use to pass role information to an EC2 instance when the instance starts.
- If a Role is created for EC2 instance or any other service that uses EC2 through AWS Management Console, AWS creates a Instance profile automatically with the same name as the Role. However, if the Role is created through CLI the instance profile needs to be created as well
- **An instance profile can contain only one IAM role.** However, a role can be included in multiple instance profiles.

CROSS-ACCOUNT ACCESS ROLES

- IAM users can be granted permission to switch roles within the same AWS account or to roles defined in other AWS accounts that you own.
- Roles can also be used to delegate permissions to IAM users from AWS accounts owned by Third parties
 - You must explicitly grant the users permission to assume the role.
 - Users must actively switch to the role using the AWS Management Console.
 - Multi-factor authentication (MFA) protection can be enabled for the role so that only users who sign in with an MFA device can assume the role
- However, only One set of permissions are applicable at a time. User who assumes a role temporarily gives up his or her own permissions and instead takes on the permissions of the role. When the user exits, or stops using the role, the original user permissions are restored.



CROSS-ACCOUNT ACCESS ROLES

- IAM users can be granted permission to switch roles within the same AWS account or to roles defined in other AWS accounts that you own.
- Roles can also be used to delegate permissions to IAM users from AWS accounts owned by Third parties
 - You must explicitly grant the users permission to assume the role.
 - Users must actively switch to the role using the AWS Management Console.
 - Multi-factor authentication (MFA) protection can be enabled for the role so that only users who sign in with an MFA device can assume the role
- However, only One set of permissions are applicable at a time. User who assumes a role temporarily gives up his or her own permissions and instead takes on the permissions of the role. When the user exits, or stops using the role, the original user permissions are restored.

POLICIES

- An IAM policy sets permission and controls access to AWS resources. Policies are stored in AWS as JSON documents. Permissions specify who has access to the resources and what actions they can perform. For example, a policy could allow an IAM user to access one of the buckets in Amazon S3. The policy would contain the following information:
 - Who can access it
 - What actions that user can take
 - Which AWS resources that user can access
 - When they can be accessed
 - In JSON format

POLICIES

```
{  
  "Version": "2017-10-17",  
  "Id": "S3-Account-Permissions",  
  "Statement": [  
    {"Sid": "AddPublicReadPermissions", "Effect": "Allow", "Principal": "*", "Action": "s3:*", "Resource": ["arn:AWS:s3:::bucket/*"]}  
  ]  
}
```

→ Specify Actions(Read/Write/Delete)

→ Give permissions(Allow/Deny)

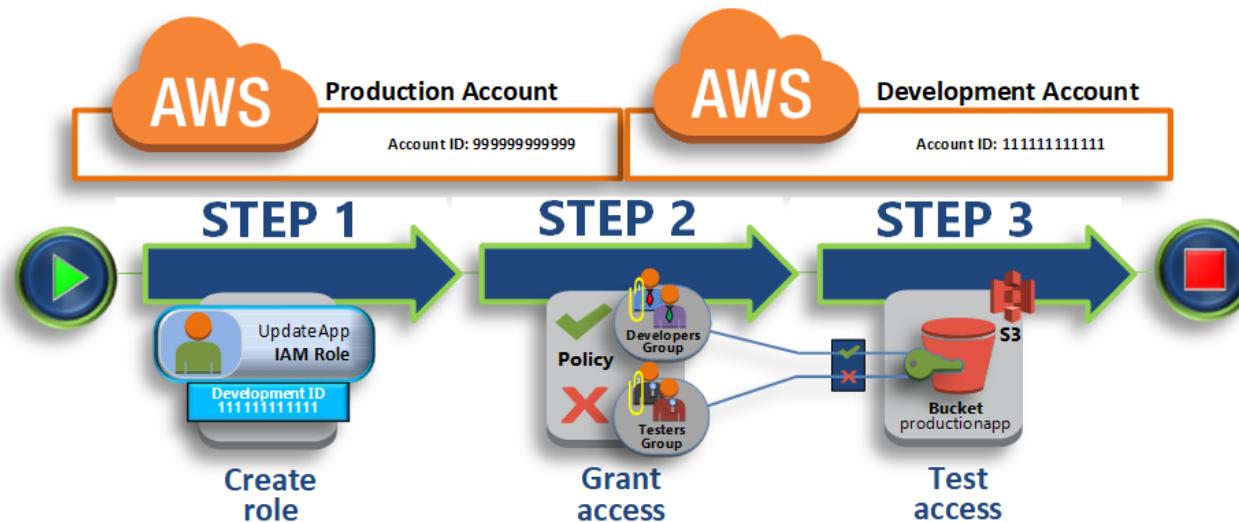
→ Who can Access it

→ What action can a user take

→ Specify the resource

HOW DOES IAM WORK?

- The IAM workflow includes the following six elements:
 1. **Principal** is an entity that can perform actions on an AWS resource. A user, a role or an application can be a principal.
 2. **Authentication** is the process of confirming the identity of the principal trying to access an AWS product. The principal must provide its credentials or required keys for authentication.
 3. **Request**: A principal sends a request to AWS specifying the action and which resource should perform it.
 4. **Authorization**: By default, all resources are denied. IAM authorizes a request only if all parts of the request are allowed by a matching policy. After authenticating and authorizing the request, AWS approves the action.
 5. **Actions** are used to view, create, edit or delete a resource.
 6. **Resources**: A set of actions can be performed on a resource related to your AWS account.



▼ Configuración de cuenta

ID de cuenta: 181901723754
 Vendedor: AWS EMEA SARL
 Nombre de cuenta: japon
 Contraseña: *****

dev

▼ Configuración de cuenta

ID de cuenta: 631662128664
 Vendedor: AWS EMEA SARL
 Nombre de cuenta: rusia
 Contraseña: *****

pro

User	Group	
devop	Developers	Usuarios de Devop
test1	Testers	

User name	Password	Console login link
test1	Test1234	https://altia.signin.aws.amazon.com/console
develop1	Test1234	https://altia.signin.aws.amazon.com/console

En la cuenta de Pro (ID number 631662128664) crear la policy con los permisos que luego se usará para crear el rol : luisa.ejemplo1

Filtro:		
Acción (4 de 84) Mostrar 80 restantes	Recurso	Condición de solicitud
Enumeración (1 de 3 acciones)		
ListAllMyBuckets	Todos los recursos	Ninguna
Lectura (1 de 35 acciones)		
GetObject	BucketName string like testluisa, ObjectPath string like All	Ninguna
Escritura (2 de 32 acciones)		
DeleteObject	BucketName string like testluisa, ObjectPath string like All	Ninguna
PutObject	BucketName string like testluisa, ObjectPath string like All	Ninguna

arn:aws:iam::631662128664:policy/luisa.ejemplo1

```
ADFS-Master-Template.yaml | ej1.json
1 {
2     "Version": "2012-10-17",
3     "Statement": [
4         {
5             "Effect": "Allow",
6             "Action": "s3>ListAllMyBuckets",
7             "Resource": "*"
8         },
9         {
10            "Effect": "Allow",
11            "Action": [
12                "s3>ListBucket",
13                "s3>GetBucketLocation"
14            ],
15            "Resource": "arn:aws:s3:::testluisa"
16        },
17        {
18            "Effect": "Allow",
19            "Action": [
20                "s3>GetObject",
21                "s3>PutObject",
22                "s3>DeleteObject"
23            ],
24            "Resource": "arn:aws:s3:::testluisa/*"
25        }
26    ]
27 }
28 }
```

En la cuenta de Pro (ID number 631662128664) crear la policy con los permisos que luego se usará para crear el rol : luisa.ejemplo1

Filtro:		
Acción (4 de 84) Mostrar 80 restantes	Recurso	Condición de solicitud
Enumeración (1 de 3 acciones)		
ListAllMyBuckets	Todos los recursos	Ninguna
Lectura (1 de 35 acciones)		
GetObject	BucketName string like testluisa, ObjectPath string like All	Ninguna
Escritura (2 de 32 acciones)		
DeleteObject	BucketName string like testluisa, ObjectPath string like All	Ninguna
PutObject	BucketName string like testluisa, ObjectPath string like All	Ninguna

arn:aws:iam::631662128664:policy/luisa.ejemplo1

Se crea el rol que permite que alguien de la cuenta de DEV, con STS pueda asumir el rol trustluisa de PRO

Identity and Access Management (IAM)

- AWS Account (631662128664)
 - Panel
 - Grupos
 - Usuarios
 - Roles**
 - Políticas
 - Proveedores de identidad
 - Configuración de cuenta
 - Informe de credenciales
- Buscar en IAM

Roles > trustluisa

Resumen

ARN de rol: arn:aws:iam::631662128664:role/trustluisa

Descripción del rol: Allows S3 to call AWS services on your behalf. | Editar

ARN del perfil de instancia:

Ruta: /

Hora de creación: 2019-08-31 17:42 UTC+0200

Duración máxima de la sesión de la CLI o la API: 1 hora | Editar

Proporcionar este enlace a los usuarios que pueden cambiar de rol en la consola: <https://signin.aws.amazon.com/switchrole?roleName=trustluisa&account=altranrusia2018>

Permisos Relaciones de confianza Etiquetas Access Advisor Revocar las sesiones

Políticas de permisos (1 política aplicada)

Asociar políticas Añadir una política insertada

Nombre de la política	Tipo de política
luisa.ejemplo1	Política administrada

Documento de política

```
1 Version: "2012-10-17",
2 Statement: [
3   {
4     Effect: "Allow",
5     Principal: {
6       AWS: "arn:aws:iam::181901723754:root"
7     },
8     Action: "sts:AssumeRole",
9     Condition: {}
10 }
11 ]
12 ]
13 }
```

La relación de confianza es para toda la cuenta origen

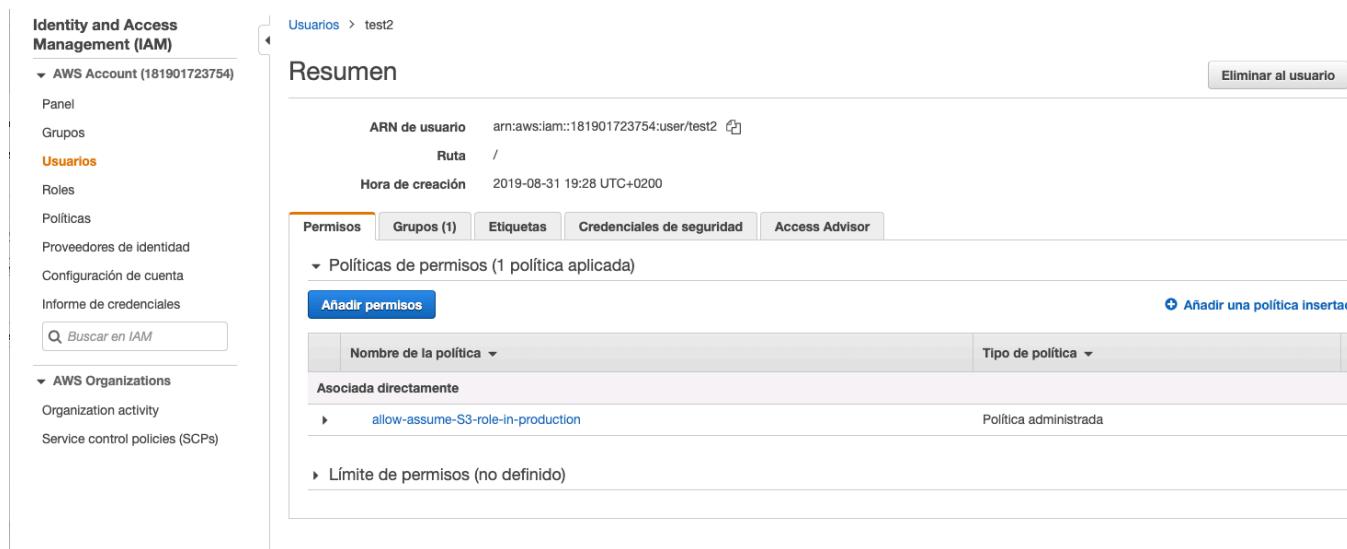
<https://signin.aws.amazon.com/switchrole?roleName=trustluisa&account=altranrusia2018>

arn:aws:iam::631662128664:role/trustluisa

Crear la policy que permite usar STS para asumir el rol en otra cuenta

```
{  
  "Version": "2012-10-17",  
  "Statement": {  
    "Effect": "Allow",  
    "Action": "sts:AssumeRole",  
    "Resource": "arn:aws:iam::631662128664:role/trustluisa"  
  }  
}
```

Asociar la policy recién creada a usuario o grupo



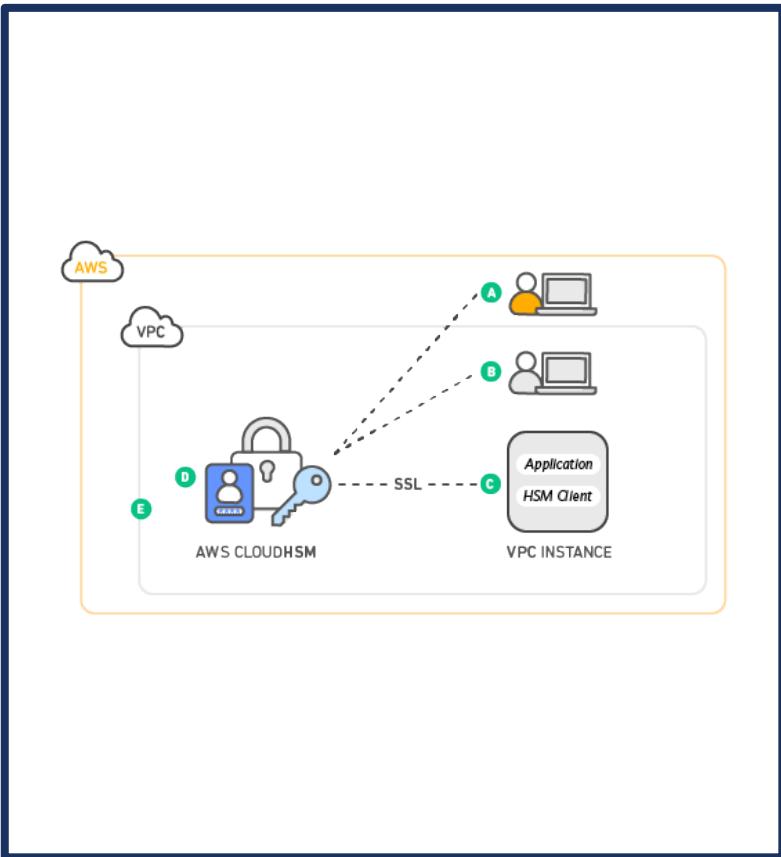
The screenshot shows the AWS Identity and Access Management (IAM) service interface. On the left, there's a navigation sidebar with options like 'Identity and Access Management (IAM)', 'AWS Account (181901723754)', 'Panel', 'Grupos', 'Usuarios' (which is selected), 'Roles', 'Políticas', 'Proveedores de identidad', 'Configuración de cuenta', 'Informe de credenciales', and a search bar labeled 'Buscar en IAM'. The main content area is titled 'Resumen' for the user 'test2'. It displays the ARN of the user, the creation date (2019-08-31 19:28 UTC+0200), and the path '/'. Below this, there are tabs for 'Permisos', 'Grupos (1)', 'Etiquetas', 'Credenciales de seguridad', and 'Access Advisor'. The 'Permisos' tab is active, showing a single policy applied: 'allow-assume-S3-role-in-production'. This policy is listed under 'Políticas de permisos (1 política aplicada)' and is described as being 'Asociada directamente'. There are buttons for 'Añadir permisos' and 'Añadir una política insertada'. At the bottom, there's a section for 'Límite de permisos (no definido)'.

AWS CloudHSM

AWS CloudHSM es un módulo de seguridad de hardware (HSM) basado en la nube que le permite generar y usar con facilidad sus propias claves de cifrado en la nube de AWS. Con CloudHSM, puede administrar sus propias claves de cifrado con los HSM validados por FIPS 140-2 de nivel 3. CloudHSM le ofrece la flexibilidad de integrarse con su aplicación con API estándares del sector, como PKCS#11, Java Cryptography Extensions (JCE) y bibliotecas de Microsoft CryptoNG (CNG).

CloudHSM cumple con los estándares y le permite exportar todas las claves a la mayoría de los demás HSM a la venta, sujeto a las configuraciones. Es un servicio totalmente administrado que automatiza las arduas tareas administrativas por usted, como el aprovisionamiento de hardware, los parches de software, la alta disponibilidad y las copias de seguridad. CloudHSM también le permite ajustar la escala con rapidez mediante la incorporación o eliminación de capacidad de HSM bajo demanda, sin costos por anticipado.

CLOUDHSM



- AWS CloudHSM se ejecuta en su propia Amazon Virtual Private Cloud (VPC), lo que le permite utilizar sus HSM con aplicaciones ejecutadas en sus instancias de Amazon EC2. Con CloudHSM, puede utilizar los controles de seguridad de VPC estándar para administrar el acceso a sus HSM. Sus aplicaciones se conectan a sus HSM con canales SSL mutuamente autenticados establecidos por su software de cliente de HSM. Como sus HSM están ubicados en centros de datos de Amazon cercanos a sus instancias de EC2, puede reducir la latencia de red entre sus aplicaciones y HSM en comparación con un HSM local.
 - A: AWS administra el dispositivo de módulo de seguridad de hardware (HSM), pero no tiene acceso a sus claves.
 - B: usted controla y administra sus propias claves.
 - C: el rendimiento de las aplicaciones mejora (debido a la proximidad con las cargas de trabajo de AWS).
 - D: proteja el almacenamiento de claves con hardware resistente a manipulaciones disponible en varias zonas de disponibilidad (AZ).
 - E: sus HSM se encuentran en su Virtual Private Cloud (VPC), aisladas de otras redes de AWS.

KMS VS CLOUDHSM VS ON-PREM HSM

	Private or Hybrid cloud	Single cloud provider	Multicloud
Recommendation	HSMs already provisioned in the enterprise's data center	Cloud provider's KMS or KMS enhanced by cloud provider's choice of HSM	HSM as a Service
Advantages	HSM security	Ease of management (optional BYOK)	Ease of management, HSM-level security, additional defense against data breaches
Alternatives	HSM as a Service to eliminate the ongoing cost and overhead of provisioning HSMs	HSM as a Service to maintain encryption keys separate from data for additional defense against data breaches	

¿Qué es AWS Directory Service?

AWS Directory Service ofrece varias formas de utilizar Amazon Cloud Directory y Microsoft Active Directory (AD) con otros servicios de AWS. En los directorios se almacena información sobre usuarios, grupos y dispositivos, y los administradores pueden usarlos para administrar el acceso a la información y recursos. **¿Cuál debe elegir?**

Qué necesita hacer?	Opciones recomendadas de AWS Directory Service
Necesito Active Directory o LDAP para mis aplicaciones en la nube	Seleccione AWS Directory Service for Microsoft Active Directory (Standard Edition o Enterprise Edition) si necesita un Microsoft Active Directory en la nube de AWS que admita cargas de trabajo compatibles con Active Directory, aplicaciones y servicios de – como AWS y Amazon WorkSpaces o Amazon QuickSightcompatibilidad con LDAP para aplicaciones Linux. Utilice AD Connector si solo necesita permitir a los usuarios locales iniciar sesión en las aplicaciones y los servicios de AWS con sus credenciales de Active Directory. También puede utilizar AD Connector para unir instancias Amazon EC2 a su dominio de Active Directory existente. Utilice Simple AD si necesita un directorio a pequeña escala y bajo costo con compatibilidad básica con Active Directory que admite aplicaciones –compatibles con Samba 4, o si necesita compatibilidad con LDAP para determinadas aplicaciones.
Desarrollo aplicaciones en la nube que administran datos jerárquicos con relaciones complejas	Utilice Amazon Cloud Directory si necesita un directorio a escala de nube para compartir y controlar el acceso a datos jerárquicos entre sus aplicaciones.
Desarrollo aplicaciones SaaS	Utilice Amazon Cognito si desarrolla aplicaciones SaaS a gran escala y necesita un directorio escalable para administrar y autenticar a sus suscriptores que funcione con identidades de redes sociales.

AWS Managed Microsoft AD, AWS Directory Service for Microsoft Active

Directory funciona con un Active Directory (AD) en Microsoft Windows Server real, administrado por AWS en la nube de AWS. Le permite migrar una amplia gama de aplicaciones compatibles con Active Directory— a la nube de AWS. AWS Managed Microsoft AD funciona con Microsoft SharePoint, grupos de disponibilidad Always On de Microsoft SQL Server y muchas aplicaciones .NET. También admite aplicaciones administradas por AWS y servicios incluidos [Amazon WorkSpaces](#), [Amazon WorkDocs](#), [Amazon QuickSight](#), [Amazon Chime](#), [Amazon Connect](#) y [Amazon Relational Database Service para Microsoft SQL Server \(RDS para SQL Server\)](#).

AWS Managed Microsoft AD está aprobado para aplicaciones en la nube de AWS sujetas al cumplimiento de la HIPAA o del PCI DSS cuando se [habilita la conformidad para su directorio](#).

Todas las aplicaciones compatibles funcionan con credenciales de usuario que almacena en AWS Managed Microsoft AD o puede [conectarse a su infraestructura de AD existente con una confianza](#) y utilizar credenciales desde un Active Directory que se ejecute localmente o en Windows EC2. Si [une instancias EC2 a su AWS Managed Microsoft AD](#), los usuarios pueden obtener acceso a cargas de trabajo de Windows en la nube de AWS con la misma experiencia de inicio de sesión único (SSO) de Windows que cuando obtienen acceso a las cargas de trabajo en su red local.

AWS Managed Microsoft AD también admite casos de uso federados utilizando credenciales de Active Directory. Solo, AWS Managed Microsoft AD le permite iniciar sesión en la [Consola de administración de AWS](#). Con [Inicio de sesión único de AWS](#), también puede obtener credenciales a corto plazo para su uso con el SDK de AWS y la CLI y utilizar las integraciones de SAML preconfiguradas para iniciar sesión en muchas aplicaciones en la nube. Al añadir Azure AD Connect y, de forma opcional, Servicios de federación de Active Directory (AD FS), puede iniciar sesión en Microsoft Office 365 y otras aplicaciones en la nube con las credenciales almacenadas en AWS Managed Microsoft AD.

El servicio incluye características clave que le permiten [ampliar el esquema](#), [administrar políticas de contraseñas](#) y [habilitar las comunicaciones de LDAP](#) seguro a través de capa de conexión segura (SSL)/Transport Layer Security (TLS). También puede [habilitar la autenticación multifactor \(MFA\)](#) para AWS Managed Microsoft AD para proporcionar una capa de seguridad adicional cuando los usuarios obtienen acceso a aplicaciones de AWS desde Internet. Dado que Active Directory es un directorio LDAP, también puede utilizar AWS Managed Microsoft AD para la autenticación en Linux Secure Shell (SSH) y otras aplicaciones compatibles con LDAP. AWS ofrece monitorización, instantáneas diarias y recuperación como parte del servicio: [añade usuarios y grupos a AWS Managed Microsoft AD](#) y administra la política de grupo utilizando herramientas conocidas de Active Directory que se ejecutan en un equipo Windows unido al dominio AWS Managed Microsoft AD. También puede escalar el directorio mediante la [implementación de controladores de dominio adicionales](#) y ayudar a mejorar el desempeño de las aplicaciones distribuyendo solicitudes a través de un gran número de controladores de dominio.

AWS Managed Microsoft AD está disponible en dos ediciones: Standard y Enterprise.

- **Standard Edition:** AWS Managed Microsoft AD (Standard Edition) está optimizado para servir como directorio principal para compañías pequeñas y medianas con hasta 5 000 empleados. Le facilita suficiente capacidad de almacenamiento como para dar cabida a 30 000* objetos de directorio, como usuarios, grupos y equipos.
- **Enterprise Edition:** AWS Managed Microsoft AD (Enterprise Edition) está diseñado para su uso en grandes organizaciones y compañías con hasta 500 000* objetos de directorio.

* Los límites superiores son aproximaciones. Su directorio podría admitir más o menos objetos de directorio en función del tamaño de los mismos, y el comportamiento y las necesidades de rendimiento de sus aplicaciones.

Cuándo se debe usar

AWS Managed Microsoft AD es la mejor opción si necesita características reales de Active Directory para trabajar con aplicaciones de AWS o cargas de trabajo de Windows, incluido Amazon Relational Database Service para Microsoft SQL Server. También es la mejor opción si quiere un AD independiente en la nube de AWS compatible con Office 365 o si necesita un directorio LDAP para trabajar con sus aplicaciones Linux. Para obtener más información, consulte [AWS Managed Microsoft AD](#).

AD Connector

AD Connector es un servicio de proxy que proporciona una forma sencilla de conectar aplicaciones de AWS compatibles, como Amazon WorkSpaces, Amazon QuickSight y [Amazon EC2](#) para instancias de Windows Server, con su Microsoft Active Directory local existente. Con AD Connector, puede simplemente [añadir una cuenta de servicio](#) a su directorio Active Directory. AD Connector también elimina la necesidad de sincronización de directorios, y los costos y las dificultades que conlleva alojar una infraestructura federada.

Al añadir usuarios a aplicaciones de AWS como Amazon QuickSight, AD Connector lee el directorio Active Directory existente para crear listas de usuarios y grupos entre los que seleccionar. Cuando los usuarios inician sesión en las aplicaciones de AWS, AD Connector reenvía las solicitudes de inicio de sesión a los controladores del dominio local de Active Directory para su autenticación. AD Connector funciona con muchas aplicaciones y servicios de AWS, incluidos [Amazon WorkSpaces](#), [Amazon WorkDocs](#), [Amazon QuickSight](#), [Amazon Chime](#), [Amazon Connect](#) y [Amazon WorkMail](#). También puede [unir sus instancias EC2 de Windows](#) a su dominio local de Active Directory existente mediante AD Connector con una [unión al dominio fluida](#). AD Connector también permite a los usuarios obtener acceso a la Consola de administración de AWS y administrar los recursos de AWS iniciando sesión con las credenciales existentes de Active Directory. AD Connector no es compatible con RDS SQL Server.

También puede utilizar AD Connector para habilitar la autenticación multifactor (MFA) para los usuarios de las aplicaciones de AWS conectándolo con su infraestructura de MFA basada en RADIUS existente. Esto proporciona una capa adicional de seguridad cuando los usuarios obtienen acceso a las aplicaciones de AWS. Con AD Connector podrá seguir administrando su Active Directory como lo hace ahora. Por ejemplo, puede agregar nuevos usuarios y grupos y actualizar contraseñas con las herramientas estándar de administración de Active Directory en su Active Directory local. Así, podrá aplicar políticas de seguridad que ya existan de forma coherente, como la fecha de vencimiento de la contraseña, el historial de contraseñas y los bloqueos de cuentas, tanto si los usuarios obtienen acceso a recursos en su entorno local o en la nube de AWS.

Cuándo se debe usar

AD Connector es su mejor opción si desea utilizar su directorio local existente con los servicios de AWS compatibles. Para obtener más información, consulte [Conector de Active Directory](#).

Simple AD

Es un directorio *compatible* con Microsoft Active Directory desde AWS Directory Service y está basado en Samba 4. Simple AD admite características de Active Directory básicas, como cuentas de usuario, pertenencias a grupos, unión a un dominio Linux o a instancias EC2 basadas en Windows, inicio de sesión único (SSO) basado en Kerberos y políticas de grupo. AWS facilita las tareas de monitoreo, instantáneas diarias y recuperación como parte del servicio.

Simple AD es un directorio independiente en la nube que permite crear y administrar identidades de usuarios y administrar el acceso a las aplicaciones. Puede utilizar muchas aplicaciones y herramientas habituales –compatibles con Active Directory que requieren las características básicas de Active Directory. Simple AD es compatible con las siguientes aplicaciones de AWS: [Amazon WorkSpaces](#), [Amazon WorkDocs](#), [Amazon QuickSight](#) y [Amazon WorkMail](#). También puede iniciar sesión en la Consola de administración de AWS con cuentas de usuario de Simple AD y administrar recursos de AWS.

Simple AD no es compatible con las características siguientes: autenticación multifactor (MFA), relaciones de confianza, actualización dinámica de DNS, ampliaciones de esquemas, comunicación a través de LDAPS, cmdlets de PowerShell para AD y transferencia de roles FSMO. Simple AD no es compatible con RDS SQL Server. Los clientes que necesiten las características de un servicio Microsoft Active Directory real o que contemplen el uso de su directorio con RDS SQL Server deben utilizar AWS Managed Microsoft AD. Compruebe que las aplicaciones que necesita sean totalmente compatibles con Samba 4 antes de usar Simple AD. Para obtener más información, visite <https://www.samba.org>.

Cuándo se debe usar

Puede utilizar Simple AD como un directorio independiente en la nube para las cargas de trabajo de Windows que requieran características básicas de AD, aplicaciones compatibles de AWS o para las cargas de trabajo de Linux que necesiten un servicio LDAP. Para obtener más información, consulte [Simple Active Directory](#).

Amazon Cloud Directory

Es un directorio nativo en la nube que puede almacenar cientos de millones de objetos específicos de la aplicación con varias relaciones y esquemas. Utilice Amazon Cloud Directory si necesita un almacén de directorios muy escalable para almacenar datos jerárquicos de su aplicación.

Cuándo se debe usar

Amazon Cloud Directory es la opción ideal si necesita crear directorios de aplicaciones, por ejemplo, registros de dispositivos, catálogos, redes sociales, estructuras de la organización y topologías de red. Para obtener más información, consulte [¿Qué es Amazon Cloud Directory?](#) en la *Guía para desarrolladores de Amazon Cloud Directory*.

Amazon Cognito

Es un directorio de usuarios que añade inscripciones e inicio de sesión a su aplicación móvil o aplicación web utilizando grupos de usuarios de Amazon Cognito.

Cuándo se debe usar

También puede utilizar Amazon Cognito si necesita crear campos de registro personalizados y almacenar los metadatos en su directorio de usuarios. Este servicio totalmente administrado puede adaptarse para admitir cientos de millones de usuarios. Para obtener más información, consulte [Crear y administrar grupos de usuarios](#).



Amazon S3

S3

S3

- Amazon Simple Storage Service (Amazon S3) is an object storage service that offers industry-leading scalability, data availability, security, and performance. This means customers of all sizes and industries can use it to store and protect any amount of data for a range of use cases, such as websites, mobile applications, backup and restore, archive, enterprise applications, IoT devices, and big data analytics. Amazon S3 provides easy-to-use management features so you can organize your data and configure finely-tuned access controls to meet your specific business, organizational, and compliance requirements. Amazon S3 is designed for 99.999999999% (11 9's) of durability, and stores data for millions of applications for companies all around the world. Virtual computing environments, known as ***Ec2 instances***



Amazon S3

Scalable storage in the cloud



Amazon Glacier

Low-cost archive storage in the cloud



Amazon EBS

Persistent block storage volumes for Amazon EC2 virtual machines



Amazon EC2 Instance Storage

Temporary block storage volumes for Amazon EC2 virtual machines

	 Amazon EBS	 Amazon S3	 Amazon Glacier	 Amazon EFS
Interface	Block interface	HTTP(S)/API	API	NFSv4
Concurrency	Single instance	Many connections	Low	Thousands
Consistency	Strongly consistent	Eventually consistent	Consistent	Strongly consistent
Latency	Low latency	Higher latency	3-5 hour retrieval	Low latency
Size	16TiB/volume	5TiB per item Unlimited storage	40TiB per archive Unlimited archives	52TiB per file
Durability	0.1-0.2% AFR Volumes are AZ-specific	99.999999999%	99.999999999%	Multi-AZ storage
Use case	Databases, file servers	Static storage, high concurrency	Archival	Shared files Big data analysis

	S3 Standard	S3 Intelligent-Tiering*	S3 Standard-IA	S3 One Zone-IA†	S3 Glacier	S3 Glacier Deep Archive
Designed for durability	99.999999999% (11 9's)	99.999999999% (11 9's)	99.999999999% (11 9's)	99.999999999% (11 9's)	99.999999999% (11 9's)	99.999999999% (11 9's)
Designed for availability	99.99%	99.9%	99.9%	99.5%	99.99%	99.99%
Availability SLA	99.9%	99%	99%	99%	99.9%	99.9%
Availability Zones	≥3	≥3	≥3	1	≥3	≥3
Minimum capacity charge per object	N/A	N/A	128KB	128KB	40KB	40KB
Minimum storage duration charge	N/A	30 days	30 days	30 days	90 days	180 days
Retrieval fee	N/A	N/A	per GB retrieved	per GB retrieved	per GB retrieved	per GB retrieved
First byte latency	milliseconds	milliseconds	milliseconds	milliseconds	select minutes or hours	select hours
Storage type	Object	Object	Object	Object	Object	Object
Lifecycle transitions	Yes	Yes	Yes	Yes	Yes	Yes

■ Bucket name, type a unique DNS-compliant name for your new bucket.

Follow these naming guidelines:

- The name must be unique across all existing bucket names in Amazon S3.
- The name must not contain uppercase characters.
- The name must start with a lowercase letter or number.
- The name must be between 3 and 63 characters long.
- After you create the bucket you cannot change the name, so choose wisely.
- Choose a bucket name that reflects the objects in the bucket because the bucket name is visible in the URL that points to the objects that you're going to put in your bucket.

■ Bucket visibility

- **Public** – Everyone has access to one or more of the following: List objects, Write objects, Read and write permissions.
- **Objects can be public** – The bucket is not public, but anyone with the appropriate permissions can grant public access to objects.
- **Buckets and objects not public** – The bucket and objects do not have any public access.
- **Only authorized users of this account** – Access is isolated to IAM users and roles in this account and AWS service principals because there is a policy that grants public access.

S3 buckets			
Search for buckets		All access types	Discover the
+ Create bucket		Edit public access settings	Empty Delete
<input type="checkbox"/>	Bucket name	Access	Region Date created
<input type="checkbox"/>	alejandro.bucket	Objects can be public	US East (N. Virginia) May 27, 2011 PM GMT+02
<input type="checkbox"/>	altran.barcelona.test.jordi.s3	Public	US East (Ohio) May 29, 2011 PM GMT+02
<input type="checkbox"/>	altran.jordi	Objects can be public	EU (Paris) May 27, 2011 PM GMT+02
<input type="checkbox"/>	catiza	Objects can be public	US East (Ohio) May 29, 2011 PM GMT+02
			May 29, 2011

testluisa

[Copy Bucket ARN](#)

Properties

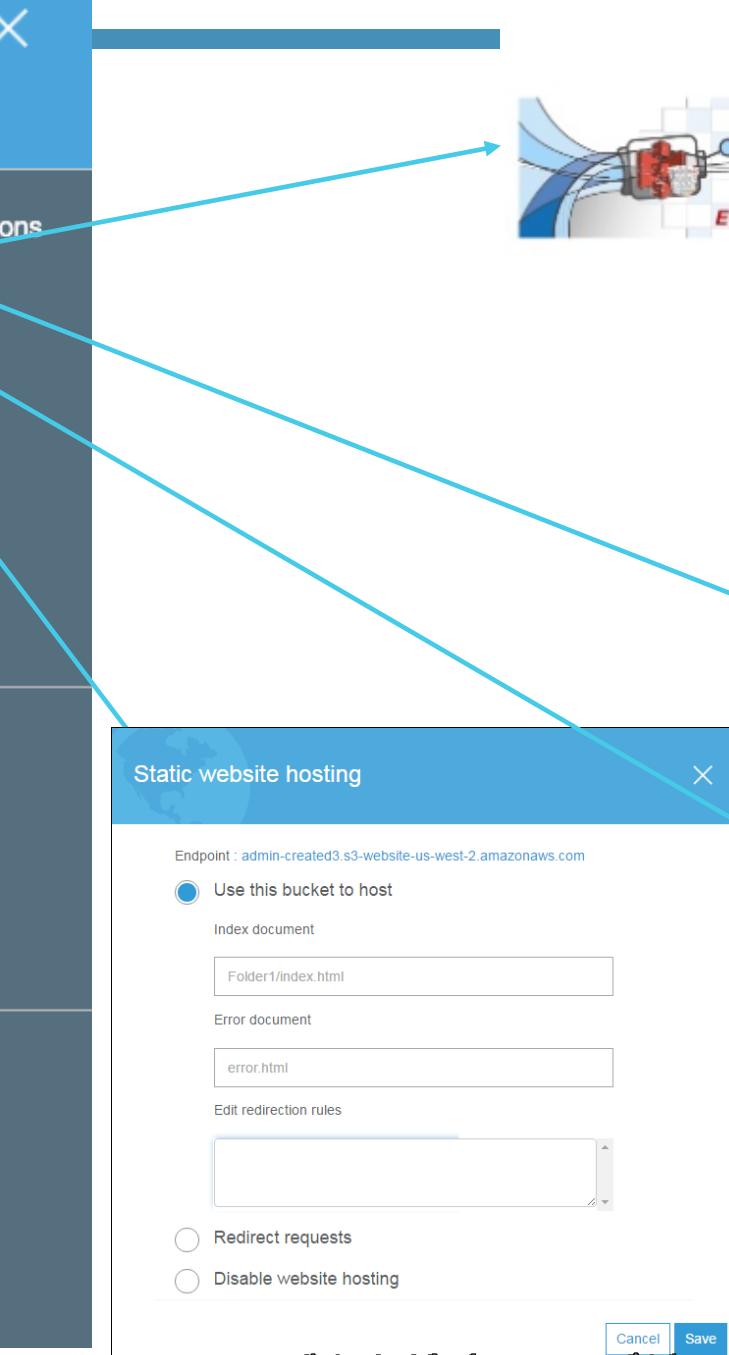
Events	0 Active notifications
Versioning	Disabled
MFA delete	Disabled
Logging	Disabled
Static web hosting	Disabled
Tags	0 Tags
Requester pays	Disabled
Object lock	Disabled
Transfer acceleration	Disabled

Permissions

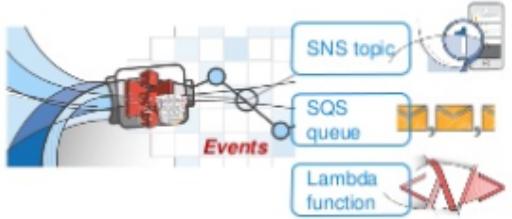
Owner	rusia.it
Block public access	Enabled
Bucket policy	No
Access control list	1 Grantees
CORS configuration	No

Management

Lifecycle	Disabled
Replication	Disabled
Analytics	Disabled
Inventory	Disabled
Metrics	Disabled



The screenshot shows the 'Properties' tab of an AWS S3 bucket named 'testluisa'. In the 'Events' section, it displays '0 Active notifications'. Below this, other settings like Versioning, MFA delete, Logging, Static web hosting, Tags, Requester pays, Object lock, and Transfer acceleration are listed as disabled. A large blue arrow points from the 'Events' section towards a diagram illustrating event notifications.



- Notification when objects are created via PUT, POST, Copy, or Multipart Upload, DELETE
- Filtering on prefixes and suffixes for all types of notifications

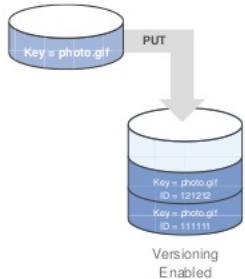
S3 versioning

Preserve, retrieve, and restore every version of every object stored in your bucket

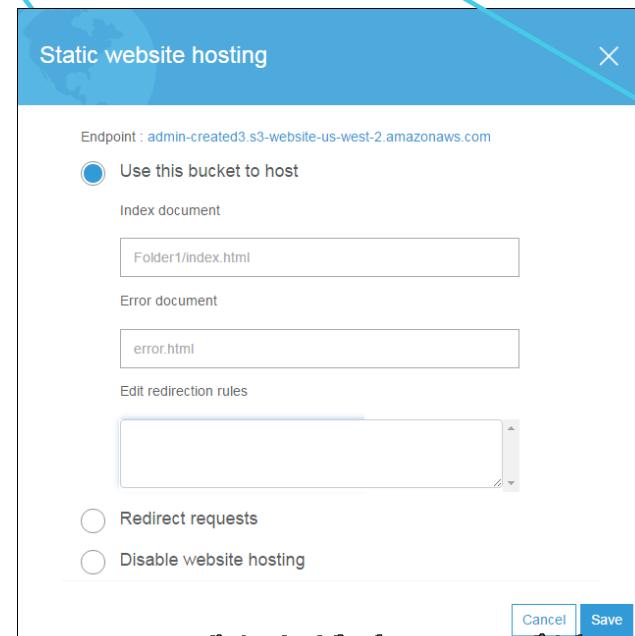
S3 automatically adds new versions and preserves deleted objects with delete markers

Easily control the number of versions kept by using lifecycle expiration policies

Easy to turn on in the AWS Management Console



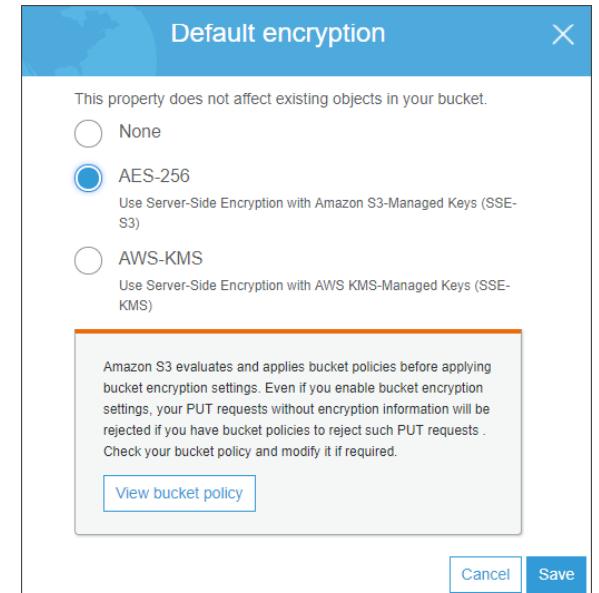
Using MFA-protected S3 buckets will enable an extra layer of protection to ensure that the S3 objects (files) cannot be accidentally or intentionally deleted by the AWS users that have access to the buckets. Note: Only the bucket owner that is logged in as AWS root account can enable MFA Delete feature and perform DELETE actions on S3 buckets.



The screenshot shows the 'Static website hosting' section of the S3 bucket properties. It includes fields for 'Index document' (set to 'Folder1/index.html'), 'Error document' (set to 'error.html'), and 'Edit redirection rules'. There are also options for 'Redirect requests' and 'Disable website hosting'. A 'Save' button is at the bottom right.

You have the following options for protecting data at rest in Amazon S3:

- **Client-Side Encryption** – Encrypt data client-side and upload the encrypted data to.
 - In this case, you manage the encryption process, the encryption keys, and related tools.
- **Server-Side Encryption** – Request Amazon S3 to encrypt your object before saving it on disks in its data centers and then decrypt it when you download the objects.
 - **Server-Side Encryption with Amazon S3-Managed Keys (SSE-S3)** – Each object is encrypted with a unique key. As an additional safeguard, it encrypts the key itself with a master key that it regularly rotates.
 - **Use Server-Side Encryption with Keys Stored in AWS KMS (SSE-KMS)** – Similar to SSE-S3. There are separate permissions for the use of an envelope key (that is, a key that protects your data's encryption key) that provides added protection against unauthorized. Additionally, you have the option to create and manage encryption keys yourself.
 - **Use Server-Side Encryption with Customer-Provided Keys (SSE-C)** – You manage the encryption keys and Amazon S3 manages the encryption, as it writes to disks, and decryption, when you access your objects.



Log Properties	AWS CloudTrail	Amazon S3 Server Logs
Can be forwarded to other systems (CloudWatch Logs, CloudWatch Events)	Yes	
Deliver logs to more than one destination (for example, send the same logs to two different buckets)	Yes	
Turn on logs for a subset of objects (prefix)	Yes	
Cross-account log delivery (target and source bucket owned by different accounts)	Yes	
Integrity validation of log file using digital signature/hashing	Yes	
Default/choice of encryption for log files	Yes	
Object operations (using Amazon S3 APIs)	Yes	Yes
Bucket operations (using Amazon S3 APIs)	Yes	Yes
Searchable UI for logs	Yes	
Fields for object lock parameters, Amazon S3 select properties for log records	Yes	
Fields for Object Size, Total Time, Turn-Around Time, and HTTP Referrer for log records		Yes
Lifecycle transitions, expirations, restores		Yes
Logging of keys in a batch delete operation		Yes
Authentication failures ¹		Yes
Accounts where logs get delivered	Bucket owner ² , and requester	Bucket owner only

Performance and Cost	AWS CloudTrail	Amazon S3 Server Logs
Price	Management events (first delivery) are free; data events incur a fee, in addition to storage of logs	No additional cost in addition to storage of logs
Speed of log delivery	Data events every 5 mins; management events every 15 mins	Within a few hours
Log format	JSON	Log file with space-separated, newline-delimited records

IAM policies vs. S3 bucket policies

- IAM policies specify what actions are allowed or denied on what AWS resources (e.g. allow ec2:TerminateInstance on the EC2 instance with instance_id=i-8b3620ec). You attach IAM policies to IAM users, groups, or roles, which are then subject to the permissions you've defined. In other words, IAM policies define what a principal can do in your AWS environment.
- S3 bucket policies, on the other hand, are attached only to S3 buckets. S3 bucket policies specify what actions are allowed or denied for which principals on the bucket that the bucket policy is attached to (e.g. allow user Alice to PUT but not DELETE objects in the bucket). S3 bucket policies are a type of access control list, or ACL (here I mean "ACL" in the generic sense, not to be confused with S3 ACLs, which is a separate S3 feature discussed later in this post).

Sample S3 Bucket Policy

This S3 bucket policy enables the root account 111122223333 and the IAM user Alice under that account to perform any S3 operation on the bucket named "my_bucket", as well as that bucket's contents.

```
{  
  "Version": "2012-10-17",  
  "Statement": [  
    {  
      "Effect": "Allow",  
      "Principal": {  
        "AWS": ["arn:aws:iam::111122223333:user/Alice",  
               "arn:aws:iam::111122223333:root"]  
      },  
      "Action": "s3:*",  
      "Resource": ["arn:aws:s3:::my_bucket",  
                  "arn:aws:s3:::my_bucket/*"]  
    }  
  ]  
}
```

Sample IAM Policy

This IAM policy grants the IAM entity (user, group, or role) it is attached to permission to perform any S3 operation on the bucket named "my_bucket", as well as that bucket's contents.

```
{  
  "Version": "2012-10-17",  
  "Statement": [  
    {  
      "Effect": "Allow",  
      "Action": "s3:*",  
      "Resource": ["arn:aws:s3:::my_bucket",  
                  "arn:aws:s3:::my_bucket/*"]  
    }  
  ]  
}
```

Note that the S3 bucket policy includes a "Principal" element, which lists the principals that bucket policy controls access for. The "Principal" element is unnecessary in an IAM policy, because the principal is by default the entity that the IAM policy is attached to.

When to use IAM policies vs. S3 policies

Use IAM policies if:

- You need to control access to AWS services other than S3. IAM policies will be easier to manage since you can centrally manage all of your permissions in IAM, instead of spreading them between IAM and S3.
- You have numerous S3 buckets each with different permissions requirements. IAM policies will be easier to manage since you don't have to define a large number of S3 bucket policies and can instead rely on fewer, more detailed IAM policies.
- You prefer to keep access control policies in the IAM environment.
- "What can this user do in AWS?" then IAM policies are probably the way to go.
- IAM policies specify what actions are allowed or denied on what AWS. You attach IAM policies to IAM users, groups, or roles, which are then subject to the permissions you've defined. In other words, IAM policies define what a principal can do in your AWS environment.

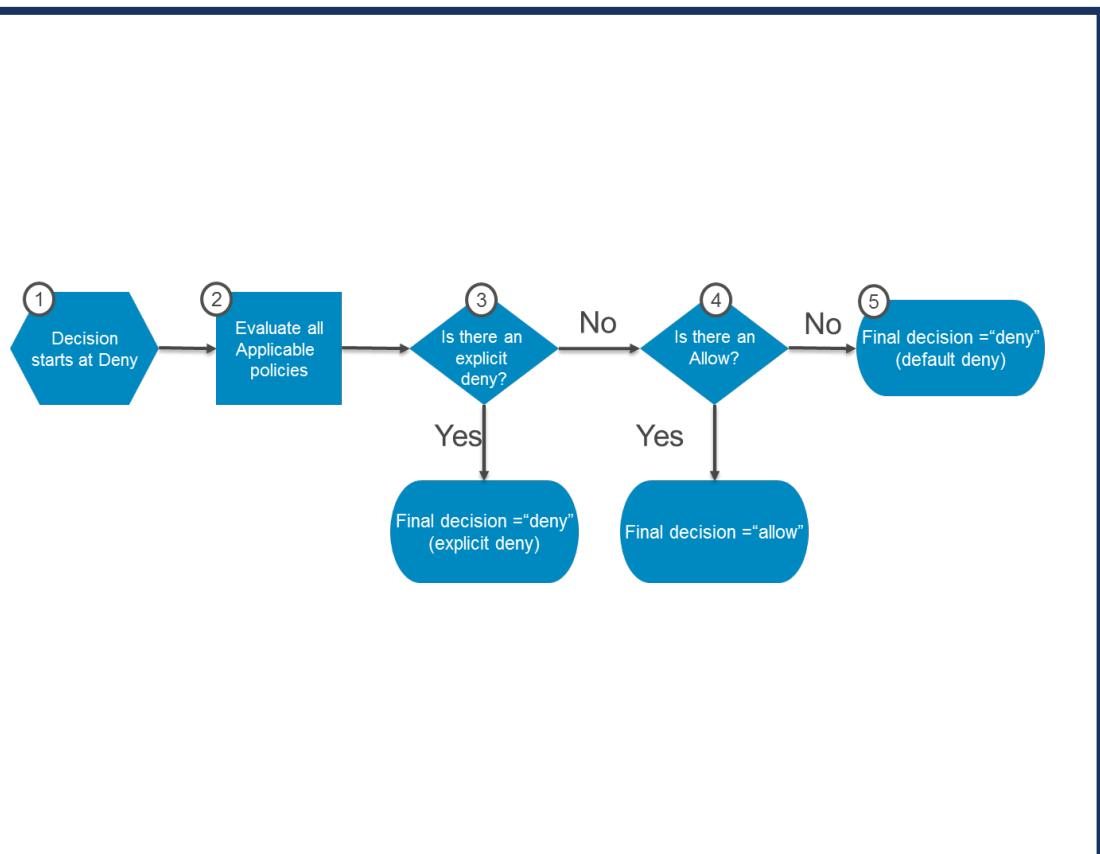
■ Use S3 bucket policies if:

- You want a simple way to grant cross-account access to your S3 environment, without using IAM roles.
- Your IAM policies bump up against the size limit (up to 2 kb for users, 5 kb for groups, and 10 kb for roles). S3 supports bucket policies of up 20 kb.
- You prefer to keep access control policies in the S3 environment.
- "Who can access this S3 bucket?" then S3 bucket policies will likely suit you better.
- S3 bucket policies, on the other hand, are attached only to S3 buckets. S3 bucket policies specify what actions are allowed or denied for which principals on the bucket that the bucket policy is attached to. S3 bucket policies are a type of access control list, (here I mean "ACL" in the generic sense, not to be confused with S3 ACLs, which is a separate S3 feature discussed later).

What about S3 ACLs?

- As a general rule, AWS recommends using S3 bucket policies or IAM policies for access control. S3 ACLs is a legacy access control mechanism that predates IAM. However, if you already use S3 ACLs and you find them sufficient, there is no need to change.
- An S3 ACL is a sub-resource that's attached to every S3 bucket and object. It defines which AWS accounts or groups are granted access and the type of access. When you create a bucket or an object, Amazon S3 creates a default ACL that grants the resource owner full control over the resource.
- You can attach S3 ACLs to individual objects within a bucket to manage permissions for those objects. S3 bucket policies and IAM policies define object-level permissions by providing those objects in the Resource element in your policy statements. The statement will apply to those objects in the bucket. Consolidating object-specific permissions into one policy (as opposed to multiple S3 ACLs) makes it simpler for you to determine effective permissions for your users and roles.

EC2 HOW DOES AUTHORIZATION WORK WITH MULTIPLE ACCESS CONTROL MECHANISMS?



- In accordance with the principle of least-privilege, decisions default to DENY and an explicit DENY always trumps an ALLOW.
- For example, if an IAM policy grants access to an object, the S3 bucket policies denies access to that object, and there is no S3 ACL, then access will be denied.
- Similarly, if no method specifies an ALLOW, then the request will be denied by default.
- Only if no method specifies a DENY and one or more methods specify an ALLOW will the request be allowed.

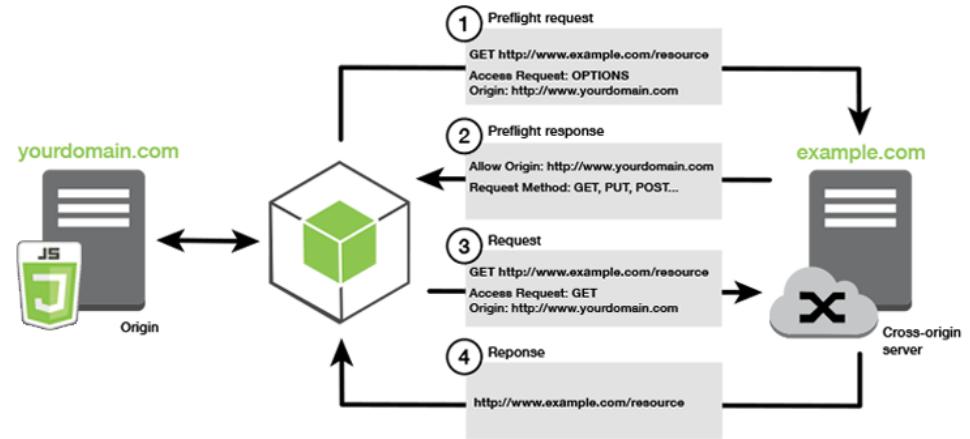
CORS



El uso compartido de recursos entre orígenes (CORS) define una manera para que las aplicaciones web en un dominio interactúen con los recursos de un dominio diferente. CORS es un factor cuando se desarrollan aplicaciones de navegador con AWS SDK para JavaScript, ya que la mayoría de las solicitudes a recursos se envían a un dominio externo como, por ejemplo, el punto de enlace de un servicio web.



Si el dominio de solicitud o el tipo de solicitud HTTP no está autorizado, se denegará la solicitud. Sin embargo, con CORS es posible realizar una solicitud preliminar antes de enviarla realmente. En dicho caso, se realiza una solicitud preliminar en la que se envía la operación de solicitud de acceso OPTIONS. Si la configuración CORS del servidor de origen otorga acceso al dominio que realiza la solicitud, el servidor enviará una respuesta preliminar que contenga una lista de todos los tipos de solicitud HTTP que el dominio que realiza la solicitud puede hacer al recurso solicitado.



En el caso más sencillo, el navegador script realiza una solicitud GET para obtener un recurso de un servidor que se encuentra en otro dominio. En función de cómo sea la configuración de CORS de dicho servidor, si la solicitud proviene de un dominio con permiso para enviar solicitudes GET, el servidor de orígenes cruzados responderá devolviendo el recurso que se ha solicitado.

Es preciso configurar CORS en los buckets de Amazon S3 para poder realizar operaciones en ellos. Puede que en algunos entornos JavaScript no se aplique CORS y, por consiguiente, no sea necesario configurarlo. Por ejemplo, si aloja su aplicación de un bucket Amazon S3 y obtiene acceso a recursos de *.s3.amazonaws.com o algún otro punto de enlace específico, sus solicitudes no tendrán acceso a un dominio externo. Por lo tanto, esta configuración no necesita CORS. En este caso, se seguirá utilizando CORS para servicios que no sean Amazon S3.

[Access Control List](#)[Bucket Policy](#)[CORS configuration](#)

CORS configuration editor ARN: arn:aws:s3:::cardslots

Add a new cors configuration or edit an existing one in the text area below.

```
1 <!-- Sample policy -->
2 <CORSConfiguration>
3   <CORSRule>
4     <AllowedOrigin>*</AllowedOrigin>
5     <AllowedMethod>GET</AllowedMethod>
6     <MaxAgeSeconds>3000</MaxAgeSeconds>
7     <AllowedHeader>Authorization</AllowedHeader>
8   </CORSRule>
9 </CORSConfiguration>
```

Una configuración de CORS es un archivo XML que contiene una serie de reglas dentro de una `<CORSRule>`. Una configuración puede tener hasta 100 reglas. Una regla se define con una de las siguientes etiquetas:

- `<AllowedOrigin>`, que especifica orígenes de dominios a los que permite realizar solicitudes de dominio cruzado.
- `<AllowedMethod>`, que especifica un tipo de solicitud que permite (GET, PUT, POST, DELETE, HEAD) en solicitudes de dominio cruzado.
- `<AllowedHeader>`, que especifica los encabezados que están permitidos en una solicitud preliminar.

Ejemplo de configuración de CORS

La siguiente muestra de configuración de CORS permite a un usuario ver, añadir, eliminar o actualizar objetos dentro de un bucket desde el dominio `example.org`, aunque se recomienda que defina el ámbito de `<AllowedOrigin>` al dominio de su sitio web. Puede especificar "*" para permitir cualquier origen.

```
<?xml version="1.0" encoding="UTF-8"?>
<CORSConfiguration xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
  <CORSRule>
    <AllowedOrigin>https://example.org</AllowedOrigin>
    <AllowedMethod>HEAD</AllowedMethod>
    <AllowedMethod>GET</AllowedMethod>
    <AllowedMethod>PUT</AllowedMethod>
    <AllowedMethod>POST</AllowedMethod>
    <AllowedMethod>DELETE</AllowedMethod>
    <AllowedHeader>*</AllowedHeader>
    <ExposeHeader>ETag</ExposeHeader>
    <ExposeHeader>x-amz-meta-custom-header</ExposeHeader>
  </CORSRule>
</CORSConfiguration>
```

Esta configuración no autoriza al usuario a realizar acciones en el bucket. Habilita al modelo de seguridad del navegador para que permita una solicitud a Amazon S3. Los permisos tienen que configurarse a través de permisos de bucket o permisos de rol de IAM.

Puede utilizar `ExposeHeader` para permitir que el SDK lea los encabezados de respuestas que Amazon S3 devuelve. Por ejemplo, si desea leer el encabezado `ETag` de una operación `PUT` o una carga multipart, debe incluir la etiqueta `ExposeHeader` en su configuración, tal y como se muestra en el ejemplo anterior. El SDK solo puede obtener acceso a los encabezados que se exponen a través de la configuración de CORS. Si establece metadatos en el objeto, los valores se devuelven como encabezados con el prefijo `x-amz-meta-`, como `x-amz-meta-my-custom-header`, y también deben exponerse de la misma manera.



Copy Bucket ARN

Properties

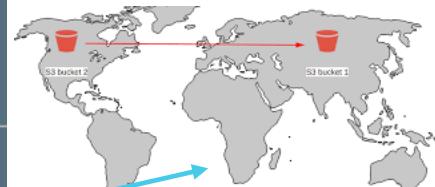
Events	0 Active notifications
Versioning	Disabled
MFA delete	Disabled
Logging	Disabled
Static web hosting	Disabled
Tags	0 Tags
Requester pays	Disabled
Object lock	Disabled
Transfer acceleration	Disabled

Permissions

Owner	rusia.it
Block public access	Enabled
Bucket policy	No
Access control list	1 Grantees
CORS configuration	No

Management

Lifecycle	Disabled
Replication	Disabled
Analytics	Disabled
Inventory	Disabled
Metrics	Disabled



Lifecycle Rules

Step 1: Choose Rule Target
Step 2: Configure Rule
Step 3: Review and Name

period.
Choose different options below to see what works best for your use case. No rule will take effect until you activate them at the end of this wizard.

Action on Objects

Transition to the Standard - Infrequent Access Storage Class Days after the object's creation date
Standard - Infrequent Access has a 30-day minimum retention period and a 128KB minimum object size. Lifecycle policy will not transition objects that are less than 128KB. Refer here to learn more about Standard - Infrequent Access.

Archive to the Glacier Storage Class Days after the object's creation date
This rule could reduce your storage costs. Refer here to learn more about Glacier pricing. Note that objects archived to the Glacier Storage Class are not immediately accessible.

Permanently Delete 14 Days after the object's creation date

EXAMPLE:

January 26 2016 Day 0	Object Uploaded	>	February 9 2016 Day 14	Rule: Expire	Object Deleted
--------------------------	-----------------	---	---------------------------	--------------	----------------

[Cancel](#) [Set Target](#) [Review >](#)

La *replicación entre regiones* (CRR) permite copiar de manera automática y asincrónica los objetos en buckets de diferentes regiones de AWS. Los buckets configurados para replicación entre regiones pueden pertenecer a la misma cuenta de AWS o a cuentas diferentes.

La replicación entre regiones se habilita con una configuración a nivel de bucket. Añada la configuración de replicación al bucket de origen la configuración mínima:

- El bucket de destino donde desea que Amazon S3 replique los objetos
- En AWS IAM una función de que Amazon S3 pueda asumir para replicar objetos en su nombre

Create S3 Bucket

1. Navigate to the S3 portion of the AWS Management Console.
2. Create a bucket, choosing a globally unique name.
3. Select US East (N. Virginia) region.
4. Click **Next**.
5. Leave options as defaults; click **Next**.
6. Under permissions, uncheck all four permissions restrictions.
7. Click **Next**.
8. Click **Create bucket**.
9. Select bucket name.
10. Click **Upload**.
11. Add files (use your own or those from the sample website).
12. Click **Upload**.

Enable Static Website Hosting

1. Click the bucket name.
2. Navigate to **Properties > Static website hosting**.
3. Select **Use this bucket to host a website**.
4. For **Index document**, enter `index.html`.
5. For **Error document**, enter `error.html`.
6. Click **Save**.

Apply Bucket Policy



1. Navigate to **Permissions > Bucket policy**.
2. Add the following JSON statement (replacing `<my-bucket>` with your bucket name):

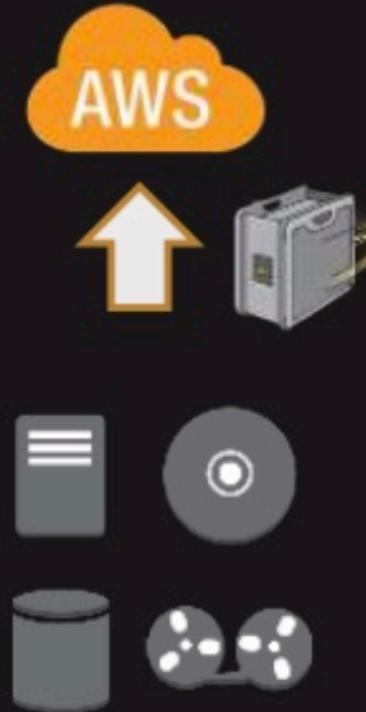
```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Sid": "PublicReadGetObject",  
            "Effect": "Allow",  
            "Principal": "*",  
            "Action": ["s3:GetObject"],  
            "Resource": [ "arn:aws:s3:::<my-bucket>/*" ]  
        }]  
}
```

Ensure the trailing `/*` is present so that the policy applies to all objects within the bucket.



SNOWBALL

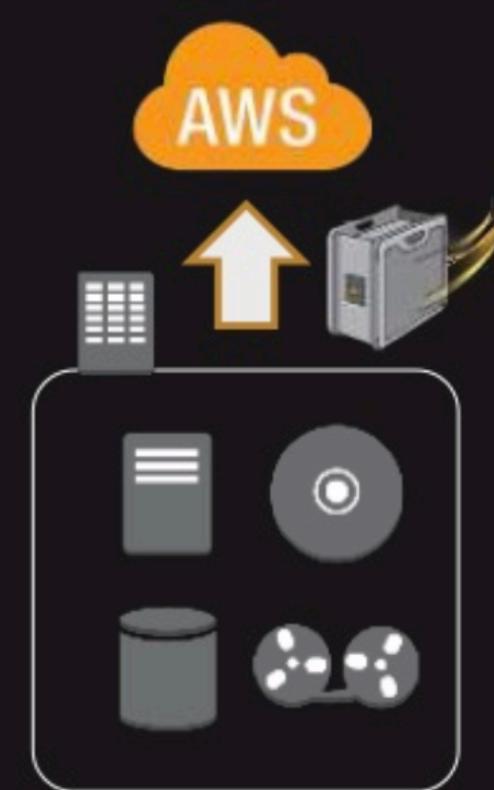
When to use AWS Snowball



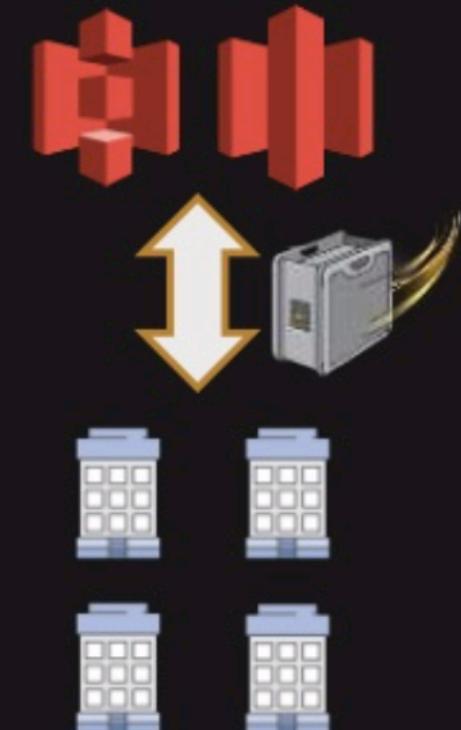
Cloud
Migration



Disaster
Recovery



Datacenter
Decommission



Content
Distribution

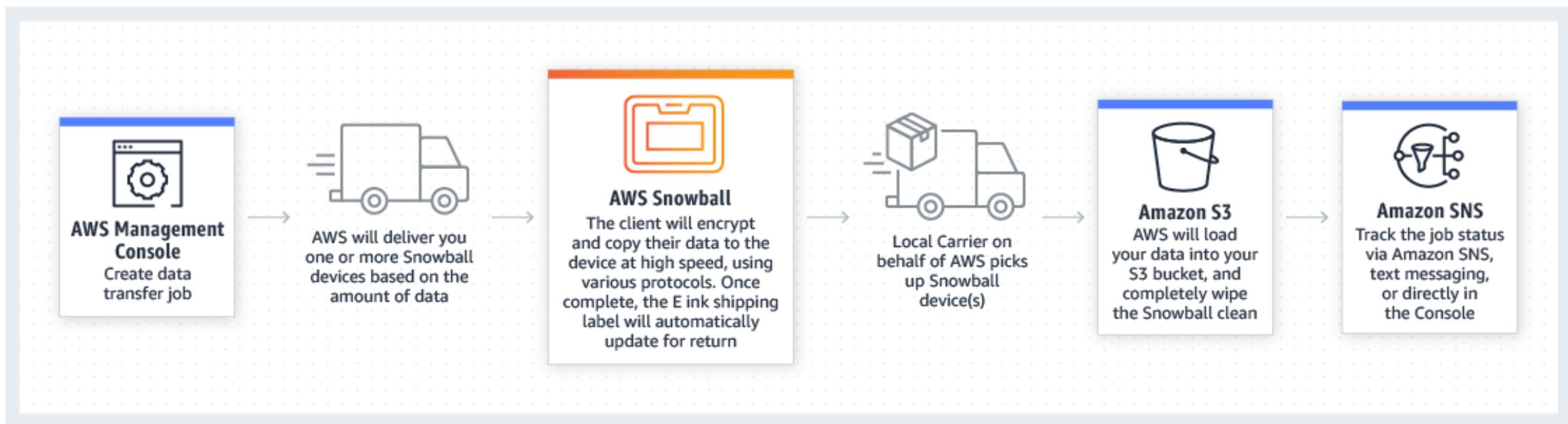
SNOWBALL

Velocidad (MB/s)	Tiempo de transferencia de 42 TB	Tiempo de transferencia de 72 TB
800	14 horas	1 día
450	1.09 días	1.8 días
400	1.16 días	2.03 días
300	1.54 días	2.7 días
277	1.67 días	2.92 días
200	2.31 días	4 días
100	4.63 días	8.10 días
60	8 días	13 días
30	15 días	27 días
10	46 días	81 días

Conección a Internet disponible	Mínimo teórico Días para transferir 100 TB con una utilización de la red del 80%	¿Cuándo considerar el uso de AWS Snowball?
T3 (44,736 Mbps)	269 días	2 TB o más
100 Mbps	120 días	5 TB o más
1000 Mbps	12 días	60 TB o más

- En la siguiente tabla se muestra cómo la velocidad de transferencia de la red afecta al tiempo que se tarda en llenar un Snowball con datos. La transferencia de archivos más pequeños sin agruparlos en archivos mayores reduce la velocidad de transferencia debido a una mayor sobrecarga.

SNOWBALL



AWS Snow Family



Snowball

Petabyte-scale
data migration



Snowball Edge

Compute & Storage for
Hybrid/Edge workloads



Snowmobile

Exabyte-scale
data migration

Summary

			
	Snowball	Snowball Edge	Snowmobile
Typical scenario	Data Migration	Data Migration or Local compute	Data Migration
Storage	50TB, 80TB	100TB	100PB
Compute	N/A	Comparable to m4.4xlarge	N/A
AWS Services available	S3	Lambda, File Gateway, S3	N/A
NFS Endpoint	N/A	Yes	Yes
Encryption	Yes	Yes	Yes
Clustering for local use	N/A	Available	N/A
Typical job lifetime	Days - weeks	Data Migration – Days-Weeks Local compute – Weeks - Years	Weeks – months
HIPAA	Yes	Roadmap	No

STORAGE GATEWAY

Storage Gateway family



File Gateway

Store and access objects
in Amazon S3 from
file-based applications
with local caching

Windows & Linux apps.
using Amazon S3



Volume Gateway

Block storage on-premises
backed by cloud storage with
local caching, Amazon Elastic
Block Store (Amazon EBS)
snapshots, and clones,
integrated with AWS Backup

SAN-like
w/ cloud recovery



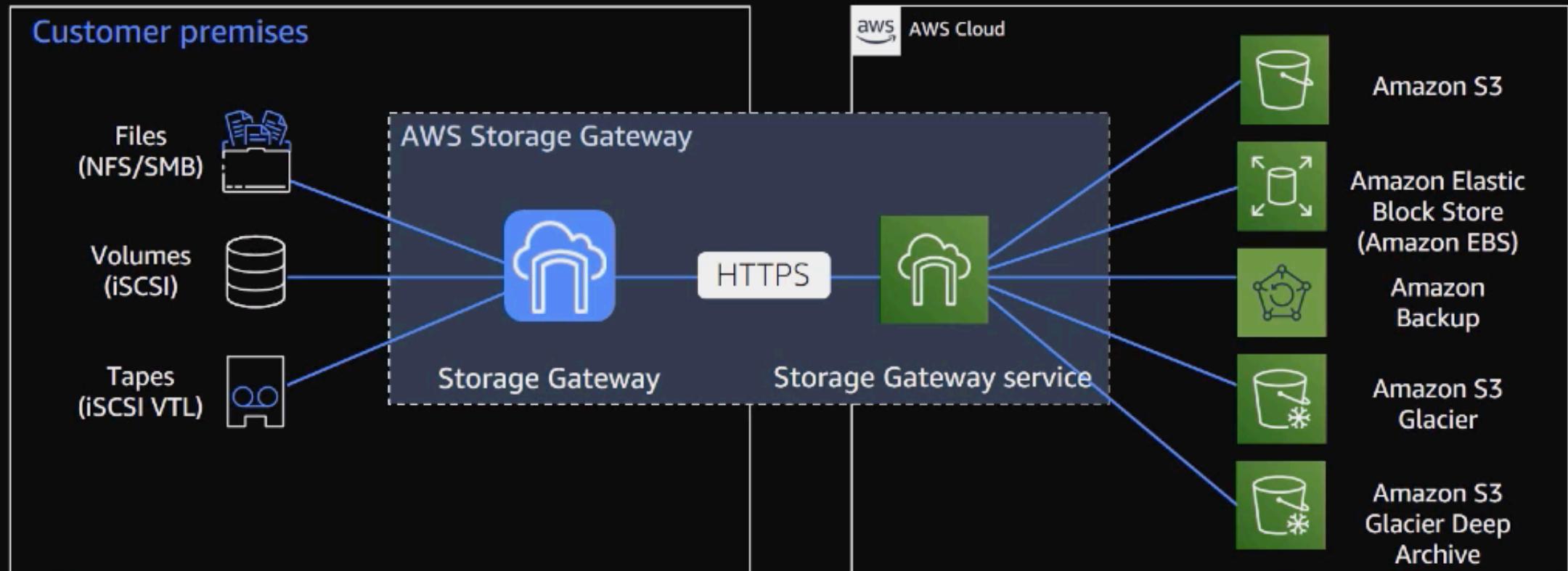
Tape Gateway

Drop-in replacement for
physical tape infrastructure
backed by cloud storage with
local caching

Easily switch tape
backups to AWS

AWS Storage Gateway

Provides on-premises access to virtually unlimited cloud storage



Configuration: VMware, Hyper-V,
Amazon Elastic Compute Cloud (Amazon EC2),
Hardware appliance

Integrated with AWS Identity and Access Management
(IAM), AWS Key Management Service (AWS KMS),
AWS CloudTrail, Amazon CloudWatch services

Gateway Type	Cache (Minimum)	Cache (Maximum)	Upload Buffer (Minimum)	Upload Buffer (Maximum)	Other Required Local Disks
File gateway	150 GiB	16 TiB	—	—	—
Cached volume gateway	150 GiB	16 TiB	150 GiB	2 TiB	—
Stored volume gateway	—	—	150 GiB	2 TiB	1 or more for stored volume or volumes
Tape gateway	150 GiB	16 TiB	150 GiB	2 TiB	—

Requisitos de hardware para las máquinas virtuales locales

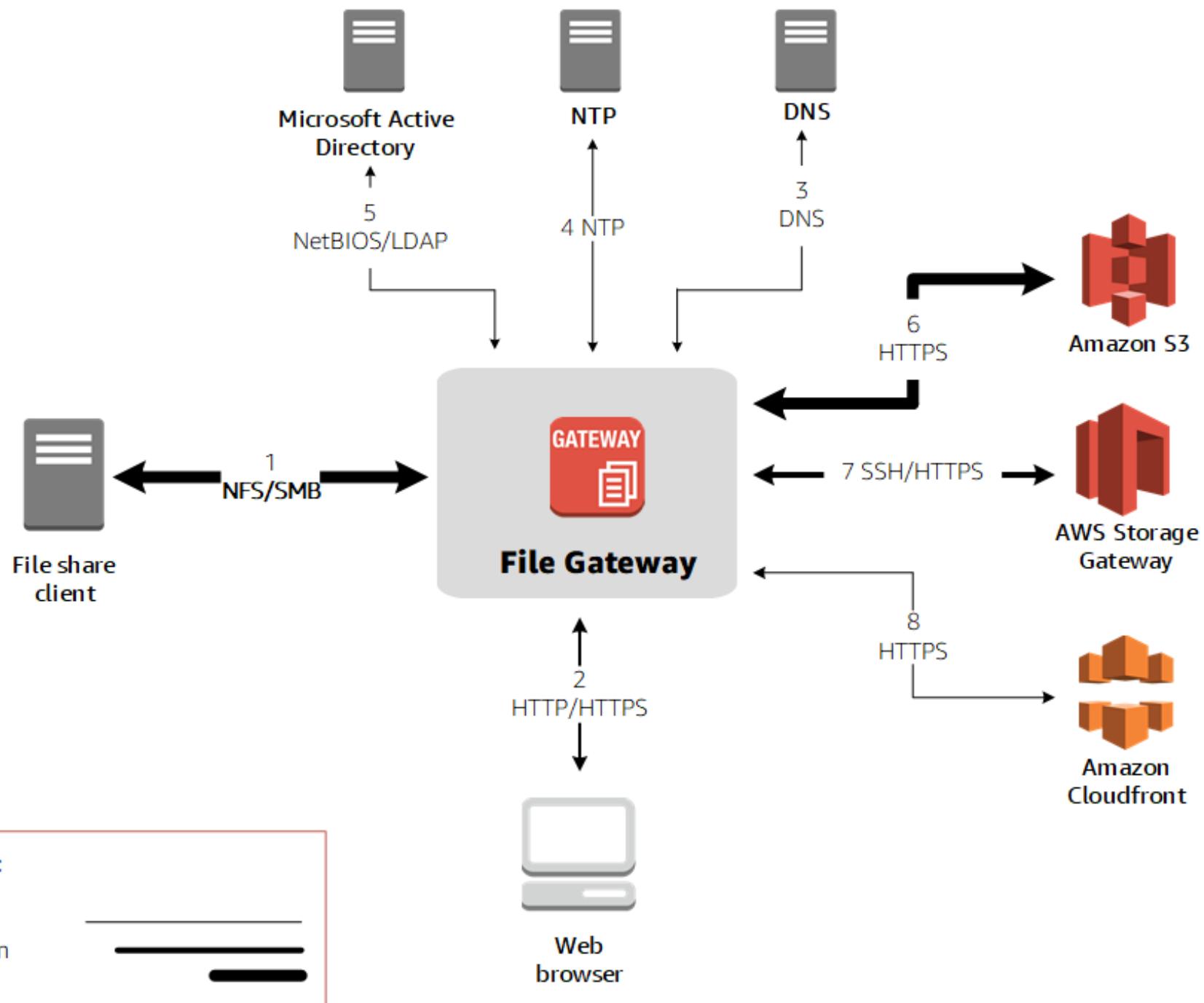
Cuando implemente la gateway localmente, debe asegurarse de que el hardware subyacente en el que esté implementando la máquina virtual de la gateway pueda dedicar los siguientes recursos mínimos:

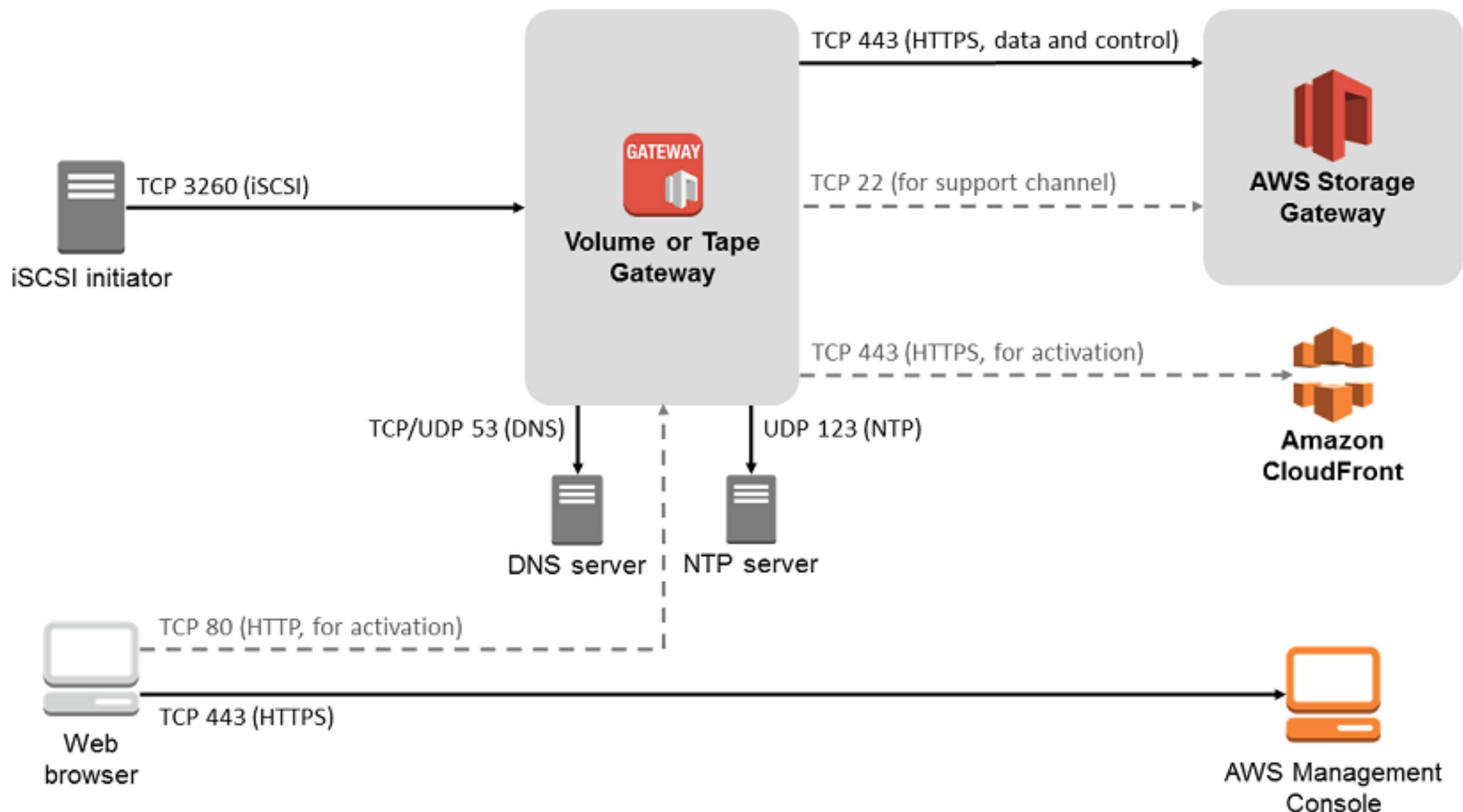
- Cuatro procesadores virtuales asignados a la MV.
- 16 GiB de RAM reservada asignados a la máquina virtual.
- 80 GiB de espacio de disco para la instalación de los datos del sistema y la imagen de la máquina virtual.

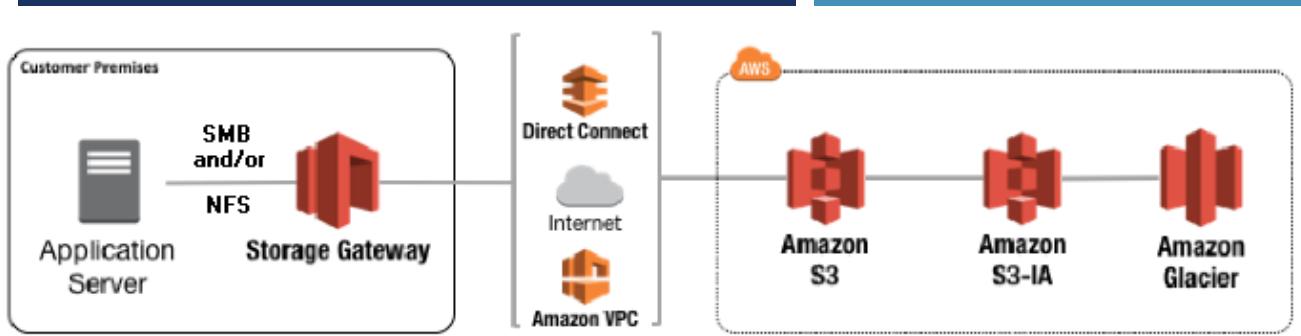
Requisitos para los tipos de instancias Amazon EC2

Cuando implemente la gateway en Amazon EC2, el tamaño de la instancia debe ser al menos `xlarge` para que la gateway funcione. Familia de instancias de uso general: tipos de instancias `m4` o `m5`.

- Familia de instancias optimizadas para computación: tipos de instancias `c4` o `c5`. Seleccione el tamaño de instancia `2xlarge` o superior para satisfacer los requisitos de RAM necesarios.
- Familia de instancias optimizadas para memoria: tipos de instancias `r3`.
- Familia de instancias optimizadas para almacenamiento: tipos de instancias `i3`.





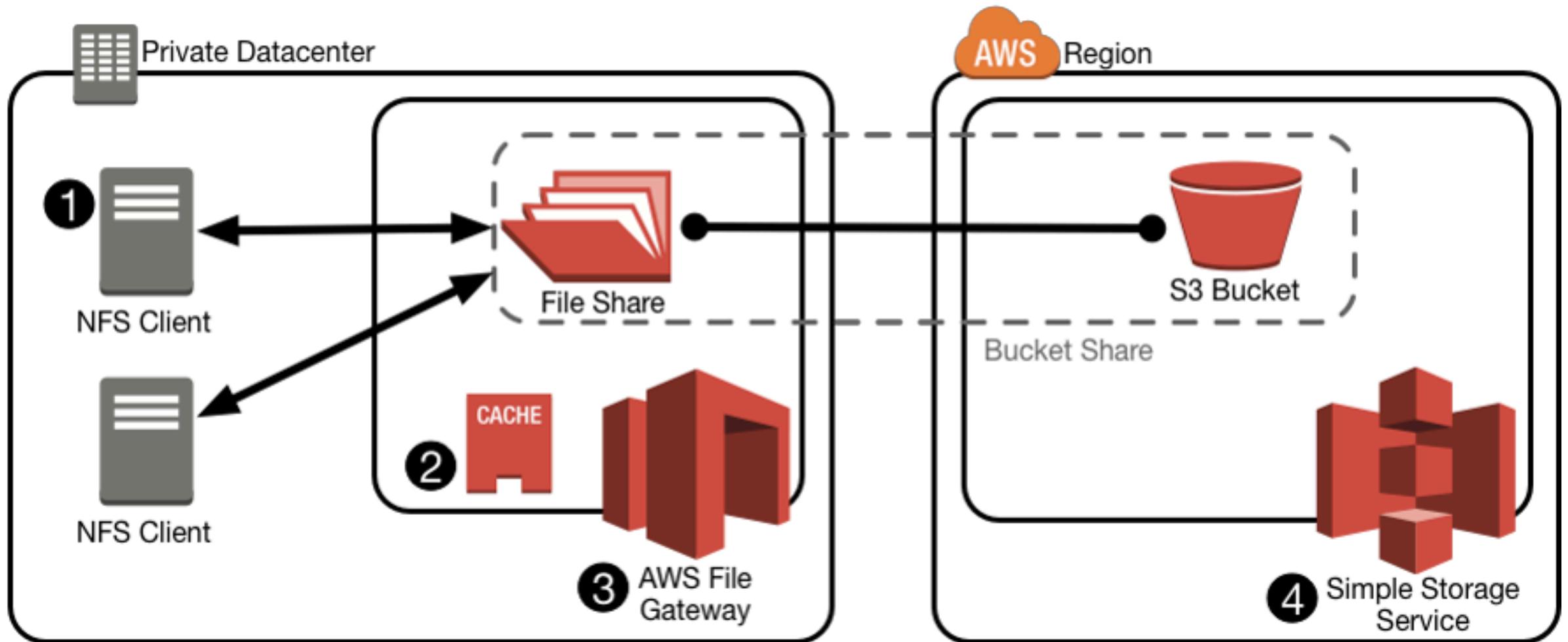


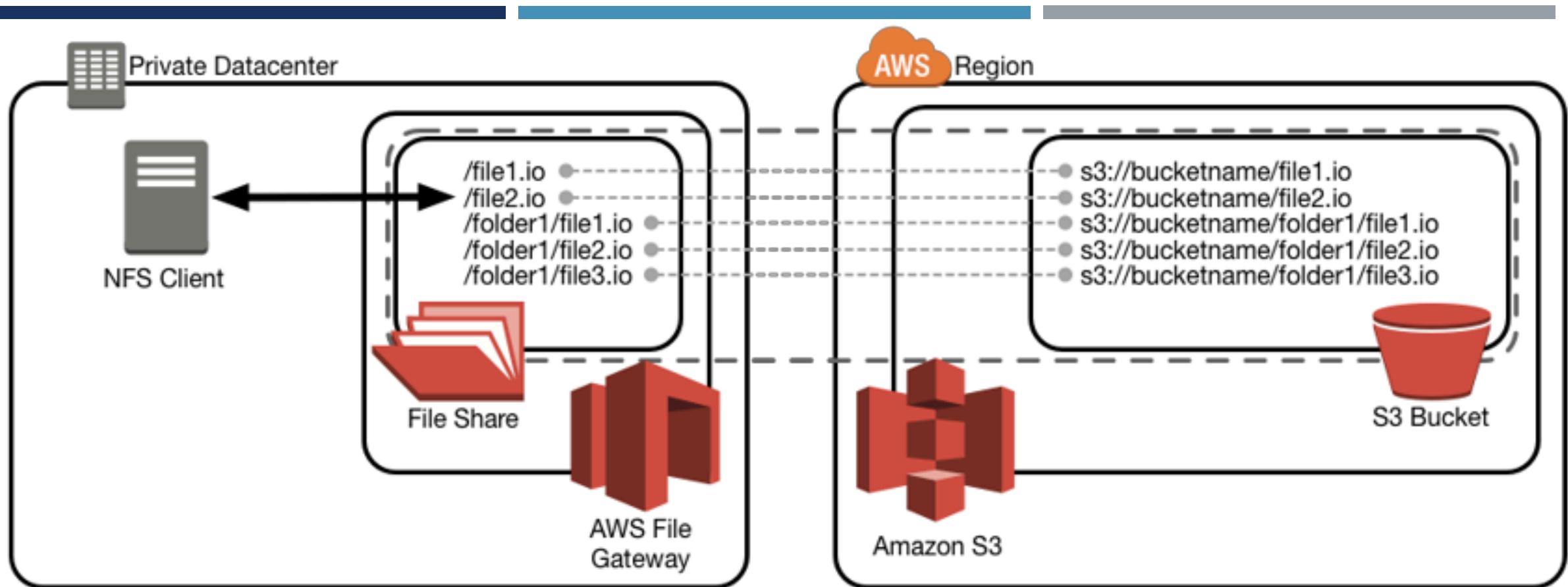
Gateway de archivos

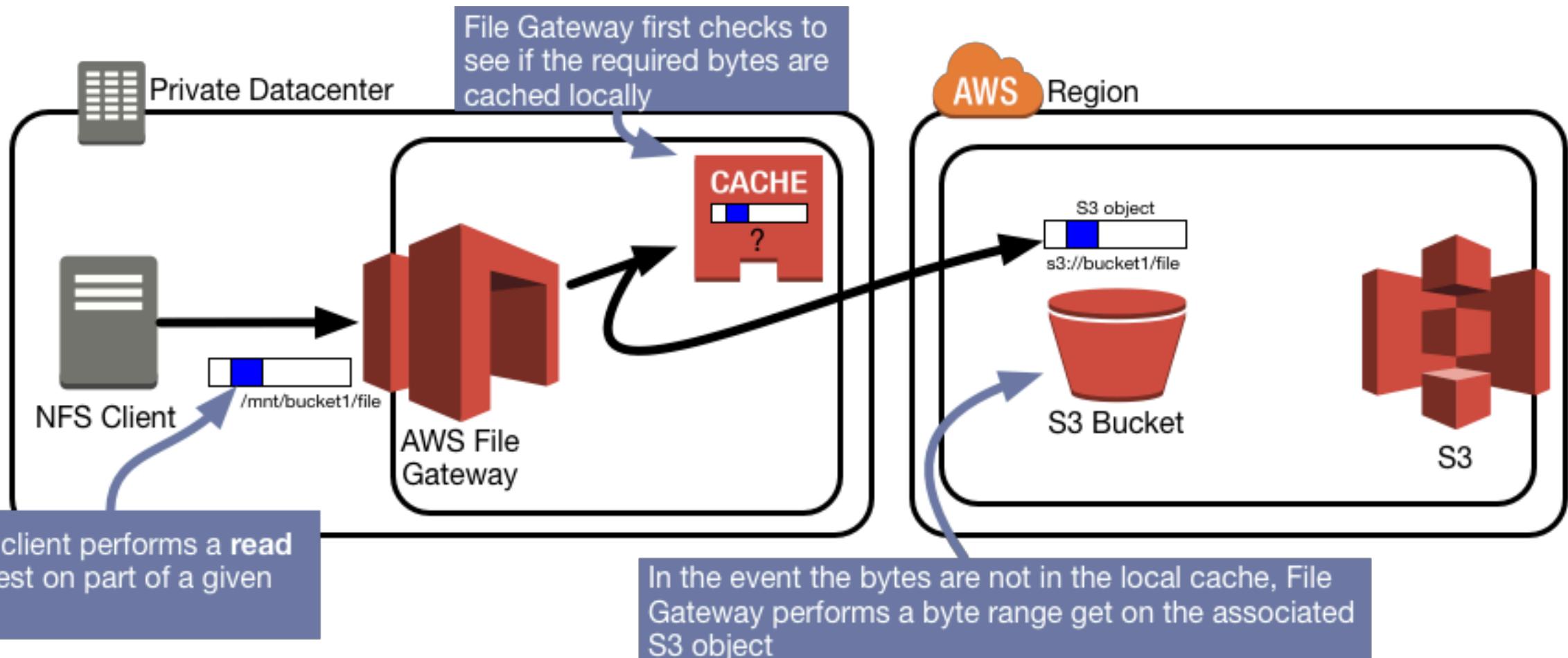
Para utilizar un file gateway, comience por descargar una imagen de máquina virtual de la gateway de archivos. A continuación, active la gateway de archivos desde la Consola de administración de AWS o a través de la API de Storage Gateway. También puede crear una gateway de archivos usando una imagen de Amazon EC2.

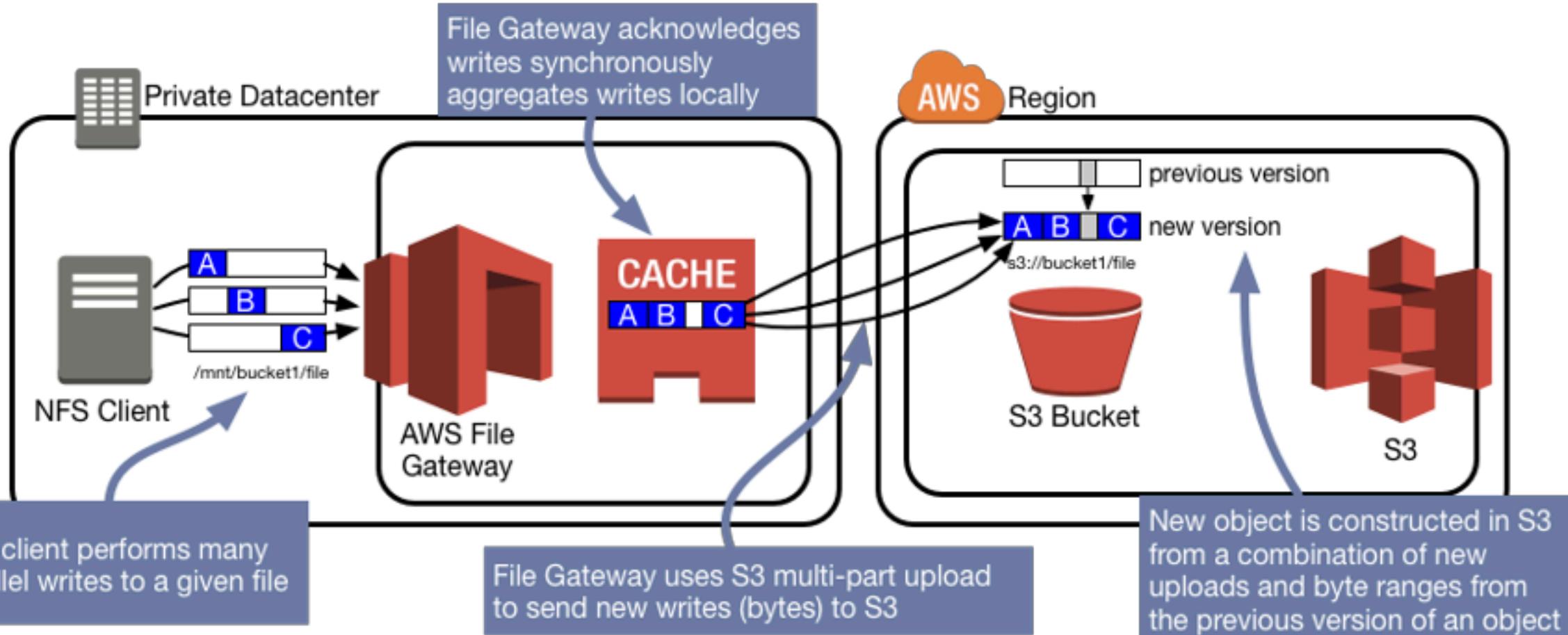
Una vez que la gateway de archivos esté activada, debe crear y configurar el recurso compartido de archivos y asociarlo al bucket de Amazon S3. Esto hace que el recurso compartido de archivos esté al alcance de los clientes que utilizan el protocolo NFS o SMB. Los archivos que se escriben en un recurso compartido de archivos se convierten en objetos en Amazon S3, con la ruta como clave. Existe una correlación de uno a uno entre los archivos y los objetos, y la gateway actualiza de forma asíncrona los objetos de Amazon S3 cuando se realizan cambios en los archivos. Los objetos existentes en el bucket aparecen como archivos en el sistema de archivos y la clave se convierte en la ruta. Los objetos se cifran con las claves de cifrado del lado del servidor de Amazon S3 (SSE-S3). Todas las transferencias de datos se realizan a través de HTTPS.

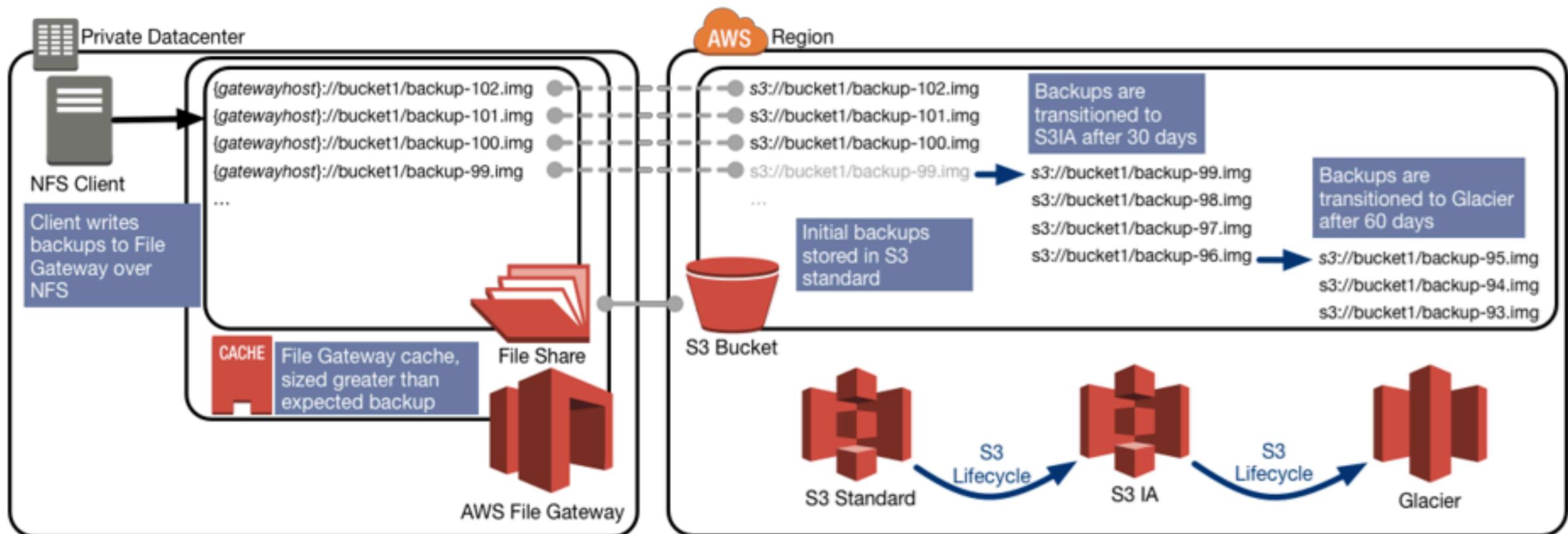
El servicio optimiza la transferencia de datos entre la gateway y AWS mediante cargas paralelas de varias partes o descargas de rango de bytes, para utilizar mejor el ancho de banda disponible. Se mantiene una caché local para proporcionar acceso de baja latencia a los datos a los que se ha tenido acceso recientemente y reducir los cargos por salida de datos. Las métricas de CloudWatch muestran información sobre el uso de los recursos de la máquina virtual y la transferencia de datos a y desde AWS. CloudTrail rastrea todas las llamadas a la API. Con el almacenamiento de la gateway de archivos, puede realizar tareas como llevar cargas de trabajo de nube a S3, realizar copias de seguridad y archivar y estratificar y migrar datos de almacenamiento a la nube de AWS. En el diagrama siguiente se proporciona información general de la implementación del almacenamiento de archivos en Storage Gateway.

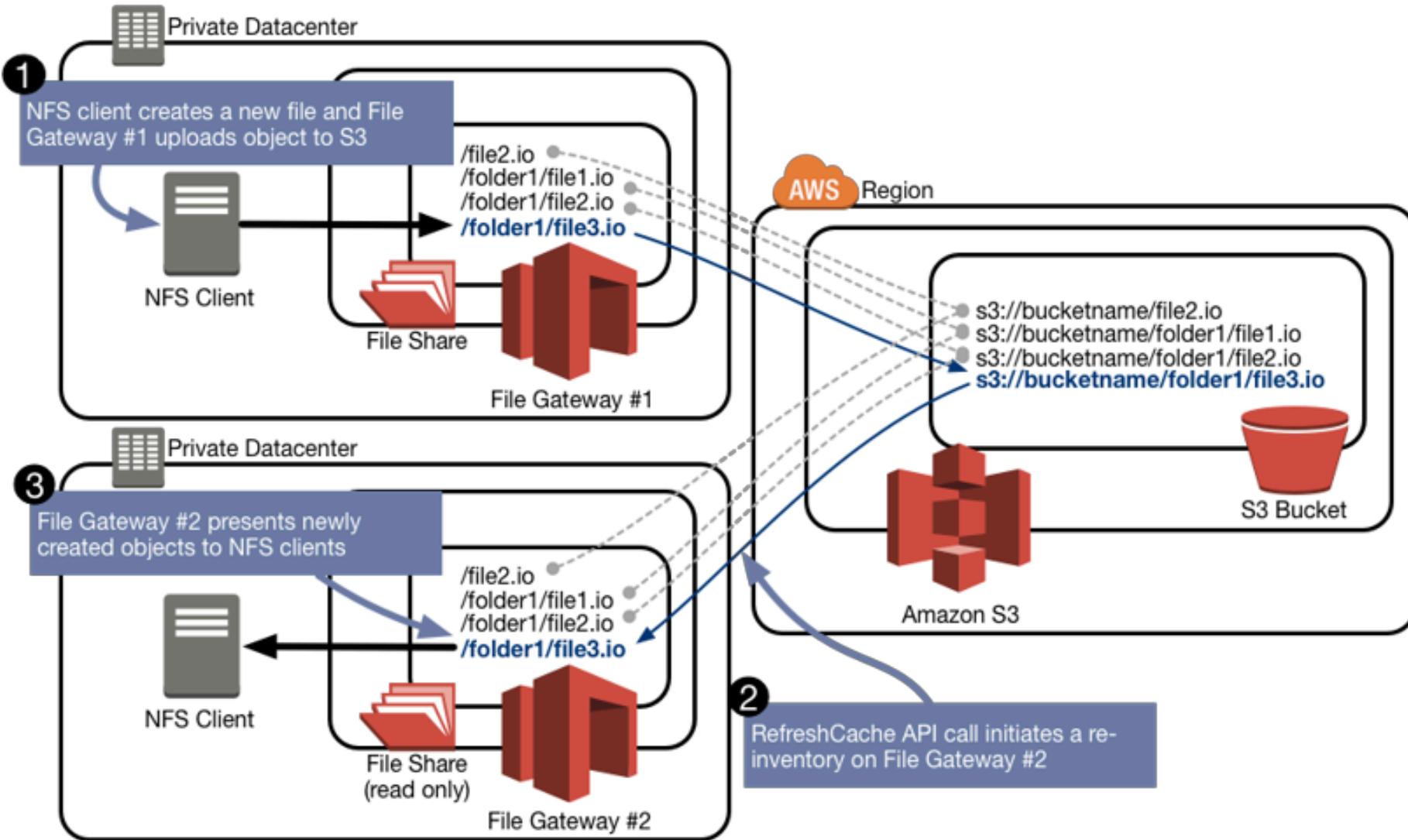












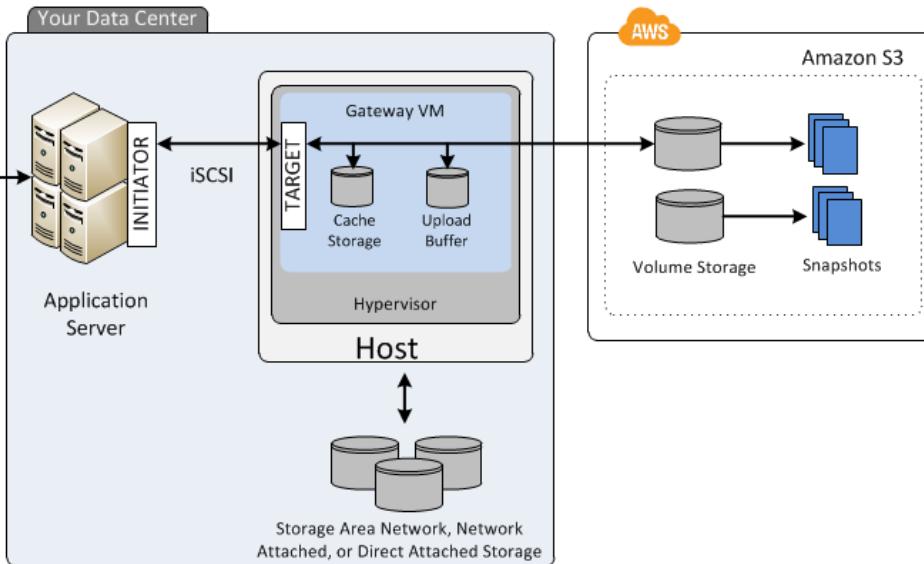
Volume gateway proporciona almacenamiento en bloques a sus aplicaciones mediante el protocolo iSCSI. Los datos de los volúmenes se almacenan en Amazon S3. Para acceder a sus volúmenes iSCSI en AWS, puede tomar instantáneas de EBS que pueden utilizarse para crear volúmenes de EBS.

P: ¿Qué es volume gateway?

Volume gateway proporciona un destino iSCSI, que le permite crear volúmenes y montarlos como dispositivos iSCSI desde sus servidores de aplicaciones locales o de EC2. Volume gateway se ejecuta en modo caché o almacenado.

- En modo caché, los datos principales se escriben en S3, mientras que los datos a los que accede a menudo de forma local se retienen en una caché para el acceso de baja latencia.
- En modo almacenado, los datos principales se almacenan localmente y todo el conjunto de datos se encuentra disponible para el acceso de bajo latencia al tiempo que se realiza una copia de seguridad asíncrona en AWS.

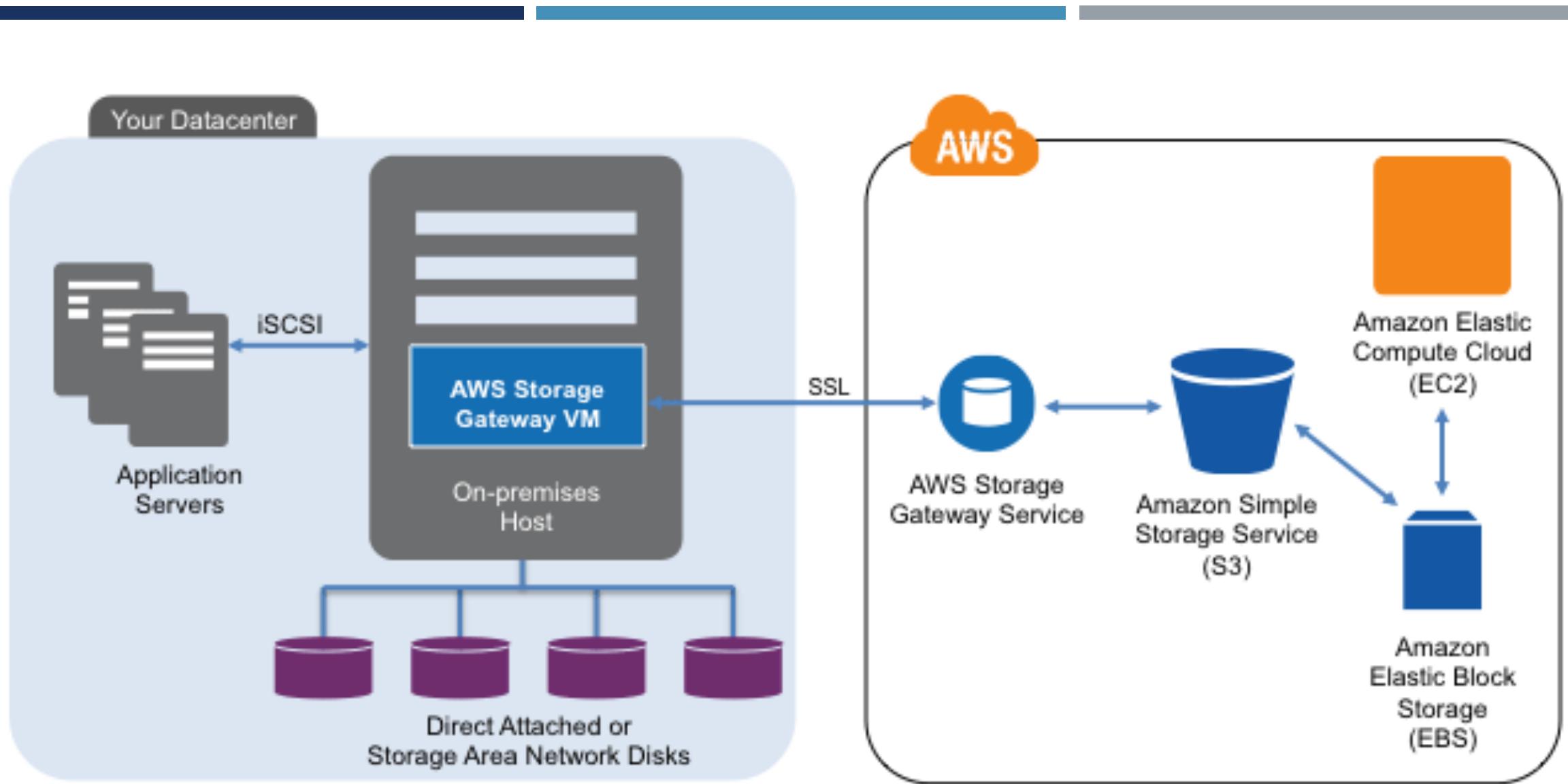
En cualquiera de los modos, puede tomar instantáneas puntuales de sus volúmenes, que se almacenan como instantáneas de Amazon EBS en AWS, lo que le permite hacer copias con control de versiones de sus volúmenes con eficiencia de espacio para protección de datos, recuperación, migración y varias otras necesidades de datos de copias.



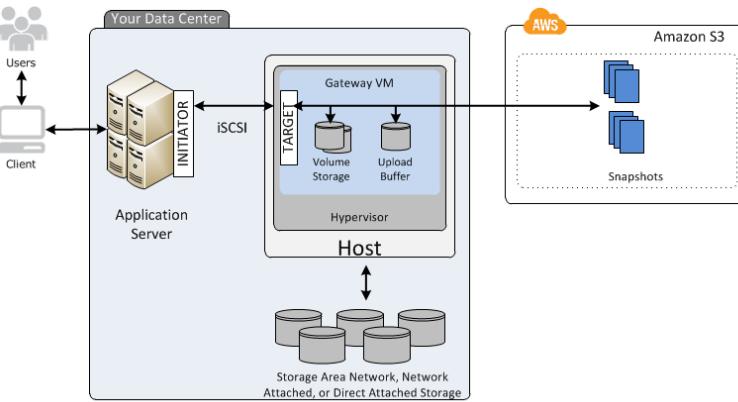
Stored Volumes Architecture

Mediante el uso de volúmenes almacenados en caché, puede usar Amazon S3 como almacenamiento de datos principal manteniendo localmente los datos de acceso frecuente en la gateway de almacenamiento. Los volúmenes almacenados en caché reducen al mínimo la necesidad de escalar la infraestructura de almacenamiento local a la vez que proporcionan a sus aplicaciones acceso de baja latencia a los datos de acceso frecuente. Puede crear volúmenes de almacenamiento con un tamaño de hasta 32 TiB y asociarlos como dispositivos iSCSI desde los servidores de aplicaciones locales. El gateway almacena los datos que se escriben en estos volúmenes en Amazon S3 y conserva los datos leídos recientemente en la caché de la gateway de almacenamiento on-prem y en el almacenamiento del búfer de carga.

Los volúmenes almacenados en caché pueden ir de 1 GiB a 32 TiB de tamaño y deben redondearse al GiB más próximo. Cada gateway configurada para volúmenes almacenados en caché admite hasta 32 volúmenes para un volumen de almacenamiento máximo de 1 024 TiB (1 PiB).



Cache Volumes Architecture



Almacenar los datos principales localmente y realizar una copia de seguridad asíncrona de los datos en AWS. Los volúmenes almacenados proporcionan aplicaciones locales con acceso de baja latencia a conjuntos de datos completos. Asimismo, proporcionan copias de seguridad duraderas externas. Puede crear volúmenes de almacenamiento y montarlos como dispositivos iSCSI desde los servidores de aplicaciones on-premises. Los datos escritos en los volúmenes almacenados se almacenan en el hardware de almacenamiento on-premises. Estos datos se copian de manera asíncrona en Amazon S3 en forma de instantáneas de Amazon Elastic Block Store (Amazon EBS).

Los volúmenes almacenados pueden ir de 1 GiB a 16 TiB de tamaño y deben redondearse al GiB más próximo. Cada gateway configurada para volúmenes almacenados en la gateway admite hasta 32 volúmenes y un almacenamiento de volumen total de 512 TiB (0,5 PiB).

Con los volúmenes almacenados, mantiene el almacenamiento de volumen on-premises en el centro de datos. Es decir, almacena todos los datos de aplicación en hardware de almacenamiento on-premises. A continuación, la gateway utiliza características que ayudan a mantener la seguridad de los datos para cargar datos en la nube de AWS para una copia de seguridad económica y una rápida recuperación de desastres. Esta solución es ideal si desea mantener los datos localmente on-premises, porque necesita un acceso de baja latencia a todos los datos y, además, mantener copias de seguridad en AWS.

P: ¿Por qué no puedo ver los datos de mi volumen en Amazon S3?

El servicio AWS Storage Gateway almacena los volúmenes en un bucket de Amazon S3. A través de AWS Storage Gateway puede acceder a sus volúmenes para realizar operaciones de E/S. No puede acceder a ellos directamente usando las acciones de la API de Amazon S3. Puede tomar instantáneas puntuales de volúmenes de la gateway que están disponibles en forma de instantáneas de Amazon EBS, y que se pueden convertir en volúmenes de Storage Gateway o volúmenes de EBS. Use la interfaz de archivos para trabajar con los datos en S3 de forma nativa.

P: ¿Qué tipo de cifrado utiliza volume gateway para proteger mis datos?

Todos los datos transferidos entre la gateway y el almacenamiento de AWS se cifran con SSL. De manera predeterminada, todos los datos almacenados por un volume gateway en S3 se cifran del lado del servidor con Amazon S3-Managed Encryption Keys (SSE-S3).

Opcionalmente, puede configurar el cifrado de los datos almacenados en AWS en volúmenes con las claves administradas de AWS KMS a través de la API de Storage Gateway. Podrá especificar una de las claves maestras de clientes (CMK) administradas como la clave de KMS. La CMK configurada que se usa para cifrar un volumen no se puede modificar una vez creada. Para obtener más información, consulte "[Cómo cifrar sus datos con AWS Key Management System](#)", en la guía del usuario de Storage Gateway, que incluye detalles fundamentales sobre el uso de la característica.

P: ¿Cómo puedo restablecer una instantánea en una gateway?

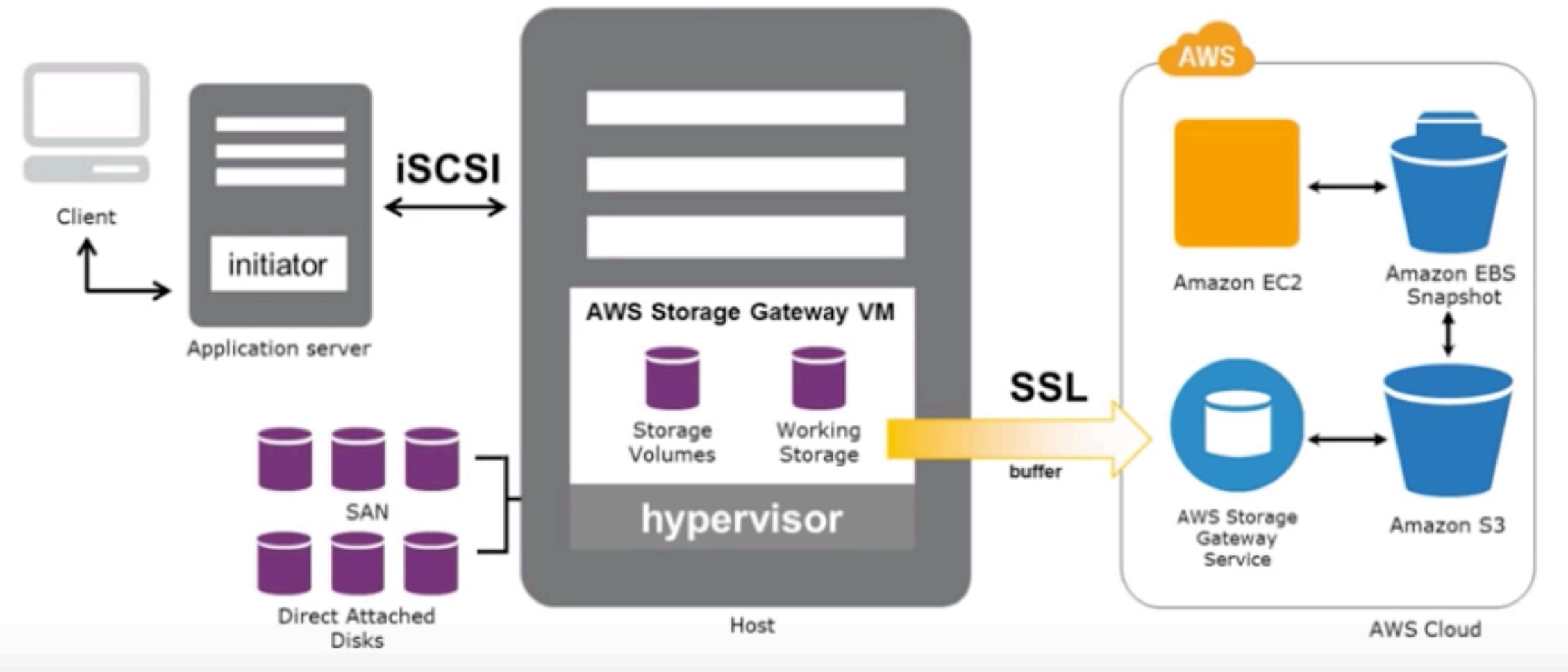
A cada instantánea se le asigna un identificador único que se puede visualizar mediante la consola de administración de AWS. Puede especificar este identificador único para crear volúmenes de AWS Storage Gateway o Amazon EBS a partir de cualquiera de sus instantáneas existentes.

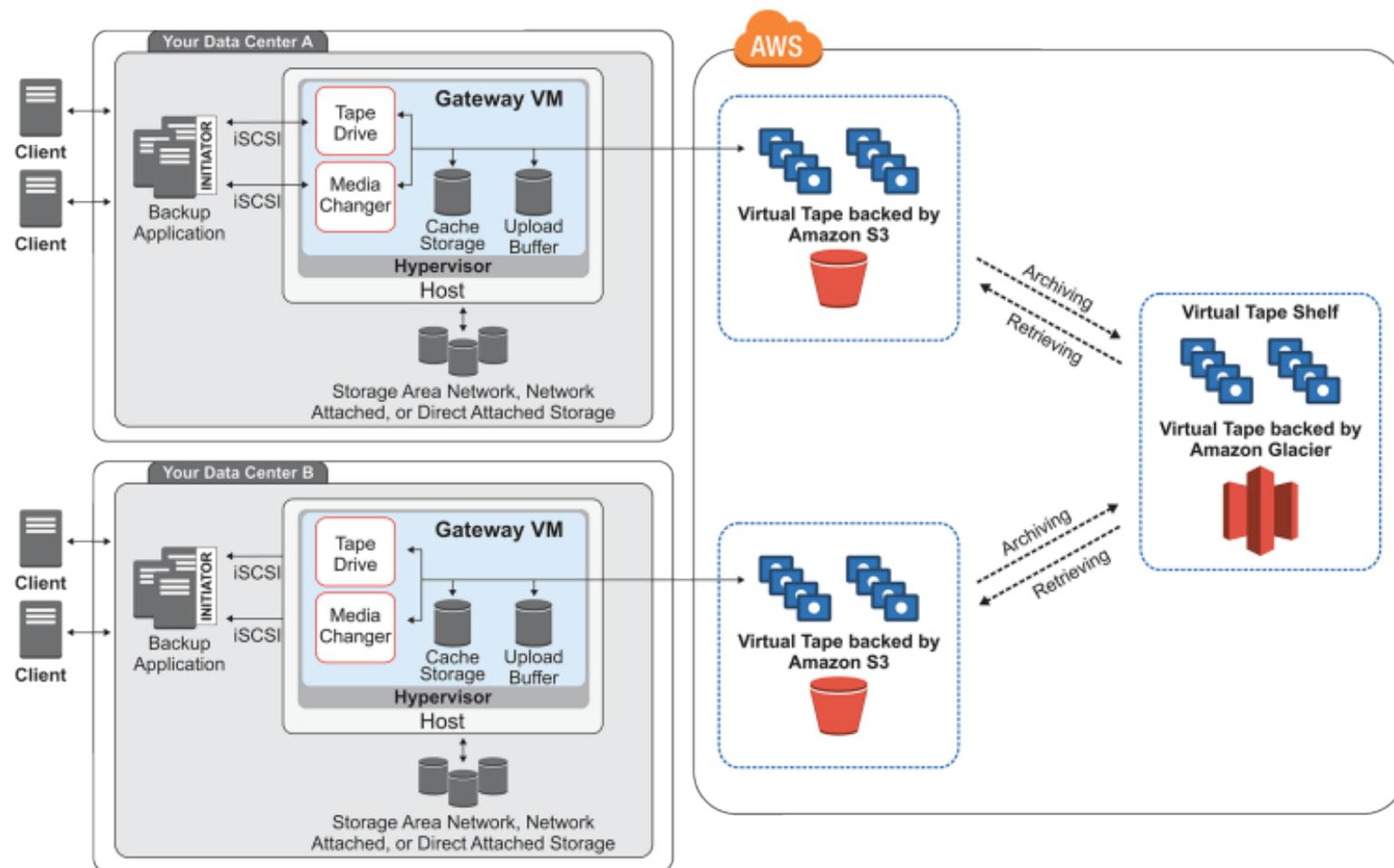
Mediante el uso de la consola de administración de AWS, puede crear un nuevo volumen de a partir de una instantánea que haya almacenado en Amazon S3. Después, puede montar este volumen como un dispositivo iSCSI en su servidor de aplicaciones locales.

Dado que los volúmenes en caché de gateway almacenan los datos principales de Amazon S3, al crear un volumen nuevo a partir de una instantánea, la gateway conserva los datos de la instantánea en Amazon S3, donde pasan a ser los datos principales del nuevo volumen.

Como los volúmenes almacenados almacenan sus datos principales a nivel local, al crear un nuevo volumen a partir de una instantánea, su gateway descarga los datos que contiene la instantánea al hardware local. A partir de entonces, se convierten en datos principales del nuevo volumen.

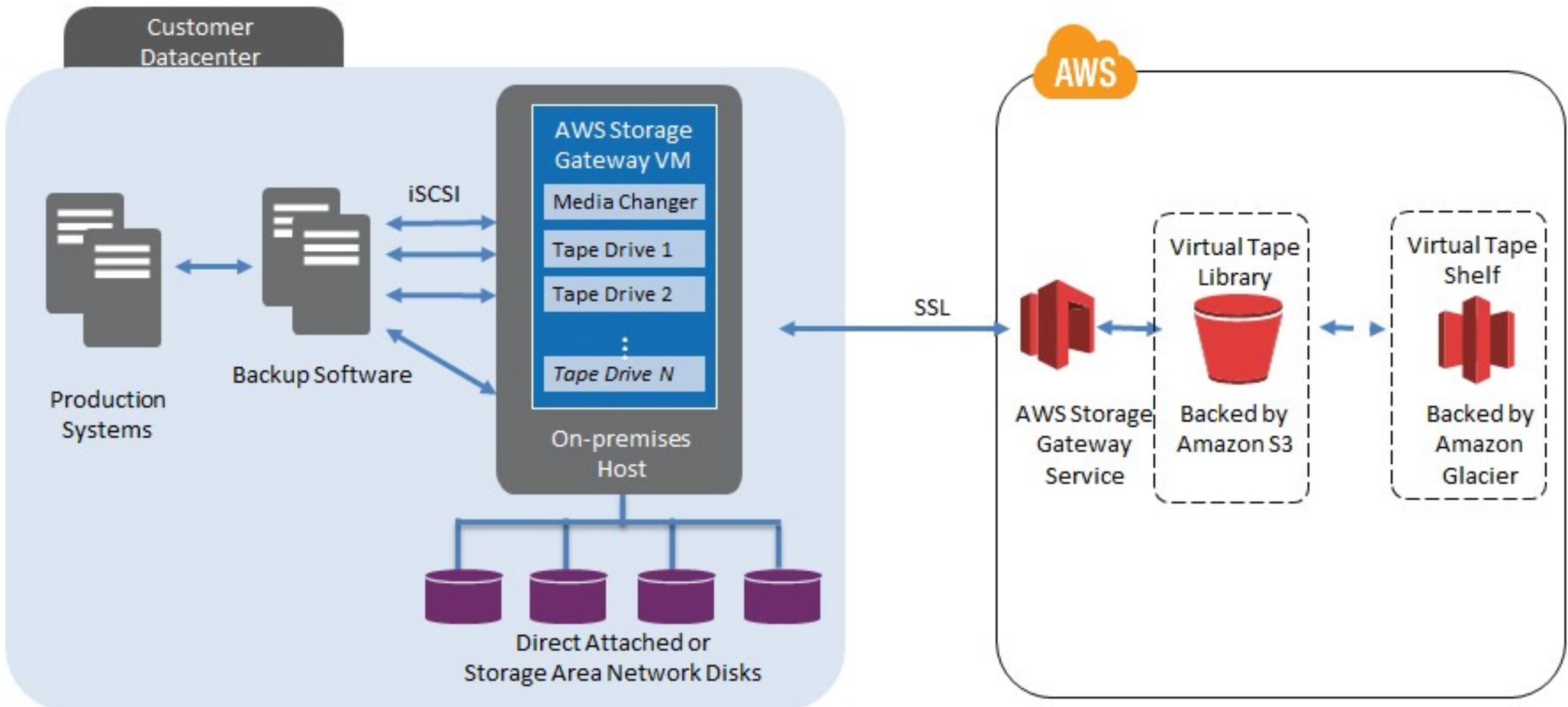
How AWS Storage Gateway Works





Tape Gateways

Gateway de cinta ofrece una solución duradera y económica para archivar datos en la nube de AWS. Con su interfaz de biblioteca de cintas virtuales (VTL), puede utilizar la infraestructura de copia de seguridad existente basada en cintas para almacenar datos en cartuchos de cinta virtuales creados en la gateway de cinta. Cada gateway de cinta está preconfigurada con un cambiador de medios y unidades de cinta. Estos están disponibles para las aplicaciones de copia de seguridad cliente existentes como dispositivos iSCSI. Agregue los cartuchos de cinta que necesite para archivar los datos.



- **Cinta virtual:** una cinta virtual es como una cinta física. Sin embargo, los datos de las cintas virtuales se almacenan en la nube de AWS. Para crear cintas virtuales, puede utilizar la consola de Storage Gateway o crearlas mediante programación utilizando la API de Storage Gateway. Cada gateway puede contener hasta 1500 cintas o hasta 1 PiB de datos de cinta totales a la vez. El tamaño de cada cinta virtual, que puede configurar al crear la cinta, está entre 100 GiB y 5 TiB.
- **Biblioteca de cintas virtuales (VTL):** una VTL es como una biblioteca de cintas física La VTL incluye la colección de cintas virtuales almacenadas. Cada gateway de cinta viene con una VTL.
- **Unidad de cinta:** una unidad de cinta VTL es análoga a una unidad de cinta física capaz de realizar operaciones de búsqueda y E/S en una cinta. Cada VTL viene con un conjunto de 10 unidades de cinta, que están disponibles para la aplicación de copia de seguridad como dispositivos iSCSI.
- **Cambiador de medios:** es análogo a un robot que traslada cintas entre ranuras y unidades de cinta de una biblioteca de cintas físicas. Cada VTL tiene uno que está disponible para la aplicación de copia de seguridad como un dispositivo iSCSI.
- **Archivo:** el archivo es análogo a una instalación externa donde se almacenan cintas. Puede archivar las cintas de la VTL de la gateway en el archivo de almacenamiento. Si es necesario, puede recuperar las cintas del archivo de almacenamiento y volver a colocarlas en la VTL de la gateway.
- **Archivado de cintas:** cuando el software de copia de seguridad expulsa una cinta, la gateway la traslada al archivo para almacenarla a largo plazo. Las cintas archivadas se almacenan en la estantería de cintas virtuales (VTS). La VTS se basa en [S3 Glacier](#) o en [S3 Glacier Deep Archive](#), un servicio de almacenamiento de bajo costo para archivar datos, crear copias de seguridad y para la retención de datos a largo plazo.
- **Recuperación de cintas:** las cintas archivadas no se pueden leer directamente. Para leer una cinta archivada, primero debe recuperarla en la gateway de cinta, ya sea mediante la consola de Storage Gateway o mediante la API de Storage Gateway.
 - Si archiva una cinta en GLACIER, normalmente puede recuperarla en un plazo de entre 3 y 5 horas.
 - Si archiva una cinta en DEEP_ARCHIVE, normalmente puede recuperarla en un plazo de 12 horas.



Amazon
EC2

EC2

AMAZON ELASTIC COMPUTE CLOUD (EC2)

- Virtual computing environments, known as ***Ec2 instances***
- Preconfigured templates for your instances, known as ***Amazon Machine Images (AMIs)***, that package the bits you need for your server (including the operating system and additional software)
- Various configurations of CPU, memory, storage, and networking capacity for your instances, known as ***Instance types***
- Secure login information for your instances using ***key pairs*** (AWS stores the public key, and you store the private key in a secure place)
- Storage volumes for temporary data that's deleted when you stop or terminate your instance, known as ***Instance store volumes***
- Persistent storage volumes for your data using Amazon Elastic Block Store (Amazon EBS), known as ***Amazon EBS volumes***
- Multiple physical locations for your resources, such as instances and Amazon EBS volumes, known as ***Regions and Availability Zones***
- A firewall that enables you to specify the protocols, ports, and source IP ranges that can reach your instances using ***security groups***
- Static IP addresses for dynamic cloud computing, known as ***Elastic IP addresses***
- Metadata, known as ***tags***, can be created and assigned to EC2 resources
- Virtual networks you can create that are logically isolated from the rest of the AWS cloud, and that you can optionally connect to your own network, known as ***Virtual private clouds (VPCs)***

EC2 INSTANCE TYPES

- General purpose
 - A1, T2, T3, M4, M5, M5D
- Compute optimized
 - C4, C5, C5D
- Memory optimized
 - R4, R5, R5D, X1E, Z1D
- Storage optimized
 - D2, H1, I3
- Accelerated computing
 - F1, G3, P2, P3
- Bare metal
 - I3, U-5TB1, U-9TB1, U-12TB1

General Purpose	Compute Optimised	Memory Optimised	Accelerated Computing	Storage Optimised
 A1 ARM based core and custom silicon	 C4 Compute - CPU intensive apps and DBs	 R4 RAM - Memory intensive apps and DB's	 P2 Processing optimised-Machine Learning	 H1 High Disk Throughput - Big data clusters
 T2 Tiny - Web servers and small DBs		 X1 Xtreme RAM - For SAP/Spark	 G3 Graphics Intensive - Video and streaming	 I3 IOPS - NoSQL DBs
 M4 Main - App servers and general purpose		 z1d High Compute and High Memory - Gaming	 F1 Field Programmable - Hardware acceleration	 D2 Dense Storage - Data Warehousing

[Https://aws.amazon.com/ec2/instance-types/](https://aws.amazon.com/ec2/instance-types/)

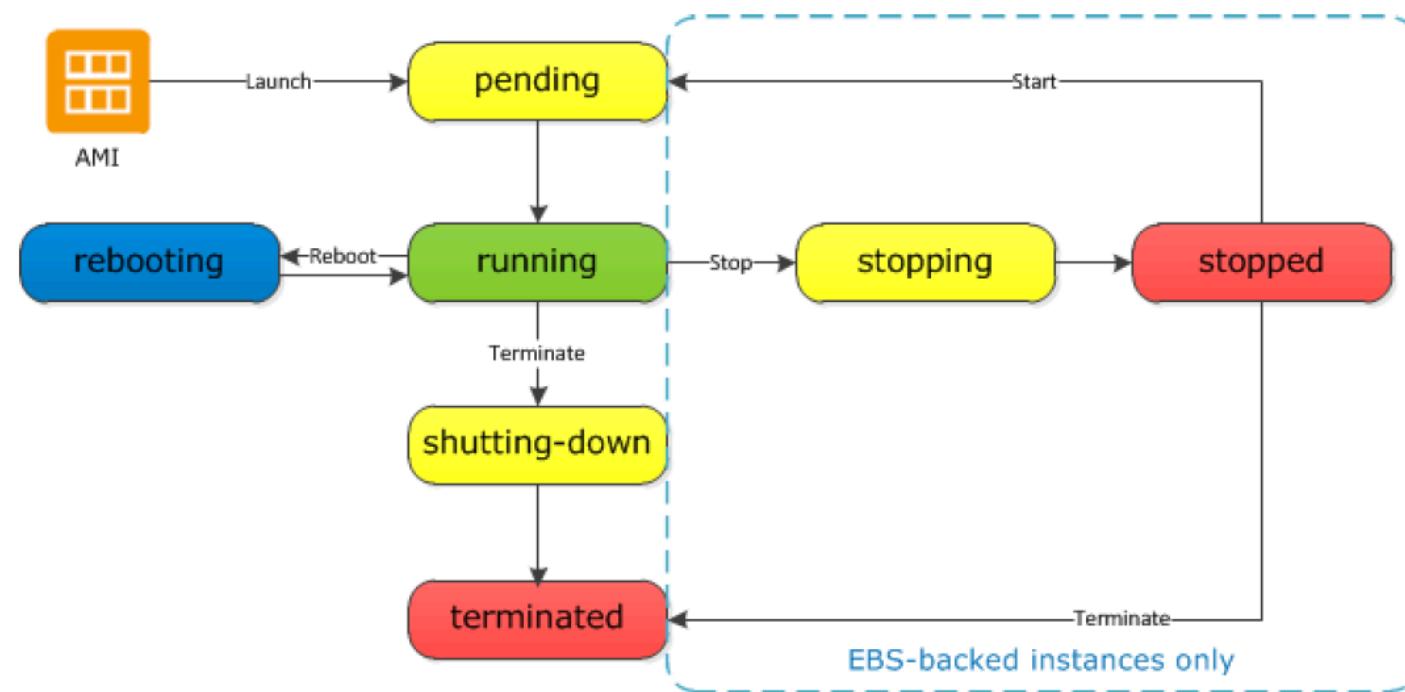
EC2 PRICING OPTIONS

- **On-Demand Instances** – Pay, by the second, for the instances that you launch.
- **Reserved Instances** – Purchase, at a significant discount, instances that are always available, for a term from one to three years.
- **Scheduled Instances** – Purchase instances that are always available on the specified recurring schedule, for a one-year term.
- **Spot Instances** – Request unused EC2 instances, which can lower your Amazon EC2 costs significantly.
- **Dedicated Hosts** – Pay for a physical host that is fully dedicated to running your instances, and bring your existing per-socket, per-core, or per-VM software licenses to reduce costs.
- **Dedicated Instances** – Pay, by the hour, for instances that run on single-tenant hardware.
- **Capacity Reservations** – Reserve capacity for your EC2 instances in a specific Availability Zone for any duration.

EC2 PRICING OPTIONS

Description	Reserved Instances	On-Demand Instances	Spot Instances
Available Instance Types	All	All	All
Launch Priority	1	2	3
Pricing Model	Upfront Fee + Reduced Fixed Price	Fixed Price	Bidding (Variable Spot Price)
Predictable Server Uptime?	Yes	Yes	No

AWS EC2 INSTANCE LIFECYCLE



AWS EC2 INSTANCE LIFECYCLE

- **Pending**
 - When the instance is first launched it enters into the pending state
- **Running**
 - After the instance is launched, it enters into the running state
 - Charges are incurred for every hour or partial hour the instance is running even if it is idle
- **Instance reboot**
 - Both EBS-backed and Instance store-backed instances can be rebooted
 - An instance retains its public DNS, public and private IP address during the reboot
 - Data on the EBS and Instance store volume is also retained
 - Amazon recommends to use Amazon EC2 to reboot the instance instead of running the operating system reboot command from your instance as it performs a hard reboot if the instance does not cleanly shutdown within four minutes also creates an API record in CloudTrail, if enabled.

AWS EC2 INSTANCE LIFECYCLE

- ***Start & Stop*** (EBS-backed instances only)

- Only an EBS-backed instance can be stopped and started. Instance store-backed instance cannot be stopped and started
- An instance can be stopped & started in case the instance fails a status check or is not running as expected
- Stop
 - After the instance is stopped, it enters into stopping state and then into stopped state.
 - Charges are only incurred for the EBS storage and not for the instance hourly charge or data transfer.
 - While the instance is stopped, you can treat its root volume like any other volume, and modify it for e.g. repair file system problems or update software or change the instance type, user data, EBS optimization attributes etc
 - Volume can be detached from the stopped instance, and attached to a running instance, modified, detached from the running instance, and then reattached to the stopped instance. It should be reattached using the storage device name that's specified as the root device in the block device mapping for the instance.
- Start
 - When the instance is started, it enters into pending state and then into running
 - An instance when stopped and started is launched on a new host
 - Any data on an instance store volume (not root volume) would be lost while data on the EBS volume persists
- EC2 instance retains its private IP address as well as the Elastic IP address. However, the public IP address, if assigned instead of the Elastic IP address, would be released
- Charges for full hour are incurred for every transition from stopped to running, even if the transition is within the same hour for e.g. if you stop and start your instances 2 times in an hour, you would be charged for 3 full hours, one for the start and then for the 2 transitions as if you had 3 instances running during that hour

AWS EC2 INSTANCE LIFECYCLE

■ *Instance retirement*

- An instance is scheduled to be retired when AWS detects irreparable failure of the underlying hardware hosting the instance.
- When an instance reaches its scheduled retirement date, it is stopped or terminated by AWS.
- If the instance root device is an Amazon EBS volume, the instance is stopped, and can be started again at any time.
- If the instance root device is an instance store volume, the instance is terminated, and cannot be used again.

AWS EC2 INSTANCE LIFECYCLE

- ***Instance Termination.***

- An instance can be terminated, and it enters into the shutting-down and then the terminated state
- After an instance is terminated, it can't be connected and no charges are incurred
- Instance Shutdown behaviour
 - **EBS-backed instance** supports **InstanceInitiatedShutdownBehavior** attribute which determines whether the instance would be stopped or terminated when a shutdown command is initiated from the instance itself for e.g. *shutdown, halt or poweroff command in linux*.
 - Default behaviour for the the instance to be stopped.
 - A shutdown command for an Instance store-backed instance will always terminate the instance

- **Termination protection**

- Termination protection (**DisableApiTermination** attribute) can be enabled on the instance to prevent it from being accidentally terminated
- **DisableApiTermination** from the Console, CLI or API.
- Instance can be terminated through Amazon EC2 CLI.
- Termination protection does not work for instances when
 - part of an Autoscaling group
 - launched as Spot instances
 - terminating an instance by initiating shutdown from the instance

EC2 LAUNCH

1. Choose AMI 2. Choose Instance Type 3. Configure Instance 4. Add Storage 5. Add Tags 6. Configure Security Group 7. Review

Step 3: Configure Instance Details

Auto-assign Public IP: Use subnet setting (Enable)

Placement group: Add instance to placement group

Capacity Reservation: Open | Create new Capacity Reservation

IAM role: CompleteAccess | Create new IAM role

Shutdown behavior: Stop

Enable termination protection: Protect against accidental termination

Monitoring: Enable CloudWatch detailed monitoring
Additional charges apply.

Tenancy: Shared - Run a shared hardware instance
Additional charges will apply for dedicated tenancy.

Elastic Inference: Add an Elastic Inference accelerator
Additional charges apply.

T2/T3 Unlimited: Enable
Additional charges may apply

Advanced Details

User data:

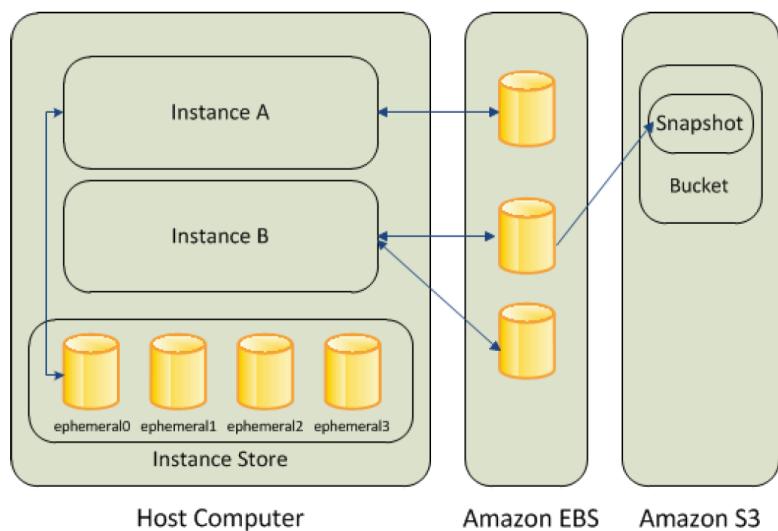
As text (radio button selected) As file Input is already base64 encoded

```
#!/bin/bash
sudo apt update -y && sudo apt install -y nginx
sudo systemctl start nginx
```

Reboot vs. Stop vs. Terminate

Characteristic	Reboot	Stop/Start (EBS-backed instances only)	Terminate
Host computer	The instance stays on the same host computer .	The instance runs on a new host computer .	
Public IP address	No change	New address assigned	
Elastic IP addresses (EIP)	EIP remains associated with the instance.	EIP remains associated with the instance.	EIP is disassociated from the instance.
Instance store volumes	Preserved	Erased	Erased
EBS volume	Preserved	Preserved	Boot volume is deleted by default .
Billing	Instance billing hour doesn't change.	You stop incurring charges as soon as state is changed to <i>stopping</i> .	You stop incurring charges as soon as state is changed to <i>shutting-down</i> .

EC2 STORAGE OVERVIEW



- Amazon EC2 provides flexible, cost effective and easy-to-use EC2 storage options with a unique combination of performance and durability
 - Amazon Elastic Block Store (EBS)
 - Amazon EC2 Instance Store
 - Amazon Simple Storage Service (S3)
- While EBS and Instance store are Block level, Amazon S3 is an Object level storage

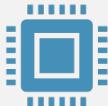
EBS TYPE

AWS provides the following EBS volume types, which differ in performance characteristics and price which can be tailored for storage performance and cost to the needs of the applications



SSD-backed volumes optimized for transactional workloads involving frequent read/write operations with small I/O size, where the dominant performance attribute is **IOPS**

General Purpose SSD (gp2)
Provisioned IOPS SSD (io1)



HDD-backed volumes optimized for large streaming workloads where **throughput** (measured in MiB/s) is a better performance measure than IOPS

Throughput Optimized HDD (st1)
Cold HDD (sc1)

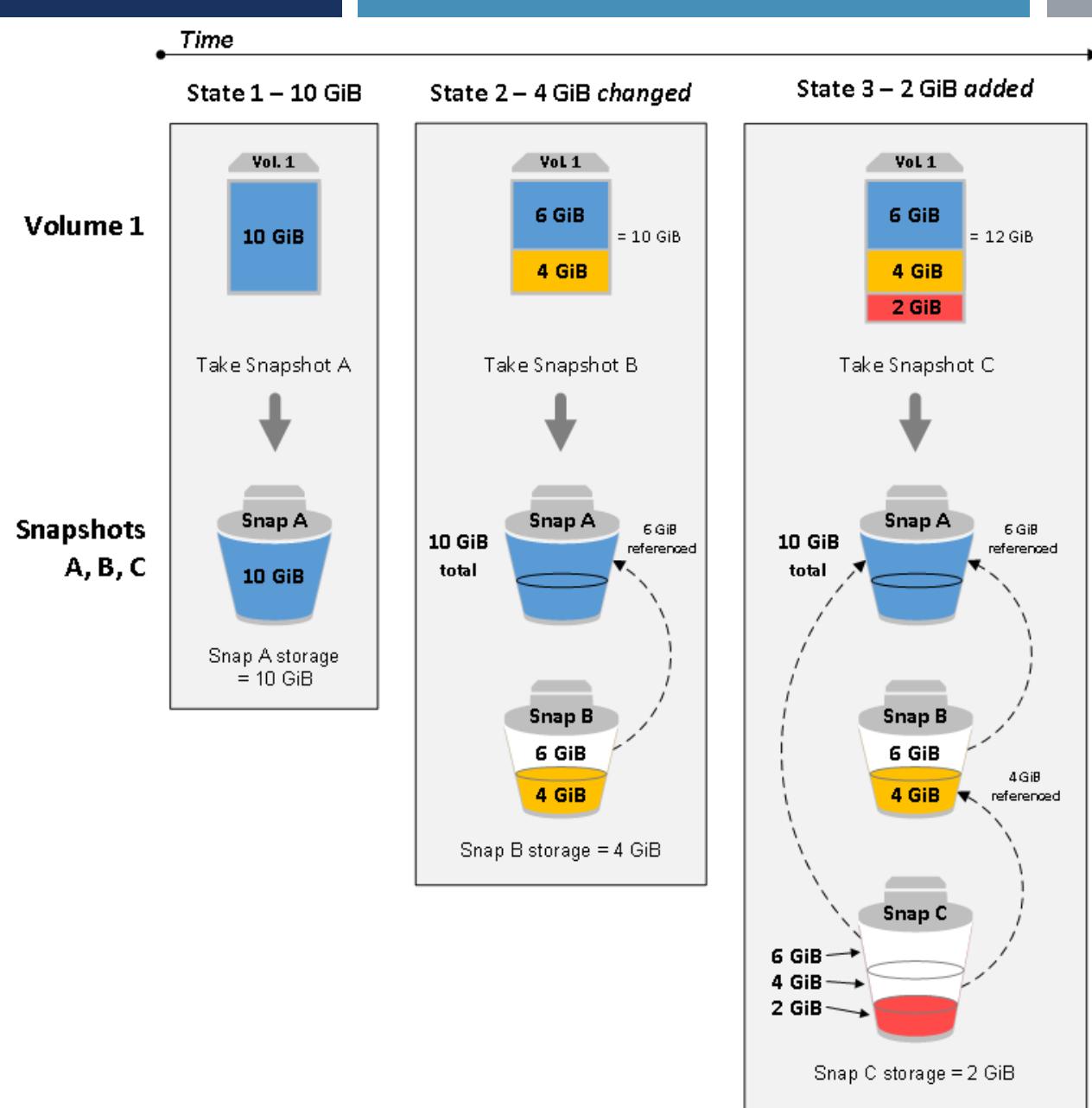
	Solid-State Drives (SSD)		Hard disk Drives (HDD)	
Volume Type	General Purpose SSD (gp2)*	Provisioned IOPS SSD (io1)	Throughput Optimized HDD (st1)	Cold HDD (sc1)
Description	General purpose SSD volume that balances price and performance for a wide variety of workloads	Highest-performance SSD volume for mission-critical low-latency or high-throughput workloads	Low cost HDD volume designed for frequently accessed, throughput-intensive workloads	Lowest cost HDD volume designed for less frequently accessed workloads
Use Cases	<ul style="list-style-type: none"> Recommended for most workloads System boot volumes Virtual desktops Low-latency interactive apps Development and test environments 	<ul style="list-style-type: none"> Critical business applications that require sustained IOPS performance, or more than 10,000 IOPS or 160 MiB/s of throughput per volume Large database workloads, such as: <ul style="list-style-type: none"> MongoDB Cassandra Microsoft SQL Server MySQL PostgreSQL Oracle 	<ul style="list-style-type: none"> Streaming workloads requiring consistent, fast throughput at a low price Big data Data warehouses Log processing Cannot be a boot volume 	<ul style="list-style-type: none"> Throughput-oriented storage for large volumes of data that is infrequently accessed Scenarios where the lowest storage cost is important Cannot be a boot volume
API Name	gp2	io1	st1	sc1
Volume Size	1 GiB - 16 TiB	4 GiB - 16 TiB	500 GiB - 16 TiB	500 GiB - 16 TiB
Max. IOPS**/Volume	10,000	32,000***	500	250
Max. Throughput/Volume	160 MiB/s	500 MiB/s†	500 MiB/s	250 MiB/s
Max. IOPS/Instance	80,000	80,000	80,000	80,000
Max. Throughput/Instance††	1,750 MiB/s	1,750 MiB/s	1,750 MiB/s	1,750 MiB/s
Dominant Performance Attribute	IOPS	IOPS	MiB/s	MiB/s

RAID

Configuration	Use	Advantages	Disadvantages
RAID 0	When I/O performance is more important than fault tolerance; for example, as in a heavily used database (where data replication is already set up separately).	I/O is distributed across the volumes in a stripe. If you add a volume, you get the straight addition of throughput.	Performance of the stripe is limited to the worst performing volume in the set. Loss of a single volume results in a complete data loss for the array.
RAID 1	When fault tolerance is more important than I/O performance; for example, as in a critical application.	Safer from the standpoint of data durability.	Does not provide a write performance improvement; requires more Amazon EC2 to Amazon EBS bandwidth than non-RAID configurations because the data is written to multiple volumes simultaneously.

EBS Snapshots

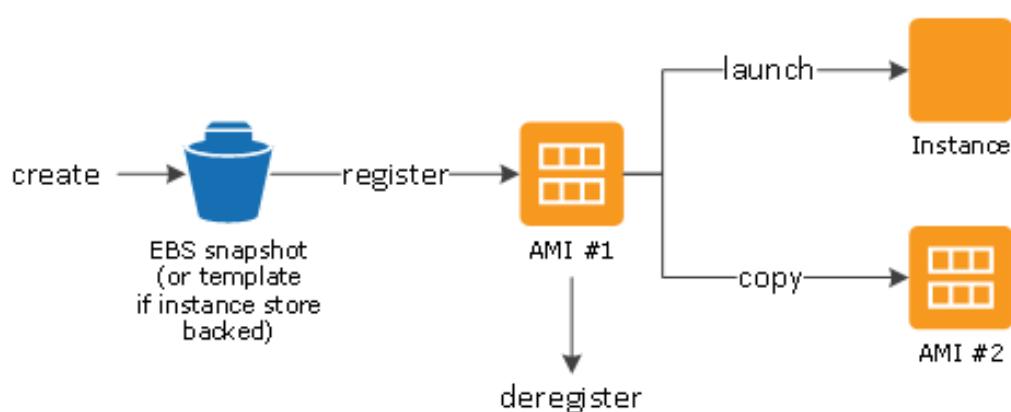
- Snapshots exist in S3
- Snapshots are point in time copies of volumes
- Snapshots are incremental
- You can create AMI's from EBS-backed instances and Snapshots
- You can change EBS volumes sizes on the fly, including changing the size and storage type
- To move an EC2 volume from one AZ/Region to another, take a snapshot or an image of it, then copy to a new AZ/Region
- Snapshots of encrypted volumes are encrypted by default
- Volumes restores from encrypted snapshots are encrypted by default.
- You can share snapshots (other AWS accounts or made public), but only if they are unencrypted



AMI

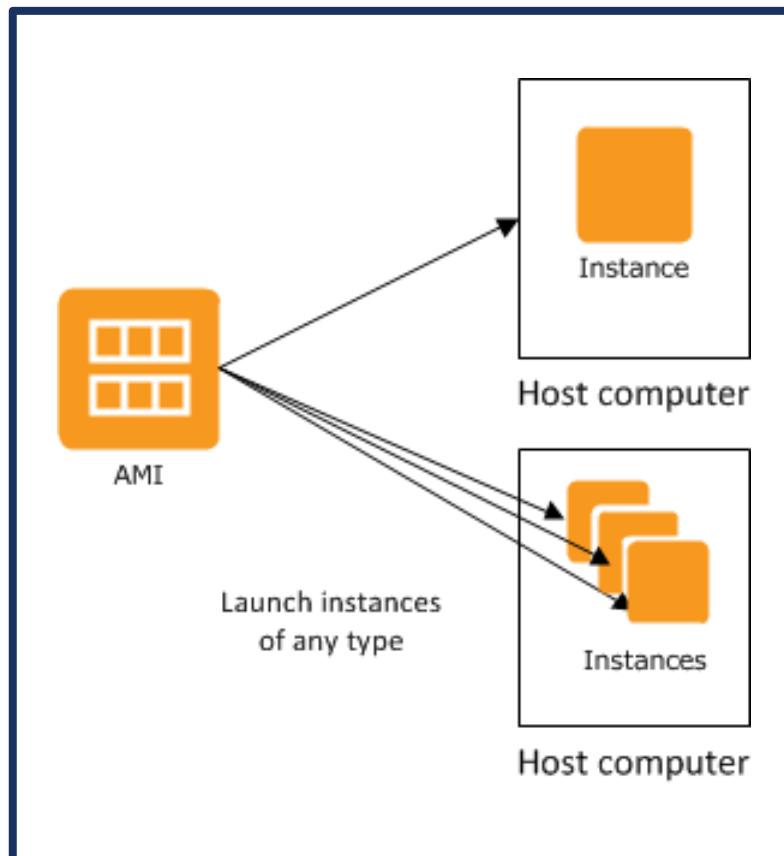
- An Amazon Machine Image (AMI) provides the information required to launch an instance. You must specify an AMI when you launch an instance. You can launch multiple instances from a single AMI when you need multiple instances with the same configuration. You can use different AMIs to launch instances when you need instances with different configurations.
- An AMI includes the following:
 - One or more EBS snapshots, or, for instance-store-backed AMIs, a template for the root volume of the instance (for example, an operating system, an application server, and applications).
 - Launch permissions that control which AWS accounts can use the AMI to launch instances.
 - A block device mapping that specifies the volumes to attach to the instance when it's launched.

AMI



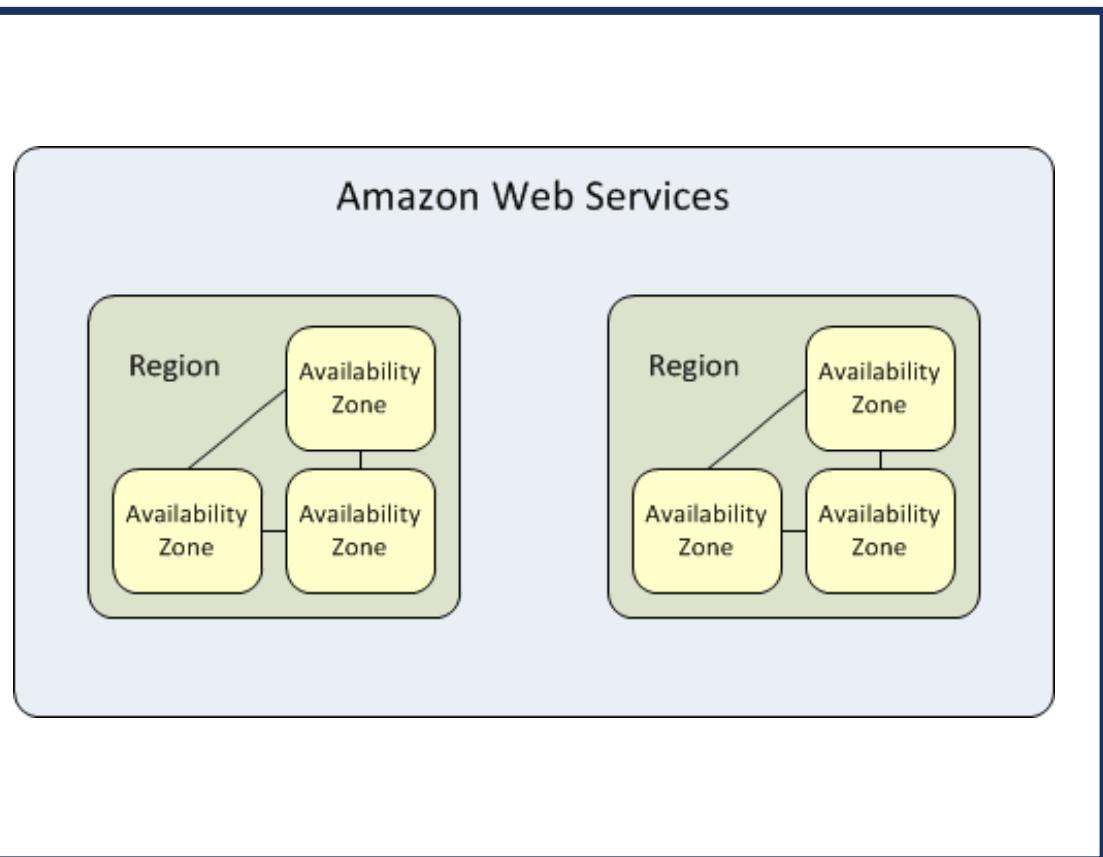
- The following diagram summarizes the AMI lifecycle. After you create and register an AMI, you can use it to launch new instances. (You can also launch instances from an AMI if the AMI owner grants you launch permissions.) You can copy an AMI within the same region or to different regions. When you no longer require an AMI, you can deregister it.

AMI AND INSTANCE



- An *Amazon Machine Image (AMI)* is a template that contains a software configuration (for example, an operating system, an application server, and applications).
- From an AMI, you launch an *instance*, which is a copy of the AMI running as a virtual server in the cloud. You can launch multiple instances of an AMI, as shown in the following figure.

REGION AND AVAILABILITY ZONE CONCEPTS

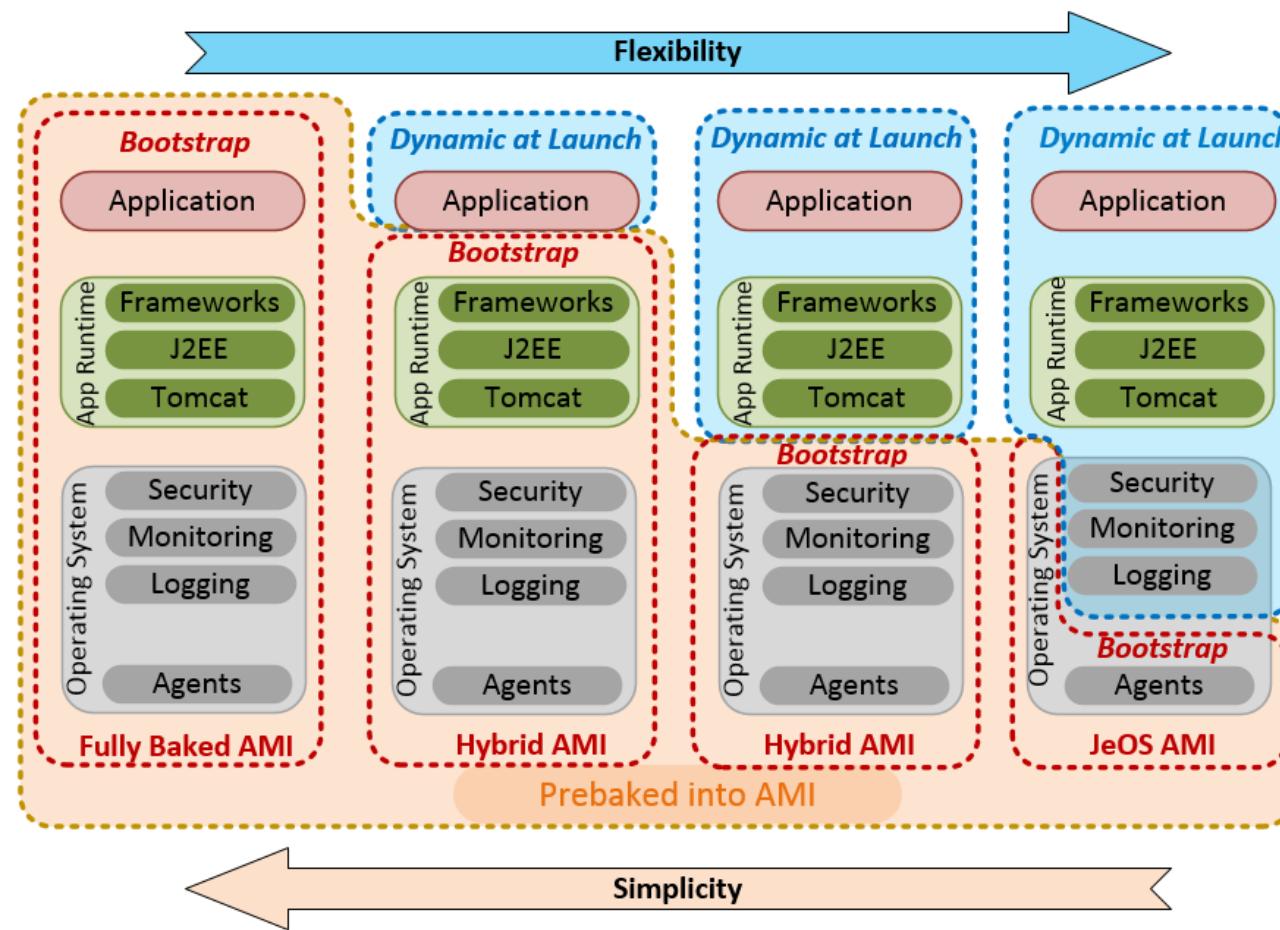


- Each Region is completely independent.
- Each Availability Zone is isolated, but the Availability Zones in a Region are connected through low-latency links.
- The following diagram illustrates the relationship between Regions and Availability Zones. An *Amazon Machine Image (AMI)* is a template that contains a software configuration (for example, an operating system, an application server, and applications).

AWS AMI DESIGN

- Amazon Web Services (AWS) offers its customers several methods that make it easier to provision Amazon Elastic Compute Cloud (Amazon EC2) instances and store instance configurations across a variety of different server and application deployment models. The most common unit of management is the Amazon Machine Image (AMI), which provides the information required to launch an EC2 instance. AWS customers specify an AMI when launching an instance and can use a single AMI to launch multiple EC2 instances. This webpage provides AWS customers with strategic guidance and common methods for determining what software to include in the image when building custom AMIs.
- AMI design options fall along a spectrum of deployment simplicity in relation to deployment flexibility. The simplest AMIs are fully baked and purpose-built to deploy a complete running instance, including the installation and configuration of all required software. However, this approach limits flexibility, as a fully baked AMI can only be used to deploy a single instance or a farm of identical instances. The most flexible AMIs include only minimal configurations and software before then dynamically installing the required packages on first boot. This approach trades simplicity for flexibility as each instance must be properly bootstrapped before it can function as intended.

AWS AMI DESIGN



AWS AMI DESIGN

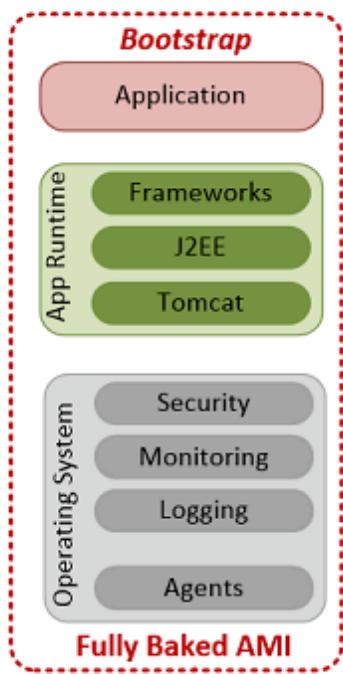
Fully Baked AMI

On one end of the spectrum is the most common method for AMI design that provisions a fully functional instance, including all necessary software for a specific role or even a single instance. Organizations often create a baseline AMI that conforms to minimal security and configuration requirements, and then build on this baseline to create fully baked AMIs that are specific to applications or infrastructures for actual instance deployment.

Fully baked AMIs are the simplest to deploy and provide the fastest launch times. For this reason, fully baked AMIs are useful to quickly deploy replacement instances or add additional capacity. Fully baked AMIs also require minimal changes when launched and can be used to capture manual installation and configuration steps that cannot be automated. The process of using fully baked AMIs is similar to how many organizations deploy virtual servers in their existing data centers, which makes this simple approach applicable to customers who are new to the AWS platform and are accustomed to image-based server deployments.

Considerations

At scale, it can be expensive and cumbersome to maintain unique AMIs for every instance or group of instances. This approach is therefore more suitable for smaller AWS deployments or when combined with an automated AMI build and management system. Even though this approach deploys a fully functional system, AWS highly recommends including a mechanism to read and process EC2-instance user data at launch. It is often advantageous to be able to pass launch instructions for minor configuration changes to your OS or application, such as those specific to a region, availability zone, or lifecycle environment.



AWS AMI DESIGN

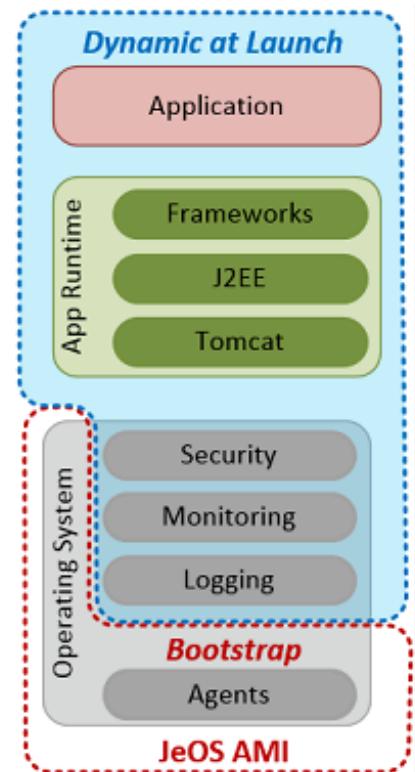
JeOS AMI

At the other end of the spectrum is the *just enough operating system* (JeOS) approach. This approach creates an AMI that combines a minimal operating system with a configuration management agent that builds a fully functional system at instance launch. On first boot, the configuration agent downloads, installs, configures, and integrates all required software.

JeOS AMIs offer the most flexibility during deployment and the highest levels of portability because they leverage minimal server images. Customers who prefer this approach typically have experience with AWS and configuration automation tools. They want deployment flexibility to be able to pull the latest software builds and updates at launch, or they might require minimal complexity in order to deploy resources to both AWS and on-premises environments.

Considerations

Downloading, installing, and configuring all required software on first boot can significantly increase instance launch times, making this approach less suitable for quickly replacing a failed instance or adding capacity. Also, JeOS AMIs require tight coupling with a configuration management system and can be more difficult to use with legacy software that typically has limited automation capabilities.



AWS AMI DESIGN

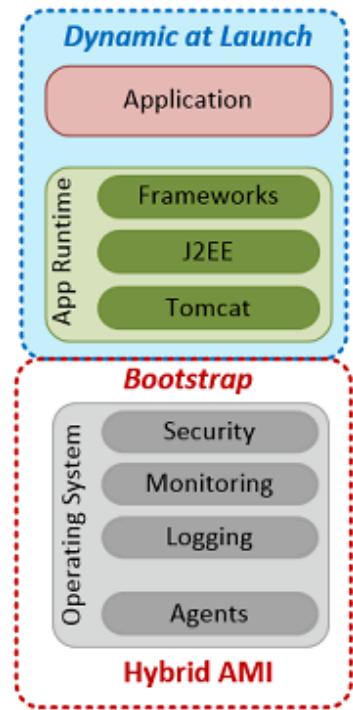
Hybrid AMI

Hybrid AMIs provide a subset of the software needed to produce a fully functional instance, falling in between the fully baked and JeOS options on the AMI design spectrum. This approach creates a partially baked, generic infrastructure AMI that is further configured on first boot based on specific application requirements. The decision about what to bake into the AMI and what to install and configure at launch is typically based on how reusable the underlying software package is across all instances and how long it takes to download, install, and configure the software component. For example, to reduce instance launch times, include server management agents and baseline security configurations into an AMI. To further reduce instance launch times, include the necessary software to create role-specific AMIs (e.g., web servers, application servers, database servers).

Hybrid AMIs combine the flexibility of post-launch changes with the speed of preinstalled and preconfigured infrastructure components. This can be especially useful for merging frequently changing software (such as application code or security updates) with infrequently changing software (such as management agents and security baselines). Customers who use this approach typically have a moderate amount of experience with AWS and configuration automation tools, and want to combine the deployment simplicity of a baked image with the deployment flexibility that pulls the latest software builds and updates at launch.

Considerations

Be aware that each change to software that is tightly coupled with the AMI requires a new AMI to be built. Although this approach requires fewer AMIs to maintain than the fully baked AMI approach, it will still result in more AMIs to maintain than the JeOS approach. We therefore recommend that customers who take this approach combine it with an automated AMI build-management system and a configuration management tool.



Create a Default VPC

From the VPC console:

1. Click **Your VPCs**.
2. Click **Actions** and then **Create Default VPC**.
3. Click **Create**.
4. Click **Close**.

Create an EC2 Instance

From the EC2 console:

1. Click **Launch Instance**.
2. Click **Select** next to **Amazon Linux 2 AMI (HVM), SSD Volume Type**, ensuring **64-bit (x86)** is selected.
3. Choose **T3.Micro (NOT T3a.Micro; "a" is AMD)**.
4. Click **Configure Instance Details**.
5. Ensure **Auto-assign Public IP** is **Enabled**.
6. Expand **Advanced Details** and paste in the **User Data** from the lab instructions.
7. Click **Add Storage**.
8. Verify **8GiB, General Purpose SSD** (or more).
9. Click **Add Tags**.
10. Click **Add Tag**, and set **Key** to **Name** and **Value** to **Cat-Hall-o-Fame**.
11. Click **Configure Security Group**.
12. Click **Add Rule**, and set **Type** to **HTTP** and **Source** to **MY IP**.
13. Click **Review and Launch**, and then **Launch**.
14. Select **Create a new key pair**, and call it "ec2lab".
15. Click **Download Key Pair**, **Launch Instances**, and then **View Instances**.

Manage the EC2 Instance



Once the instance is running with 2/2 status checks:

1. Locate the instance public DNS (IPv4), and browse to it.
2. Right-click the instance, and select **Connect**.
3. Run `chmod 400 ec2lab.pem` to adjust the permissions on the file.
 - If using Windows, you will need to follow the [instructions found here](#).
4. Connect to the instance using the `ssh` command provided (or using the PuTTY instructions).
5. Note the `IPv4 Public IP` of the instance.
6. Right-click the instance, **Instance State, Reboot**.
 - Does the IP change?
7. Right-click the instance, **Instance State, Stop**, followed by **Start** when ready.
 - Does the IP change?
8. Right-click the instance, **Instance State, Stop**.
9. Right-click the instance, **Instance Settings**, and change the instance size to **t3.small**.
10. Right-click the instance, **Instance State, Start**.

Launch a Configuration Instance

Launch an instance from a Base AWS Linux AMI:

- Login to the Management Console
- Launch EC2 instance using the AWS Linux AMI:
 - t2.micro
 - Public IP
 - Create a Security Group with SSH (TCP 22) and HTTP (TCP 80) access
 - Download a Key Pair

Install Apache and PHP

SSH to the instance and execute the following commands to install Apache and PHP:

(Windows users see: <https://www.youtube.com/watch?v=bi7ow5NGC-U> for using PuTTY for SSH)

- `sudo yum update -y`
- `sudo yum install -y httpd php`
- `sudo systemctl start httpd`
- `sudo systemctl enable httpd`

Create a PHP page with the following commands:

- `sudo usermod -a -G apache ec2-user`
- `sudo chown -R ec2-user:apache /var/www`
- `echo "<?php phpinfo(); ?>" > /var/www/html/phpinfo.php`

To display the PHP Info page, open a web browser to: [Your EC2 Instance's Public IP]/phpinfo.php



Create a Custom AMI

In the EC2 Management Console, create an image from the config instance:

- Select the instance in the console
- Under the Actions menu, choose Create Image:
 - Name the Image
- Launch an instance from the new Image by going under "MyAMI's"
- Configure the new instance's details as before.
- Verify the PHP page displays in the browser as before (you can use your existing SG that you created).

Create an EBS Snapshot

Create a snapshot from a web application server's EBS volume.

1. Navigate to **EC2 > Instances**.
2. Check the box beside one of the "webserver-instance" instances.
3. Click the link to the **Root device**, and then the EBS volume link.
4. On the **Volumes** page, click **Actions**, and then **Create Snapshot**.
5. Enter a description.
6. Click **Create Snapshot**.

Create a New EBS Volume from a Snapshot



Create a new EBS volume from the snapshot you just created in the previous step. This is how you restore an EBS volume from a snapshot.

1. Navigate to the **Snapshots** page.
2. Check the box beside the snapshot you just created.
3. Click **Actions**, and then **Create Volume**.
4. Change the volume size to 10 GiB.
5. Click **Create Volume**.
6. Click **Close**.

Create Two EC2 AMIs

Amazon Machine Images can be created via several different methods. Create two AMIs following the methods detailed in the video.

Method 1

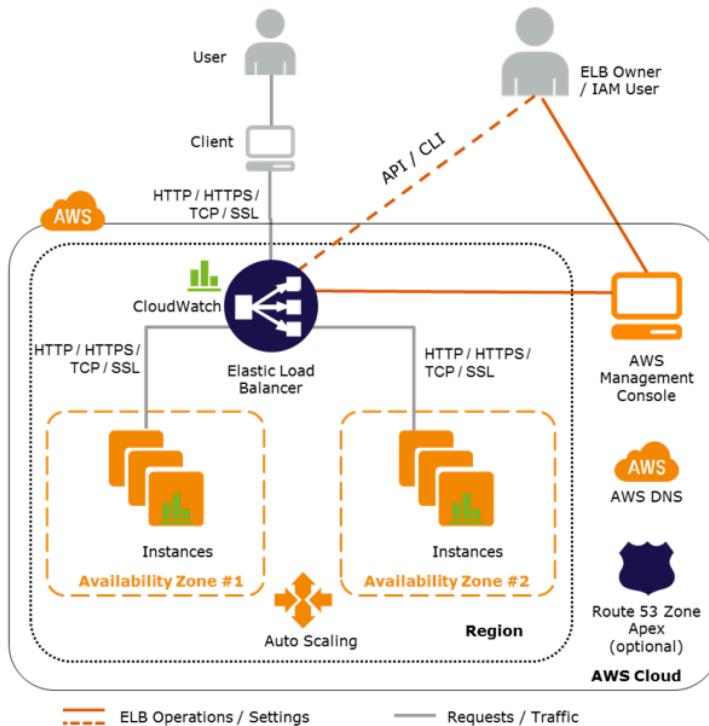
1. Navigate to the **Instances** page.
2. Check the box beside one of the **webserver-instance** instances.
3. Click **Actions**, then **Image**, and then **Create Image**.
4. Add a name and description.
5. Click **Create Image**, and then **Close**.
6. Choose **AMIs** from the left menu to see the image you created.
7. Once the image is available, navigate to the instances page.
8. Click **Launch Instance**.
9. Choose **My AMIs** in the left column.
10. Click **Select**, and then you can configure all the other options for a new instance.
11. For now, click **Cancel**.



Elastic
Load Balancer

ELASTIC
LOAD
BALANCER

ELASTIC LOAD BALANCER



- Elastic Load Balancing automatically distributes incoming application traffic across multiple targets, such as Amazon EC2 instances, containers, IP addresses, and Lambda functions. It can handle the varying load of your application traffic in a single Availability Zone or across multiple Availability Zones.
- Elastic Load Balancing offers three types of load balancers that all feature the high availability, automatic scaling, and robust security necessary to make your applications fault tolerant.

ELASTIC LOAD BALANCER

Elastic Load Balancing supports three types of load balancers: Application Load Balancers, Network Load Balancers (new), and Classic Load Balancers. Choose the load balancer type that meets your needs. [Learn more about which load balancer is right for you](#)

Application Load Balancer



Create

Choose an Application Load Balancer when you need a flexible feature set for your web applications with HTTP and HTTPS traffic. Operating at the request level, Application Load Balancers provide advanced routing and visibility features targeted at application architectures, including microservices and containers.

[Learn more >](#)

Network Load Balancer



Create

Choose a Network Load Balancer when you need ultra-high performance and static IP addresses for your application. Operating at the connection level, Network Load Balancers are capable of handling millions of requests per second while maintaining ultra-low latencies.

[Learn more >](#)

Classic Load Balancer

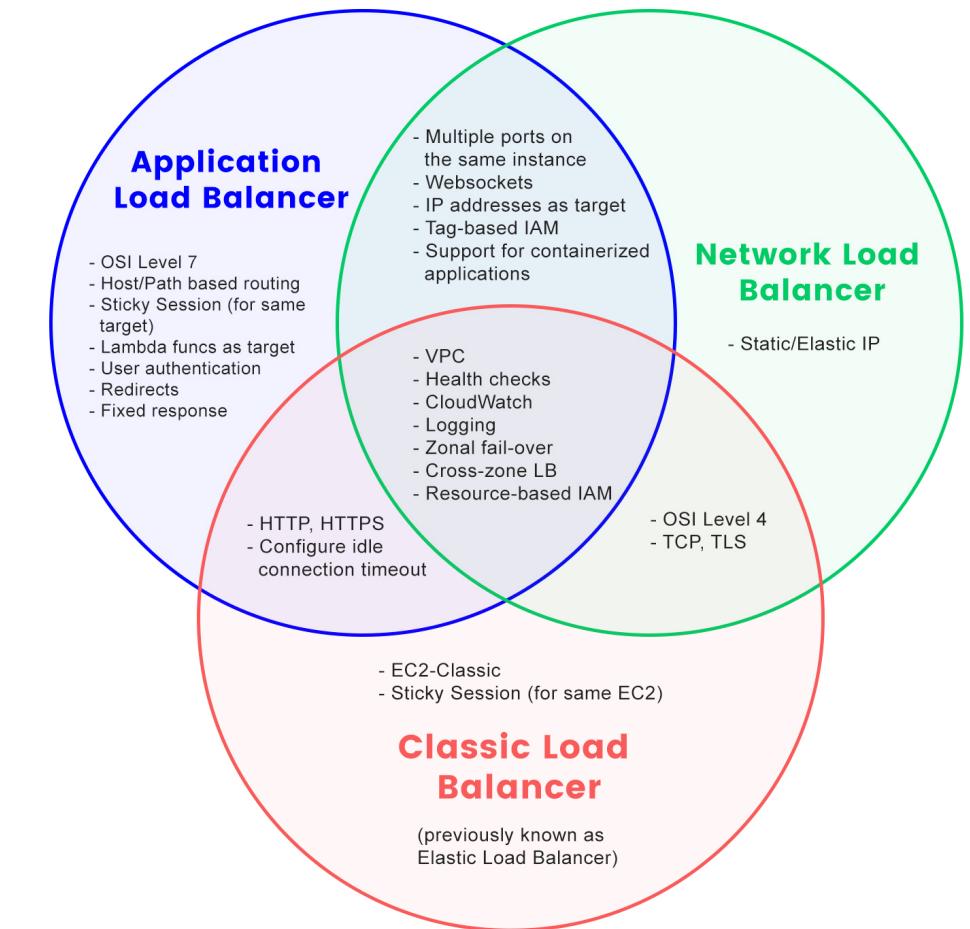
PREVIOUS GENERATION
for HTTP, HTTPS, and TCP

Create

Choose a Classic Load Balancer when you have an existing application running in the EC2-Classic network.

[Learn more >](#)

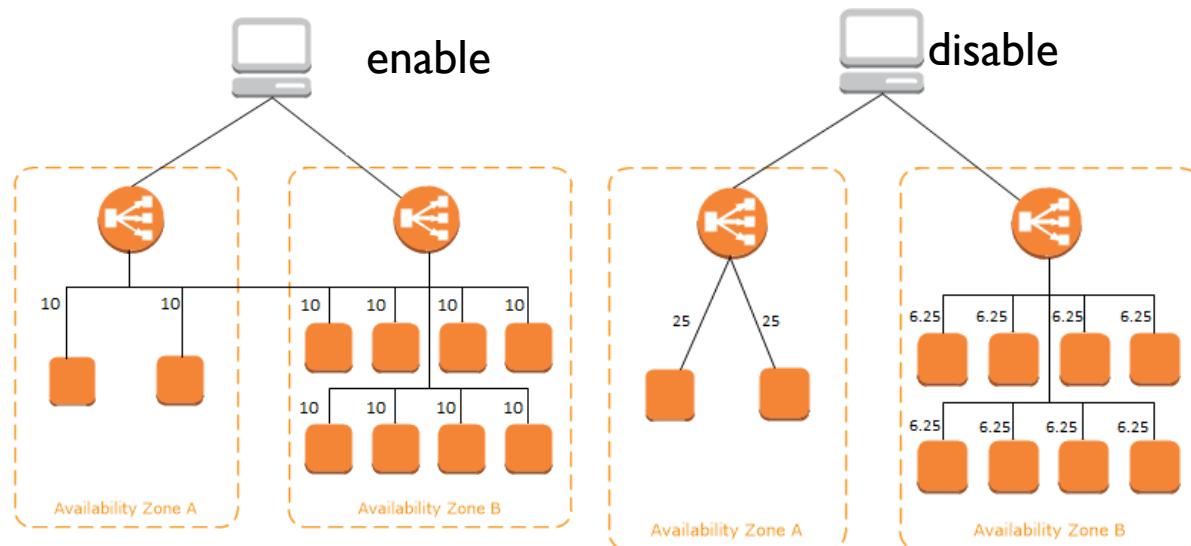
ELB type	CLB	ALB	NLB
Protocols	TCP, SSL/TLS, HTTP, HTTPS	HTTP, HTTPS	TCP, TLS
Performance (a higher number is slower): the ability to handle more traffic	2	3	1 (fastest)
Host/Path-based routing	No	Yes	No
Sticky Session (for session-based applications)	Yes (redirect to the same machine)	Yes (redirect to the same target)	No
Static/Elastic IP	No	No	Yes
Load balancing to multiple ports on the same instance	No	Yes	Yes
Configurable idle connection timeout	Yes	Yes	No



AVAILABILITY ZONES AND LOAD BALANCER NODES

When you enable an Availability Zone for your load balancer, Elastic Load Balancing creates a load balancer node in the Availability Zone. If you register targets in an Availability Zone but do not enable the Availability Zone, these registered targets do not receive traffic. Note that your load balancer is most effective if you ensure that each enabled Availability Zone has at least one registered target.

Cross-Zone Load Balancing



If cross-zone load balancing is disabled, each of the 2 targets in Availability Zone A receives 25% of the traffic and each of the 8 targets in Availability Zone B receives 6.25% of the traffic. This is because each load balancer node can route its 50% of the client traffic only to targets in its Availability Zone.

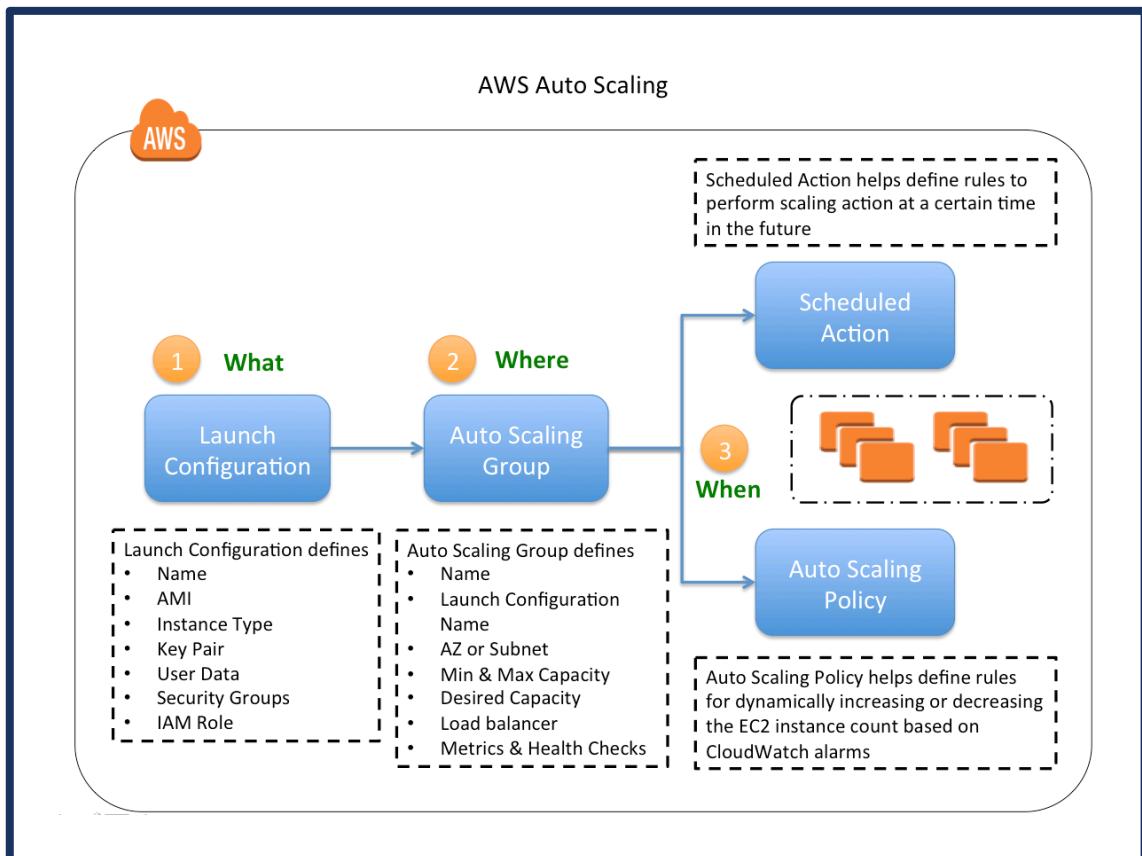
ELB WITH AUTO SCALING GROUP

- Auto Scaling integrates with Elastic Load Balancing and enables to attach one or more load balancers to an existing Auto Scaling group.
- ELB registers the EC2 instance using its IP address and routes requests to the primary IP address of the primary interface (eth0) of the instance.
- After the ELB is attached, it automatically registers the instances in the group and distributes incoming traffic across the instances
- When ELB is detached, it enters the Removing state while deregistering the instances in the group.
- If connection draining is enabled, ELB waits for in-flight requests to complete before deregistering the instances.
- Instances remain running after they are deregistered from the ELB
- Auto Scaling adds instances to the ELB as they are launched, but this can be suspended. Instances launched during the suspension period are not added to load balancer, after resumption, and must be registered manually.
- Auto Scaling group determines the health state of each instance by periodically checking the results of EC2 instance status checks
- Auto Scaling marks the instance as unhealthy and replaces the instance, if the instance fails the EC2 instance status check
- ELB also performs health checks on the EC2 instances that are registered with it for e.g. application is available by pinging and health check page
- Auto Scaling, by default, does not replace the instance, if the ELB health check fails

¿QUÉ ES AWS AUTO SCALING?

- AWS Auto Scaling le permite configurar el escalado automático de los recursos de AWS que forman parte de la aplicación en cuestión de minutos. La consola de AWS Auto Scaling ofrece una interfaz de usuario único que permite usar las características de escalado automático de varios servicios de AWS. Puede configurar el escalado automático para recursos individuales o para aplicaciones completas.
- Con AWS Auto Scaling, la configuración y la administración del escalado de los recursos se lleva a cabo mediante un plan de escalado. Este plan utiliza el escalado dinámico y el escalado predictivo para escalar automáticamente los recursos de la aplicación. De este modo, se garantiza que se añade la potencia de cómputo necesaria para satisfacer la carga en la aplicación y, a continuación, se elimina cuando ya no es necesaria. El plan de escalado le permite elegir las estrategias de escalado para definir cómo optimizar la utilización de recursos. Puede optimizar la disponibilidad, los costos o mantener un equilibrio entre ambos. También puede crear estrategias de escalado personalizadas.
- AWS Auto Scaling resulta útil para aplicaciones que experimentan variaciones diarias o semanales en el flujo de tráfico, incluidas las siguientes:
 - Aplicaciones de tráfico cíclico; por ejemplo, las que experimentan un uso elevado de recursos durante las horas laborables y un uso reducido de recursos por la noche
 - Patrones de carga de trabajo de activación y desactivación, como procesamiento por lotes, pruebas o análisis periódicos
 - Aplicaciones de patrones de tráfico variables, como campañas de marketing con períodos de picos de crecimiento

AUTO SCALING GROUP



- Auto Scaling provides the ability to ensure a correct number of EC2 instances are always running to handle the load of the application
- Auto Scaling helps to achieve better fault tolerance, better availability and cost management
- Auto Scaling also helps specify Scaling policies which can be used to launch and terminate EC2 instances to handle any increase or decrease in demand on the application.
- Auto Scaling attempts to distribute instances evenly between the AZs that are enabled for the Auto Scaling group.
- Auto Scaling does this by attempting to launch new instances in the AZ with the fewest instances. If the attempt fails, Auto Scaling attempts to launch the instances in another Availability Zone until it succeeds

Componentes de Auto Scaling

En la siguiente tabla se describen los principales componentes de Amazon EC2 Auto Scaling.

	<h3>Grupos</h3> <p>Las instancias EC2 se organizan en <i>grupos</i> para que puedan tratarse como una unidad lógica a efectos de escalado y administración. Cuando crea un grupo, puede especificar su número mínimo, máximo y deseado de instancias EC2. Para obtener más información, consulte Grupos de Auto Scaling.</p>
	<h3>Plantillas de configuración</h3> <p>El grupo utiliza una <i>plantilla de lanzamiento</i> o una <i>configuración de lanzamiento</i> como plantilla de configuración para sus instancias EC2. Puede especificar información como el ID de AMI, el tipo de instancia, el par de claves, los grupos de seguridad y el mapeo de dispositivos de bloques para las instancias. Para obtener más información, consulte Plantillas de lanzamiento y Configuraciones de lanzamiento.</p>
	<h3>Opciones de escalado</h3> <p>Amazon EC2 Auto Scaling ofrece varias formas de escalar sus grupo de Auto Scaling. Por ejemplo, puede configurar un grupo para escalarlo basado en la aparición de determinadas condiciones (escalado dinámico) o en una programación. Para obtener más información, consulte Opciones de escalado.</p>

LAUNCH CONFIGURATION

1 What

Launch Configuration

Launch Configuration defines

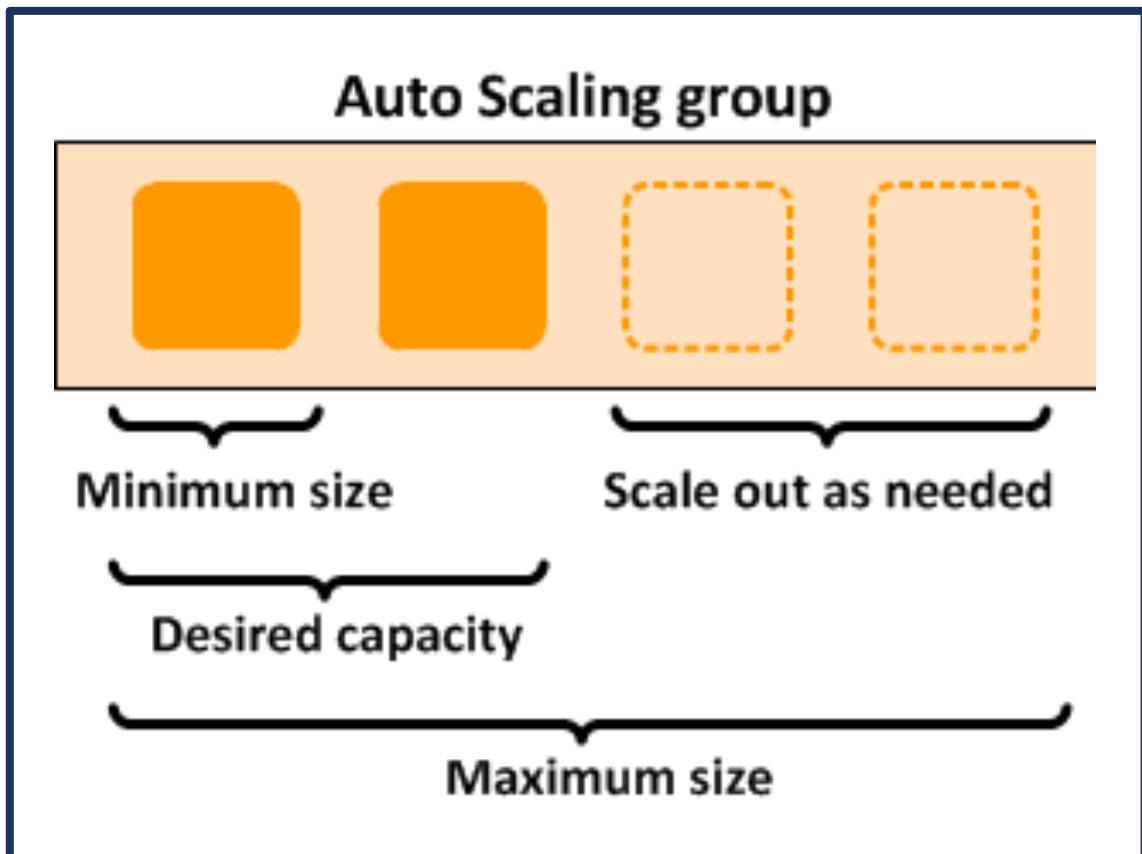
- Name
- AMI
- Instance Type
- Key Pair
- User Data
- Security Groups
- IAM Role

- Launch configuration is a template that an Auto Scaling group uses to launch EC2 instances.
- Launch configuration is similar to EC2 configuration and involves selection of the Amazon Machine Image (AMI), the instance type, a key pair, one or more security groups, and a block device mapping.
- Launch configuration can be associated multiple Auto Scaling groups
- Launch configuration can't be modified after creation and needs to be created new if any modification required
- Basic or Detailed monitoring for the instances in the Auto Scaling group can be enabled when a launch configuration is created.
- By default, basic monitoring is enabled when you create the launch configuration using the AWS Management Console and detailed monitoring is enabled when you create the launch configuration using the AWS CLI or an API

AUTO SCALING POLICY

- Maintain steady count of Instances : numero fijo de instancias
- Manual scaling: se añaden manualmente
- Scheduled scaling : En función de la hora
- Dynamic scaling : en función de la carga
- Multiple Policies : combinado

AUTO SCALING GROUP



- Auto Scaling groups is the core of Auto Scaling and contains a collection of EC2 instances that share similar characteristics and are treated as a logical grouping for the purposes of instance scaling and management.
- Auto Scaling group requires
 - **Launch configuration** to determine the EC2 template to use for launching the instance
 - **Minimum & Maximum capacity** to determine the number of instances when an autoscaling policy is applied. The number of instances cannot grow beyond this boundaries
 - **Desired capacity**
 - to determine the number of instances the ASG must maintain at all times. If missing, it equals to the minimum size.
 - Desired capacity is different from minimum capacity.
 - An Auto Scaling group's desired capacity is the default number of instances that should be running. A group's minimum capacity is the fewest number of instances the group can have running
 - **Availability Zones or Subnets** in which the instances will be launched.
 - **Metrics & Health Checks** – metrics to determine when it should launch or terminate instances and health checks to determine if the instance is healthy or not

- Auto Scaling group starts by launching a desired capacity of instances and maintains this number by performing periodic health checks.
- If an instance becomes unhealthy, it terminates and launches a new instance
- Auto Scaling group can also use scaling policies to increase or decrease the number of instances automatically to meet changing demands
- An Auto Scaling group can contain EC2 instances in one or more AZs within the same region.
- Auto Scaling groups cannot span multiple regions.
- To merge separate single-zone Auto Scaling groups into a single Auto Scaling group spanning multiple AZs, rezone one of the single-zone groups into a multi-zone group, and then delete the other groups. This process works for groups with or without a load balancer, as long as the new multi-zone group is in one of the same AZs as the original single-zone groups.
- Auto Scaling group can be associated with a single launch configuration.
- As the Launch Configuration can't be modified once created, only way to update the Launch Configuration for an Auto Scaling group is to create a new one and associate it with the Auto Scaling group.
- When the launch configuration for the Auto Scaling group is changed, any new instances launched use the new configuration parameters, but the existing instances are not affected.
- Auto Scaling group can be deleted from CLI, if it has no running instances else need to set the minimum and desired capacity to 0. This is handled automatically when deleting an ASG from AWS management console.

Mantener el número de instancias en su grupo de Auto Scaling

Una vez que haya creado su grupo de Auto Scaling, el grupo de Auto Scaling empieza lanzando un número suficiente de instancias EC2 para satisfacer su capacidad mínima (o la capacidad deseada, si se ha especificado). Si no hay otras condiciones de escalado asociadas al grupo de Auto Scaling, el grupo de Auto Scaling mantiene este número de instancias en ejecución en todo momento aunque una instancia deje de estar en buen estado.

Para mantener el mismo número de instancias, Amazon EC2 Auto Scaling realiza una comprobación de estado periódica en las instancias en ejecución dentro de un grupo de Auto Scaling. Cuando encuentra una instancia en mal estado, termina la instancia y lanza una nueva. Si detiene o termina una instancia en ejecución, se considera que la instancia está en mal estado y se sustituye

Comprobaciones de estado de las instancias de Auto Scaling

Una instancia de Auto Scaling puede estar en buen o en mal estado. Todas las instancias de su grupo de Auto Scaling se inician con un estado correcto. Se entiende que las instancias están en buen estado a menos que Amazon EC2 Auto Scaling reciba una notificación de que están en mal estado. Esta notificación puede provenir de una o más de las siguientes fuentes: Amazon EC2, Elastic Load Balancing o una comprobación de estado personalizada.

Cuando Amazon EC2 Auto Scaling marca una instancia como en mal estado, se programa su sustitución. Si no desea que se reemplacen las instancias, puede suspender el proceso de comprobación de estado para cualquier grupo de Auto Scaling.

Estado de una instancia

Amazon EC2 Auto Scaling puede determinar el estado de una instancia a partir de lo siguiente:

- Las comprobaciones de estado proporcionadas por Amazon EC2 para identificar problemas de hardware y software que puedan afectar a una instancia. Entre estas se incluyen las comprobaciones de estado de las instancias y las comprobaciones de estado del sistema.
- Las comprobaciones de estado proporcionadas por Elastic Load Balancing.
- Comprobaciones de estado personalizadas.

Las comprobaciones de estado de EC2 son las comprobaciones de estado predeterminadas de Amazon EC2 Auto Scaling y no requieren ninguna configuración especial. Sin embargo, puede personalizar las comprobaciones de estado personalizadas realizadas por el grupo de Auto Scaling especificando comprobaciones adicionales. Para obtener más información, consulte [Añadir comprobaciones de estado de Elastic Load Balancing a un grupo de Auto Scaling y Comprobaciones de estado personalizadas](#).

Tipos de comprobaciones de estado

Existen dos tipos de comprobaciones de estado: comprobaciones de estado de sistemas y comprobaciones de estado de instancias.

Comprobaciones de estado de sistemas

Monitorice los sistemas de AWS en los que se ejecuta la instancia. Estas comprobaciones detectan problemas subyacentes con la instancia que requieren la implicación de AWS para su reparación. Cuando una comprobación de estado de sistemas falla, puede elegir esperar a que AWS repare el problema o puede resolverlo usted mismo. En el caso de las instancias respaldadas por Amazon EBS, puede detener e iniciar la instancia usted mismo, lo que en la mayoría de los escenarios hace que la instancia migre a un nuevo host. Para instancias respaldadas por un almacén de instancias, puede terminar y reemplazar la instancia.

A continuación se muestran ejemplos de problemas que pueden provocar errores en las comprobaciones de estado del sistema:

- Pérdida de conectividad de red
- Pérdida de potencia del sistema
- Problemas de software en el host físico
- Problemas de hardware en el host físico que afectan a la accesibilidad a la red

Comprobaciones de estado de instancias

Monitorice la configuración de software y de red de la instancia. Amazon EC2 comprueba el estado de la instancia enviando una solicitud del protocolo de resolución de direcciones (ARP) a la interfaz de red (NIC).. Cuando una comprobación de estado de instancias falla, debe resolver el problema por sí mismo (por ejemplo, reiniciando la instancia o realizando cambios en la configuración de la instancia).

A continuación se muestran ejemplos de problemas que pueden provocar errores en las comprobaciones de estado de la instancia:

- Error de las comprobaciones de estado del sistema
- Configuración de red o de inicio incorrecta
- Memoria agotada
- Sistema de archivos dañado
- Kernel incompatible

[Description](#) [Status Checks](#) [Monitoring](#) [Tags](#)

Status checks detect problems that may impair this instance from running your applications. [Learn more](#) about status checks.

[Create Status Check Alarm](#)

System Status Checks ⓘ

These checks monitor the AWS systems required to use this instance and ensure they are functioning properly.

[System reachability check passed](#)

Instance Status Checks ⓘ

These checks monitor your software and network configuration for this instance.

[Instance reachability check failed at October 7, 2013 11:52:11 AM UTC+2 \(16 minutes ago\)](#)

[Learn more about this issue](#)

Additional Resources

[Submit feedback](#) if our checks do not reflect your experience with this instance or if they do not detect the issues you are having.

Please note that we will not respond to customer support issues reported via this form. Please post your issue on the [Developer Forums](#) or contact [AWS Support](#) if you need technical assistance with this instance.

¿Qué es una estrategia de escalado?

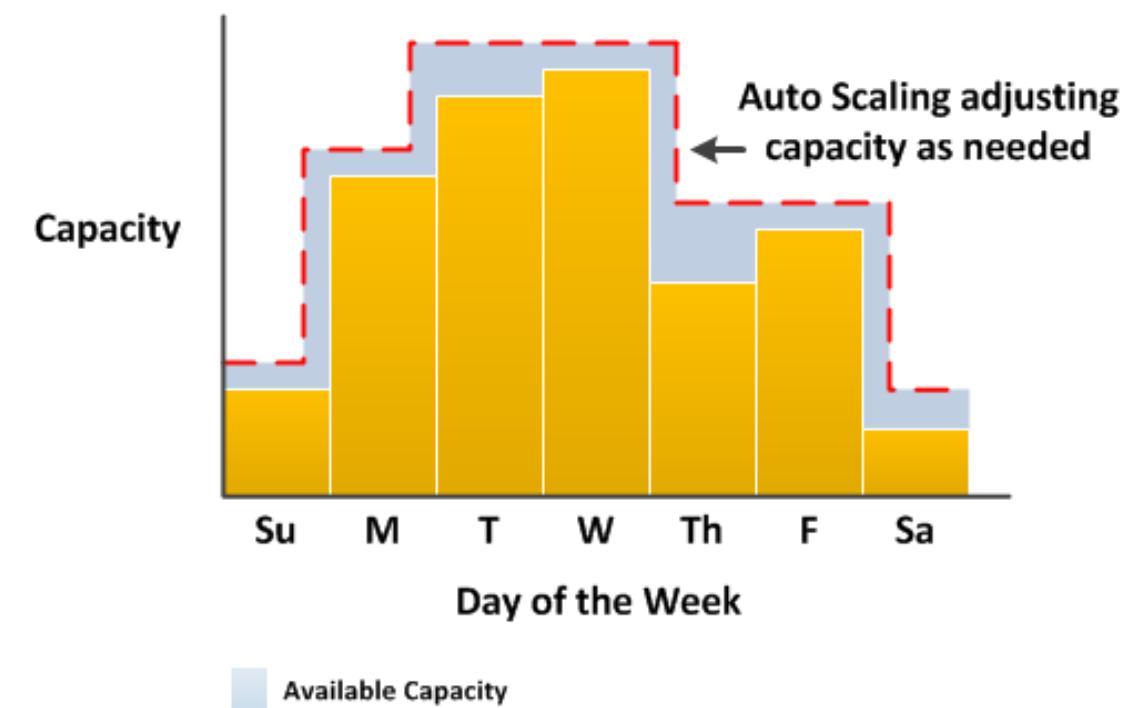
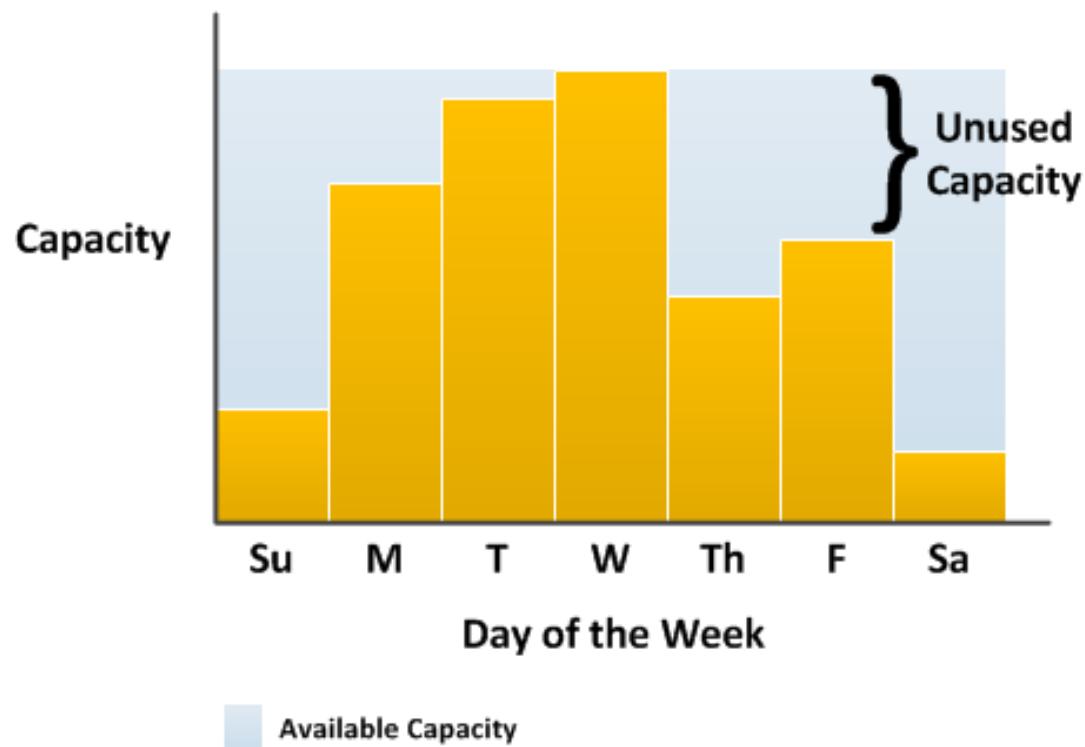
Para cada tipo de recurso, AWS Auto Scaling elige la métrica que se usa habitualmente para determinar la cantidad del recurso que está en uso en un momento dado. Usted elige la estrategia de escalado más apropiada para optimizar el rendimiento de la aplicación en función de esta métrica. Cuando habilita la característica de escalado dinámico y la característica de escalado predictivo, la estrategia de escalado se comparte entre ellas. Para obtener más información, consulte [Funcionamiento de AWS Auto Scaling](#).

Las siguientes estrategias de escalado están disponibles:

- **Optimize for availability (Optimizar para disponibilidad)** — AWS Auto Scaling escala el recurso de forma ascendente o descendente para mantener la utilización del recurso al 40 por ciento. Esta opción es útil cuando la aplicación tiene necesidades de escalado urgentes y a veces impredecibles.
- **Balance availability and cost (Equilibrar disponibilidad y costo)** — AWS Auto Scaling escala el recurso de forma ascendente o descendente para mantener la utilización del recurso al 50 por ciento. Esta opción le ayuda a mantener una alta disponibilidad y a reducir los costos al mismo tiempo.
- **Optimize for cost (Optimizar para costo)** — AWS Auto Scaling escala el recurso de forma ascendente o descendente para mantener la utilización del recurso al 70 por ciento. Esta opción es útil para reducir los costos si la aplicación es capaz de mantener una capacidad del búfer reducida cuando se producen cambios inesperados en la demanda.

¿Qué es el escalado dinámico?

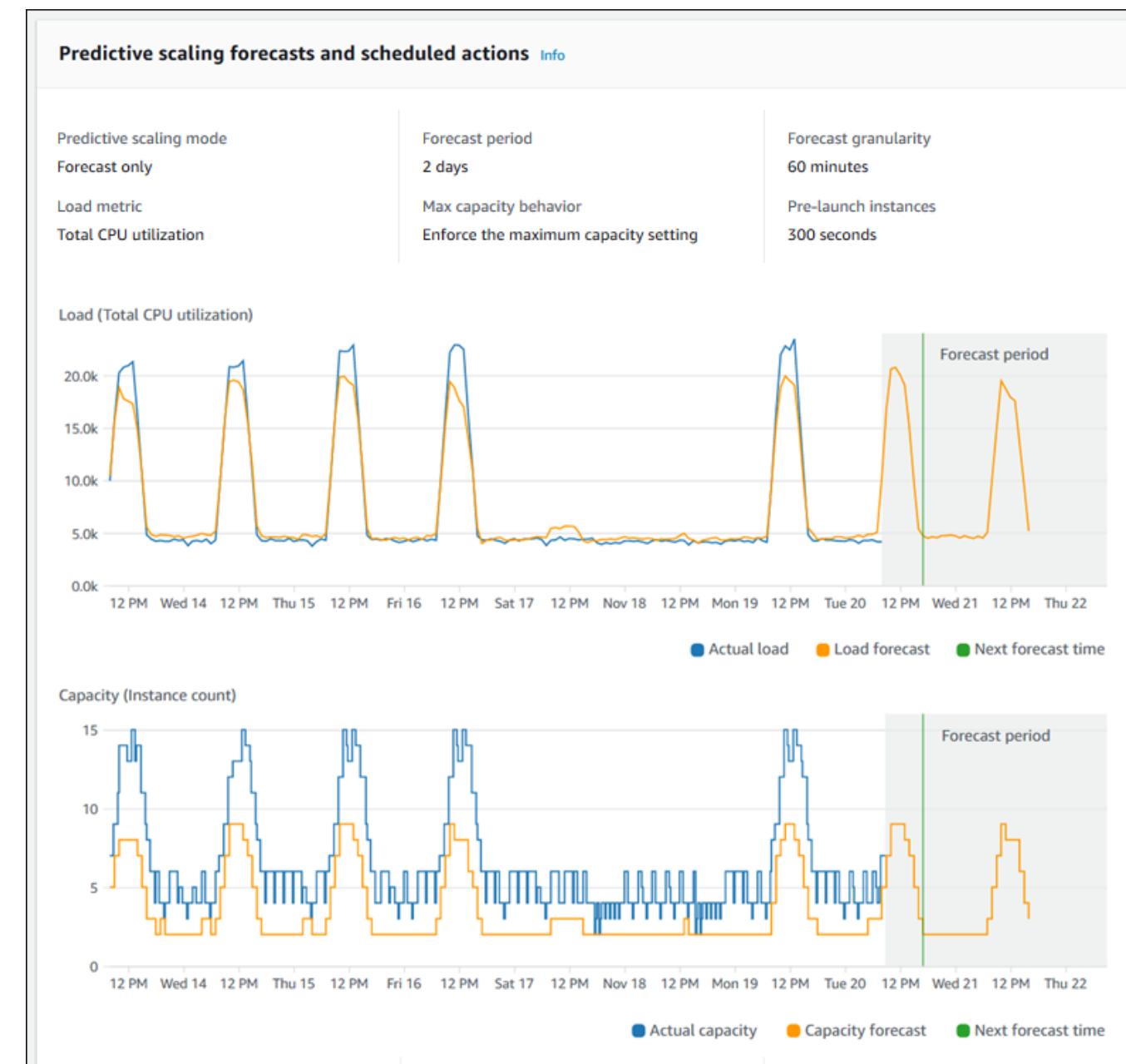
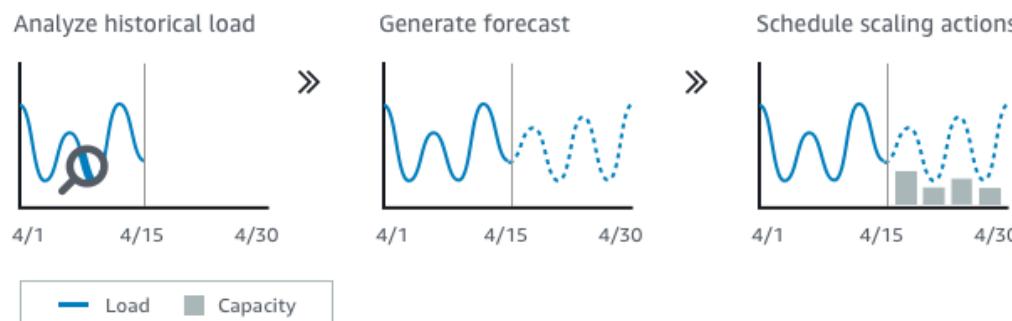
El escalado dinámico crea políticas de escalado de seguimiento de destino para los recursos del plan de escalado. Estas políticas de escalado ajustan la capacidad de los recursos en respuesta a los cambios en la utilización de los recursos. El objetivo es proporcionar suficiente capacidad para mantener la utilización de recursos en el valor de destino especificado por la estrategia de escalado. Se asemeja a los termostatos que se utilizan para mantener la temperatura del hogar. Se elige la temperatura y el termostato hace el resto.



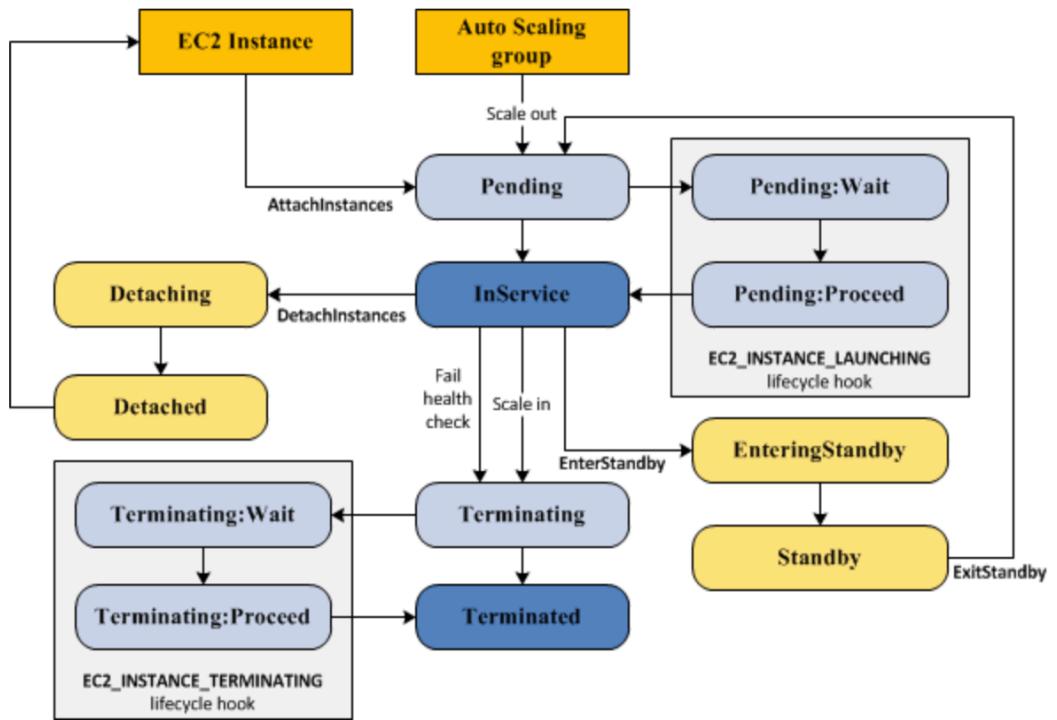
¿Qué es el escalado predictivo?

El escalado predictivo utiliza el aprendizaje automático para analizar la carga de trabajo histórica de cada uno de los recursos y prevé de forma periódica la carga futura para los próximos dos días, de forma similar a como lo hacen las previsiones meteorológicas.

Utilizando la previsión, genera acciones de escalado programadas para garantizar que la capacidad de los recursos esté disponible antes de que la aplicación la necesite. Al igual que el escalado dinámico, el escalado predictivo mantiene la utilización en el valor objetivo especificado por la estrategia de escalado.



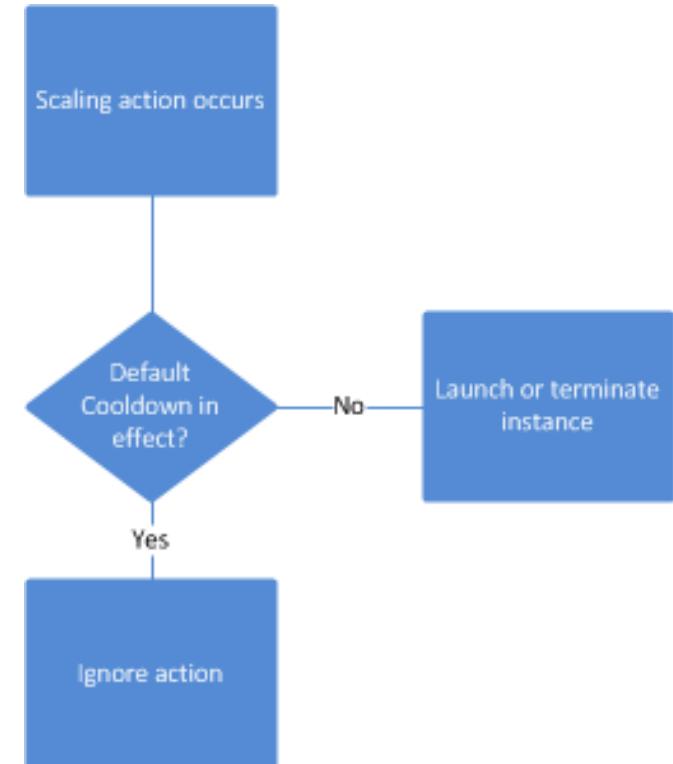
AUTO SCALING GROUP



- Instances launched through Auto Scaling group have a different lifecycle than that of other EC2 instances
- Auto Scaling lifecycle starts when the Auto Scaling group launches an instance and puts it into service.
- Auto Scaling lifecycle ends when the instance is terminated either by the user, or the Auto Scaling group takes it out of service and terminates it
- AWS charges for the instances as soon as they are launched, including the time it is not in InService

Auto Scaling Cooldown

- Auto Scaling cooldown period is a configurable setting for your Auto Scaling group that helps to ensure that Auto Scaling doesn't launch or terminate additional instances before the previous scaling activity takes effect and allows the newly launched instances to start handling traffic and reduce load.
- When Auto Scaling group dynamically scales using a simple scaling policy and launches an instance, Auto Scaling suspends the scaling activities for the cooldown period (default 300 seconds) to complete before resuming scaling activities.
- When manually scaling the Auto Scaling group, the default is not to wait for the cooldown period, but you can override the default and honor the cooldown period.
- Note that if an instance becomes unhealthy, Auto Scaling does not wait for the cooldown period to complete before replacing the unhealthy instance.
- Cooldown periods are automatically applied to dynamic scaling activities for simple scaling policies and is not supported for step scaling policies.

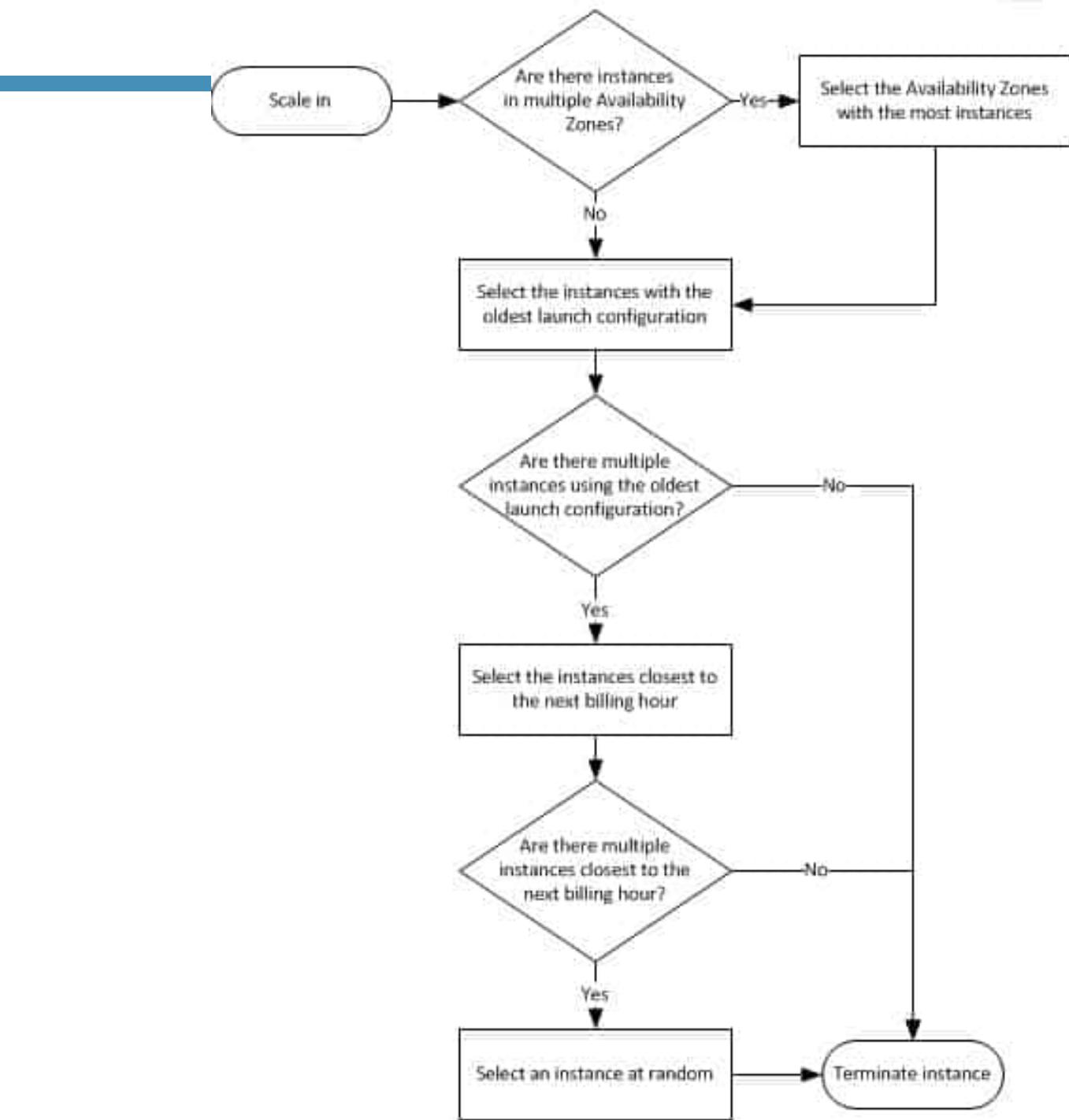
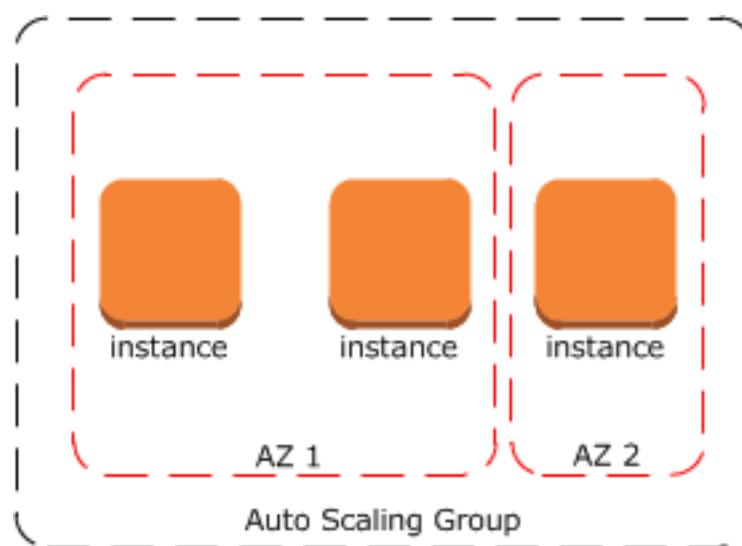


Termination Policy

Termination policy helps the Auto Scaling to decide which instances it should terminate first when Auto Scaling automatically scales in.

Default Termination Policy

- Next diagram



Customized Termination Policy

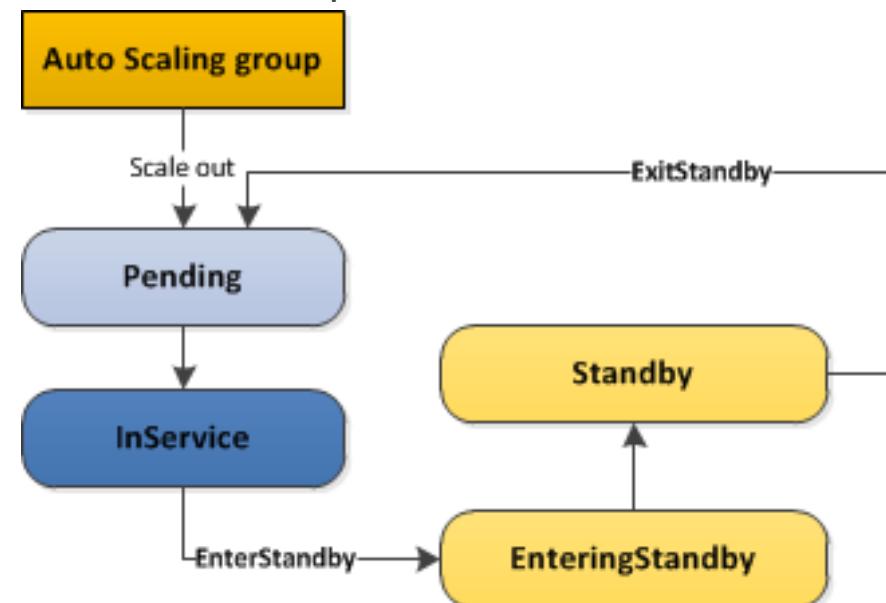
- Auto Scaling first assesses the AZs for any imbalance. If an AZ has more instances than the other AZs that are used by the group, then it applies the specified termination policy on the instances from the imbalanced AZ
- If the Availability Zones used by the group are balanced, then Auto Scaling applies the termination policy that you specified.
- Following Customized Termination policies are supported
 - OldestInstance – terminates the oldest instance in the group and can be useful to upgrade to new instance types
 - NewestInstance – terminates the newest instance in the group and can be useful when testing a new launch configuration
 - OldestLaunchConfiguration – terminates instances that have the oldest launch configuration
 - ClosestToNextInstanceHour – terminates instances that are closest to the next billing hour and helps to maximize the use of your instances and manage costs.
 - Default – terminates as per the default termination policy

Eliminar temporalmente instancias de su grupo de Auto Scaling

- Puede poner una instancia que tiene el estado InService en el estado Standby, actualizar o resolver los problemas de la instancia y, a continuación, poner de nuevo la instancia en servicio. Las instancias que están a la espera siguen formando parte del grupo de Auto Scaling, pero no controlan activamente el tráfico de las aplicaciones.
- Por ejemplo, puede cambiar la configuración de lanzamiento de un grupo de Auto Scaling en cualquier momento, y todas las instancias posteriores que lance el grupo de Auto Scaling utilizarán esta configuración. Sin embargo, el grupo de Auto Scaling no actualiza las instancias que están actualmente en servicio. Puede terminar estas instancias y dejar que el grupo de Auto Scaling las sustituya. O bien, puede poner las instancias en espera, actualizar el software y, a continuación, volver a poner las instancias en servicio.

importante

- Se le cobrarán las instancias que se encuentran en estado de espera.







Create an Application Load Balancer

^

Create an Application Load Balancer to load balance between EC2 instances you will create later on inside your Auto Scaling group:

1. Navigate to the EC2 portion of the console.
2. Click on the **Load Balancers** section under **Load Balancing**.
3. Press the **Create** button under the **Application Load Balancer**.
4. Name your load balancer **LABALB**, leave the ALB set to **internet facing**, and set the IP address type as **ipv4**.
5. Select the VPC, and add the **us-east-1a** and **us-east-1b** AZs to your ALB (**not** the **us-east-1c** AZ).
6. Create a new security group for your ALB, and use **ALBSG** for the name and description. Configure rules ensuring HTTP is allowed from 0.0.0.0/0 and ::/0 (IPV6).
7. Configure a target group for your ALB, naming it **ALBTG**.
8. Expand **Advanced health check settings**, and reduce the healthy threshold check down to 2.
9. Proceed to create your ALB.

Create a Launch Template



Create an SSH key pair that the EC2 instances will use to control access:

1. Navigate to **EC2 > Network & Security > Key Pairs**.
2. Click **Create Key Pair**.
3. Call it **ALBASG**, and download the file to your local machine.

Create a security group for EC2 instances:

1. Navigate to **EC2 > Network & Security > Security Groups**.
2. Click **Create Security Group**.
3. The name and description are **ec2web**.
4. Set the VPC to the lab VPC.
5. Add a rule allowing SSH from 0.0.0.0/0 and ::/0 (IPV6).
6. Add a rule allowing HTTP from the Security Group ID of the ALB.
7. Create the security group.

Create a launch template that will be used by the Auto Scaling group:

1. Navigate to **EC2 > Instances > Launch Templates**.
2. Create a new template, and call it **LABLT** for the name and description.
3. The **Source Template** is none.
4. Search for AMI, and pick the Amazon Linux 2 AMI (64-bit x86).
5. Set the instance type as **t3.micro** (not T3a.micro).
6. Select the key pair created earlier.
7. Network type is VPC.
8. Select the security group created earlier.
9. Storage should automatically be populated with a volume, so leave that as default and don't add anything to the network section.
10. Expand **Advanced Details**, and paste the user data for this lesson in the box.
11. Click **Create Launch Template**.
12. Click **Close**.

Create an Auto Scaling Group



1. **EC2 > Auto Scaling > Auto Scaling Groups**
2. Click **Create Auto Scaling group**.
3. Select **Launch Template**, and choose the template you just created.
4. Call the group **LABASG**.
5. Start with two instances.
6. Pick the VPC from the lab environment, and select **us-east-1a** and **us-east-1b** as subnets.
7. Click **Next: Configure Scaling Policies**.
8. For now, keep the group at the initial size.
9. Click **Next: Configure Notifications > Next: Configure Tags > Review > Create Auto Scaling group > Close**.

Enable Group Metrics Collection

1. Click the **Monitoring** tab of the ASG.
2. Click **Enable Group Metrics Collection**.

the ASG to Allow Scaling and Use the ALB

1. Click **Actions > Edit**
2. Set **Max instances** as **6***, **click *Target groups**, and pick the target group.
3. Click **Save**.

Configure Auto Scaling Group Scaling Policies

1. Select the **Scaling Policies** tab of the ASG.
2. Click **Add policy**.
3. Click **Create a simple scaling policy**.
4. Name it **SCALEOUT**, set it to **Take the action** to **Add 1 instances, And then wait 300** seconds before allowing another scaling activity.
5. Click **Create new alarm**, and uncheck notification.
6. Average CPU Utilization is \geq 40 percent for 1 period of 5 minutes. Call it **HIGHCPU**.
7. Click **Create alarm** and **Close**.
8. Click **Create**.
9. Click **Add policy** again.
10. Click **Create a simple scaling policy**.
11. Name it **SCALEIN**, set it to **Take the action** to **Remove 1 instances, And then wait 300** seconds before allowing another scaling activity.
12. Click **Create new alarm**, and uncheck notification.
13. Average CPU utilization is \leq 20 percent for 1 period of 5 minutes. Call it **LOWCPU**.
14. Click **Create alarm > Close > Create**.

Test Horizontal Scaling

1. Connect to one of the EC2 instances.
2. Install the stress test application:

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

3. Run the stress test on the EC2 instance:

```
stress --cpu 2 --timeout 300
```

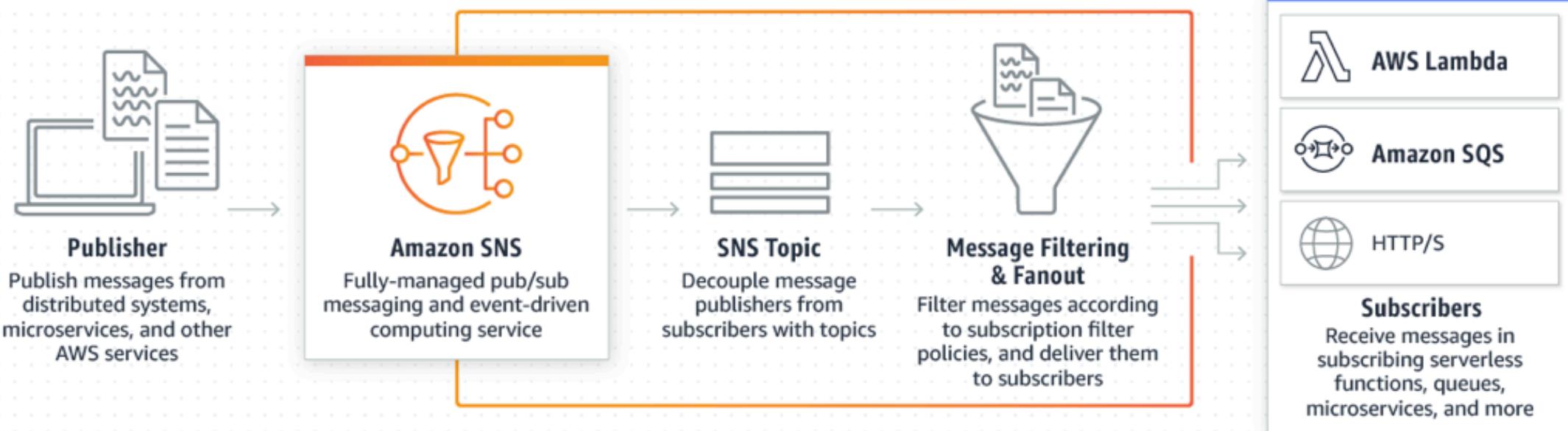
(Optionally, use **3000** for timeout.)

4. After a few minutes, watch the number of instances increase.

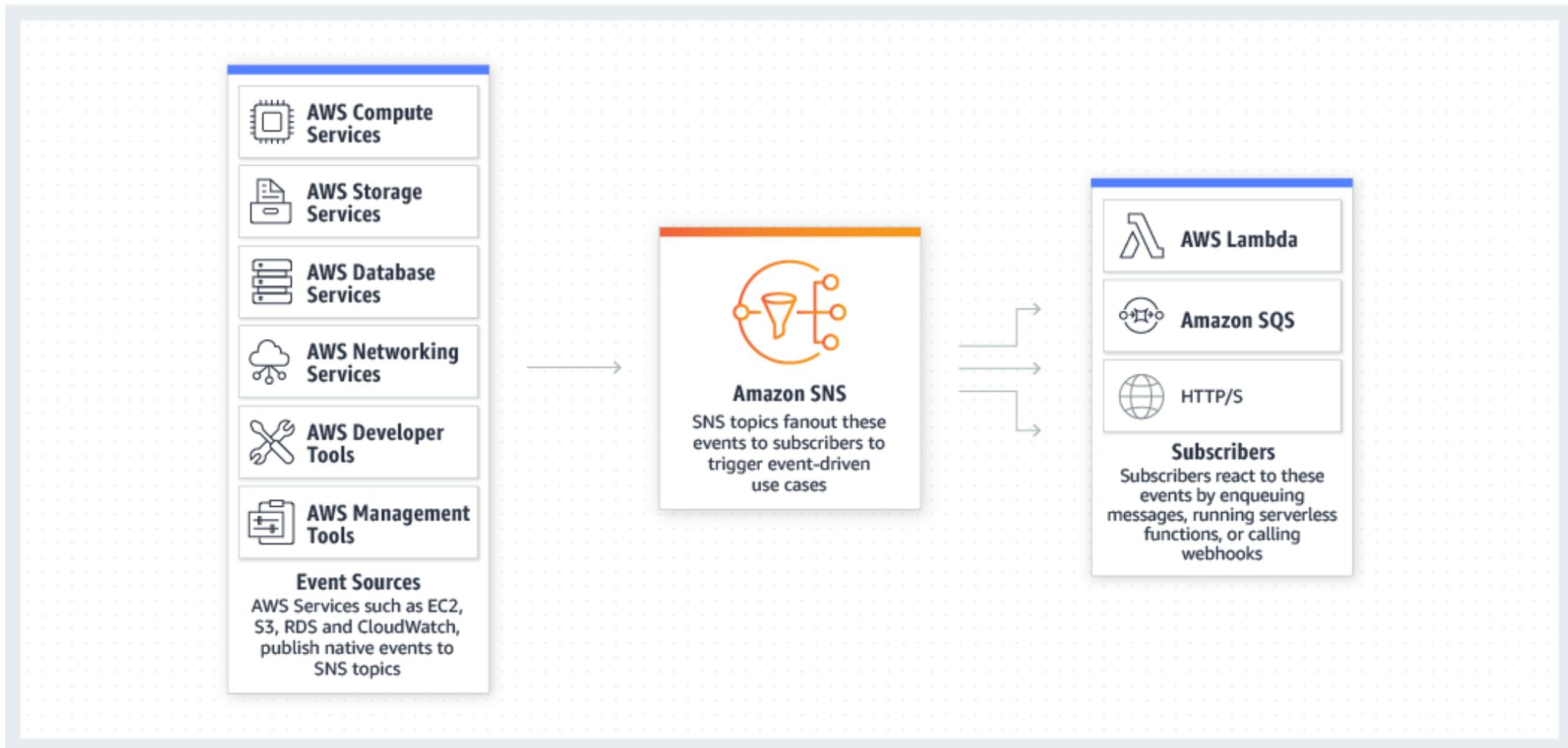
SIMPLE NOTIFICATION SERVICE (SNS)

- Amazon Simple Notification Service (Amazon SNS) is a web service that makes it easy to set up, operate, and **send notifications from the cloud**. It provides developers with a highly scalable, flexible, and cost-effective capability to publish messages from an application and immediately deliver them to subscribers or other applications. It is designed to make web-scale computing easier for developers. Amazon SNS follows the “**publish-subscribe**” (**pub-sub**) **messaging paradigm**, with notifications being delivered to clients using a “push” mechanism that eliminates the need to periodically check or “poll” for new information and updates. With simple APIs requiring minimal up-front development effort, no maintenance or management overhead and pay-as-you-go pricing, Amazon SNS gives developers an easy mechanism to incorporate a powerful notification system with their applications.

SIMPLE NOTIFICATION SERVICE (SNS)



SIMPLE NOTIFICATION SERVICE (SNS)



SIMPLE NOTIFICATION SERVICE (SNS)

Amazon Simple Notification Service (SNS)



Amazon SNS enables you to **set up, operate, and send notifications** to subscribing services other applications.

- 💡 Messages published to topic.
- 💡 Topic subscribers receive message.

Subscriber types:

- 💡 Email (plain or JSON)
- 💡 HTTP/HTTPS
- 💡 Short Message Service (SMS) clients (USA only)
- 💡 Amazon SQS queues
- 💡 Mobile push messaging
- 💡 AWS Lambda Function

SIMPLE NOTIFICATION SERVICE (SNS)

- Amazon SNS offers several benefits making it a versatile option for building and integrating loosely-coupled, distributed applications:
 - Instantaneous, **push-based delivery (no polling)**
 - Simple APIs and easy integration with applications
 - Flexible **message delivery over multiple transport protocols**
 - Inexpensive, pay-as-you-go model with no up-front costs
 - Web-based AWS Management Console offers the simplicity of a point-and-click interface

SIMPLE NOTIFICATION SERVICE (SNS)

- It is very easy to get started with Amazon SNS. **Developers must first create a “topic” which is an “access point” – identifying a specific subject or event type – for publishing messages and allowing clients to subscribe for notifications.** Once a topic is created, the topic owner can set policies for it such as limiting who can publish messages or subscribe to notifications, or specifying which notification protocols will be supported (i.e. HTTP/HTTPS, email, SMS). Subscribers are clients interested in receiving notifications from topics of interest; they can subscribe to a topic or be subscribed by the topic owner. Subscribers specify the protocol and end-point (URL, email address, etc.) for notifications to be delivered. When publishers have information or updates to notify their subscribers about, they can publish a message to the topic – which immediately triggers Amazon SNS to deliver the message to all applicable subscribers.

SIMPLE NOTIFICATION SERVICE (SNS)

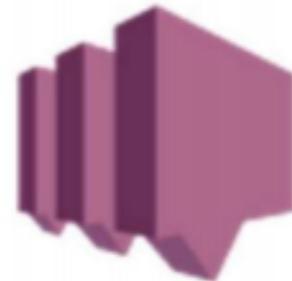
- In order for customers to have broad flexibility of delivery mechanisms, Amazon SNS supports notifications over multiple transport protocols. Customers can select one the following transports as part of the subscription requests:
 - “HTTP”, “HTTPS” – Subscribers specify a URL as part of the subscription registration; notifications will be delivered through an HTTP POST to the specified URL.
 - “Email”, “Email-JSON” – Messages are sent to registered addresses as email. Email-JSON sends notifications as a JSON object, while Email sends text-based email.
 - “SQS” – Users can specify an SQS standard queue as the endpoint; Amazon SNS will enqueue a notification message to the specified queue (which subscribers can then process using SQS APIs such as ReceiveMessage, DeleteMessage, etc.). Note that FIFO queues are not currently supported.
 - “SMS” – Messages are sent to registered phone numbers as SMS text messages.

Lab

SIMPLE NOTIFICATION SERVICE (SNS)

Characteristics Of Amazon SNS

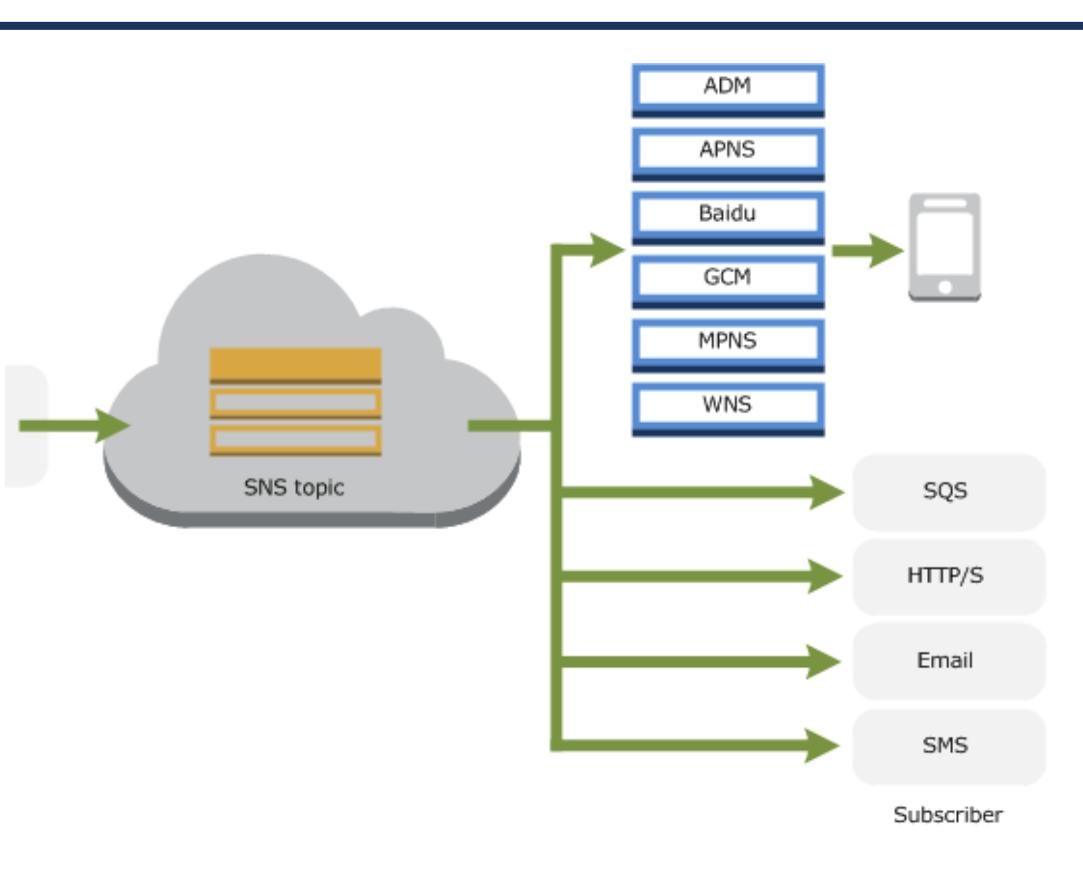
- 📦 Single published message
- 📦 Order is not guaranteed
- 📦 No recall
- 📦 HTTP/HTTPS retry
- 📦 256 KB max per message



SIMPLE QUEUE SERVICE (SQS)

- Amazon SQS is a **message queue service** used by **distributed applications** to exchange messages through a **polling model**, and can be used to decouple sending and receiving components.
- A queue is a temporary repository for messages awaiting for processing and acts as a buffer between the component producer and the consumer
- Amazon SQS
 - offers a reliable, highly-scalable, hosted queue for storing messages in transit between computers
 - provides fault tolerant, loosely coupled, flexibility of distributed components of applications to send & receive without requiring each component to be concurrently available
 - helps build distributed application with decoupled components
 - requires no administrative overhead and little configuration
 - supports the HTTP over SSL (HTTPS) and Transport Layer Security (TLS) protocols for security
- SQS provides two types of Queues – Standard & FIFO

SIMPLE QUEUE SERVICE (SQS)



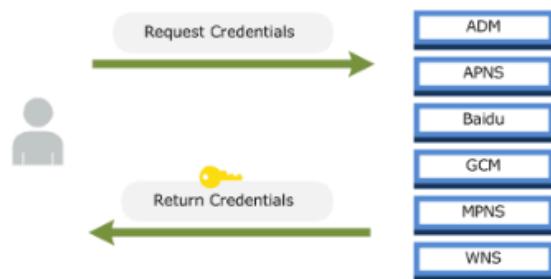
- **Cómo funcionan las notificaciones de usuario**
- Los mensajes de notificaciones de inserción se envían a dispositivos móviles y a escritorios utilizando uno de los siguientes servicios de notificaciones de inserción compatibles:
 - Amazon Device Messaging (ADM)
 - Apple Push Notification Service (APNs) para iOS y Mac OS X
 - Baidu Cloud Push (Baidu)
 - Mensajería de la nube de Firebase (FCM)
 - Microsoft Push Notification Service for Windows Phone (MPNS)
 - Windows Push Notification Services (WNS)

Información general del proceso

Para comenzar con la inserción en móvil de Amazon SNS, primero debe [obtener las credenciales y el token del dispositivo](#) para las plataformas móviles que desea admitir. A continuación, utilizando la información de las plataformas móviles, puede utilizar Amazon SNS para [publicar un mensaje en un dispositivo móvil](#).

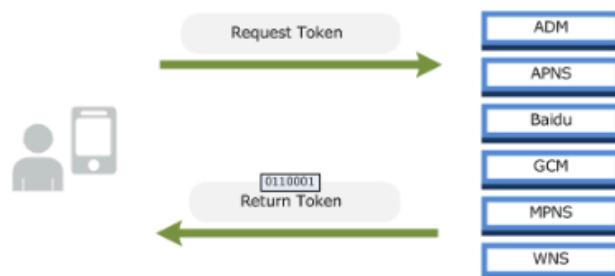
Paso 1: Solicitar credenciales a las plataformas móviles

Para utilizar la inserción en móviles de Amazon SNS, primero debe solicitar las credenciales necesarias a las plataformas móviles.



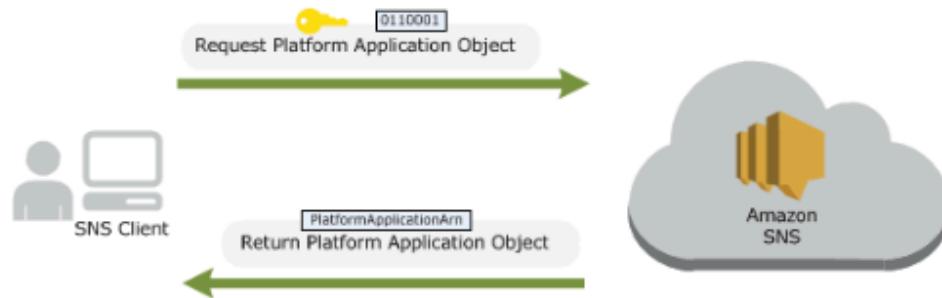
Paso 2: Solicitar un token a las plataformas móviles

Seguidamente, deberá utilizar las credenciales obtenidas para solicitar un token para su aplicación móvil y el dispositivo a las plataformas móviles. El token que reciba representa su aplicación móvil y el dispositivo.



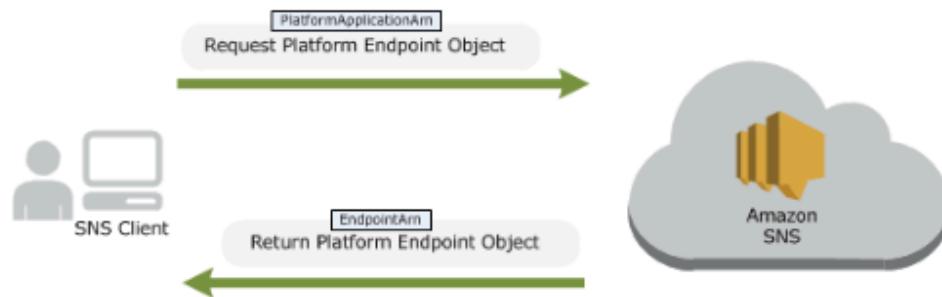
Paso 3: Crear un objeto de aplicación de plataforma

Las credenciales y el token se utilizan para obtener un objeto de aplicación de plataforma (`PlatformApplicationArn`) de Amazon SNS. Para obtener más información, consulte [Creación de un punto de enlace de plataforma y administración de tokens de dispositivos](#).



Paso 4: Crear un objeto de punto de enlace de plataforma

El valor de `PlatformApplicationArn` se utiliza para obtener un objeto de punto de enlace de plataforma (`EndpointArn`) de Amazon SNS. Para obtener más información, consulte [Creación de un punto de enlace de plataforma y administración de tokens de dispositivos](#).



Paso 5: Publicar un mensaje en el punto de enlace móvil

El valor EndpointArn se utiliza para publicar un mensaje en una aplicación de un dispositivo móvil. Para obtener más información, consulte [Envío de un mensaje directo a un dispositivo móvil](#) y la API [Publish](#) de la Amazon Simple Notification Service API Reference.



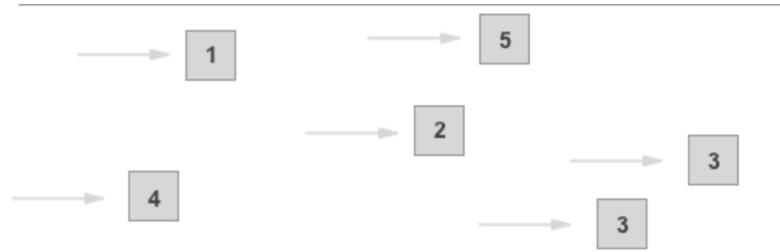
SIMPLE QUEUE SERVICE (SQS)

Standard Queue

High Throughput: Standard queues have nearly-unlimited transactions per second (TPS).

At-Least-Once Delivery: A message is delivered at least once, but occasionally more than one copy or a message is delivered.

Best-Effort Ordering: Occasionally, messages are delivered in an order different from which they were sent.



Send data between applications when the throughput is important, for example:

- Decouple live user requests from intensive background work: let users upload media while resizing or encoding it.
- Allocate tasks to multiple worker nodes: process a high number of credit card validation requests.
- Batch messages for future processing: schedule multiple entries to be added to a database.

FIFO Queue

First-In-First-out Delivery: The order in which messages are sent and received is strictly preserved.

Exactly-Once Processing: A message is guaranteed to be delivered at least once, but all duplicates of the message are removed.

Limited Throughput: 300 transactions per second (TPS).



Send data between applications when the order of events is important, for example:

- Ensure that user-entered commands are executed in the right order.
- Display the correct product price by sending price modifications in the right order.
- Prevent a student from enrolling in a course before registering for an account.

SIMPLE QUEUE SERVICE (SQS)

Amazon Simple Queue Service (SQS)



Amazon SQS is a fully managed message queueing service. Transmit any volume of messages at any level of throughput without losing messages or requiring other services to be always available.

Messages



- Generated by one component to be consumed by another.
- Can contain 256 KB of text in any format.

Amazon SQS



- Ensures delivery of each message at least once.
- Supports multiple readers and writers on the same queue.
- Does not guarantee first in, first out.

Queues



- Repository for messages awaiting processing.
- Acts as a buffer between the components which produce and receive data.

HOW SQS QUEUES WORKS

- SQS allows queues to be created, deleted and messages can be sent and received from it
- SQS queue retains messages for four days, by default.
- Queues can configured to retain messages for 1 minute to 14 days after the message has been sent.
- SQS can delete a queue without notification if one of the following actions hasn't been performed on it for 30 consecutive days.
- SQS allows the deletion of the queue with messages in it

SQS – STANDARD QUEUES

- **Redundant infrastructure**

- offers reliable and scalable hosted queues for storing messages
- is engineered to always be available and deliver messages
- provides ability to store messages in a fail safe queue
- highly concurrent access to messages

- **At-Least-Once delivery**

- **ensures delivery of each message at least once**
- might deliver duplicate copy of messages, if the servers storing a copy of a message is unavailable when you receive or delete the message and the copy of the message is not deleted on that unavailable server
- Applications should be designed to be idempotent with the ability to handle duplicate messages and not be adversely affected if it processes the same message more than once

- **Message Attributes**

- SQS message can contain up to 10 metadata attributes.
- take the form of name-type-value triples
- can be used to separate the body of a message from the metadata that describes it.
- helps process and store information with greater speed and efficiency because the applications don't have to inspect an entire message before understanding how to process it

SQS – STANDARD QUEUES

- **Order**
 - makes a best effort to preserve order in messages does not guarantee first in, first out delivery of messages (SQS now offers FIFO queues which maintain order and Exactly-Once Processing)
 - can be handled by placing sequencing information within the message and performing the ordering on the client side
- **Loose coupling**
 - removes tight coupling between components
 - provides the ability to move data between distributed components of the applications that perform different tasks without losing messages or requiring each component to be always available
- **Multiple writers and readers**
 - supports multiple readers and writers interacting with the same queue at the same time
 - locks the message during processing, using Visibility Timeout, preventing it to be processed by any other consumer

SQS – STANDARD QUEUES

- **Order**
 - makes a best effort to preserve order in messages does not guarantee first in, first out delivery of messages (SQS now offers FIFO queues which maintain order and Exactly-Once Processing)
 - can be handled by placing sequencing information within the message and performing the ordering on the client side
- **Loose coupling**
 - removes tight coupling between components
 - provides the ability to move data between distributed components of the applications that perform different tasks without losing messages or requiring each component to be always available
- **Multiple writers and readers**
 - supports multiple readers and writers interacting with the same queue at the same time
 - locks the message during processing, using Visibility Timeout, preventing it to be processed by any other consumer

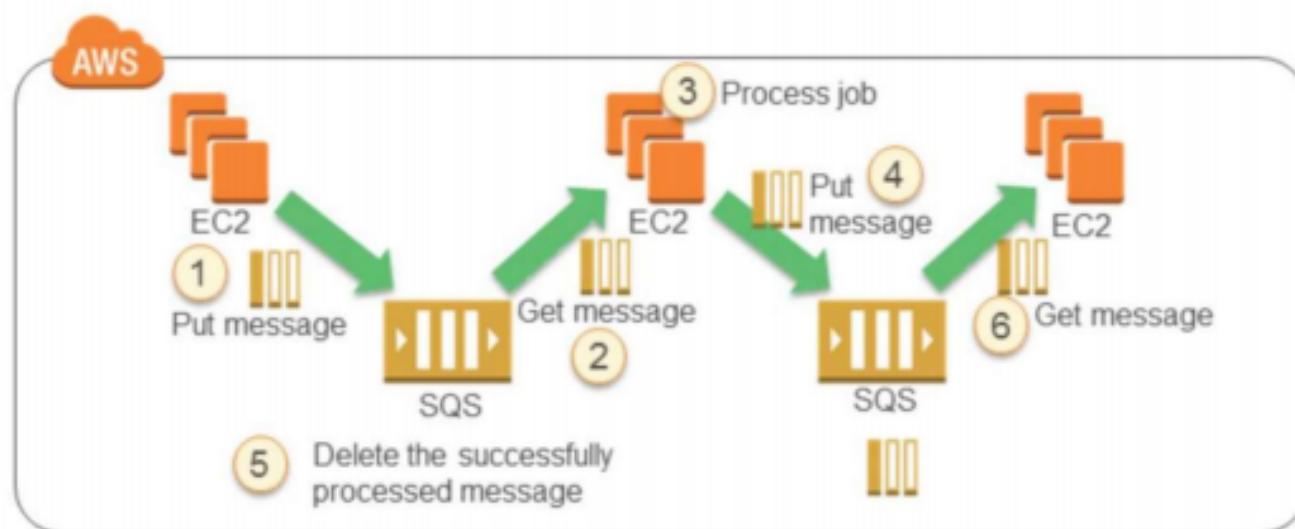
SQS – STANDARD QUEUES

- **Variable message size**
 - supports message in any format up to 256KB of text.
 - messages larger than 256 KB can be managed using the S3 or DynamoDB, with SQS holding a pointer to the S3 object
- **Access Control**
 - Access can be controlled for who can produce and consume messages to each queue
- **Dead Letter Queues**
 - Dead letter queue is a queue for messages that were not able to be processed after a maximum number of attempts
- **PCI Compliance**
 - supports the processing, storage, and transmission of credit card data by a merchant or service provider, and has been validated as being PCI-DSS (Payment Card Industry – Data Security Standard) compliant

SQS – STANDARD QUEUES

Loose Coupling With Amazon SQS

The queuing chain pattern enables **asynchronous processing**:



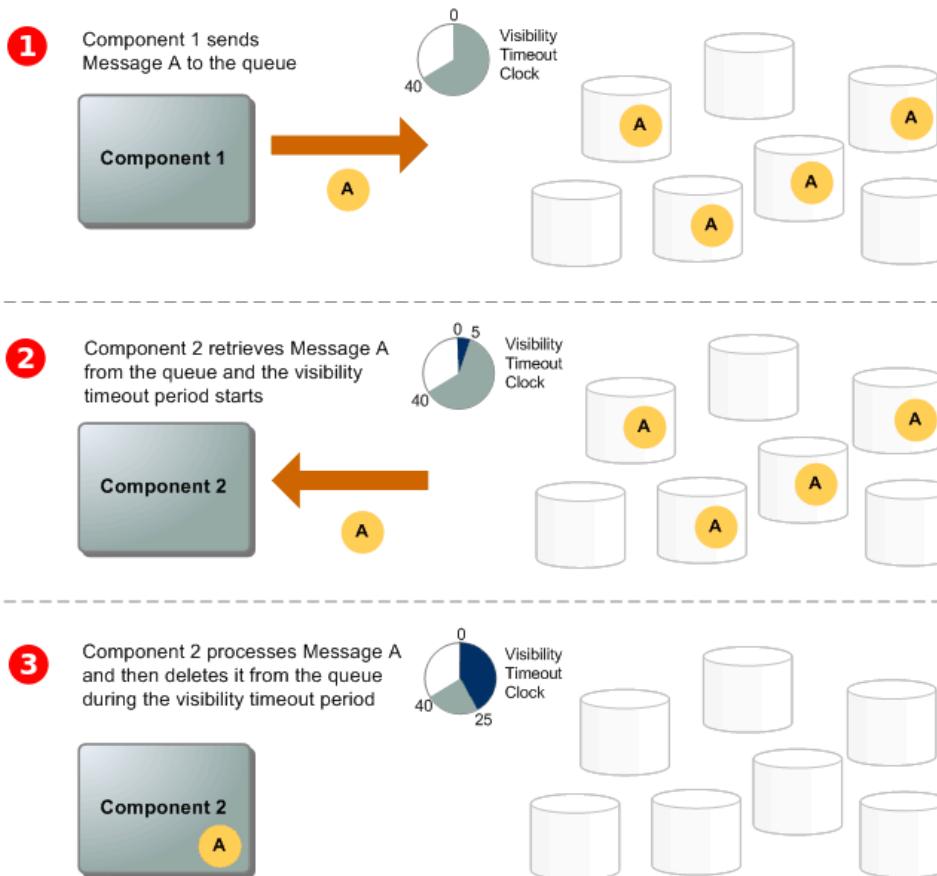
QUEUE AND MESSAGE IDENTIFIERS

- **Queue URLs**
 - Queue is identified by a unique queue name within the same AWS account
 - SQS assigns each queue with a Queue URL identifier for e.g. <http://sq.s.us-east-1.amazonaws.com/123456789012/queue2>
 - Queue URL is needed to perform any operation on the Queue
- **Message ID**
 - Message IDs are useful for identifying messages
 - Each message receives a system-assigned message ID that SQS returns to with the SendMessage response
 - To delete a message, the message's receipt handle instead of the message ID is needed
 - Message ID can be of is 100 characters max
- **Receipt Handle**
 - When a message is received from a queue, a receipt handle is returned with the message which is associated with the act of receiving the message rather then the message itself
 - Receipt handle is required, not the message id, to delete a message or to change the message visibility
 - If a message is received more than once, each time its received, a different receipt handle is assigned and the latest should be used always

VISIBILITY TIMEOUT

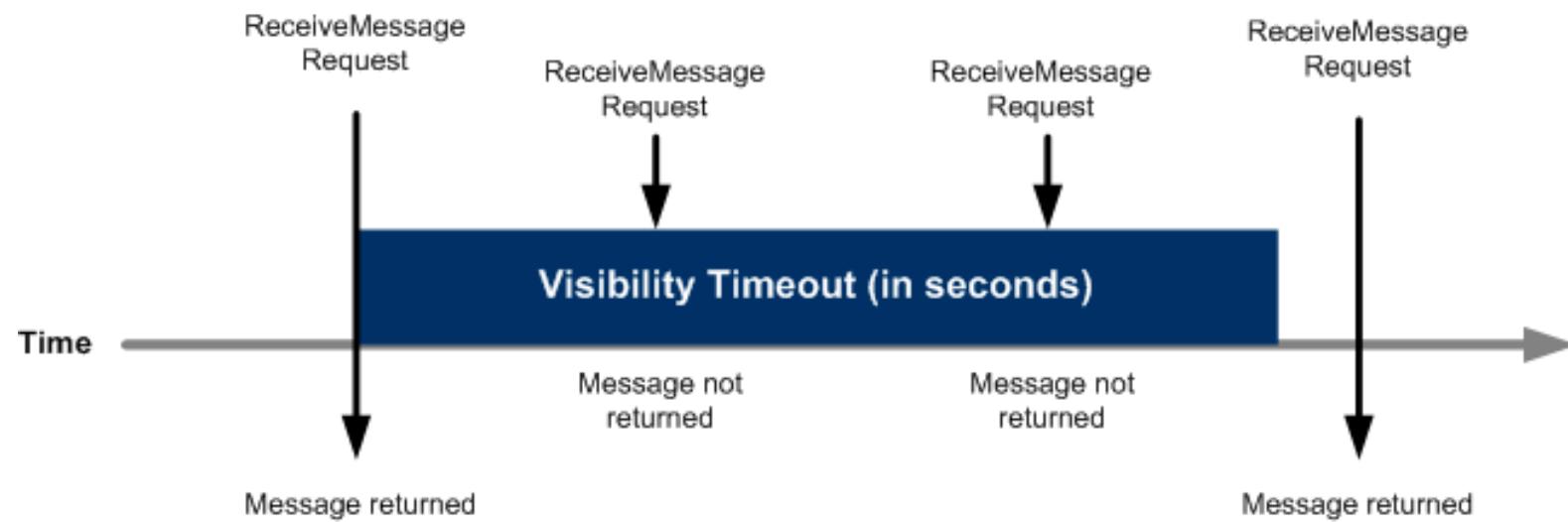
- SQS does not delete the message once it is received by a consumer because the system is distributed, there's no guarantee that the consumer will actually receive the message (it's possible the connection could break or the component could fail before receiving the message)
- Consumer should explicitly delete the message from the Queue once it is received and successfully processed
- As the message is still available on the Queue, other consumers would be able to receive and process and this needs to be prevented
- SQS handles the above behaviour using **Visibility timeout**.
- SQS blocks the visibility of the message for the **Visibility timeout period**, which is the time during which SQS prevents other consuming components from receiving and processing that message
- Consumer should delete the message within the Visibility timeout. If the consumer fails to delete the message before the visibility timeout expires, the message is visible again for other consumers.

SQS



SQS

Cuando un consumidor recibe y procesa un mensaje de una cola, el mensaje permanece en la cola. Amazon SQS no elimina automáticamente el mensaje. Dado que Amazon SQS es un sistema distribuido, no garantiza que el consumidor reciba realmente el mensaje (por ejemplo, debido a un problema de conectividad o a un problema en la aplicación del consumidor). Por tanto, el consumidor debe eliminar el mensaje de la cola después de recibirllo y procesarlo.



SQS – FIFO Queues

Lab

SNS VS SQS

How Is Amazon SNS Different From Amazon SQS?

Amazon SQS and Amazon SNS are both messaging services within AWS.

	Amazon SNS	Amazon SQS
Message persistence	No	Yes
Delivery mechanism	Push (Passive)	Poll (Active)
Producer/consumer	Publish/subscribe	Send/receive

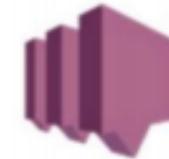
- A common pattern is to use SNS to publish messages to Amazon SQS queues to reliably send messages to one or many system components asynchronously.

Simple Notification Service (SNS)

- Amazon Simple Notification Service (Amazon SNS) is a web service that makes it easy to set up, operate, and **send notifications from the cloud**. It provides developers with a highly scalable, flexible, and cost-effective capability to publish messages from an application and immediately deliver them to subscribers or other applications. It is designed to make web-scale computing easier for developers. Amazon SNS follows the “**publish-subscribe**” (**pub-sub**) **messaging paradigm**, with notifications being delivered to clients using a “push” mechanism that eliminates the need to periodically check or “poll” for new information and updates. With simple APIs requiring minimal up-front development effort, no maintenance or management overhead and pay-as-you-go pricing, Amazon SNS gives developers an easy mechanism to incorporate a powerful notification system with their applications.

Simple Notification Service (SNS)

Amazon Simple Notification Service (SNS)



Amazon SNS enables you to **set up, operate, and send notifications** to subscribing services other applications.

- 💡 Messages published to topic.
- 💡 Topic subscribers receive message.

Subscriber types:

- 💡 Email (plain or JSON)
- 💡 HTTP/HTTPS
- 💡 Short Message Service (SMS) clients (USA only)
- 💡 Amazon SQS queues
- 💡 Mobile push messaging
- 💡 AWS Lambda Function

Simple Notification Service (SNS II)

- Amazon SNS offers several benefits making it a versatile option for building and integrating loosely-coupled, distributed applications:
 - Instantaneous, **push-based delivery (no polling)**
 - Simple APIs and easy integration with applications
 - Flexible **message delivery over multiple transport protocols**
 - Inexpensive, pay-as-you-go model with no up-front costs
 - Web-based AWS Management Console offers the simplicity of a point-and-click interface

Simple Notification Service (SNS III)

- It is very easy to get started with Amazon SNS. **Developers must first create a “topic” which is an “access point” – identifying a specific subject or event type – for publishing messages and allowing clients to subscribe for notifications.** Once a topic is created, the topic owner can set policies for it such as limiting who can publish messages or subscribe to notifications, or specifying which notification protocols will be supported (i.e. HTTP/HTTPS, email, SMS). Subscribers are clients interested in receiving notifications from topics of interest; they can subscribe to a topic or be subscribed by the topic owner. Subscribers specify the protocol and end-point (URL, email address, etc.) for notifications to be delivered. When publishers have information or updates to notify their subscribers about, they can publish a message to the topic – which immediately triggers Amazon SNS to deliver the message to all applicable subscribers.

Simple Notification Service (SNS IV)

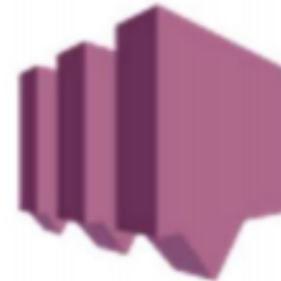
- In order for customers to have broad flexibility of delivery mechanisms, Amazon SNS supports notifications over multiple transport protocols. Customers can select one of the following transports as part of the subscription requests:
 - “HTTP”, “HTTPS” – Subscribers specify a URL as part of the subscription registration; notifications will be delivered through an HTTP POST to the specified URL.
 - “Email”, “Email-JSON” – Messages are sent to registered addresses as email. Email-JSON sends notifications as a JSON object, while Email sends text-based email.
 - “SQS” – Users can specify an SQS standard queue as the endpoint; Amazon SNS will enqueue a notification message to the specified queue (which subscribers can then process using SQS APIs such as ReceiveMessage, DeleteMessage, etc.). Note that FIFO queues are not currently supported.
 - “SMS” – Messages are sent to registered phone numbers as SMS text messages.

Lab

Simple Notification Service (SNS IV)

Characteristics Of Amazon SNS

- 📦 Single published message
- 📦 Order is not guaranteed
- 📦 No recall
- 📦 HTTP/HTTPS retry
- 📦 256 KB max per message



Simple Notification Service (SNS IV)

- In order for customers to have broad flexibility of delivery mechanisms, Amazon SNS supports notifications over multiple transport protocols. Customers can select one of the following transports as part of the subscription requests:
 - “HTTP”, “HTTPS” – Subscribers specify a URL as part of the subscription registration; notifications will be delivered through an HTTP POST to the specified URL.
 - “Email”, “Email-JSON” – Messages are sent to registered addresses as email. Email-JSON sends notifications as a JSON object, while Email sends text-based email.
 - “SQS” – Users can specify an SQS standard queue as the endpoint; Amazon SNS will enqueue a notification message to the specified queue (which subscribers can then process using SQS APIs such as ReceiveMessage, DeleteMessage, etc.). Note that FIFO queues are not currently supported.
 - “SMS” – Messages are sent to registered phone numbers as SMS text messages.

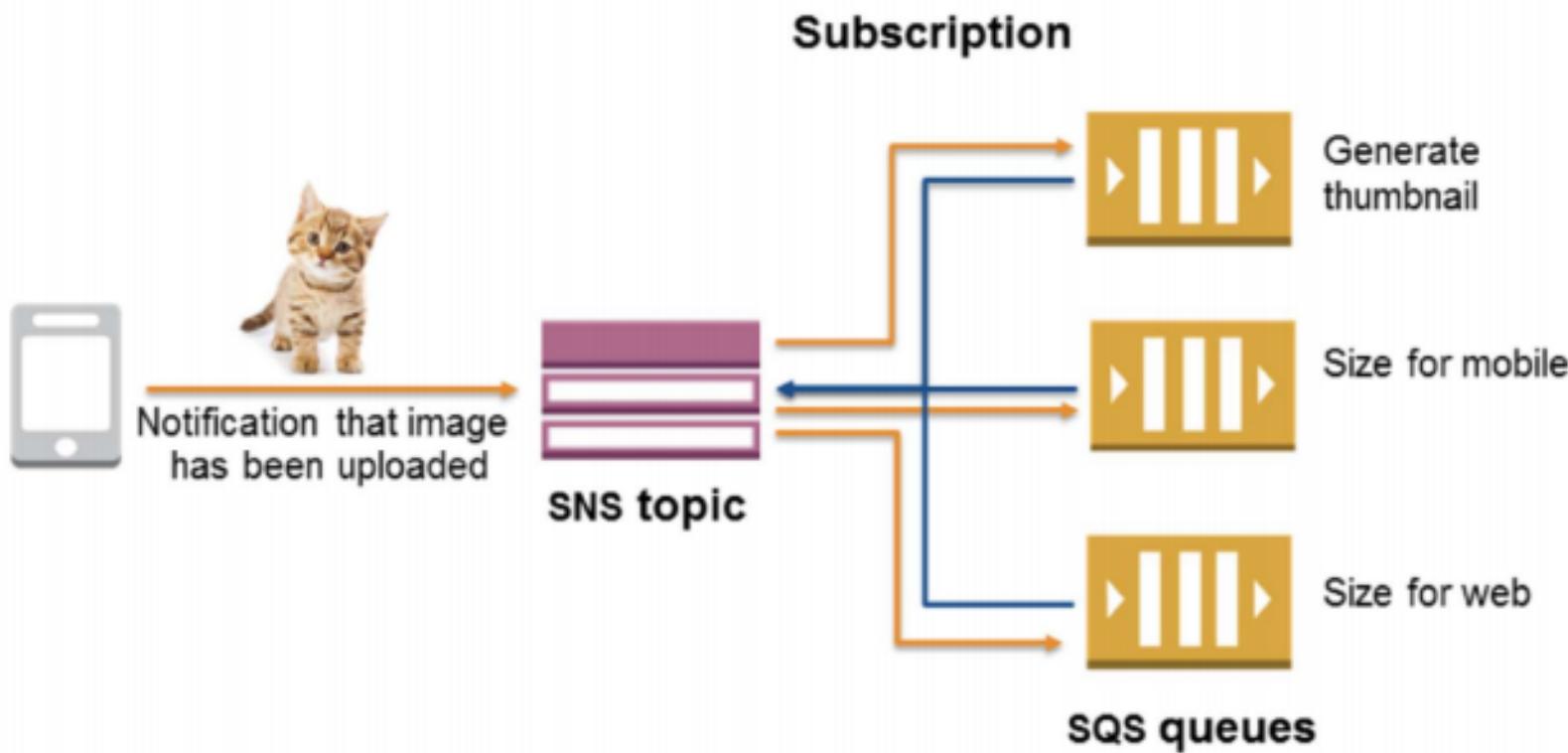
Lab

SNS vs SQS

- Amazon Simple Queue Service (SQS) and Amazon SNS are both messaging services within AWS, which provide different benefits for developers. Amazon SNS allows applications to send time-critical messages to multiple subscribers through a “push” mechanism, eliminating the need to periodically check or “poll” for updates. Amazon SQS is a message queue service used by distributed applications to exchange messages through a polling model, and can be used to decouple sending and receiving components. Amazon SQS provides flexibility for distributed components of applications to send and receive messages without requiring each component to be concurrently available.
- A common pattern is to use SNS to publish messages to Amazon SQS queues to reliably send messages to one or many system components asynchronously.

SNS + SQS

Use Case: Fan-Out



DB

<https://aws.amazon.com/products/databases/>



En memoria

Las bases de datos en memoria se utilizan en aplicaciones que requieren acceso en tiempo real a los datos. Al almacenar los datos directamente en la memoria, estas bases de datos proporcionan una latencia de microsegundos cuando la latencia de milisegundos no es suficiente.

Se utilizan en: almacenamiento en caché, marcadores de videojuegos y análisis en tiempo real.

Productos de AWS:

- [Amazon ElastiCache para Redis](#)
- [Amazon ElastiCache para Memcached](#)



Gráficos

Las bases de datos de gráficos se utilizan en aplicaciones que deben permitir a millones de usuarios consultar las relaciones entre conjuntos de datos altamente conectados, así como navegar por ellas, con una latencia de milisegundos.

Se utilizan en: detección de fraudes, redes sociales y motores de recomendaciones

Producto de AWS:

- [Amazon Neptune](#)



Series temporales

Las bases de datos de series temporales se utilizan para recopilar, sintetizar y sacar conclusiones de forma eficaz de enormes cantidades de datos que cambian con el paso del tiempo (denominados datos de series temporales).

Se utilizan en: aplicaciones de IoT, DevOps y telemetría industrial.

Producto de AWS:

- [Amazon Timestream](#)



Contabilidad

Las bases de datos de contabilidad se utilizan cuando se necesita una autoridad centralizada de confianza para conservar un registro de transacciones escalable, completo y criptográficamente verificable.

Se utilizan en: sistemas de registro, cadenas de suministros, registros y transacciones bancarias.

Producto de AWS:

- [Amazon Quantum Ledger Database \(QLDB\)](#)



Relacionales

Las bases de datos relacionales almacenan datos cuyas relaciones y esquema están predefinidos, diseñadas para admitir transacciones ACID y conservar la integridad referencial, así como la coherencia de los datos.

Se utilizan en: aplicaciones tradicionales, ERP, CRM y e-commerce.

Productos de AWS:

- [Amazon Aurora](#)
MySQL, PostgreSQL
- [Amazon RDS](#)
MySQL, PostgreSQL, MariaDB, Oracle, SQL Server
- [Amazon Redshift](#)



Pares clave-valor

Las bases de datos de pares de clave-valor están optimizadas para almacenar y recuperar pares de clave-valor en grandes volúmenes en milisegundos, sin la sobrecarga en el rendimiento y las limitaciones de escala propias de las bases de datos relacionales.

Se utilizan en: aplicaciones a escala de Internet, pujas en tiempo real, carros de la compra y preferencias de los clientes.

Producto de AWS:

- [Amazon DynamoDB](#)



Documentales

Las bases de datos documentales están diseñadas para almacenar datos semiestructurados en forma de documentos y su uso resulta muy intuitivo para los desarrolladores, ya que los datos suelen representarse como un documento legible.

Se utilizan en: administración de contenido, personalización y aplicaciones móviles.

Producto de AWS:

- [Amazon DocumentDB \(compatible con MongoDB\)](#)

<https://aws.amazon.com/products/databases/>

If You Need	Consider Using	Product Type
A fully managed MySQL and PostgreSQL-compatible relational database with the performance and availability of enterprise databases at 1/10th the cost.	Amazon Aurora	Relational Database
A managed relational database in the cloud that you can launch in minutes with just a few clicks.	Amazon RDS	Relational Database
A serverless, NoSQL database that delivers consistent single-digit millisecond latency at any scale.	Amazon DynamoDB	NoSQL Database
A fast, fully managed, petabyte-scale data warehouse at 1/10th the cost of traditional solutions.	Amazon Redshift	Data Warehouse
To deploy, operate, and scale an in-memory data store based on Memcached or Redis in the cloud.	Amazon ElastiCache	In-Memory Data Store
A fast, reliable, fully managed graph database to store and manage highly connected data sets.	Amazon Neptune	Graph Database
Help migrating your databases to AWS easily and inexpensively with minimal downtime.	AWS Database Migration Service	Database Migration

Applications	Consider Using
Transactional applications like ERP, CRM, and eCommerce to log transactions and store structured data.	Amazon Aurora , Amazon RDS
Internet scale applications like hospitality, dating, and ride sharing to serve content and store structured and unstructured data.	Amazon DynamoDB
Analytic applications for operational reporting and querying Terabyte to Exabyte scale data.	Amazon Redshift
Real-time application use cases that require sub-millisecond latency like gaming leaderboards, chat/messaging, streaming, and IoT.	Amazon ElastiCache
Applications with use cases that require navigation of highly connected data like social news feeds, recommendations, and fraud detection.	Amazon Neptune

Relational/SQL And NoSQL Databases

	Relational/SQL	NoSQL
Data Storage	Rows and Columns	Key-Value, Documents, and Graphs
Schemas	Fixed	Dynamic
Querying	Using SQL	Focused on collection of documents
Scalability	Vertical	Horizontal

Relational/SQL

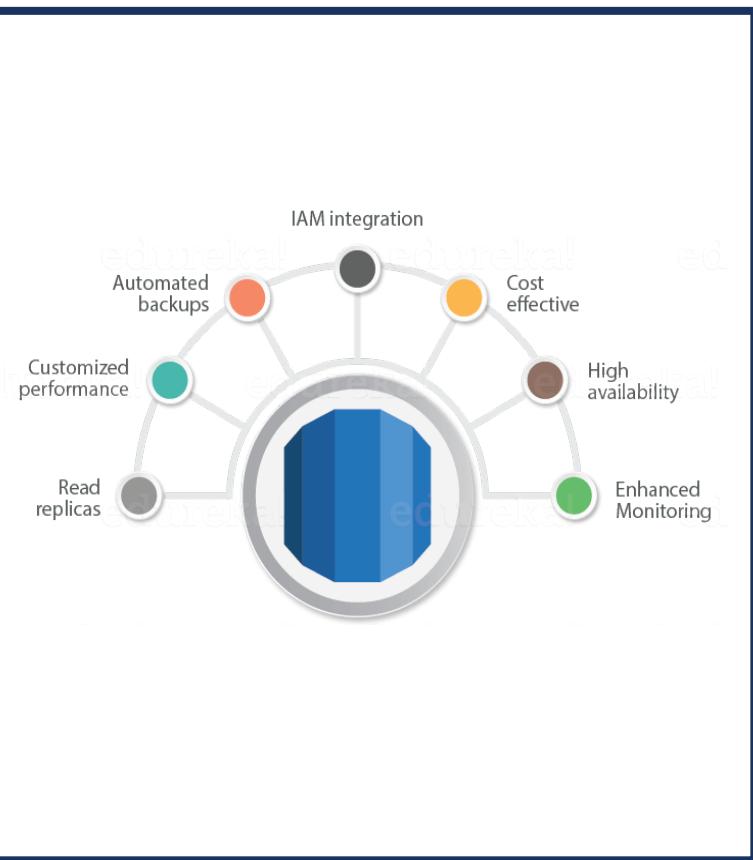
ISBN	Title	Author	Format
9182932465265	Cloud Computing Concepts	Wilson, Joe	Paperback

NoSQL

```
{
  ISBN: 9182932465265,
  Title: "Cloud Computing Concepts",
  Author: "Wilson, Joe",
  Format: "Paperback"
}
```

	Amazon RDS	Amazon DynamoDB	Amazon ElastiCache	Amazon EMR	Amazon Redshift
Access	SQL	API	API	API	SQL
Consistency	ACID	BASE	Depends	Depends	ACID
Scaling	Scale up/down; add read replicas	UpdateTable API	Add/remove cluster nodes	Add task/core nodes; remove task nodes	Migrate to new cluster
Schema	Static; strings, numbers, dates, etc.	Dynamic; scalar, multi-valued, document	No fixed schema	No fixed schema	Static
Size	Terabytes	Petabytes	Gigabytes	Petabytes	Petabytes
Storage	EBS	Distributed	In memory	Local or S3	Local to instances
Availability	Multi-AZ option	Multi-AZ fault tolerant	Multi-AZ for Redis	No multi-AZ	No multi-AZ

AWS RELATIONAL DATABASES (RDS)



- Amazon Relational Database Service (RDS) is a web service that makes it easier to set up, operate, and scale a relational database in the cloud.
- RDS features & benefits
 - CPU, memory, storage, and IOPS can be scaled independently.
 - manages backups, software patching, automatic failure detection, and recovery.
 - automated backups can be performed as needed, or manual backups can be triggered as well. Backups can be used to restore a database, and the RDS restore process works reliably and efficiently.
 - provides high availability with a primary instance and a synchronous secondary instance that can be failovered to seamlessly when a problem occurs.
 - provides elasticity & scalability by enabling MySQL, MariaDB, or PostgreSQL Read Replicas to increase read scaling.
 - supports MySQL, MariaDB, PostgreSQL, Oracle, Microsoft SQL Server, and the new, MySQL-compatible Amazon Aurora DB engine
 - in addition to the security in the database package, IAM users and permissions can help to control who has access to the RDS database service
 - databases can be further protected by putting them in a VPC, using SSL for data in transit and encryption for data at rest
 - However, as it is a **managed service**, **shell (root ssh) access to DB instances is not provided**, and this restricts access to certain system procedures and tables that require advanced privileges.
 - resizable capacity and manages common database administration tasks.

RDS COMPONENTS

■ DB Instance

- is a basic building block of RDS
- is an isolated database environment in the cloud
- each DB instance runs a DB engine. AWS currently supports MySQL, MariaDB, PostgreSQL, Oracle, and Microsoft SQL Server & Aurora DB engines
- can be accessed from Amazon AWS command line tools, Amazon RDS APIs, or the AWS Management RDS Console.
- computation and memory capacity of an DB instance is determined by its DB instance class, which can be selected as per the needs
- for each DB instance, 5 GB to 6 TB of associated storage capacity can be selected
- storage comes in three types: Magnetic, General Purpose (SSD), and Provisioned IOPS (SSD), which differ in performance characteristics and price
- each DB instance has a DB instance identifier, which is customer-supplied name and must be unique for that customer in an AWS region. It uniquely identifies the DB instance when interacting with the Amazon RDS API and AWS CLI commands.
- each DB instance can host multiple databases, or a single Oracle database with multiple schemas.
- can be hosted in an AWS VPC environment for better control

RDS COMPONENTS

■ Regions and Availability Zones

- AWS resources are housed in highly available data center facilities in different areas of world, these data centers are called regions which further contain multiple distinct locations called Availability Zones
- Each AZ is engineered to be isolated from failures in other AZs, and to provide inexpensive, low-latency network connectivity to other AZs in the same region
- DB instances can be hosted in several AZs, an option called a Multi-AZ deployment.
 - Amazon automatically provisions and maintains a synchronous standby replica of the DB instance in a different AZ.
 - Primary DB instance is synchronously replicated across AZs to the standby replica
 - Provides data redundancy, failover support, eliminate I/O freezes, and minimize latency spikes during system backups.

■ Security Groups

- Security group controls the access to a DB instance, by allowing access to the specified IP address ranges or EC2 instances

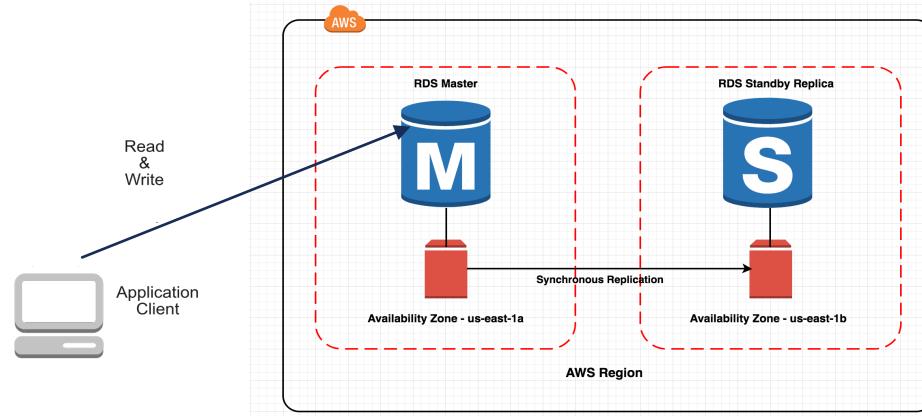
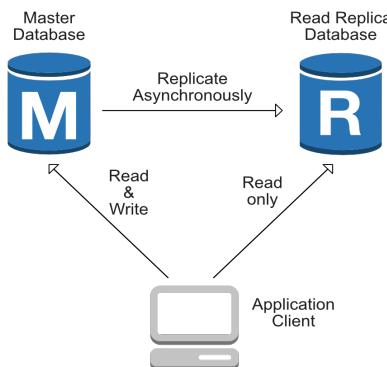
■ DB Parameter Groups

- A DB parameter group contains engine configuration values that can be applied to one or more DB instances of the same instance type

■ DB Option Groups

- Some DB engines offer tools that simplify managing the databases and making the best use of data.
- Amazon RDS makes such tools available through option groups for e.g. *Oracle Application Express (APEX)*, *SQL Server Transparent Data Encryption*, and *MySQL memcached support*.

RDS READ REPLICA AND MULTI-AZ



■ Read Replica :

- Read Replicas provide enhanced performance and durability for database (DB) instances.
- This feature makes it easy to elastically scale out beyond the capacity constraints of a single DB instance for read-heavy database workloads.
- You can create one or more replicas of a given source DB Instance and serve high-volume application read traffic from multiple copies of your data, thereby increasing aggregate read throughput.

■ Multi-AZ:

- Amazon RDS automatically provisions and maintains a synchronous standby replica in a different Availability Zone.
- The primary DB instance is synchronously replicated across Availability Zones to a standby replica to provide data redundancy, eliminate I/O freezes, and minimize latency spikes during system backups.
- Running a DB instance with high availability can enhance availability during planned system maintenance, and help protect your databases against DB instance failure and Availability Zone disruption.

RDS READ REPLICA AND MULTI-AZ

Implementaciones Multi-AZ	Réplicas de lectura
Replicación síncrona de larga duración	Replicación asíncrona altamente escalable
Solo el motor de base de datos en la instancia principal está activo	Todas las réplicas de lectura son accesibles y se pueden usar para la escalabilidad de lectura
Las copias de seguridad automáticas se realizan a partir de la instancia en espera	No hay copias de seguridad configuradas de manera predeterminada
Siempre abarca dos zonas de disponibilidad dentro de una sola región	Puede estar dentro de una zona de disponibilidad, entre zonas distintas o entre regiones distintas
Las actualizaciones de la versión del motor de base de datos ocurren en la instancia principal	La actualización de la versión del motor de base de datos es independiente de la instancia de origen
Comutación por error automática al modo de espera cuando se detecta un problema	Se puede promocionar manualmente a una instancia de base de datos independiente

DYNAMODB

Amazon DynamoDB Review

Amazon DynamoDB is a **fully managed NoSQL database service**.



- Consistent, single-digit millisecond latency at any scale
- No table size or throughput limits
- Runs exclusively on SSDs
- Document and key-value store models supported
- Ideal for mobile, web, gaming, ad tech, and IoT applications
- Accessible via the AWS Management Console, the AWS Command-Line Interface, or simple API calls

- Amazon DynamoDB es un servicio de bases de datos NoSQL totalmente administrado que ofrece un desempeño rápido y previsible, así como una escalabilidad óptima. DynamoDB le permite trasladar a AWS las cargas administrativas que supone tener que utilizar y escalar una base de datos distribuida, para que no tenga que preocuparse del aprovisionamiento, la instalación y la configuración del hardware, ni tampoco de las tareas de replicación, revisión del software o escalado de clústeres. Además, DynamoDB ofrece el cifrado en reposo, que elimina la carga y la complejidad operativa que conlleva la protección de información confidencial. Para obtener más información, consulte [Cifrado en reposo de DynamoDB](#).

- Con DynamoDB, puede crear tablas de base de datos capaces de almacenar y recuperar cualquier cantidad de datos, así como de atender cualquier nivel de tráfico de solicitudes. Puede escalar la capacidad de desempeño de las tablas para aumentarla o reducirla sin tiempos de inactividad ni reducción del rendimiento. Puede utilizar la Consola para monitorizar la utilización de recursos y las métricas de rendimiento.
- DynamoDB proporciona una funcionalidad de copia de seguridad bajo demanda. Le permite crear copias de seguridad completas de las tablas para una retención y archivado a largo plazo con el objetivo de cumplir los requisitos de conformidad normativa. Para obtener más información, consulte [Backup y restauración bajo demanda para DynamoDB](#).
- Puede crear copias de seguridad bajo demanda y habilitar recuperaciones a un momento dado en las tablas de Amazon DynamoDB. La recuperación a un momento dado ayuda a proteger las tablas de operaciones accidentales de escritura o eliminación. Con la recuperación a un momento dado, puede restaurar la tabla a cualquier momento de los últimos 35 días. Para obtener más información, consulte [Recuperación a un momento dado: funcionamiento](#).
- DynamoDB permite eliminar automáticamente los elementos vencidos de las tablas, para ayudarle a reducir el consumo de almacenamiento y el costo que suponen los datos que ya no son pertinentes. Para obtener más información, consulte [Tiempo de vida](#).

DYNAMODB TABLAS, ELEMENTOS Y ATRIBUTOS

- **Tablas (Tables)** – Al igual que otros sistemas de administración de bases de datos, DynamoDB almacena datos en tablas. Una *tabla* es una colección de datos. Por ejemplo, consulte la tabla de ejemplo denominada *People*, que puede utilizar para almacenar información de contacto personal sobre amigos, familiares u otras personas de interés. También podría utilizar una tabla *Cars* para almacenar información sobre los vehículos que conducen las personas.
- **Elementos (Items)** – Cada tabla contiene cero o más elementos. Un *elemento* es un grupo de atributos que puede identificarse de forma exclusiva entre todos los demás elementos. En una tabla *People*, cada elemento representa a una persona. En una tabla *Cars*, cada elemento representa un vehículo. Los elementos de DynamoDB son similares en muchos aspectos a las filas, los registros o las tuplas de otros sistemas de bases de datos. En DynamoDB, no existe ningún límite respecto al número de elementos que pueden almacenarse en una tabla.
- **Atributos (Attributes)** – Cada elemento se compone de uno o varios atributos. Un *atributo* es un componente fundamental de los datos, que no es preciso dividir más. Por ejemplo, un elemento de una tabla *People* contiene los atributos *PersonID*, *Lastname*, *Firstname*, etc. En una tabla *Department*, un elemento podría tener atributos tales como *DepartmentID*, *Name*, *Manager*, etc. En DynamoDB, los atributos se parecen en muchos aspectos a los campos o columnas en otros sistemas de bases de datos.

People

- Cada elemento de la tabla tiene un identificador único, o clave principal, que lo distingue de todos los demás. En la tabla *People*, la clave principal consta de un atributo (*PersonID*).

```
{  
    "PersonID": 101,  
    "LastName": "Smith",  
    "FirstName": "Fred",  
    "Phone": "555-4321"  
}
```

```
{  
    "PersonID": 102,  
    "LastName": "Jones",  
    "FirstName": "Mary",  
    "Address": {  
        "Street": "123 Main",  
        "City": "Anytown",  
        "State": "OH",  
        "ZIPCode": 12345  
    }  
}
```

```
{  
    "PersonID": 103,  
    "LastName": "Stephens",  
    "FirstName": "Howard",  
    "Address": {  
        "Street": "123 Main",  
        "City": "London",  
        "PostalCode": "ER3 5K8"  
    },  
    "FavoriteColor": "Blue"  
}
```

Music

- Dejando a un lado la clave principal, la tabla *People* no tiene esquema. Esto significa que no es preciso definir de antemano los atributos ni sus tipos de datos. Cada elemento puede tener sus propios atributos diferentes.

La mayoría de los atributos son *escalares*, lo que significa que solo pueden tener un valor. Las cadenas y los números son ejemplos comunes de escalares.

Algunos de los elementos tienen un atributo anidado (*Address*). DynamoDB admite atributos anidados hasta 32 niveles de profundidad.

```
{  
    "Artist": "No One You Know",  
    "SongTitle": "My Dog Spot",  
    "AlbumTitle": "Hey Now",  
    "Price": 1.98,  
    "Genre": "Country",  
    "CriticRating": 8.4  
}
```

```
{  
    "Artist": "No One You Know",  
    "SongTitle": "Somewhere Down The Road",  
    "AlbumTitle": "Somewhat Famous",  
    "Genre": "Country",  
    "CriticRating": 8.4,  
    "Year": 1984  
}
```

```
{  
    "Artist": "The Acme Band",  
    "SongTitle": "Still in Love",  
    "AlbumTitle": "The Buck Starts Here",  
    "Price": 2.47,  
    "Genre": "Rock",  
    "PromotionInfo": {  
        "RadioStationsPlaying": [  
            "KHCR",  
            "KQBX",  
            "WTNR",  
            "WJJH"  
        ],  
        "TourDates": [  
            "Seattle": "20150625",  
            "Cleveland": "20150630"  
        ],  
        "Rotation": "Heavy"  
    }  
}
```

```
{  
    "Artist": "The Acme Band",  
    "SongTitle": "Look Out, World",  
    "AlbumTitle": "The Buck Starts Here",  
    "Price": 0.99,  
    "Genre": "Rock"  
}
```

- La clave principal de *Music* consta de dos atributos (*Artist* y *SongTitle*). Cada elemento de la tabla debe tener estos dos atributos. La combinación de *Artist* y *SongTitle* distingue a cada elemento de la tabla de todos los demás.

Dejando a un lado la clave principal, la tabla *Music* no tiene esquema. Esto significa que no es preciso definir de antemano los atributos ni sus tipos de datos. Cada elemento puede tener sus propios atributos diferentes.

- Uno de los elementos tiene un atributo anidado (*PromotionInfo*), que contiene otros atributos anidados. DynamoDB admite atributos anidados hasta 32 niveles de profundidad.

DYNAMODB DATA TYPES

- DynamoDB admite muchos tipos de datos para los atributos de una tabla :
- **Scalar Types** – Un tipo escalar es aquel que puede representar exactamente un valor. Los tipos escalares son:
 - Number: Los números pueden ser positivos, negativos o cero. Los números pueden tener hasta 38 dígitos de precisión.
 - String: Los valores de tipo String son Unicode con codificación binaria UTF-8. La longitud de una cadena debe ser mayor que cero y menor que el máximo de tamaño de elemento de DynamoDB, 400 KB.
 - Binary: Los atributos de tipo Binary pueden almacenar cualquier tipo de datos binarios, como texto comprimidos, datos cifrados o imágenes. Siempre que DynamoDB compara valores de tipo Binary, trata cada byte de los datos binarios como sin signo.
 - Boolean: Un atributo de tipo Boolean puede almacenar los valores true o false.
 - Null: Null representa un atributo con un estado desconocido o sin definir.
- **Document Types** – un tipo de documento puede representar una estructura compleja con atributos anidados, como los que se encontraría en un documento JSON. Los tipos de documentos son :
 - List puede almacenar una colección ordenada de valores. Las listas deben ir entre corchetes: [...].
 - Map. almacenar una colección desordenada de pares nombre-valor. Los mapas deben ir entre llaves: { ... }
- **Set Types** – Un tipo de conjunto puede representar varios valores escalares. Los tipos
 - String Set, ["Black", "Green", "Red"]
 - Number Set [42.2, -19, 7.5, 3.14]
 - Binary Set. ["U3Vubnk=", "UmFpbnk=", "U25vd3k="]

```
FavoriteThings: ["Cookies", "Coffee", 3.14159]
```

```
{  
    Day: "Monday",  
    UnreadEmails: 42,  
    ItemsOnMyDesk: [  
        "Coffee Cup",  
        "Telephone",  
        {  
            Pens: { Quantity : 3},  
            Pencils: { Quantity : 2},  
            Erasers: { Quantity : 1}  
        }  
    ]  
}
```

DYNAMODB LECTURA

- Amazon DynamoDB está disponible en varias regiones y cada región es independiente y se encuentra aislada de las demás regiones de AWS. Por ejemplo, si tenemos una tabla denominada *People* en la región *us-east-2* y otra tabla denominada *People* en la región *us-west-2*, se consideran dos tablas completamente independientes.
- Cada región de AWS consta de distintas zonas de disponibilidad. Cada zona de disponibilidad está aislada de los errores que se produzcan en d y proporciona conectividad de red de baja latencia con otras zonas de disponibilidad de la misma región. Esto permite la replicación rápida de los datos entre varias zonas de disponibilidad de una región.
- Cuando se escribe datos en una tabla y recibe una respuesta HTTP 200 (OK), la escritura se ha realizado y es duradera. Los datos presentan consistencia final en todas las ubicaciones de almacenamiento, en un segundo o menos.
- Lectura. DynamoDB soporta dos tipo de modo de lectura
 - **Eventually Consistent Reads.** la respuesta puede mostrar datos antiguos que no reflejan de una operación de escritura reciente.. Si repite la solicitud de lectura tras un breve intervalo de tiempo, la respuesta seria los datos de la última escritura.
 - **Strongly Consistent Reads.** Se devuelve una respuesta con los datos más actualizados, de las escrituras realizadas correctamente. Una lectura de este tipo podría no estar disponible si se produce un retraso o una interrupción en la red. Las lecturas Strongly no se admiten para los índices secundarios globales.



Provisioned Throughput

Read/write capacity mode

Select on-demand if you want to pay only for the reads and writes you perform, with no capacity planning required. Select provisioned to save on throughput costs if you can reliably estimate your application's throughput requirements. See the [DynamoDB pricing page](#) and [DynamoDB Developer Guide](#) to learn more.

Read/write capacity mode can be changed later.

- Provisioned (free-tier eligible)
- On-demand

Provisioned vs On demand pricing



You specify your throughput capacity requirements (read/write), and **DynamoDB allocates the resources you need**.

Read capacity unit:

- One **strongly** consistent read per second for items as large as 4 KB.
- Two **eventually** consistent reads per second for items as large as 4 KB.

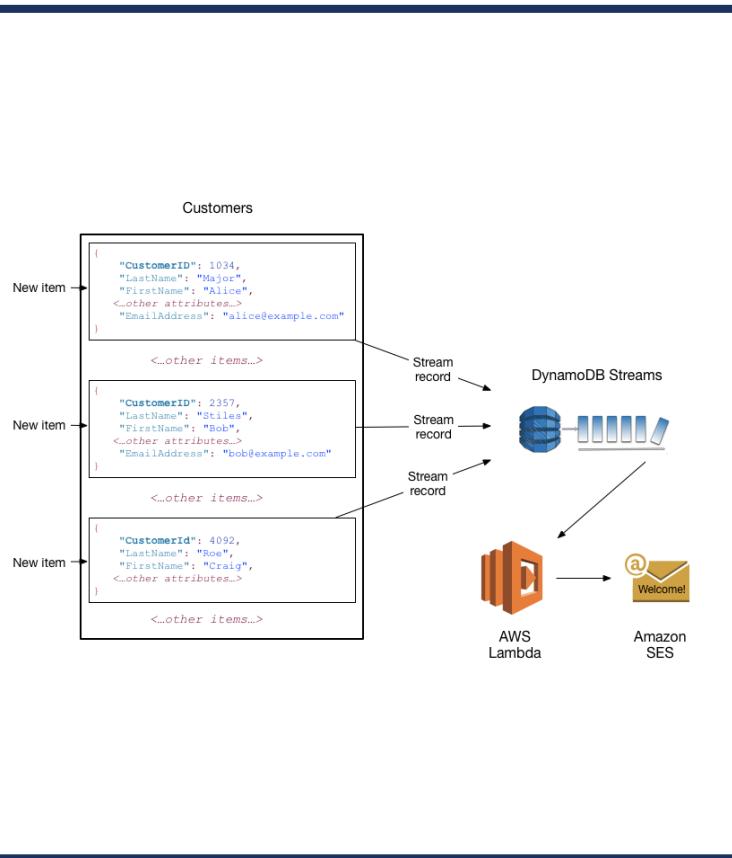
Write capacity unit:

- One write per second for items as large as 1 KB.

On-demand Mode

- On-demand mode provides flexible billing option capable of serving thousands of requests per second without capacity planning
- There is no need to specify the expected read and write throughput
- Charged for only the reads and writes that the application performs on the tables in terms of read request units and write request units.

DYNAMODB STREAM



- DynamoDB Streams provides a **time-ordered sequence of item-level changes** made to data in a table in the last 24 hours, after which they are erased
- Maintains ordered sequence of the events per item however across item are not maintained
- Can be used for multi-region replication to keep other data stores up-to-date with the latest changes to DynamoDB or to take actions based on the changes made to the table
- Helps developers consume updates and receive the item-level data before and after items are changed
- DynamoDB Streams is designed so that every update made to the table will be represented exactly once in the stream No Duplicates

DYNAMODB KEY

■ Primary Key

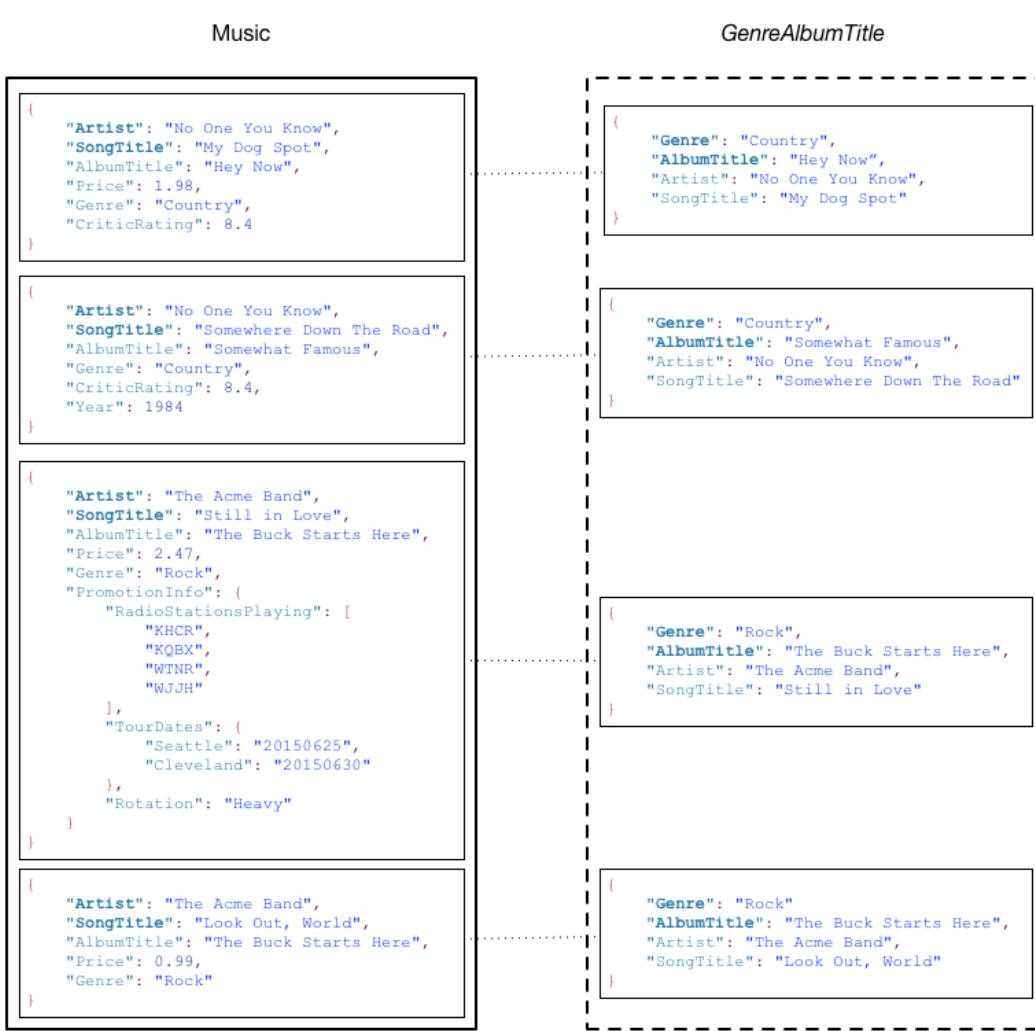
- identifica de forma única a cada elemento de la tabla, de manera que no puede haber dos elementos con la misma clave.
- DynamoDB admite dos tipos distintos de clave principal:
 - Partition key – usa solo un atributo llamado partition key. No puede haber dos elementos que tengan el mismo valor de partition key
 - Partition key and sort key – Esta compuesta por dos claves, la primera es la partition key y la segunda es la sort key (clave de ordenación).
 - En este tipo puede haber dos elementos con la misma partition key, pero la sort key debe ser distinta.

■ Secondary index

- Se puede crear uno o varios índices secundarios en una tabla para hacer consultas a los datos de la tabla usando una key alternativa
- DynamoDB admite dos tipos de índices:
 - Global secondary index – Índice en el que la partition key y/o sort key distinta de la tabla.
 - Local secondary index – Índice con la misma partition key, pero una sort key distinta.

DYNAMODB KEY

Valor de clave de partición	Uniformidad
ID de usuario, en caso de que la aplicación tenga muchos usuarios.	Buena
Código de estado, aunque solo hay algunos códigos de estado posibles.	Mala
Fecha de creación del elemento, redondeada al periodo más próximo (por ejemplo, día, hora o minuto).	Mala
ID de dispositivo, en un caso en que cada dispositivo obtiene acceso a los datos a intervalos relativamente similares.	Buena
ID del dispositivo, donde, aunque se hace un seguimiento de muchos dispositivos, uno de ellos es muchísimo más popular que los demás.	Mala



En el ejemplo de la tabla *Music* se puede consultar por *Artist*(clave de partición) o por *Artist* y *SongTitle* (claves de partición y ordenación). ¿Qué sucede si también desea consultar los datos por género musical (*Genre*) y título de álbum (*AlbumTitle*)? Para ello, puede crear un índice basado en *Genre* y *AlbumTitle* y consultararlo.

Cada índice pertenece a una tabla, que se denomina la tabla base del índice. En el ejemplo anterior, *Music* es la tabla base del índice *GenreAlbumTitle*.

DynamoDB mantiene los índices automáticamente. Al agregar, actualizar o eliminar un elemento de la tabla base, DynamoDB agrega, actualiza o elimina el elemento correspondiente en los índices que pertenecen a dicha tabla.

Al crear un índice, se especifica qué atributos de la tabla base se copiarán, o proyectarán, en el índice. Como mínimo, DynamoDB proyecta en el índice los atributos de clave de la tabla base. Esto es lo que sucede con el índice *GenreAlbumTitle*, en el que únicamente se proyectan los atributos de clave de la tabla *Music*.

Table

Primary Key		Data Attributes...				
Partition Key	Sort Key					
Player_ID	Game_ID	Attribute 1		Attribute 2		Attribute 3
Rick	Game_1	Score:	36,750 <i>(game score)</i>	Date:	2017-11-14 <i>(date of game)</i>	
	Game_2	Score:	69,450 <i>(game score)</i>	Date:	2017-12-31 <i>(date of game)</i>	
	Game_3	Score:	135,900 <i>(game score)</i>	Date:	2018-01-19 <i>(date of game)</i>	Award: Champ <i>(type of award)</i>
Padma	Game_4	Score:	25,350 <i>(game score)</i>	Date:	2018-01-27 <i>(date of game)</i>	
	Game_5	Score:	69,450 <i>(game score)</i>	Date:	2028-01-19 <i>(date of game)</i>	
	Game_6	Score:	147,300 <i>(game score)</i>	Date:	2018-02-02 <i>(date of game)</i>	Award: Champ <i>(type of award)</i>
	Game_7	Score:	169,100 <i>(game score)</i>	Date:	2018-03-10 <i>(date of game)</i>	Award: Champ <i>(type of award)</i>

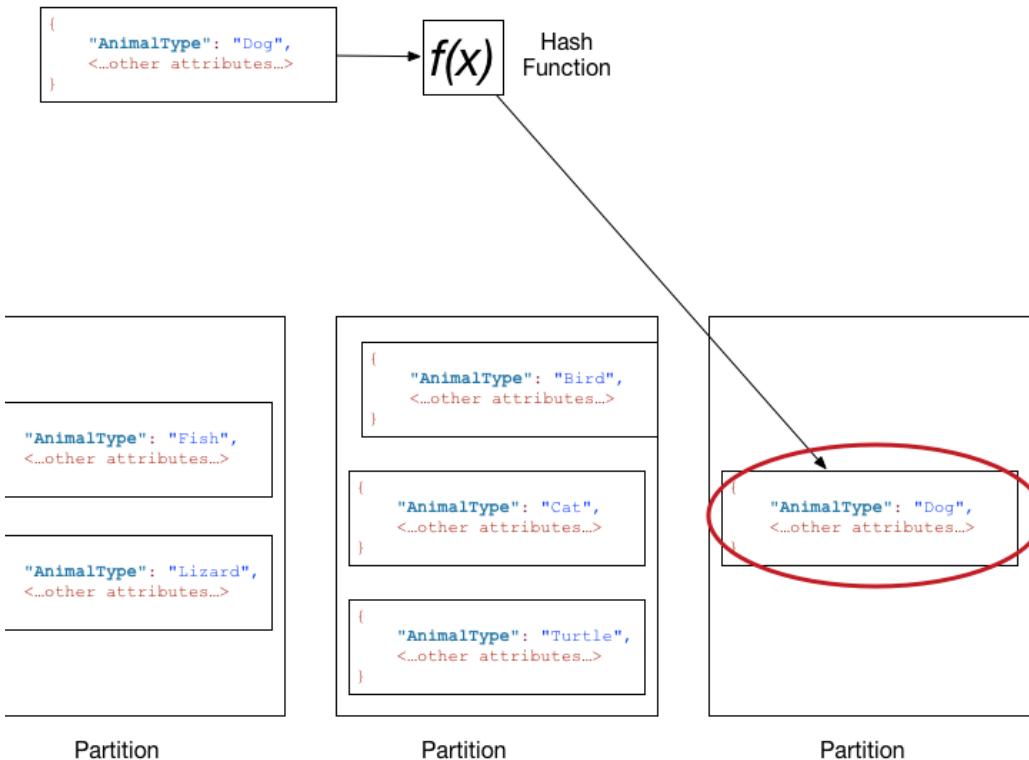
GSI

Primary Key		Projected Attributes...			
Partition Key					
Award	Player_ID	Game_ID	Score	Date	
Champ	Rick	Game_3	135,900	2018-01-19	
	Padma	Game_6	147,300	2018-02-02	
	Padma	Game_7	169,100	2018-03-10	

Aquí, Rick ha jugado tres partidas y ha conseguido el estado **Champ** en una de ellas. Padma ha jugado cuatro partidas y ha conseguido el estado **Champ** en dos de ellas. Observe que el atributo **Award** solamente está presente en los elementos en los que el usuario consiguió una recompensa. El índice secundario global asociado sería similar al siguiente:

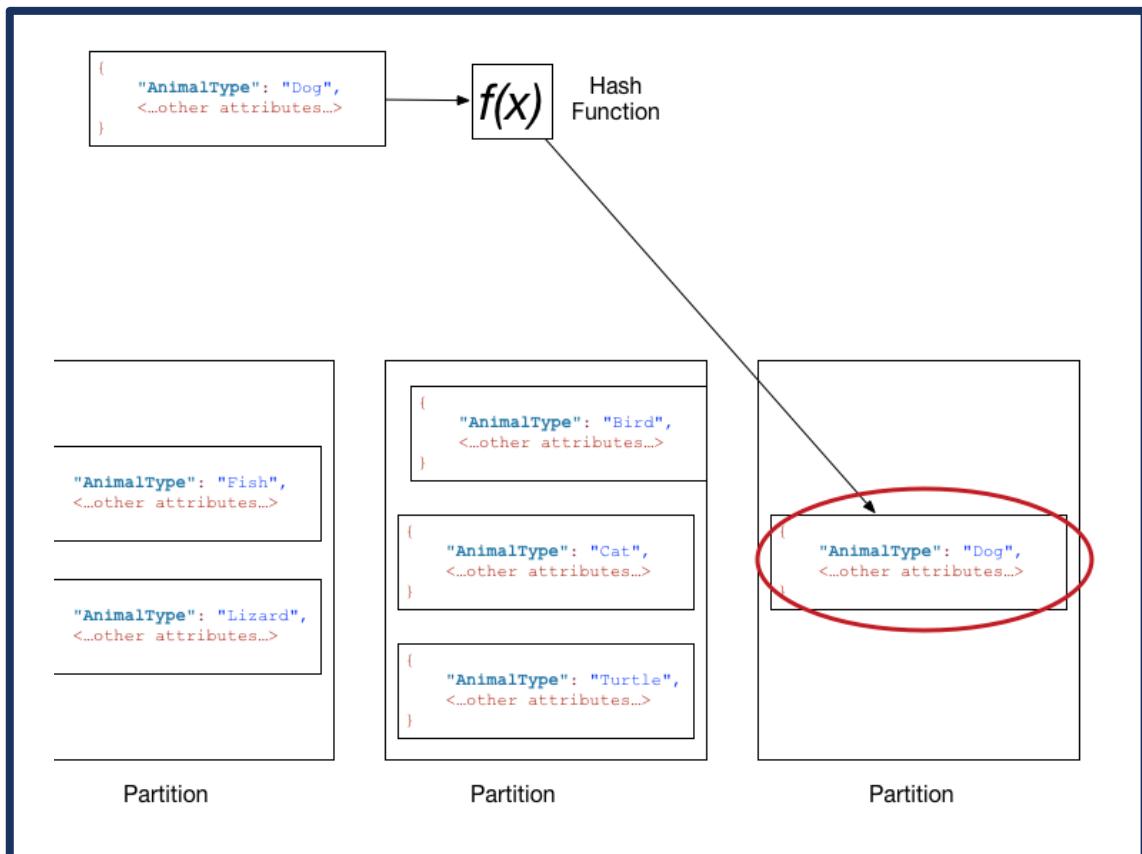
El índice secundario global únicamente contiene las puntuaciones máximas que se consultan con frecuencia, lo que conforma un pequeño subconjunto de los elementos de la tabla principal.

PARTICIONES



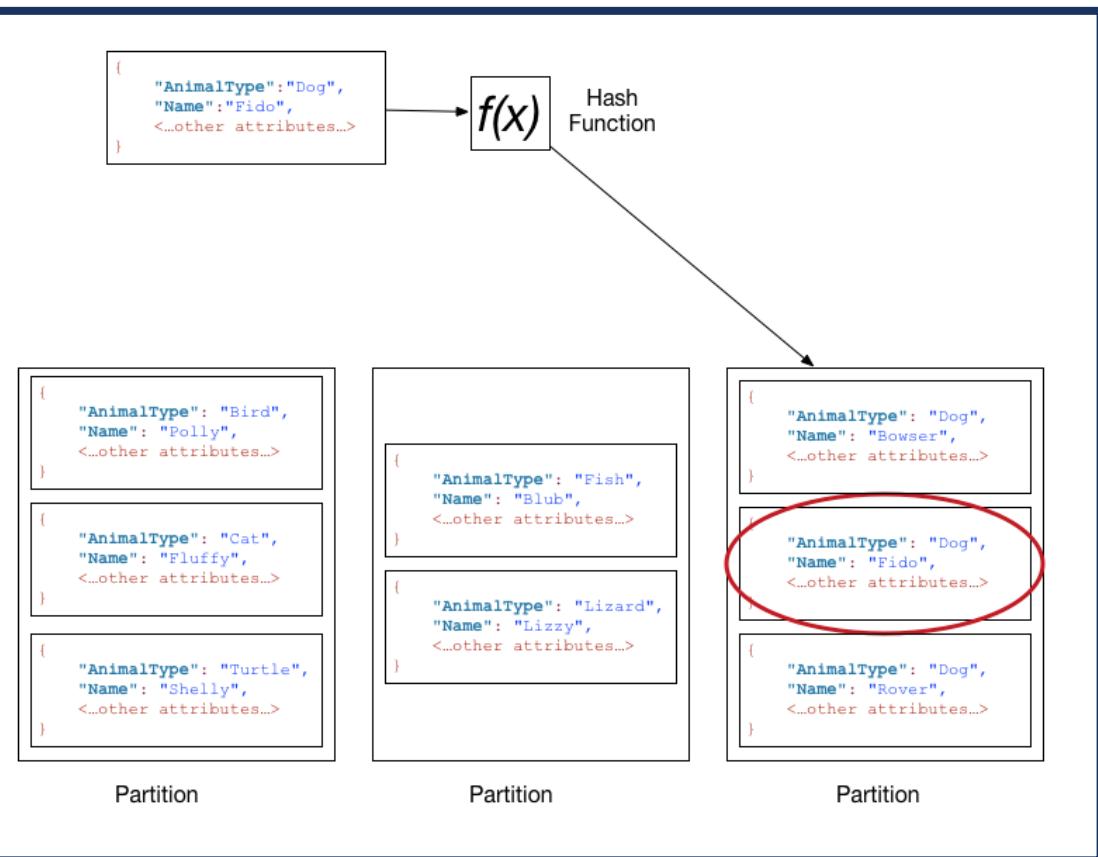
- Si la tabla tiene una partition key (solo clave de partición), DynamoDB almacenará y recuperará cada elemento basándose solo en ese valor.
- Para escribir un elemento en la tabla, DynamoDB utiliza el valor de la clave de partición y el hash determinada donde guardarlo.
- Para leer un elemento de la tabla, debe especificar el valor de clave de partición del elemento. DynamoDB utiliza este valor como información de entrada para la función hash para obtener la partición en la que se encuentra el elemento.
- Para distribuir los datos mejor se recomienda usar un atributo que varie mucho.

PARTICIONES



- Si la tabla tiene una partition key (solo clave de partición), DynamoDB almacenará y recuperará cada elemento basándose solo en ese valor.
- Para escribir un elemento en la tabla, DynamoDB utiliza el valor de la clave de partición y el hash determinada donde guardarlo.
- Para leer un elemento de la tabla, debe especificar el valor de clave de partición del elemento. DynamoDB utiliza este valor como información de entrada para la función hash para obtener la partición en la que se encuentra el elemento.
- Para distribuir los datos mejor se recomienda usar un atributo que varie mucho.

PARTICIONES



- La table tiene una partition key que se usa para determiner en que particion se salva cada elemento . El shot key se usa para almacenar los elementos ordenados.
- Para escribir, DynamoDB utiliza la partition key para determinar la partición y el hash determinada el orden para guardarla.
- Para leer un elemento de la tabla, debe especificar el valor de partition key a consultar.

NUMERO DE PARTICIONES

$$\# \text{ of Partitions} = \frac{\text{Table Size in bytes}}{10 \text{ GB}}$$

(for size)

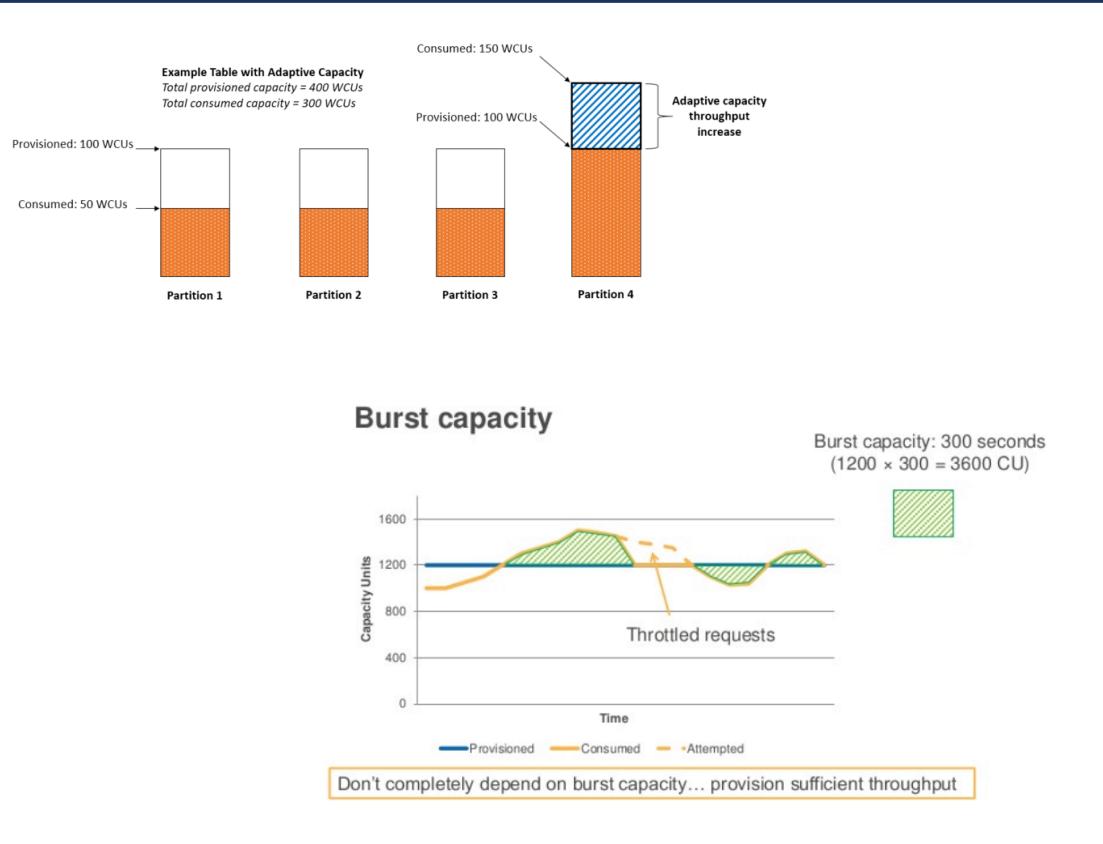
$$\# \text{ of Partitions} = \frac{RCU_{\text{for reads}}}{3000 \text{ RCU}} + \frac{WCU_{\text{for writes}}}{1000 \text{ WCU}}$$

(for throughput)

- Write capacity units (WCUs): 1 KB
- Read capacity units (RCUs): 4 KB
 - 1 RCU => 1 strongly consistent read
 - 1 RCU => 2 eventually consistent reads

- El número de particiones es el maximo de las particiones .

PARTICIONES



- No siempre es posible distribuir uniformemente la actividad de lectura y escritura. Si el acceso a los datos está desequilibrado, una partición "de moda" podría recibir un volumen mayor de tráfico de lectura y escritura que otras particiones.
- Para adaptarse a los patrones de acceso desiguales, la *capacidad de adaptación* de DynamoDB permite más lecturas y escrituras que la capacidad de la partición siempre que este por debajo del límite de tabla y el acumulado.

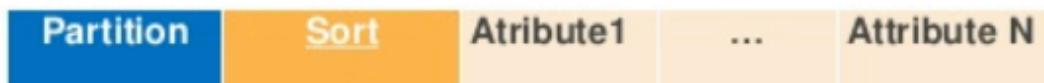
Query rather than scan

Query

- Specify partition key name
- Condition on sort key
- Cheap with high cardinality keys

Scan

- Reads all data
- Conditions available through filters
- Expensive for large tables



Hierarchical data structures

Games table

Gameld	Date	Host	Opponent	Status
d9bl3	2014-10-02	David	Alice	DONE
72f49	2014-09-30	Alice	Bob	PENDING
o2pnb	2014-10-08	Bob	Carol	IN_PROGRESS
b932s	2014-10-03	Carol	Bob	PENDING
ef9ca	2014-10-03	David	Bob	IN_PROGRESS

Approach 1: Query filter

Partition key Sort key

Secondary index

Opponent	Date	Gameld	Status	Host
Alice	2014-10-02	d9bl3	DONE	David
Carol	2014-10-08	o2pnb	IN_PROGRESS	Bob
Bob	2014-09-30	72f49	PENDING	Alice
Bob	2014-10-03	b932s	PENDING	Carol
Bob	2014-10-03	ef9ca	IN_PROGRESS	David

Query for incoming game requests

DynamoDB indexes provide partition and sort
What about queries for two equalities and a sort?

(hash) (?) (range)

```
SELECT * FROM Game
WHERE Opponent='Bob'
AND Status='PENDING'
ORDER BY Date DESC
```

Approach 1: Query filter

SELECT * FROM Game
WHERE Opponent='Bob'
ORDER BY Date DESC
FILTER ON Status='PENDING'

Secondary Index

Opponent	Date	Gameld	Status	Host
Alice	2014-10-02	d9bl3	DONE	David
Carol	2014-10-08	o2pnb	IN_PROGRESS	Bob
Bob	2014-09-30	72f49	PENDING	Alice
Bob	2014-10-03	b932s	PENDING	Carol
Bob	2014-10-03	ef9ca	IN_PROGRESS	David

Se recuperan 3 registros de los que luego se descarta 1.

Approach 2: Composite key

Status	Date	StatusDate
DONE	2014-10-02	DONE_2014-10-02
IN_PROGRESS	2014-10-08	IN_PROGRESS_2014-10-08
IN_PROGRESS	2014-10-03	IN_PROGRESS_2014-10-03
PENDING	2014-10-03	PENDING_2014-09-30
PENDING	2014-09-30	PENDING_2014-10-03

+

=

Partition key Sort key

Secondary Index

Opponent	StatusDate	Gameld	Host
Alice	DONE_2014-10-02	d9bl3	David
Carol	IN_PROGRESS_2014-10-08	o2pnb	Bob
Bob	IN_PROGRESS_2014-10-03	ef9ca	David
Bob	PENDING_2014-09-30	72f49	Alice
Bob	PENDING_2014-10-03	b932s	Carol

```
SELECT * FROM Game
WHERE Opponent='Bob'
AND StatusDate BEGINS_WITH 'PENDING'
```



Secondary index

Opponent	StatusDate	Gameld	Host
Alice	DONE_2014-10-02	d9bl3	David
Carol	IN_PROGRESS_2014-10-08	o2pnb	Bob
Bob	IN_PROGRESS_2014-10-03	ef9ca	David
Bob	PENDING_2014-09-30	72f49	Alice
Bob	PENDING_2014-10-03	b932s	Carol

Large and small attributes mixed

Partition key				Sort key
				Messages table
Recipient	Date	Sender	Message	
David	2014-10-02	Bob	...	
... 48 more messages for David ...				
David	2014-10-03	Alice	...	
Alice	2014-09-28	Bob	...	
Alice	2014-10-01	Carol	...	
(Many more messages)				



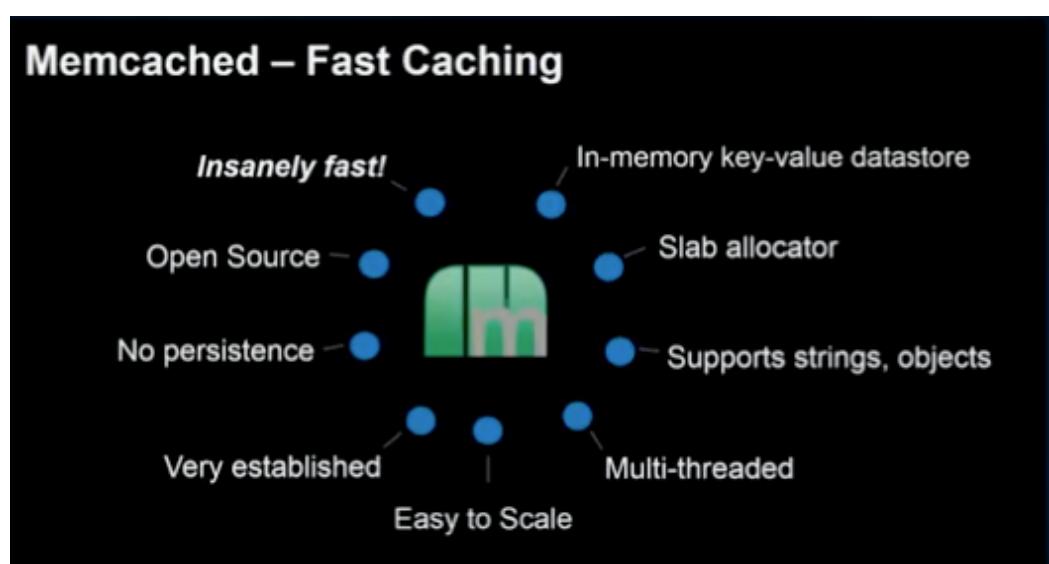
Computing inbox query cost

$$50 \times 256\text{KB} \times (1 \text{ RCU} / 4\text{KB}) \times (1 / 2) = 1600 \text{ RCU}$$

Items evaluated by query Average item size Conversion ratio Eventually consistent reads

All those RCUs against one partition key

MEMCACHE



- Amazon ElastiCache para **Memcached** es un servicio de almacén de clave-valor en memoria compatible con Memcached que se puede utilizar como caché o almacén de datos. Ofrece el rendimiento, la facilidad de uso y la simplicidad de Memcached. ElastiCache para Memcached es escalable, seguro y está completamente administrado, lo que lo transforma en el candidato ideal para casos de uso en los que los datos de acceso frecuente deben permanecer en memoria. Es una excelente opción para casos de uso de la web, aplicaciones móviles, videojuegos, tecnología publicitaria y E-Commerce.

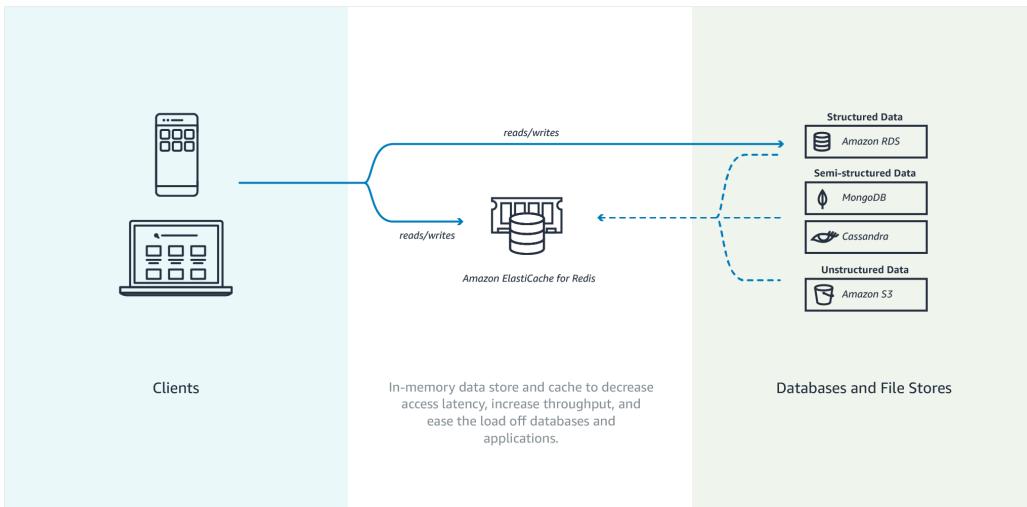
Qué es Amazon ElastiCache for Memcached?

Amazon ElastiCache es un servicio web que permite configurar, administrar y escalar fácilmente un almacén de datos en memoria distribuido o un entorno de caché en la nube. Proporciona una solución de almacenamiento en caché de alto rendimiento, escalable y rentable. Al mismo tiempo, ayuda a eliminar la complejidad asociada a la implementación y administración de un entorno de caché distribuido.

Las aplicaciones existentes que utilizan Memcached pueden utilizar ElastiCache sin prácticamente ninguna modificación. Las aplicaciones simplemente necesitan información acerca de los nombres de host y los números de puerto de los nodos de ElastiCache que se han implementado. La característica de detección automática de ElastiCache para Memcached permite que las aplicaciones identifiquen todos los nodos de un clúster de caché y se conecten a ellos. Así pues, usted no tiene que mantener una lista de nombres de host y números de puerto disponibles. De esta forma, sus aplicaciones se aíslan de forma eficaz de los cambios en la pertenencia a los nodos de un clúster.

ElastiCache for Memcached cuenta con diversas características que mejoran la fiabilidad para las implementaciones de producción críticas:

- Detección automática y recuperación de los errores de los nodos de caché.
- Detección automática de los nodos de un clúster habilitada, para que no sea necesario realizar cambios en la aplicación cuando se agreguen o quiten nodos.
- Colocación flexible de zonas de disponibilidad de nodos y clústeres.
- Integración con otros servicios de AWS como Amazon EC2, Amazon CloudWatch, AWS CloudTrail y Amazon SNS para ofrecer una solución de almacenamiento en caché en memoria administrada, segura y de alto desempeño.



El propósito principal del almacenamiento en memoria clave-valor es proporcionar un acceso ultrarrápido (latencia de milisegundos) y económico a las copias de datos. La mayoría de almacenes de datos tienen áreas de datos a los que se accede con frecuencia, pero que raramente se actualizan. Además, la consulta de una base de datos siempre será más lenta y más cara que la localización de una clave en una caché de par clave-valor. Algunas consultas de bases de datos tienen un costo especialmente elevado. Un ejemplo de ello son las consultas que implican uniones en varias tablas o consultas con cálculos intensivos.

Considere la posibilidad de almacenar en caché los datos si:

- Los datos resultan lentos o caros de adquirir en comparación con la recuperación de la memoria caché.
- Los usuarios acceden a sus datos con frecuencia.
- Sus datos siguen siendo relativamente iguales, o si cambian rápidamente, la obsolescencia no es un gran problema.

Nodos de ElastiCache

Un *nodo* es el componente básico más pequeño de toda implementación de ElastiCache. Un nodo puede existir de forma aislada o guardando alguna relación con otros nodos.

Un nodo es un fragmento de tamaño fijo de RAM segura conectada a la red. Cada nodo ejecuta una instancia de Memcached.

Si es necesario, puede escalar los nodos de un clúster para ampliar o reducir a un tipo de instancia diferente.

Escalado de clústeres de Memcached

Acción	Tema/enlace
Escalado ascendente	Adición de nodos a un clúster
Escalado descendente	Eliminación de nodos de un clúster
Cambios de tipos de nodos	Escalado vertical de Memcached

Cada nodo tiene su propio puerto y nombre de servicio de nombres de dominio (DNS). Se admiten varios tipos de nodos de ElastiCache, cada uno de los cuales tiene asociada una cantidad de memoria y unos recursos informáticos diferentes.

Puede adquirir nodos a medida que sea necesario, donde solo paga por el uso que haga de un nodo (*on demand*). También puede adquirir nodos reservados a una tarifa por hora considerablemente reducida.

La *detección automática* es la capacidad de que los programas cliente identifiquen automáticamente todos los nodos de un clúster de caché e inicien y mantengan las conexiones a todos estos nodos. Con la detección automática, la aplicación no tiene que conectarse manualmente a cada nodo individual. En cambio, se conecta a un punto de enlace de configuración. El extremo de entrada contiene la configuración DNS entradas de CNAME para cada uno de los extremos de los nodos de caché. Por lo tanto, al conectarse al punto de enlace de configuración, la aplicación dispone inmediatamente de información sobre todos los nodos del clúster y puede conectarse a todos y cada uno de ellos.

Copy Node Endpoint

Configuration Endpoint

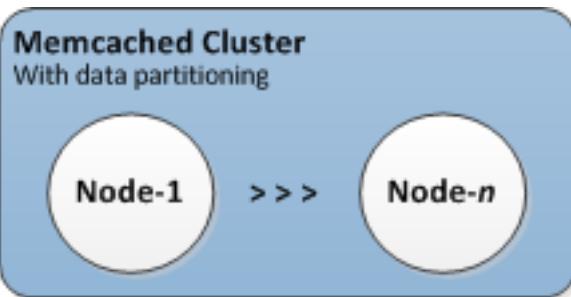
[.cfg.usw2.cache.amazonaws.com:11211](http://cfg.usw2.cache.amazonaws.com:11211)

Use the ElastiCache Cluster Client and Configuration Endpoint to automatically discover hosts.

[Download the client.](#)

Node Endpoints

.0001.usw2.cache.amazonaws.com:11211
.0002.usw2.cache.amazonaws.com:11211
.0003.usw2.cache.amazonaws.com:11211

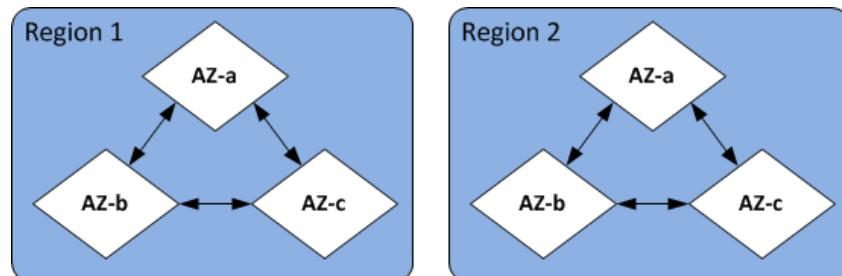


Configuraciones típicas de clúster

Memcached admite hasta 100 nodos por cliente y región de AWS, donde cada clúster tiene 1–20 nodos. Los datos se partitionan entre los nodos del clúster. La base de datos se divide entre los nodos. Su aplicación lee y escribe en cada punto de conexión del nodo.

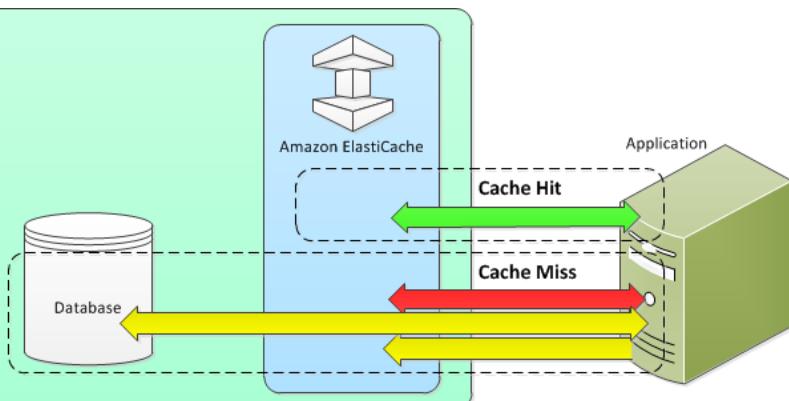
Para mejorar la tolerancia a errores, los nodos de Memcached puede usar varias zonas de disponibilidad en la región de AWS del clúster. De ese modo, un error de una zona de disponibilidad tendrán un impacto mínimo en todo su clúster y su aplicación

A medida que cambie la demanda de su clúster de Memcached, puede escalar de forma ascendente o descendente al añadir o quitar nodos, con lo que se vuelven a partitionar los datos entre el nuevo número de nodos. Para partitionar los datos, se recomienda hacer un uso consistente de la función hash.



Regiones y zonas de disponibilidad de AWS

Amazon ElastiCache para Memcached está disponible en varias regiones de AWS de todo el mundo . Cada región de AWS se ha diseñado para que esté totalmente aislada de las demás regiones AWS. Dentro de cada una de ellas hay varias zonas de disponibilidad. Al lanzar los nodos en zonas de disponibilidad diferentes, puede lograr la máxima tolerancia a errores. Para obtener más información acerca de las regiones y las zonas de disponibilidad de AWS.



Carga diferida (Lazy Loading)

es una estrategia de caché que carga los datos en la caché solo cuando es necesario. Es un almacén de claves/valores en memoria que se sitúa entre su aplicación y el almacén de datos (base de datos) al que accede. Siempre que su aplicación solicite datos, primero realizará una solicitud a la caché de ElastiCache. Si los datos existen en la caché y son actuales, ElastiCache devuelve los datos a su aplicación. Si los datos no existen en la caché o han caducado, la aplicación solicita los datos de su almacén de datos. El almacén de datos devuelve luego los datos a su aplicación. La aplicación escribe después los datos recibidos del almacén en la caché. De esa forma, se puede recuperar más rápido la próxima vez que se solicite.

Las ventajas :

- Solo se almacenan en la caché los datos solicitados.
- Puesto que nunca se solicita la mayoría de los datos, la carga diferida evita llenar la caché con datos que no se solicitan.
- Los errores de nodo no son fatales para su aplicación.
- Cuando se produce un error en un nodo y se reemplaza por un nodo nuevo y vacío, la aplicación sigue funcionando, aunque con mayor latencia. A medida que se realizan solicitudes al nuevo nodo, cada error de caché genera una consulta de la base de datos. Al mismo tiempo, se añade la copia de datos a la caché para que se recuperen las solicitudes posteriores de la caché.

Las desventajas

- Existe una penalización de errores de caché.
- Cada error de caché genera tres acciones:
 1. Solicitud inicial de los datos a la caché
 2. Consulta de los datos en la base de datos
 3. Escritura de los datos en la caché
- Datos obsoletos.
- Si los datos se escriben en la caché solo cuando hay un error de caché, los datos en la caché pueden volverse obsoletos. Esto se produce porque no hay actualizaciones en la caché cuando se modifican los datos en la base de datos.

Escritura indirecta (Write-Through)

La estrategia de escritura indirecta añade o actualiza los datos de la caché siempre que se escriben datos en la base de datos.

Las ventajas :

Los datos de la caché nunca quedan obsoletos.

Los datos de la caché se actualizan cada vez que se escriben en la base de datos, estos siempre se mantienen actualizados.

Penalización de escritura frente a penalización de lectura.

Toda operación de escritura implica dos acciones:

1. Una operación de escritura en la caché
2. Una operación de escritura en la base de datos

Estas acciones añaden latencia al proceso. Dicho esto, los usuarios finales suelen ser más tolerantes con la latencia a la hora de actualizar datos que con la latencia a la hora de recuperar datos.

Las desventajas

Pérdida de datos

Si activa un nuevo nodo, ya sea debido a un error del nodo o el escalado ascendente, faltan datos. Estos datos continúan faltando hasta que se añadan o actualicen en la base de datos.

Pérdida de caché

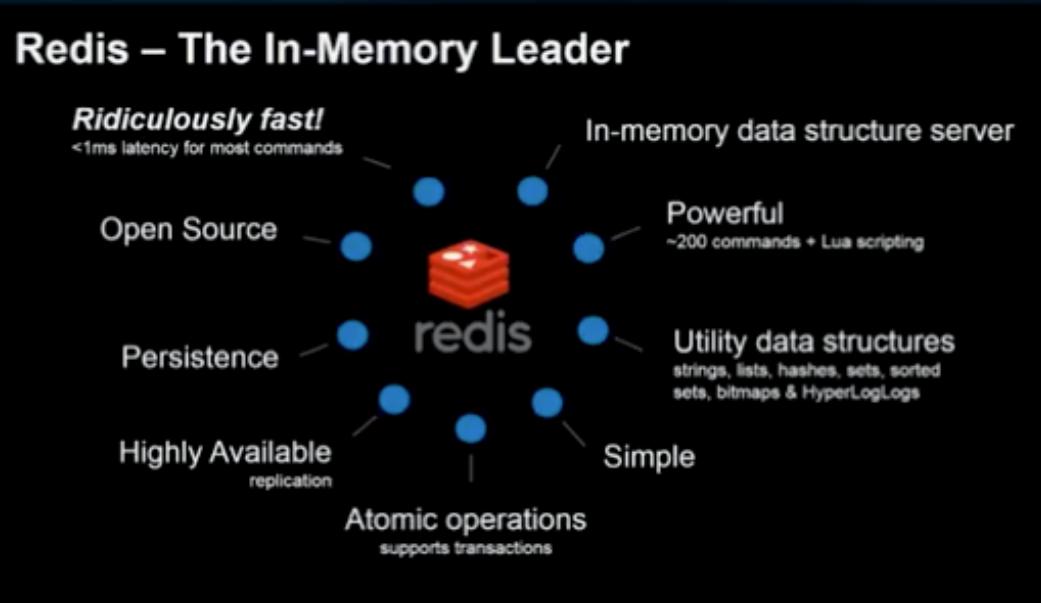
La mayoría de los datos nunca se leen, lo que supone un desperdicio de recursos. Al añadir un valor de tiempo de vida (TTL), puede minimizar el espacio perdido.

Añadir TTL

La carga diferida da lugar a que los datos queden obsoletos, pero no da error con nodos vacíos. La escritura indirecta mantiene los datos siempre actualizados, pero puede dar error con nodos vacíos y puede llenar la caché con datos superfluos. Al añadir un valor de tiempo de vida (TTL) a cada escritura, puede disfrutar de las ventajas de cada estrategia. Al mismo tiempo, evita en gran medida abarrotar la caché con datos adicionales.

El tiempo de vida (TTL) es un valor entero que especifica el número de segundos que transcurrirán hasta que la clave caduque. Cuando una aplicación intenta leer una clave caducada, se trata como si no se encontrara la clave. Se consulta la clave en la base de datos y se actualiza la caché. Este enfoque no garantiza que un valor no esté obsoleto. Sin embargo, evita que los datos queden demasiado obsoletos y se actualizan los valores de la caché con frecuencia desde la base de datos.

REDIS



Amazon ElastiCache for **Redis** es un almacén de datos en memoria increíblemente rápido que ofrece una latencia inferior a un milisegundo para aplicaciones en tiempo real a escala de Internet. Creado sobre Redis de código abierto y compatible con las API de Redis, ElastiCache para Redis se puede usar con clientes de Redis y utiliza el formato de datos de Redis abierto para el almacenamiento. Las aplicaciones de Redis autoadministradas pueden funcionar sin problemas con ElastiCache para Redis sin realizar modificaciones de código. ElastiCache para Redis combina la velocidad, simplicidad y versatilidad del almacén de código abierto Redis con la manejabilidad, seguridad y escalabilidad de Amazon para atender las aplicaciones en tiempo real de mayor demanda de las áreas de videojuegos, tecnología publicitaria, comercio electrónico, sanidad, servicios financieros e IoT.

Clúster de caché o nodo frente a nodo

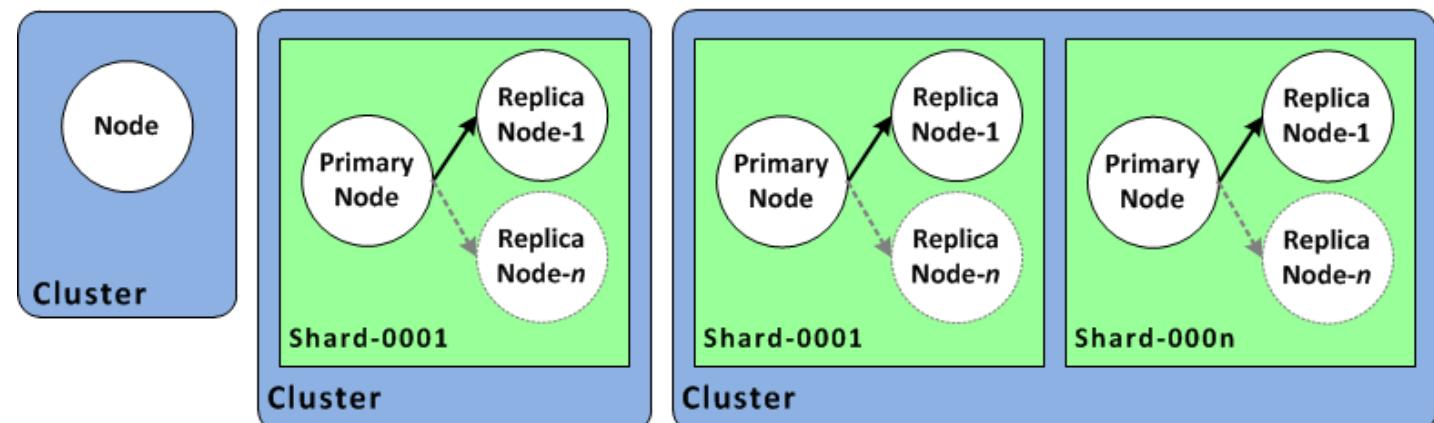
Existe una relación de uno a uno entre un nodo y un clúster de la caché cuando no hay nodos de réplica. Además, la consola de ElastiCache suele utilizar los términos de forma intercambiable. La consola utiliza el término *nodo* para todo. La única excepción es el botón **Create Cluster**, que lanza el proceso para crear un clúster con o sin nodos de réplica.

Clúster frente a grupo de replicación

La consola utiliza ahora el término clúster para todos los clústeres de ElastiCache para Redis. La consola utiliza el término clúster en todas estas circunstancias:

- Cuando el clúster es un clúster de Redis de un único nodo.
- Cuando el clúster es un clúster de Redis (modo clúster deshabilitado) que admite la replicación en un único fragmento (en la API y la CLI, denominado *grupo de nodos*).
- Cuando el clúster es un clúster de Redis (modo clúster habilitado) que admite la replicación en 1–90 fragmentos.

ElastiCache for Redis: Console View



Configuración de su cliente de ElastiCache para un equilibrio de carga eficaz (Memcached de varios nodos)

Para utilizar varios nodos de Memcached de ElastiCache de forma eficaz, debe poder repartir sus claves de caché entre los nodos. Una manera sencilla de equilibrar la carga de un clúster con n nodos es calcular el hash de la clave del objeto y aplicar la función mod al resultado mediante $n - \text{hash}(\text{key}) \bmod n$. El valor resultante (de 0 a $n-1$) es el número del nodo en el que deberá colocar el objeto.

Este enfoque es sencillo y funciona bien siempre que el número de nodos (n) sea constante. Sin embargo, siempre que añada o elimine un nodo del clúster, el número de claves que deben moverse será $(n-1)/n$ (donde n es el nuevo número de nodos). Por lo tanto, este enfoque da como resultado el traslado de un gran número de claves, lo que se traduce en un gran número de pérdidas iniciales de caché, especialmente cuando el número de nodos es elevado. En el mejor de los casos, al escalar de 1 a 2 resultados de nodos, se obtienen $(2-1)/2$ (50 por ciento) de claves para trasladar. Al escalar de 9 a 10 nodos, se obtienen $(10-1)/10$ (90 por ciento) de claves para trasladar. Si va a ampliar debido a un pico de tráfico, no deseará tener muchas pérdidas de caché. Un gran número de pérdidas de caché devuelve coincidencias con la base de datos, que ya está sobrecargada por el pico de tráfico.

La solución a este dilema es un uso consistente de la función hash. Un uso consistente de hash emplea un algoritmo según el cual, siempre que se añada o elimine un nodo de un clúster, el número de claves que deba moverse será aproximadamente $1/n$ (donde n es el nuevo número de nodos). En el peor de los casos, al escalar de 1 a 2 resultados de nodos, se obtienen $1/2$ (50 por ciento) de claves para trasladar. Al escalar de 9 a 10 nodos, se obtienen $1/10$ (10 por ciento) de claves para trasladar.

Como usuario, deberá controlar qué algoritmo de hash se usa para los clústeres de varios nodos. Recomendamos configurar sus clientes para que utilicen hash de forma consistente. Afortunadamente, hay muchas bibliotecas de cliente de Memcached en la mayoría de los idiomas comunes que implementan hash de forma consistente. Consulte la documentación de la biblioteca que va a utilizar para ver si admite el uso consistente de hash y saber cómo implementarlo.

Redis topologies

Cluster Mode Disabled

Max Storage 407 GiB

Primary Endpoint



| Primary

Vertically Scaled



0–5 Replicas

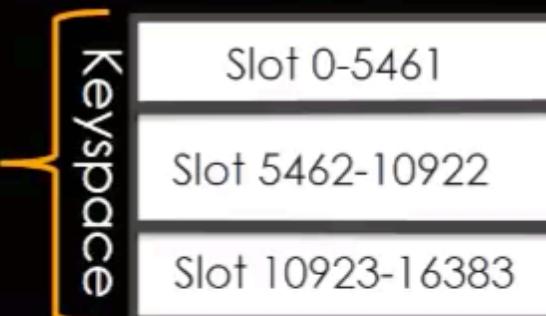
Cluster Mode Enabled

Max Storage 6+ TiB

Configuration Endpoint



Horizontally Scaled



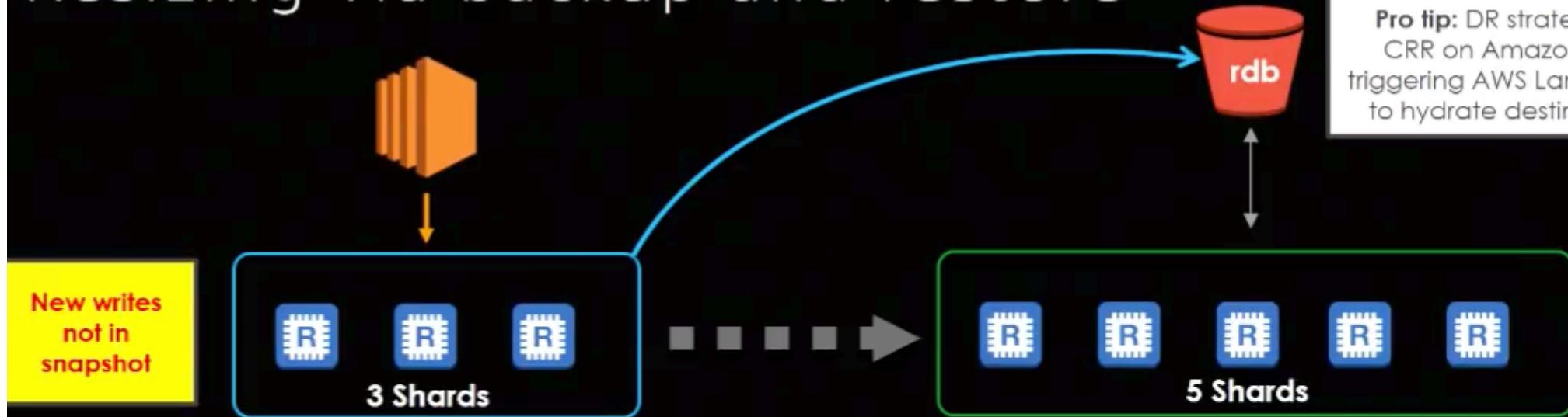
0–5 Replicas

1–15 Primaries/Shards

Redis cluster-mode enabled vs. disabled

Feature	Enabled	Disabled
Failover	15–30 sec (Non-DNS)	~1.5 min (DNS-based)
Failover risk	<ul style="list-style-type: none">Writes affected—partial dataset (less risk with more partitions)Reads available	<ul style="list-style-type: none">Writes affected on entire datasetReads available
Performance	Scales with cluster size (90 nodes—15 primaries + 0–5 replicas per shard)	6 nodes (1 primary + 0–5 replicas)
Max connections	<ul style="list-style-type: none">Primaries ($65,000 \times 15 = 975,000$)Replicas ($65,000 \times 75 = 4,875,000$)	<ul style="list-style-type: none">Primary: 65,000Replicas: ($65,000 \times 5 = 325,000$)
Storage	6+ TiB	407 GB
Cost	Smaller nodes but more \$\$	Larger nodes less \$

Resizing via backup and restore

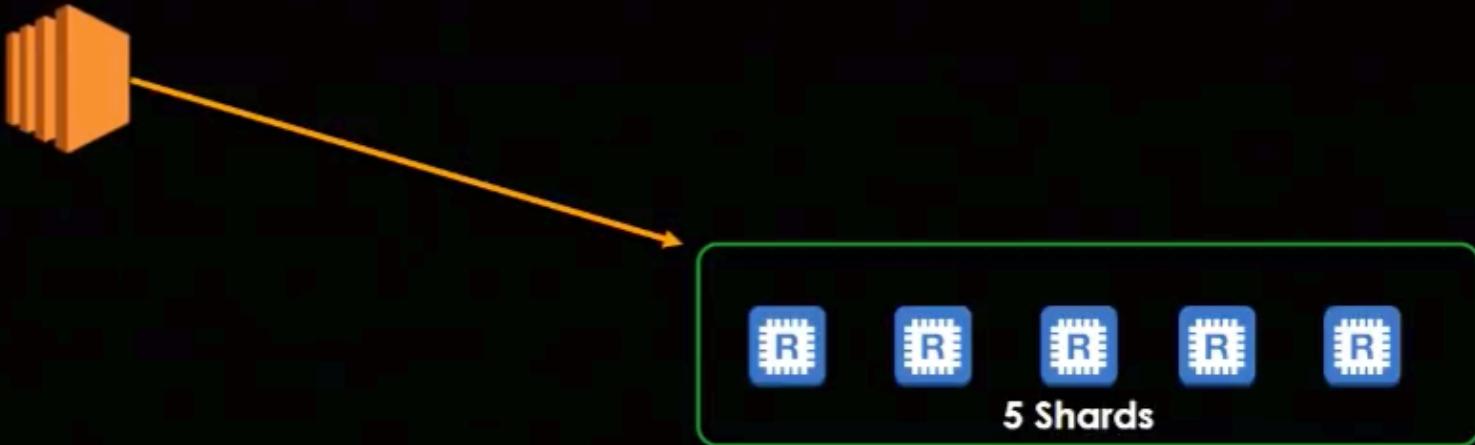


Step 1 `aws elasticache create-snapshot --replication-group-id redisclusterID --snapshot-name sname`

Step 2 `aws elasticache copy-snapshot --source-snapshot-name sname --target-snapshot-name sname --target-bucket s3bucketname`

Step 3 `aws elasticache create-replication-group --replication-group-id NewRedisClusterID ... --snapshot-arns arn:aws:s3:::bucketname/redisbackup-0001.rdb, etc.`

Resizing via backup and restore



- Step 1** `aws elasticache create-snapshot --replication-group-id redisclusterID --snapshot-name sname`
- Step 2** `aws elasticache copy-snapshot --source-snapshot-name sname --target-snapshot-name sname --target-bucket s3ucketname`
- Step 3** `aws elasticache create-replication-group --replication-group-id NewRedisClusterID ... --snapshot-arns arn:aws:s3:::bucketname/redisbackup-0001.rdb, etc.`
- Step 4** Once the new cluster is up, update your app with new Amazon ElastiCache endpoint, then terminate old cluster

Scenario 1: Single primary failure

Mitigation:

1. Automatic failure detection and replica promotion (~15–30 s)
2. Repair failed node



Scenario 2: Majority of primaries fail

Mitigation: Redis enhancements on ElastiCache

- Automatic failure detection and replica promotion
- Repair failed nodes



	Memcached	Redis
<u>Sub-millisecond latency</u>	Yes	Yes
<u>Developer ease of use</u>	Yes	Yes
<u>Data partitioning</u>	Yes	Yes
<u>Support for a broad set of programming languages</u>	Yes	Yes
<u>Advanced data structures</u>	-	Yes
<u>Multithreaded architecture</u>	Yes	-
<u>Snapshots</u>	-	Yes
<u>Replication</u>	-	Yes
<u>Transactions</u>	-	Yes
<u>Pub/Sub</u>	-	Yes
<u>Lua scripting</u>	-	Yes
<u>Geospatial support</u>	-	Yes

Use of Memcached

- 1. In cases when you need the simplest model for caching.
- 2. To distribute your data over multiple nodes. It is also useful in those case when you need to run large nodes with multiple cores and threads.
- 3. For caching objects like the database, you can use Memcached.
- 4. To increase or decrease your system by scaling out and for addition and deletion of nodes you can use Memcached.

Use of Redis

- 1. For complex data types like hash, strings, etc.
- 2. To sort and rank in-memory datasets.
- 3. Automatic failover in case of failover.
- 4. For replication of data to one or more read replicas from primary for availability.
- 5. Backup and restore features required in caching.
- 6. Pub and sub-capabilities are required.
- 7. Persistence of key stores.

Requirement	Memcached	Redis
Simple Cache to offload database	Yes	Yes
Ability to scale horizontally	Yes	No
Multithreaded performance	Yes	No
Advance data type	No	Yes
Ranking/sorting data sets	No	Yes
Pub/Sub capabilities	No	Yes
Persistence	No	Yes
MultiAZ	No	Yes
Backup and restore capabilities	No	Yes

Table 7-01. Memcached v/s Redis



EXAM TIP: According to your requirement, Amazon ElastiCache allows you to choose Cache engine. Memcached is used in case of simple storage of in-memory objects, which are scaled horizontally. Redis engine is used in case of backup and restore data. It needs a large number of read replica, and it is used for data structures like sorted sets and lists.

HPC

AWS proporciona la infraestructura de nube más elástica y escalable para ejecutar sus aplicaciones de HPC. Con una capacidad virtualmente ilimitada, ingenieros, investigadores y propietarios de sistemas HPC pueden innovar más allá de las limitaciones de una infraestructura HPC local. AWS ofrece un conjunto integrado de servicios que proporciona todo lo necesario para crear y administrar, de forma rápida y fácil, clústeres HPC en la nube para ejecutar las cargas de trabajo más intensas a través de diferentes sectores verticales de la industria. Estas cargas de trabajo abarcan las aplicaciones HPC tradicionales, como genómica, química computacional, creación de modelos de riesgo financiero, ingeniería asistida por computadora, predicción del tiempo e imaginería sísmica, además de aplicaciones emergentes, como aprendizaje automático, aprendizaje profundo y conducción autónoma.

HPC en AWS elimina los largos tiempos de espera y la pérdida de productividad que se asocia a menudo con clústeres HPC on-premises. La configuración flexible y la escalabilidad prácticamente ilimitada, le permite aumentar o disminuir su infraestructura según sus cargas de trabajo, y no al revés. Asimismo, gracias al acceso a una amplia gama de servicios basados en la nube como Data Analytics, Artificial Intelligence (AI), y Machine Learning (ML), puede redefinir los flujos de trabajo HPC para innovar con mayor rapidez.

En la actualidad, hay aplicaciones HPC basadas en la nube ejecutándose en AWS que en cualquier otra nube. Clientes como Bristol-Myers Squibb, FINRA, BP y Autodesk, confían en AWS para ejecutar sus cargas de trabajo HPC más críticas.

Resultados con mayor rapidez

Al mover sus cargas de trabajo HPC a AWS puede obtener acceso instantáneo a la capacidad de infraestructura que necesite para ejecutar sus aplicaciones HPC. HPC en AWS elimina los tiempos de espera y las largas colas de trabajo que se asocian con frecuencia a recursos HPC on-premises limitados, lo que le ayuda a obtener resultados más rápido. Además, gracias al acceso a una amplia gama de servicios basados en la nube, puede innovar más rápido al combinar flujos de trabajo HPC con nuevas tecnologías, como inteligencia artificial y aprendizaje automático.

Configuraciones flexibles

Mover las cargas de trabajo HPC a la nube puede ayudar a aumentar la productividad al adaptar la configuración de la infraestructura a la aplicación. Con HPC en AWS, los ingenieros ya no están limitados a realizar su trabajo con la configuración disponible. Cada carga de trabajo se puede ejecutar en su propio clúster a petición, utilizando un conjunto de servicios óptimo para esta aplicación. Tanto miembros individuales como equipos pueden aumentar o disminuir con rapidez dichos recursos según sea necesario, al poner en marcha o retirar clústeres HPC en cuestión de minutos, en lugar de días o semanas.

Operaciones rentables

Con HPC en AWS, no hay gastos de capital iniciales ni ciclos largos de adjudicación. Solo paga por la capacidad que usa, y nuestros modelos de tarificación flexibles brindan un ahorro de costos significante cuando procesa cargas de trabajo sin estado y con flexibilidad temporal. Puede migrar con rapidez tecnologías más recientes tan pronto como estas estén disponibles en AWS. De este modo, se elimina el riesgo de que los clústeres HPC on-premise se queden obsoletos o se utilicen poco a medida que sus necesidades cambien. El resultado es un gasto HPC más eficiente, y menor desperdicio de recursos.



Determine the compute, storage and networking requirements for your HPC codes

Create and configure your HPC cluster using AWS ParallelCluster

Submit your HPC job using a job scheduler, or directly from the CLI

Visualize your results using NICE DCV or pipe your results into other AWS services like ML/DL or Athena for post processing

Monitor performance and change your cluster as needed when your workloads evolve

HPC en AWS - Servicios de componentes

Los servicios que se enumeran a continuación como componentes de la solución HPC son un gran punto de partida para configurar y administrar su clúster HPC. AWS publica de forma constante nuevos servicios y funciones, y le animamos a que explore cómo los servicios de nube adyacentes le pueden ayudar a redefinir sus flujos de trabajo HPC.

Administración y transferencia de datos	Computación y redes	Almacenamiento	Automatización y organización	Visualización	Operaciones y administración	Seguridad y conformidad
AWS DataSync	Instancias de Amazon EC2 (CPU, GPU, FPGA)	Volúmenes de IOPS provisionadas de Amazon EBS	AWS Batch	NICE DCV	Amazon CloudWatch	AWS Identity and Access Management
AWS Snowball	Amazon EC2 Spot	Amazon FSx for Lustre	AWS ParallelCluster	Amazon AppStream 2.0	Presupuestos de AWS	
AWS Snowmobile	AWS Auto Scaling	Amazon EFS	NICE EnginFrame			
AWS DirectConnect	Grupos de ubicación	Amazon S3				
	Redes mejoradas					
	Elastic Fabric Adapter					
	AWS VPC					

QUÉ ES AWS PARALLELCLUSTER

AWS ParallelCluster es una herramienta de administración de clústeres de código abierto compatible con AWS que le ayuda a implementar y administrar clústeres de informática de alto rendimiento (HPC) en la nube de AWS. AWS ParallelCluster se basa en el proyecto CfnCluster de código abierto y permite crear con rapidez un entorno informático de HPC en AWS. Configura automáticamente los recursos informáticos y el sistema de archivos compartido. Puede usar AWS ParallelCluster con una variedad de programadores por lotes, como AWS Batch, SGE, Torque, y Slurm. AWS ParallelCluster facilita una prueba de inicio rápido de implementaciones de concepto e implementaciones de producción. También puede crear flujos de trabajo de nivel superior como, por ejemplo, un portal de genómica que automatice el flujo de trabajo completo de la secuenciación del ADN encima de AWS ParallelCluster.

Los siguientes servicios de Amazon Web Services (AWS) se utilizan en AWS ParallelCluster.

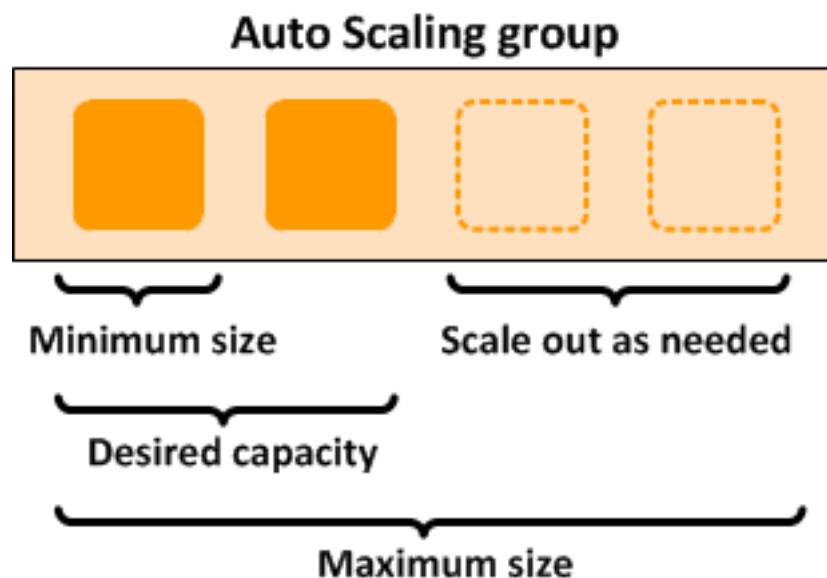
Temas

- [AWS Auto Scaling](#)
- [AWS Batch](#)
- [AWS CloudFormation](#)
- [Amazon CloudWatch](#)
- [AWS CodeBuild](#)
- [Amazon DynamoDB](#)
- [Amazon Elastic Block Store](#)
- [Amazon Elastic Compute Cloud](#)
- [Amazon EC2 Container Registry](#)
- [AWS Identity and Access Management](#)
- [AWS Lambda](#)
- [Amazon Simple Notification Service](#)
- [Amazon Simple Queue Service](#)
- [Amazon Simple Storage Service](#)

AWS Auto Scaling

AWS Auto Scaling se utiliza para administrar las instancias de ComputeFleet. Dichas instancias se administran como un grupo de Auto Scaling y se pueden administrar de manera elástica por carga de trabajo, o bien pueden ser estáticas y administrarse con la configuración. AWS Auto Scaling no se usa con clústeres de AWS Batch.

Para obtener información detallada acerca de AWS Auto Scaling, consulte <https://aws.amazon.com/autoscaling/>.



AWS Batch

AWS Batch es el programador de trabajo administrado por AWS que aprovisiona de forma dinámica la cantidad óptima y el tipo de recursos informáticos (por ejemplo, instancias optimizadas para memoria o CPU). Aprovisiona recursos en función del volumen y los requisitos de los trabajos por lotes que se envían. Con AWS Batch no es necesario instalar ni administrar los clústeres de servidores o el software de computación por lotes para ejecutar sus trabajos.

AWS Batch se usa solo con clústeres de AWS Batch.

Para obtener más información, consulte <https://aws.amazon.com/batch/>.

AWS CloudFormation

AWS CloudFormation es el servicio principal que AWS ParallelCluster utiliza. Cada clúster se representa como una stack. Todos los recursos que el clúster necesita se definen en la plantilla de AWS CloudFormation de AWS ParallelCluster. Normalmente, los comandos de la CLI de AWS ParallelCluster se asignan a comandos de stack de AWS CloudFormation, como crear, actualizar y eliminar. Las instancias que se lanzan dentro de un clúster realizan llamadas HTTPS al punto de enlace de AWS CloudFormation para la región donde se ha lanzado el clúster.

Para obtener información detallada sobre AWS CloudFormation, consulte <https://aws.amazon.com/cloudformation/>

Amazon Simple Storage Service

Amazon Simple Storage Service (Amazon S3) se usa para almacenar las plantillas AWS ParallelCluster usadas en cada región. Se puede configurar AWS ParallelCluster para permitir que las herramientas de CLI o SDK usen Amazon S3.

Cuando se utiliza un clúster de AWS Batch, se usa un bucket de Amazon S3 en la cuenta del cliente para almacenamiento. Por ejemplo, almacena artefactos de los que se sirve la creación de la imagen de Docker y scripts de trabajos enviados.

Para obtener información detallada, consulte <https://aws.amazon.com/s3/>.

Amazon CloudWatch

Amazon CloudWatch (CloudWatch) se utiliza para registrar los pasos de creación de la imagen de Docker y la salida y los errores estándar de los trabajos de AWS Batch.

CloudWatch se usa solo con clústeres de AWS Batch.

Para obtener más detalles, consulte <https://aws.amazon.com/cloudwatch/>.

AWS CodeBuild

AWS CodeBuild (CodeBuild) se utiliza para crear imágenes de Docker de forma automática y transparente en el momento de crear el clúster.

CodeBuild se usa solo con clústeres de AWS Batch.

Para obtener información detallada, consulte <https://aws.amazon.com/codebuild/>.

Amazon DynamoDB

Amazon DynamoDB (DynamoDB) se utiliza para almacenar el estado mínimo del clúster. MasterServer realiza un seguimiento de las instancias aprovisionadas en una tabla de DynamoDB.

DynamoDB no se usa con clústeres de AWS Batch.

Para obtener información detallada, consulte <https://aws.amazon.com/dynamodb/>.

Amazon Elastic Block Store

Amazon Elastic Block Store (Amazon EBS) proporciona el almacenamiento persistente de los volúmenes compartidos. Toda la configuración de Amazon EBS se puede transmitir a través de la configuración. Los volúmenes de Amazon EBS se pueden inicializar vacíos o desde una instantánea de Amazon EBS existente.

Para obtener información detallada acerca de Amazon EBS, consulte <https://aws.amazon.com/ebs/>.

Amazon Elastic Compute Cloud

Amazon Elastic Compute Cloud (Amazon EC2) proporciona la capacidad de computación para AWS ParallelCluster. MasterServer y ComputeFleet son instancias Amazon EC2. Se puede seleccionar cualquier tipo de instancia compatible con HVM. MasterServer y ComputeFleet pueden ser tipos de instancias diferentes y ComputeFleet también se puede lanzar como instancia de spot. Los volúmenes de almacenamiento de instancias que se encuentran en las instancias se montan como volúmenes LVM fragmentados.

Para obtener información detallada acerca de Amazon EC2, consulte <https://aws.amazon.com/ec2/>.

Amazon EC2 Container Registry

Amazon EC2 Container Registry (Amazon ECR) almacena las imágenes de Docker que se crean en el momento de crear el clúster. Luego, AWS Batch utiliza dichas imágenes para ejecutar los contenedores para los trabajos enviados.

Amazon ECR se usa solo con clústeres de AWS Batch.

Para obtener más información, consulte <https://aws.amazon.com/ecr/>.

AWS Identity and Access Management

AWS Identity and Access Management (IAM) se usa dentro de AWS ParallelCluster. Proporciona un rol de IAM menos privilegiado para Amazon EC2 para la instancia que es específica de cada clúster individual. A las instancias de AWS ParallelCluster se les da acceso solo a las llamadas a la API específicas necesarias para implementar y administrar el clúster.

Con los clústeres de AWS Batch, también se crean roles de IAM para los componentes que se incluyen en el proceso de creación de la imagen de Docker cuando se crea el clúster. Estos componentes incluyen las funciones Lambda que tienen permiso para añadir y eliminar imágenes de Docker desde el repositorio de Amazon ECR o en este, y para eliminar el bucket de Amazon S3 que se ha creado para el proyecto de clúster y CodeBuild. También hay roles para los recursos, las instancias y los trabajos de AWS Batch.

Para obtener información detallada acerca de IAM, consulte <https://aws.amazon.com/iam/>.

AWS Lambda

AWS Lambda (Lambda) ejecuta las funciones que organizan la creación de imágenes de Docker. Lambda también se encarga de la limpieza de recursos del clúster personalizado, como las imágenes de Docker almacenadas en el repositorio de Amazon ECR y en Amazon S3. Lambda se usa solo con clústeres de AWS Batch.

Para obtener información detallada, consulte <https://aws.amazon.com/lambda/>.

Amazon Simple Notification Service

Amazon Simple Notification Service (Amazon SNS) se utiliza para recibir notificaciones desde Auto Scaling. Estos eventos se denominan eventos de ciclo de vida y se generan cuando se lanza o termina una instancia en un grupo de Auto Scaling. Dentro de AWS ParallelCluster, el tema de Amazon SNS para el grupo de Auto Scaling está suscrito a una cola de Amazon SQS.

Amazon SNS no se usa con clústeres de AWS Batch.

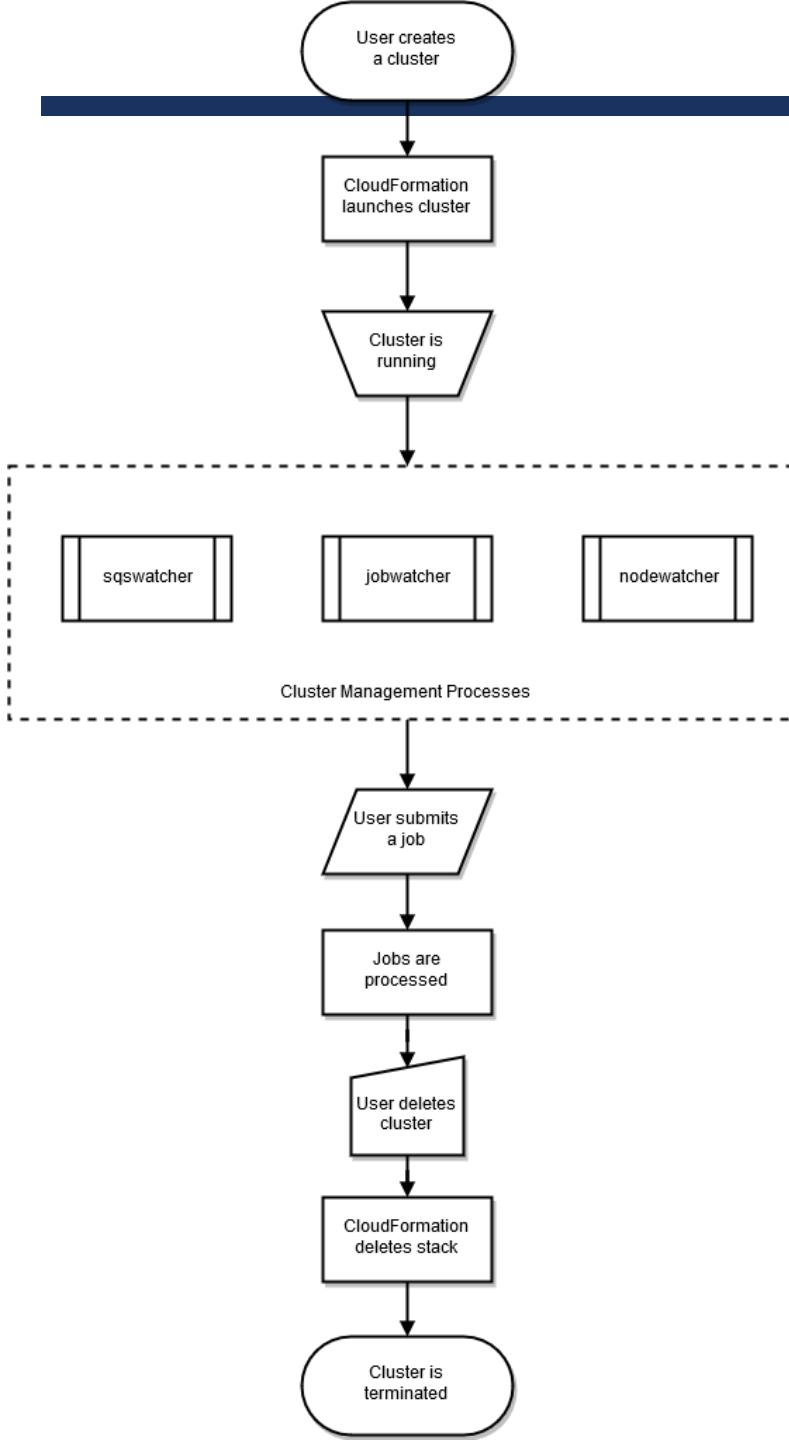
Para obtener información detallada acerca de Amazon SNS, consulte <https://aws.amazon.com/sns/>.

Amazon Simple Queue Service

Amazon Simple Queue Service (Amazon SQS) se utiliza para almacenar notificaciones de Auto Scaling (enviadas a través de Amazon SNS) y notificaciones de las instancias de ComputeFleet. El uso de Amazon SQS desacopla el envío de notificaciones de su recepción y permite al maestro gestionarlas a través del sondeo. MasterServer ejecuta Amazon SQSwatcher y sondea la cola. Auto Scaling y las instancias de ComputeFleet publican mensajes en la cola.

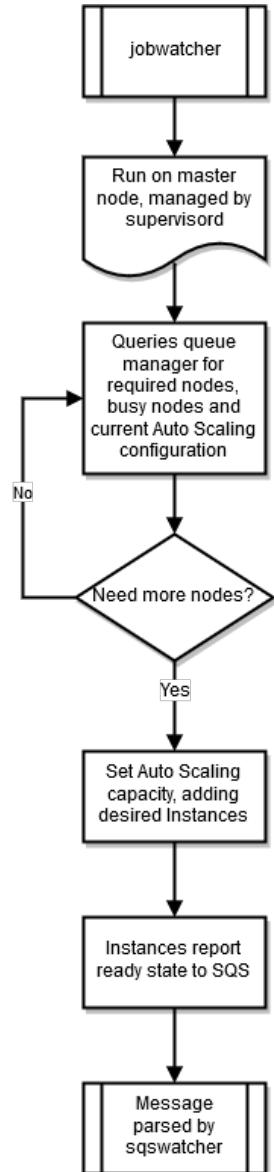
Amazon SQS no se usa con clústeres de AWS Batch.

Para obtener información detallada acerca de Amazon SQS, consulte <https://aws.amazon.com/sqs/>.



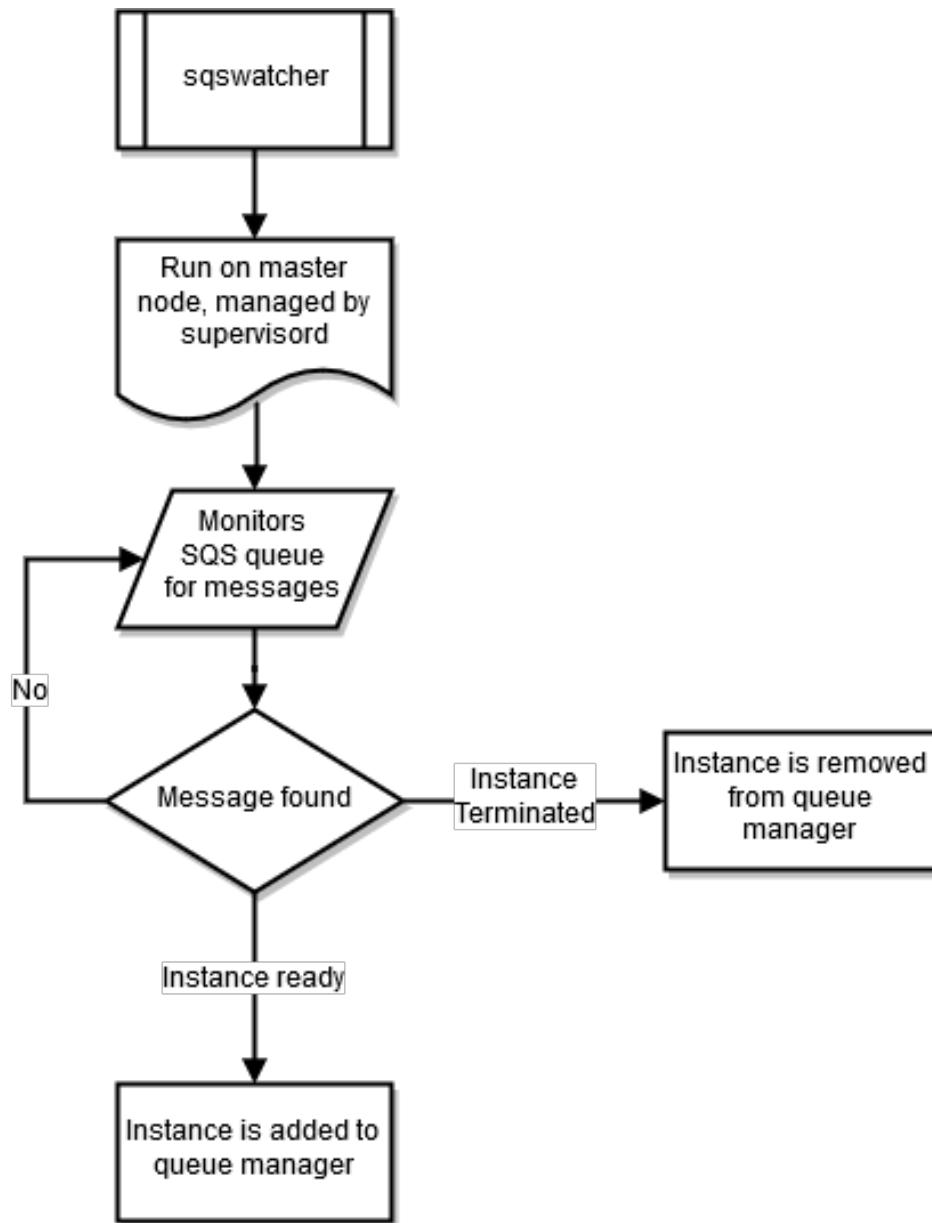
Información general

El ciclo de vida de un clúster comienza después de crearlo el usuario. Normalmente, un clúster se crea a partir de la interfaz de línea de comandos (CLI). Después de crearse, un clúster existe hasta que se elimina. Los demonios de AWS ParallelCluster se ejecutan en los nodos de clúster, principalmente para administrar la elasticidad del clúster HPC. En el siguiente diagrama se muestran un flujo de trabajo de usuario y el ciclo de vida del clúster. En las secciones que aparecen a continuación se describen los demonios de AWS ParallelCluster que se utilizan para administrar el clúster.



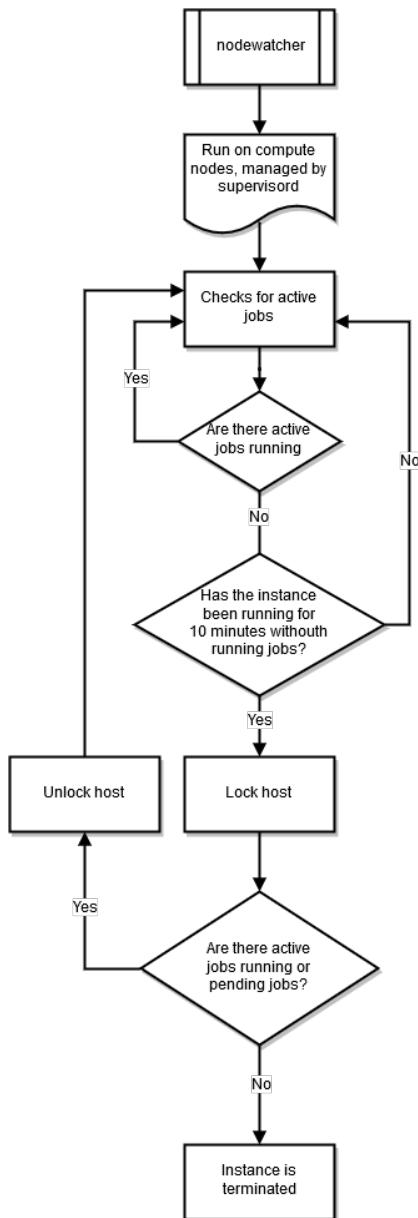
jobwatcher

Al ejecutarse un clúster, un proceso que pertenece al usuario raíz monitoriza el programador configurado (SGE, Slurm o Torque) y cada minuto evalúa la cola para decidir cuándo realizar una ampliación con escalado.



sqswatcher

El proceso sqswatcher monitoriza los mensajes de Amazon SQS que Auto Scaling envía para notificarle los cambios de estado en el clúster. Cuando una instancia está online, envía un mensaje "instancia lista" a Amazon SQS. sqs_watcher recoge este mensaje y se ejecuta en el nodo maestro. Estos mensajes se utilizan para notificar al administrador de la cola que hay instancias nuevas online o que se han terminado instancias, de modo que se puedan añadir o eliminar de la cola.



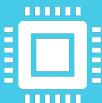
Nodewatcher

El proceso nodewatcher se ejecuta en cada nodo de la flota de computación. Transcurrido el periodo scaledown_idletime, tal como define el usuario, la instancia se termina.

WHAT IS SERVERLESS COMPUTING AND WHY USE IT?



AWS Lambda offers competitive pricing and an event driven computing model. That is, Lambda executes your code in response to events. An event can be a change to an Amazon S3 bucket, an update to an Amazon DynamoDB table, or custom events generated by various applications or devices. Moments after an event trigger, Lambda automatically prepares compute resources and runs Lambda function or code.



It eradicates issues related to unused utilized server capacity without compromising on scalability or speed of response. Startups have moved from monolithic application architecture to microservices driven architecture by using AWS Lambda. Startups like group chat and messaging app SendBird and analytics platform Wavefront are using Lambda. Many startups like Click Travel have moved from monolithic app architecture to microservice-driven architecture (using AWS Lambda).



Lambda eliminates the issues related to unused server capacity without sacrificing scalability or responsiveness. For e.g., developers working on a hotel booking website experiences user escalation on holidays. Usually, developers must create mechanisms to deal with this demand surge. Instead, Lambda takes care of this aspect for you. It supports multiple languages and libraries, including Python and JavaScript.

Serverless applications

Event source Function Services (anything)



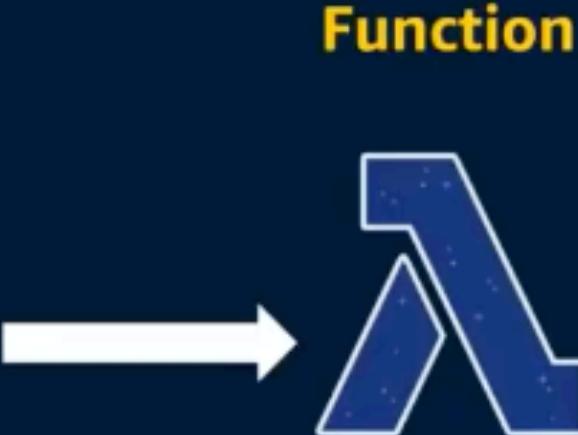
Changes in
data state



Requests to
endpoints



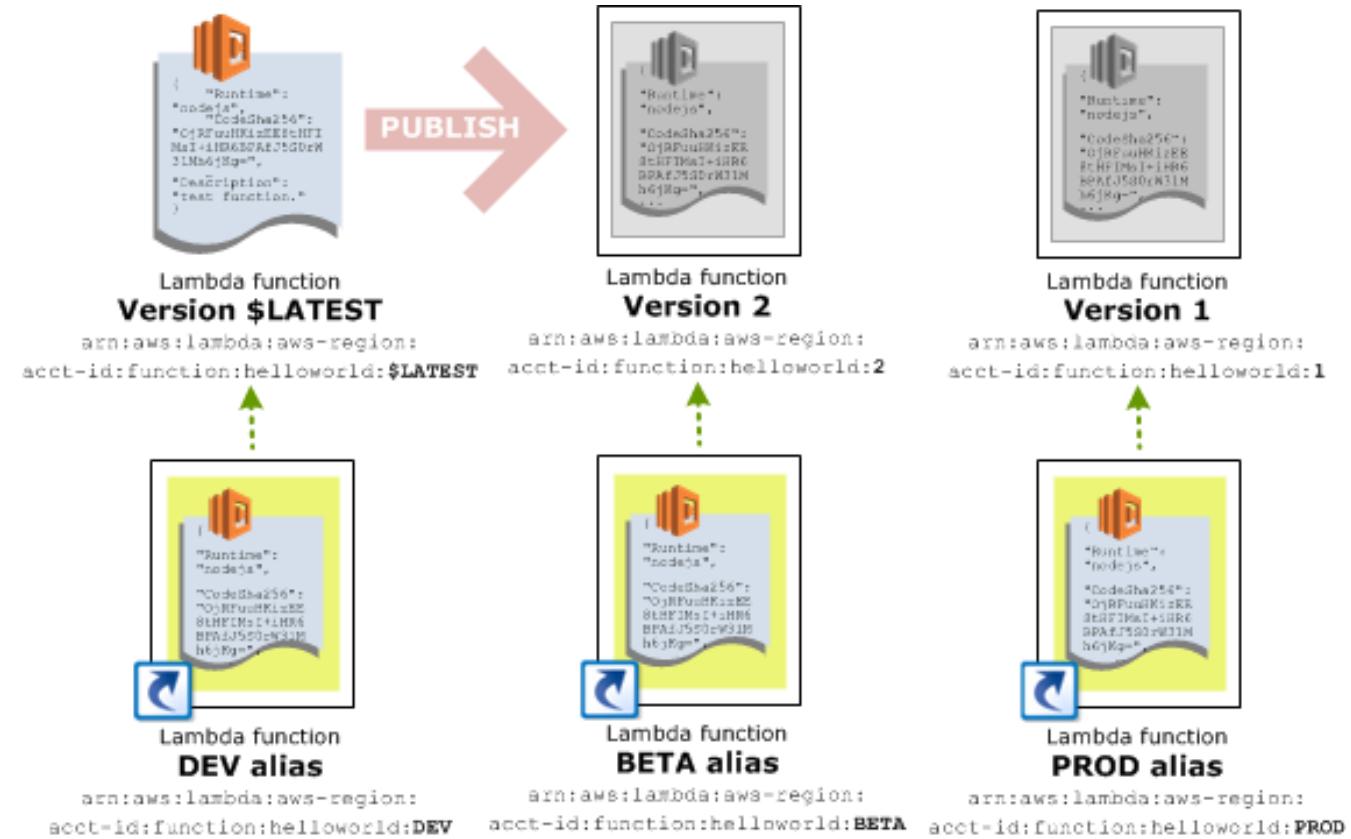
Changes in
resource state



Node.js
Python
Java
C#
Go
Ruby
Runtime API



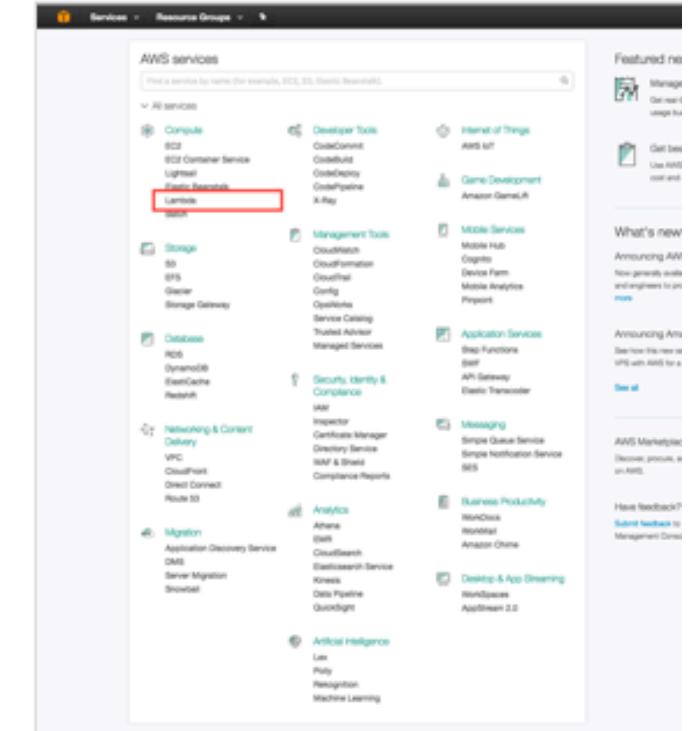
4



- Versions = immutable copies of code + configuration
- Aliases = mutable pointers to versions
- Development against \$LATEST version
- Each version/alias gets its own ARN
- Enables rollbacks, staged promotions, "locked" behavior for client

1: Acceder a la consola de Lambda

- › [haga clic aquí](#), se abrirá la consola de administración de AWS en una ventana del dor nueva, para que pueda seguir teniendo abierta esta guía paso a paso. Encuentre **a** en *Compute* y haga clic para abrir la consola de AWS Lambda.



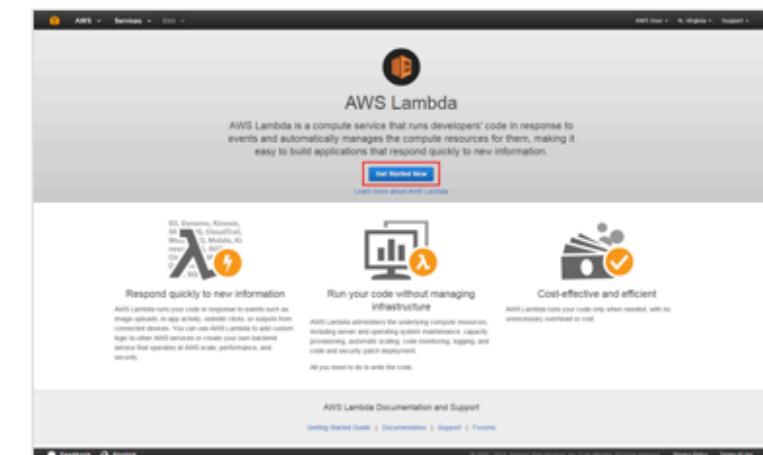
(Haga clic para ampliar).

Paso 2: Seleccionar un plano de Lambda

Los planos proporcionan código de ejemplo para realizar tareas mínimas de procesamiento. La mayoría de los planos procesan eventos de fuentes de eventos específicas, como Amazon S3, DynamoDB o una aplicación personalizada.

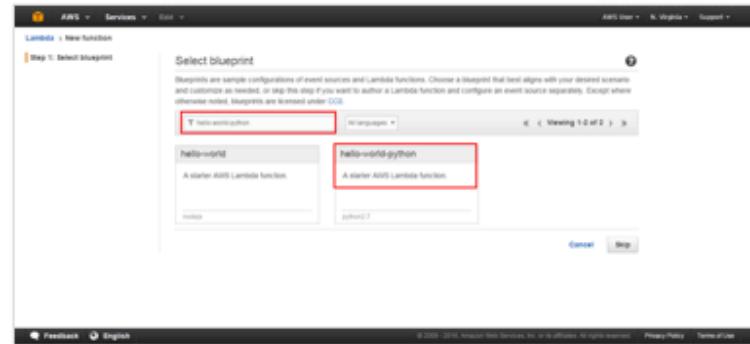
a. En la consola de AWS Lambda, seleccione **Get Started Now**.

Si ya posee funciones de Lambda, seleccione **Create a Lambda function**.



(Haga clic para ampliar).

b. En el recuadro Filter, escriba *hello-world-python* y seleccione el plano hello-world-python.



(Haga clic para ampliar).

proporciona incluye los recursos informáticos que desea asignar (por ejemplo, memoria), límite del tiempo de ejecución y una función de IAM que AWS Lambda puede asumir para ejecutar su función de Lambda por usted.

a. Ahora va a configurar su función de Lambda. La lista que aparece a continuación explica las configuraciones y proporciona valores de ejemplo.

Función Configure:

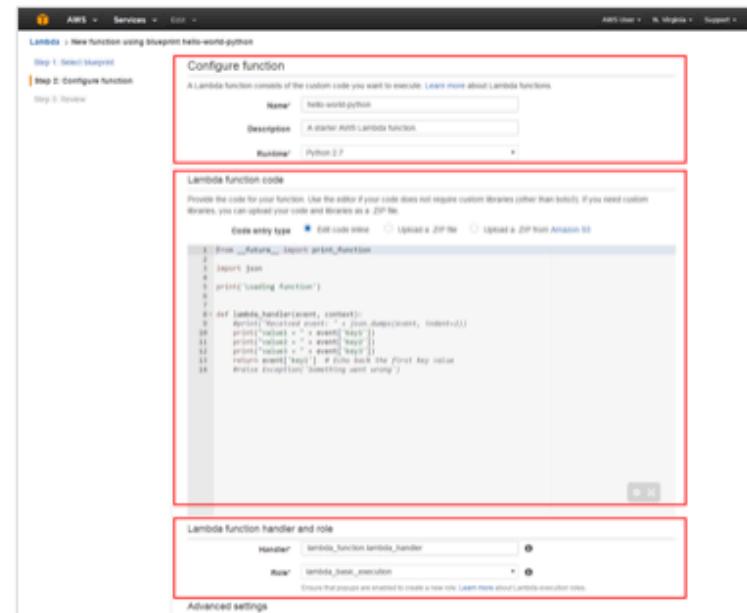
- **Name:** Aquí puede introducir el nombre de su función de Lambda. Para este tutorial, introduzca *hello-world-python*.
- **Description:** Aquí puede introducir una breve descripción de la función. Se rellena automáticamente con *A starter AWS Lambda Function*.
- **Runtime:** En la actualidad, puede escribir el código de su función de Lambda en Java, Node.js o Python 2.7. Para este tutorial, deje *Python 2.7*.

Lambda function code:

- En esta sección, puede examinar el código de ejemplo creado en Python.

Lambda function handler and role:

- **Handler:** Puede especificar un handler (método/función en su código) en el que AWS Lambda puede comenzar a ejecutar su código. AWS Lambda proporciona datos de evento de entrada a este handler, que procesa el evento. En este ejemplo, Lambda lo identifica a partir de la muestra de código, y debería llenarse automáticamente con



(Haga clic para ampliar).

b. Creará una función de IAM (denominada función de ejecución) con los permisos necesarios que AWS Lambda puede asumir para invocar su función de Lambda por usted. Haga clic en **Allow**.

Volverá a la página de la función Configure y se habrá seleccionado *lambda_basic_execution*.



(Haga clic para ampliar).

c. En la sección *Advanced settings*, puede configurar la memoria, límite de tiempo y ajustes de VPC. Para este tutorial, deje los valores predeterminados de la configuración de la función de Lambda y haga clic en **Next**.

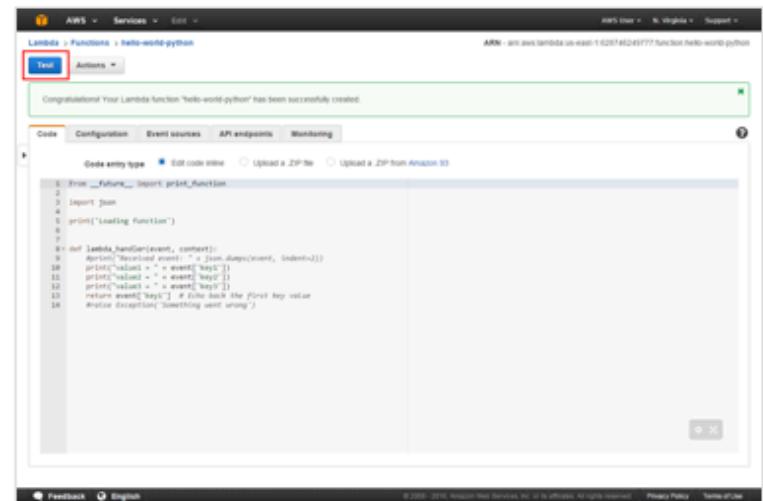


(Haga clic para ampliar).

Paso 4: Invocar la función de Lambda y verificar los resultados

La consola muestra la función de Lambda hello-world-python Lambda. Ahora puede probar la función, verificar los resultados y comprobar los logs.

a. Haga clic en **Test**.

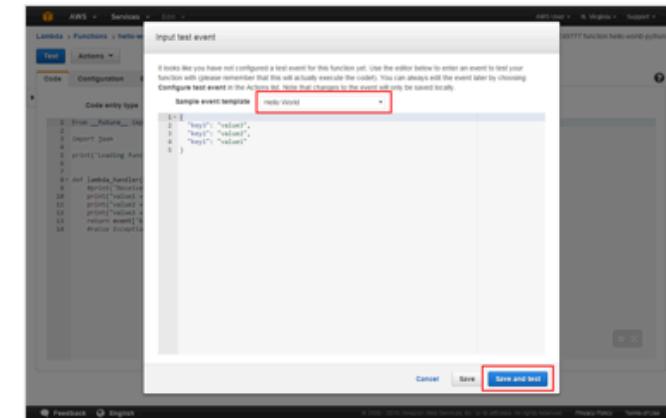


(Haga clic para ampliar).

b. El editor aparece para introducir un evento y probar la función.

- Elija *Hello World* de la lista de plantillas de evento de muestra en la página de pruebas de evento de entrada.
- Puede cambiar los valores en el JSON de muestra, pero no cambie la estructura del evento. Para este tutorial, sustituya *value1* por *hello, world!*

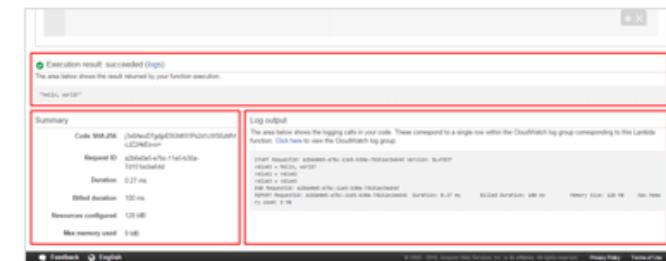
Haga clic en **Save and test**. AWS Lambda ejecutará su función por usted.



(Haga clic para ampliar).

c. Una vez realizada la ejecución con éxito, vea los resultados en la consola:

- La sección **Execution results** verifica que la ejecución ha tenido éxito.
- La sección **Summary** muestra la información clave proporcionada en los resultados del log.
- La sección **Log output** muestra los logs generados por la ejecución de la función de Lambda.

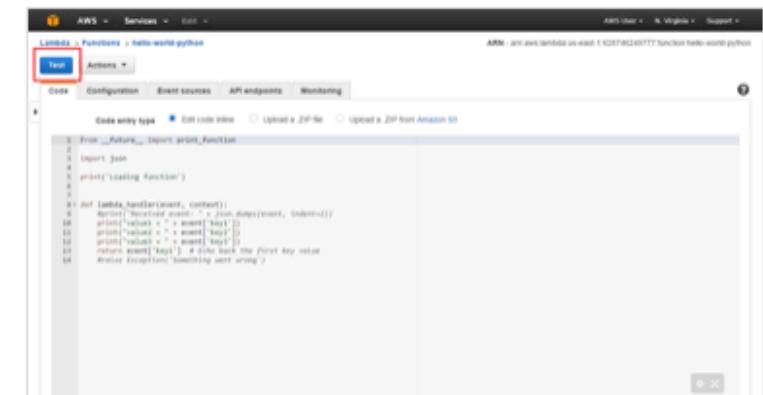


(Haga clic para ampliar).

Paso 5: Monitorizar las métricas

AWS Lambda monitoriza de forma automática las funciones de Lambda y proporciona informes de métricas a través de Amazon CloudWatch. Para ayudarle a monitorizar el código que ejecuta, Lambda controla automáticamente la cantidad de solicitudes, la latencia por solicitud y el número de solicitudes que han resultado en un error, y publica las métricas asociadas.

- Invoque la función de Lambda unas cuantas veces haciendo clic repetidamente en el botón **Test**. Se generarán las métricas que se pueden ver en el siguiente paso.



(Haga clic para ampliar).

Seleccione la pestaña **Monitoring** para ver las métricas de la función de Lambda. Las métricas Lambda se proporcionan en informes a través de Amazon CloudWatch. Puede utilizarlas para configurar alarmas personalizadas. Para obtener más información sobre CloudWatch, consulte [Guía para desarrolladores de Amazon CloudWatch](#).

La pestaña Monitoring muestra cuatro métricas de CloudWatch: *recuento de invocaciones, duración de las invocaciones, errores de invocación e invocaciones limitadas*.

En AWS Lambda, paga por lo que utiliza. Una vez que supera el límite de la capa gratuita de AWS Lambda, se le cobra en función de la cantidad de solicitudes de sus funciones (recuento de invocaciones) y el tiempo durante el que se ejecuta su código (duración de las invocaciones). Para tener más información, consulte los [precios de AWS Lambda](#).



(Haga clic para ampliar).

AWS STEP FUNCTIONS

- AWS Step Functions permite coordinar múltiples servicios de AWS en flujos de trabajo sin servidor para poder crear y actualizar aplicaciones rápidamente. Mediante Step Functions, puede diseñar y ejecutar flujos de trabajo que unen servicios como AWS Lambda y Amazon ECS en aplicaciones con muchas características
- Los flujos de trabajo se componen de una serie de pasos, con la salida de un paso que actúa como entrada en el siguiente. El desarrollo de aplicaciones es más simple y más intuitivo usando Step Functions, ya que convierte su flujo de trabajo en un diagrama de máquina de estado fácil de entender, fácil de explicar a otros y fácil de cambiar.
- Puede monitorizar cada paso de la ejecución tal como sucede, lo que significa que puede identificar y solucionar problemas rápidamente. Step Functions activa y monitoriza cada paso de manera automática; además, realiza reintentos cuando se producen errores, por lo que su aplicación se ejecuta en orden y según lo previsto.

AWS STEP FUNCTIONS

- AWS Step Functions permite coordinar múltiples servicios de AWS en flujos de trabajo sin servidor para poder crear y actualizar aplicaciones rápidamente. Mediante Step Functions, puede diseñar y ejecutar flujos de trabajo que unen servicios como AWS Lambda y Amazon ECS en aplicaciones con muchas características
- Los flujos de trabajo se componen de una serie de pasos, con la salida de un paso que actúa como entrada en el siguiente. El desarrollo de aplicaciones es más simple y más intuitivo usando Step Functions, ya que convierte su flujo de trabajo en un diagrama de máquina de estado fácil de entender, fácil de explicar a otros y fácil de cambiar.
- Puede monitorizar cada paso de la ejecución tal como sucede, lo que significa que puede identificar y solucionar problemas rápidamente. Step Functions activa y monitoriza cada paso de manera automática; además, realiza reintentos cuando se producen errores, por lo que su aplicación se ejecuta en orden y según lo previsto.
- LAB https://aws.amazon.com/es/getting-started/tutorials/create-a-serverless-workflow-step-functions-lambda/?nc1=h_ls

Benefits and features

Built-in error handling

AWS Step Functions tracks the state of each step, so you can automatically retry failed or timed-out tasks, catch specific errors, and recover gracefully, whether the task takes seconds or months to complete.

Automatic scaling

AWS Step Functions automatically scales the operations and underlying compute to run the steps of your application for you in response to changing workloads. Step Functions scales automatically to help ensure the performance of your application workflow remains consistently high as the frequency of requests increases.

Pay per use

With AWS Step Functions, you pay only for the transition from one step of your application workflow to the next, called a state transition. Billing is metered by state transition, regardless of how long each state persists (up to one year).

Execution event history

AWS Step Functions creates a detailed event log for every execution, so when things do go wrong, you can quickly identify not only where, but why. All of the execution history is available visually and programmatically to quickly troubleshoot and remediate failures.

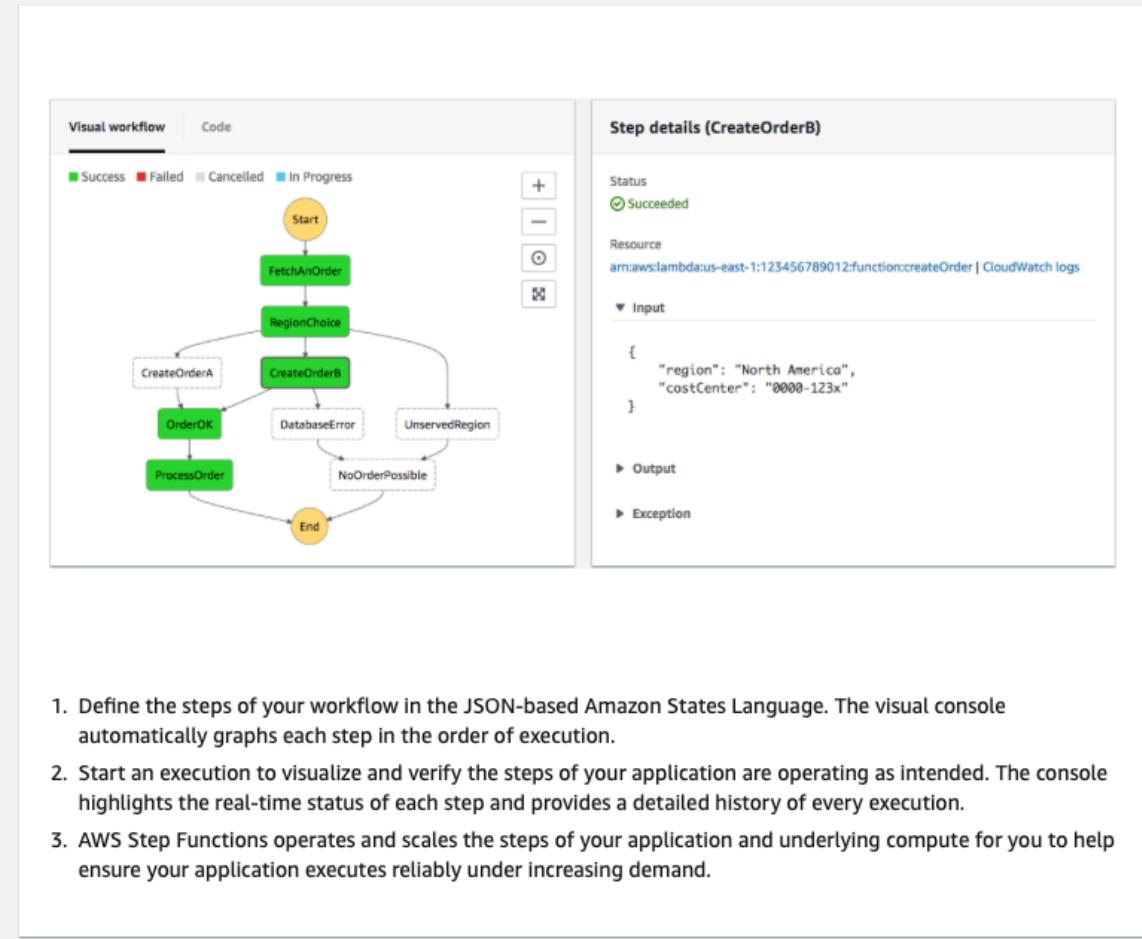
High availability

AWS Step Functions has built-in fault tolerance. Step Functions maintains service capacity across multiple Availability Zones in each region to help protect application workflows against individual machine or data center facility failures. There are no maintenance windows or scheduled downtimes.

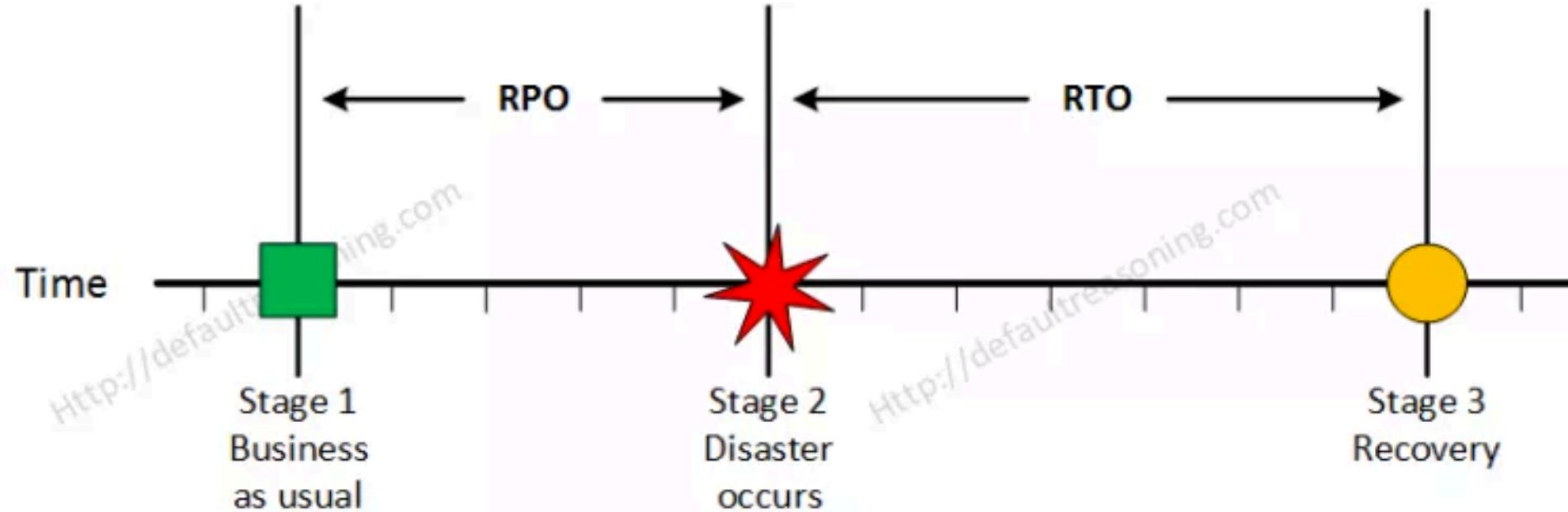
Administrative security

AWS Step Functions is integrated with AWS Identity and Access Management (IAM). IAM policies can be used to control access to the Step Functions APIs.

How it works

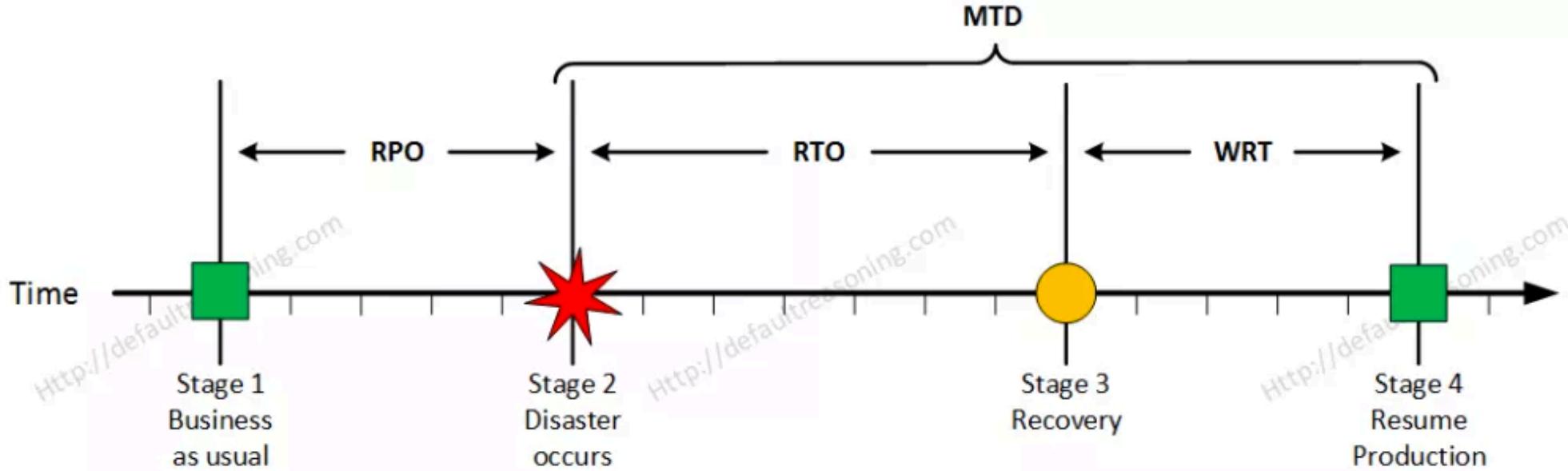


Domain 3: High Availability and Business Continuity



On a given point in time, disaster occurs and systems needs to be recovered. At this point the **Recovery Point Objective (RPO)** determines the maximum acceptable amount of data loss measured in time. For example, the maximum tolerable data loss is 15 minutes.

At this stage the system are recovered and back online but not ready for production yet. The **Recovery Time Objective (RTO)** determines the maximum tolerable amount of time needed to bring all critical systems back online. This covers, for example, restore data from back-up or fix of a failure. In most cases this part is carried out by system administrator, network administrator, storage administrator etc.

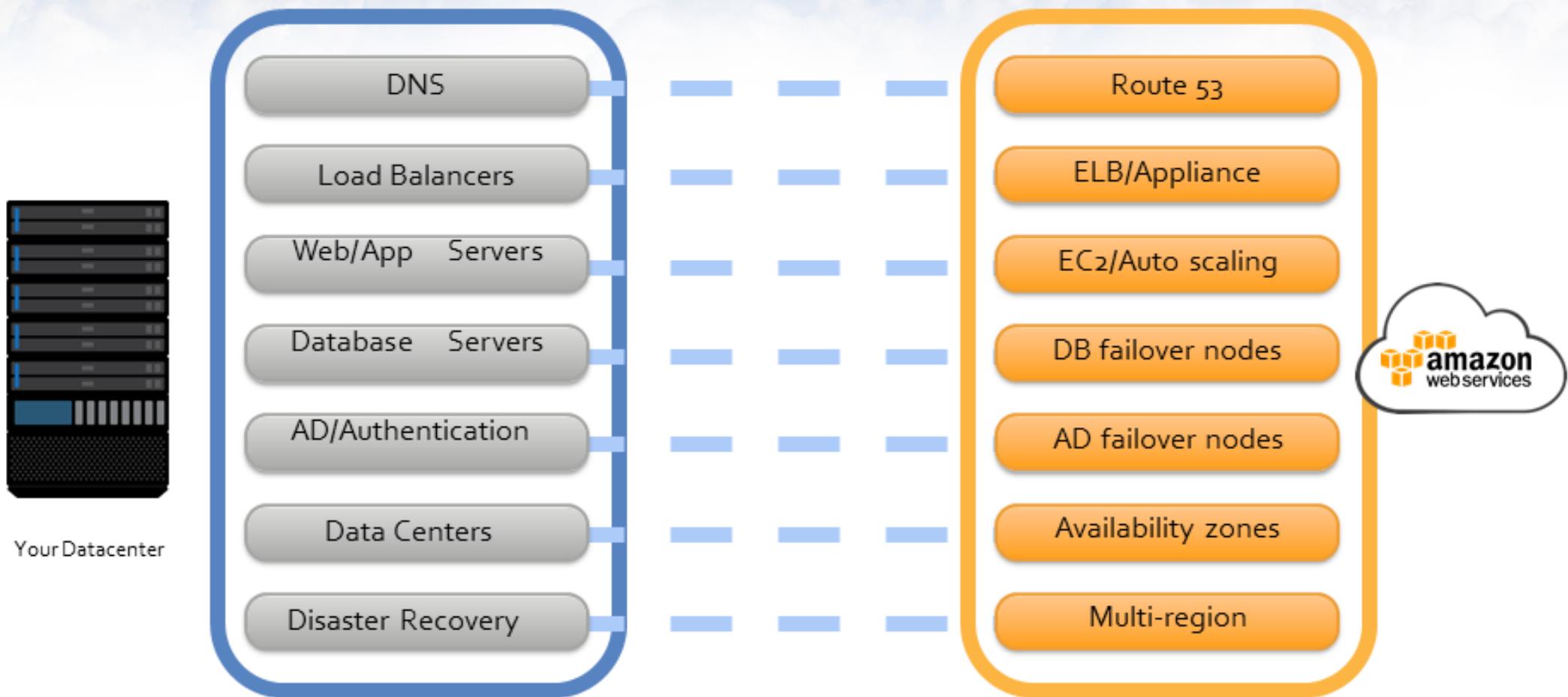


At this stage all systems are recovered, integrity of the system or data is verified and all critical systems can resume normal operations. The **Work Recovery Time (WRT)** determines the maximum tolerable amount of time that is needed to verify the system and/or data integrity. This could be, for example, checking the databases and logs, making sure the applications or services are running and are available. In most cases those tasks are performed by application administrator, database administrator etc. When all systems affected by the disaster are verified and/or recovered, the environment is ready to resume the production again.

The sum of RTO and WRT is defined as the **Maximum Tolerable Downtime (MTD)** which defines the total amount of time that a business process can be disrupted without causing any unacceptable consequences. This value should be defined by the business management team or someone like CTO, CIO or IT manager.

This is of course a simple example of a Business Continuity/Disaster Recovery plan and should be included in your **Business Impact Analysis (BIA)**.

DR Services Mapping



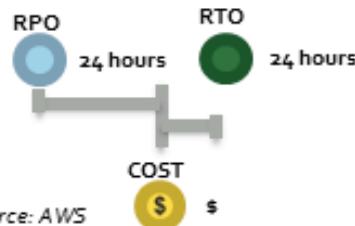
Backup & Restore	Pilot Light	Warm Standby	Multi Site
RTO/RPO: Hours	RTO/RPO: 10s of Minutes	RTO/RPO: Minutes	RTO/RPO: Real-Time
<ul style="list-style-type: none"> • Lower priority use cases • Solutions: Cloud Storage, Backup Solutions • Cost: \$ to \$\$ 	<ul style="list-style-type: none"> • Lower RTO/RPO requirements • Solutions: Database Service, Replication Solutions • Cost: \$\$ 	<ul style="list-style-type: none"> • Core Applications and Services • Solutions: Cloud Storage, Database Service, Replication Solutions • Cost: \$\$\$ 	<ul style="list-style-type: none"> • Mission Critical Applications and Services • Solutions: Database Service, Replication Solutions • Cost:\$\$\$\$

OPTIONS FOR DISASTER RECOVERY IN THE CLOUD

DR Architectures

Backup & Restore

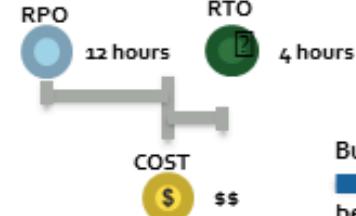
Backup of on-premises data to AWS to use in a DR event



Source: AWS

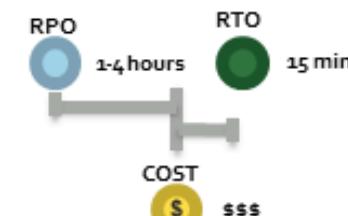
Pilot Light

Replicate data and minimal running services into AWS, ready to take over and flare up



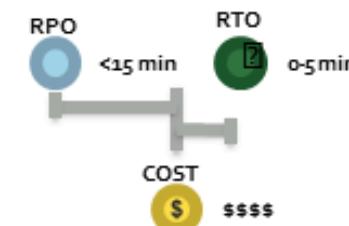
Hot Standby

Replicate data and services into AWS ready to take over



Multi-Site

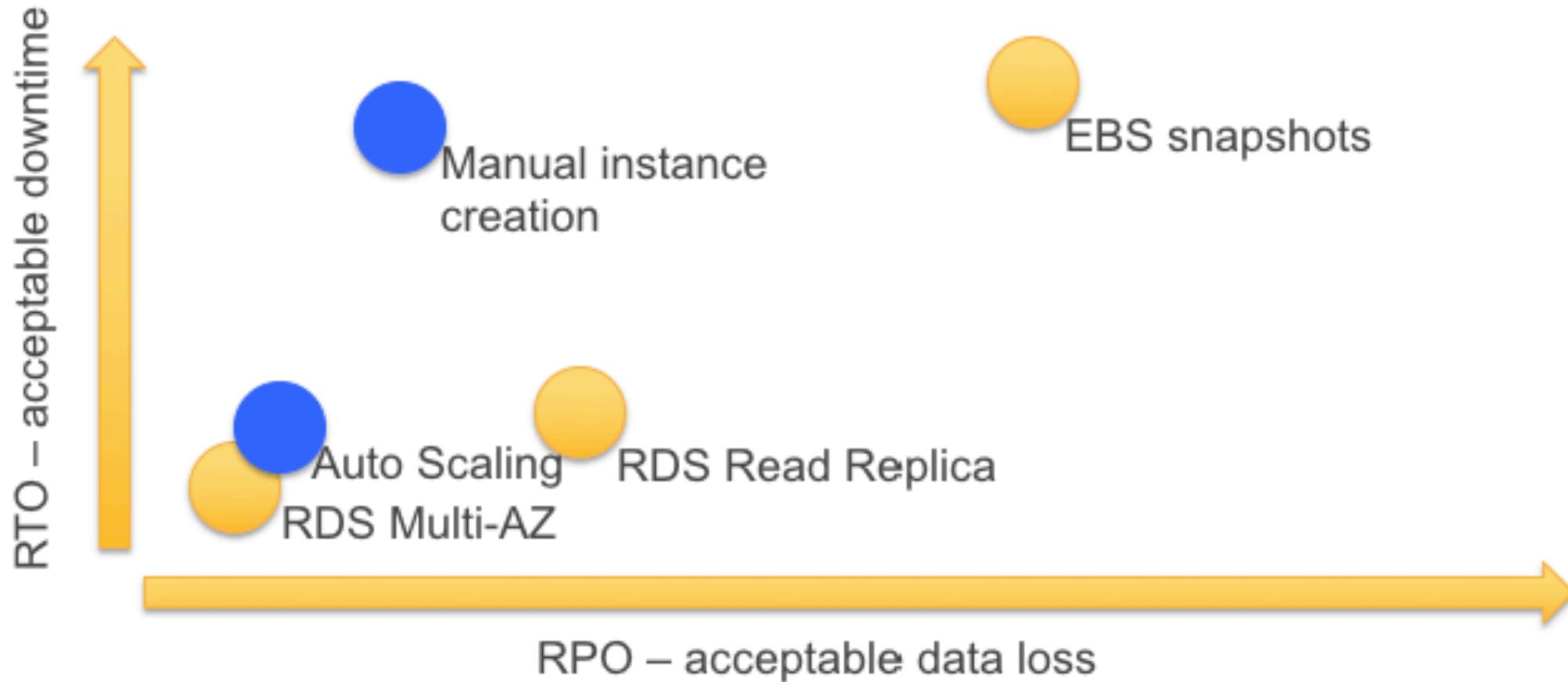
Replicated and load balanced environments that are both actively taking production traffic

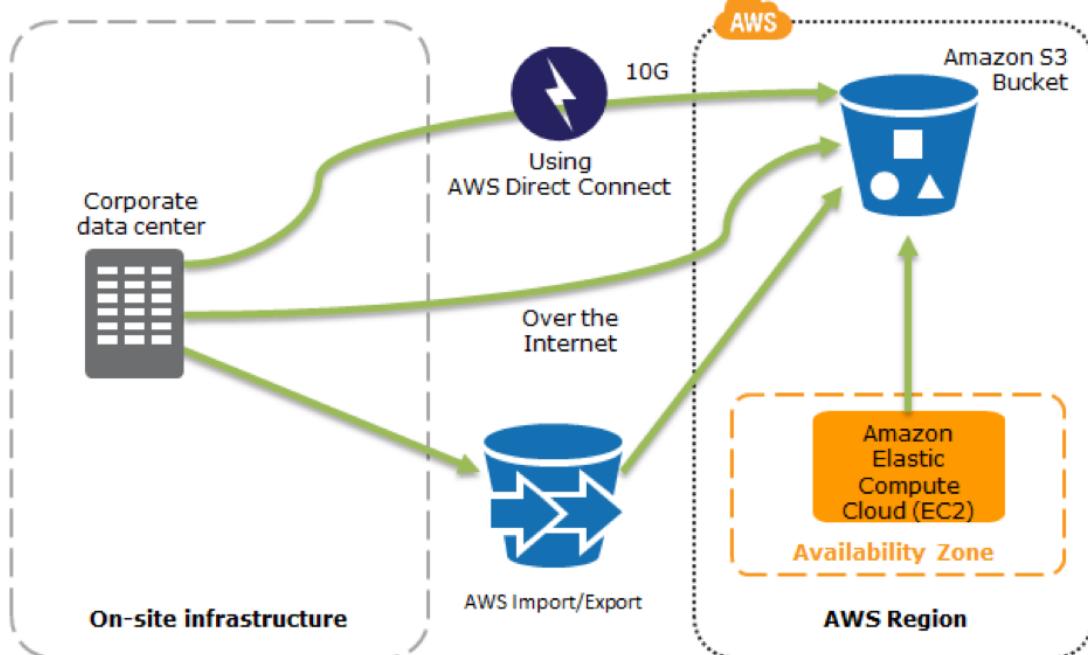


Business continuity
begins

Un-interrupted Business
continuity

Implement DR for Systems based on RPO and RTO (Cont'd.)



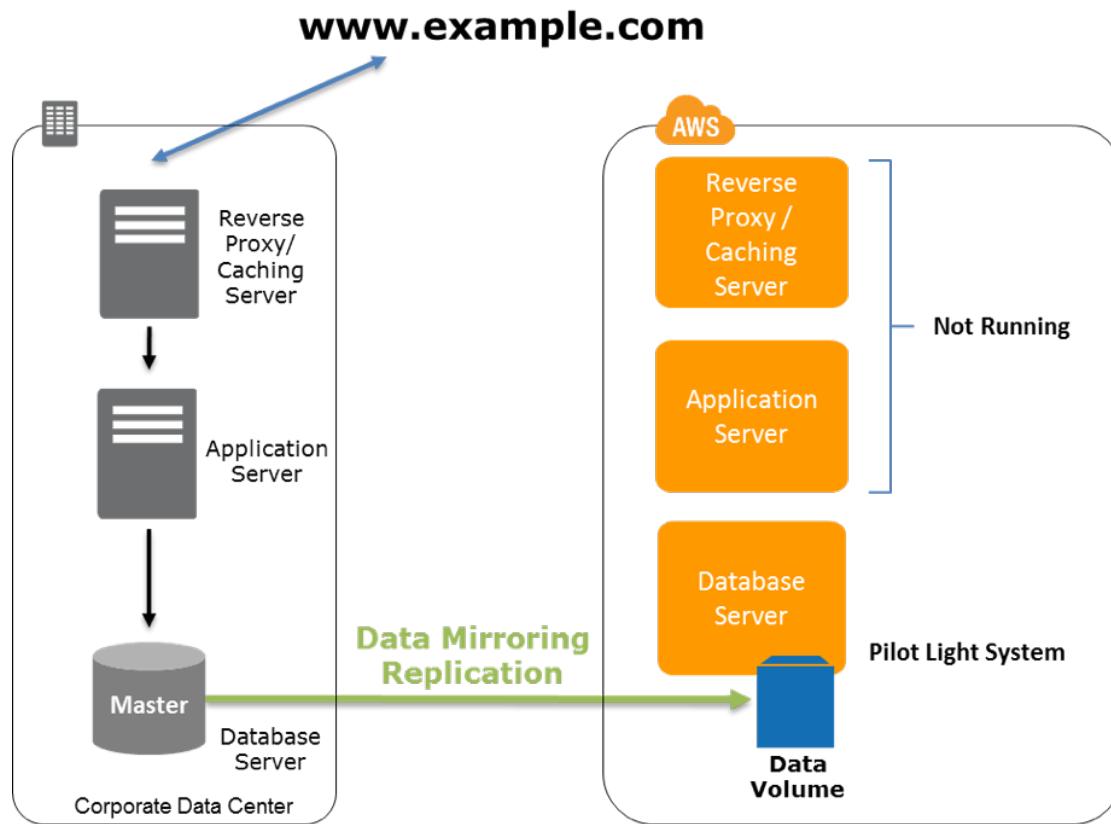


Backup and Restore

Traditionally, companies have used off-site backup tapes as their primary means for restoring data in the event of a disaster. This typically involved retrieving tapes from cold storage and recovering data when the primary facility has been restored or when the tapes have been sent to a cold secondary site only turned on when a disaster has occurred.

Companies have started to leverage public cloud storage services such as Amazon Simple Storage Service (S3) and Azure Blob Storage as alternatives to archiving tape to an off-site facility. Not only is this a more cost-effective solution than tape, it delivers better RTO and RPO since the data is already in the cloud where it can be used to launch a DR site on-demand.

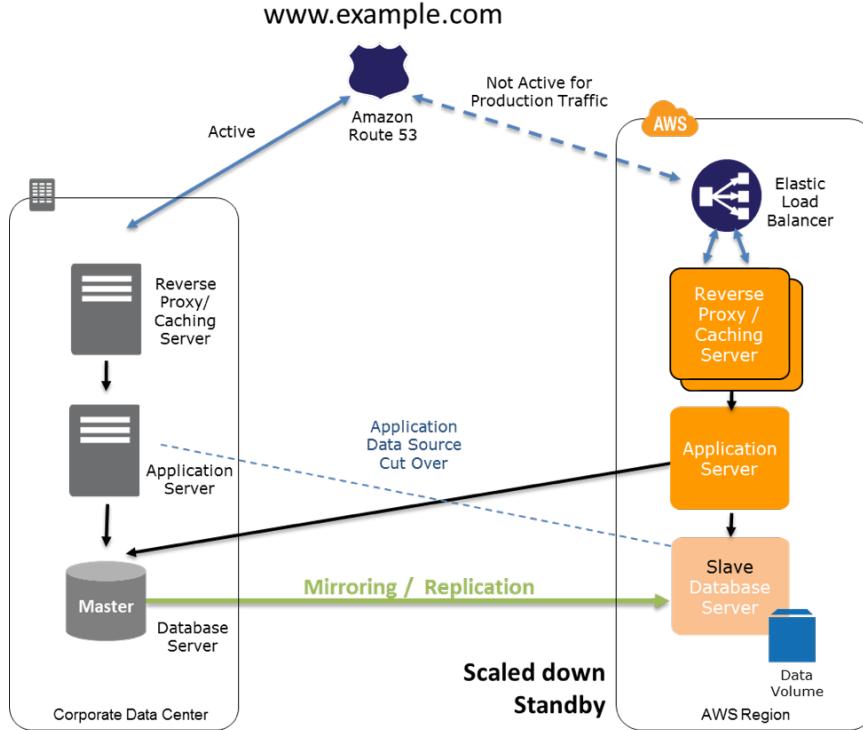
There are various approaches for transferring data from the user's on-premises infrastructure to the public cloud. These include migration tools specific to a particular cloud vendor, as well as third party migration and backup and restore tools. When a disaster is declared, new instances/virtual machines (VM) can be launched using machine images created from on-premises production servers. If needed, application data is restored from object storage. If any application exists that needs very low RPO and RTO, a replication solution may need to be used in conjunction with the backup and restore option. This option can be implemented for the lowest cost at the expense of requiring the longest RTO and RPO.



Pilot Light

The Pilot Light option is named after the constantly-on gas heater pilot light that is used to quickly light the furnace. With this approach, a minimal copy of the production environment is maintained in the cloud. Core components whose state must be maintained and updated, such as a production database, run continuously in the cloud and are synced regularly with production. Servers in the cloud can be provisioned but turned off until a disaster is declared. Alternatively server images can be maintained for launching instances/VMs when needed.

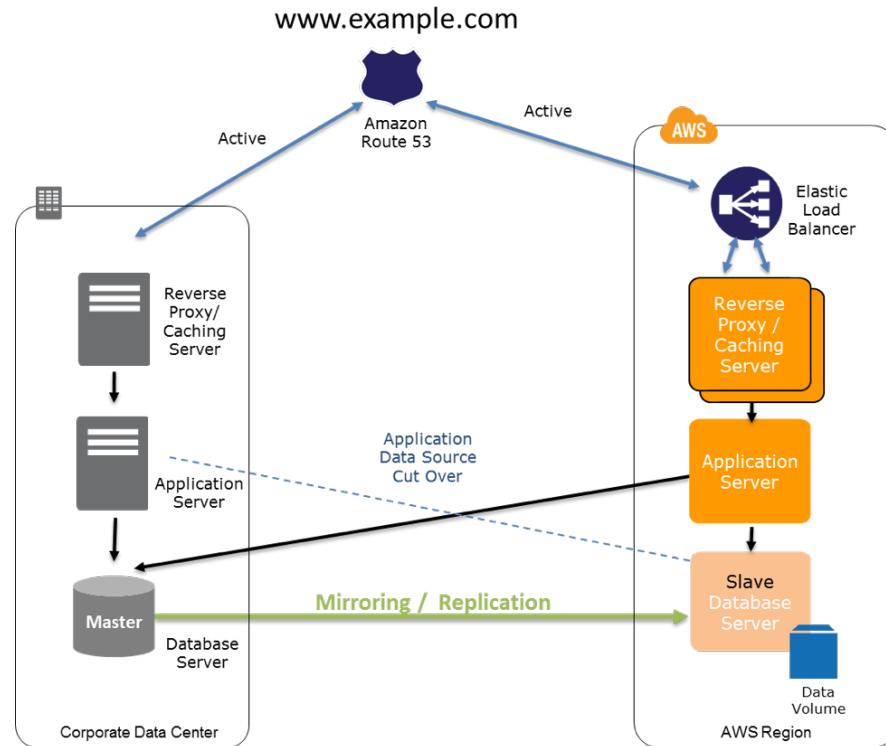
Compared to the Backup and Restore option, the Pilot Light scenario offers a better RTO since the core components are already running in the cloud and servers are already provisioned or ready to be provisioned. It also offers better RPO since core services are regularly updated and synced with production. However, the cost will be higher.



Warm Standby

The Warm Standby option requires a scaled down copy of production to be provisioned and run continuously in the cloud. Stateful core components are also updated and synced regularly with production. A subset of servers, found in production, run continuously as instances/VMs in the cloud and can be scaled up as needed.

Compared to the previous two options, the Warm Standby scenario offers a better RTO since the core components are already running in the cloud and critical servers are already provisioned and running. In a disaster, production traffic for critical workloads can be redirected to the cloud while additional instances/VMs are launched to take on additional workloads. The Warm Standby option also offers better RPO since core services are being regularly updated and synced with production. The cost is higher than the earlier two options since more resources are provisioned and continuously running.



Hot Site

Similar to the Warm Standby option, a copy of the production environment runs continuously in the cloud. But in the hot site scenario, a copy of the full production environment runs in the cloud. This allows for immediate failover during a disaster, with the cloud provisioned to run the same amount of workload as production. In addition, if core components are being updated synchronously, then the cloud can be used for production, along with the user's on-premises infrastructure, in an active-active setup.

This option has the best RTO and RPO since the user is running an exact replica of the on-premises infrastructure in the cloud. As expected, it also has the highest cost, particularly if core components for both the on-premises and cloud environments are being completely synced.

Different features of Amazon RDS support different RTOs and RPOs at different cost points:

Feature	RTO	RPO	Cost	Scope
Automated backups	Good	Better	Low	Single Region
Manual snapshots	Better	Good	Medium	Cross-Region
Read replicas	Best	Best	High	Cross-Region

Amazon RDS backups

Backups are a key component of a DR plan for your database. Amazon RDS supports two different types of backups: automated backups, and manual snapshots.

Amazon RDS backups follow these rules:

- Your DB instance must be in the ACTIVE state for backups to occur.
- Automated backups and automated snapshots do not occur while a copy is executing in the same Region for the same DB instance.
- The first snapshot of a DB instance contains the data of the full DB instance.
- The snapshots taken after the first snapshot are incremental snapshots. This means that only the latest changed data is captured and saved.
- If it's a Multi-AZ configuration, backups occur on the standby to reduce impact on the primary.

Note: Automated backups and manual snapshots are stored in an S3 bucket that is owned and managed by the Amazon RDS service. Hence, you are not able to see them from your Amazon S3 console.

For detailed information on backup mechanisms and backup storage, see [Working with Backups](#) in the *Amazon RDS User Guide*.

Automated backups

The automated backup feature of Amazon RDS is turned on by default. Amazon RDS creates a storage volume snapshot of your DB instance, backing up the entire DB instance and not just individual databases. The first backup consists of a full instance backup. Subsequent backups are incremental in nature with snapshots containing only the blocks that changed since the previous backup. Each snapshot contains pointers to all of the snapshot data blocks that are required to reconstruct it. Deleting the earlier snapshot does not cause data loss as long as the data is still referenced by at least one snapshot.

Why do we need automated backups?

There are several benefits of having automated backups:

- Data is stored in a S3 bucket that is owned and managed by Amazon RDS service.
- You avoid the pressure of having to set aside time to do a manual backup and transfer it to a safe location.
- You can choose a timeline that works for you: daily, weekly, or monthly.
- Both manual and natural calamities are mitigated (for example, viruses, software malfunctions, or power outages).
- Most importantly, you avoid the adverse effects of losing valuable data.

Automated backup window

The automated backup window is a weekly time period used for creating automated backups. The window is selected at random from an 8-hour block of time for each AWS Region. However, I strongly suggest that you set the backup window during low peak hours to prevent undue load on the server. For a list of the time blocks for each Region, see [Backup Window](#) in the *Amazon RDS User Guide*.

Automated backup retention period

The backup retention period is the time window during which you can perform a point-in-time restore (PITR). You can set a different backup retention period when you create a DB instance, and you can modify the retention period at any time.

For more detailed information, see [Backup Retention Period](#).

There are a few differences between manual snapshots and automated backups:

- Manual snapshot limits (100 snapshots per Region) do not apply to automated backups
- The backup retention period does not apply for manual snapshots
- Manual snapshots are not automatically deleted; they must be explicitly deleted.

Retaining automated backups

When you delete a DB instance, you can choose to retain its automated backups. This can be useful if you later decide to restore the DB instance. Retained automated backups contain automated snapshots and transaction logs from a DB instance. They also include your DB instance properties (such as allocated storage and DB instance class), which are required to restore it to an active instance. You can restore or remove retained automated backups using the AWS Management Console, the Amazon RDS API, and the AWS CLI. See [Retaining Automated Backups](#) in the *Amazon RDS User Guide* for more information on limitations and recommendations for retaining automated backups.

Restoring to a specified point in time

Point-in-time recovery (PITR) is the process of restoring a database to the state it was in at a specified date and time. When automated backups are turned on for your DB instance, Amazon RDS automatically performs a full daily snapshot of your data. The snapshot occurs during your preferred backup window. It also captures transaction logs to Amazon S3 every 5 minutes (as updates to your DB instance are made). Archiving the transaction logs is an important part of your DR process and PITR. When you initiate a point-in-time recovery, transactional logs are applied to the most appropriate daily backup in order to restore your DB instance to the specific requested time.

Database snapshots

Database snapshots are manual (user-initiated) backups of your complete DB instance that serve as full backups. They're stored in Amazon S3, and are retained until you explicitly delete them. These snapshots can be copied and shared to different Regions and accounts. Because DB snapshots include the entire DB instance, including data files and temporary files, the size of the instance affects the amount of time it takes to create the snapshot.

Creating a DB snapshot on a Single-AZ DB instance leads to a brief I/O suspension. The I/O suspension can last a few seconds or minutes depending on the instance size and class of your DB instance. Multi-AZ DB instances are not affected by the I/O suspension because the backup is taken from the standby.

Viewing snapshots

You can view snapshots in the Amazon RDS console. Click Snapshots, and then choose Manual Snapshots. Then choose a snapshot from the list.

The screenshot shows the Amazon RDS console with the 'Schemas' section selected. On the left, a sidebar lists various services: Dashboard, Databases, Query Editor, Performance Insights, Snapshots (which is highlighted in orange), Automated backups, Reserved instances, and Subnet groups. The main content area is titled 'Schemas (8)' and shows a table of database schemas. The table has columns for Name, DB instance or cluster, Status, and Last modified. A dropdown menu is open over the 'Actions' column, showing options: 'Manual Snapshots', 'Automated Snapshots', 'Shared with Me', and 'All Public Snapshots'. The 'Manual Snapshots' option is currently selected. The table data is as follows:

Name	DB instance or cluster	Status	Last modified
california	hacker	available	Fri Dec 28 12:20:14 GMT+530 2018
anuraag-encryption	postgres	available	Fri Dec 28 12:20:14 GMT+530 2018
test-snapshot	postgres	available	Thu Dec 27 15:30:28 GMT+530 2018

Copying and sharing snapshots

In Amazon RDS, you can copy automated or manual DB snapshots. When you create a copy of a snapshot, that copy becomes a manual snapshot. You can copy a snapshot within the same AWS Region or across AWS Regions, and you can even copy a snapshot across AWS accounts. If you copy a DB snapshot to another AWS Region, you create a manual DB snapshot that is retained in that AWS Region. Copying a DB snapshot out of the source AWS Region incurs Amazon RDS data transfer charges.

You cannot directly copy an automated snapshot to another AWS account. Instead, it is a two-step process, where you first share the snapshot, and then copy it in the other account.

Amazon RDS cross-account snapshot sharing

Amazon RDS enables you to share DB snapshots or cluster snapshots with other AWS accounts. Sharing snapshots with other highly secure accounts can be helpful if you are concerned about a “bad actor” disrupting operations in your production accounts. You can share manual DB snapshots with up to 20 AWS accounts.

- Automated Amazon RDS snapshots cannot be shared directly with other AWS accounts. To share an automated snapshot, you first make a copy of the snapshot, which turns it into a manual version. Then you share the copy with the other account.
- Manual snapshots of DB instances that use custom option groups with persistent or permanent options, such as Transparent Data Encryption (TDE) and time zone, cannot be shared.
- Snapshots that use the default Amazon RDS encryption key (aws/rds) cannot be shared directly. Instead, you first copy the snapshot by choosing a custom encryption key, and then you share the custom key and the copied snapshot.

Restoring from a DB snapshot

If a disaster occurs, you can create a new DB instance by restoring from a DB snapshot. When you restore the DB instance, you choose the name of the DB snapshot from which you want to restore. Then, you provide a name for the new DB instance that is created. Here are a few things to note about the restoration process:

- You cannot restore from a DB snapshot to an existing DB instance. Instead, you create a new DB instance when you restore. If you want to use the same name as the existing DB instance, you must first delete or rename the existing one.
- While it's possible to restore a DB snapshot to a DB instance with a different storage type than the source DB instance, the restoration process is slower. There is additional work required to migrate the data to a new storage type.
- You can't restore a DB instance from a shared DB snapshot that is encrypted. Instead, you make a copy of the DB snapshot and then restore the DB instance from the copy.
- It's a good practice to retain the parameter group of any DB snapshots that you create. This enables you to restore the DB instance with the correct parameter group.
- When you restore from a DB snapshot, by default the option group that is associated with the DB snapshot is associated with the restored DB instance. You can associate a different option group with a restored DB instance. However, the new option group must contain any persistent or permanent options that were included in the original option group.

Cross-Region restore

As discussed, when you perform a cross-Region restore of a DB snapshot, first you copy the snapshot to the desired Region. Then, you can restore the DB snapshot to a new DB instance.

Integrating with AWS Backup

Amazon RDS DB snapshots can be integrated with AWS Backup. AWS Backup is a fully managed backup service that you can use to centralize and automate the backup of data across AWS services in the cloud and on premises. Using AWS Backup, you can centrally configure backup policies and monitor backup activity for your AWS resources. For more information, see [AWS Backup](#).

Read Replicas

Amazon RDS for MariaDB, MySQL, PostgreSQL, and Oracle support the ability to create Read Replicas of a source database. When you create a Read Replica, Amazon RDS first takes a snapshot of the source DB instance, and then creates a read-only instance. Amazon RDS then uses the asynchronous replication method of the DB engine to update the Read Replica whenever there is a change made on the source DB instance. The Read Replica operates as a DB instance that allows only read-only connections. Applications can connect to a Read Replica the same way they do to any DB instance. Amazon RDS replicates all objects in the source DB instance. By default, a Read Replica is created with the same instance and storage type as the source DB instance. However, you can create a Read Replica that has a different storage type from the source DB instance. You can create up to five Read Replicas per source DB instance.

In addition to using Read Replicas to reduce the load on your source DB instance, you can also use Read Replicas to implement a DR solution for your production DB environment. If the source DB instance fails, you can promote your Read Replica to a standalone source server. Read Replicas can also be created in a different Region than the source database. Using a cross-Region Read Replica can help ensure that you get back up and running if you experience a regional availability issue.

An important metric to monitor with a Read Replica is the replica lag, which is the amount of time that the replica is behind the source database. A replica lag can impact your recovery. Replica lag can vary based on the network latency between the source and destination Regions. It can also be affected by the amount of traffic that is being replicated. Because Read Replicas have a running DB instance, the time required to recover after a disaster is lower. However, using Read Replicas in this way is generally a more expensive option than using automated backups or database snapshots.

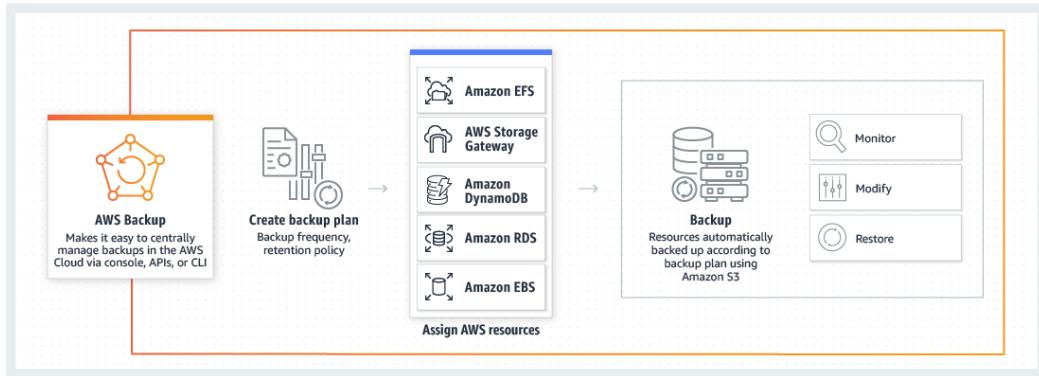
Promoting a Read Replica

Unlike an Amazon RDS Multi-AZ configuration, failover to a Read Replica is not an automated process. If you are using cross-Region Read Replicas, you should be certain that you want to switch your AWS resources between Regions. Cross-Region traffic can experience latency, and reconfiguring applications can be complicated.

For instructions, see [Promoting a Read Replica](#) in the *Amazon RDS User Guide*.

After you promote a cross-Region Read Replica to be a standalone instance, if you want to later switch back to the original Region, you must create a new Read Replica. Unlike an Amazon RDS Multi-AZ configuration, this is not done for you automatically.

AWS BACKUP



■ AWS Backup es un servicio de copias de seguridad completamente administrado que facilita la tarea de centralizar y automatizar el respaldo de los datos en los servicios de AWS en la nube y en las instalaciones mediante AWS Storage Gateway. Al utilizar AWS Backup, puede configurar centralmente las políticas de copia de seguridad y supervisar la actividad de copia de seguridad para recursos de AWS, tales como los volúmenes de Amazon EBS, bases de datos de Amazon RDS, tablas de Amazon DynamoDB, sistemas de archivos ed Amazon EFS y volúmenes de AWS Storage Gateway. AWS Backup automatiza y consolida las tareas de respaldo que anteriormente se realizaban servicio por servicio, lo que elimina la necesidad de elaborar secuencias de comandos y procesos manuales personalizados. Con unos pocos clics en la consola de AWS Backup, puede crear políticas de respaldo que automaticen los cronogramas de creación de copias de seguridad y la administración de los períodos de retención. AWS Backup proporciona una solución de respaldo completamente administrada y basada en políticas, lo que permite simplificar la gestión de copias de seguridad y cumplir requisitos de conformidad normativos y empresariales relacionados con el respaldo.

Recursos admitidos

A continuación, se muestran los recursos de AWS de los que puede realizar una copia de seguridad y restaurar utilizando AWS Backup.

Servicio admitido	Recurso admitido
Amazon Elastic File System (Amazon EFS)	Sistemas de archivos de Amazon EFS
Amazon DynamoDB (DynamoDB)	Tablas de DynamoDB
Amazon Elastic Block Store (Amazon EBS)	Volúmenes de Amazon EBS
Amazon Relational Database Service (Amazon RDS)	Bases de datos de Amazon RDS*
AWS Storage Gateway (Volume Gateway)	Volúmenes de AWS Storage Gateway

PASO 1**Comenzar con AWS Backup**

Para comenzar con AWS Backup, mire el [video de introducción](#), inicie sesión en su cuenta de AWS y lance la consola de AWS Backup

[Iniciar sesión »](#)

PASO 2 »**Crear un plan de copias de seguridad**

Un plan de copias de seguridad define parámetros como la frecuencia con la que desea realizar copias de seguridad de los recursos y el tiempo de almacenamiento de estas.

[Más información »](#)

PASO 3**Asignar recursos de AWS**

Asigne recursos a los planes de copias de seguridad y AWS Backup iniciará automáticamente la copia de seguridad de estos y administrará la retención en su nombre

[Más información »](#)

PASO 4**Monitorizar, modificar, restaurar**

Una vez que se haga una copia de seguridad de los recursos, puede monitorizarla, modificarla o restaurarla según sea necesario

[Más información »](#)

Migración a Amazon Web Services

Costos operativos

Los principales componentes de los costos operativos son el precio unitario de infraestructura, la capacidad para hacer coincidir la oferta y la demanda, la búsqueda de una vía a la opcionalidad, el empleo de base de costos elástica y la transparencia.

Productividad del personal

Normalmente, la productividad aumenta debido a dos factores clave: no tener que esperar a la infraestructura y tener acceso a toda la extensión de AWS con sus más de 165 servicios que, de otro modo, tendría que crear y mantener. De hecho, es habitual ver mejoras de la productividad del personal del 30 al 50 % tras una migración de grandes dimensiones.

Reducción de costos

La eliminación de la necesidad de contar con programas de actualización de hardware y programas de mantenimiento constantes es el principal factor que contribuye a reducir los costos. Vemos que los clientes no tienen ningún interés en los costos y el esfuerzo necesarios para ejecutar un gran ciclo de actualización o la renovación de un centro de datos.

Resiliencia operativa

Esto puede parecer obvio, pero la reducción del perfil de riesgo de su organización también reduce el costo de la mitigación de los riesgos. Con 21 regiones formadas por 66 zonas de disponibilidad, AWS cuenta con presencia mundial para mejorar el tiempo de actividad y, por lo tanto, reducir los costos asociados al riesgo.

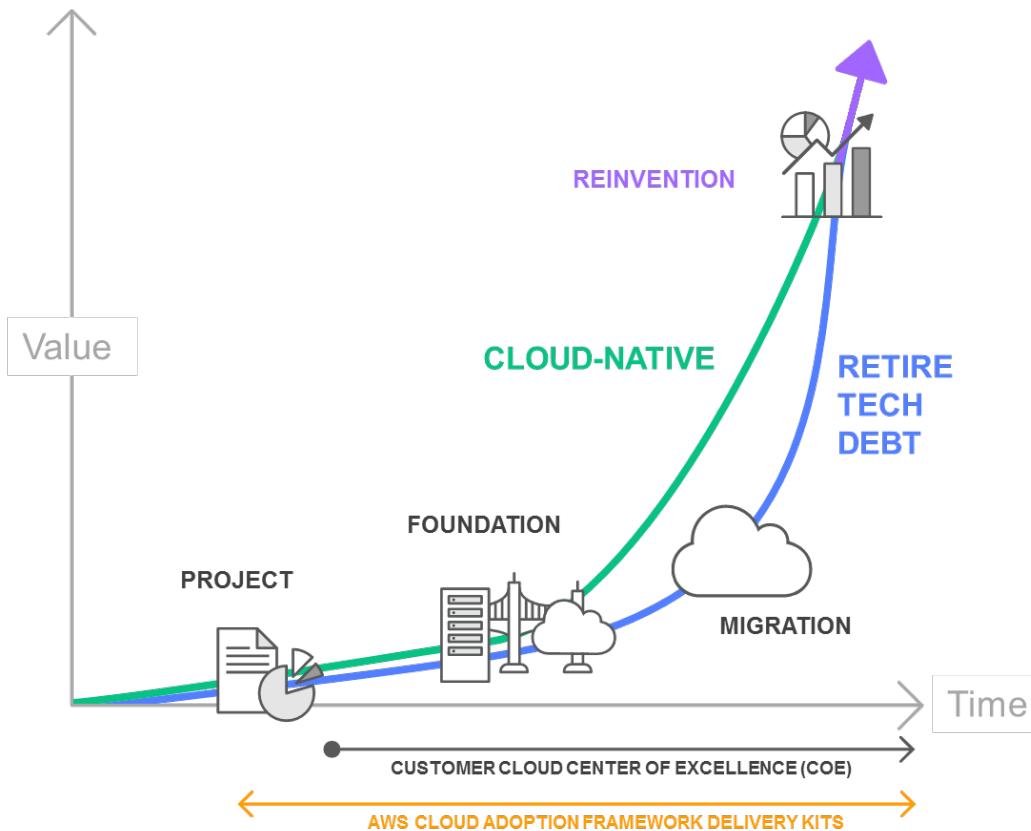
Agilidad empresarial

La migración a la nube de AWS lo ayuda a aumentar la agilidad operativa general. Esto le permite reaccionar a las condiciones del mercado con mayor rapidez mediante actividades, como la expansión a nuevos mercados, la venta de líneas del negocio y la adquisición de recursos disponibles que ofrecen una ventaja competitiva.

Fases de la adopción de la nube

La ruta a la adopción de la nube es única para cada empresa. Las fases de la adopción que aquí se describen pueden ser una buena forma de comprender algunos de los pasos.

La migración de un cliente a la nube fases:



PROYECTO

En la fase de proyecto, se ejecutan proyectos para familiarizarse con la nube y experimentar sus beneficios.

BASES

Después de experimentar los beneficios de la nube, se crea la base para escalar su adopción. Esto incluye la creación de una zona de contacto (entorno de AWS preconfigurado, seguro y con varias cuentas), de Cloud Center of Excellence (CCoE) y de un modelo de operaciones, así como la garantía de la seguridad y la preparación para la conformidad.

MIGRACIÓN

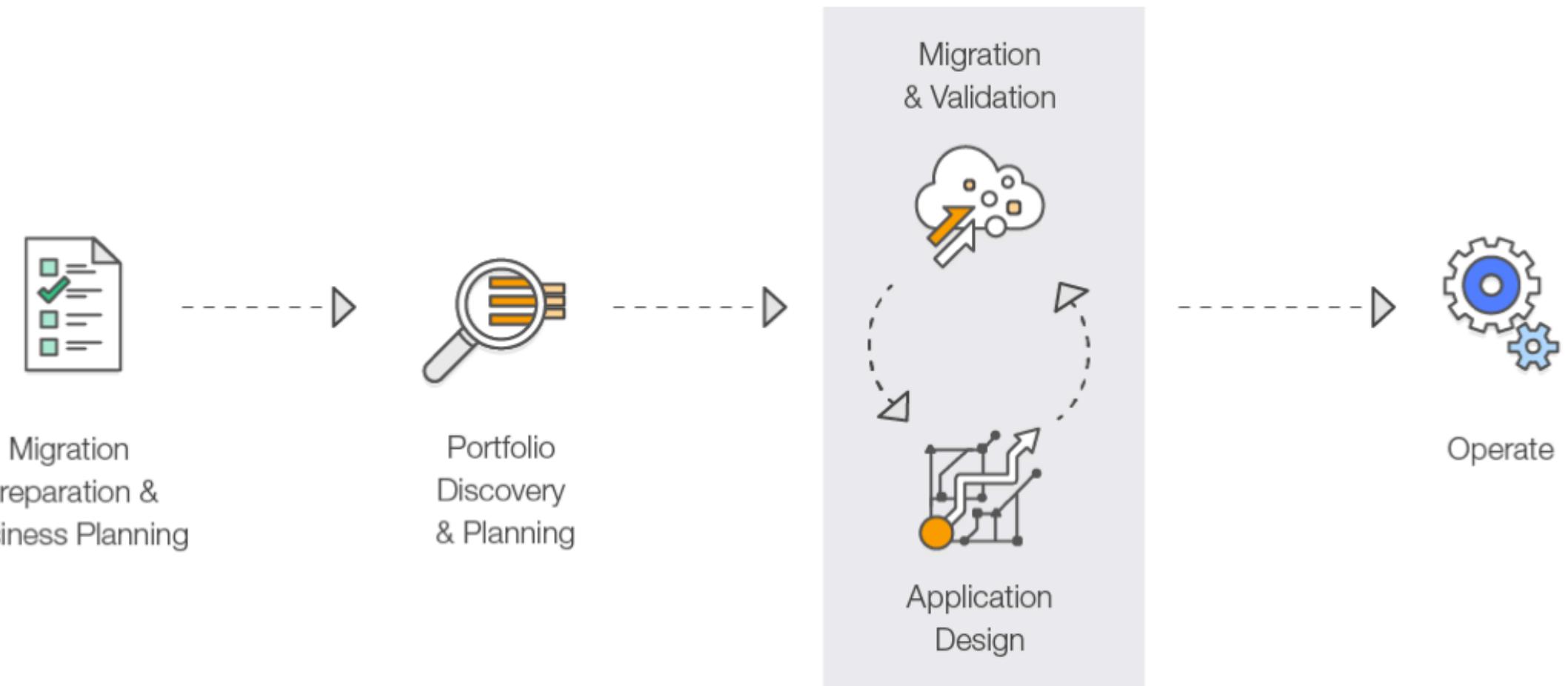
En esta fase, se migran a la nube las aplicaciones existentes, incluidas las aplicaciones críticas o los centros de datos completos, a medida que se adopta una parte cada vez mayor del portafolio de TI.

REINVENCIÓN

Ahora que sus operaciones se encuentran en la nube, puede centrarse en la reinención al aprovechar la flexibilidad y la funcionalidad de AWS para transformar su empresa, acelerando el tiempo de comercialización y aumentando la atención en la innovación.

El proceso de migración

En algunos casos, es posible que se planteen grandes migraciones heredadas de forma aislada; sin embargo, nos hemos dado cuenta de que las migraciones suelen ser parte de un esfuerzo de transformación empresarial mayor. Los patrones de migración a la nube que hemos observado suelen constar de cinco fases:



Fase 1: preparación de la migración y planificación empresarial

Aquí, se determinan los objetivos correctos y se comienzan a ver los tipos de beneficios que puede conseguir. Comienza con un poco de experiencia básica y el desarrollo de un caso empresarial preliminar para una migración. Esto requiere tener en cuenta los objetivos, además de la antigüedad y la arquitectura de las aplicaciones existentes y sus limitaciones. Tenemos socios, como RISC Networks, Atadata, Cloudamize, TSOLogic y Apptio, que tienen experiencia en esta área.

Fase 2: identificación del portafolio y planificación

A continuación, necesita conocer su portafolio de TI y las dependencias entre las aplicaciones, y empezar a considerar qué tipos de estrategias de migración tendrá que emplear para cumplir los objetivos de su caso empresarial. Una vez realizada la identificación y la migración del portafolio, podrá crear un caso empresarial completo. Si necesita ayuda para comprender su portafolio de TI, puede trabajar con socios como RISC Networks, Cloudamize y Atadata, o utilizar [AWS Application Discovery Service](#).

Fases 3 y 4: diseño, migración y validación de la aplicación

Aquí, el foco se traslada del nivel del portafolio al nivel de cada aplicación, que se diseña, se migra y se valida de forma individual. Cada aplicación se diseña, se migra y se valida de acuerdo con una de las seis estrategias de aplicaciones comunes, conocidas como ["las 6 R"](#). Una vez que ya cuente con un poco de experiencia en la migración de algunas aplicaciones y tenga un plan que respalde a la organización, habrá llegado el momento de acelerar la migración y aumentar la escala. Socios como CloudVelox, Atadata, Racemi y Attuinity pueden ayudarlo en tal sentido, así como también pueden hacerlo los productos [AWS Server Migration Service \(SMS\)](#), [AWS Database Migration Service \(DMS\)](#) y [CloudEndure Migration](#).

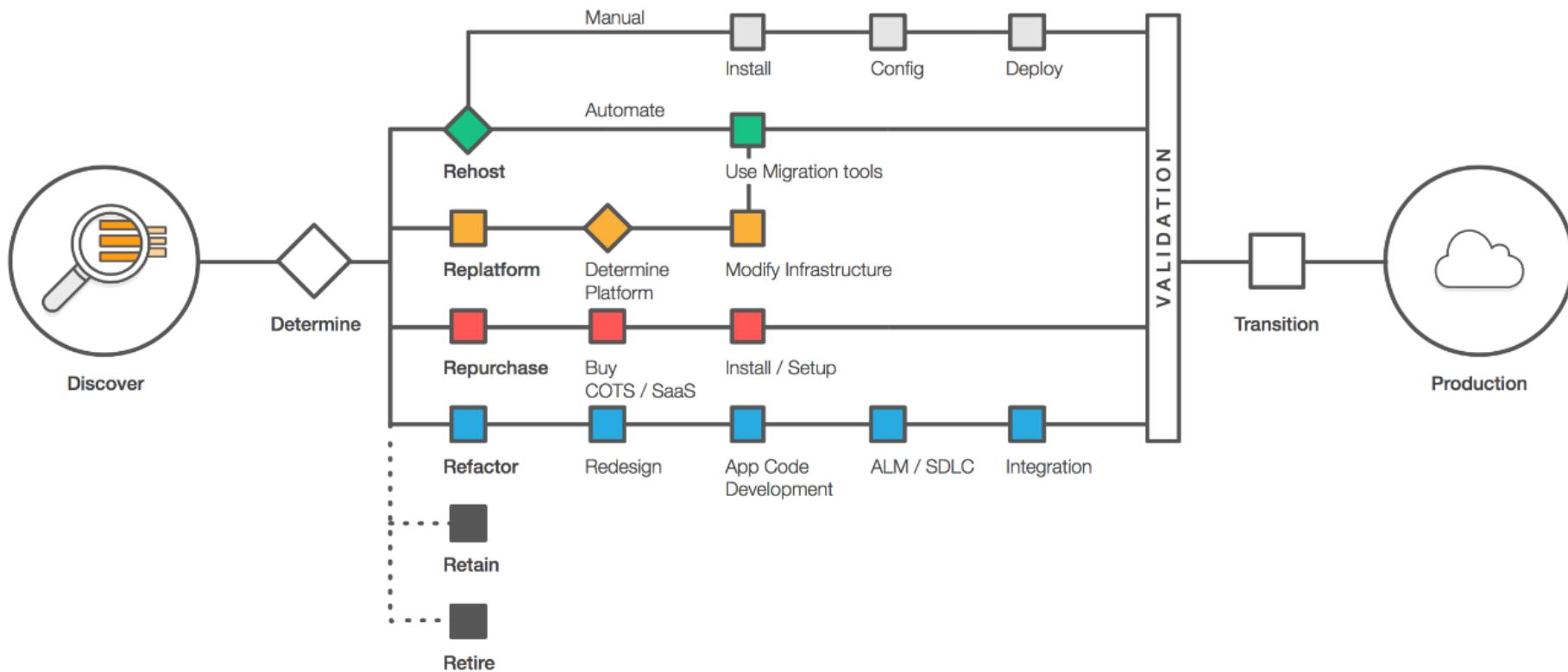
Fase 5: operación

A medida que se migran las aplicaciones, usará de forma repetida su nueva base, desactivará los sistemas antiguos y avanzará constantemente hacia un modelo operativo moderno. Su modelo operativo se convierte en un conjunto duradero de personas, procesos y tecnología que mejoran constantemente a medida que migra más aplicaciones. Contamos con algunos socios, como AppDynamics, NewRelic y Dynatrace, que pueden ayudarlo a seguir avanzando en su modelo operativo a medida que mueve más elementos a la nube.

Seis estrategias comunes de migración de aplicaciones

Las organizaciones suelen empezar a pensar en cómo migrar una aplicación durante la fase 2 del proceso de migración. Aquí es cuando se determina lo que hay en el entorno y la estrategia de migración para cada aplicación. Los seis enfoques que se describen a continuación son estrategias de migración comunes y se basan en los cinco criterios que Gartner planteó en 2011.

Debe estudiar a fondo cuál es la estrategia de migración que mejor se ajusta a determinadas partes de su portafolio



1. Realojamiento (“lift and shift”)

En los escenarios de grandes migraciones heredadas, en los que la organización busca implementar rápidamente su migración y escalar para poder dar servicio a un caso empresarial, vemos que la mayoría de las aplicaciones se ubican en otro alojamiento. La mayoría del realojamiento puede automatizarse con herramientas como [AWS SMS](#), aunque quizás prefiera hacerlo de forma manual mientras aprende a aplicar sus sistemas heredados en la nube.

Es posible que descubra también que es más fácil rediseñar las aplicaciones una vez que se están ejecutando en la nube. Esto ocurre, en parte, porque su organización ya habrá desarrollado mejores habilidades para hacerlo y también porque la parte difícil (la migración de la aplicación, los datos y el tráfico) ya se ha realizado.

2. Reforma de la plataforma (“lift, tinker and shift”)

Esto implica hacer algunas optimizaciones de la nube para lograr algún beneficio tangible sin cambiar la arquitectura central de la aplicación. Por ejemplo, es posible que desee reducir la cantidad de tiempo que emplea en administrar instancias de bases de datos mediante la migración a un servicio de bases de datos relacionales administrado, como, por ejemplo, [Amazon Relational Database Service \(RDS\)](#), o mediante la migración de su aplicación a una plataforma completamente administrada como [AWS Elastic Beanstalk](#).

3. Readquisición (“drop and shop”)

Se decide cambiar a un producto diferente, lo que probablemente significa que su organización está dispuesta a cambiar el modelo de licencias que ha estado usando. Para las cargas de trabajo que pueden actualizarse fácilmente a versiones más recientes, esta estrategia podría permitir actualizar el conjunto de características y realizar una implementación sin problemas.

4. Refactorización/rediseño

Normalmente, el factor que impulsa esto es una fuerte necesidad empresarial de añadir características, escala o rendimiento que, de otra manera, serían difíciles de conseguir en el entorno existente de la aplicación. Si su organización busca potenciar la agilidad o mejorar la continuidad del negocio mediante un cambio a una arquitectura orientada a servicios (SOA), podría resultar interesante seguir esta estrategia aunque suele ser la solución más cara.

5. Retiro

Identificar los recursos de TI que ya no son útiles y pueden desactivarse lo ayudará a impulsar su caso empresarial y a centrarse en mantener los recursos más utilizados.

6. Retención

Quizás quiera conservar parte de su portafolio de TI porque todavía no está preparado para migrar algunas aplicaciones y se siente más cómodo manteniéndolas de forma local, o quizás no está listo para priorizar una aplicación que se actualizó recientemente y volver a cambiarla.

Herramientas y servicios de migración de AWS

Existen varias herramientas que ayudan a automatizar la migración de las aplicaciones:

Seguimiento de las tareas de detección y migración



AWS Migration Hub

[AWS Migration Hub](#) ofrece una ubicación única para realizar el seguimiento de los avances de las migraciones de aplicaciones en varias soluciones de AWS y sus socios. El uso de Migration Hub le permite elegir las herramientas de migración de AWS y de socios que mejor se adapten a sus necesidades, al mismo tiempo que se le ofrece visibilidad acerca del estado de las migraciones que conforman su portafolio de aplicaciones. Migration Hub también suministra información sobre el progreso y las métricas clave de aplicaciones individuales, independientemente de las herramientas que se estén utilizando para migrarlas.



AWS Application Discovery Service

[AWS Application Discovery Service](#) lo ayuda a planificar proyectos de migración al recopilar información sobre sus centros de datos locales. La planificación de las migraciones de centros de datos puede conllevar miles de cargas de trabajo que, a menudo, tienen un profundo nivel de interdependencia. AWS Application Discovery Service recopila y presenta datos de configuración, uso y comportamiento de sus servidores para ayudarlo a entender mejor el funcionamiento de sus cargas de trabajo.

TSO Logic

[TSO Logic](#) ofrece recomendaciones precisas basadas en datos para calcular el tamaño y el costo correctos. Nuestros análisis predictivos proporcionan información de forma continua para garantizar que siempre esté ejecutando cada aplicación en el mejor lugar, con el software adecuado y con el menor costo total de propiedad (TCO), incluso cuando el entorno, las opciones de la nube y los precios cambian. TSO Logic lo ayuda a crear un caso empresarial claro para acelerar la planificación de su migración.

Migración de base de datos y de servidor



AWS Server Migration Service

[AWS Server Migration Service \(SMS\)](#) es un servicio sin agente que le permite migrar de forma más rápida y sencilla miles de cargas de trabajo locales a AWS. Con AWS SMS, puede automatizar, programar y monitorizar replicaciones graduales de volúmenes de servidores en directo, lo que facilita la coordinación de migraciones de servidores a gran escala.

AWS Database Migration Service

[AWS Database Migration Service \(DMS\)](#) ayuda a migrar las bases de datos a AWS de manera fácil y segura. La base de datos de origen permanece totalmente operativa durante la migración, lo que minimiza el tiempo de inactividad de las aplicaciones que dependen de ella. AWS Database Migration Service puede migrar datos hacia y desde la mayoría de las bases de datos comerciales de código abierto de uso más generalizado.

VMware Cloud on AWS

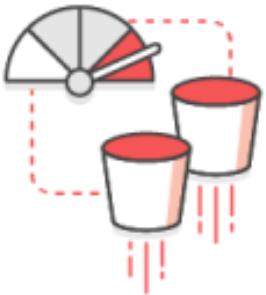
[VMware Cloud on AWS](#) es un producto integrado de la nube que fue desarrollado de manera conjunta por AWS y VMware, y que provee un servicio con alto nivel de escalabilidad, seguro e innovador cuyo fin es permitir a las organizaciones ampliar sin problemas sus entornos locales basados en VMware vSphere y migrarlos a la nube de AWS, con infraestructura dedicada de Amazon Elastic Compute Cloud (Amazon EC2) de última generación. VMware Cloud on AWS es ideal para las organizaciones con operaciones e infraestructuras de TI empresariales que desean migrar sus cargas de trabajo locales basadas en vSphere a la nube pública, consolidar y ampliar sus capacidades de centros de datos, además de optimizar, simplificar y modernizar sus soluciones de recuperación de desastres. VMware y sus socios brindan soporte para VMware Cloud on AWS, además de vender y entregar este producto en todo el mundo.



CloudEndure Migration

[CloudEndure Migration](#) simplifica, agiliza y automatiza la migración de aplicaciones desde infraestructuras físicas, virtuales y basadas en la nube a AWS. También convierte de forma automática cualquier aplicación que se ejecute en sistemas operativos compatibles, lo que permite una funcionalidad completa en AWS sin problemas de compatibilidad. Durante este proceso de replicación, sus aplicaciones continuarán ejecutándose en el origen sin tiempo de inactividad o interrupción del rendimiento. Luego de un período mínimo de transición, sus servidores migrados se ejecutarán de forma nativa en AWS.

Migración de datos



Amazon S3 Transfer Acceleration

[Amazon S3 Transfer Acceleration](#) incrementa la velocidad de las transferencias a Amazon S3 a través del Internet público. Puede maximizar su ancho de banda independientemente de la distancia o de las variaciones en la capacidad de Internet, sin necesidad de clientes especiales o protocolos de red propios. Simplemente cambie el punto de enlace que utiliza con el bucket de S3 y se aplicará la aceleración automáticamente.



AWS Snowball

[AWS Snowball](#) es una solución de transferencia de datos a escala de petabytes que emplea dispositivos seguros para transferir grandes volúmenes de datos hacia y desde AWS. La utilización de Snowball permite resolver los desafíos propios de las transferencias de datos a gran escala, entre los que se incluye los altos costos de red, los tiempos prolongados de transferencia y los riesgos de seguridad.



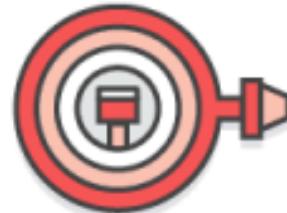
AWS Snowmobile

[AWS Snowmobile](#) es un servicio de transferencia de datos a escala de exabytes que se utiliza para transferir volúmenes extremadamente grandes de datos a AWS. Puede transferir hasta 100 PB por Snowmobile, un contenedor de envío reforzado de 13,71 metros de longitud, colocado en un camión semitráiler. Snowmobile facilita la transferencia de volúmenes masivos de datos a la nube, incluidas bibliotecas de videos, repositorios de imágenes o incluso la migración de un centro de datos completo.



AWS Direct Connect

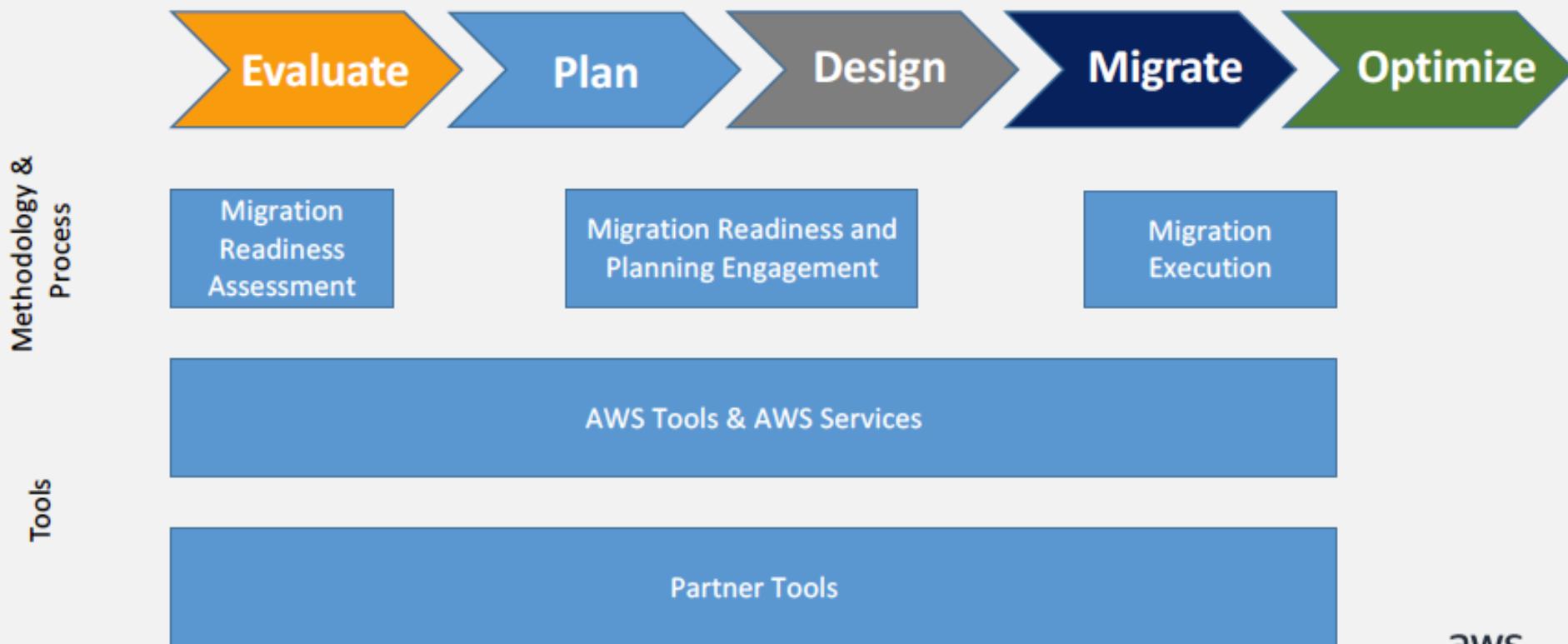
[AWS Direct Connect](#) le permite establecer una conexión de red dedicada entre su red y una de las ubicaciones de AWS Direct Connect. Gracias al uso de redes estándares del sector VLAN 802.1q, esta conexión exclusiva se puede dividir en varias interfaces virtuales. Esto le permite utilizar la misma conexión para obtener acceso a recursos públicos, como objetos almacenados en Amazon S3 utilizando un espacio de dirección IP pública, además de acceso a recursos privados, como instancias Amazon EC2 que se ejecutan dentro de una Amazon Virtual Private Cloud (VPC) utilizando un espacio de dirección IP privada, al mismo tiempo que se mantiene la separación de red entre los entornos públicos y privados. Las interfaces virtuales se pueden volver a configurar en cualquier momento para que satisfagan sus necesidades a medida que cambian.



Amazon Kinesis Firehose

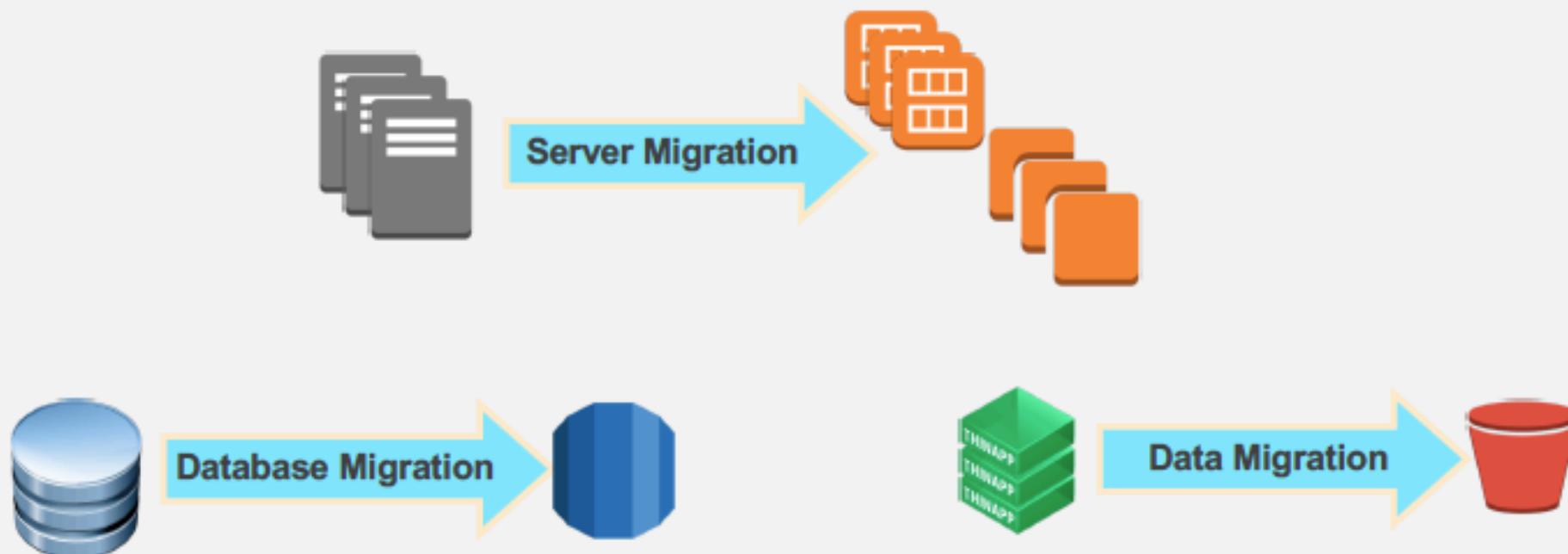
[Amazon Kinesis Firehose](#) es la forma más fácil de cargar datos de streaming en AWS. Puede capturar y cargar automáticamente los datos de streaming en Amazon S3 y Amazon RedShift, lo que habilita el análisis casi en tiempo real con las herramientas y los paneles de control de inteligencia empresarial existentes que ya emplea hoy en día. Se trata de un servicio completamente administrado cuya capacidad se ajusta automáticamente para adaptarse al nivel de procesamiento de los datos y que no necesita administración permanente. También puede procesar por lotes, comprimir y cifrar los datos antes de cargarlos, a fin de minimizar la cantidad de almacenamiento utilizado en el destino y aumentar la seguridad. Puede crear fácilmente una transmisión de entrega de Firehose desde la consola de administración de AWS, configurarla con unos pocos clics y comenzar a enviar datos a la transmisión desde cientos de miles de orígenes de datos para que se carguen continuamente en AWS, y todo en pocos minutos.

Migrating to AWS



Migration Execution

Each application migration usually involves one or more of each type different types of resource:





Tracking Migrations

Workload migrations require **different** migration **tools** like Database Migration Service, Server Migration Service, Application Discovery Service, Snowball etc.



Existing tools don't track migrations by **application**

No easy way to **track** the state of **migrations** in a single location

The screenshot shows the AWS Migration Hub console. The left sidebar has a 'Migration Hub' header and two main sections: 'Discover' (with 'Data collection', 'Servers', and 'Applications' options) and 'Migrate' (with 'Applications', 'Updates', and 'Tools' options). The main content area has a title 'AWS Migration Hub' and a subtitle: 'Migration Hub simplifies and accelerates discovery and migration from your data centers to the AWS Cloud.' It features three large buttons: 'Discover' (yellow), 'Migrate' (green), and 'Track' (blue). Below each button is a list of associated actions: 'Discover' lists 'Deploy AWS discovery tools (Optional)'; 'Migrate' lists 'Connect migration tools', 'Migrate using connected tools', and 'Group servers as applications'; 'Track' lists 'Track status of migrations' with a 'View example' link. A section titled 'What would you like to do?' contains 'Get started with discovery' and 'Get started migrating' buttons. At the bottom, there are four columns: 'Integrated discovery tools:' (AWS Discovery Connectors, AWS Discovery Agents), 'Integrated migration tools:' (AWS Database Migration Service, AWS Server Migration Service, CloudEndure Live Migration, Racemi DynaCenter), 'AWS migration programs:' (Professional Services, Migration Acceleration Program, Migration Partners Solutions), and 'Documentation & support:' (User Guide, Forums, Contact us). The footer includes links for 'Feedback', 'English', 'Privacy Policy', 'Terms of Use', and copyright information: '© 2008 - 2017, Amazon Web Services, Inc. or its affiliates. All rights reserved. © 2018, Amazon Web Services, Inc. or its Affiliates. All rights reserved. Amazon Confidential and Trademark'.

Introducing AWS Migration Hub!

- Migration Hub provides a single location to track the progress of application migrations across multiple AWS and partner solutions
- Using Migration Hub allows you to choose the AWS and partner migration tools that best fit your needs, while providing visibility into the status of migrations across your portfolio of applications



Integrated Migration Tools

With SMS, Import VM images from your existing environment to ready-to-use Amazon EC2 instances



AWS Server
Migration Service

DMS helps you migrate databases to AWS easily and securely



AWS Database
Migration Service

ATADATA ATAmotion auto-migrates live workloads directly to AWS EC2 or VPC from any physical, virtual or cloud source, without agents



CloudEndure Live Migration provides automated migration to AWS from any physical or virtual infrastructure



Racemi DynaCenter enables partners and customers to easily migrate server workloads between dissimilar physical and virtual platforms



© 2018, Amazon Web Services, Inc. or its Affiliates. All rights reserved. Amazon Confidential and Trademark



AWS
Migration Hub



Key Benefits



Improved Visibility



Centralized Migration Tracking



Control Access



Migration Flexibility



Multi-Region Migrations



Extend Migration Tracking with API

The screenshot shows the AWS Migration Hub console. On the left, there's a sidebar with navigation links: Dashboard, Discover, Data collection, Servers, Applications, Migrate (selected), Applications, Updates, Tools, Help & Support. The main area shows the 'Time Tracking App' status as 'In-progress'. A description states: 'This is an application to track employee time.' There are four tabs for server status: 'Servers - Inactive (1)', 'Servers - In-progress (4)', 'Servers - Completed (2)', and 'Servers - Error (0)'. Below these tabs is a table with columns: Server ID, Host/VM name, Tool, Status, Migration result, and Last updated time. The table lists eight servers, each with a progress bar and a status message. At the bottom, there are links for 'Feedback' and 'English'.

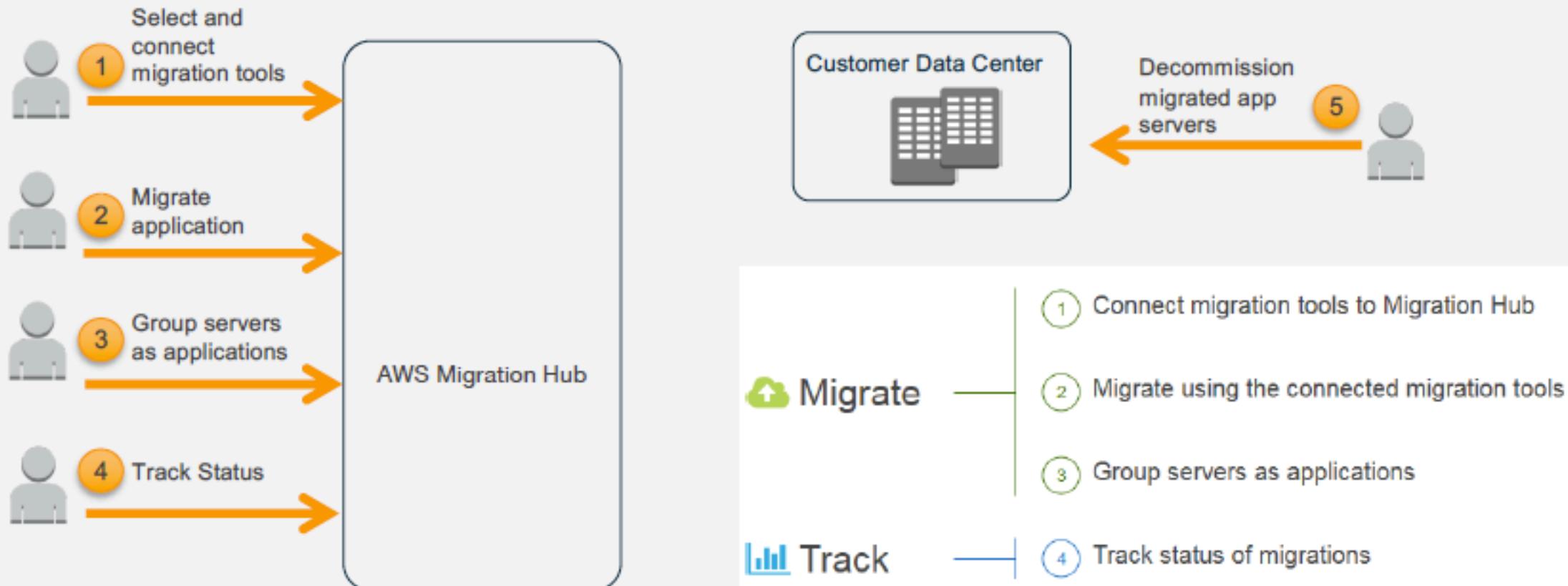
Server ID	Host/VM name	Tool	Status	Migration result	Last updated time
...485dd3ee	time1.acme.com	SMS	✓ Completed, next run sched...	Latest AMI	2017-08-14 08:00 AM
...49770313	time4.acme.com	Racemi	⌚ 19% Deploying target wh...		2017-08-14 07:58 AM
...c14edd9a	time3.acme.com	SMS	✓ Completed, next run sched...	Latest AMI	2017-08-14 07:56 AM
...ed1c2be7	time5.acme.com	DMS	⌚ 90% in-progress	Target endpoint	2017-08-14 07:50 AM
...48556723	time6.acme.com	CloudEndure	✓ 100% Cut over	EC2 instance	2017-08-14 07:49 AM
...4nkenelie	time2.acme.com	SMS	⌚ Preparing AMI		2017-08-14 07:48 AM
...09dfebfk	time7.acme.com	SMS	⌚ Preparing AMI		2017-08-14 07:46 AM
...23fd9r3h	time8.acme.com		Not started		

Migration Tracking

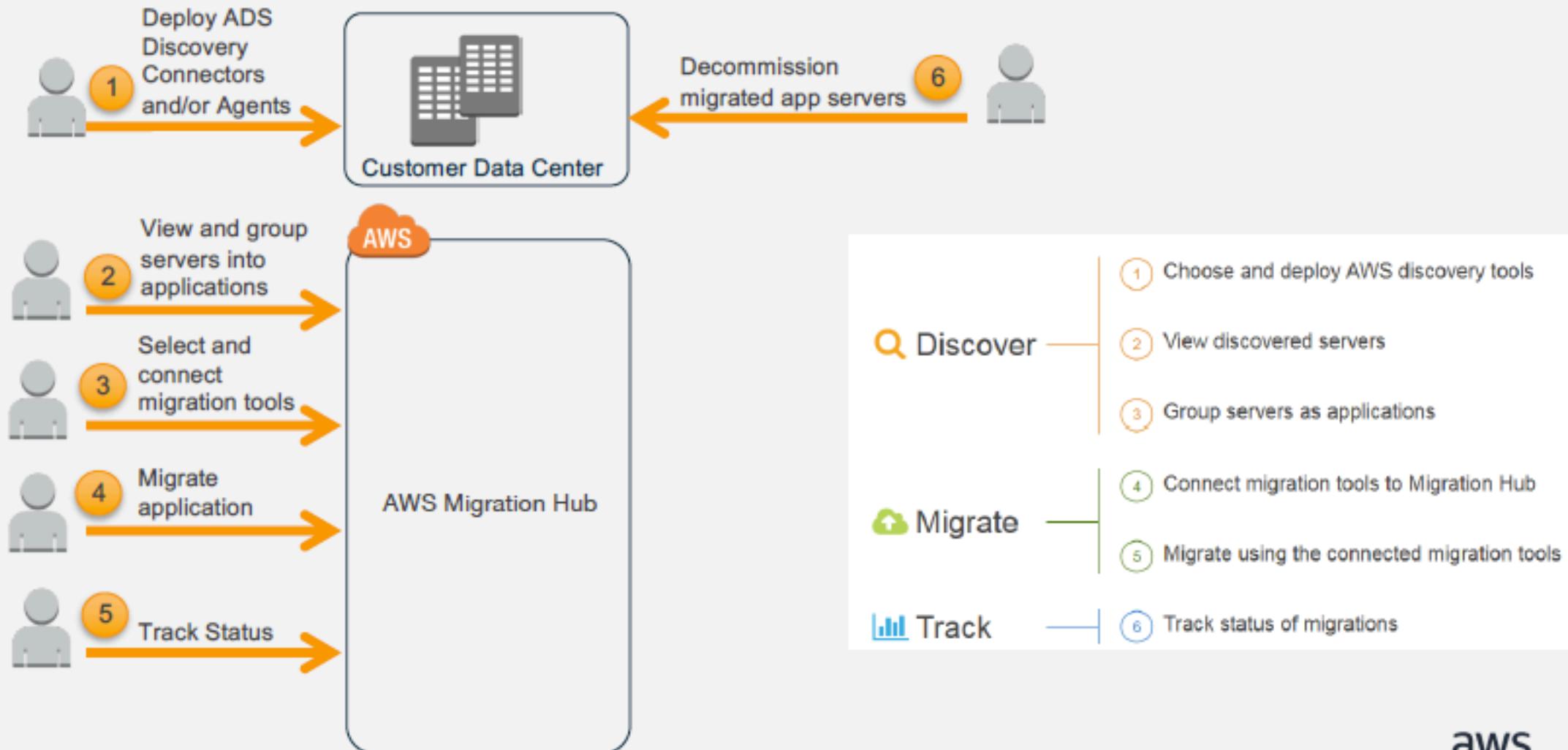
- The migration tools, if authorized, automatically send status updates and results back to Migration Hub, for display on the migration status page for the application.



Migration only using AWS Migration Hub



Discovery and Migration using AWS Migration Hub



AWS Database Migration Service (AWS DMS)

DMS migrates databases to AWS easily and securely with minimal downtime. It can migrate your data to and from most widely used commercial and open-source databases.



ORACLE



Amazon Aurora



New NoSQL support

Migrate to AWS

- Move from MongoDB to Amazon DynamoDB
- Move from MongoDB to relational db's



© 2018, Amazon Web Services, Inc. or its Affiliates. All rights reserved.

Move between NoSQL and SQL

- Change technologies

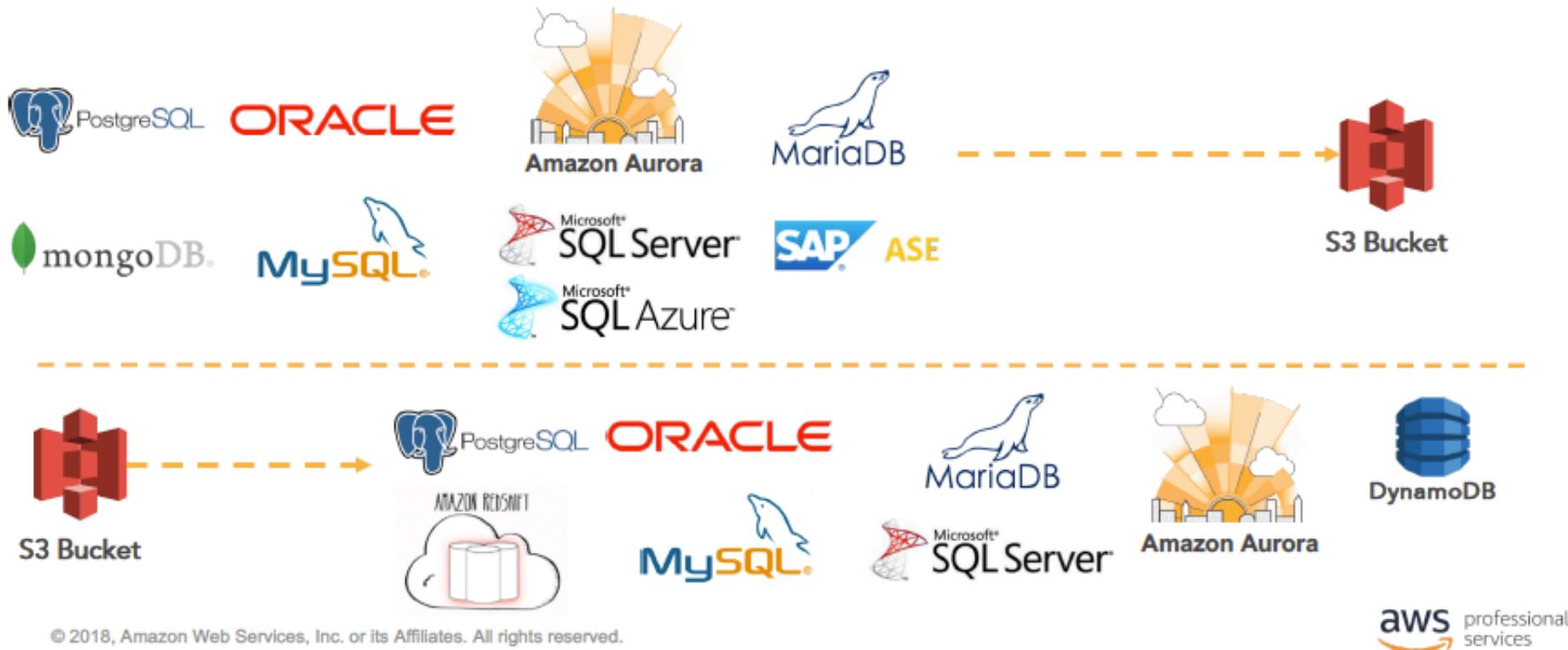
ORACLE



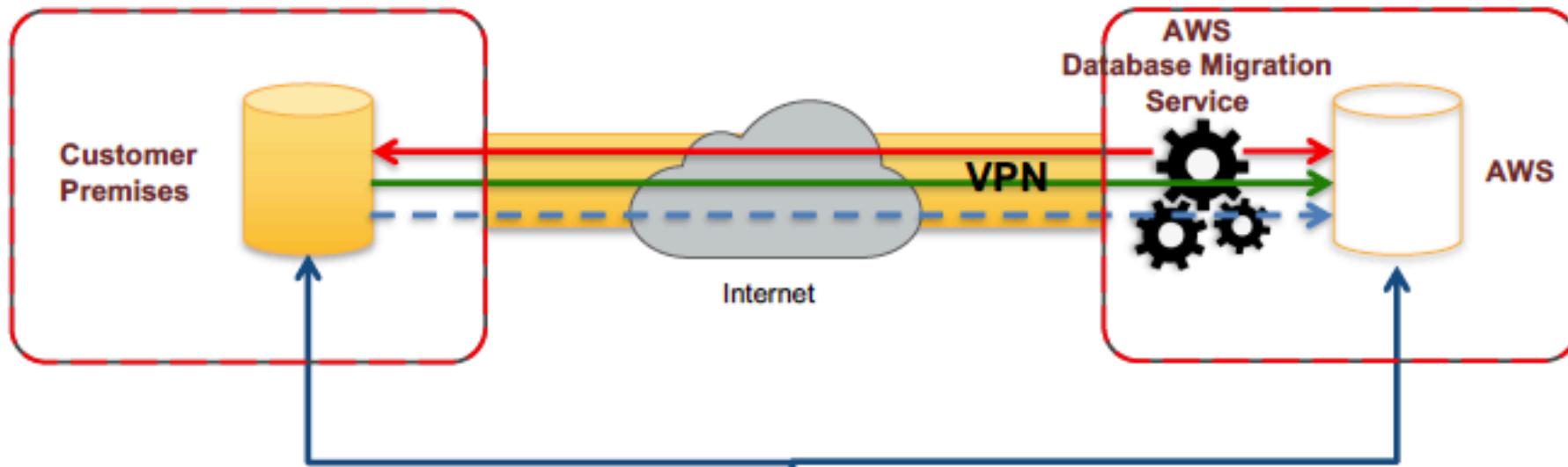
aws professional services

New Support for S3 as a Source and Target

Extract Data from any supported DMS source to S3 and to any DMS target



© 2018, Amazon Web Services, Inc. or its Affiliates. All rights reserved.

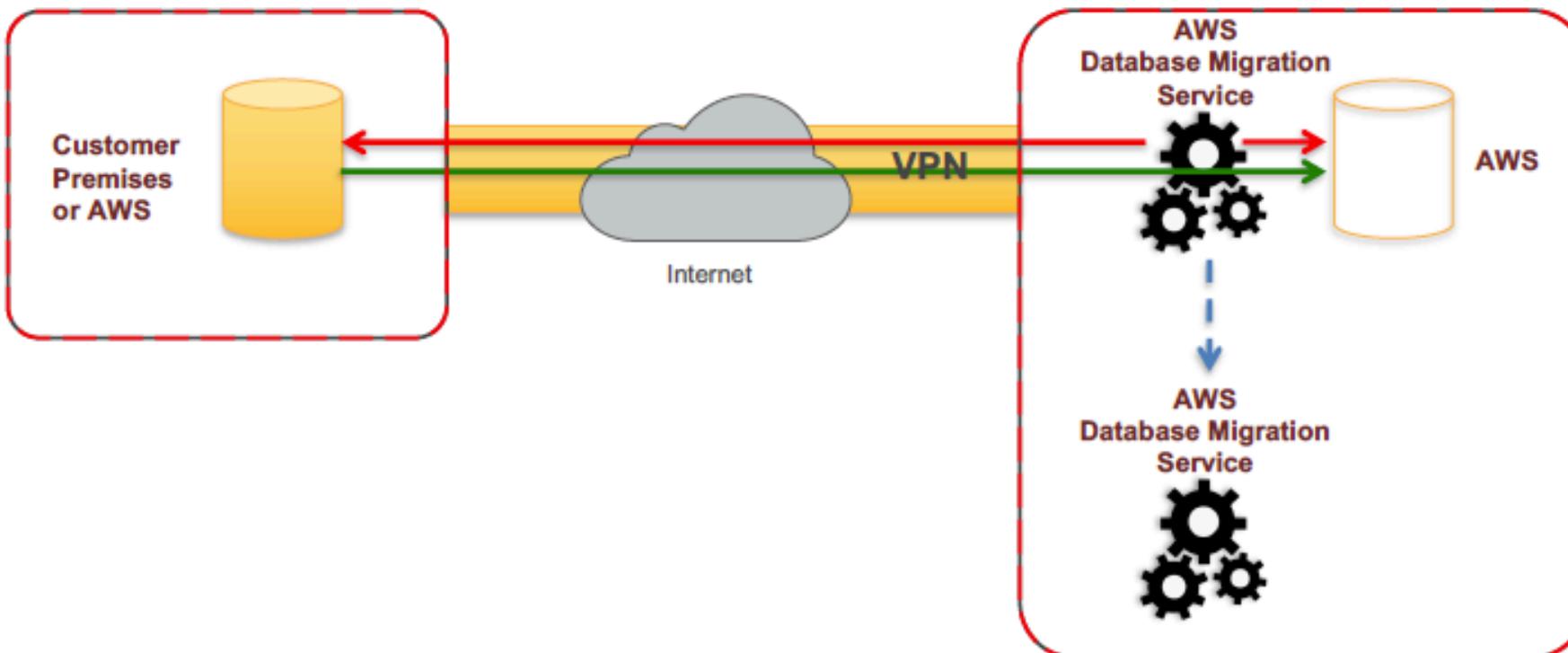


Start a replication instance
Connect to source and target databases
Select tables, schemas, or databases

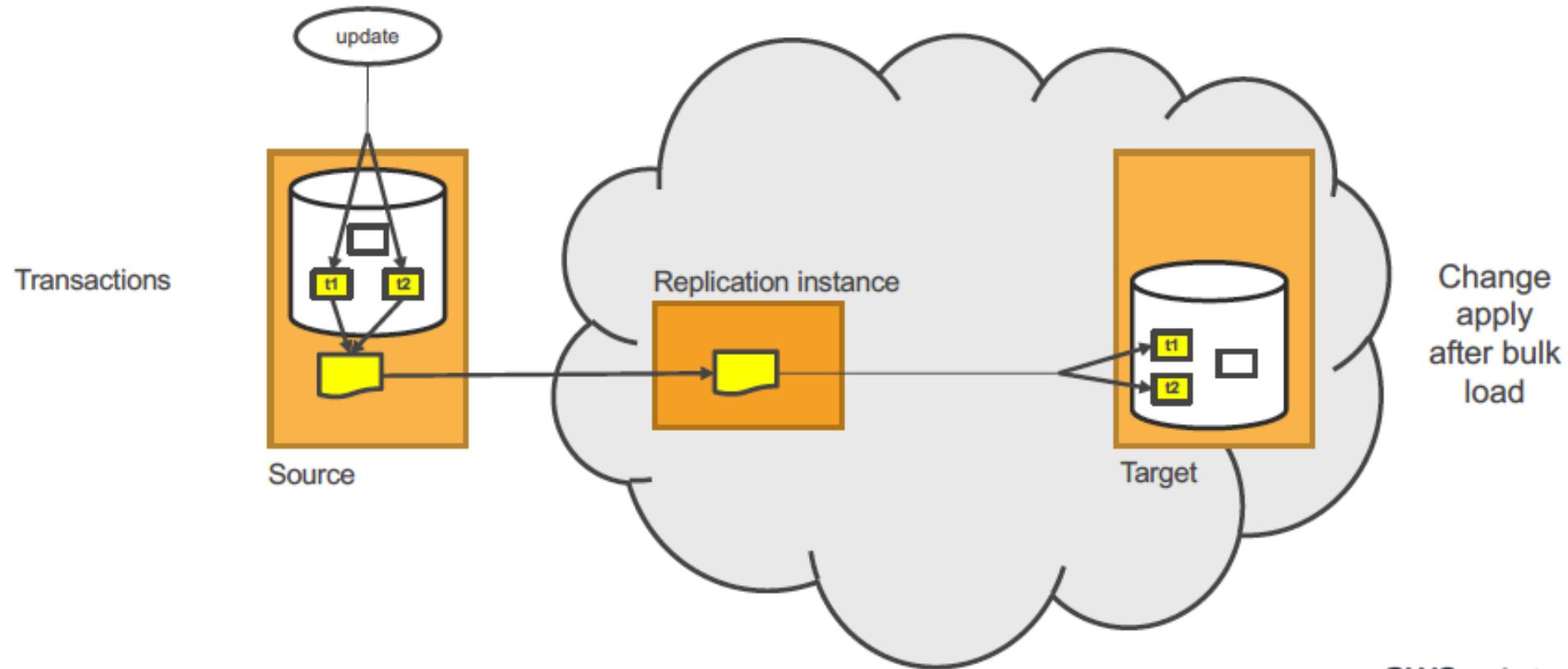


Let AWS DMS create tables, load data, and keep them in sync
Switch applications over to the target at your convenience

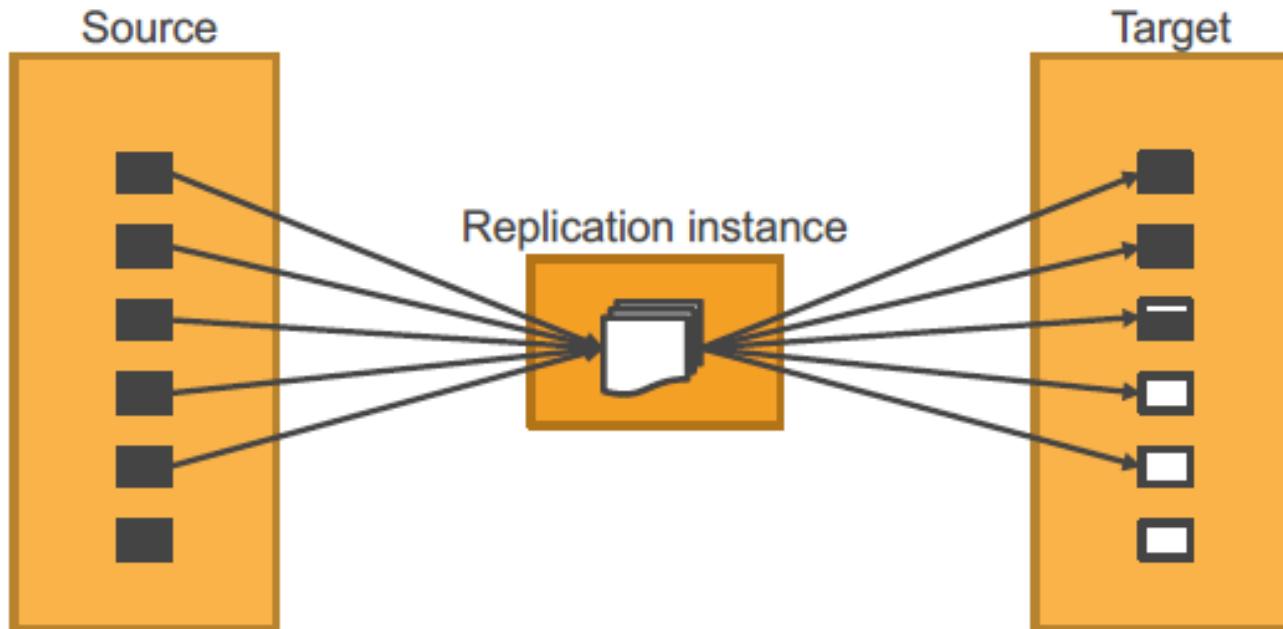
Multi-AZ Option for High Availability



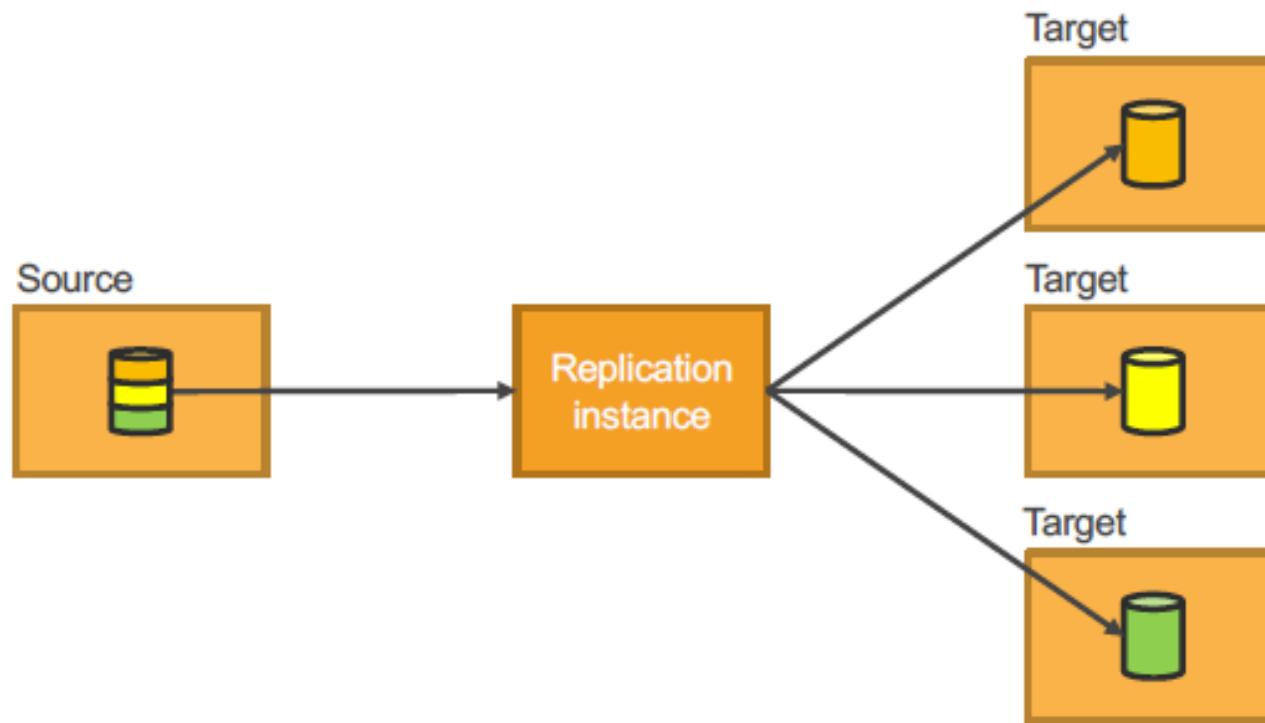
Change Data Capture (CDC) and Apply



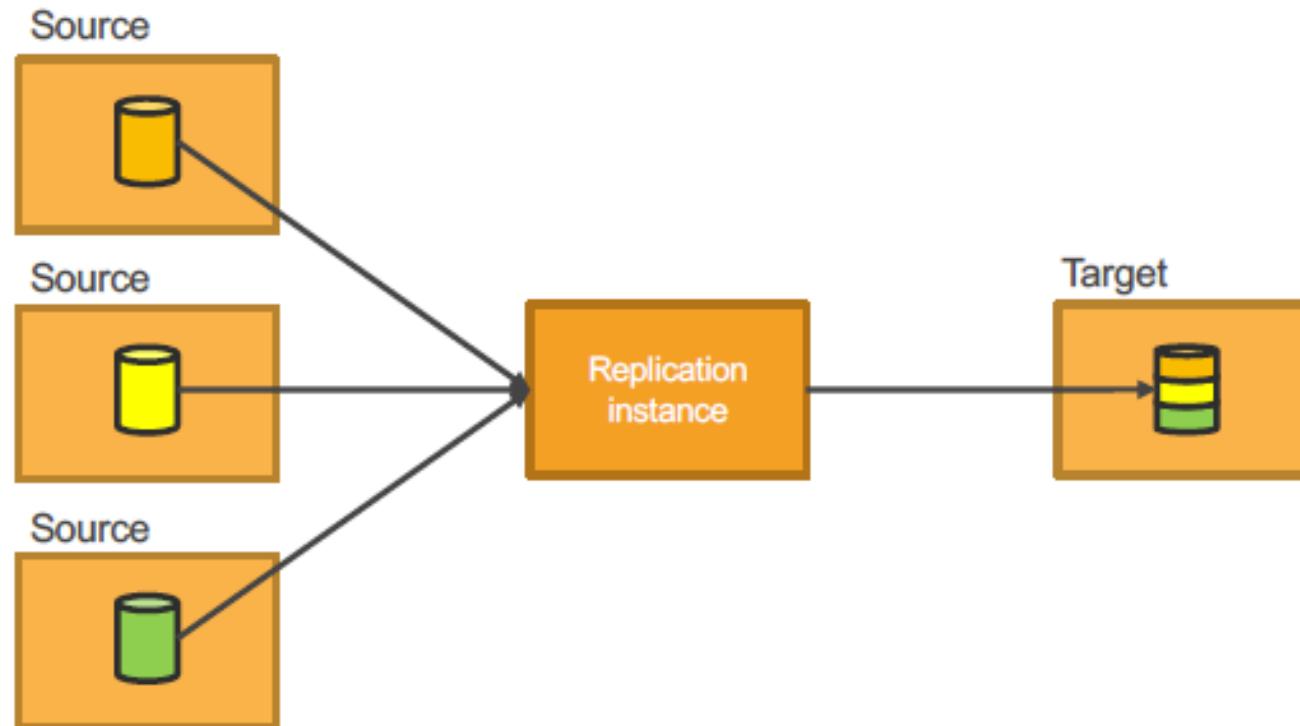
Load is Table by Table



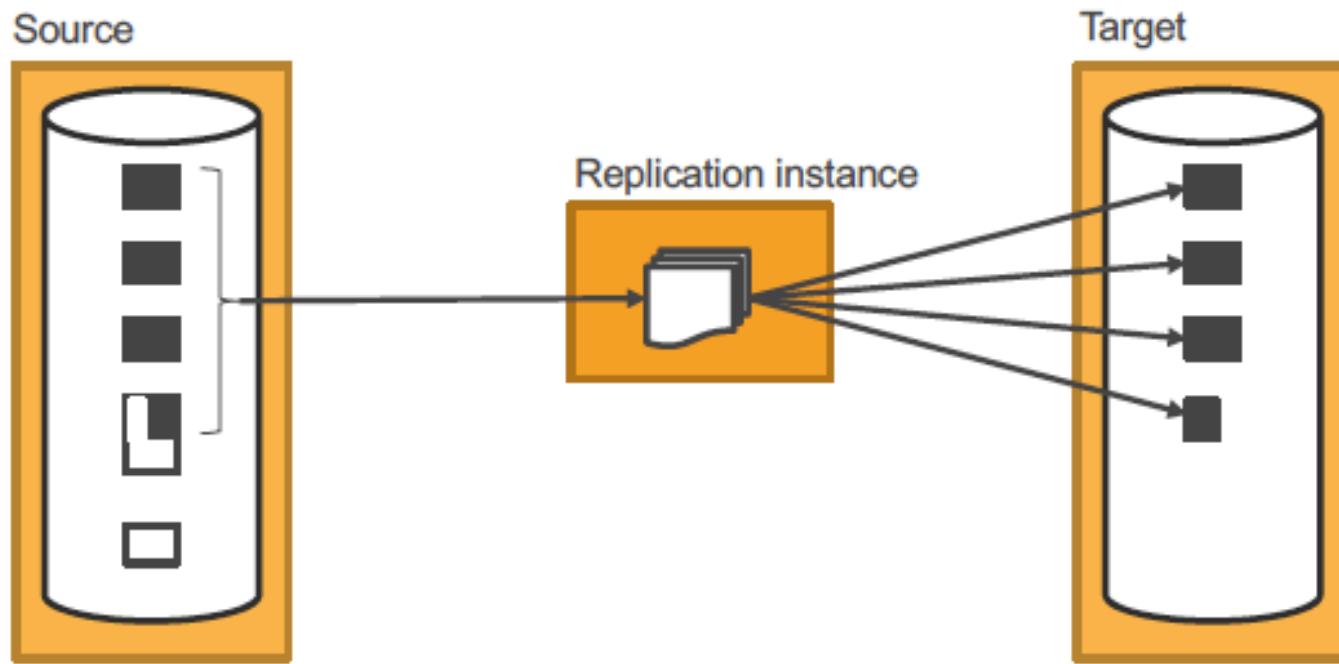
Multiple Targets



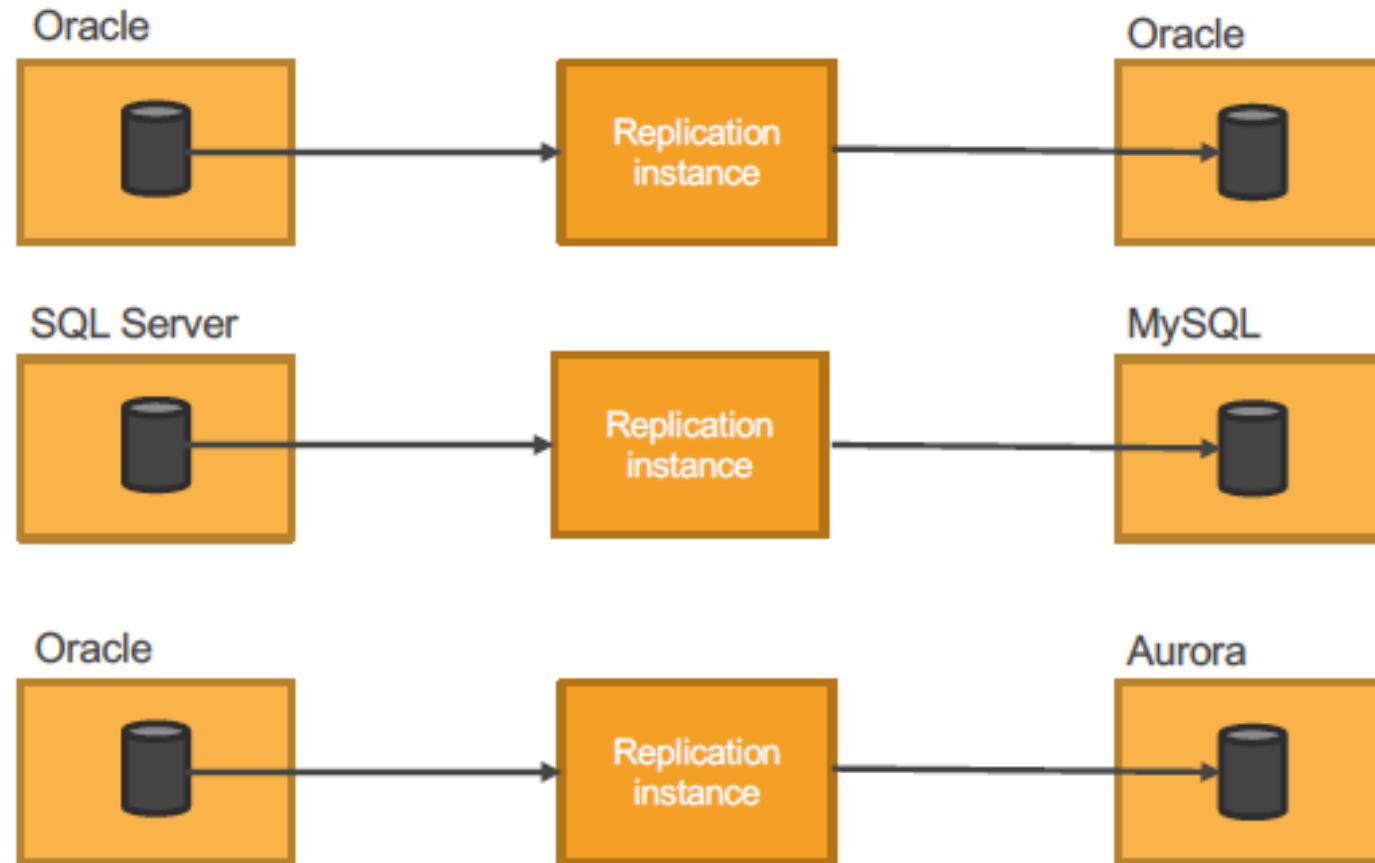
Multiple Sources



Customers Don't Have to Take Everything

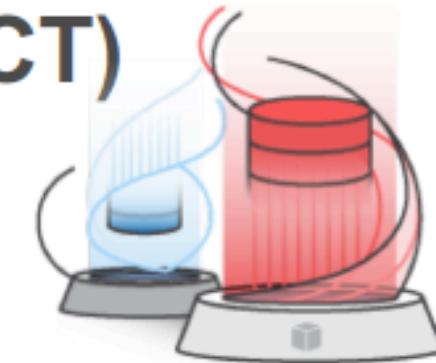


Homogenous or Heterogeneous



AWS Schema Conversion Tool (AWS SCT)

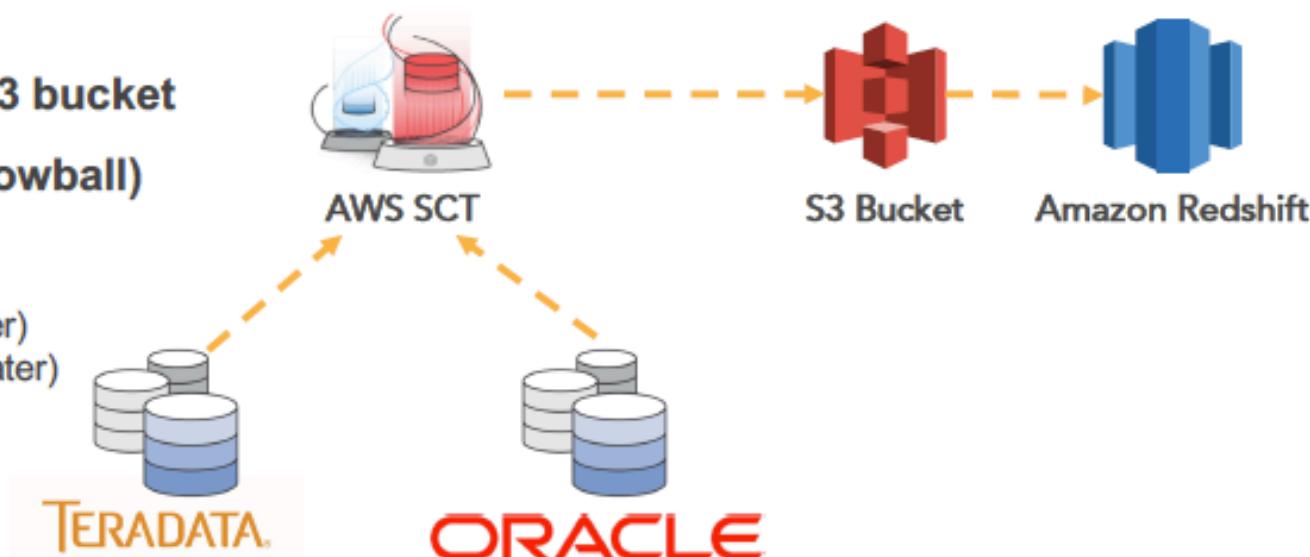
SCT helps automate many database schema and code conversion tasks when migrating between database engines or data warehouse engines



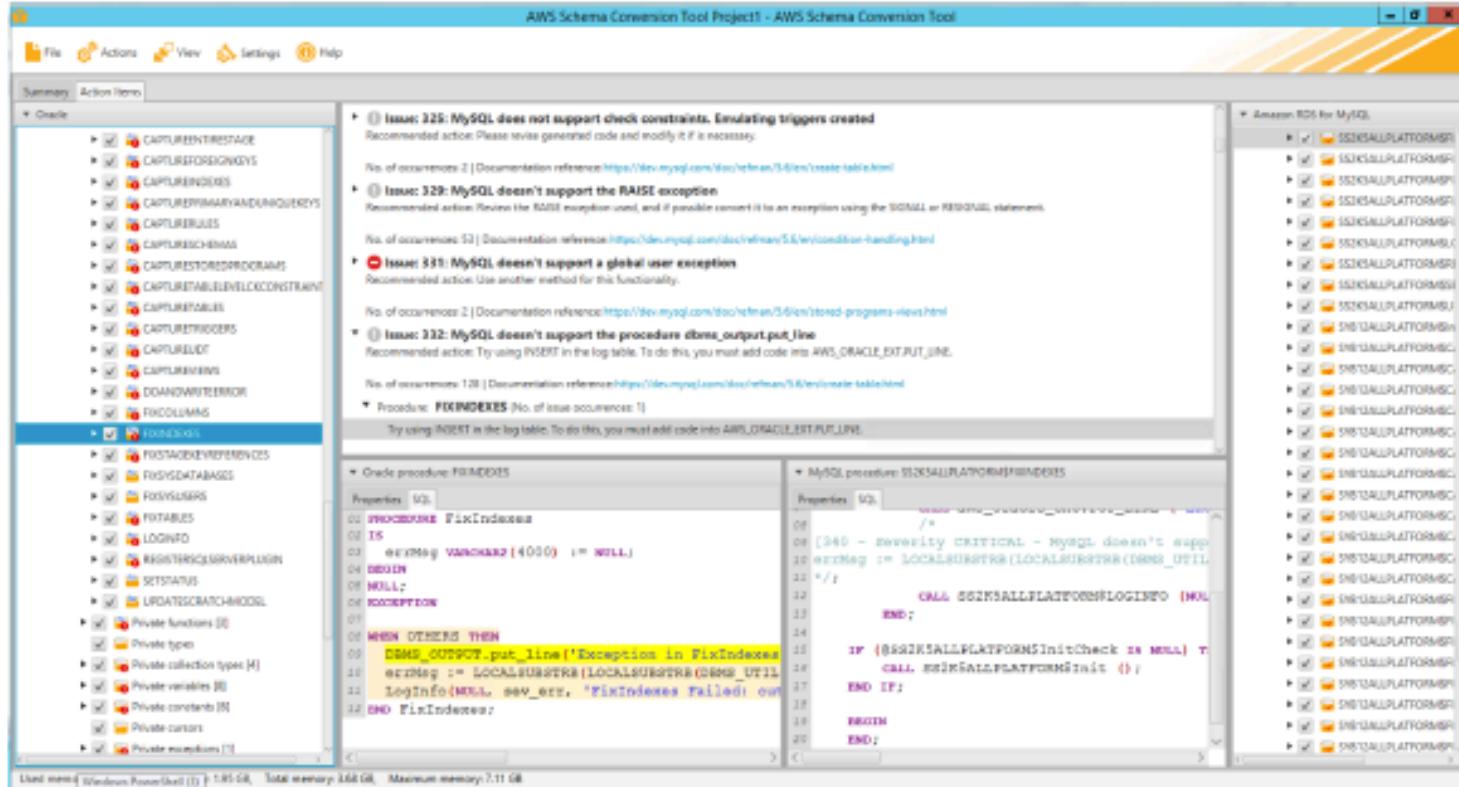
SCT Data Extraction Agents

Extract Data from your data warehouse and migrate to Amazon Redshift

- Extracts through local migration agents
- Data is optimized for Redshift and Saved in local files
- Files are loaded to an Amazon S3 bucket (through network or Amazon Snowball) and then to Amazon Redshift
 - Greenplum Database (version 4.3 and later)
 - Microsoft SQL Server (version 2008 and later)
 - Netezza (version 7.0.3 and later)
 - Oracle (version 10 and later)
 - Teradata (version 13 and later)
 - Vertica (version 7.2.2 and later)



SCT Helps with Converting Tables, Views & Code



Sequences
User-Defined Types
Synonyms
Packages
Stored Procedures
Functions
Triggers
Schemas
Tables
Indexes
Views
Sort and distribution keys

SCT can tell you how hard the migration will be

Database Migration Assessment Report



Executive Summary

We completed the analysis of your Oracle source database and estimate that 91% of the database storage objects and 100% of database code objects can be converted automatically or with minimal changes if you select Amazon Aurora as your migration target. Database storage objects include schemas, tables, columns, constraints, indexes, sequences, synonyms, user-defined types and functions. Database code objects include functions, procedures, packages, triggers, views, materialized views, events, SQL scalar functions, SQL inline functions, SQL table functions, attributes, variables, constants, table types, private types, cursors, exceptions, parameters and other objects. Based on our analysis of SQL system elements of your source database schema, we estimate that 99.9% of your entire database schema can be converted automatically to Amazon Aurora. To complete the migration, we recommend 397 conversion action(s) ranging from simple tasks to medium-complexity actions to significant conversion actions.

Database Objects with Conversion Actions for Amazon Aurora

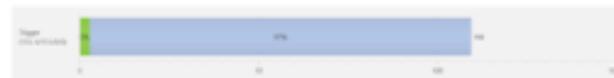
Of the total 1,576 database storage object(s) and 135 database code object(s) in the source database, we were able to identify 1,427 (91%) database storage object(s) and 135 (100%) database code objects that can be converted automatically or with minimal changes to Amazon Aurora.

149 (9%) database storage object(s) required 149 significant user action(s) to complete the conversion.

Figure: Conversion statistics for database storage objects



Figure: Conversion statistics for database code objects



Detailed Recommendations for Amazon Aurora Migrations

If you choose to migrate your Oracle database to Amazon Aurora, we recommend the following actions:

1. Connect SCT to Source and Target databases.

2. Run Assessment Report.

3. Read Executive Summary.

4. Follow detailed instructions.

Database Migration Assessment Report



Storage Object Actions

Sequence Changes

Some changes are required to sequences that cannot be converted automatically. You'll need to address these issues manually.

Issue 341: MySQL doesn't support sequences

Recommended Action: Try developing a system for sequences in your application.

Issue Code: 341 | No. of Occurrences: 134 | Estimated Complexity: Significant

Schemas.RDS_ADMINISTRATION.Sequences.BACKUP_ID_SEQUENCE
Schemas.RDS_ADMINISTRATION.Sequences.CERTIFICATE_ID_SEQUENCE
Schemas.RDS_ADMINISTRATION.Sequences.CHARACTER_SET_ID_SEQ
Schemas.RDS_ADMINISTRATION.Sequences.CUSTOMER_SUBNET_GROUP_ID_SEQ
Schemas.RDS_ADMINISTRATION.Sequences.CUSTOMER_SUBNET_ID_SEQ
0 issues

Index Changes

Some changes are required to indexes that cannot be converted automatically. You'll need to address these issues manually.

Issue 297: MySQL doesn't support function indexes

Recommended Action: Revise your code and try to use simple index.

Issue Code: 297 | No. of Occurrences: 3 | Estimated Complexity: Significant

Documentation Reference: <https://dev.mysql.com/doc/refman/5.6/en/create-table.html>

Schemas.RDS_ADMINISTRATION.Tables.DBL_ENGINE_THREADS.Indices.I_DBLENGO_NEED_DBLENGO_CONFIG_ID
Schemas.RDS_ADMINISTRATION.Tables.RDS_SYSTEM_ACCOUNTS.Indices.I_LVNL_ACCOUNT_DEFAULT
Schemas.RDS_ADMINISTRATION.Tables.RUNNABLE_DB_CONFIG.Indices.U_RNBLE_DBH_CFG_PREFERRED

Constraint Changes

Some changes are required to constraints that cannot be converted automatically. You'll need to address these issues manually.

Issue 210: MySQL doesn't support FUNCTION AS DEFAULT VALUE

Recommended Action: Try using a trigger.

Issue Code: 210 | No. of Occurrences: 1 | Estimated Complexity: Simple

Documentation Reference: <https://dev.mysql.com/doc/refman/5.6/en/create-table.html>

Schemas.RDS_ADMINISTRATION.Tables.CUSTOMERS.Constraints.CK_CUSTOMER_TRUST_LEVEL_STATE: 0/10
Schemas.RDS_ADMINISTRATION.Tables.STORAGE_VOLUMES.Constraints.CK_SV_LIFECYCLE: 0/8

Issue 325: MySQL does not support check constraints. Enabling triggers created

Recommended Action: Please revise generated code and modify it if necessary.

Issue Code: 325 | No. of Occurrences: 263 | Estimated Complexity: Simple

Documentation Reference: <https://dev.mysql.com/doc/refman/5.6/en/create-table.html>

Strengths and Focus Areas

DMS can act as a replication/migration Swiss Army knife, but is not a magic wand.



Use It

- Heterogeneous migrations
- Minimal downtime required
- No native solution

Caution

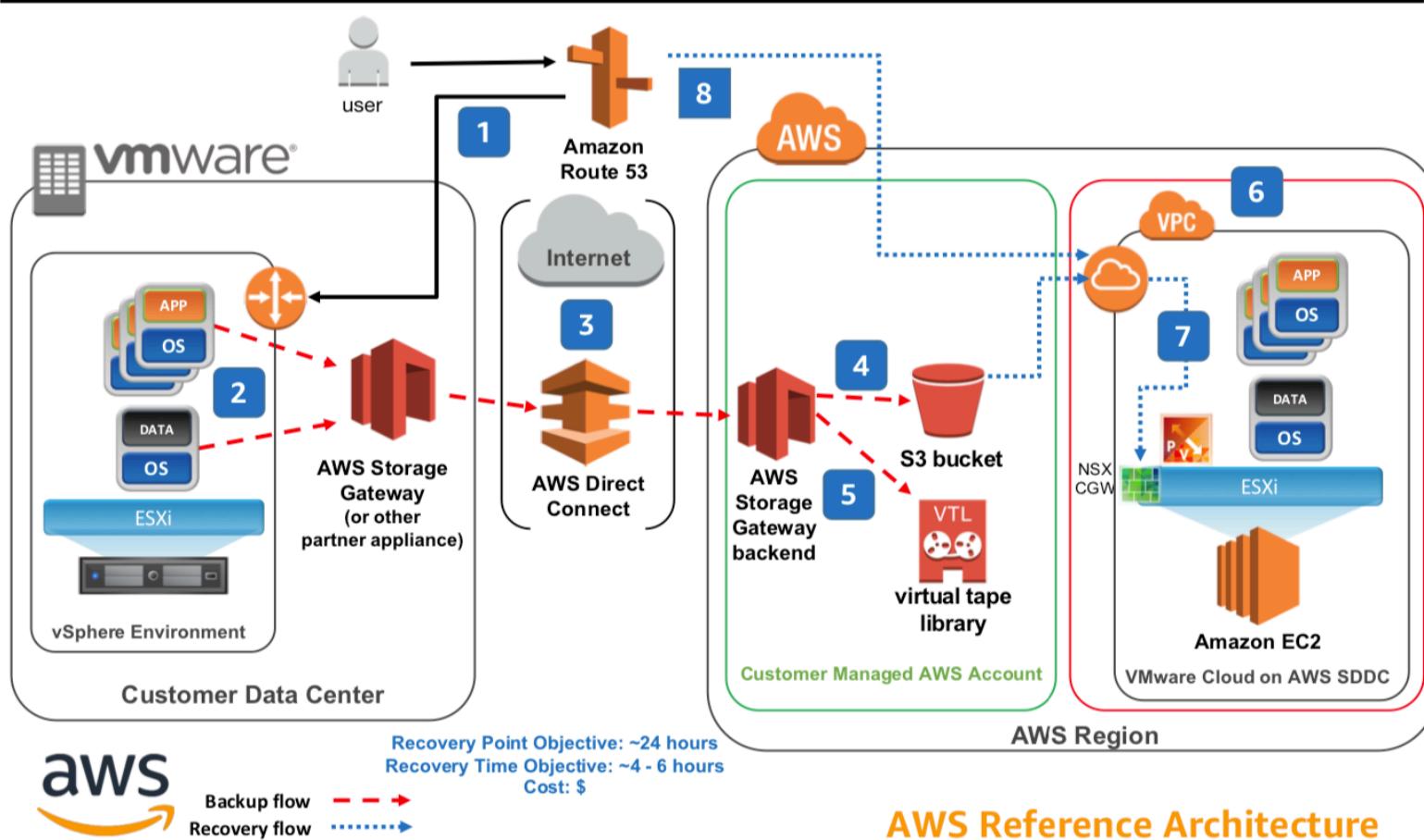
- Some tables with large LOBs
- Complex data types
- High load database

Don't Use It

- Native no downtime solution exists
- Can take downtime + native
- > 5 TB + slow Internet****

Backup and Restore to VMware Cloud on AWS

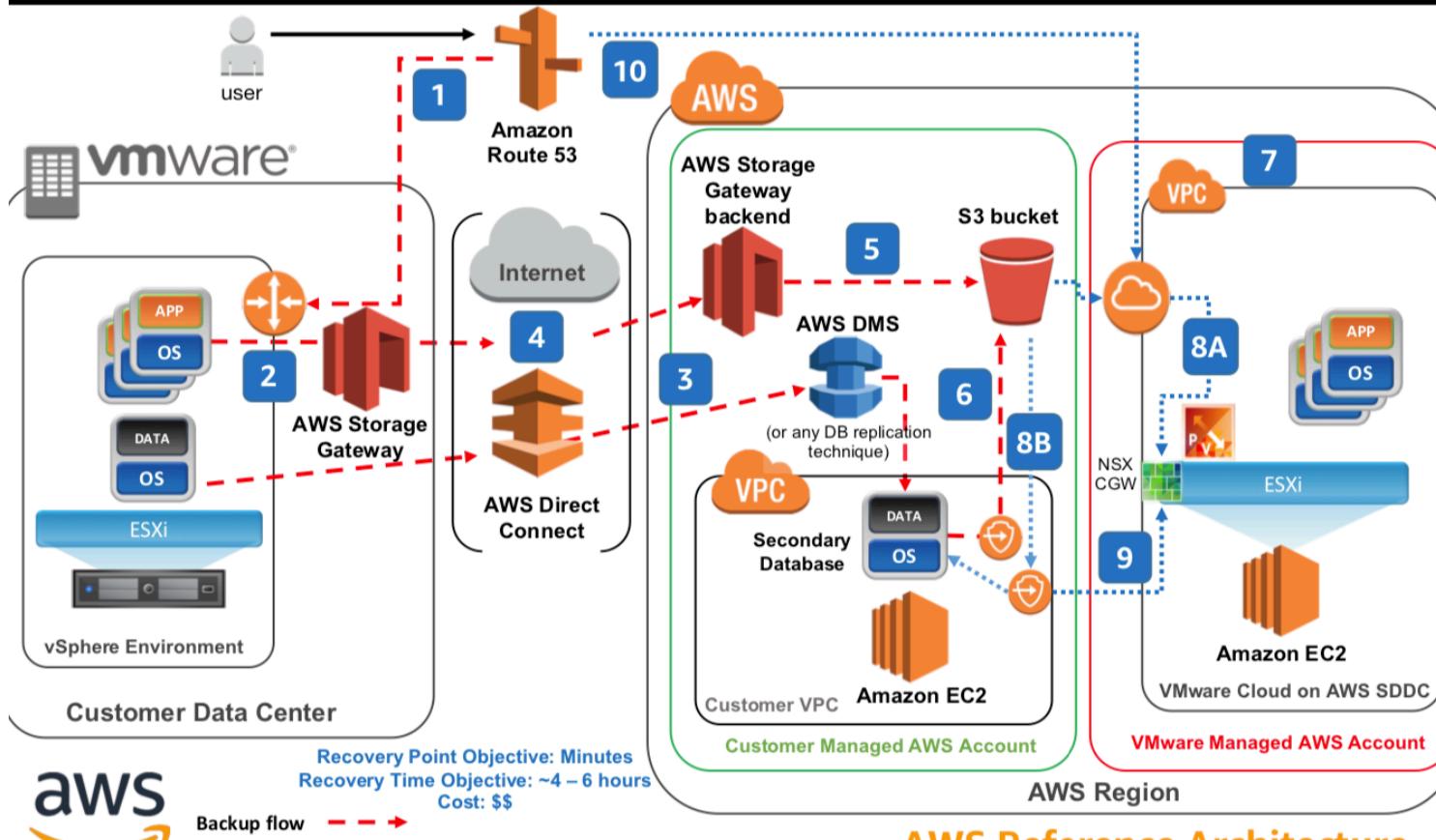
Native Services Integration: Storage Gateway, S3, Direct Connect, and Route53



- | # | Description |
|---|--|
| 1 | Amazon Route 53 routes DNS requests to the primary domain controller on-premises. |
| 2 | VM and application backups are stored in Amazon S3 using an AWS Storage Gateway or other storage appliance using a partner-integrated solution or application-level backup software. |
| 3 | The on-premises Storage Gateway securely transfers the backup data to the Storage Gateway backend using Direct Connect or through an SSL Internet connection. |
| 4 | File gateway uses an AWS Identity and Access Management role to access the customer backup data and securely store it in Amazon S3. |
| 5 | Use the virtual tape library configuration in the Storage Gateway for long term data archiving to AWS Glacier or other archive service. |
| 6 | The recovery process starts by launching and configuring a VMware SDDC cluster in AWS with the web portal or through automation scripts using AWS CloudFormation, VMware vRA, or vCLI. |
| 7 | After VMware Cloud on AWS SDDC is ready, deploy the software to restore the backed up application and VM data from Amazon S3. |
| 8 | The final recovery step is updating the Route 53 DNS records to route new requests to secondary domain controller in AWS. |

Pilot Light on VMware Cloud on AWS

Native Services Integration: Storage Gateway, EC2, S3, DMS, Direct Connect, and Route53

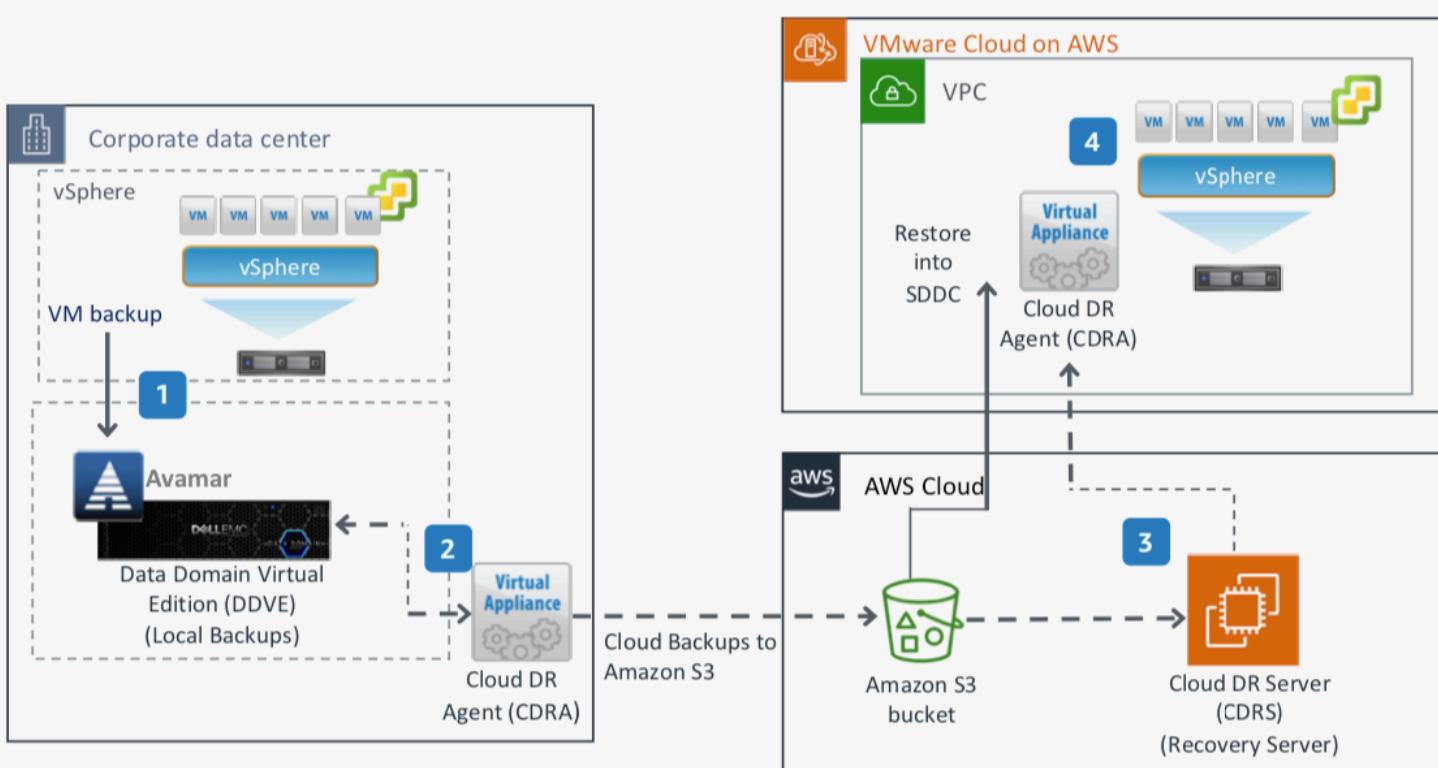


© 2018, Amazon Web Services, Inc. or its Affiliates. All rights reserved.

#	Description
1	Amazon Route 53 routes DNS requests to the primary domain controller at the customer data center.
2	VM and application backups are stored in Amazon S3 using an AWS Storage Gateway or another storage appliance or software backup solution.
3	AWS Database Migration Service (DMS) replicates data from primary database to secondary database in AWS.
4	Storage Gateway and DMS connect to the backend AWS services endpoints over Direct Connect or the Internet.
5	File gateway uses an AWS Identity and Access Management role to access the customer backup data and securely store it in Amazon S3.
6	Single point-in-time backups can be created on the secondary database using EBS snapshots stored in S3.
7	The recovery process starts by launching and configuring a VMware SDDC cluster in AWS with the web portal or through automation scripts using AWS CloudFormation, VMware vRA, or vCLI.
8	After VMware Cloud on AWS SDDC is ready, retrieve backed up data using (A) public S3 endpoint or (B) VMware endpoint using S3 VPC endpoint.
9	Recovered applications in VMware SDDC directly connect to the secondary database through VMware endpoints.
10	The final recovery step is updating the DNS records to route new requests to the secondary domain controller in AWS.

Data Domain Cloud Disaster Recovery for VMware Cloud on AWS

Fully Automated DR Solution for VMware Cloud on AWS with low RTO and RPO



1 Avamar performs regular backups of virtual machines (VMs) locally based on policies defined by administrators. If localized failures occur, VMs are restored from backups that are stored in Data Domain Virtual Edition (DDVE) storage.

2 Each local backup of VM is also duplicated in the AWS Cloud. Cloud DR Agent (CDRA) creates a secondary backup copy of the VM image in an Amazon S3 bucket owned by the customer.

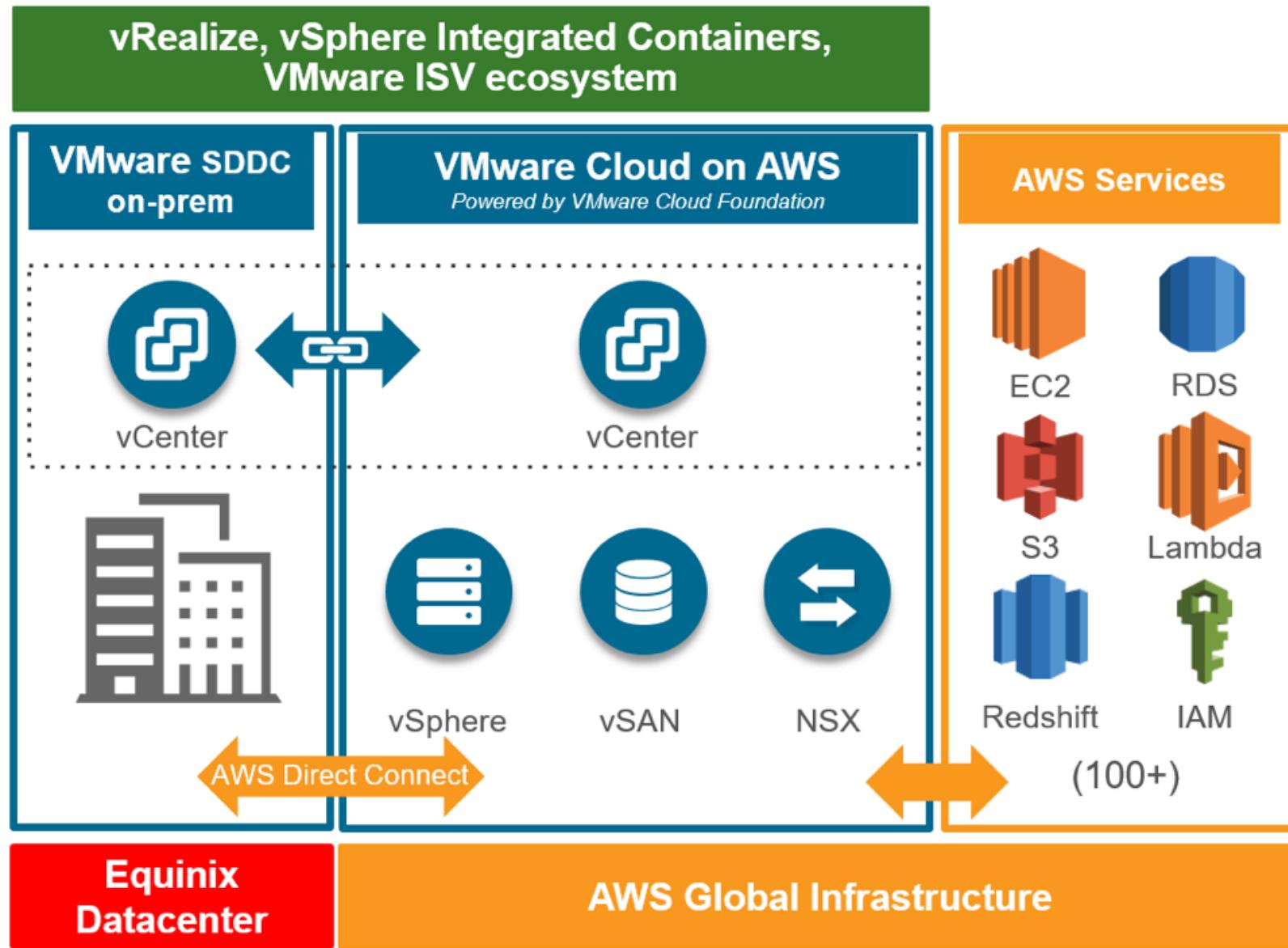
3 Cloud DR Server (CDRS) is the main backup and restore orchestrator. It runs on an Amazon EC2 instance in customer's AWS account and keeps track of all backup activity. In response to a DR scenario, CDRS pulls the backed up VM images from Amazon S3 and restores them in VMware Cloud on AWS software-defined data center (SDDC) based on predefined restore policies.

4 CDRS restores VMs to an existing VMware vCenter environment. The VMs resume normal operations immediately.

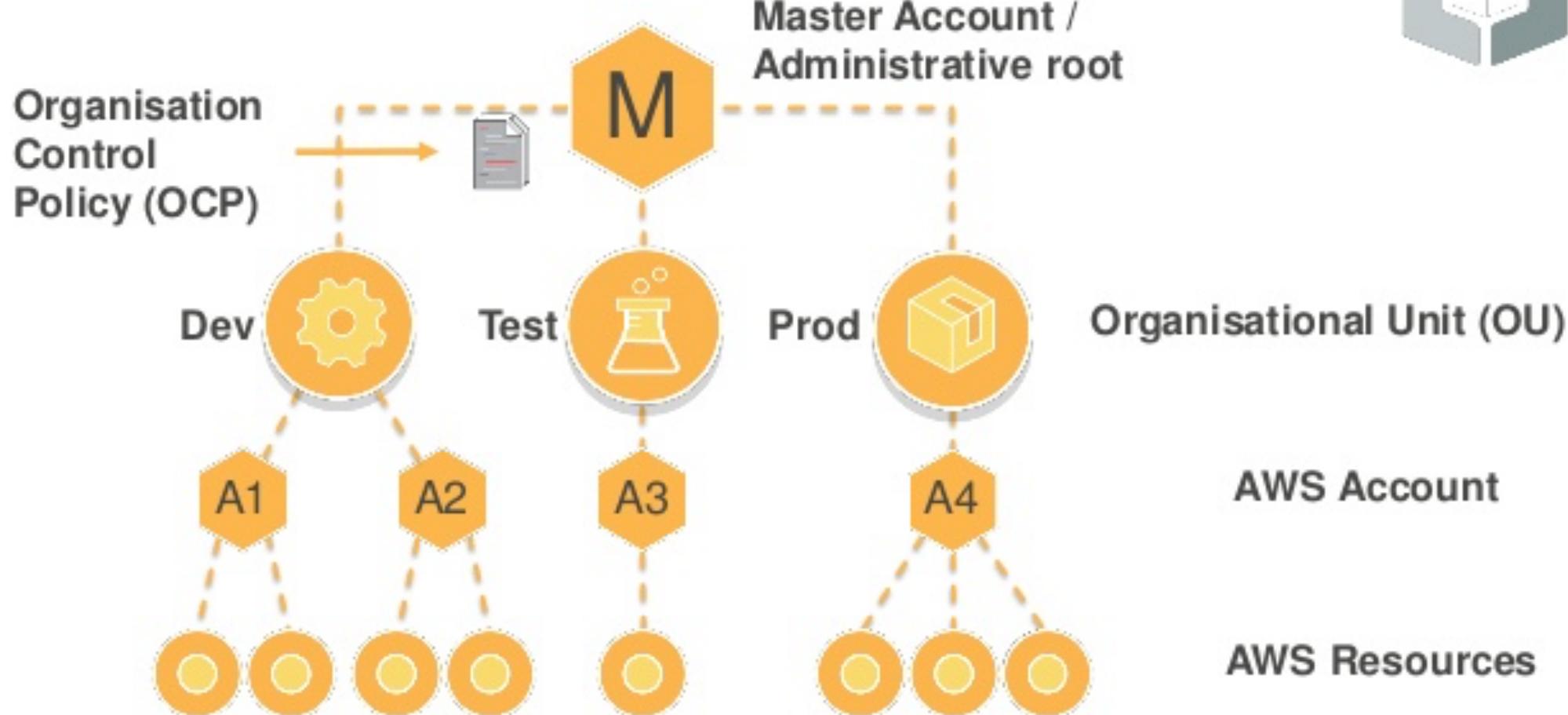


© 2019, Amazon Web Services, Inc. or its affiliates. All rights reserved.

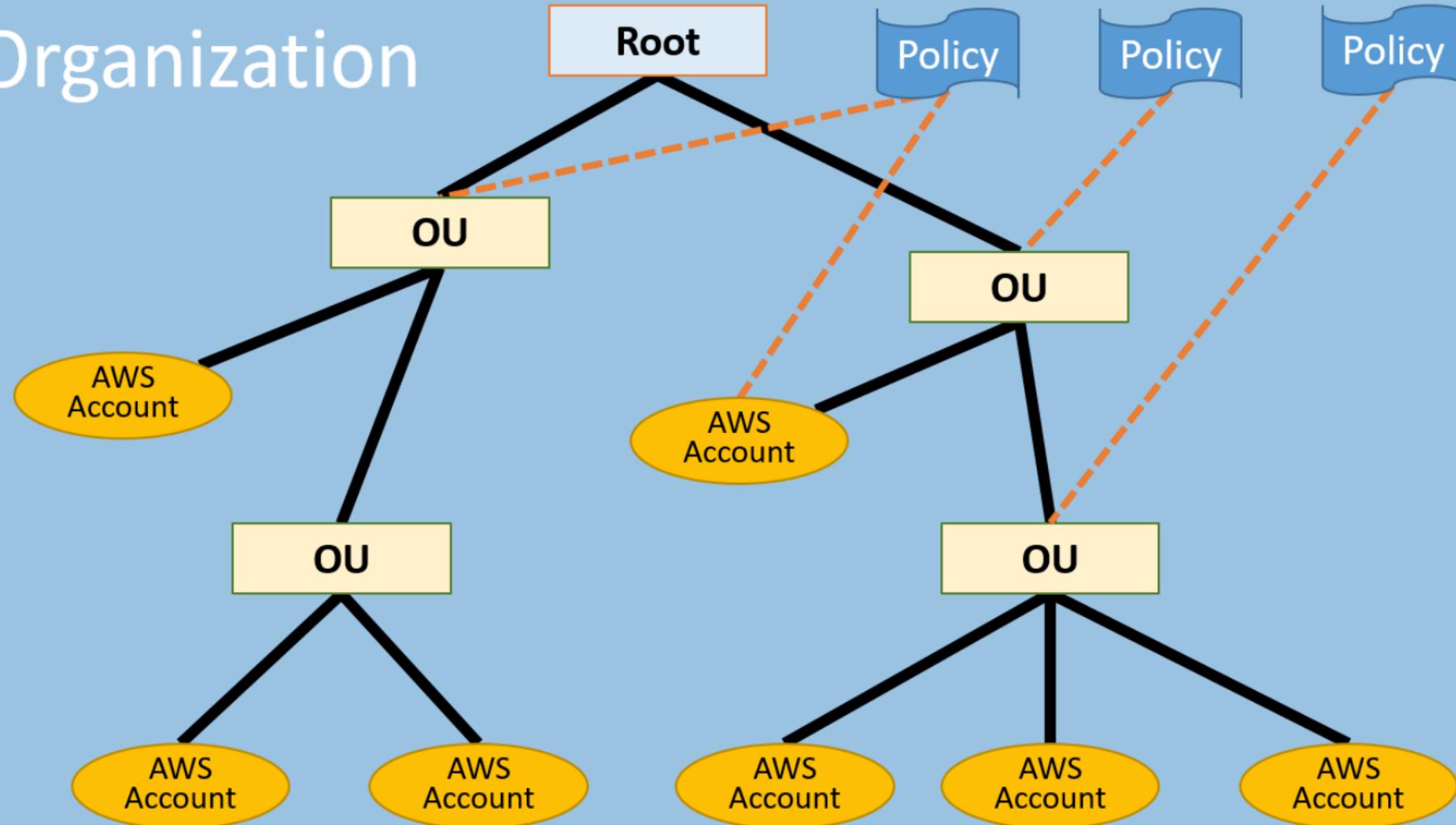
AWS Reference Architecture



AWS Organisations



Organization



AWS ORGANIZATIONS

AWS Organizations

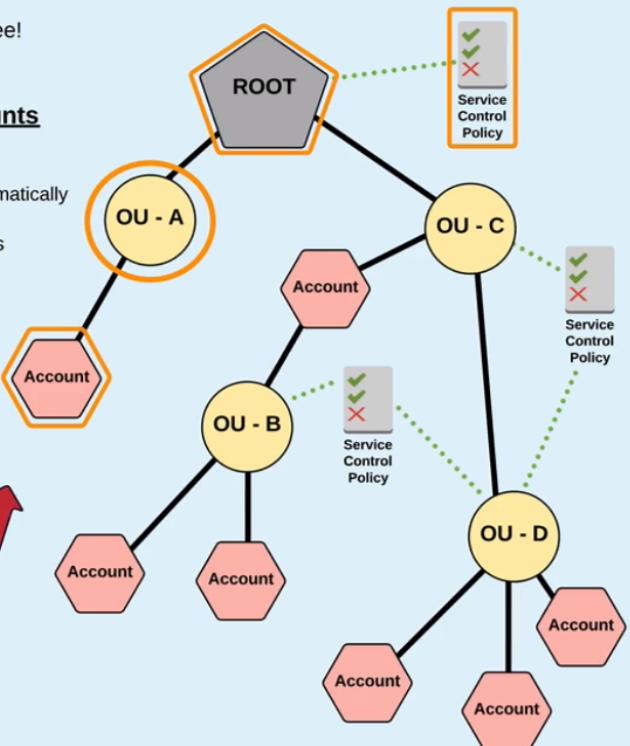
AWS Organizations is a service that lets you consolidate multiple AWS accounts into a single organization. This allows you to manage all accounts within the organization in one place.

AWS Organizations makes billing and permissions easier and allows for the creation of managed accounts. Accounts are organized hierarchically, which provides better security and compliance controls.

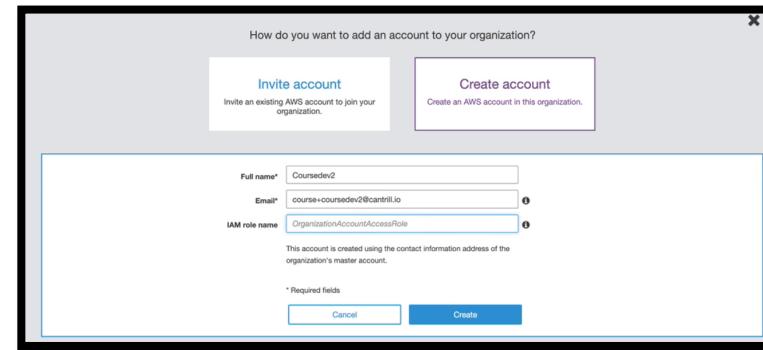
Best of all, this service is free!

Manage Multiple Accounts

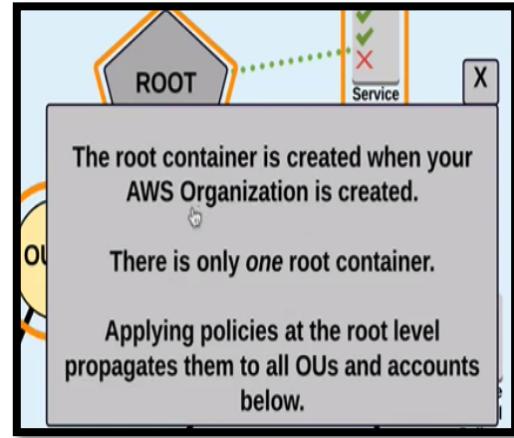
- IAM policy management
 - Account groups
 - Create accounts programmatically
- Consolidated billing
 - Manage payment methods



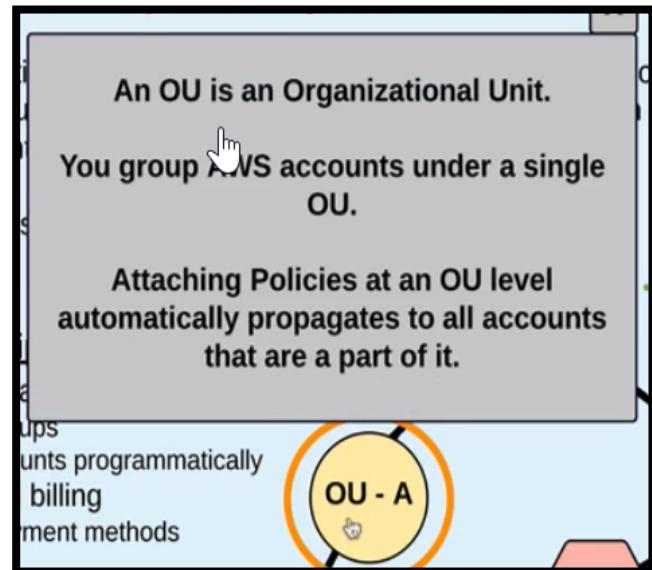
Organizations operate in either "Consolidated Billing" or "All Features" mode. This can be modified at any time.



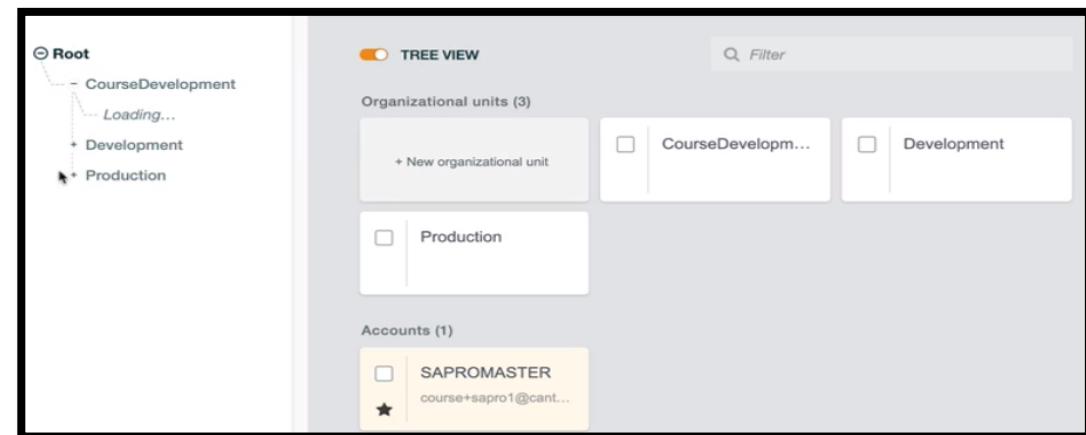
- Max 20 accounts (Only 1 master)



ROOT : is an account management service that allows you to consolidate multiple AWS accounts into an organization, enabling you to create a hierarchical structure that can be managed centrally.



- Nested under Root Account



AWS Organization Features

Centralized management of all of your AWS accounts

- Combine existing accounts into or create new ones within an organization that enables them to be managed centrally
- Policies can be attached to accounts that affect some or all of the accounts

Consolidated billing for all member accounts

- Consolidated billing is a feature of AWS Organizations.
- Master account of the organization can be used to consolidate and pay for all member accounts.

Hierarchical grouping of accounts to meet budgetary, security, or compliance needs

- Accounts can be grouped into organizational units (OUs) and each OU can be attached different access policies.
- OUs can also be nested to a depth of five levels, providing flexibility in how you structure your account groups.

Control over AWS services and API actions that each account can access

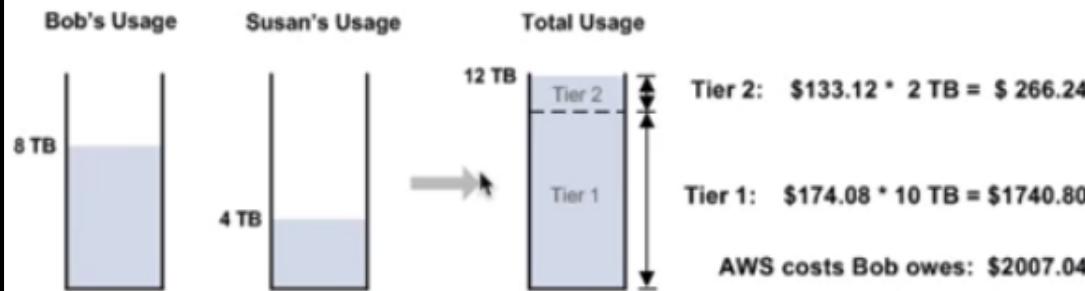
- As an administrator of the master account of an organization, access to users and roles in each member account can be restricted to which AWS services and individual API actions
- Organization permissions overrule account permissions.
- This restriction even overrides the administrators of member accounts in the organization.
- When AWS Organizations blocks access to a service or API action for a member account, a user or role in that account can't access any prohibited service or API action, even if an administrator of a member account explicitly grants such permissions in an IAM policy.

Consolidated Billing

- Default mode
- Only Consolidated billing features: This mode only provides the consolidated billing features
(End of the month report)

For the purposes of this example, AWS charges \$0.17 per GB for the first 10 TB of data transferred and \$0.13 for the next 40 TB. This translates into \$174.08 per TB ($= .17 * 1024$) for the first 10 TB, and \$133.12 per TB ($= .13 * 1024$) for the next 40 TB. Remember that 1 TB = 1024 GB.

For the 12 TB that Bob and Susan used, Bob's payer account is charged $(\$174.08 * 10 \text{ TB}) + (\$133.12 * 2 \text{ TB}) = \$1740.80 + \$266.24 = \$2,007.04$.



The average cost-per-unit of data transfer out for the month is therefore $\$2,007.04 / 12 \text{ TB} = \167.25 per TB . That is the average tiered rate that is shown on the Bills page and in the downloadable cost report for each linked account on the consolidated bill.

Without the benefit of tiering across the consolidated bill, AWS would have charged Bob and Susan each \$174.08 per TB for their usage, for a total of \$2,088.96.



EXAM TIP: Consolidated Billing allows you to get volume discounts on all your accounts. When consolidated billing is enabled, unused reserved instances for EC2 are applied across the group.

La facturación consolidada tiene las siguientes ventajas:

- **Una factura:** obtendrá una factura para varias cuentas. –
- **Seguimiento sencillo** – puede hacer un seguimiento de los cargos de varias cuentas y descargar los datos de uso y costo combinado.
- **Uso combinado** – Puede combinar el uso en todas las cuentas de la organización compartir los descuentos por volumen y los descuentos de instancias reservadas. Esto puede dar lugar a un cargo menor para su proyecto, departamento o empresa que con las cuentas independientes individuales. Para obtener más información, consulte [Descuentos por volumen](#).
- **Sin cargos adicionales:** la facturación unificada se ofrece sin costo adicional.

Etiquetas de asignación de costos



Etiquetas de asignación de costos generadas por AWS

Un *recurso creado por la etiqueta* es una etiqueta de asignación de costos generada por AWS que contiene información sobre el creador del recurso que se aplica automáticamente a los recursos que crea usted. Esta característica solo se encuentra disponible en la consola de administración de facturación y costos, y no aparecerá en ningún otro lugar en la consola de AWS, ni en el Editor de etiquetas.

Activar

Etiquetas de asignación de costos definidas por el usuario

- ✓ Carga de etiquetas finalizada

La activación de etiquetas para la asignación de costos indica a AWS que los datos de costo asociados para estas etiquetas deben estar disponibles en la línea de facturación. Una vez activadas, las etiquetas de asignación de costos se pueden utilizar como una dimensión de agrupación y filtro en el explorador de costos y para refinar los criterios del presupuesto de AWS.

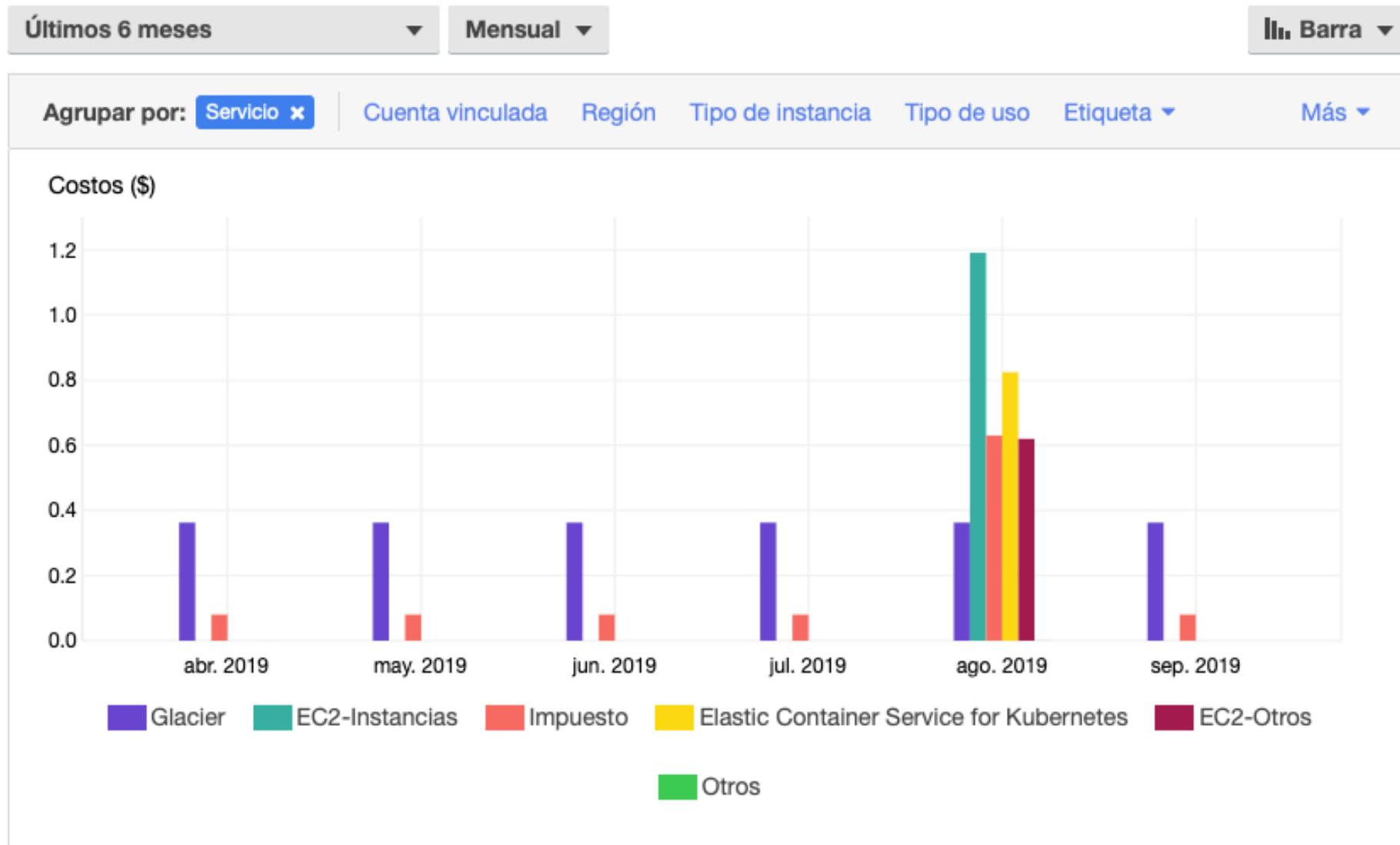
Activar

Desactivar

Deshacer

Filtro:		Todas las etiquetas*	Buscar una clave de etiqueta...	Etiquetas por página:	100
<input checked="" type="checkbox"/>	Clave de etiqueta*				
<input type="checkbox"/>	aws:cloud9:environment				Active
<input type="checkbox"/>	aws:cloud9:owner				Active
<input type="checkbox"/>	aws:cloudformation:logical-id				Active
<input type="checkbox"/>	aws:cloudformation:stack-id				Active
<input type="checkbox"/>	aws:cloudformation:stack-name				Active
<input type="checkbox"/>	aws:cloudformation:stack-status				Active

Monthly costs by service



Amazon CloudFront

Red de entrega de contenido (CDN) segura, rápida y programable

- Amazon CloudFront es un servicio rápido de red de entrega de contenido (CDN) que distribuye datos, vídeos, aplicaciones y API a clientes de todo el mundo de forma segura, con baja latencia, altas velocidades de transferencia y dentro de un entorno fácil para desarrolladores. CloudFront está integrado a AWS tanto mediante ubicaciones físicas conectadas directamente con la infraestructura global de AWS, así como otros servicios de AWS. CloudFront funciona de forma fluida con servicios como AWS Shield para mitigar ataques DDoS, Amazon S3, Elastic Load Balancing o Amazon EC2 como orígenes para sus aplicaciones y utilizando Lambda@Edge para ejecutar código personalizado más cerca de los usuarios de los clientes y personalizar así la experiencia de usuario. Por último, si usa orígenes de AWS, como Amazon S3, Amazon EC2 o Elastic Load Balancing, no tiene que pagar por ningún dato transferido entre estos servicios y CloudFront.
- Puede comenzar a usar la red de entrega de contenido (CDN) en cuestión de minutos con las mismas herramientas de AWS con las que ya está familiarizado: las API, la consola de administración de AWS, AWS CloudFormation, las CLI y los SDK. La CDN de Amazon ofrece un modelo de pago por uso sencillo sin tarifas anticipadas ni contratos a largo plazo. Además, su suscripción a AWS Support existente incluye el soporte para la CDN.

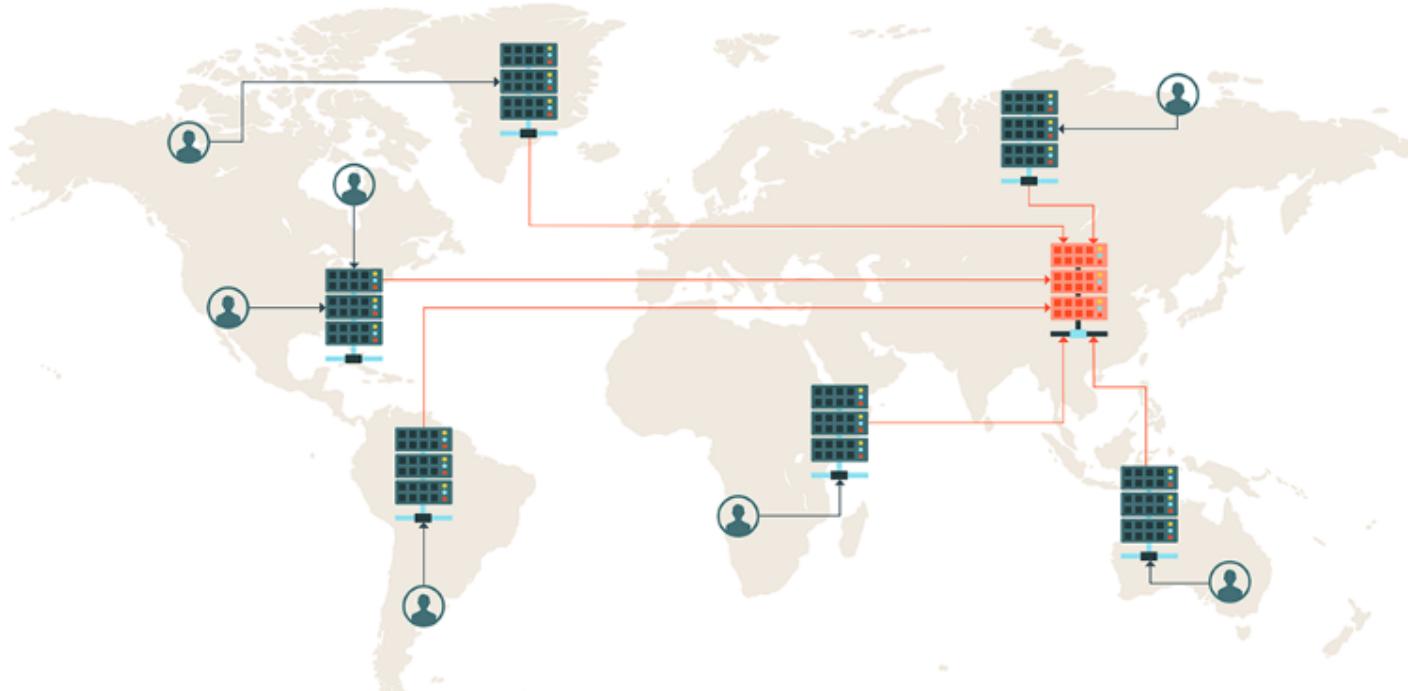
CloudFront (CDN)

- A **Content Delivery Network (CDN)** is a system of distributed servers (network) that deliver pages and other Web content to a user, based on the geographic locations of the user, the origin of the webpage and the content delivery server.
- Amazon CloudFront is a web service that gives businesses and web application developers an easy and cost effective way **to distribute content with low latency and high data transfer speeds**.
- Like other AWS services, Amazon CloudFront is a self-service, pay-per-use offering, requiring no long term commitments or minimum fees. With CloudFront, your files are delivered to end-users using a global network of **edge locations**.

Content delivery network

CDNs cache content from the origin server on geographically distributed CDN cache servers to reach users faster.

● USER ■ CDN SERVER □ ORIGIN SERVER



CloudFront (CDN)

How Can Amazon CloudFront Help?

Content Delivery Network (CDN)

- Your content is cached all around the world.
- Geographic proximity means lower latency.
- Better user experience.
- Less stress on your core infrastructure.

Key Features

- TCP/IP optimizations for the network path.
- Keep-alive connections to reduce round-trip time.
- SSL/TLS termination close to viewers.
- POST/PUT upload optimizations.
- Latency-based routing.



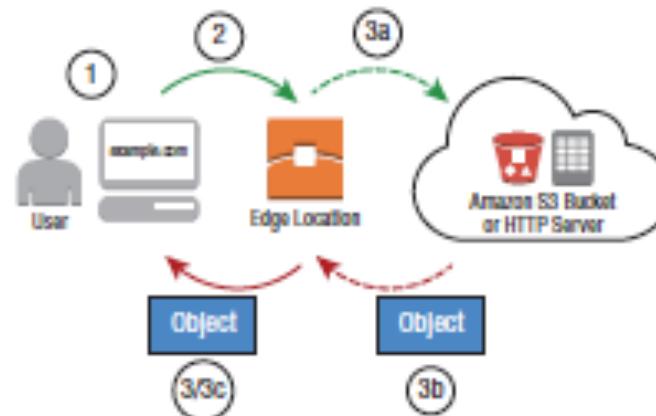
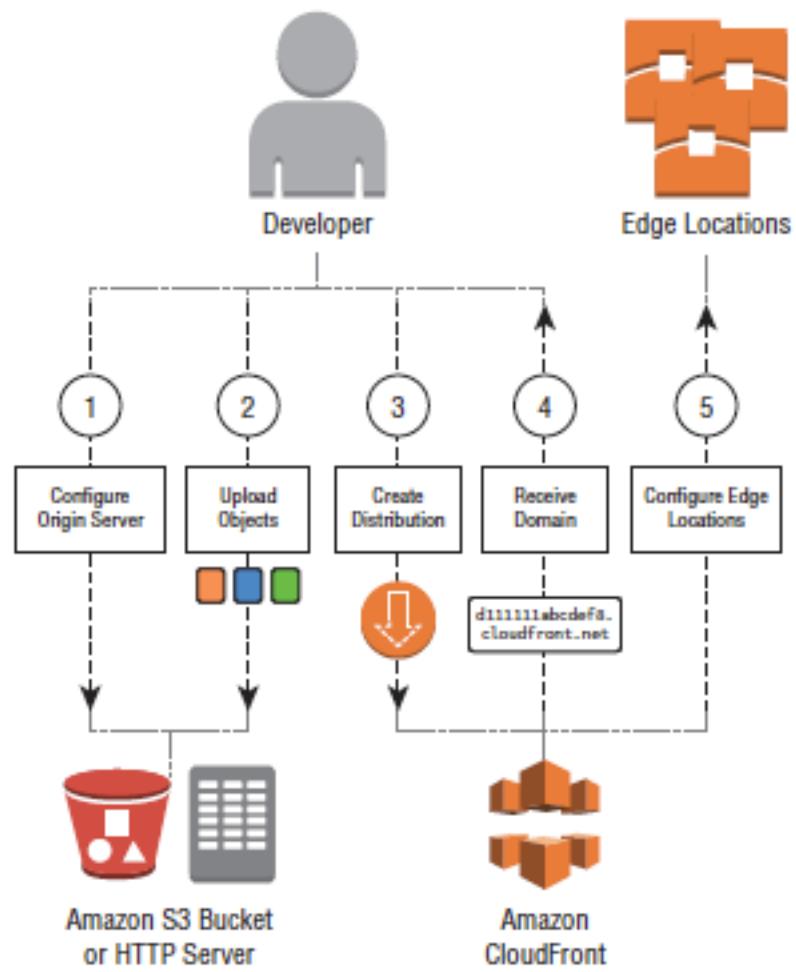
Independent of Region

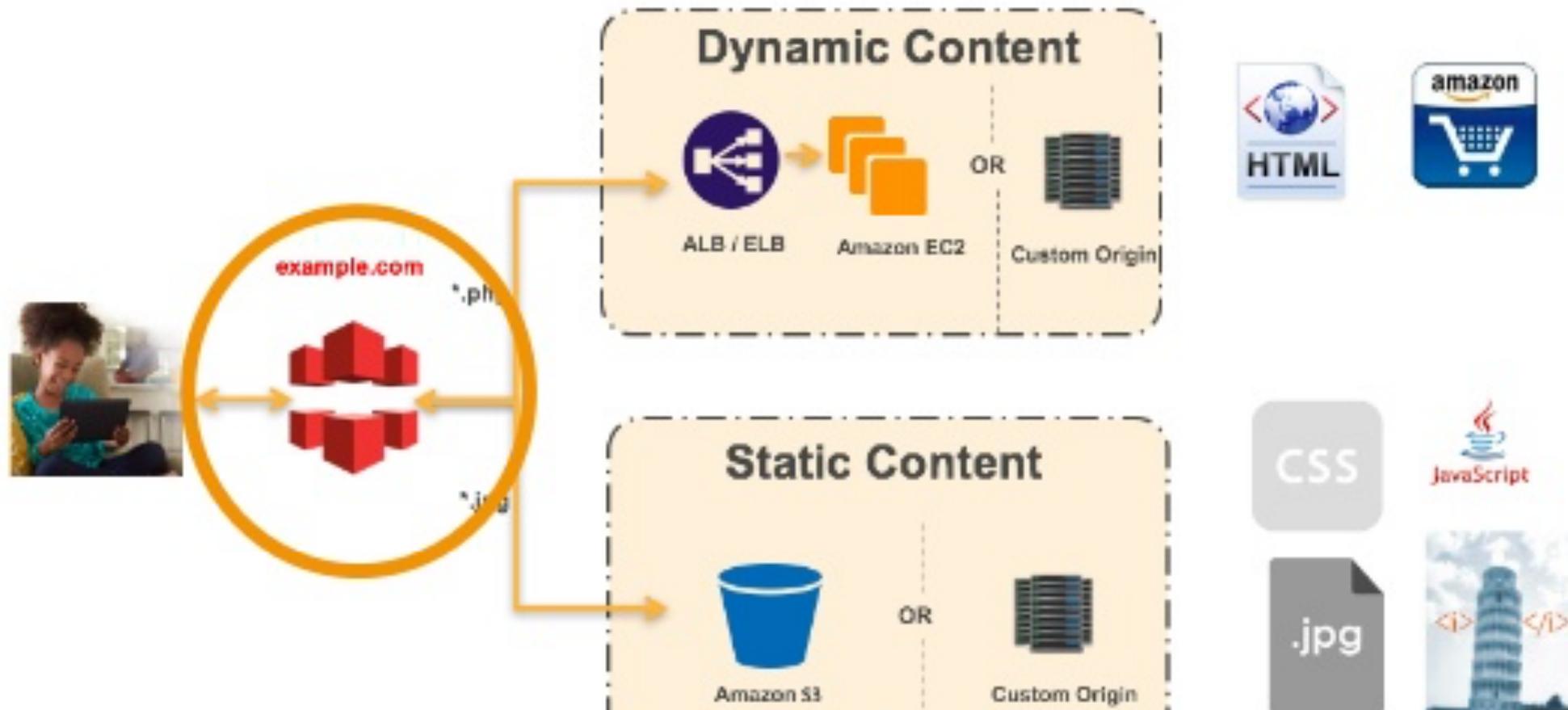
CloudFront (II)

- Amazon CloudFront supports all files that can be served over HTTP. This includes dynamic web pages, such as HTML or PHP pages, any popular static files that are a part of your web application, such as website images, audio, video, media files or software downloads. Amazon CloudFront also supports delivery of live or on-demand media streaming over HTTP.
- Edge locations are not just READ only, you can write to them too (upload objects)
- Objects are cached for the life of the TTL (Time To Live)
- You can clear cached objects, but you will be charged.

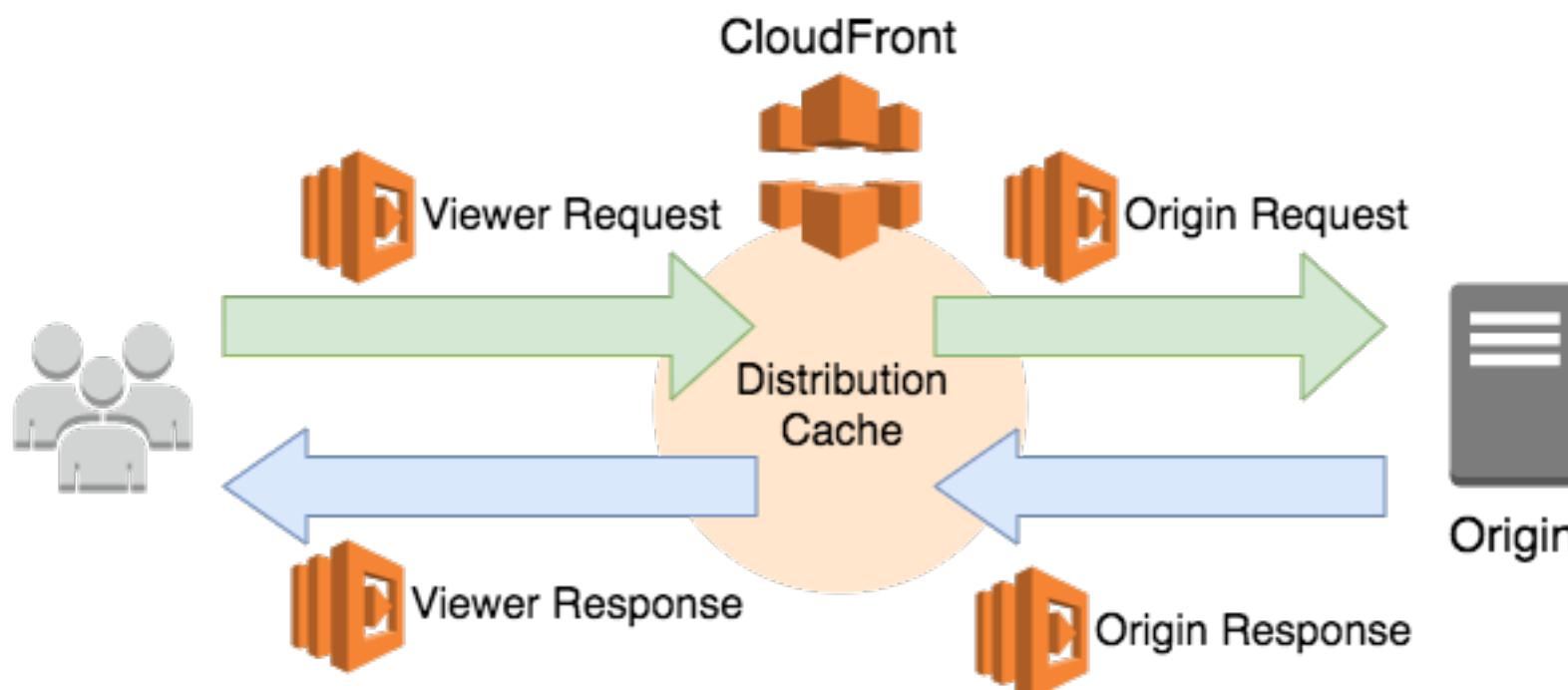
CloudFront – Key Terminology

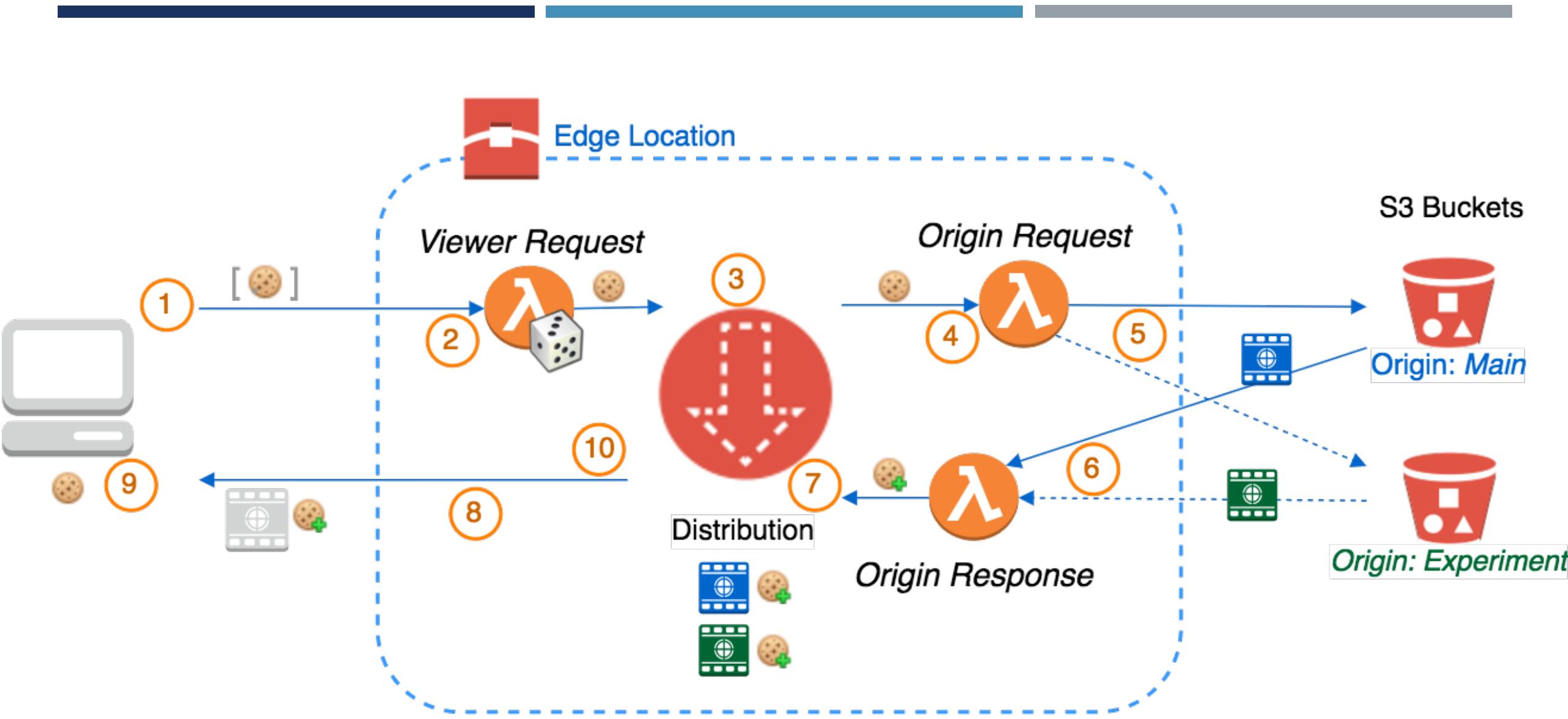
- **Edge Location** – This is the location where content will be cached. This is separate to an AWS Region/AZ . There are 125 Edge Locations and 11 Regional Edge Caches.
- **Origin** – This is the origin of all the files that the CDN will distribute. This can be either an S3 bucket, an EC2 Instance, an Elastic Load Balancer or Route 53. Amazon CloudFront also works seamlessly with any non-AWS origin server. New services added: Media Package and AWS Elemental MediaStore
- **Distribution** – This is the name given the CDN which consists of a collection of Edge Locations.
 - **Web Distribution** – Typically used for Websites
 - **RTMP** – Used for Media Streaming (Adobe Flash Media)





Lambda@Edge allows running Lambda functions at Edge Locations of the CloudFront CDN. It means you may add “intelligence” in the CDN, without having to forward the request to a backend and losing benefits of content caching and geographical proximity with the client.





Kinesis Video Streams

Registre, procese y almacene transmisiones de videos

Amazon Kinesis Video Streams facilita la transmisión segura de videos desde dispositivos conectados a AWS para realizar análisis, aprendizaje automático y otros procesos.

[Más información »](#)

Kinesis Data Streams

Registre, procese y almacene transmisiones de datos

Amazon Kinesis Data Streams es un servicio de streaming de datos en tiempo real que puede registrar de manera continua gigabytes de datos por segundo de cientos de miles de orígenes.

[Más información »](#)

Kinesis Data Firehose

Cargue transmisiones de datos en almacenes de datos de AWS

Amazon Kinesis Data Firehose es la manera más fácil de registrar, transformar y cargar transmisiones de datos en almacenes de datos de AWS para realizar análisis casi en tiempo real con herramientas de inteligencia empresarial existentes.

Kinesis Data Analytics

Analice transmisiones de datos con SQL o Java

Amazon Kinesis Data Analytics ofrece la manera más sencilla de procesar transmisiones de datos en tiempo real con SQL o Java sin tener que aprender a usar lenguajes de programación ni marcos de procesamiento nuevos.

[Más información »](#)



Input

Camera devices securely stream video to AWS using the Kinesis Video Streams SDK



Kinesis Video Streams

Ingests, durably stores, encrypts, and indexes video streams for real-time and batch analytics



Amazon Rekognition Video



Amazon SageMaker

MxNet / TensorFlow

HLS-based video playback

Custom video processing

Output

Real-time and batch-driven machine learning applications, video processing and playback services use Kinesis Video Streams API to access and retrieve indexed video



Input

Capture and send data to Amazon Kinesis Data Streams



Apache Spark
Spark on EMR

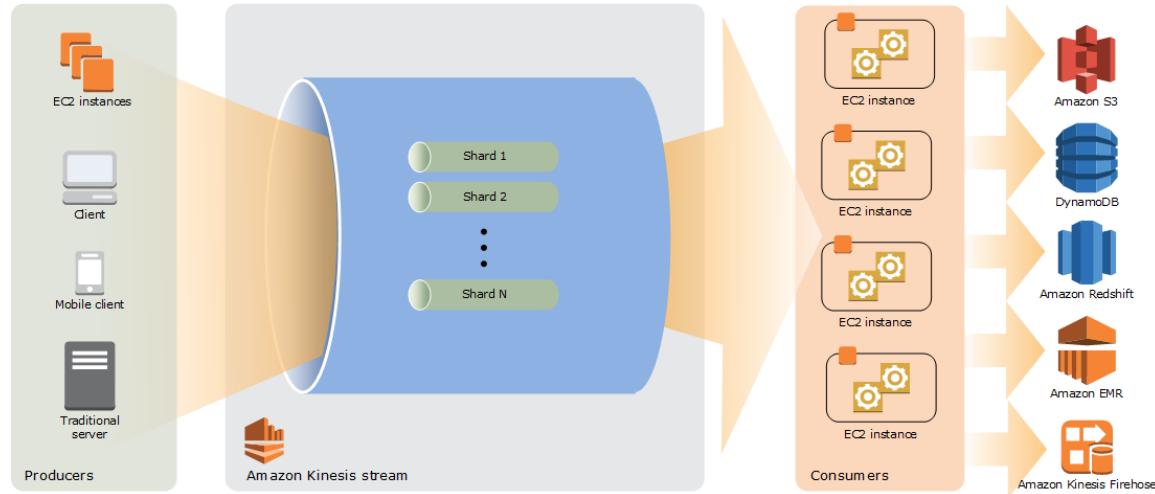


Build custom, real-time applications using Amazon Kinesis Data Analytics, stream processing frameworks like Apache Spark, or your code running Amazon EC2 or AWS Lambda



Output

Analyze streaming data using your favorite BI tools



Arquitectura de alto nivel de Kinesis Data Streams

El siguiente diagrama ilustra la arquitectura de alto nivel de Kinesis Data Streams. Los *productores* envían datos continuamente a Kinesis Data Streams, mientras que los *consumidores* los procesan en tiempo real. Los consumidores (por ejemplo una aplicación personalizada que se ejecute en Amazon EC2 o una secuencia de entrega de Amazon Kinesis Data Firehose) pueden almacenar sus resultados mediante un servicio de AWS como Amazon DynamoDB, Amazon Redshift o Amazon S3.

Terminología de Kinesis Data Streams

Kinesis Data Stream

Una *Kinesis data stream* es un conjunto de [fragmentos](#). Cada fragmento tiene una secuencia de registros de datos. A su vez, cada registro de datos contiene un [número secuencial](#) que asigna Kinesis Data Streams.

Registro de datos

Un *registro de datos* es la unidad de datos que se almacena en una [Kinesis data stream](#). Los registros de datos se componen de un [número secuencial](#), una [clave de partición](#) y un blob de datos, que es una secuencia inmutable de bytes. Kinesis Data Streams no inspecciona, interpreta ni modifica los datos del blob de ninguna forma. Un blob de datos puede ser hasta 1 MB.

Periodo de retención

El *periodo de retención* es el tiempo durante el cual se puede obtener acceso a los registros de datos después de que se agreguen a la secuencia. El periodo de retención de una secuencia se establece en un valor predeterminado de 24 horas tras su creación. Puede aumentar el periodo de retención hasta 168 horas (7 días) mediante la operación [IncreaseStreamRetentionPeriod](#) y reducirlo hasta un mínimo de 24 horas con la operación [DecreaseStreamRetentionPeriod](#). Para las secuencias con periodos de retención superiores a 24 horas se aplican cargos adicionales. Para obtener más información, consulte [Precios de Amazon Kinesis Data Streams](#).

Productor

Los *productores* insertan registros en Amazon Kinesis Data Streams. Por ejemplo, un servidor web que envía datos de registro a una secuencia es un productor.

Consumidor

Los *consumidores* obtienen registros de Amazon Kinesis Data Streams y los procesan. Estos consumidores se denominan Amazon Kinesis Data Streams Application.

Amazon Kinesis Data Streams Application

Una *Amazon Kinesis Data Streams application* es un consumidor de una secuencia que normalmente se ejecuta en una flota de instancias EC2.

Es posible desarrollar dos tipos de consumidores: consumidores de distribución ramificada compartida y consumidores de distribución ramificada mejorada. Para conocer las diferencias entre ellos y saber cómo crear cada tipo de consumidor, consulte Lectura de datos desde Amazon Kinesis Data Streams.

La salida de una aplicación de Kinesis Data Streams se puede utilizar como entrada para otra secuencia, lo que le permitirá crear topologías complejas que procesan datos en tiempo real. Una aplicación también pueden enviar datos a una serie de otros servicios de AWS. Puede haber varias aplicaciones para una secuencia, y cada aplicación puede consumir datos procedentes de la secuencia de forma independiente y de forma simultánea.

Clave de partición

Una *clave de partición* se utiliza para agrupar datos por fragmento dentro de una secuencia. Kinesis Data Streams separa los registros de datos que pertenecen a una secuencia en varios fragmentos. Utiliza la clave de partición asociada a cada registro de datos para determinar a qué fragmento pertenece un registro de datos determinado. Las claves de partición son cadenas en formato Unicode con una longitud máxima de 256 bytes. Se utiliza una función MD5 hash para asignar claves de partición a valores enteros de 128 bits y para asignar los registros de datos asociados a fragmentos. Cuando una aplicación pone los datos en una secuencia, debe especificarse una clave de partición.

Sequence Number

Cada registro de datos tiene un *número secuencial* único por clave de partición en su fragmento. Kinesis Data Streams asigna el número secuencial después de escribir en la secuencia con `client.putRecords` o `client.putRecord`. Por lo general, los números secuenciales de una misma clave de partición aumentan con el paso del tiempo. Cuanto más largo sea el período de tiempo entre las solicitudes de escritura, mayores serán los números secuenciales.

nota

Los números secuenciales no se pueden utilizar como índices de conjuntos de datos dentro de la misma secuencia. Para separar lógicamente conjuntos de datos, utilice claves de partición o cree una secuencia independiente para cada conjunto de datos.

Kinesis Client Library

La Kinesis Client Library está compilada en su aplicación para permitir un consumo de datos tolerante a errores desde la secuencia. La Kinesis Client Library garantiza que para cada fragmento haya un procesador de registros en ejecución y procesando dicho fragmento. La biblioteca también simplifica la lectura de los datos desde la secuencia. Kinesis Client Library usa una tabla de Amazon DynamoDB para almacenar los datos de control. Crea una tabla por cada aplicación que procesa datos.

Existen dos versiones principales de la Kinesis Client Library. Deberá elegir una u otra en función del tipo de consumidor que desee crear. Para obtener más información, consulte [Lectura de datos desde Amazon Kinesis Data Streams](#).

Nombre de la aplicación

El nombre de una Amazon Kinesis Data Streams application identifica la aplicación. Cada aplicación debe tener un nombre único dentro de la cuenta y la región de AWS que utiliza la aplicación. Este nombre se usa como denominación para la tabla de control en Amazon DynamoDB y para el espacio de nombres de las métricas de Amazon CloudWatch.

Cifrado en el servidor

Amazon Kinesis Data Streams puede cifrar la información confidencial automáticamente cuando un productor la introduce en una secuencia. Kinesis Data Streams usa claves maestras de [AWS KMS](#) para el cifrado. Para obtener más información, consulte [Protección de los datos en Amazon Kinesis Data Streams](#).



Input

Capture and send data to Amazon Kinesis Data Firehose



Amazon Kinesis Data Firehose

Prepares and loads the data continuously to the destinations you choose



Amazon S3



Amazon Redshift



Amazon Elasticsearch Service



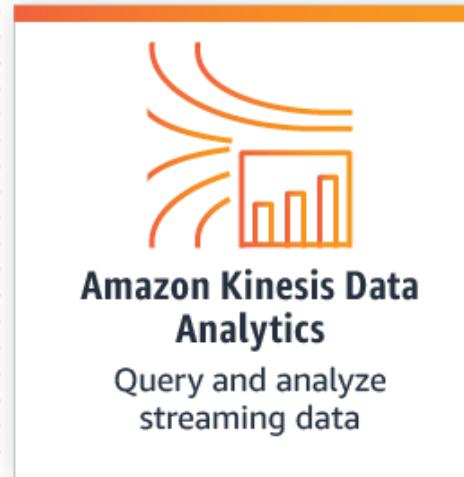
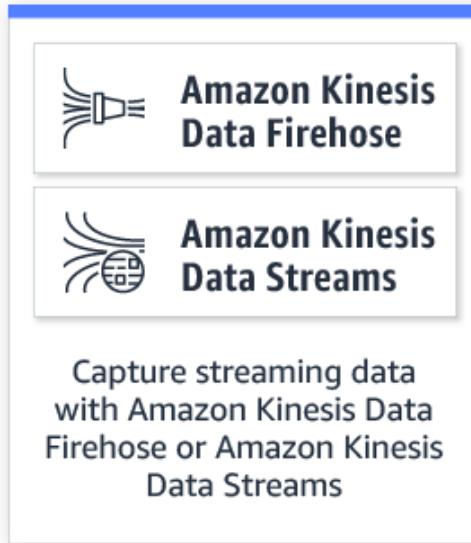
Splunk

Durably store the data for analytics



Output

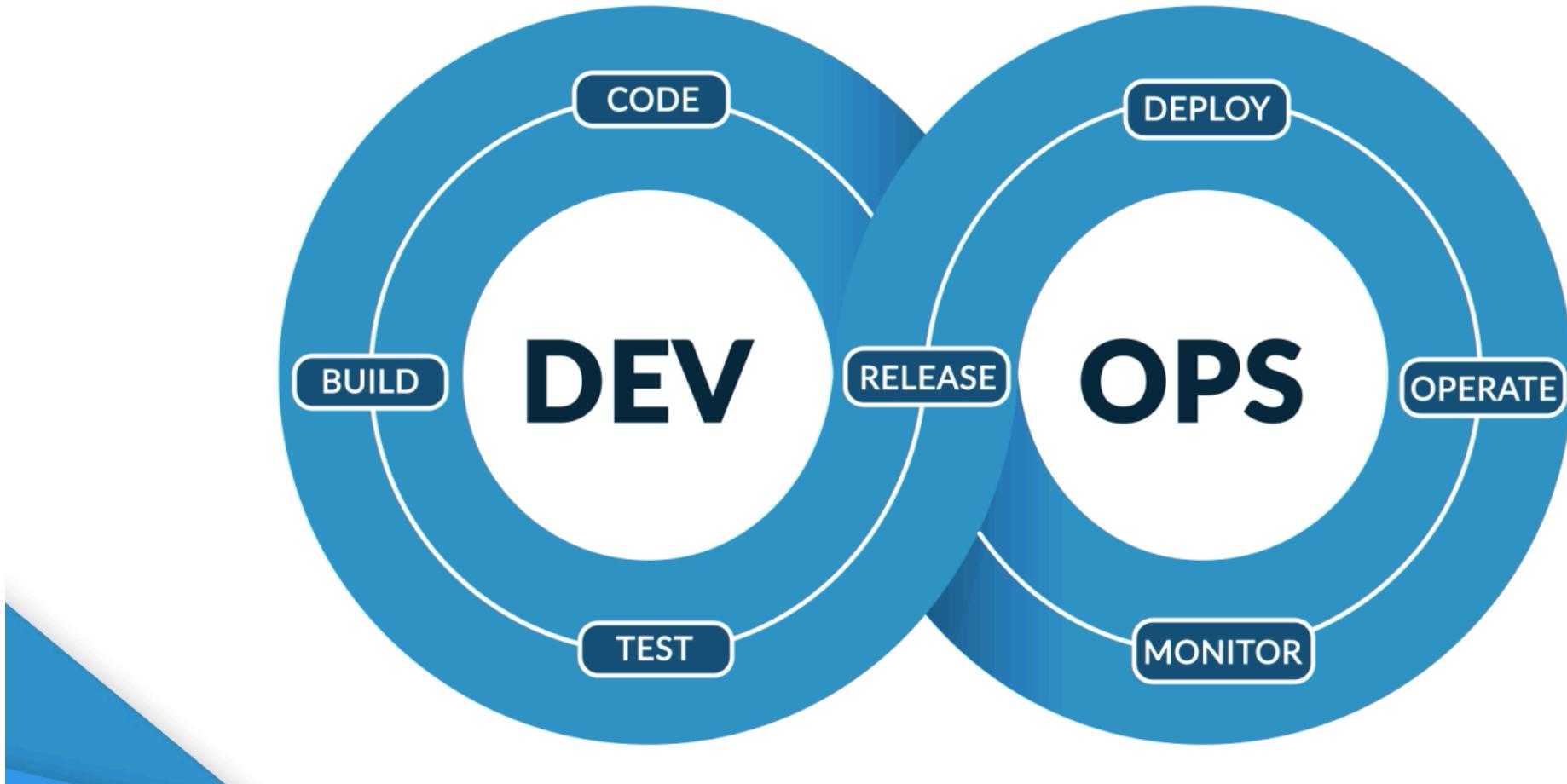
Analyze streaming data using analytics tools

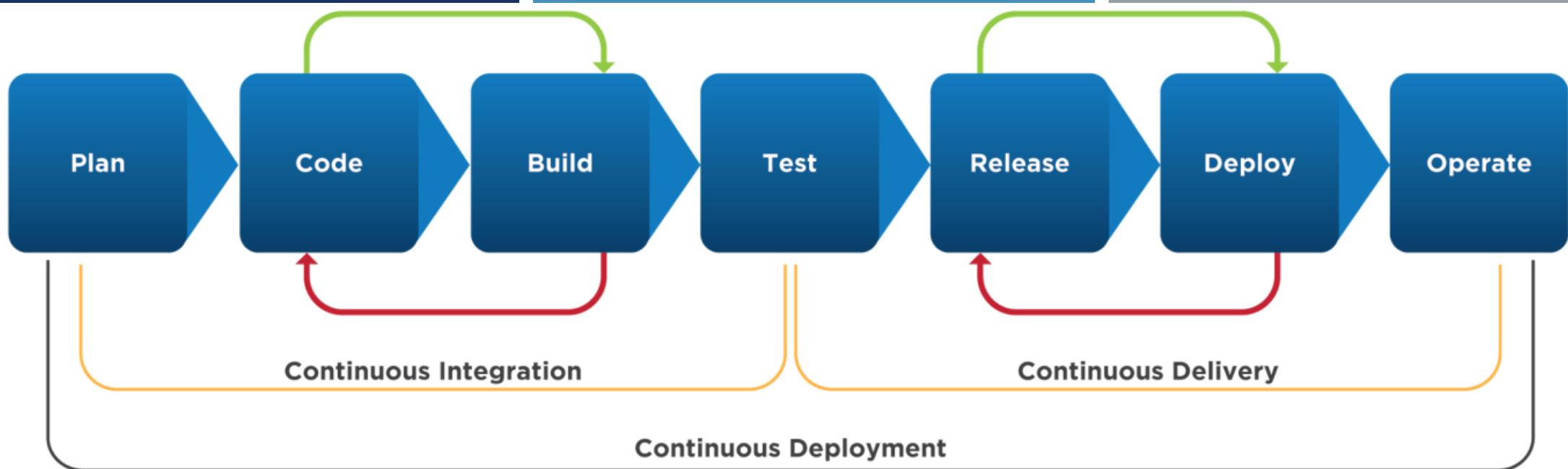


Output

Amazon Kinesis Data Analytics can send processed data to analytics tools so you can create alerts and respond in real-time

DevOps Introduction





Planning
<ul style="list-style-type: none"> Requirement finalization Updates & new changes Architecture & design Task assignment Timeline finalization

Code
<ul style="list-style-type: none"> Development Configuration finalization Check-in source code Static-code analysis Automated review & peer review

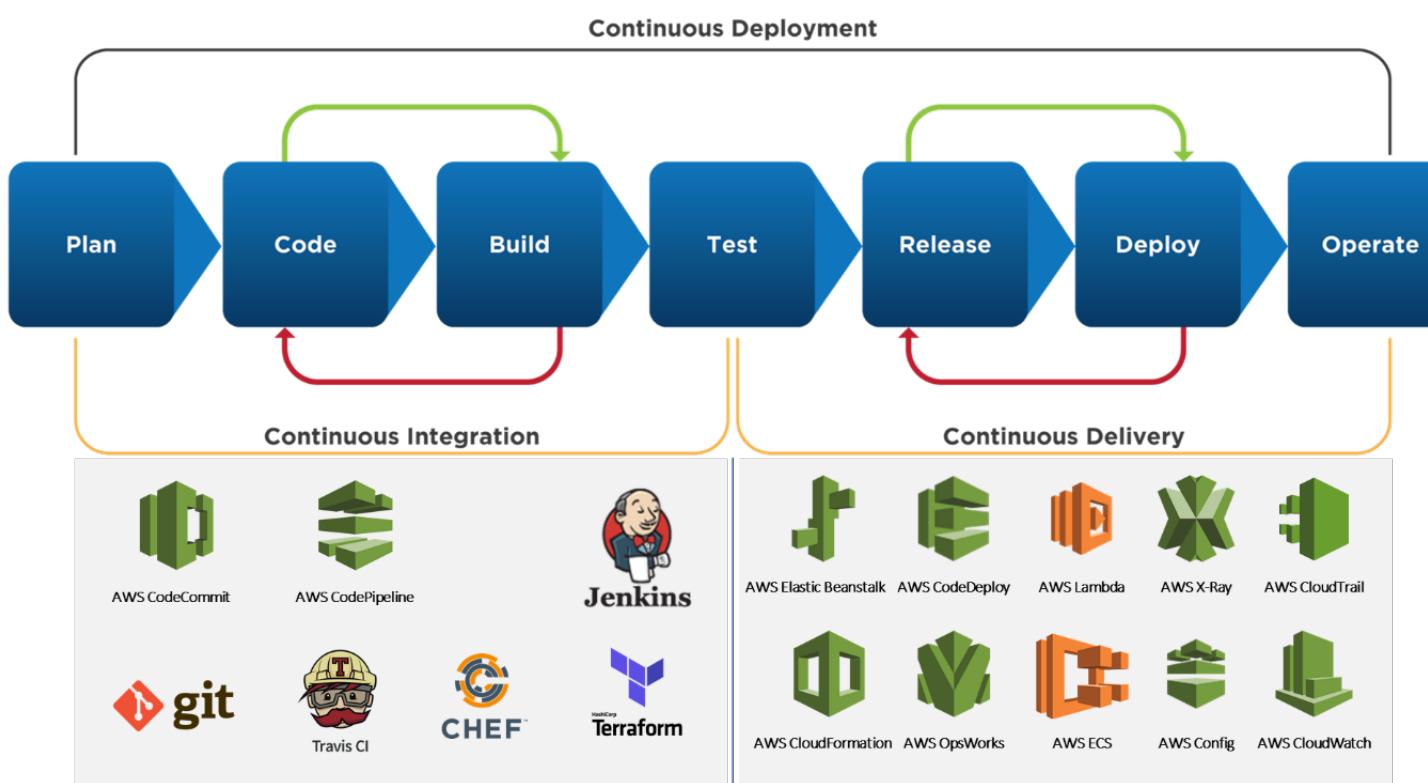
Build
<ul style="list-style-type: none"> Compile code Unit testing Code-metrics Build container images or package Preparation or update in deployment templated Create or update monitor dashboards

Test
<ul style="list-style-type: none"> Integration test with other component Load & stress test UI testing Penetration testing Requirement testing

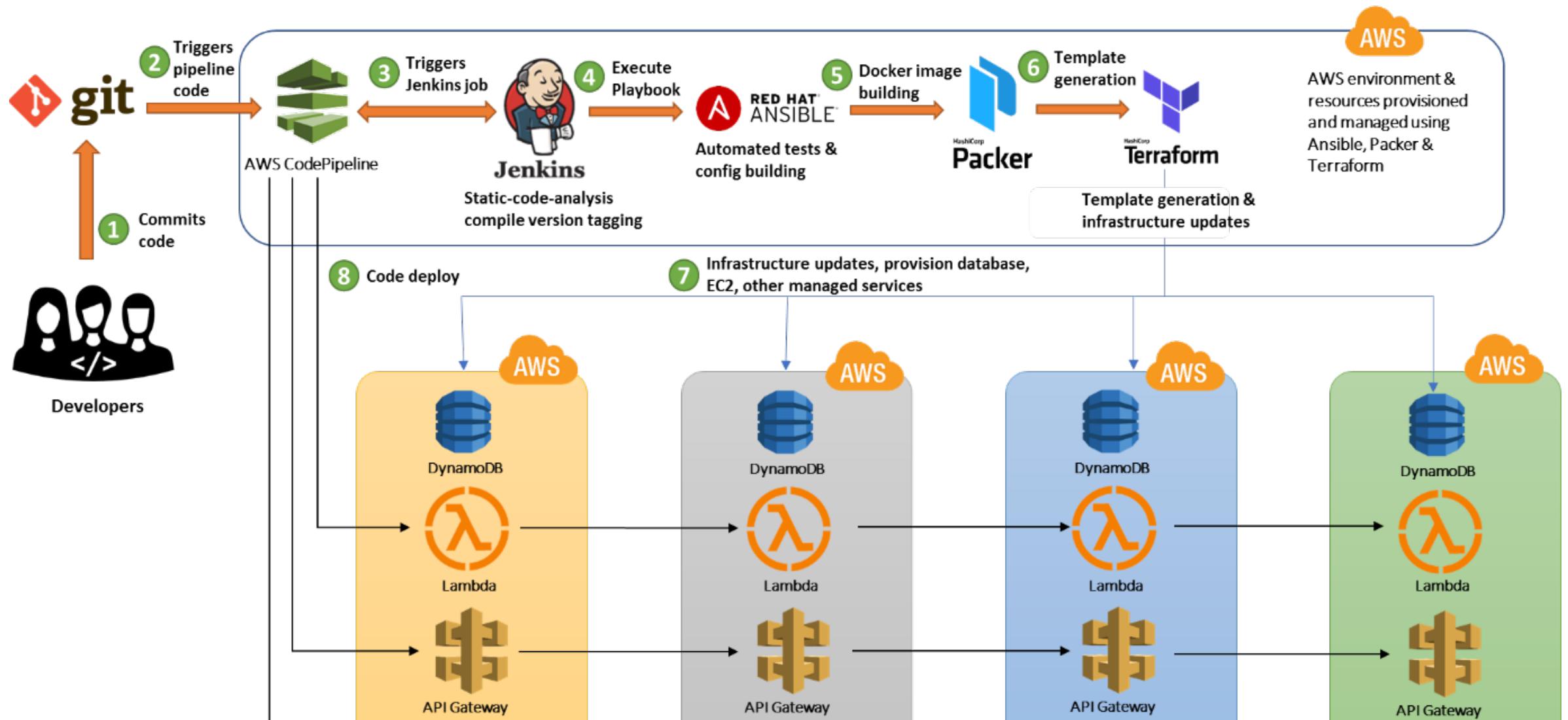
Release
<ul style="list-style-type: none"> Preparing release notes Version tagging Code freeze Feature freeze

Deploy
<ul style="list-style-type: none"> Updating the infrastructure i.e staging, production Verification on deployment i.e smoke tests

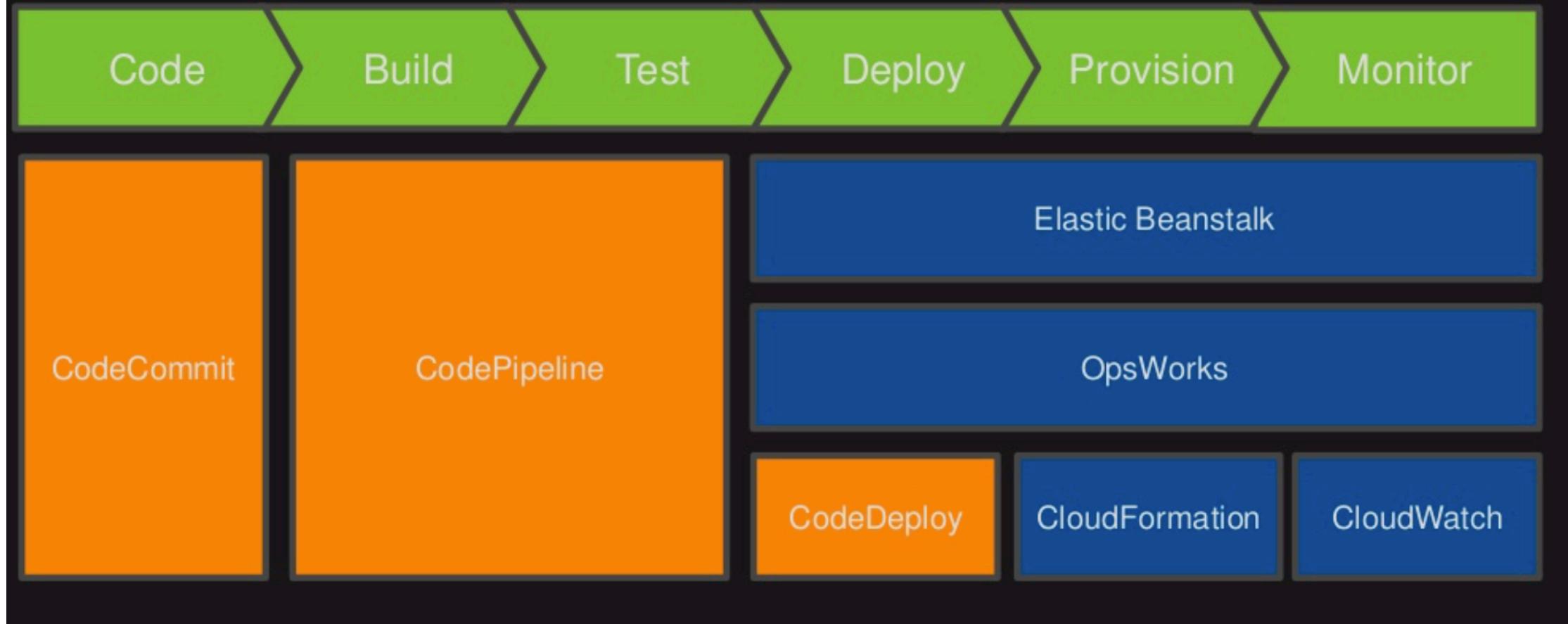
Operate
<ul style="list-style-type: none"> Monitor designed dashboard Alarm triggers Automatic critical events handler Monitor error logs



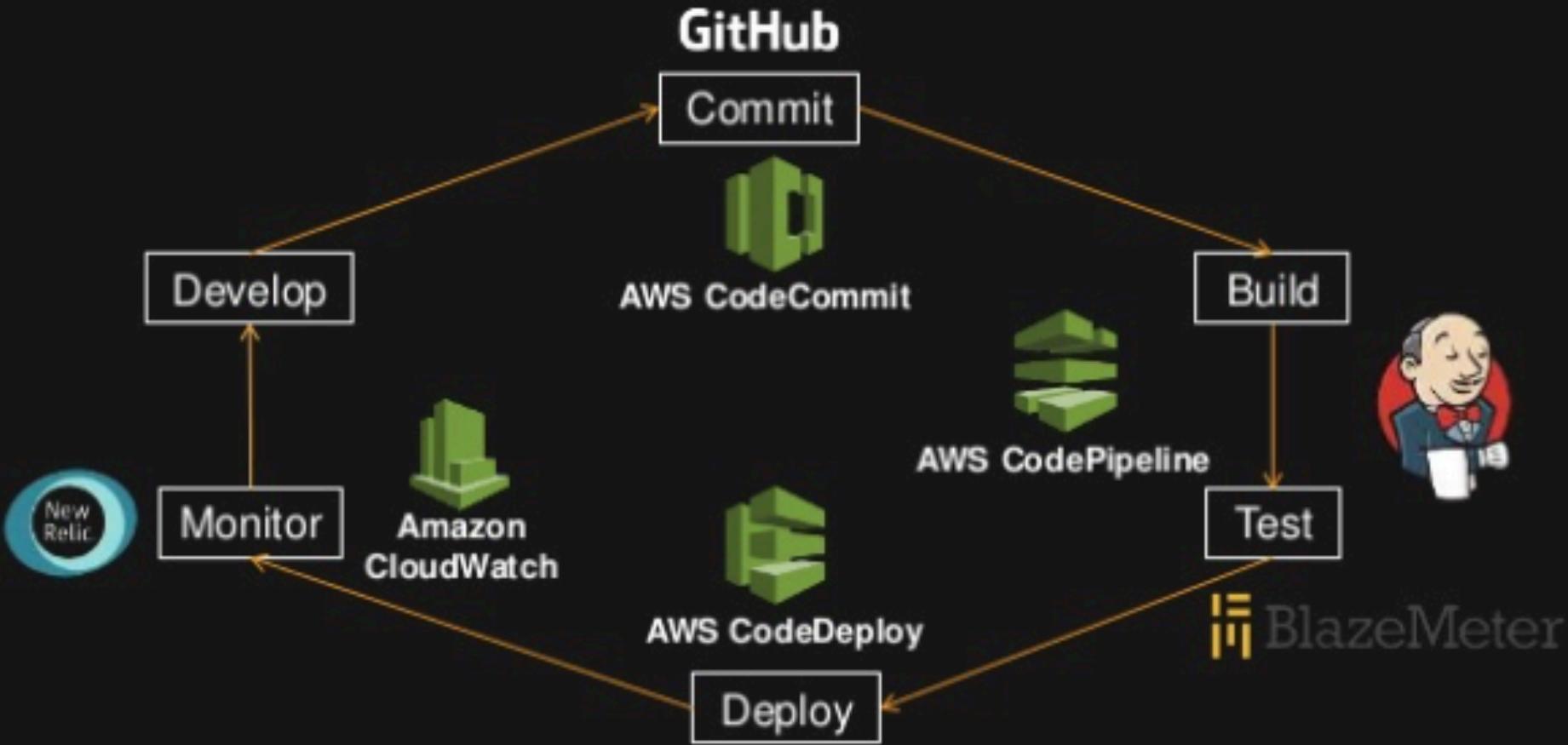
- **Continuous Integration** focuses on the software development cycle of the individual developer in the code repository. This can be executed multiple times in a day with a primary purpose to enable early detection of integration bugs, tighter cohesion, and more development collaboration. Major activities are like static code analysis, unit tests, and automated reviews.
- **Continuous Delivery** focuses on automated code deployment in testing, staging or production environment, taking the approval of updates to achieve automated software release process, pre-emptively discovering deployment issues.



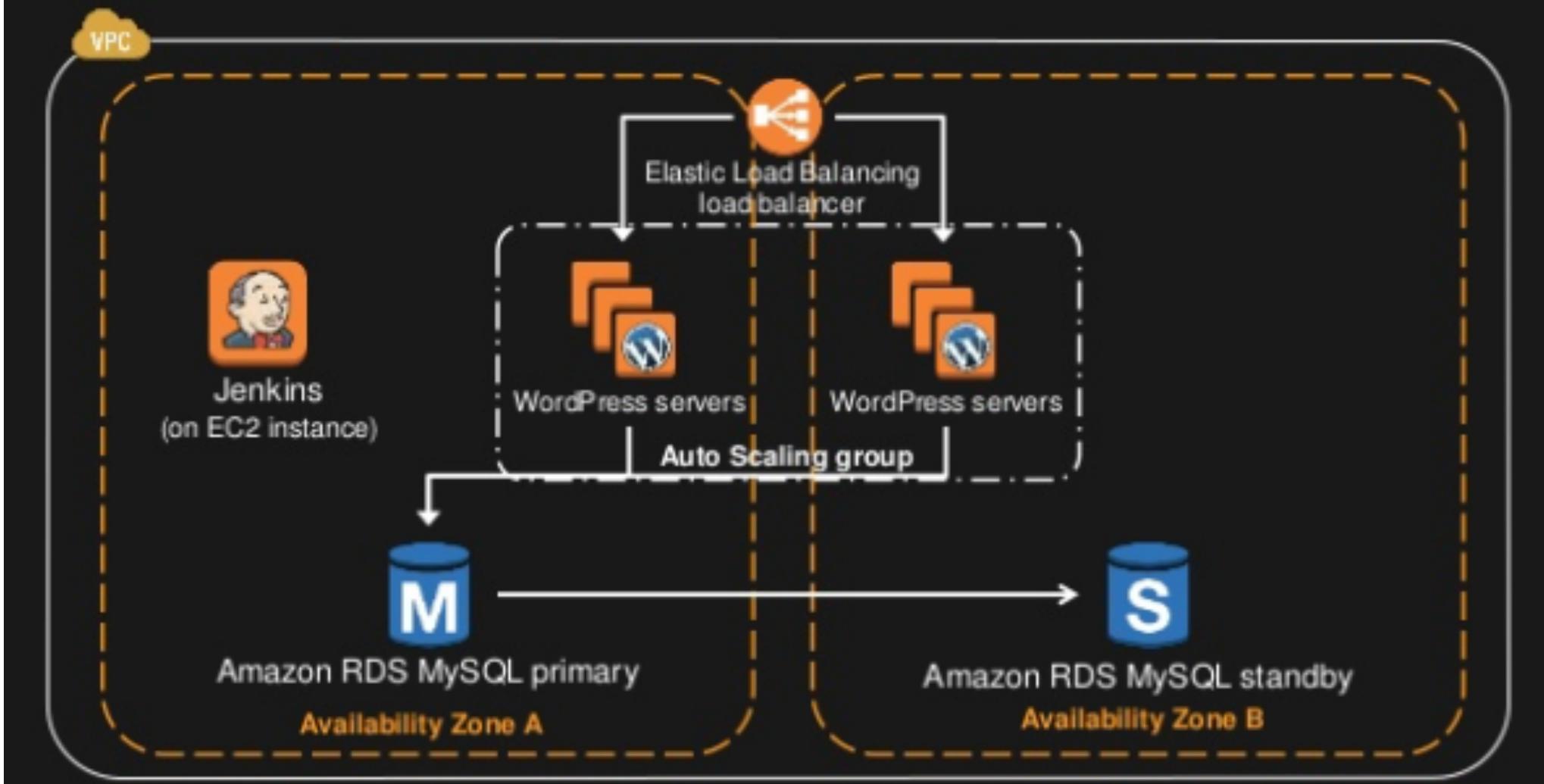
Cloud software development lifecycle



Today's demo

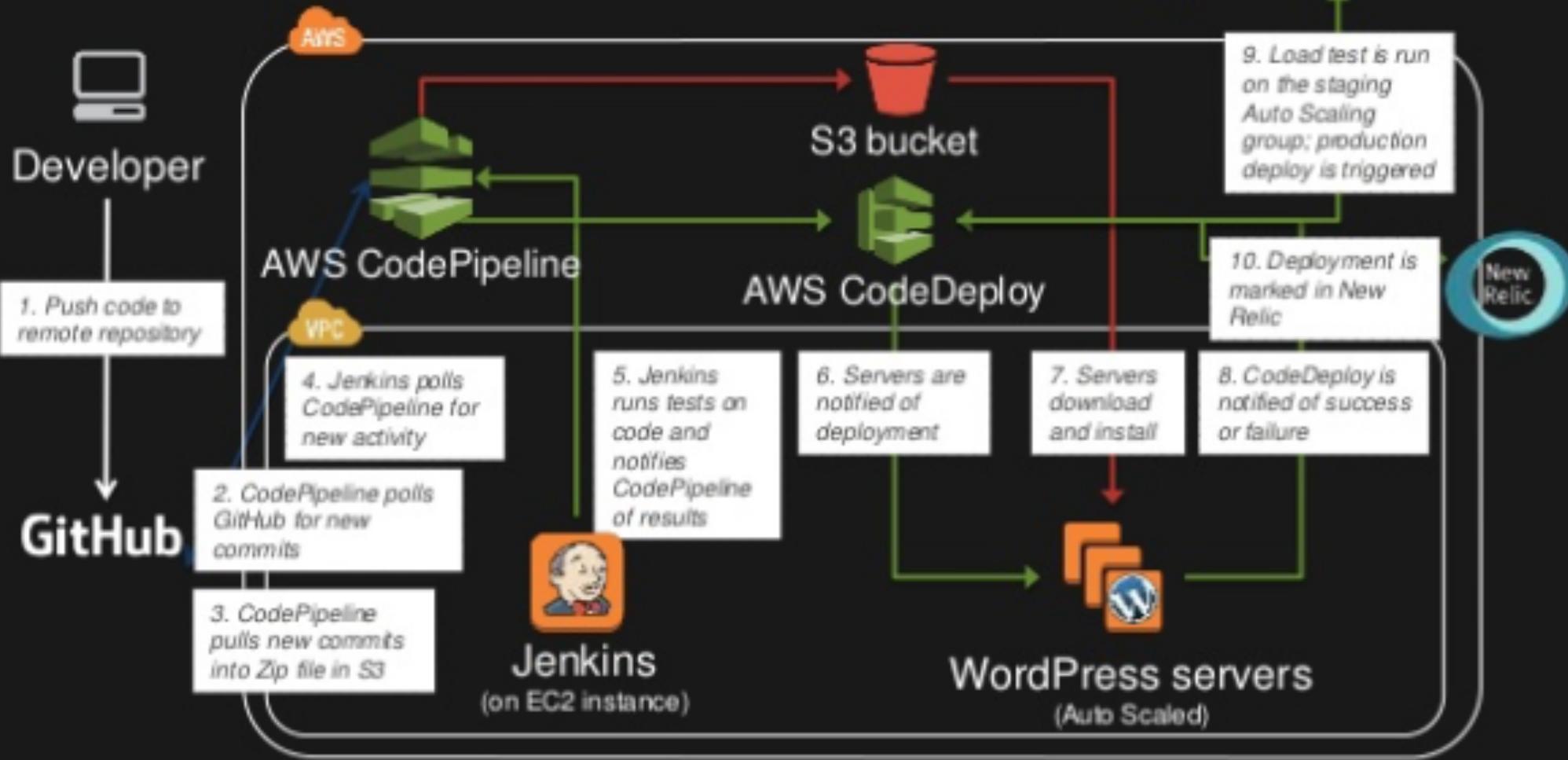


Application architecture



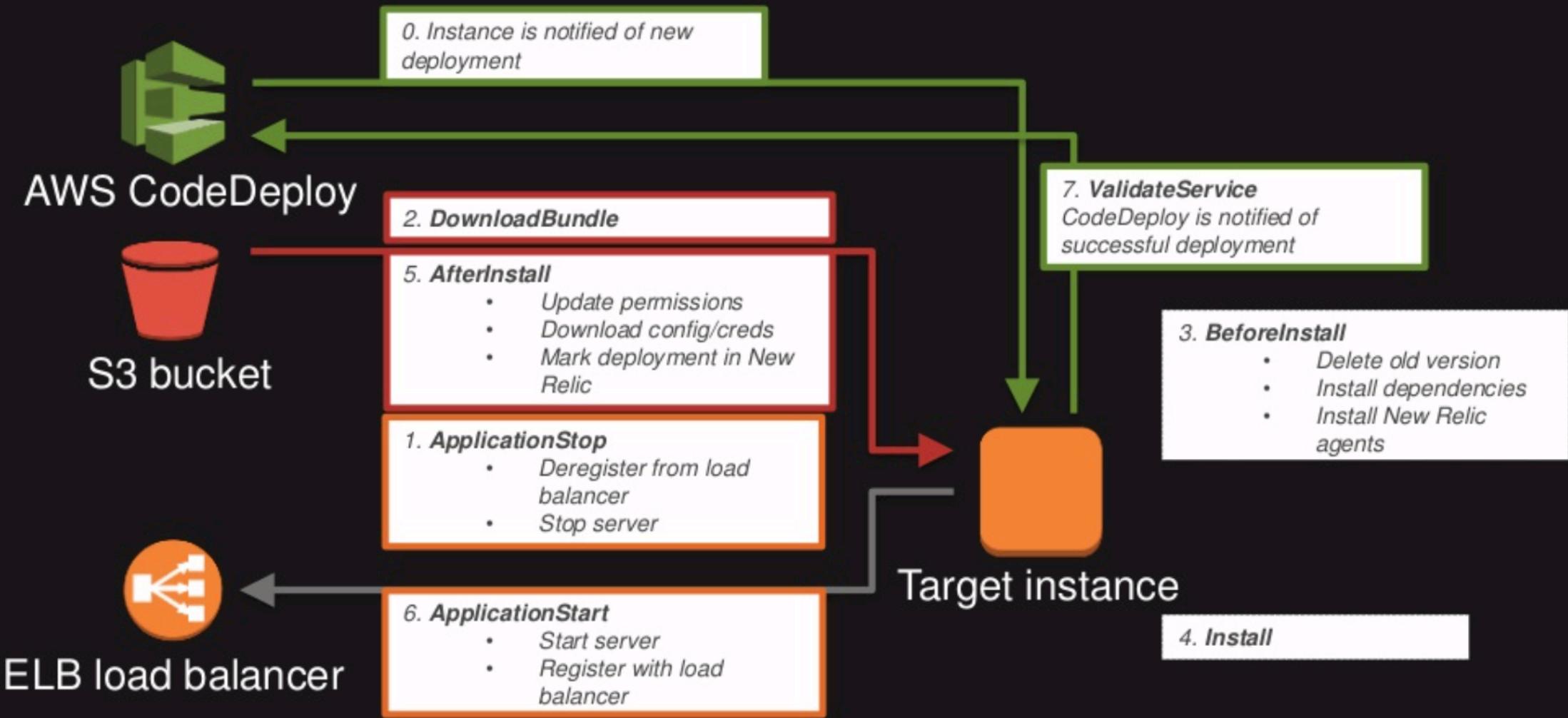
Deployment overview

BlazeMeter

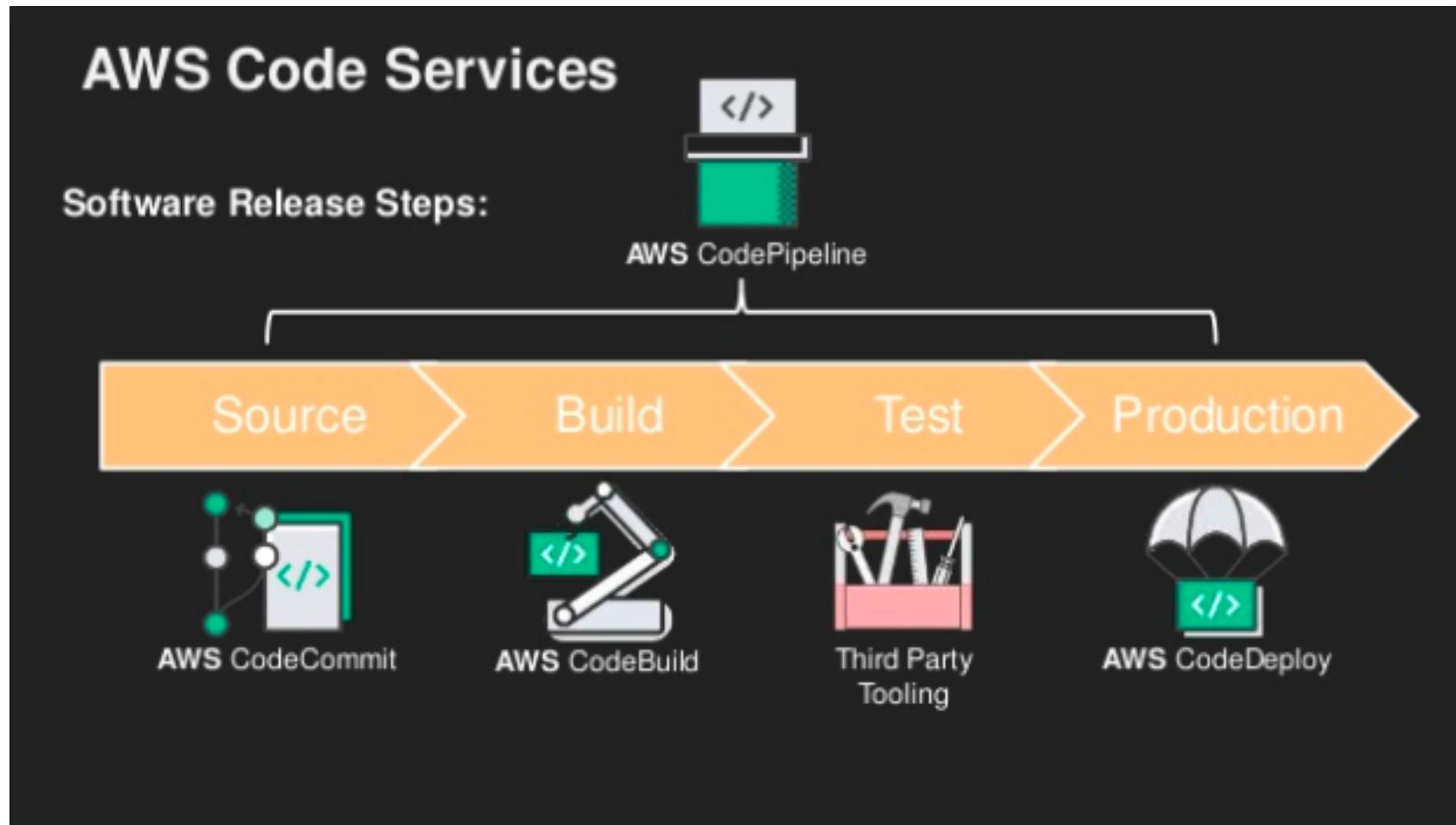


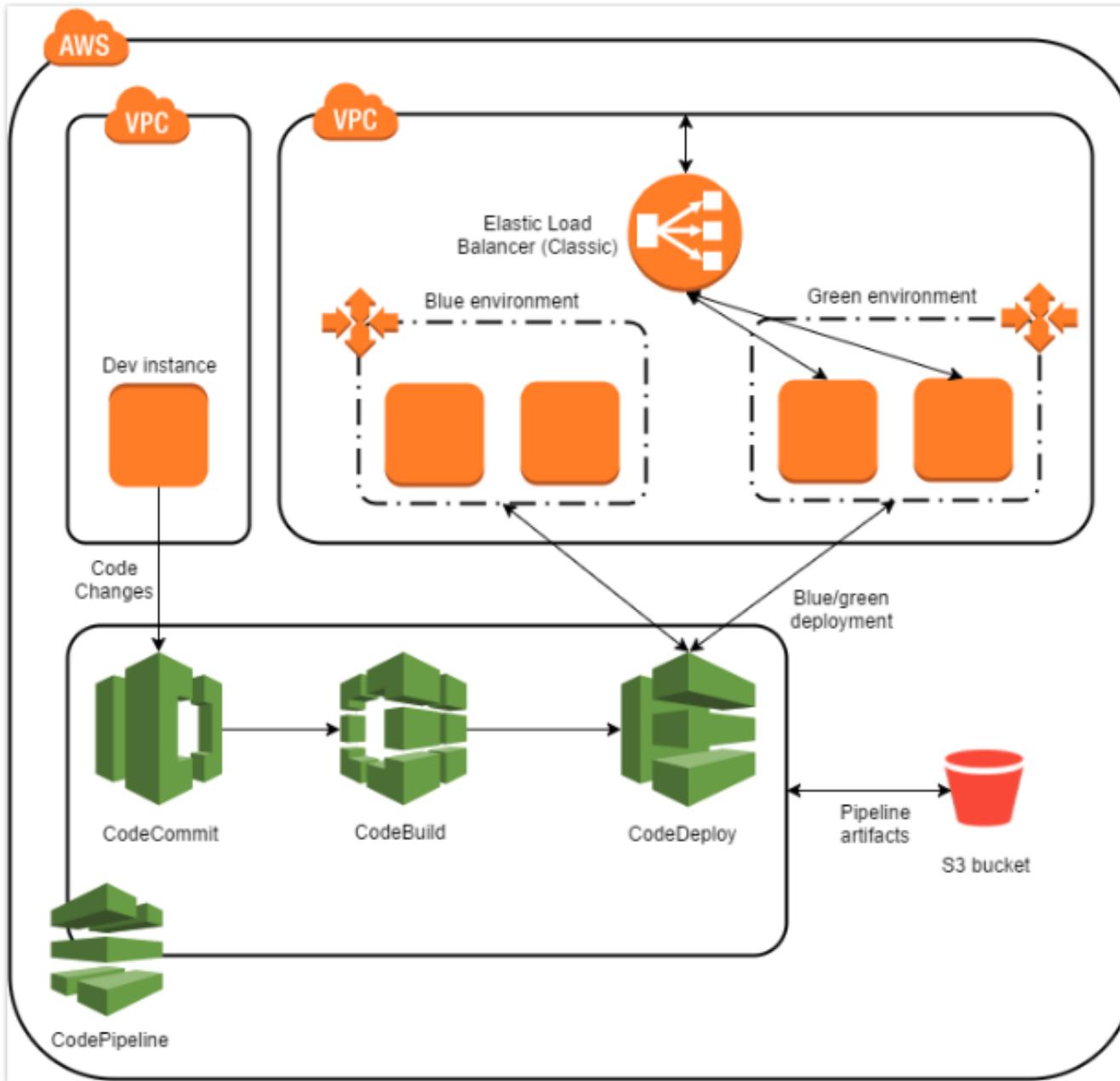
CodeDeploy lifecycle events

Clip slide



AWS Developer Tools





AWS CodeDeploy

- Automated deployment to EC2, on premises systems and Lambda
- Integrates with various CI/CD tools –Jenkins, Github, Atlassian, AWS CodePipeline- as well as configuration management tools like Ansible, Puppet and Chef.
- Two deployment approaches available:
 - In-Place (Rolling Update)
 - Blue/Green

AWS CodePipeline

- It's a fully managed Continuous Integration and Continuous Delivery service
- CodePipeline can orchestrate the Build, Test, and even Deployment of your application every time there is a change to your code – all based on a user defined software release process.
- A pipeline defines your release process workflow, and describes how a new code change progresses through your release process. A pipeline comprises a series of stages (e.g., build, test, and deploy), which act as logical divisions in your workflow. Each stage is made up of a sequence of actions, which are tasks such as building code or deploying to test environments. AWS CodePipeline provides you with a graphical user interface to create, configure, and manage your pipeline and its various stages and actions, allowing you to easily visualize and model your release process workflow.

AWS Developer Tools

- AWS CodePipeline can pull source code for your pipeline directly from AWS CodeCommit or Amazon S3. It can run builds and unit tests in AWS CodeBuild. CodePipeline can deploy your changes using AWS CodeDeploy, AWS Elastic Beanstalk, Amazon Elastic Container Service(Amazon ECS), or AWS OpsWorks.
- You can model AWS CloudFormation actions that let you provision, update, or delete AWS resources as part of your release process. This also enables you to continuously deliver serverless applications built using AWS Lambda, Amazon API Gateway, and Amazon DynamoDBwith the AWS Serverless Application Model (AWS SAM).
- You can also trigger custom functions defined by code at any stage of your pipeline using CodePipeline's integration with AWS Lambda. For example, you can trigger a Lambda function that tests whether your web application deployed successfully.
- CodePipeline lets you configure a pipeline that ties these services together along with third-party developer tools and custom systems.

AWS CodeBuild

- CodeBuild is a fully managed build service which runs a set of commands that you define –e.g. compiles code, run test and produces artifacts that are ready to deploy.

CloudFormation

- CloudFormation is a service that allows you to manage, configure and provision your AWS infrastructure as code.
- Resources are defined using a CloudFormation template.
- CloudFormation interprets the template and makes the appropriate API calls to create the resources you have defined
- Supports YAML or JSON

CloudFormation

CloudFormation: Infrastructure As Code

AWS CloudFormation allows you to **launch, configure, and connect AWS resources** with JavaScript Object Notation (JSON) templates.

Template



- 💡 JSON-formatted file describing the resources to be created
- 💡 Treat it as source code: put it in your repository

AWS CloudFormation Engine



- 💡 AWS service component
- 💡 Interprets AWS CloudFormation template into stacks of AWS resources

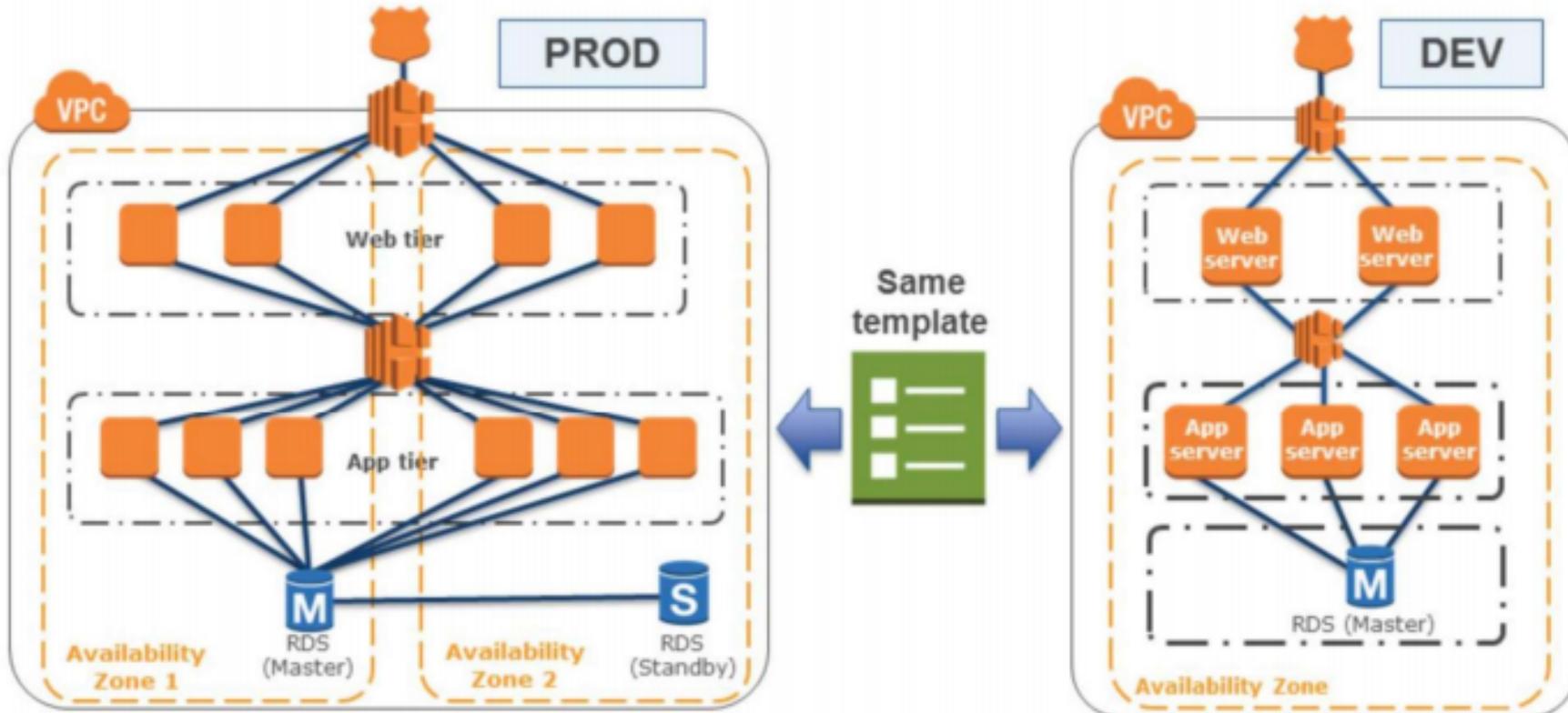
Stack



- 💡 A collection of resources created by AWS CloudFormation
- 💡 Tracked and reviewable in the AWS Management Console

CloudFormation

Building Environments With Conditions



Elastic Beanstalk

Elastic Beanstalk

- With Elastic Beanstalk, you can quickly deploy and manage applications in the AWS Cloud without worrying about the infrastructure that runs those applications. AWS Elastic Beanstalk reduces management complexity without restricting choice or control. You simply upload your application, and Elastic Beanstalk automatically handles the details of capacity provisioning, load balancing, scaling, and application health monitoring.
- Elastic Beanstalk supports applications developed in Java, .NET, PHP, Node.js, Python, Ruby, Go, and Docker on familiar servers such as Apache, Nginx, Passenger, and IIS. A configuration defines the infrastructure and software stack to be used for a given environment

Elastic Beanstalk II

- To use Elastic Beanstalk, you create an application, upload an application version in the form of an application source bundle (for example, a Java .war file) to Elastic Beanstalk, and then provide some information about the application. Elastic Beanstalk automatically launches an environment and creates and configures the AWS resources needed to run your code. After your environment is launched, you can then manage your environment and deploy new application versions.
- After you create and deploy your application, information about the application—including metrics, events, and environment status—is available through the AWS Management Console, APIs, or Command Line Interfaces, including the unified AWS CLI. For step-by-step instructions on how to create, deploy, and manage your application using the AWS Management Console, go to [Getting Started Using Elastic Beanstalk](#).

AWS Elastic Beanstalk



AWS Elastic Beanstalk is an **automated deployment and scaling service** for web applications.

AWS Elastic Beanstalk:

- Accepts Java, .NET, PHP, Node.js, Python, Ruby, Go, or Docker code
- Deploys on Apache, Nginx, Passenger, and IIS servers.

A deployment with AWS Elastic Beanstalk can automatically handle:

- Load balancing
- Health monitoring
- Autoscaling
- Application platform management
- Code deployment

Elastic Beanstalk Lab

EBS Deployment Policies

- All at once
 - Rolling
 - Rolling with additional batch
 - Immutable
-
- By default, your environment uses rolling deployments if you created it with the console or EB CLI, or all-at-once deployments if you created it with a different client (API, SDK, or AWS CLI).

Elastic Beanstalk

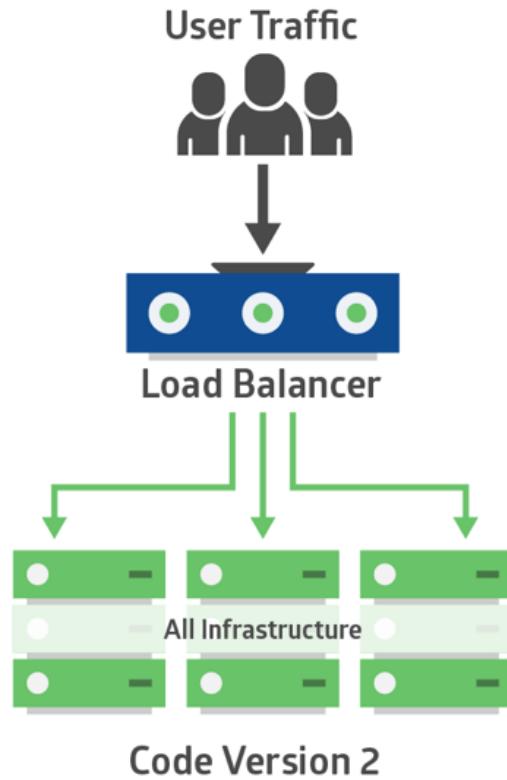
Deployment Methods

Method	Impact of Failed Deployment	Deploy Time	Zero Downtime	No DNS Change	Rollback Process	Code Deployed To
All at once	Downtime	⌚	X	✓	Manual Redeploy	Existing instances
Rolling	Single batch out of service; any successful batches prior to failure running new application version	⌚⌚⌚+†	✓	✓	Manual Redeploy	Existing instances
Rolling with additional batch	Minimal if first batch fails, otherwise, similar to Rolling	⌚⌚⌚+†	✓	✓	Manual Redeploy	New and existing instances
Immutable	Minimal	⌚⌚⌚⌚	✓	✓	Terminate New Instances	New instances
Blue/green	Minimal	⌚⌚⌚⌚	✓	X	Swap URL	New instances

† Varies depending on batch size.

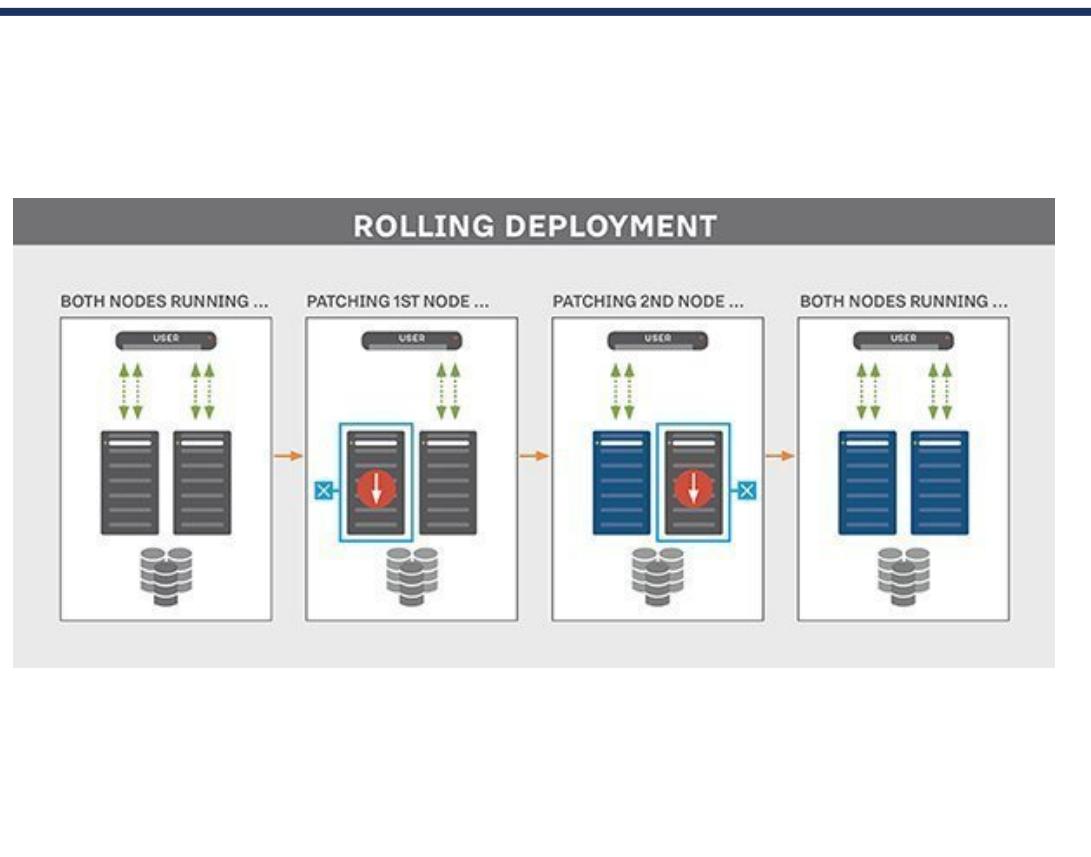
<https://docs.aws.amazon.com/elasticbeanstalk/latest/dg/using-features.deploy-existing-version.html>

ALL AT ONCE DEPLOYMENT POLICY



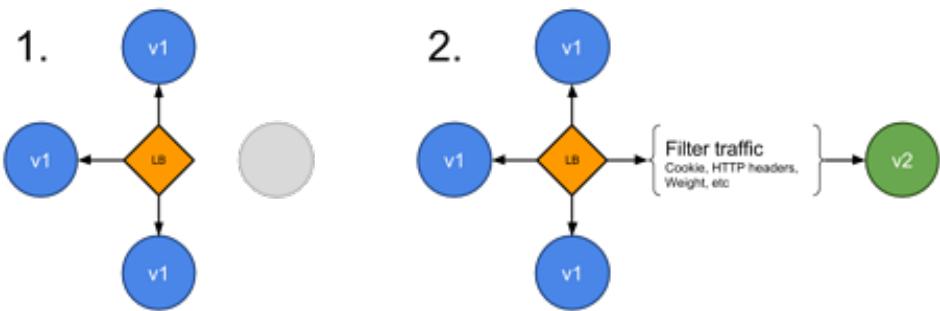
- Deploys the new version to all instances simultaneously
- All instances are out of service while the deployment takes place: not ideal for mission-critical production systems
- If the update fails, you need to roll back the changes by re-deploying the original version to all your instances

ROLLING DEPLOYMENT POLICY



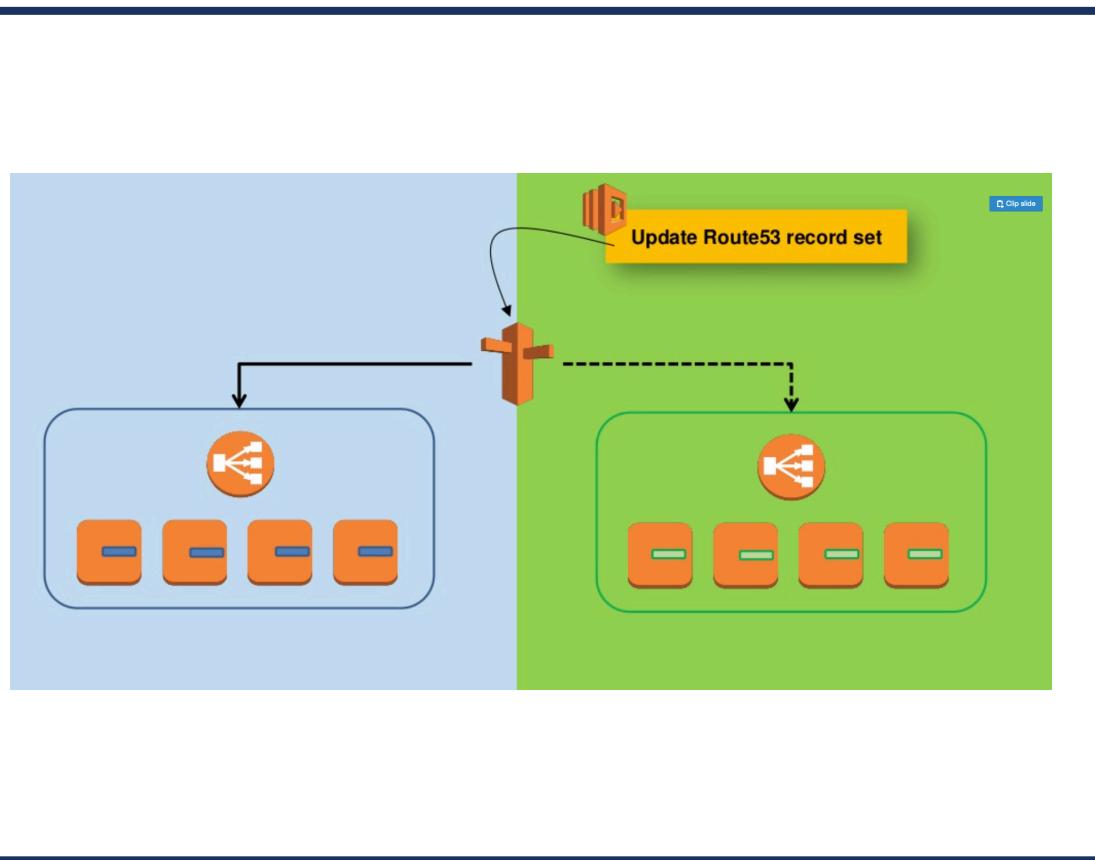
- With **rolling** deployments, Elastic Beanstalk splits the environment's EC2 instances into batches and deploys the new version of the application to one batch at a time, leaving the rest of the instances in the environment running the old version of the application.
- During a rolling deployment, some instances serve requests with the old version of the application, while instances in completed batches serve other requests with the new version.
- Not ideal for performance sensitive systems
- If the update fails, you need to perform an additional rolling update to roll back the changes

ROLLING WITH ADDITIONAL BATCH POLICY



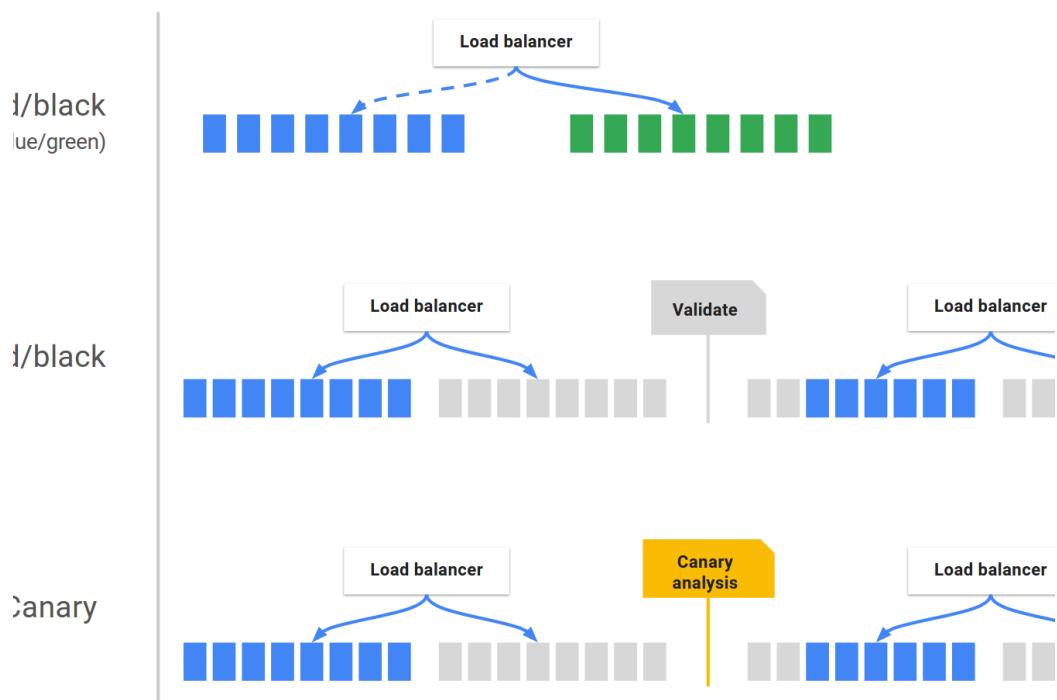
- If you need to maintain full capacity during deployments, you can configure your environment to launch a new batch of instances prior to taking any instances out of service.
- When the deployment completes, Elastic Beanstalk terminates the additional batch of instances.
- Maintains full capacity during the deployment process
- If the update fails, you need to perform an additional rolling update to roll back the changes

IMMUTABLE DEPLOYMENT POLICY



- **Immutable deployments** launch a full set of new instances running the new version of the application in a separate Auto Scaling group, alongside the instances running the old version.
- Immutable deployments can prevent issues caused by partially completed rolling deployments. If the new instances don't pass health checks, Elastic Beanstalk terminates them, leaving the original instances untouched.
- Maintains full capacity during the deployment process
- Preferred option for Mission Critical systems

BLUE/GREEN DEPLOYMENT POLICY



- Because AWS Elastic Beanstalk performs an in-place update when you update your application versions, your application can become unavailable to users for a short period of time. You can avoid this downtime by performing a blue/green deployment, where you deploy the new version to a separate environment, and then swap CNAMEs of the two environments to redirect traffic to the new version instantly.
- Blue/green deployments require that your environment runs independently of your production database, if your application uses one. If your environment has an Amazon RDS DB instance attached to it, the data will not transfer over to your second environment, and will be lost if you terminate the original environment.

Personalización avanzada de entornos con archivos de configuración (.ebextensions)

Puede agregar archivos de configuración de AWS Elastic Beanstalk (.ebextensions) al código fuente de la aplicación web para configurar el entorno y personalizar los recursos de AWS que contiene. Los archivos de configuración son documentos con formato YAML o JSON y con una extensión de archivo .config que se sitúan en una carpeta llamada .ebextensions y se implementan en el paquete de código fuente de la aplicación.

Recomendamos utilizar YAML para los archivos de configuración, ya que es más fácil de leer que JSON. YAML admite comentarios, comandos multilínea, varias alternativas de uso de comillas, etc. Sin embargo, puede realizar cualquier cambio de configuración en los archivos de configuración de Elastic Beanstalk exactamente igual usando YAML o JSON.

Sugerencia

Cuando esté desarrollando o probando nuevos archivos de configuración, lance un entorno limpio que ejecute la aplicación predeterminada e impleméntelos en este entorno. Si los archivos de configuración no tienen un formato correcto, se producirán error irrecuperables al lanzar un nuevo entorno.

En la sección option_settings de un archivo de configuración, se definen los valores de las [opciones de configuración](#). Las opciones de configuración le permiten configurar el entorno de Elastic Beanstalk, los recursos de AWS que contiene el entorno y el software que ejecuta la aplicación. Los archivos de configuración son solo uno de los diferentes mecanismos para definir las opciones de configuración.

Personalización avanzada de entornos con archivos de configuración (.ebextensions)

La [sección Resources](#) le permite personalizar aún más los recursos del entorno de la aplicación y definir otros recursos de AWS para obtener otra funcionalidad que la que proporcionan las opciones de configuración. Puede agregar y configurar los recursos admitidos por AWS CloudFormation, que Elastic Beanstalk utiliza para crear entornos.

El resto de las secciones de un archivo de configuración (packages, sources, files, users, groups, commands, container_commands y services) le permiten configurar las instancias EC2 que se lanzan en el entorno. Cuando se lanza un servidor en el entorno, Elastic Beanstalk ejecuta las operaciones definidas en estas secciones para preparar el sistema operativo y el sistema de almacenamiento de la aplicación.

Para ver ejemplos de .ebextensions utilizados, consulte el [Elastic Beanstalk Configuration Files Repository](#).

Personalización avanzada de entornos con archivos de configuración (.ebextensions)

Requisitos

- **Ubicación:** coloque todos sus archivos de configuración en una sola carpeta, denominada `.ebextensions`, en la raíz de su paquete de código fuente. Los exploradores de archivos pueden ocultar las carpetas que comienzan con un punto, así que asegúrese de agregar la carpeta al crear el paquete de código fuente. Consulte [Creación del paquete de código fuente de una aplicación](#) para obtener instrucciones.
- **Denominación:** los archivos de configuración deben tener la extensión de archivo `.config`.
- **Formato:** Los archivos de configuración deben cumplir las especificaciones de formato de YAML o JSON.
- Si utiliza YAML, use siempre espacios para aplicar sangría a las claves en los diferentes niveles de anidación. Para obtener más información acerca de YAML, consulte [YAML Ain't Markup Language \(YAML™\) \(versión 1.1\)](#).
- **Unicidad:** Use cada clave una sola vez en cada archivo de configuración.
- **Advertencia**
- Si utiliza una clave (por ejemplo, `option_settings`) dos veces en el mismo archivo de configuración, se descartará una de las secciones. Combine secciones duplicadas en una sola sección o colóquelas en archivos de configuración distintos.

La Consola de Elastic Beanstalk y la CLI de EB establecen las opciones de configuración al crear un entorno, incluidas las opciones que establece explícitamente y [valores recomendados](#) definidos por el cliente. También puede establecer opciones de configuración en las configuraciones guardadas y en los archivos de configuración. Si se ha establecido la misma opción en varias ubicaciones, el valor utilizado se determina según el [orden de prioridad](#).

Los valores de las opciones de configuración están en formato de texto y se guardan antes de que se cree el entorno, se aplican durante la creación del entorno mediante cualquier cliente compatible y se añaden, modifican o eliminan después de crear el entorno. Para obtener un listado detallado de todos los métodos disponibles para trabajar con opciones de configuración en cada una de estas tres fases, consulte los siguientes temas:

- [Ajuste de opciones de configuración antes de la creación del entorno](#)
- [Ajuste de opciones de configuración durante la creación del entorno](#)
- [Ajuste de opciones de configuración después de crear el entorno](#)

Implementación de aplicaciones de Elastic Beanstalk desde contenedores de Docker

Elastic Beanstalk permite implementar aplicaciones web desde contenedores de Docker. Con los contenedores de Docker, puede definir su propio entorno de ejecución. Puede elegir una plataforma, un lenguaje de programación y unas dependencias de la aplicación (por ejemplo, herramientas o administradores de paquetes) propios que no sean compatibles con otras plataformas. Los contenedores de Docker son autónomos y contienen todo el software y la información de configuración que la aplicación web necesita para ejecutarse. Todas las variables de entorno definidas en la consola de Elastic Beanstalk se transfieren a los contenedores.

Si usa Docker con Elastic Beanstalk, tendrá una infraestructura que administrará automáticamente los detalles de aprovisionamiento de la capacidad, el balanceo de carga, el escalado y la monitorización del estado de las aplicaciones. Puede administrar la aplicación web en un entorno que sea compatible con el conjunto de servicios que se integran con Elastic Beanstalk; por ejemplo, [VPC](#), [RDS](#) e [IAM](#). Para obtener más información sobre Docker, como su instalación, el software que requiere y el uso de imágenes para lanzar contenedores de Docker, visite [Docker: the Linux container engine](#).

nota

Si un contenedor de Docker que se ejecuta en un entorno de Elastic Beanstalk se bloquea o se cierra por algún motivo, Elastic Beanstalk lo reinicia automáticamente.

En los temas de este capítulo se parte de la base de que se tienen conocimientos de los entornos de Elastic Beanstalk. Si no ha usado Elastic Beanstalk antes, pruebe el [tutorial de introducción](#) para conocer los conceptos básicos.

Plataformas de Docker

La familia de plataformas de Docker para Elastic Beanstalk tiene dos plataformas genéricas ("single container" y "multicontainer") y varios contenedores preconfigurados.

Single Container Docker

La plataforma "single container" puede utilizarse para implementar una imagen de Docker (descrita en un archivo Dockerfile o en una definición Dockerrun.aws.json) y el código fuente en instancias EC2 que se ejecutan en un entorno de Elastic Beanstalk. Utilice la plataforma "single container" cuando solamente necesite ejecutar un contenedor por instancia. Si desea ver ejemplos y obtener ayuda para empezar a usar un entorno de Docker con un único contenedor, consulte [Entornos de Docker con un único contenedor](#)

Multicontainer Docker

La otra plataforma básica, Multicontainer Docker, utiliza Amazon Elastic Container Service para coordinar una implementación de varios contenedores de Docker en un clúster de Amazon ECS de un entorno de Elastic Beanstalk. Todas las instancias del entorno ejecutan el mismo conjunto de contenedores, que está definido en un archivo Dockerrun.aws.json. Utilice la plataforma "Multicontainer" cuando necesite implementar varios contenedores de Docker en cada instancia.

Contenedores de Docker preconfigurados

Además de las dos plataformas genéricas de Docker, existen varias versiones *preconfiguradas* de la plataforma de Docker que puede utilizar para ejecutar la aplicación en un conjunto de software que se utiliza habitualmente, como *Java con Glassfish* o *Python con uWSGI*. Utilice un contenedor preconfigurado si se ajusta al software que utilizado por la aplicación.

Deployment and Management

AWS Elastic Beanstalk

Automated resource management – web apps made easy



AWS OpsWorks

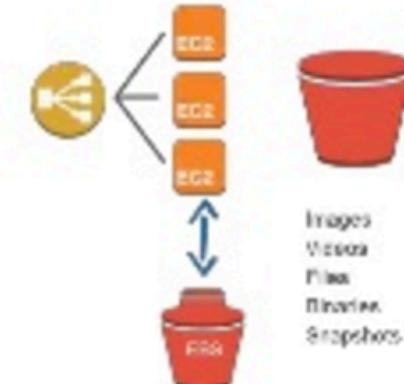
DevOps framework for application lifecycle management and automation



Templates to deploy & update infrastructure as code



DIY, on demand resources: EC2, S3, custom AMI's, etc.



Convenience

Control

AWS OpsWorks

AWS OpsWorks es un servicio de administración de configuración que ofrece instancias administradas de Chef y Puppet. Chef y Puppet son plataformas de automatización que le permiten usar su código para automatizar la configuración de sus servidores. OpsWorks le permite usar Chef y Puppet para automatizar la manera en la que los servidores se configuran, implementan y administran en las instancias de [Amazon EC2](#) o en entornos informáticos en las instalaciones.

OpsWorks ofrece tres versiones:

- [AWS Opsworks for Chef Automate](#)
- [AWS OpsWorks for Puppet Enterprise](#)
- [AWS OpsWorks Stacks](#).

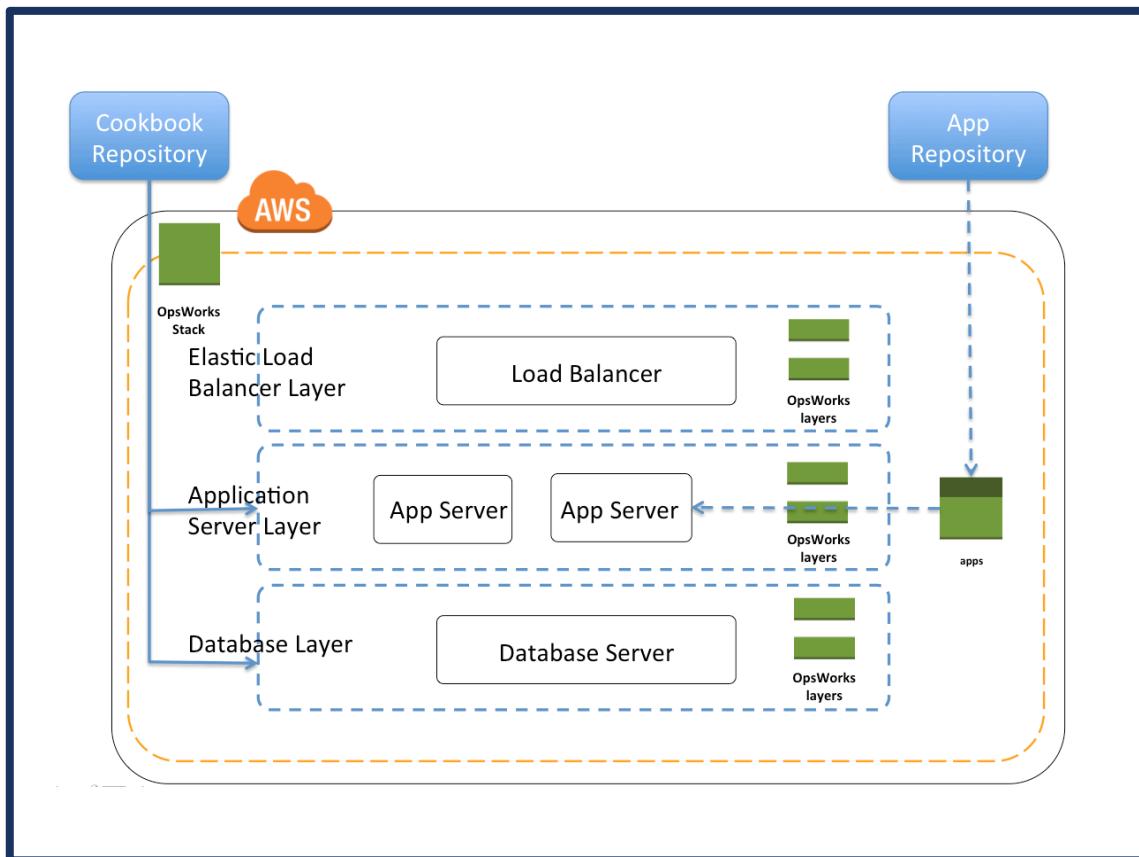
AWS OpsWorks for Chef Automate

- AWS OpsWorks for Chef Automate es un servicio de administración de configuración completamente administrado que aloja Chef Automate, un grupo de herramientas de automatización de Chef para administración de configuración, seguridad y conformidad, e implementación continua. OpsWorks también mantiene el servidor de Chef mediante la implementación de parches, la actualización y el backup automáticos del servidor. OpsWorks elimina la necesidad de operar sistemas de administración de configuración propios y de preocuparse por el mantenimiento de la infraestructura. OpsWorks le brinda acceso a todas las características de Chef Automate, como la administración de la configuración y la conformidad, que puede administrar a través de la consola de Chef o de herramientas de línea de comandos como Knife. También funciona perfectamente con sus libros de recetas de Chef existentes.

AWS OpsWorks para Puppet Enterprise

- AWS OpsWorks para Puppet Enterprise es un servicio de administración de configuración completamente administrado que aloja Puppet Enterprise, un conjunto de herramientas de automatización de Puppet para la administración de infraestructuras y aplicaciones. OpsWorks también se encarga del mantenimiento del servidor maestro de Puppet, y realiza automáticamente las tareas de aplicación de parches, actualización y backup del servidor. OpsWorks elimina la necesidad de operar sistemas de administración de configuración propios y de preocuparse por el mantenimiento de la infraestructura. OpsWorks proporciona acceso a todas las características de Puppet Enterprise, que se administran a través de la consola de Puppet. También funciona perfectamente con el código de Puppet existente.

AWS OPSWORKS STACKS



- AWS OpsWorks Stacks es un servicio de administración de servidor y aplicaciones. Con OpsWorks Stacks, puede diseñar su aplicación como una stack para contener diferentes capas, como equilibrio de cargas, bases de datos y servidor de aplicaciones. Dentro de cada capa, puede aprovisionar instancias de Amazon EC2, activar el escalado automático y configurar sus instancias con recetas de Chef con Chef Solo. Esto le permite automatizar tareas como la instalación de paquetes y lenguajes de programación o marcos, configurar software, entre otras.



AWS OpsWorks

An integrated DevOps application management solution



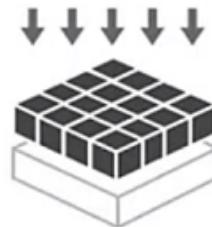
A **stack** represents the compute infrastructure and applications that you want to manage together.



A **layer** defines how to set up and configure a set of instances and related resources.



Decide how to scale: manually, with **24/7** instances, or automatically, with **load-based** or **time-based** instances.



Then deploy your app to specific instances and customize the deployment with Chef recipes.

Stacks

- Stack is the core AWS OpsWorks Stacks component.
- Stack is a container for AWS resources like EC2, RDS instances etc that have a common purpose and should be logically managed together
- Stack helps manage the resources as a group and also defines some default configuration settings, such as the instances' OS and AWS region
- Stacks can also be run in VPC to be isolated from direct user interaction
- Separate Stacks can be created for different environments like Dev, QA etc

Layers

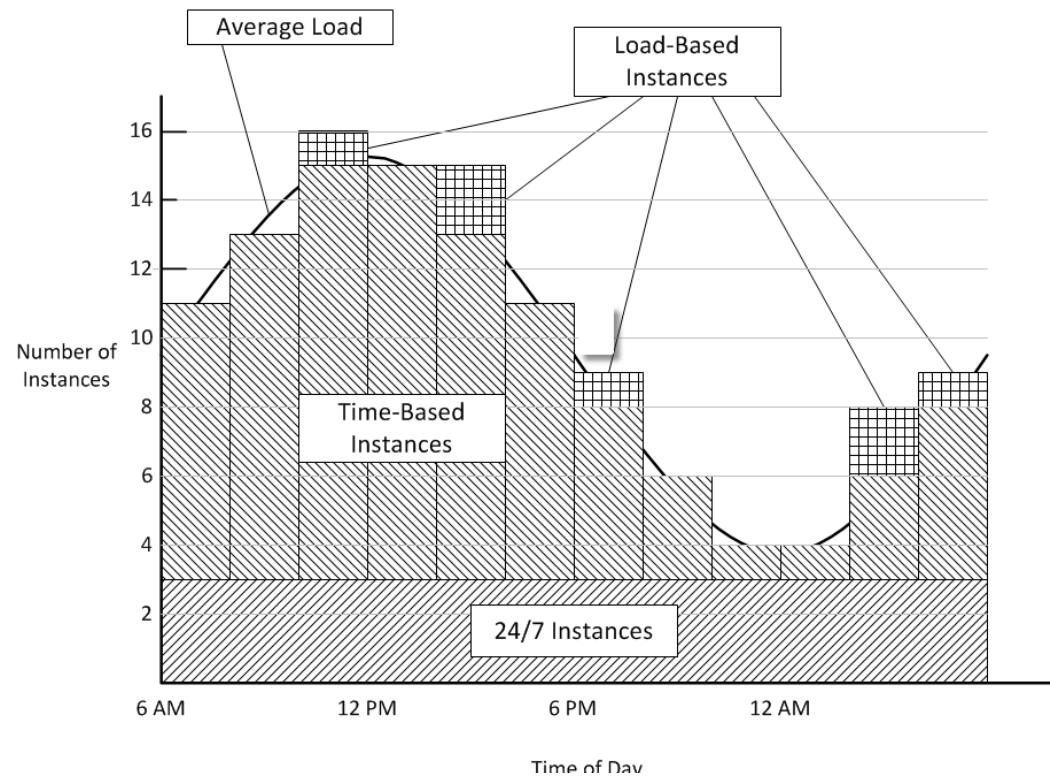
- Stacks help manage cloud resources in specialized groups called layers.
- A layer represents a set of EC2 instances that serve a particular purpose, such as serving applications or hosting a database server.
- Layers depend on Chef recipes to handle tasks such as installing packages on instances, deploying apps, and running scripts
- Custom recipes and related files is packaged in one or more cookbooks and stored in a cookbook repository such S3 or Git

RECIPES AND LIFECYCLE EVENTS

Layer	Assigned Recipes
Setup	myrecipe::default, myrecipe
Configure	myrecipe::default, myrecipe
Deploy	myrecipe::default, myrecipe
Undeploy	myrecipe::default, myrecipe
Shutdown	myrecipe::default, myrecipe

- Layers depend on [Chef recipes](#) to handle tasks such as installing packages on instances, deploying apps, running scripts, and so on. One of the key AWS OpsWorks Stacks features is a set of *lifecycle events*—Setup, Configure, Deploy, Undeploy, and Shutdown—which automatically run a specified set of recipes at the appropriate time on each instance.
- Each layer can have a set of recipes assigned to each lifecycle event, which handle a variety of tasks for that event and layer. For example, after an instance that belongs to a web server layer finishes booting, AWS OpsWorks Stacks does the following.
 - Runs the layer's Setup recipes, which could perform tasks such as installing and configuring a web server.
 - Runs the layer's Deploy recipes, which deploy the layer's applications from a repository to the instance and perform related tasks, such as restarting the service.
 - Runs the Configure recipes on every instance in the stack so each instance can adjust its configuration as needed to accommodate the new instance.
- For example, on an instance running a load balancer, a Configure recipe could modify the load balancer's configuration to include the new instance.
- If an instance belongs to multiple layers, AWS OpsWorks Stacks runs the recipes for each layer so you can, for example, have an instance that supports a PHP application server and a MySQL database server.
- If you have implemented recipes, you can assign each recipe to the appropriate layer and event and AWS OpsWorks Stacks automatically runs them for you at the appropriate time. You can also run recipes manually, at any time.

INSTANCES



- An *instance* represents a single computing resource, such as an Amazon EC2 instance. It defines the resource's basic configuration, such as operating system and size. Other configuration settings, such as Elastic IP addresses or Amazon EBS volumes, are defined by the instance's layers. The layer's recipes complete the configuration by performing tasks such as installing and configuring packages and deploying apps.
- You can use AWS OpsWorks Stacks to create instances and add them to a layer. When you start the instance, AWS OpsWorks Stacks launches an Amazon EC2 instance using the configuration settings specified by the instance and its layer. After the Amazon EC2 instance has finished booting, AWS OpsWorks Stacks installs an agent that handles communication between the instance and the service and runs the appropriate recipes in response to lifecycle events.
- AWS OpsWorks Stacks supports the following instance types, which are characterized by how they are started and stopped.
 - 24/7 instances are started manually and run until you stop them.
 - Time-based instances are run by AWS OpsWorks Stacks on a specified daily and weekly schedule. They allow your stack to automatically adjust the number of instances to accommodate predictable usage patterns.
 - Load-based instances are automatically started and stopped by AWS OpsWorks Stacks, based on specified load metrics, such as CPU utilization.
- They allow your stack to automatically adjust the number of instances to accommodate variations in incoming traffic. Load-based instances are available only for Linux-based stacks.

Apps

An AWS OpsWorks Stacks *app* represents code that you want to run on an application server. The code itself resides in a repository such as an Amazon S3 archive; the app contains the information required to deploy the code to the appropriate application server instances.

When you deploy an application, AWS OpsWorks Stacks triggers a Deploy event, which runs each layer's Deploy recipes. AWS OpsWorks Stacks also installs [stack configuration and deployment attributes](#) that contain all of the information needed to deploy the app, such as the app's repository and database connection data. You must implement custom recipes that retrieve the app's deployment data from the stack configuration and deployment attributes and handle the deployment tasks.

USING AUTO HEALING TO REPLACE FAILED INSTANCES

Layer **windowscompute**

General Settings Recipes Network EBS Volumes Security

Settings

Name	windowscompute
Short name	compute
Instance shutdown timeout	120
Auto healing enabled	<input checked="" type="checkbox"/> Yes

- Every instance has an AWS OpsWorks Stacks agent that communicates regularly with the service. AWS OpsWorks Stacks uses that communication to monitor instance health. If an agent does not communicate with the service for more than approximately five minutes, AWS OpsWorks Stacks considers the instance to have failed.

- Auto healing is set at the layer level; you can change the auto healing setting by editing layer settings, as shown in the following screenshot. After the auto-healed instance is back online, AWS OpsWorks Stacks triggers a [Configure lifecycle event](#) on all of the stack's instances. The associated [stack configuration and deployment attributes](#) include the instance's public and private IP addresses. Custom Configure recipes can obtain the new IP addresses from the node object.

- If you [specify an Amazon EBS volume](#) for a layer's instances, AWS OpsWorks Stacks creates a new volume and attaches it to each instance when the instance is started. If you later want to detach the volume from an instance, use the [Resources](#) page.

- When AWS OpsWorks Stacks auto heals one of a layer's instances, it handles volumes in the following way:

- If the volume was attached to the instance when the instance failed, the volume and its data are saved, and AWS OpsWorks Stacks attaches it to the new instance.

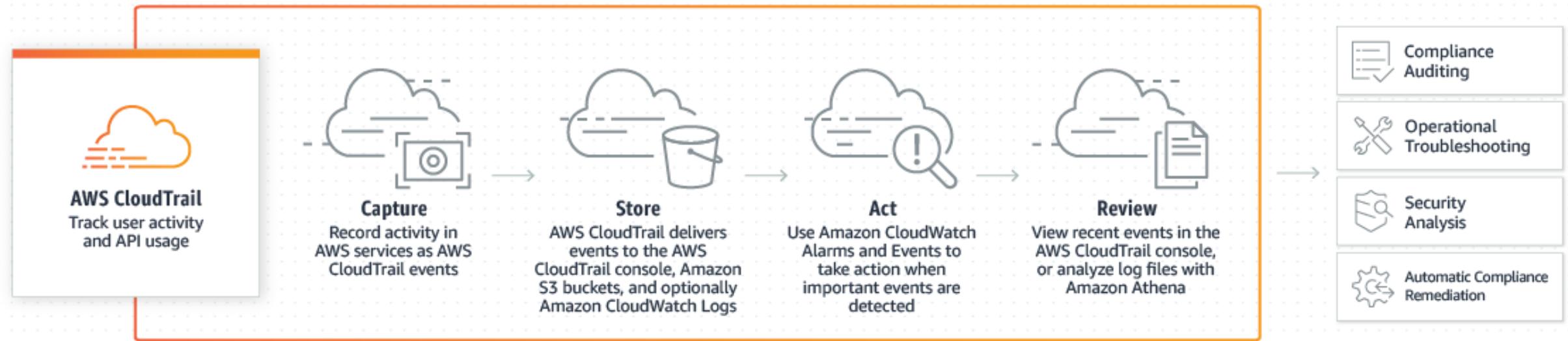
- If the volume was not attached to the instance when the instance failed, AWS OpsWorks Stacks creates a new, empty volume with the configuration specified by the layer, and attaches that volume to the new instance.

OpsWorks Databags & berkshelf

- OpsWorks Stacks exposes settings to recipes as Chef data bag content.
 - A data bag is a Chef concept.
 - It is a global variable stored as JSON data on an instance
 - the JSON data is accessible from Chef.
 - Like data bag stores global variables as
 - app's source URL
 - instance's hostname
 - associated stack's VPC identifier.
 - OpsWorks Stacks stores its data bags on each stack's instances.
 - On Linux instances, data bags is at /var/chef/runs/run-ID/data_bags directory.
 - On Windows instances, data bags is at drive:\chef\runs\run-id\data_bags directory.
 - These directories include a set of data bags (subdirectories).
 - Each data bag contains zero or more data bag items, and are JSON-formatted files that have sets of data bag content.
- Data bag content includes
- String content
 - Boolean content
 - Number content
 - List content
 - Has the form of comma-separated values enclosed in square brackets (no quotes), such as ['80', '443']
 - JSON objects
 - which contain additional data bag content, such as "my-app": {"elastic_ip": null,...}.

Topics of discussion





AWS Config vs CloudTrail

- AWS Config reports on **WHAT** has changed, whereas CloudTrail reports on **WHO** made the change, **WHEN**, and from **WHICH** location.
- AWS Config focuses on the configuration of the AWS resources and reports with detailed snapshots on **HOW** the resources have changed, whereas CloudTrail focuses on the events, or API calls, that drive those changes. It focuses on the user, application, and activity performed on the system.

Always on

AWS CloudTrail is enabled on all AWS accounts and records your account activity upon account creation. You can view and download the last 90 days of your account activity for create, modify, and delete operations of supported services without the need to manually setup CloudTrail.

Event history

You can view, search, and download your recent AWS account activity. This allows you to gain visibility into changes in your AWS account resources so you can strengthen your security processes and simplify operational issue resolution.

Multi-region configuration

You can configure AWS CloudTrail to deliver log files from multiple regions to a single Amazon S3 bucket for a single account. A configuration that applies to all regions ensures that all settings apply consistently across all existing and newly launched regions. For detailed instructions, see [Aggregating CloudTrail Log Files to a Single Amazon S3 Bucket](#) in the *AWS CloudTrail User Guide*.

Log file integrity validation

You can validate the integrity of AWS CloudTrail log files stored in your Amazon S3 bucket and detect whether the log files were unchanged, modified, or deleted since CloudTrail delivered them to your Amazon S3 bucket. You can use [log file integrity validation](#) in your IT security and auditing processes.

Log file encryption

By default, AWS CloudTrail encrypts all log files delivered to your specified Amazon S3 bucket using Amazon S3 server-side encryption (SSE). Optionally, add a layer of security to your CloudTrail log files by encrypting the log files with your AWS Key Management Service (AWS KMS) key. Amazon S3 automatically decrypts your log files if you have decrypt permissions. For more information, see [encrypting log files using your KMS key](#).

Data events

Data events provide insights into the resource (“data plane”) operations performed on or within the resource itself. Data events are often high volume activities and include operations such as Amazon S3 object level APIs and AWS Lambda function invoke APIs. For example, you can log API actions on Amazon S3 objects and receive detailed information such as the AWS account, IAM user role, and IP address of the caller, time of the API call, and other details. You can also record activity of your Lambda functions, and receive details on Lambda function executions, such as the IAM user or service that made the Invoke API call, when the call was made, and which function was executed.

Management events

Management events provide insights into the management (“control plane”) operations performed on resources in your AWS account. For example, you can log administrative actions such as creation, deletion, and modification of Amazon EC2 instances. For each event, you can get details such as the AWS account, IAM user role, and IP address of the user that initiated the action, time of the action, and which resources were affected.

Integrations

AWS Lambda

You can take advantage of the Amazon S3 bucket notification feature to direct Amazon S3 to publish object-created events to AWS Lambda. When CloudTrail writes logs to your S3 bucket, Amazon S3 can invoke your Lambda function to process the access records logged by CloudTrail.

Amazon CloudWatch Logs

Amazon CloudWatch Logs enables you to send management and data events recorded by CloudTrail to CloudWatch Logs. CloudWatch Logs allows you to create metric filters to monitor events, search events, and stream events to other AWS services, such as AWS Lambda and Amazon Elasticsearch Service.

Amazon CloudWatch Events

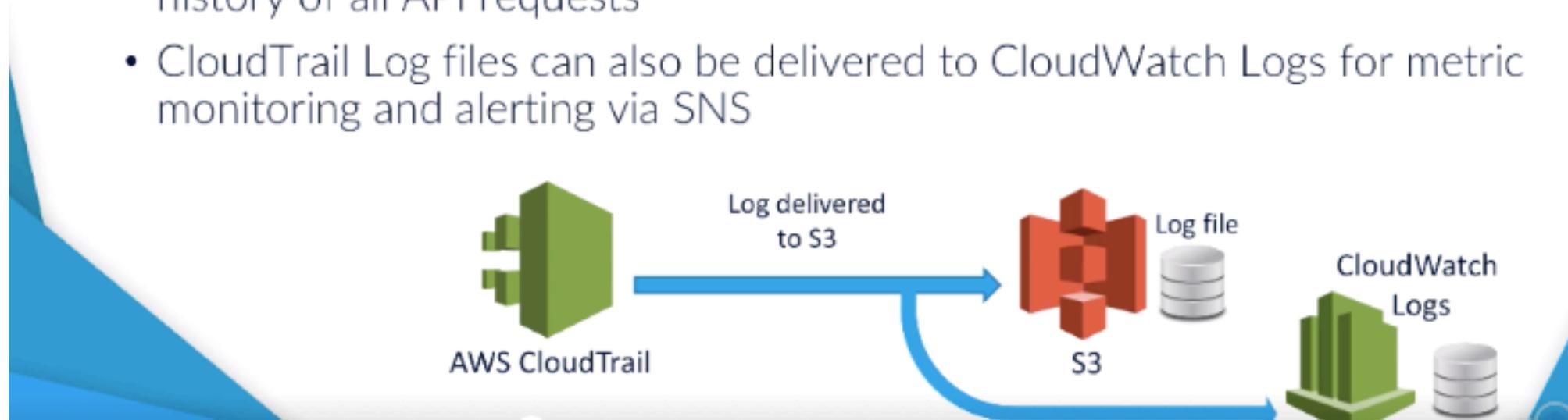
Amazon CloudWatch Events enables you to automatically respond to changes to your AWS resources. With CloudWatch Events, you are able to define actions to execute when specific events are logged by AWS CloudTrail. For example, if CloudTrail logs a change to an Amazon EC2 security group, such as adding a new ingress rule, you can create a CloudWatch Events rule that sends this activity to an AWS Lambda function. Lambda can then execute a workflow to create a ticket in your IT Helpdesk system.

CloudTrail Events

- Every API request captured is recorded as an 'event'
- Multiple events are recorded within CloudTrail Logs
- Events contain an array of associated metadata, for example:
 - Identity of the caller
 - Timestamp of request
 - Source IP address

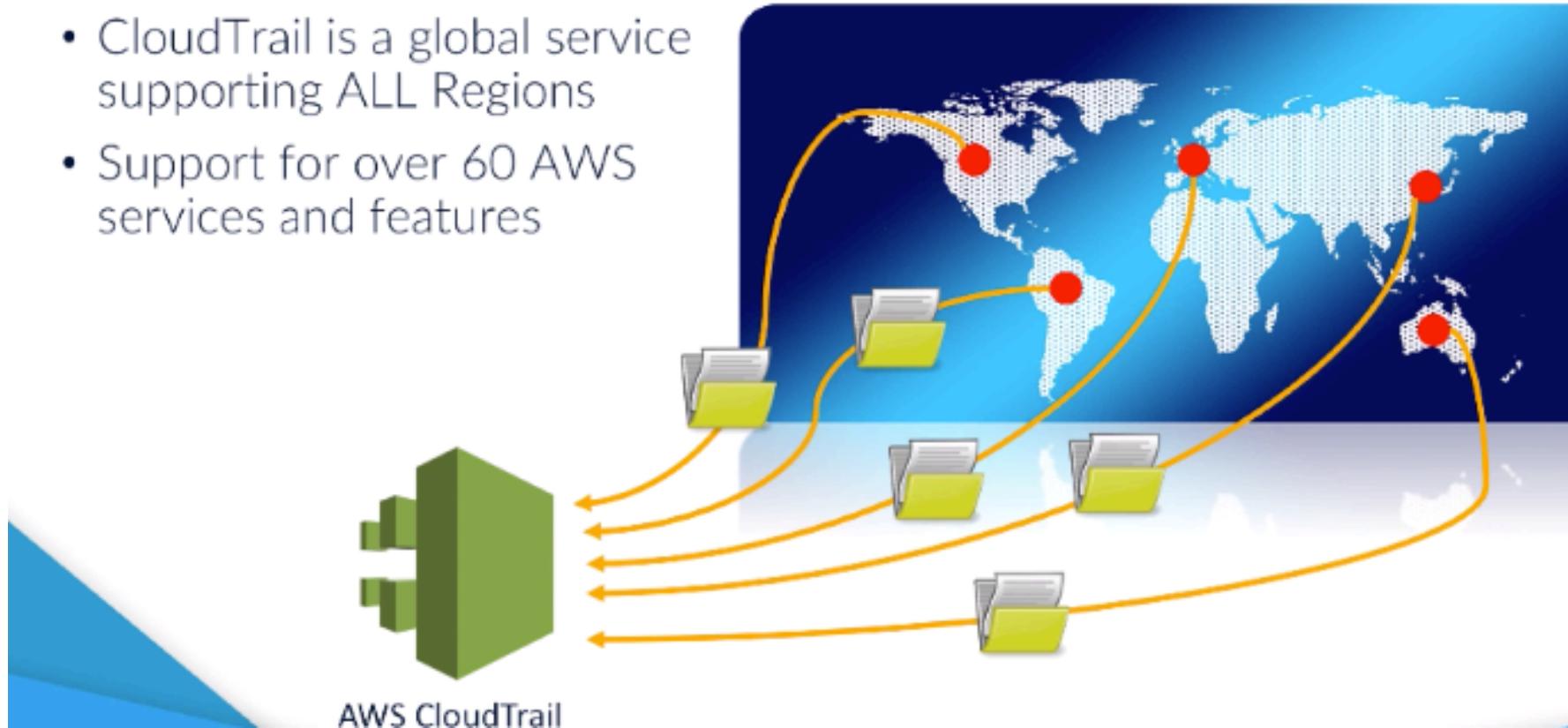
CloudTrail Logs

- New log files are created every 5 minutes
- Log files are delivered and stored within S3
- Log files can be stored for as long as required allowing you to review the history of all API requests
- CloudTrail Log files can also be delivered to CloudWatch Logs for metric monitoring and alerting via SNS



CloudTrail Infrastructure

- CloudTrail is a global service supporting ALL Regions
- Support for over 60 AWS services and features



Use cases for captured data (1 of 2)

- Effective for security analysis
 - Monitor restricted API calls
 - Notification of threshold breaches
- Resolve day to day operational issues
 - Filtering mechanisms for isolating data
 - Quicker root cause identification
 - Speedy resolution



Use cases for captured data (2 of 2)

- Able to track changes to your AWS infrastructure
- CloudTrail logs can be used as evidence for various compliance and governance controls

- ISO
- PCI DSS
- FedRamp

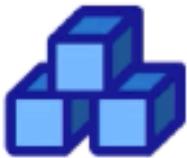


- *Security at Scale: Logging in AWS How AWS CloudTrail can help you achieve compliance by logging API calls and changes to resources*

https://d0.awsstatic.com/whitepapers/compliance/AWS_Security_at_Scale_Logging_in_AWS_Whitepaper.pdf

Core Features & Services

- Trails
 - Simple Storage Service (S3)
 - Logs
 - Key Management Service (KMS)
 - Simple Notification Service (SNS)
- CloudWatch Logs
 - Event Selectors
 - Tags
 - Events
 - API Activity Filters



Trails



S3



Logs



KMS



SNS



CloudWatch



Event
Selectors



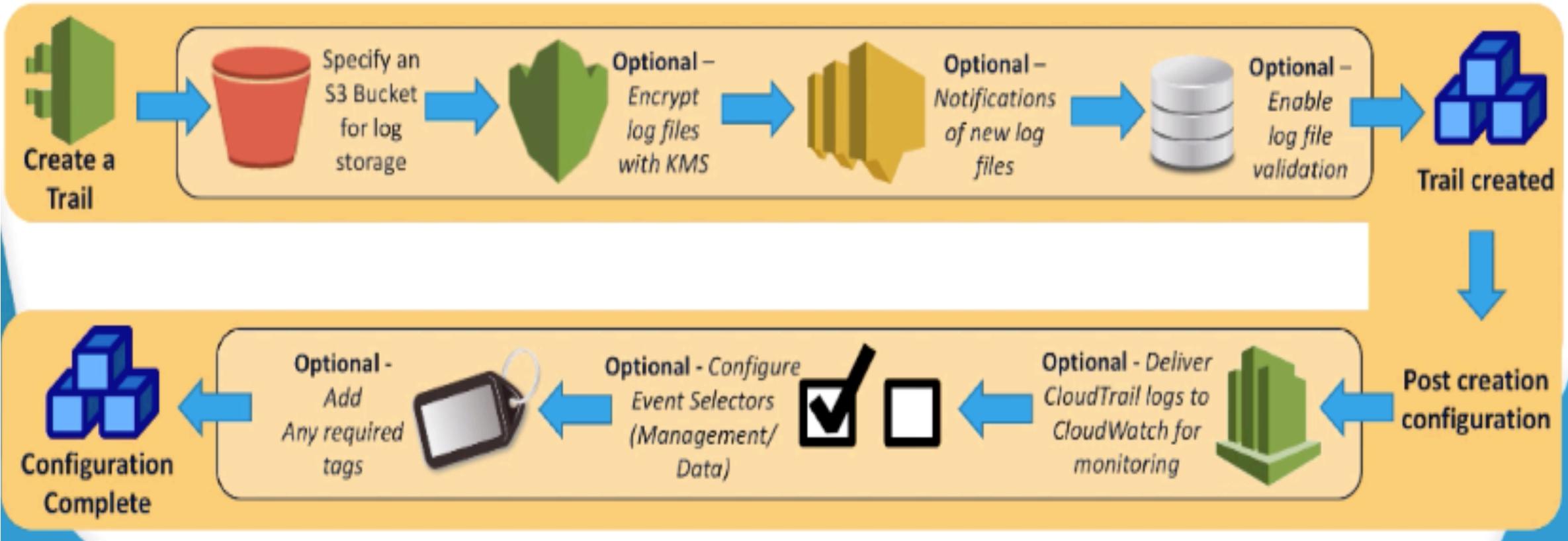
Tags



Events

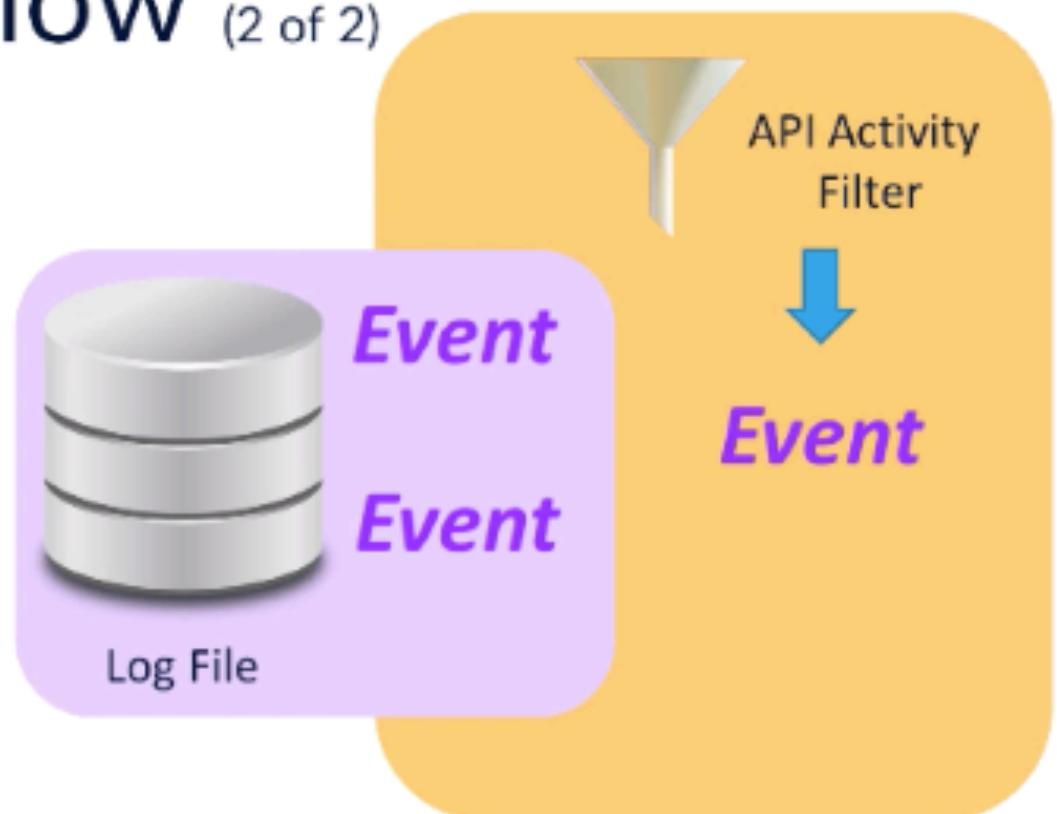


API Activity

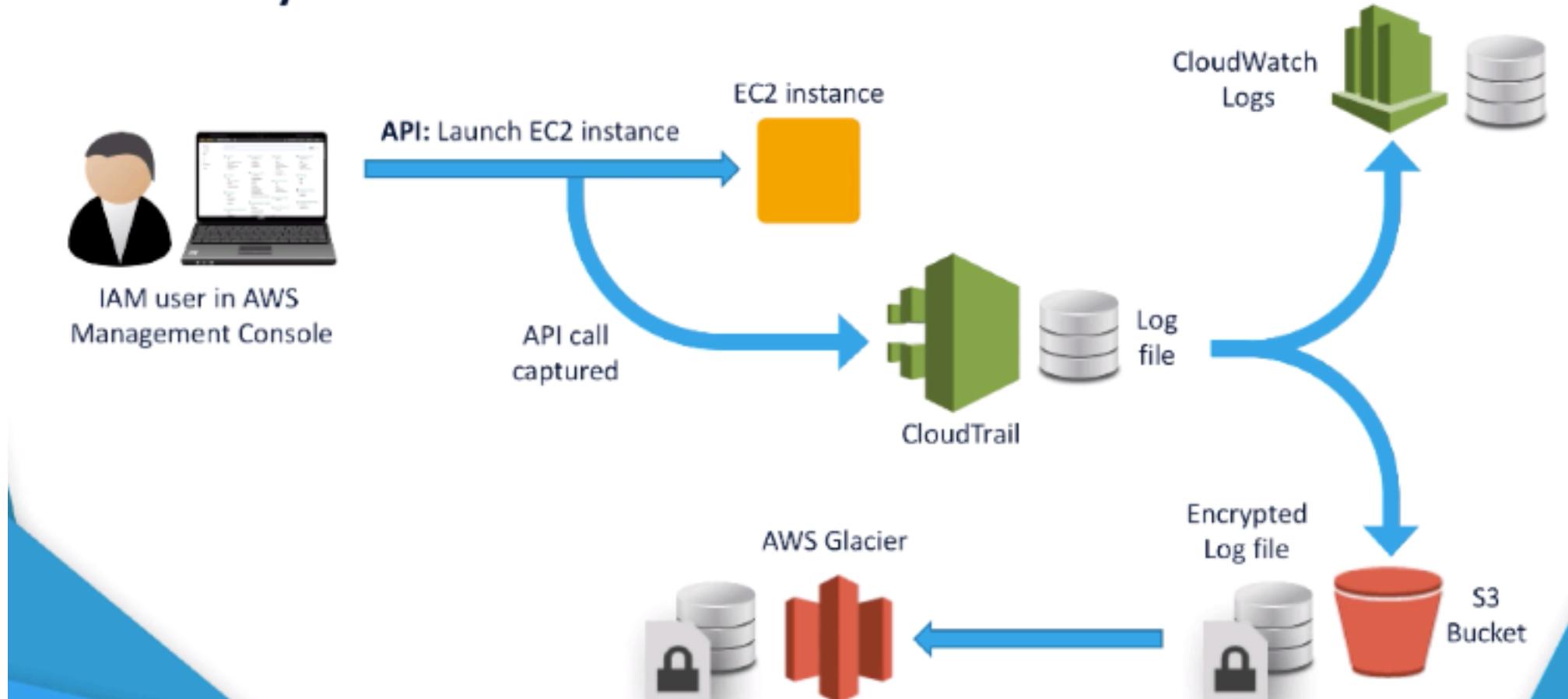


CloudTrail Process Flow (2 of 2)

API Requests



Lifecycle of an API call in CloudTrail



Granting Access to CloudTrail

- Requires proper permissions and authorization to use CloudTrail
- Create your own access policy or use AWS Managed policies
- For more information on IAM Policies:
 - Course: *Understanding of AWS Authentication, Authorization & Accounting*
 - Blog post on IAM Policies: <http://cloudacademy.com/blog/aws-iam-policy/>
- AWS Managed policies for CloudTrail:
 - AWSCloudTrailFullAccess
 - AWSCloudTrailReadOnlyAccess

```
{  
    "Action": [  
        "sns:AddPermission",  
        "sns>CreateTopic",  
        "sns>DeleteTopic",  
        "sns>ListTopics",  
        "sns:SetTopicAttributes",  
        "sns:GetTopicAttributes"  
    ],  
    "Effect": "Allow",  
    "Resource": "*"  
},  
{  
    "Action": [  
        "s3>CreateBucket",  
        "s3>DeleteBucket",  
        "s3>ListAllMyBuckets",  
        "s3>PutBucketPolicy",  
        "s3>ListBucket",  
        "s3>GetObject",  
        "s3>GetBucketLocation",  
        "s3>GetBucketPolicy"  
    ],  
    "Effect": "Allow",  
    "Resource": "*"  
}
```

SNS Permissions

S3 Permissions

S3 Bucket Policy

- Specify a bucket for your CloudTrail logs
 - Create a new S3 Bucket
 - Use an existing S3 Bucket

Create Trail

Trail name* CA-Demo

Apply trail to all regions Yes No ⓘ

Create a new S3 bucket Yes No

S3 bucket* Bucket Name ⓘ



Creating a new S3 Bucket



- CloudTrail configures and applies a Bucket Policy with the relevant permissions allowing logs to be delivered to the Bucket
- Attributes configured by CloudTrail
 - The allowed Statement IDs (SIDs)
 - The folder name where the log files will be saved
 - The service principal name for the current and future CloudTrail supported regions
 - The Bucket name
 - The optional prefix if one was specified
 - The ID of the owning account

Bucket Policy Template

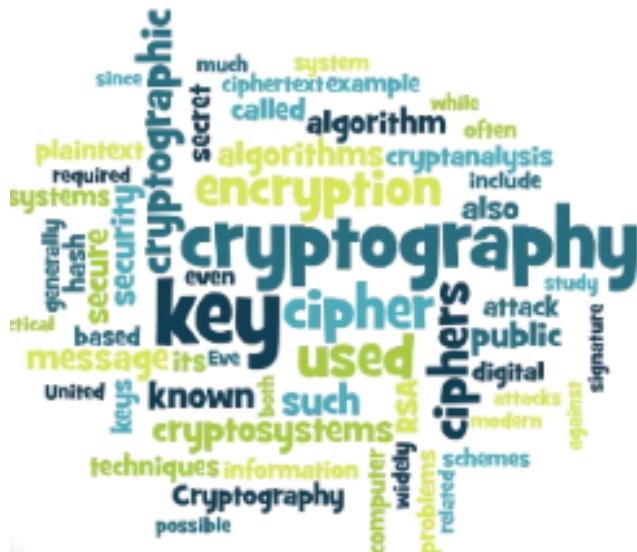
```
        "Sid": "AWSCloudTrailAclCheck20150319",
        "Effect": "Allow",
        "Principal": {
            "Service": "cloudtrail.amazonaws.com"
        },
        "Action": "s3:GetBucketAcl",
        "Resource": "arn:aws:s3:::myBucketName"
    },
    {
        "Sid": "AWSCloudTrailWrite20150319",
        "Effect": "Allow",
        "Principal": {
            "Service": "cloudtrail.amazonaws.com"
        },
        "Action": "s3:PutObject",
        "Resource": "arn:aws:s3:::myBucketName/[optional prefix]/AWSLogs/myAccount
        "Condition": {
            "StringEquals": {
                "s3:x-amz-acl": "bucket-owner-full-control"
            }
        }
    }
}
```



CloudTrail & Key Management Service

- Specific permissions are required to decrypt logs that have been encrypted with KMS, these being:
 - Decrypt permissions must be given on the Customer Master Key (CMK) Policy
 - S3 read permissions
- KMS adds another layer of encryption to CloudTrail Logs files
- The default encryption for CloudTrail log files is SSE-S3

Server-Side Encryption (SSE) - KMS



- SSE-KMS is configured during Trail creation
- Two options when creating your Customer Master Key (CMK)
 1. Create a new KMS Key
 2. Select an existing Key

Create a new KMS key Yes No

KMS key* ⓘ

KMS key and S3 bucket must be in the same region.



SSE-KMS Policy for CMK

- AWS template:

<http://docs.aws.amazon.com/awscloudtrail/latest/userguide/default-cmk-policy.html>

```
"Sid": "Allow principals in the account to decrypt log files",
"Effect": "Allow",
"Principal": {"AWS": "*"},
>Action": [
    "kms:Decrypt",
    "kms:ReEncryptFrom"
],
"Resource": "*",
"Condition": {
    "StringEquals": {"kms:CallerAccount": "aws-account-id"},
    "StringLike": {"kms:EncryptionContext:aws:cloudtrail:arn": "arn:aws:cloudtrail:*:aws-account-id:trail/*"}
}
```

CloudTrail Logs



What is a Log File?

- Log files written as JSON file format
- A new event is written for each API call
- A new log file is created every 5 minutes

```
"apiVersion": "20140328",
"awsRegion": "eu-west-1",
"eventID": "a32bce3a-035c-44cf-89f2-c293b2d3579d",
"eventName": "DescribeMetricFilters",
"eventSource": "logs.amazonaws.com",
"eventTime": "2017-01-03T13:28:10Z",
"eventType": "AwsApiCall",
"eventVersion": "1.04",
"recipientAccountId": "7           5",
"requestID": "786373df-d1b8-11e6-a1ed-67e605a8476f",
"requestParameters": {
    "limit": 50
},
"responseElements": null,
"sourceIPAddress": "90.213.22.234",
"userAgent": "signin.amazonaws.com",
"userIdentity": {
    "accessKeyId": "ASIAIHDOOCV6DNS3CFAQ",
    "accountId": "7           5",
    "arn": "arn:aws:iam::7           5:user/Cloudacademy",
    "invokedBy": "signin.amazonaws.com",
    "principalId": "AIDAIUEVSWLKPICM4ZIQ",
    "sessionContext": "
```

Events in a Log file

- **eventName:** The name of the actual API that was called
- **eventSource:** The service as to which the API called was made against
- **eventTime:** The time that the API call was made
- **SourceIPAddress:** The source IP address of the requester who made the API call
- **userAgent:** The agent method that the request was made through, for example:
 - Signin.amazonaws.com - From within the AWS Management Console
 - Console.amazonaws.com – Made by the root user of the account
 - Lambda.amazonaws.com – Made with AWS Lambda
- **userIdentity:** This contains a larger set of attributes that provides information on the identity that made the API request

```
"apiVersion": "20140328",
"awsRegion": "eu-west-1",
"eventID": "149c5ef5-ed9d-46c3-8e8d-3b98fbafda5b",
"eventName": "DescribeMetricFilters",
"eventSource": "logs.amazonaws.com",
"eventTime": "2017-01-03T13:33:13Z",
"eventType": "AwsApiCall",
"eventVersion": "1.04",
"recipientAccountId": "730739171055",
"requestID": "2ce52b1b-d1b9-11e6-be5e-2b215638ee5c",
"requestParameters": {
    "limit": 50
},
"responseElements": null,
"sourceIPAddress": "90.213.22.234",
"userAgent": "signin.amazonaws.com",
"userIdentity": {
    "accessKeyId": "ASIAIHD0OCV6DNS3CFAQ",
    "accountId": "730739171055",
    "arn": "arn:aws:iam::730739171055:user/Cloudacademy",
    "invokedBy": "signin.amazonaws.com",
    "principalId": "AIDAIEUEVSWLKPICM4ZIQ",
    "sessionContext": {
        "attributes": {
            "creationDate": "2017-01-03T11:15:25Z",
            "mfaAuthenticated": "false"
        }
    },
    "type": "IAMUser",
    "userName": "Cloudacademy"
```

Log File Naming Convention

- Log files have a standard naming convention of:

AccountID_CloudTrail_RegionName_YYYYMMDDTHHmmZ_UniqueString.FileNameFormat

- AWS Account ID
- Service creating the log file
- Region where the log file came from
- Year, month and day the log file was created
- Time
 - Hour and minutes of when the log was created
 - Indicates time is in UTC time zone
 - Random 16 digit alphanumeric character string
 - Default of json.gz

S3 Bucket Folder Structure

- S3 Bucket Folder Structure for Log Files

BucketName/prefix/AWSLogs/AccountId/CloudTrail/RegionName/YYYY/MM/DD

- Bucket selected during Trail creation
- Optional - to aid with organization
- Fixed folder name
- AWS Account ID
- Service creating the logs
- Region where the log was created
- Year the log was created
- Month the log was created
- Day the log was created



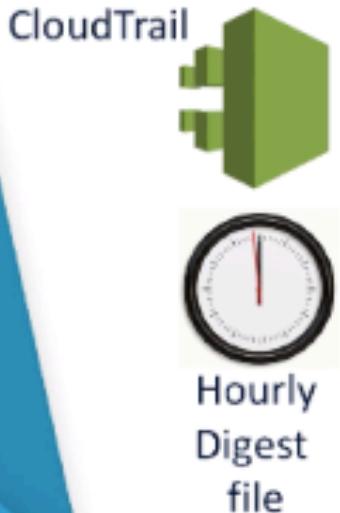
CloudTrail logs from multiple accounts

Accessing Cross-Account Log Files

1. Create Roles for each AWS Account in the Primary AWS Account
2. Assign access policy to each Role allowing only a specific AWS Account access
3. Users in secondary AWS Account would need to Assume one of these Roles for their corresponding AWS Account Log Files



Log File Integrity Process



1. A hash is created for your log file by CloudTrail
 - A hash value is a unique set of characters
 - CloudTrail uses the SHA-256 algorithm
2. CloudTrail creates a new digest file every hour to verify logs have not changed
3. Digest files contain details of all the logs delivered within the last hour
 - Stored in the same bucket as log files
 - Signed by a private key of a public and private key pair
4. To verify the integrity, the public key of this same key pair is used

Verifying Log File Integrity

- Verification of the log files can only be achieved via programmatic access
- Verify using the AWS CLI, by issuing the following command:

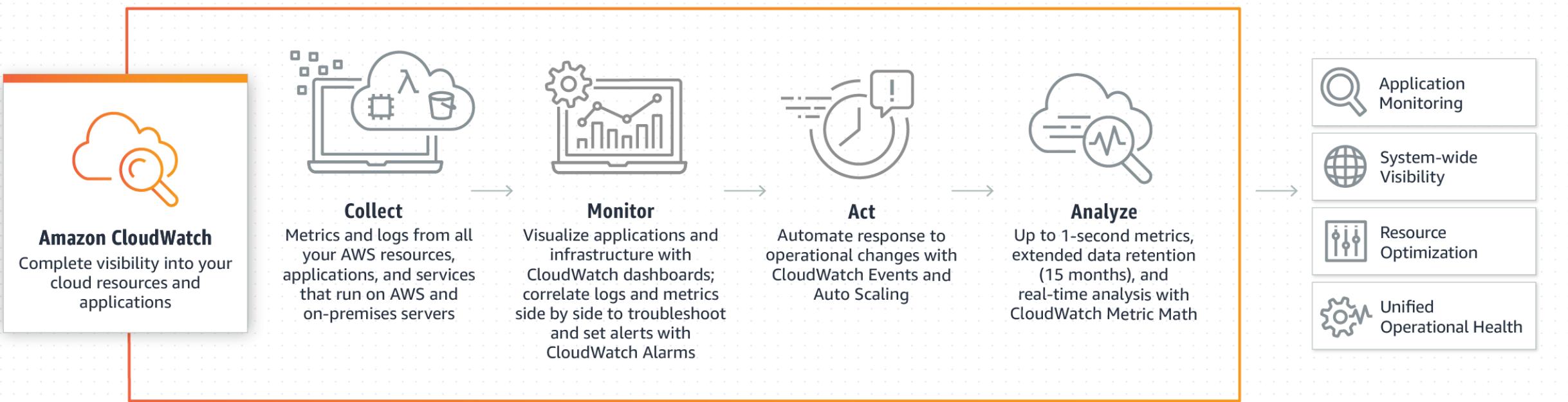
```
aws cloudtrail validate-logs --trail-arn <trailARN> --start-time <start-time>  
[--end-time <end-time>] [--s3-bucket <bucket-name>] [--s3-prefix <prefix>]  
[--verbose]
```

Digest File Folder Structure in S3

- The folder structure is very similar to the CloudTrail
`S3-bucket-name/AWSLogs/your-aws-account-id/CloudTrail-Digest/ AWSregionname/digest-end-year/digest-end-month/digest-end-date/`

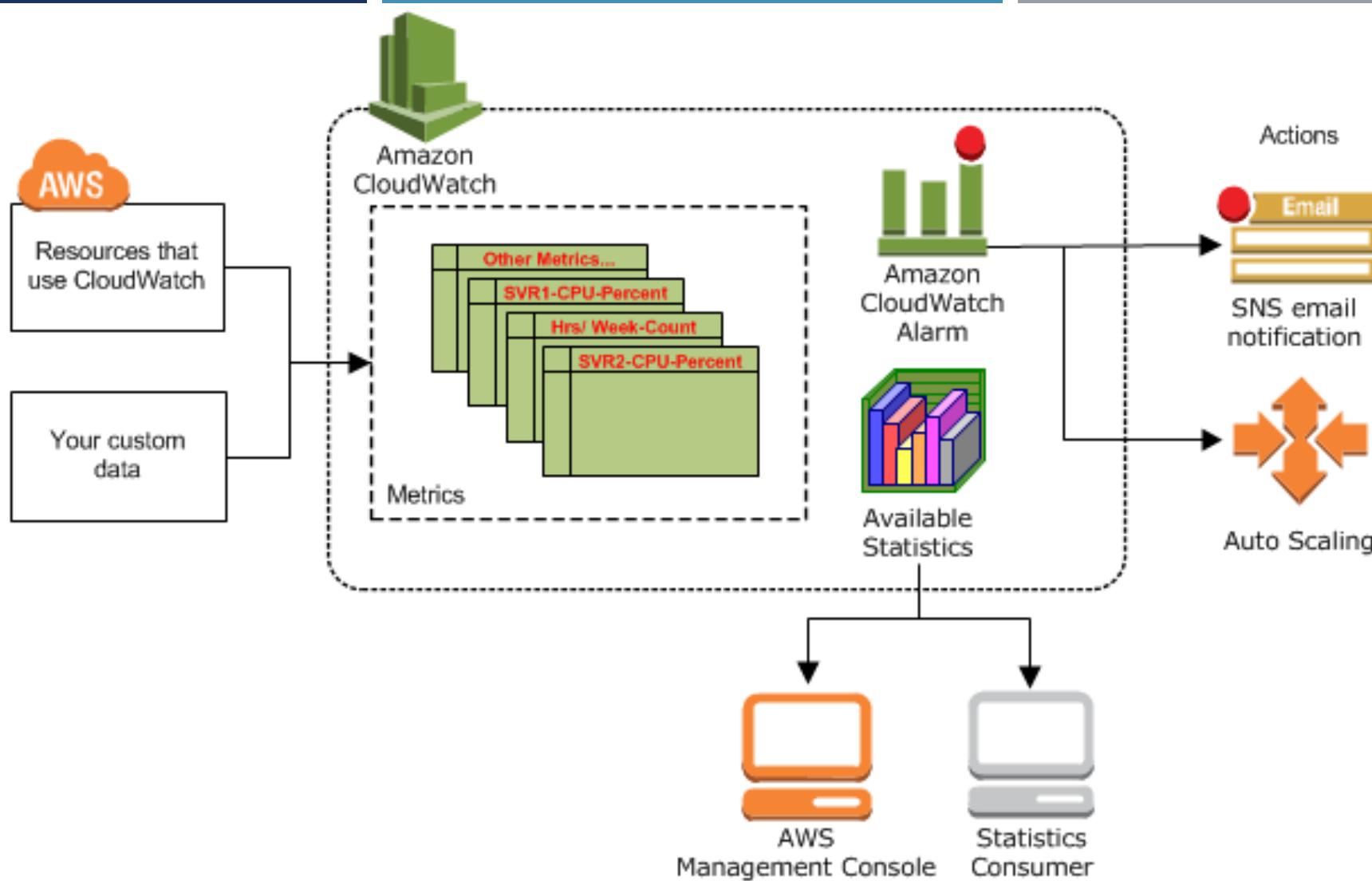
Amazon CloudWatch es un servicio de monitorización y administración que suministra datos e información procesable para aplicaciones y recursos de infraestructura locales, híbridos y de AWS. Con CloudWatch, puede recopilar y obtener acceso a todos los datos de rendimiento y operaciones en formato de registros y métricas a partir de una sola plataforma. Esto le permite superar el desafío de monitorear aplicaciones y sistemas individuales aislados (servidor, red, base de datos, etc.).

CloudWatch le permite monitorizar una stack completa (aplicaciones, infraestructura y servicios) y utilizar alarmas, registros y datos de eventos para tomar acciones automatizadas y disminuir el tiempo de resolución (MTTR). Esto libera recursos importantes y le permite enfocarse en la creación de aplicaciones y valor comercial.



CloudWatch Introduction

- Two tiers of pricing
 - Basic Monitoring – Included with all resources in all regions
 - Enhanced Monitoring – Priced per resource by region
- Basic
 - 3 Dashboards / 50 Metrics / 5 minute refresh rate
- Enhanced
 - Priced by Region -
 - \$3.00/Dashboard, Unlimited Metrics priced in unit quantities of 10,000, 240,000, 750,000, and 1,000,000



NAMESPACES

Servicio	Espacio de nombres	Documentación
Amazon API Gateway	AWS/ApiGateway	Monitorizar la ejecución de la API con Amazon CloudWatch
AppStream 2.0	AWS/AppStream	Monitoreo de recursos de Amazon AppStream 2.0
AWS AppSync	AWS/AppSync	Métricas de CloudWatch
Amazon Athena	AWS/Athena	Monitorización de consultas de Athena con métricas de CloudWatch
AWS Billing and Cost Management	AWS/Billing	Monitorización de cargos con alertas y notificaciones
ACM Private CA	AWS/ACMPvtCA	Métricas de CloudWatch admitidas
Amazon CloudFront	AWS/CloudFront	Monitorización de la actividad de CloudFront mediante CloudWatch
AWS CloudHSM	AWS/CloudHSM	Obtención de métricas de CloudWatch
Amazon CloudSearch	AWS/CloudSearch	Monitorización de un dominio Amazon CloudSearch con Amazon CloudWatch

- Es un contenedor de métricas de CloudWatch. Las métricas en distintos espacios de nombres están aisladas entre sí, de forma que las métricas de distintas aplicaciones no estén acumuladas por error en las mismas estadísticas.
- No hay ningún espacio de nombres predeterminado. Debe especificar un espacio de nombres para cada punto de datos que publique en CloudWatch. Puede especificar un nombre de espacio de nombres al crear una métrica. Estos nombres deben contener caracteres XML válidos y tener menos de 256 caracteres de longitud. Los caracteres posibles son: caracteres alfanuméricos (0-9A-Za-z), punto (.), guion (-), guion bajo (_), barra inclinada (/), almohadilla (#) y dos puntos (:).
- Los espacios de nombres de AWS utilizan normalmente la siguiente convención de nomenclatura: AWS/service. Por ejemplo, Amazon EC2 utiliza el espacio de nombres AWS/EC2. Para obtener la lista de espacios de nombres de AWS, consulte [Servicios de AWS que publican métricas de CloudWatch](#).

METRICS

The screenshot shows the AWS CloudWatch Metrics console. At the top, there are three tabs: 'All metrics' (highlighted in orange), 'Graphed metrics', and 'Graph options'. Below the tabs is a search bar with the placeholder 'Search for any metric, dimension or resource id'. The main area displays a grid of service metrics:

Service	Metrics
EBS	117 Metrics
EC2	316 Metrics
EFS	7 Metrics
ELB	210 Metrics
ElasticBeanstalk	8 Metrics
RDS	60 Metrics
S3	4 Metrics

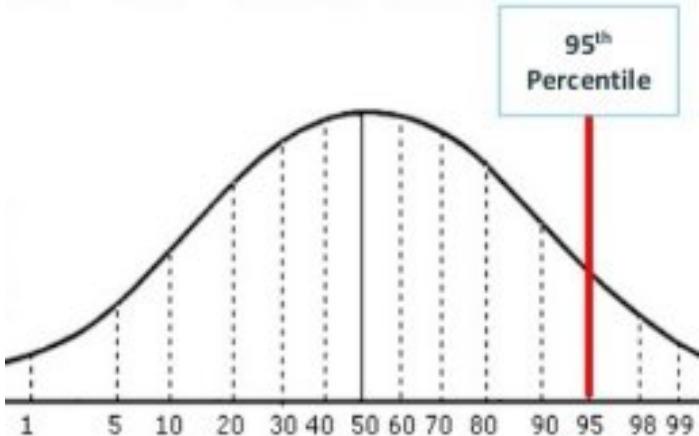
■ Las métricas son el concepto fundamental en CloudWatch. Una métrica representa una serie de puntos de datos ordenados por tiempo que se publican a CloudWatch. Una métrica es una variable que hay que monitorizar y los puntos de datos son los valores de esa variable a lo largo del tiempo. Por ejemplo, el uso de la CPU de una determinada instancia EC2 es una métrica proporcionada por Amazon EC2. Los propios puntos de datos pueden proceder de cualquier aplicación o actividad empresarial desde la que recopile datos.

■ Los servicios de AWS envían métricas a CloudWatch y puede enviar sus propias métricas personalizadas a CloudWatch. Puede añadir los puntos de datos en cualquier orden y a la velocidad que elija. Puede recuperar estadísticas sobre dichos puntos de datos como un conjunto ordenado de datos de serie temporal.

■ Las métricas existen solo en la región en que se han creado. Las métricas no se pueden eliminar, pero vencen automáticamente a los 15 meses si no se publican datos nuevos. Los puntos de datos con más de 15 meses caducarán sucesivamente; a medida que se introducen nuevos puntos de datos, los datos con más de quince meses se eliminan.

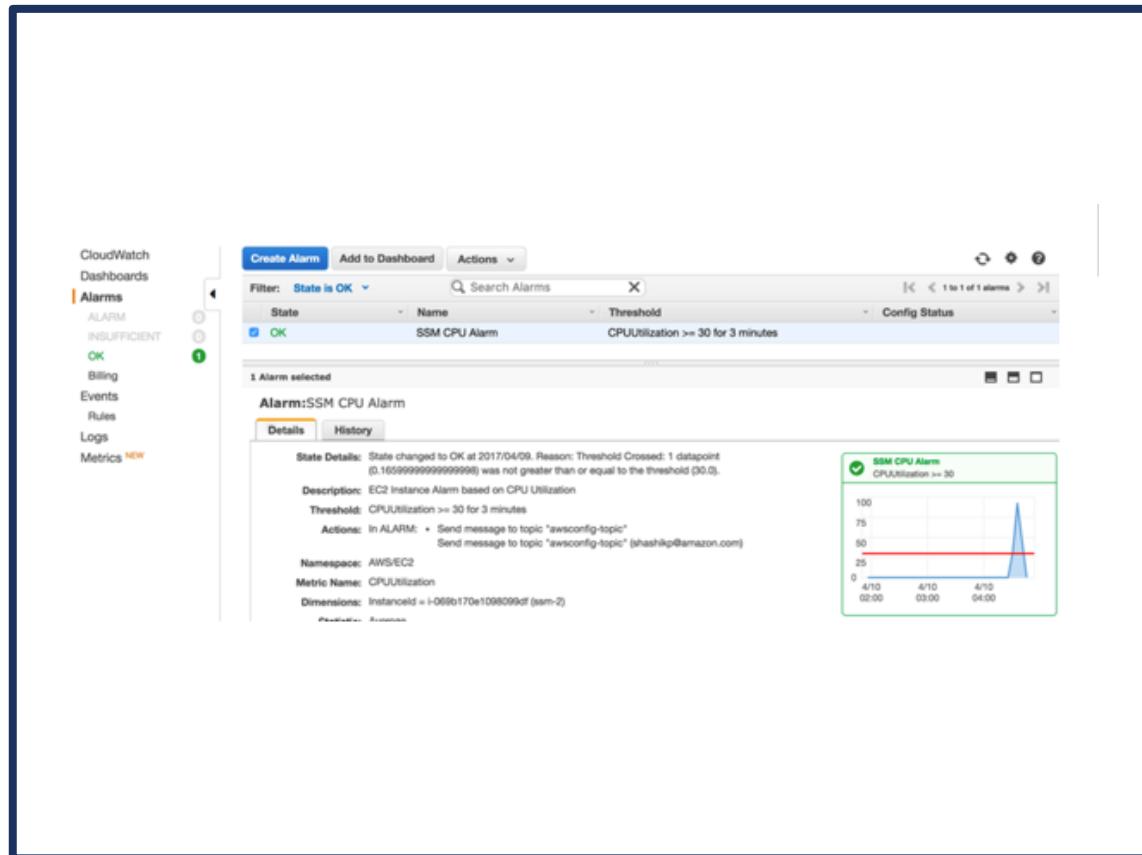
■ Las métricas se definen de forma exclusiva mediante un nombre, un espacio de nombres y cero o varias dimensiones. Cada punto de datos de una métrica tiene una marca temporal y, opcionalmente, una unidad de medida. Puede recuperar estadísticas de CloudWatch para cualquier métrica.

PERCENTILES



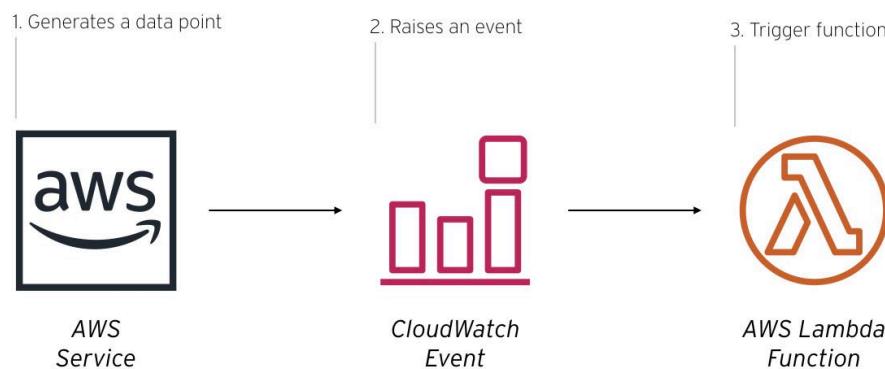
- Un *percentil* indica el peso relativo de un valor en un conjunto de datos. Por ejemplo, el percentil 95 significa que el 95 por ciento de los datos está por debajo de este valor y el 5 por ciento de los datos está por encima del mismo. Los percentiles le ayudan a entender mejor la distribución de los datos de métricas.
- Los percentiles se suelen utilizar para aislar anomalías. En una distribución normal, el 95% de los datos está dentro de dos desvíos estándar de la media y el 99,7% de los datos está dentro de tres desvíos estándar de la media. Cualquier dato que quede fuera de las tres desvíos estándar se suele considerar una anomalía ya que se aleja mucho del valor medio. Por ejemplo, suponga que está monitorizando la utilización de la CPU de las instancias EC2 para asegurarse de que sus clientes disfruten de una buena experiencia. Si monitoriza la media, esto puede ocultar anomalías. Si monitoriza el máximo, una única anomalía puede sesgar los resultados. Mediante los percentiles, puede monitorizar el percentil 95.^o de la utilización de la CPU para comprobar si hay instancias con una carga excepcionalmente alta.

ALARMA



■ Puede utilizar una *alarma* para iniciar automáticamente acciones en su nombre. Una alarma vigila una única métrica durante el período especificado y realiza una o varias acciones especificadas según el valor de la métrica relativo a un determinado umbral durante un período de tiempo. La acción es una notificación que se envía a un tema de Amazon SNS o a una política de Auto Scaling. También puede añadir alarmas a paneles.

EVENTO (CLOUDWATCH EVENTS)



Amazon CloudWatch Events delivers a near real-time stream of system events that describe changes in Amazon Web Services (AWS) resources. Using simple rules that you can quickly set up, you can match events and route them to one or more target functions or streams.

CloudWatch Events becomes aware of operational changes as they occur. CloudWatch Events responds to these operational changes and takes corrective action as necessary, by sending messages to respond to the environment, activating functions, making changes, and capturing state information.

You can also use CloudWatch Events to schedule automated actions that self-trigger at certain times using cron or rate expressions. For more information, see [Schedule Expressions for Rules](#).

CloudWatch Introduction

- Logging
 - CloudWatch also provides a repository for logging
- Alarms
 - Alarms are the process of becoming informed and acting
 - Think of alarms as pre-defined thresholds

CloudWatch Logs



CloudWatch is used to collate and collect metrics on resources, monitor their performance and respond to alerts.

Allows you to collect logs of your applications and a number of different AWS services.

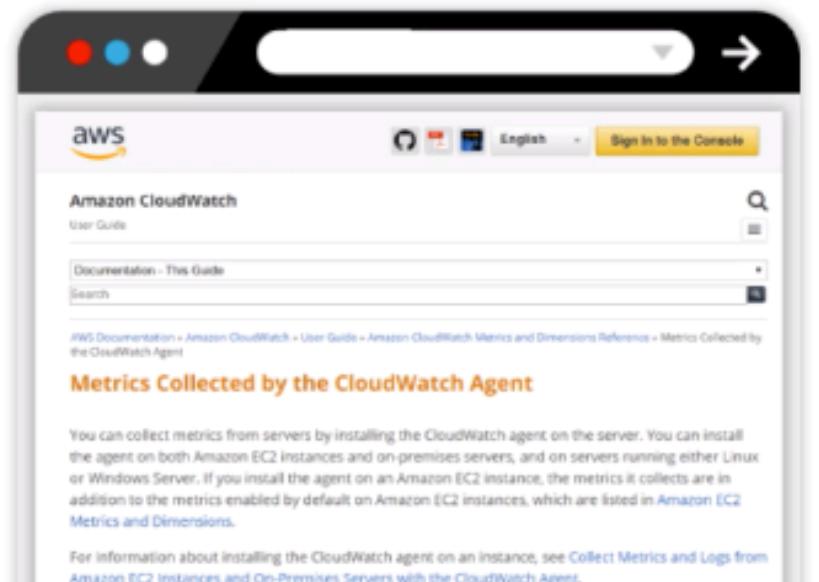
Provides the ability to monitor log streams in real-time and set up metric filters to search for specific events.

Unified CloudWatch Agent

The Unified CloudWatch Agent allows the collection of logs from EC2 instances as well from on-premise servers

OS versions supported:

- Amazon Linux version 2014.03.02 or later
- Ubuntu Server version 16.04 and 14.04
- CentOS version 7.0 and 6.5
- Red Hat Enterprise Linux (RHEL) version 7.4, 7.0, and 6.5
- Debian 8.0
- 64-bit versions of Windows Server 2016, Windows Server 2012, and Windows Server 2008



The screenshot shows a web browser window displaying the AWS Documentation site. The URL in the address bar is <https://docs.aws.amazon.com/AmazonCloudWatch/latest/monitoring/metrics-collected-by-CloudWatch-agent.html>. The page title is "Metrics Collected by the CloudWatch Agent". The content includes a brief description of how the agent collects metrics from servers, mentioning support for Amazon EC2 instances and on-premises servers running Linux or Windows Server. It also notes that on Amazon EC2 instances, metrics collected by the agent are in addition to those collected by default. A blue button at the bottom left contains the AWS logo and the same URL.

CloudWatch Agent Installation



Create a role and attach it to the instance with permissions to collect data from the instances in addition to interacting with SSM



Download and install the agent onto the EC2 instance



Configure and start the CloudWatch agent

Creating Roles

You will need to create two roles:



Used to install the agent and also to send the additional metrics gathered to CloudWatch.

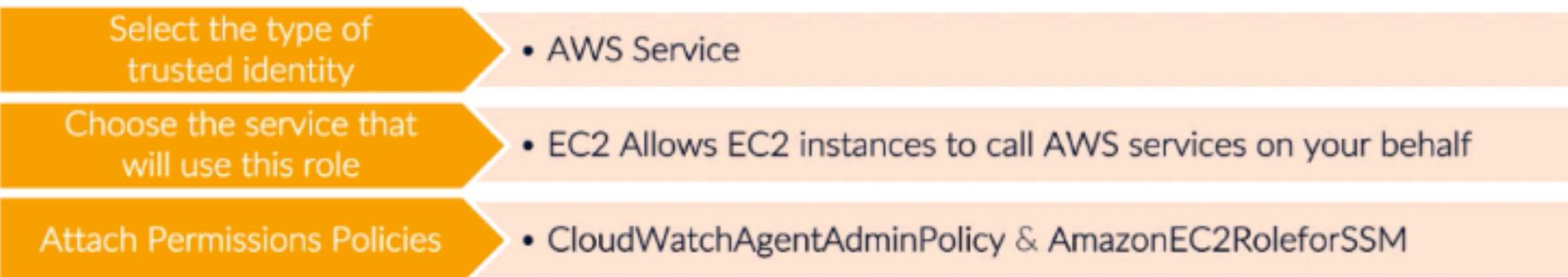


Used to communicate with the Parameter store within SSM, to store a configuration information file of the Agent



Creating Roles

Configuration of the role with the additional permissions for SSM:



The role used to install the agent and send data to CloudWatch:



Downloading the Agent

From the EC2 instance with additional permissions you then install the agent

- It can be downloaded from an S3 public link:



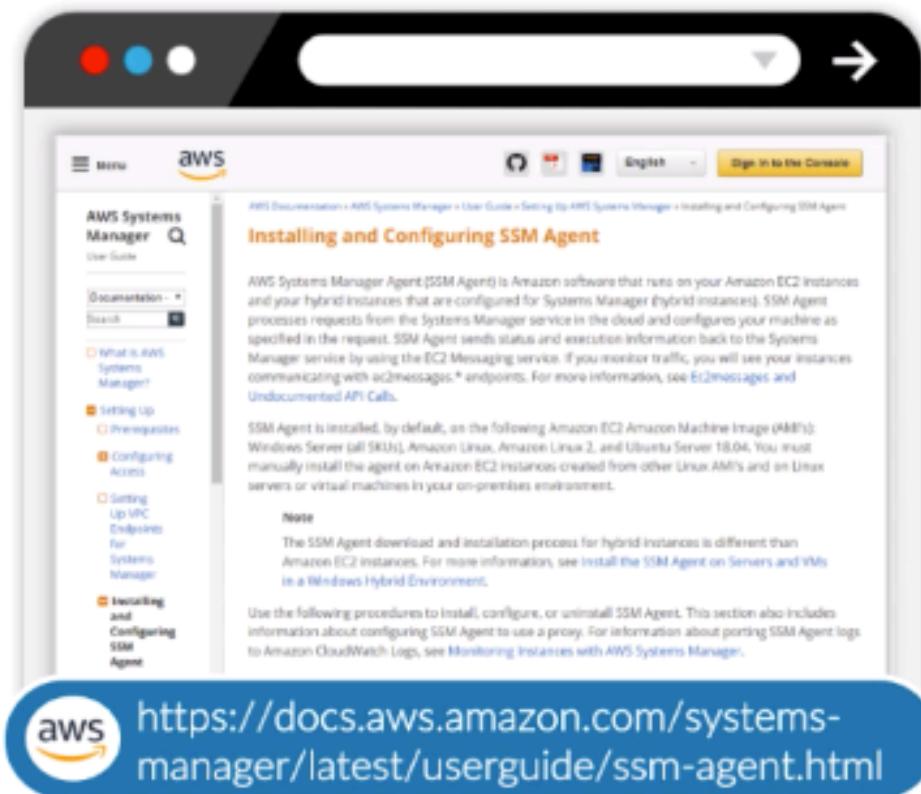
```
wget https://s3.amazonaws.com/amazoncloudwatch-  
agent/linux/amd64/latest/AmazonCloudWatchAgent.zip
```

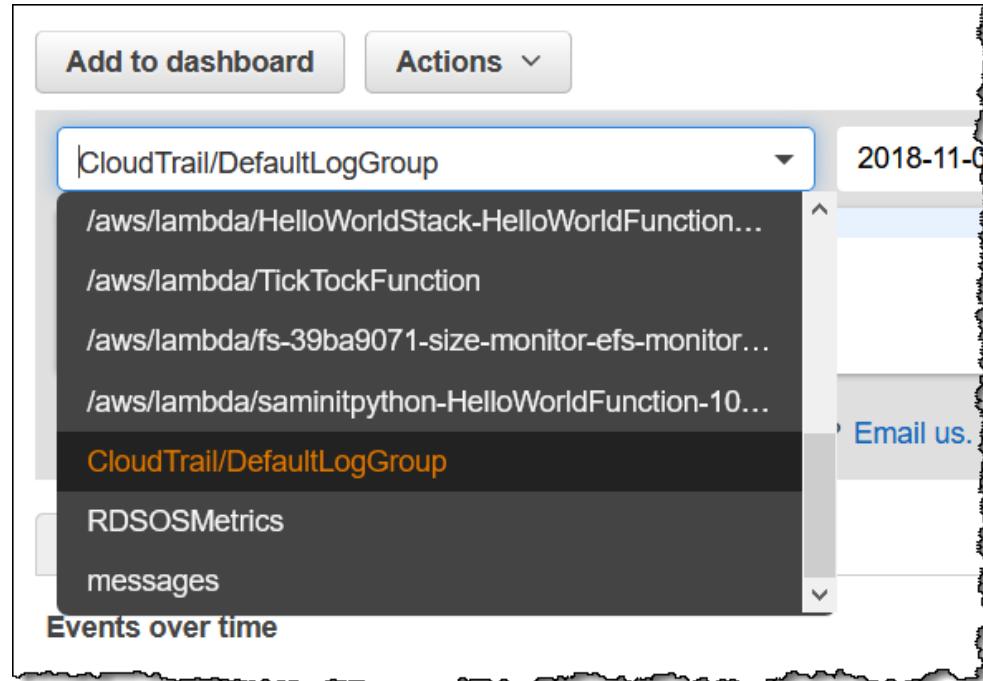


```
https://s3.amazonaws.com/amazoncloudwatch-  
agent/windows/amd64/latest/AmazonCloudWatchAgent.zip
```

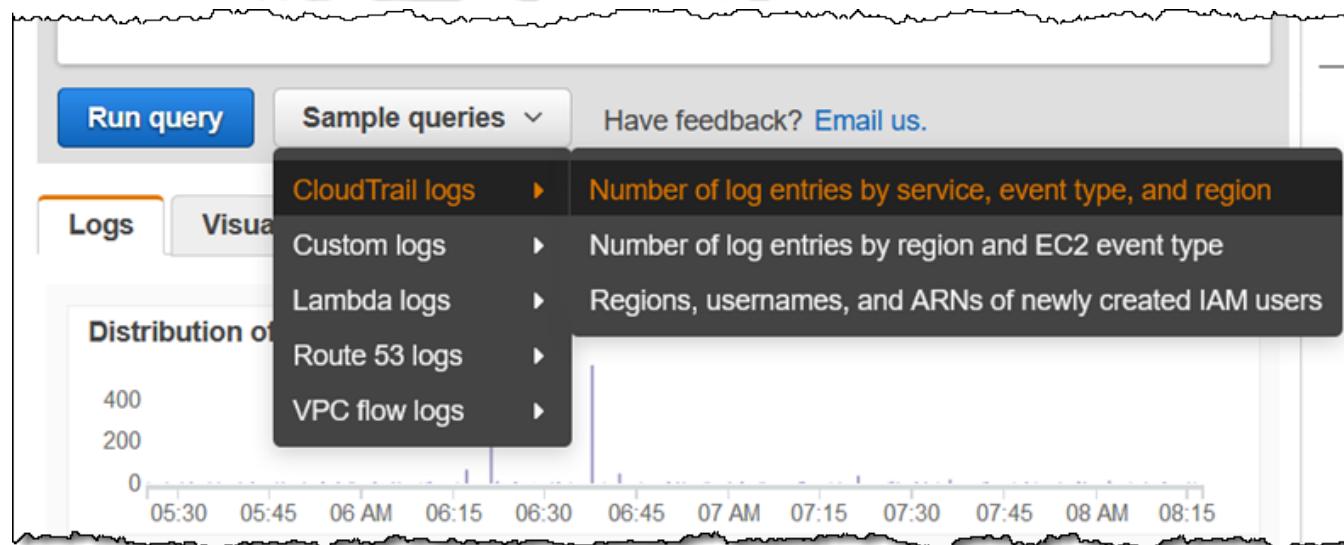
Prerequisites

- Verify that your EC2 instance has access to the internet
- You must also have the SSM agent installed
- It is already installed for:
 - Linux AMIs dated 2017.09 and later
 - Windows Server 2016 instances
 - Instances created from Windows Server 2003-2012 R2 AMIs published in November 2016 or later

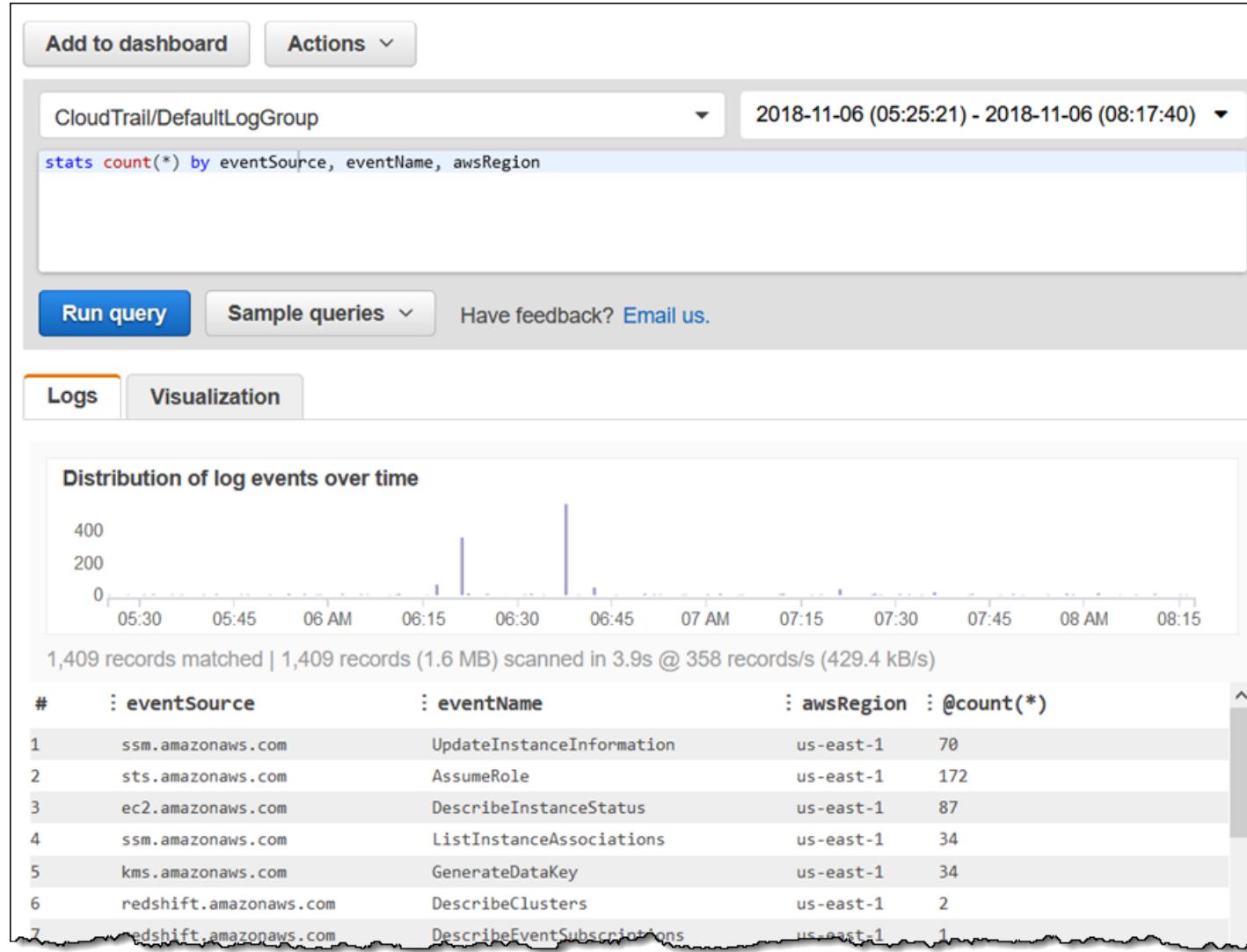




Buscar log



Hacer query



Resultado del LOG filter

CloudTrail/DefaultLogGroup ▼ 2018-11-06 (05:25:21) - 2018

```
stats count(*) by eventSource, eventName, awsRegion  
| filter eventSource like /ec2/
```

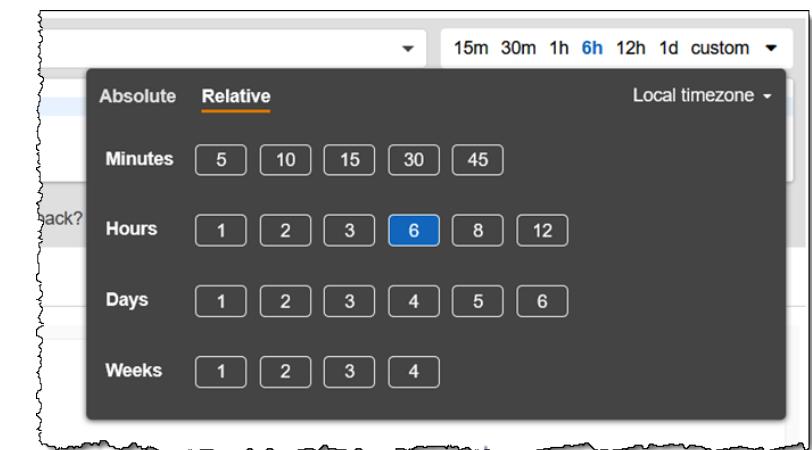
Run query **Sample queries** ▼ Have feedback? [Email us.](#)

Logs **Visualization**

Distribution of log events over time

141 records matched | 1,409 records (1.6 MB) scanned in 3.9s @ 363 records/s (436.4 kB/s)

#	: eventSource	: eventName	: awsRegion	: @count(*)
1	ec2.amazonaws.com	DescribeInstanceStatus	us-east-1	87
2	ec2.amazonaws.com	DescribeSecurityGroups	us-east-1	7
3	ec2.amazonaws.com	DescribeSubnets	us-east-1	7
4	ec2.amazonaws.com	DescribeNetworkAcls	us-east-1	3
5	ec2.amazonaws.com	DescribeInstances	us-east-1	23
6	ec2.amazonaws.com	DescribeNetworkInterfaces	us-east-1	2



Añadido del LOG filter

The Log Filter query language supports six types of commands:

- **fields** – Retrieves one or more log fields. It can also make use of functions such as abs, sqrt, strlen, trim, and more.
- **filter** – Retrieves log fields based on one or more conditions built from Boolean operators, comparison operators, and regular expressions.
- **stats** – Calculates aggregate statistics such as sum, avg, count, min, max, and percentile for a log field, across a given time interval (specified using the optional by modifier).
- **sort** – Sorts logs events in ascending or descending order.
- **limit** – Limits the number of log events returned by a query.
- **parse** – Extracts data from a log field, creating one or more ephemeral fields that can be further processed by the query.

Alarms

- Alarms are thresholds we specify
- Set the upper or lower limit, i.e. tolerance
- Alarms can be associated with a simple action, or a complex series of actions

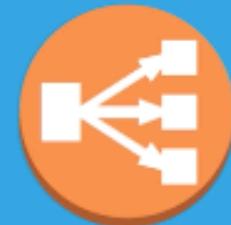
Define The Terms - Monitoring



Monitoring:

Observe and check the progress or quality of (something) over a period of time; keep under systematic review.

In AWS: Verification your cloud works.



Define The Terms - Metrics



Metrics:

A system or standard of measurement.

In AWS: Quantifying cloud behavior and state.

Define The Terms - Logging



Logging:

Recording of performance, events, or day-to-day activities.

In AWS: Time sequence of system occurrences.

Monitoring + Metrics + Logging = Events

Every health check, metrics collection ping, line of log content is an *event*, or *thing that happened*.

Like normal analytics, “what happens” has value.

Streams are time-sequenced event buffers.

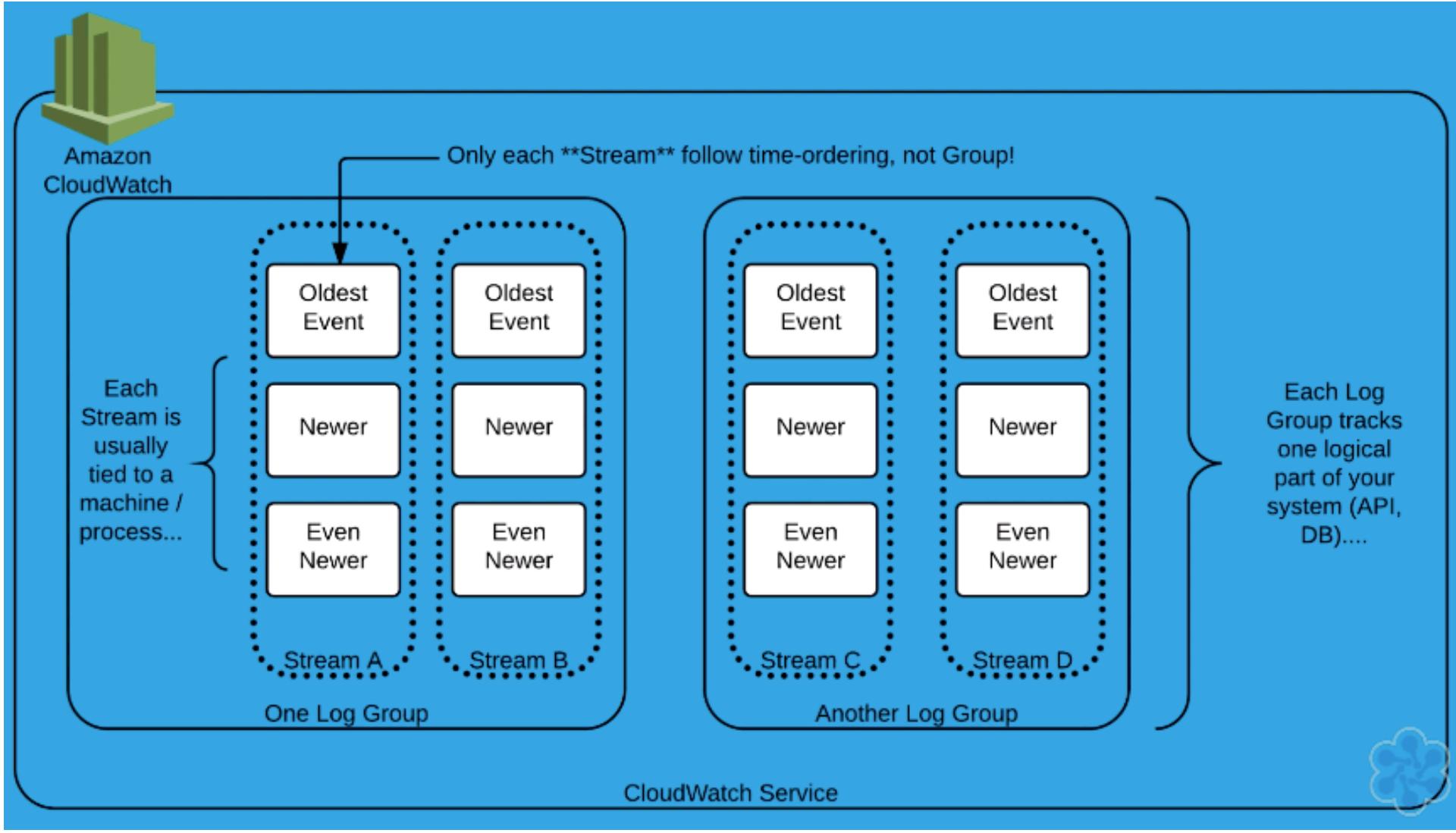
Logs are time-sequenced events. Streams:

- 1. Natively support log event style**
- 2. Provide unified transport to other services**
- 3. Allow different consumption/production rates**
- 4. Decouple producer and consumer systems**

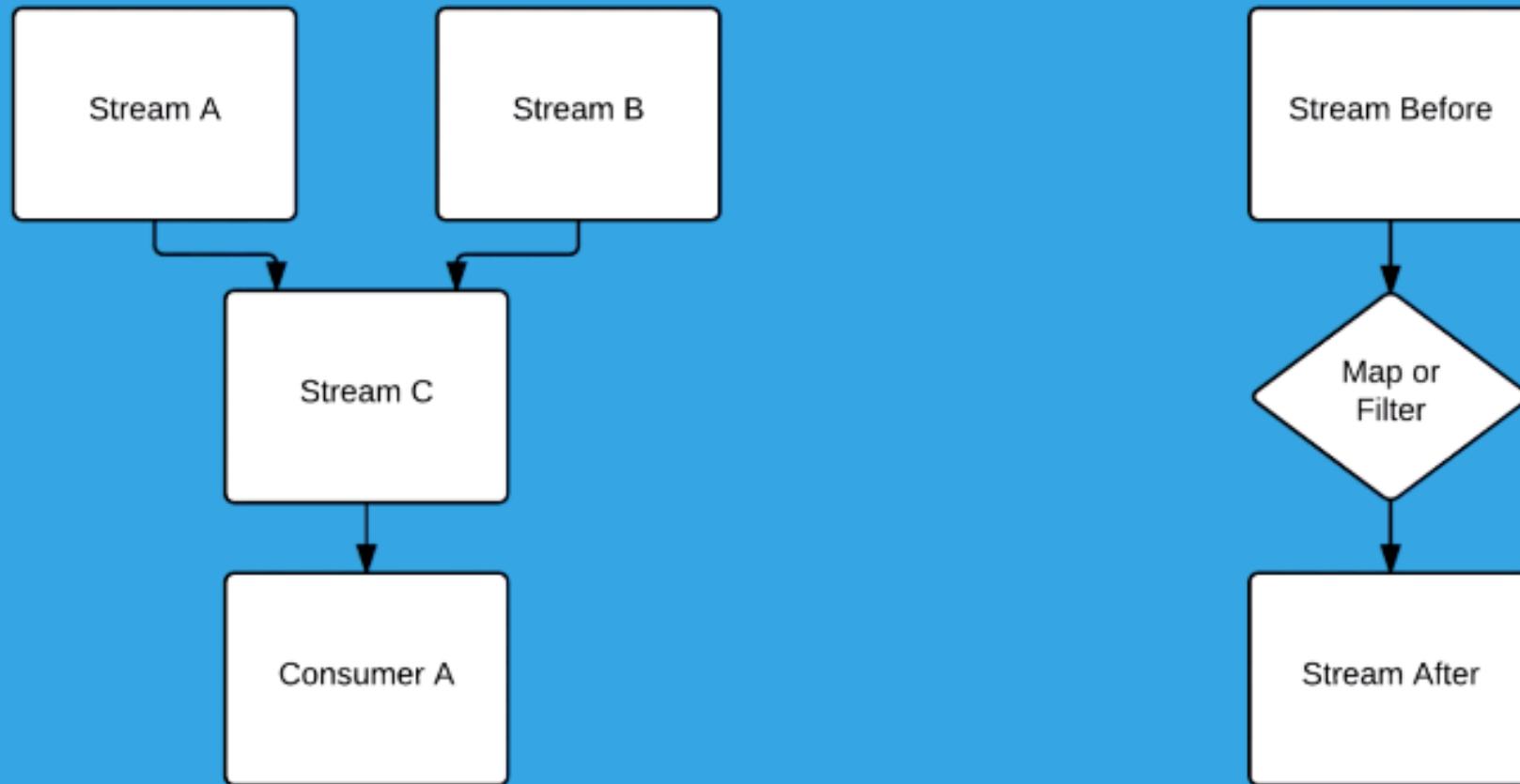
Logs aren't files you open and scan when something breaks.

Logs are a primary data transport mechanism.

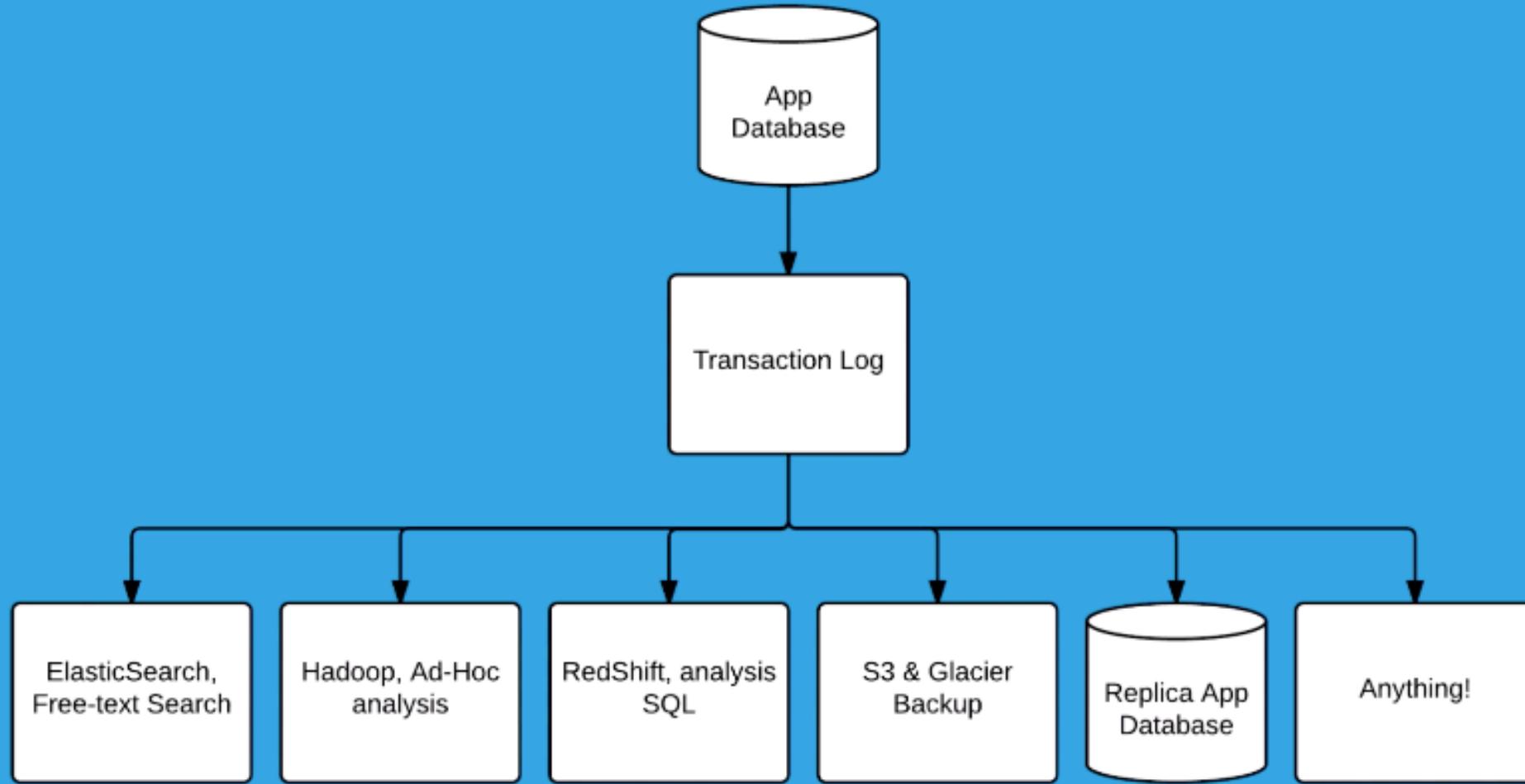
Log events are fundamental design building blocks.



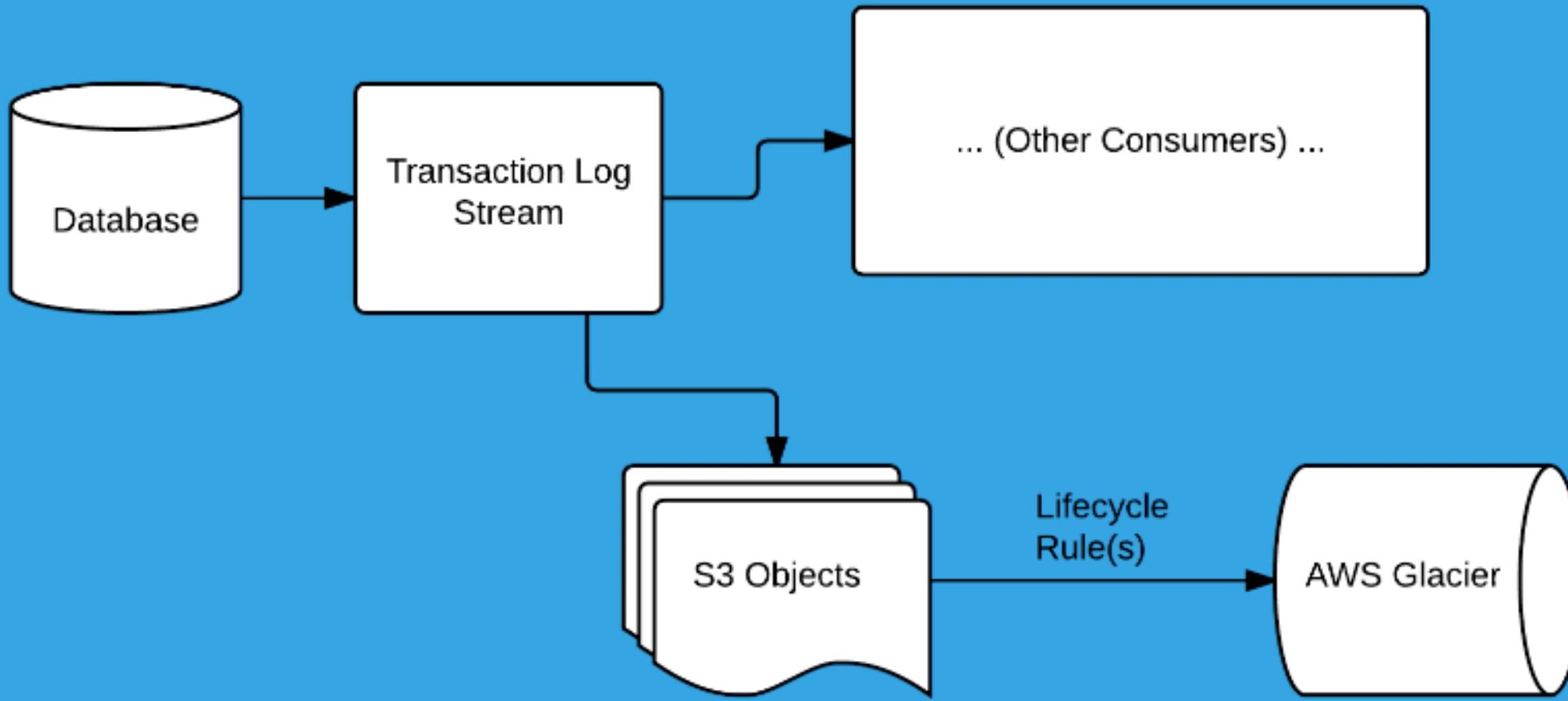
Unification And Distribution



Stream Sinks



Archival And Backup



AWS Config



- AWS Config is security and compliance tool that integrates with many AWS services
- The Configuration History uses Configuration Items (CIs) to collate and produce a history of changes to a particular resource
- Configuration History files are sent to an S3 bucket for each resource type
- This Configuration file typically delivered every 6 hours
- A CI is comprised of a JSON file that holds configuration information, relationship information and other metadata as a point in time snapshot view of a supported resource

AWS Config



- A CI is created every time a supported resource has a change made to its configuration
- A CI consists of the following sections:
 - Metadata
 - Attributes
 - Relationships
 - Current Configuration
 - Related Events
- You can aggregate Configuration History files from multiple accounts into one S3 Bucket

What can AWS Config do?



Store configuration history



Provide a snapshot of configurations



Capture resource changes



Act as resource inventory



CloudAcademy



Identify relationships



Notifications about changes



Security analysis



Provide AWS CloudTrail integration



Use Rules to check compliance

Logging in AWS Config

The Configuration History uses Configuration Items (CIs) to collate and produce a history of changes to a particular resource

- The Information can be accessed through the AWS CLI
- You can also specify the 'Resource Type'

```
aws configservice get-resource-config-history  
--resource-type AWS::EC2::Subnet --resource-id  
subnet-ef2263b7
```

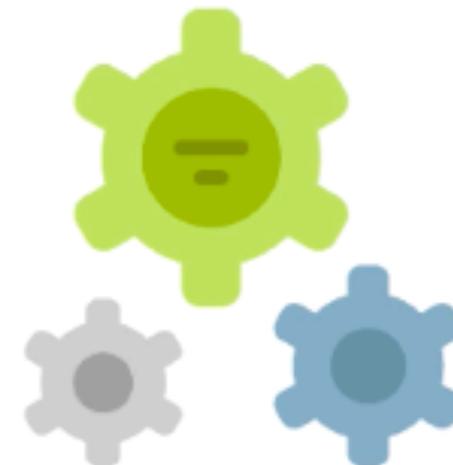
Logging in AWS Config



Config sends a Configuration History file for each resource type to an S3 bucket

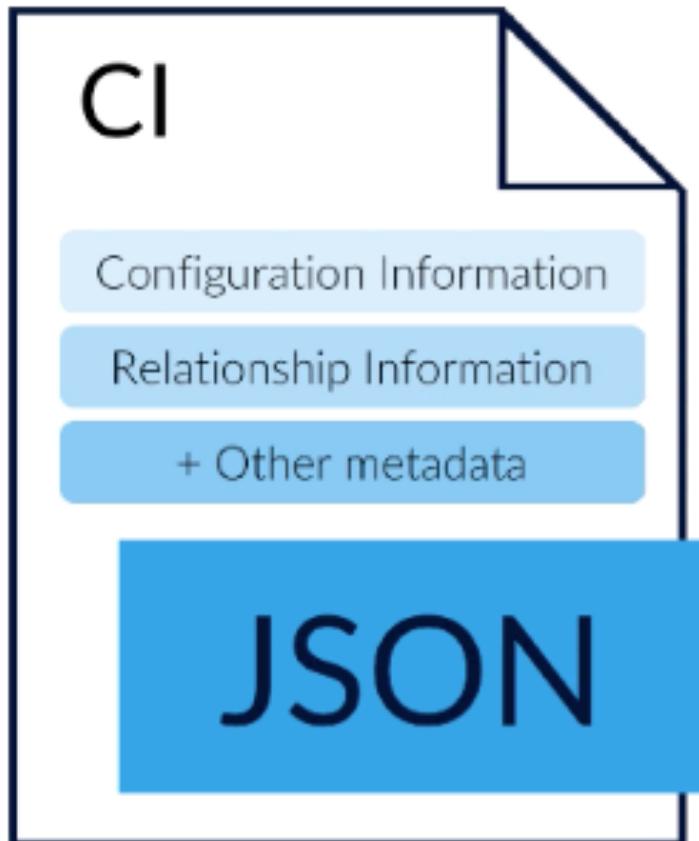


Configuration History files are delivered every 6 hours

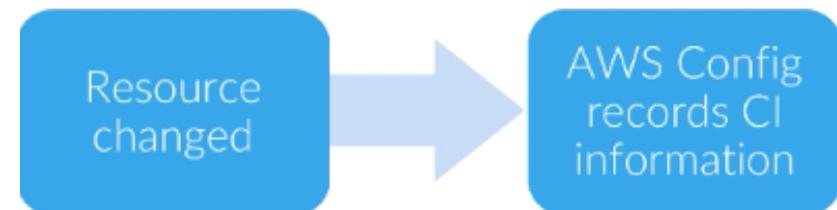


It contains all CI changes for all resources of a particular type

Configuration Item (CI)



- A CI is created every time a supported resource has a change made to its configuration in any way
- AWS Config will also record CI's for any directly related resources



Configuration Item (CI)

1. Metadata

- Version and Configuration ID
- MD5Hash (for comparison)
- Time of capture and State ID

2. Attributes

- Unique resource ID
- Key-value tags
- Resource type
- Amazon Resource Name (ARN)

3. Relationships

- Description of relationships to other resources

CI

JSON

4. Current Configuration

- Displays info from 'Describe' or 'List' calls from AWS CLI

5. Related Events

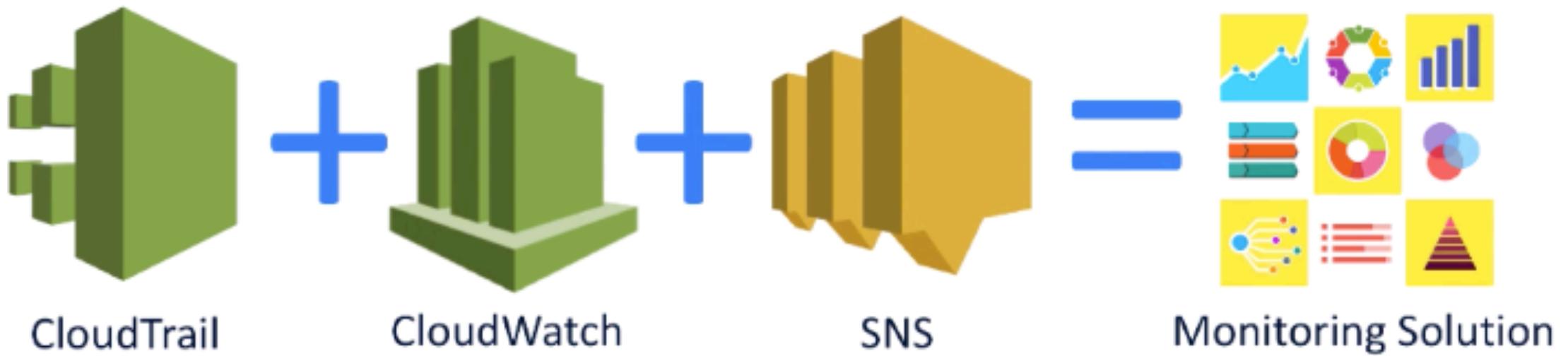
- Displays the related AWS CloudTrail event ID

Configuration Item (CI)



- The S3 bucket is selected at time of configuration and is used to store all the Configuration History files
- You may aggregate your Configuration History files into the same S3 Bucket for your primary account
- You need to grant write access for the service principal in your secondary accounts and write access to the S3 bucket in your primary account.

Monitoring

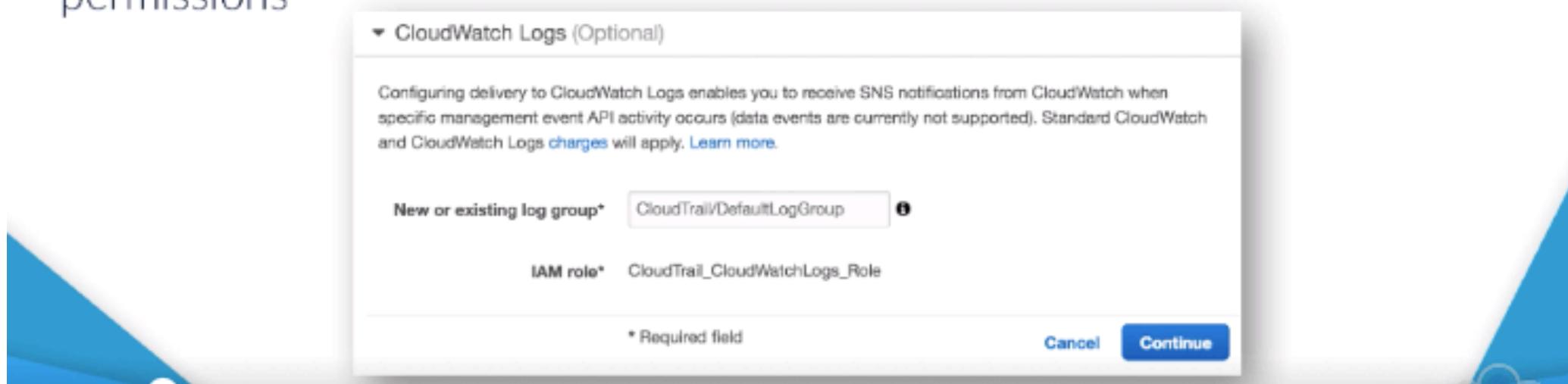


Monitoring Use Cases

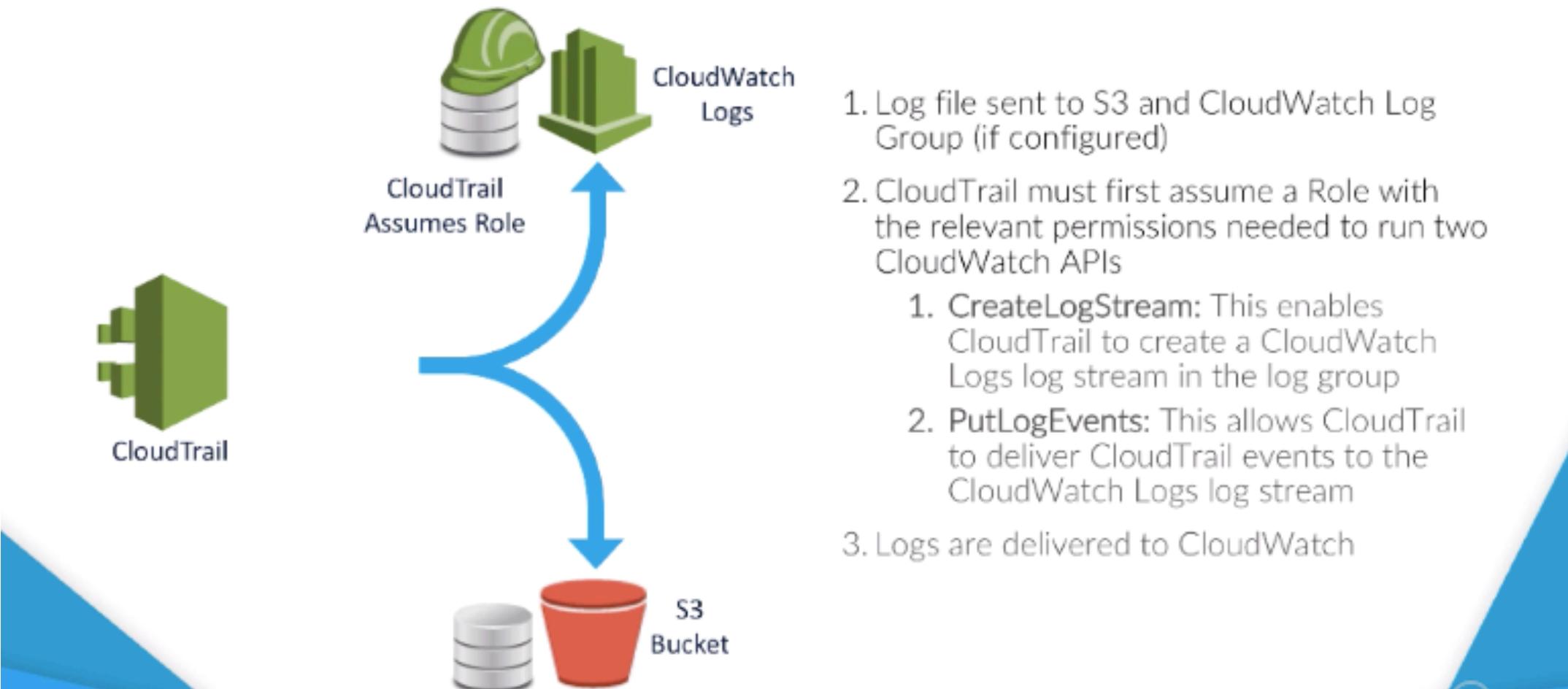
- Starting, stopping, rebooting and terminating EC2 instances
- Changes to security policies within IAM and S3
- Failed login attempts to the Management Console
- API calls that result in failed authorization

CloudTrail & CloudWatch

- To deliver logs to CloudWatch, your Trail must be configured to use a new or existing CloudWatch Log Group
- I recommend allowing CloudTrail to create a new Log Group for you
- Using existing Log Groups requires configuration and necessary permissions



CloudTrail & CloudWatch Process



CloudTrail Role Policy

```
{  
  "Version": "2012-10-17",  
  "Statement": [  
    {  
      "Sid": "AWSCloudTrailCreateLogStream20141101",  
      "Effect": "Allow",  
      "Action": [  
        "logs>CreateLogStream"  
      ],  
      "Resource": [  
        "arn:aws:logs:eu-west-1:730739171055:log-group:CloudTrail/DefaultLogGroup:log-stream:730739171055_CloudTrail_eu-west-1"  
      ]  
    },  
    {  
      "Sid": "AWSCloudTrailPutLogEvents20141101",  
      "Effect": "Allow",  
      "Action": [  
        "logs:PutLogEvents"  
      ],  
      "Resource": [  
        "arn:aws:logs:eu-west-1:730739171055:log-group:CloudTrail/DefaultLogGroup:log-stream:730739171055_CloudTrail_eu-west-1"  
      ]  
    }  
  ]  
}
```

CloudWatch Configuration



- CloudWatch Log Events have a size limitation of 256KB
- Add Metric Filters to allow search of the Logs
- Each Metric Filter requires a Filter Pattern
 - Filter Patterns determines what data CloudWatch is monitoring
 - Filter Pattern syntax documentation:

<https://docs.aws.amazon.com/AmazonCloudWatch/latest/logs/FilterAndPatternSyntax.html>

La guía [AWS Command Line Interface User Guide](#) proporciona instrucciones detalladas sobre la instalación y configuración de la herramienta. Después podrá empezar a realizar llamadas a los servicios de AWS desde la línea de comandos.

```
$ aws ec2 describe-instances
```

```
$ aws ec2 start-instances --instance-ids i-1348636c
```

```
$ aws sns publish --topic-arn arn:aws:sns:us-east-1:546419318123:OperationsError --message "Script Failure"
```

```
$ aws sqs receive-message --queue-url https://queue.amazonaws.com/546419318123/Test
```

Puede recibir ayuda sobre la línea de comandos para ver los servicios soportados, las operaciones de un servicio, las operaciones de un servicio,

```
$ aws help
```

```
$ aws autoscaling help
```

```
$ aws autoscaling create-auto-scaling-group help
```

1. View Current Status of an Instance

The following “aws ec2 describe-instances” will display detailed information about all instances that are managed by you. The output will be in JSON format.

```
aws ec2 describe-instances
```

If you have way too many instances, you can use the filter option to view a specific instance. The following will display only the instance which has the “Name” tag set as “dev-server”.

```
# aws ec2 describe-instances --filter Name=tag:Name,Values=dev-server
..
..
{
  "State": {
    "Code": 80,
    "Name": "stopped"
  },
..
..
  "InstanceId": "i-e5888e46",
```

2. Start an Instance

The following “aws ec2 start-instances” command will start the instance that is specified in the –instance-ids field. This will also display the current state and the previous state of the instance in the output. As you see from the following output, previously this instance was “stopped” and now it is in “pending” state and will be started soon.

```
# aws ec2 start-instances --instance-ids i-ddddd70
{
  "StartingInstances": [
    {
      "InstanceId": "i-ddddd70",
      "CurrentState": {
        "Code": 0,
        "Name": "pending"
      },
      "PreviousState": {
        "Code": 80,
        "Name": "stopped"
      }
    }
}
```

3. Stop an Instance

The following “aws ec2 stop-instances” command will stop the instance that is specified in the –instance-ids field.

```
# aws ec2 stop-instances --instance-ids i-5c8282ed
{
  "StoppingInstances": [
    {
      "InstanceId": "i-5c8282ed",
      "CurrentState": {
        "Code": 64,
        "Name": "stopping"
      },
      "PreviousState": {
        "Code": 16,
        "Name": "running"
      }
    }
}
```

The following are the possible state name and state code for an instance:

- 0 is for pending
- 16 is for running
- 32 is for shutting-down
- 48 is for terminated
- 64 is for stopping
- 80 is for stopped

To stop multiple instances together, specify one or more instances ids as shown below.

```
aws ec2 stop-instances --instance-ids i-5c8282ed i-e5888e46
```

You can also force an instance to stop. This will not give the system an opportunity to flush the filesystem level cache. Use this only when you know exactly what you are doing.

```
aws ec2 stop-instances --force --instance-ids i-ddddd70
```

4. Terminate an Instance

The following “aws ec2 terminate-instances” command will terminate the instance that is specified in the –instance-ids field.

```
# aws ec2 terminate-instances --instance-ids i-44a44ac3
{
    "TerminatingInstances": [
        {
            "InstanceId": "i-44a44ac3",
            "CurrentState": {
                "Code": 48,
                "Name": "terminated"
            },
            "PreviousState": {
                "Code": 80,
                "Name": "stopped"
            }
        }
    ]
}
```

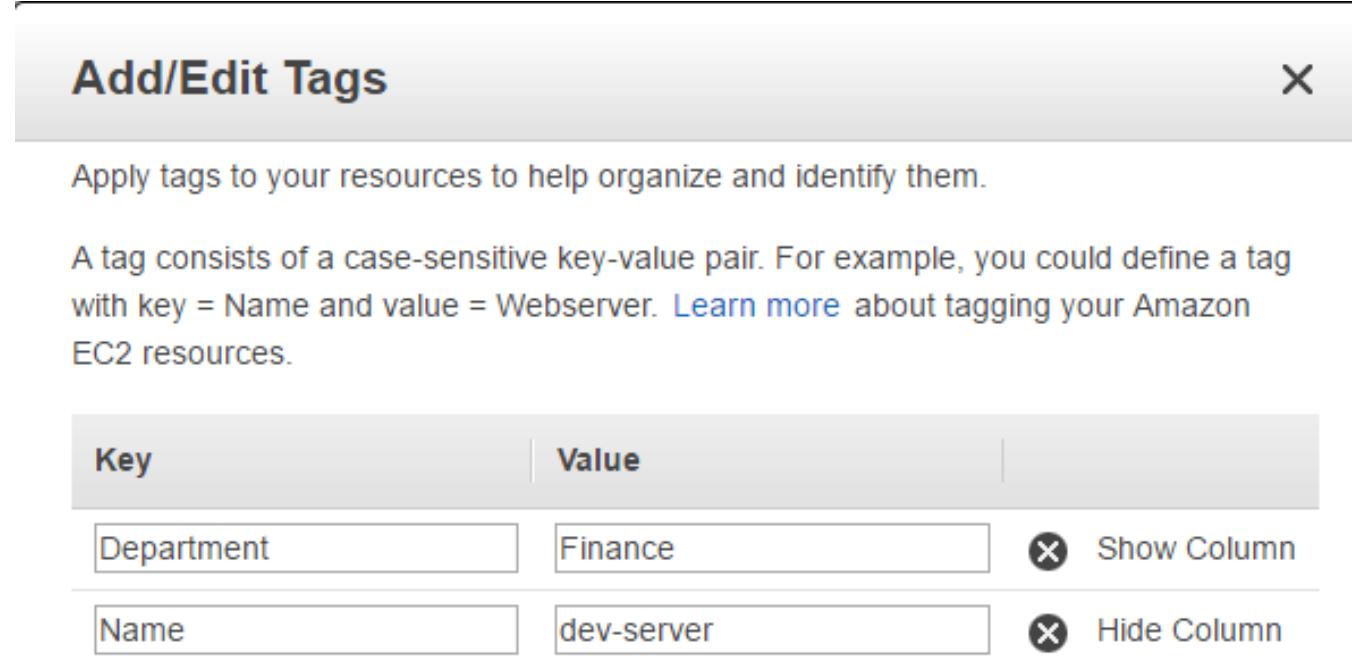
5. Add Name Tag to an Instance

The following “aws ec2 create-tags” command will add a new tag to the specified instance.

In this example, we are adding a tag with Key as “Department”, and it’s Value as “Finance”

```
aws ec2 create-tags --resources i-ddddd70 --tags Key=Department,Value=Finance
```

```
# aws ec2 describe-instances
...
"Tags": [
    {
        "Value": "Finance",
        "Key": "Department"
    },
    {
        "Value": "dev-server",
        "Key": "Name"
    }
],
```



6. Add Storage (Block Device) to an Instance

First, use the following command to get a list of all block device volumes that are available for you. Look for those volumes that has the State as “available”

```
aws ec2 describe-volumes
..
{
    "AvailabilityZone": "us-east-1b",
    "Attachments": [],
    "Encrypted": false,
    "VolumeType": "standard",
    "VolumeId": "vol-1d5cc8cc",
    "State": "available",
    "SnapshotId": "",
    "CreateTime": "2016-04-17T15:08:40.469Z",
    "Size": 1
}
..
```

From the above, get the VolumeId, and use that in the following “aws ec2 attach-volume” command to attach that volume to a particular instance.

In the following command, you should also specify the –device option, which will be the disk name that will be used at the OS level for this particular volume.

In this example, this volume will be attached as “/dev/sdh” disk.

```
# aws ec2 attach-volume --volume-id vol-1d5cc8cc --instance-id i-ddddddd7
{
    "AttachTime": "2016-04-17T15:14:10.144Z",
    "InstanceId": "i-ddddddd70",
    "VolumeId": "vol-1d5cc8cc",
    "State": "attaching",
    "Device": "/dev/sdh"
}
```

Note: When you attach a volume to an instance from the AWS management console, by default it will automatically populate the device. But in the AWS EC2 CLI, you have to specify the device name as shown below.

After attaching the device, you'll notice that the state changed from "available" to "attached" for this particular volume.

```
# aws ec2 describe-volumes  
..  
"Attachments": [  
    {  
        "AttachTime": "2016-04-17T15:14:10.000Z",  
        "InstanceId": "i-ddddd70",  
        "VolumeId": "vol-1d5cc8cc",  
        "State": "attached",  
    ..
```

7. Launch a New EC2 Instance

The following command will create a new AWS EC2 instance for you.

This is equivalent to the “Launch Instance” that you’ll perform the AWS management console.

To launch an instance, use “aws ec2 run-instances” command as shown below.

```
# aws ec2 run-instances --image-id ami-22111148 --count 1 --instance-type t1.micro --key-name stage-key --  
security-groups my-aws-security-group
```

In the above command:

–image-id Specify the image id for the AMI that you want to launch. You can browse the AWS marketplace and choose the correct image that is required for your project.

–count Specify the number of instance that you want to launch from this image. In this case, we are creating only one new instance.

–instance-type In this example, I’m launching this instance as a t1.micro type, which doesn’t use have CPU and RAM.

–key-name Specify the name of the key pair that you want to use this with system. You should create your own key pair before launching your instance.

–security-groups Specify the name of the security groups. You should create a security group with appropriate firewall rules that are required for your project.

```
{  
    "OwnerId": "353535354545",  
    "ReservationId": "r-d6668103",  
    "Groups": [  
        {  
            "GroupName": "my-aws-security-group",  
            "GroupId": "sg-6cbebe01"  
        }  
    ],  
    "Instances": [  
        {  
            "Monitoring": {  
                "State": "disabled"  
            },  
            "PublicDnsName": "",  
            "KernelId": "aki-91acfcaf8",  
            "State": {  
                "Code": 0,  
                "Name": "pending"  
            },  
            "EbsOptimized": false,  
            "LaunchTime": "2016-04-17T19:13:56.000Z",  
            "ProductCodes": [],  
            "StateTransitionReason": "",  
            "InstanceId": "i-44a44ac3",  
            "ImageId": "ami-22111148",  
            "PrivateDnsName": ""  
        }  
    ]  
}
```

```
"KeyName": "stage-key",  
"SecurityGroups": [  
    {  
        "GroupName": "my-aws-security-group",  
        "GroupId": "sg-6cbebe01"  
    }  
],  
"ClientToken": "",  
"InstanceType": "t1.micro",  
"NetworkInterfaces": [],  
"Placement": {  
    "Tenancy": "default",  
    "GroupName": "",  
    "AvailabilityZone": "us-east-1c"  
},  
"Hypervisor": "xen",  
"BlockDeviceMappings": [],  
"Architecture": "x86_64",  
"StateReason": {  
    "Message": "pending",  
    "Code": "pending"  
},  
"RootDeviceName": "/dev/sda1",  
"VirtualizationType": "paravirtual",  
"RootDeviceType": "ebs",  
"AmiLaunchIndex": 0  
}
```

If you get the following error message, then the instance type you've selected is not supported for this AMI. Change the instance type and try again.

```
# aws ec2 run-instances --dry-run --image-id ami-08111162 --count 1 --instance-type t1.micro --key-name MyKeyPair
```

A client error (InvalidParameterCombination) occurred when calling the RunInstances operation: Non-Windows instances with a virtualization type of 'hvm' are currently not supported for this instance type.

The following are additional parameters that you can pass with the “aws ec2run-instances” command

–subnet-id Use the appropriate subnet id to launch a EC2 VPC instance

–block-device-mappings file://mymap.json In this JSON file you can specify the volumes that you want to attach to the instance that you want to launch

–user-data file://myuserdata.txt In this text file you can specify the userdata that need to be executed when the EC2 instance is launched

–iam-instance-profile Name=myprofile You can also specify your IAM profile that you want to use while launching the instance

8. Reboot an Instance (and General Options)

To reboot an instance, use “aws ec2 reboot-instances” command as shown below.

```
aws ec2 reboot-instances --instance-ids i-ddddd70
```

The are few options that you can use pretty much with most of the AWS EC2 cli commands.

For example, you can use “–dry-run” option pretty much with all the AWS EC2 cli command. As the name suggests, it will not really execute the command. This will only perform a dry-run and display all possible error messages without really doing anything.

For example, the following is a dry-run operation when you want to stop an instance.

```
# aws ec2 stop-instances --dry-run --instance-ids i-ddddd70
```

A client error (DryRunOperation) occurred when calling the StopInstances operation: Request would have succeeded, but DryRun flag is set.

When you are performing a dry-run the following are the two possible errors:

- If you have appropriate permission, it will display “DryRunOperation” error, and any other real error message that are related to that specific command that you are executing.
- If you don’t have permission to execute that particular command, it will display “UnauthorizedOperation” error

If you don't know exactly what kind of information needs to be passed for a particular EC2 command in JSON format, you can use `--generate-cli-skeleton` as shown below. Once you have the JSON output, modify the appropriate values, and use it as an input to `--cli-input-json` option.

```
# aws ec2 stop-instances --dry-run --force --generate-cli-skeleton --instance-ids i-ddddd70
{
    "DryRun": true,
    "InstanceIds": [ "i-ddddd70" ],
    "Force": true
}
```

The following is an example JSON file that can be used as an input to AWS EC2 CLI command.

```
# cat stop.json
{
    "DryRun": true,
    "InstanceIds": [ "i-ddddd70" ],
    "Force": true
}
```

In the following example, we are using the above stop.json file as an value for the –client-input-json option as shown below. Don't forget to give “file://”

```
aws ec2 stop-instances --cli-input-json file://stop.json
```

9. Change Instance Type

You can change the above instance to a different instance type.

For that, first stop the instance. Without stopping you cannot change the instance type.

```
aws ec2 stop-instances --instance-ids i-44a44ac3
```

The following “aws ec2 modify-instance-attribute” is used to change the instance type. In this example, we are changing the instance type to “m1.small”

```
aws ec2 modify-instance-attribute --instance-id i-44a44ac3 --instance-type "{\"Value\": \"m1.small\"}"
```

If an instance type is not supported for your particular image, you’ll get the following error message. In this example, t2.nano is not supported for this particular image.

```
# aws ec2 modify-instance-attribute --instance-id i-44a44ac3 --instance-type "{\"Value\": \"t2.nano\"}"
```

A client error (InvalidParameterCombination) occurred when calling the ModifyInstanceAttribute operation: Virtualization type 'hvm' is required for instances of type 't2.nano'. Ensure that you are using an AMI with virtualization type 'hvm'.

10. Create a New Image

From your particular instance that is running with all the configuration changes that you've done so far, you can create a new image using the following “aws ec2 create-image” command.

```
# aws ec2 create-image --instance-id i-44a44ac3 --name "Dev AMI" --description "AMI for development server"
{
  "ImageId": "ami-2d574747"
}
```

This is helpful when you want to launch new instance based on this new image that you created which has your changes in it.

11. Delete an Image

When you create an image, it also creates a snapshot.

So, when you are deleting your image you have to do two things.

First, use the “aws ec2 deregister-image” command to deregister the Image.

```
aws ec2 deregister-image --image-id ami-2d574747
```

Next, use the “aws ec2 delete-snapshot” command to delete the snapshot that is associated with your image.

```
aws ec2 delete-snapshot --snapshot-id snap-4e665454
```

12. Get System Log (View Console Output)

Since you don't have a physical access to the console for the instances that are running on AWS EC2, use the following command.

This “aws ec2 get-console-output” command will display whatever was sent to the system console for your particular instance.

```
aws ec2 get-console-output --instance-id i-44a44ac3
```

13. AWS EC2 Key Pairs

The following “aws ec2 describe-key-pairs” command will display all keypairs that you’ve created so far in AWS.

```
# aws ec2 describe-key-pairs
{
  "KeyPairs": [
    {
      "KeyName": "prod-key",
      "KeyFingerprint": "61:7c:f1:13:53:b0:3a:01:dd:dd:6c:90"
    },
    {
      "KeyName": "stage-key",
      "KeyFingerprint": "41:6c:d1:23:a3:c0:2a:0a:dc:db:60:4c"
    }
  ]
}
```

To create a new Keypair use the following “aws ec2 create-key-pair” command. In this example, I’m creating a key pair with name “dev-servers”. I’ll be using this key-pair for all my dev instances.

```
# aws ec2 create-key-pair --key-name dev-servers
{
    "KeyName": "dev-servers",
    "KeyMaterial": "-----BEGIN RSA PRIVATE KEY-----\n
dYXbKYMRLI59J5XKyPgC/67GL8\nXg
....
-----END RSA PRIVATE KEY-----",
    "KeyFingerprint": "3d:c2:c8:7f:d2:ee:1d:66"
}
```

If you have created a keypair by mistake, use the following command to delete it.

```
# aws ec2 delete-key-pair --key-name dev-servers
```

14.Delete an S3 bucket and all its contents with just one command

Sometimes you may end up with a bucket full of hundreds or thousands of files that you no longer need. If you have ever had to delete a substantial number of items in S3, you know that this can be a little time-consuming. The following command will delete a bucket and all of its content including directories:

```
aws s3 rb s3://bucket-name –force
```

15. Recursively copy a directory and its subfolders from your PC to Amazon S3

If you have used the S3 Console, at some stage, you've probably found yourself having to copy a ton of files to a bucket from your PC. It can be a little clunky at times, especially if you have multiple directory levels that need to be copied. The following AWS CLI command will make the process a little easier, as it will copy a directory and all of its subfolders from your PC to Amazon S3 to a specified region.

```
aws s3 cp MyFolder s3://bucket-name — recursive [–region us-west-2]
```

16 . Display subsets of all available ec2 images

The following will display all available ec2 images, filtered to include only those built on Ubuntu (assuming, of course, that you're working from a terminal on a Linux or Mac machine).

```
aws ec2 describe-images | grep ubuntu
```

17. List users in a different format

Sometimes, depending on the output format you chose as default when you invoke long lists – like a large set of users – the display format can be a little hard to read. Including the –output parameter with, say, the table argument, will display a nice, easy-to-read table this one time without having to change your default.

```
aws iam list-users –output table
```

18. List the sizes of an S3 bucket and its contents

The following command uses JSON output to list the size of a bucket and the items stored within. This might come in handy when auditing what is taking up all your S3 storage.

```
aws s3api list-objects --bucket BUCKETNAME --output json --query "[sum(Contents[].Size), length(Contents[])]"
```

19. Move S3 bucket to a different location

If you need to quickly move an S3 bucket to a different location, then this command just might save you a ton of time.

```
aws s3 sync s3://oldbucket s3://newbucket --source-region us-west-1 --region us-west-2
```

20. List all of your instances that are currently stopped and the reason for the stop

Here's another use of the JSON output parameter. This one will list all of your stopped instances and, best of all, show the reason that they were stopped:

```
aws ec2 describe-instances --filters Name=instance-state-name,Values=stopped --region eu-west-1 --output json | jq -r .Reservations[].Instances[].StateReason.Message
```

```
# cambiar permisos pem
```

```
chmod 400 cloud5.pem
```

```
# conectar a instancia
```

```
ssh -i "cloud5.pem" ec2-user@ec2-3-208-10-215.compute-1.amazonaws.com
```

```
# copiar ficheros desde pc a EC2
```

```
scp -i "luisa.pem" hola.zip ec2-user@ec2-15-188-59-219.eu-west-3.compute.amazonaws.com:/home/ec2-user
```

```
# limpiar terminal
```

```
clear
```

ls - lista dir y fichero

ls directorio y files

ls -l

ls -a ficheros ocultos

ls . carpeta actual

ls.. carpeta padre

cambio directorio

cd .. sube a padre

cd carpeta se mueve a esa carpeta

cd - vuelve a la carpeta anterior.

pwd - te dice en que directorio estas

```
## mkdir crear directorio  
    mkdir nombre dir  
    mkdir -p dir1/dir2/dir3 crea toda la cadena  
    mkdir dir1/dir2/dir3 crea dir3  
  
## crear fichero de prueba  
    touch file crea fichero vacío o cambia fecha modificado  
  
## copiar carpeta o fichero  
    cp \[-R [-H | -L | -P]] \[-fi | -n]() [-apvXc] source\_file target\_file  
    cp \[-R [-H | -L | -P]] \[-fi | -n]() [-apvXc] source\_file ... target\_directory  
    cp sourcefile destinationfile  
    cp source file /directoriode destino  
    -R recursive copy directory contents
```

```
# mover/renombrar carpeta o fichero  
mv nombreinicial nombrefinal cambia nombre  
mv fichero carpeta/ mueve el fichero  
  
## borrar carpeta o fichero  
rm nombre borra fichero  
rm carpeta/ borra carpeta y ficheros  
rm -d carpeta/ borra solo carpeta vacías  
rm -R carpeta borra carpetas y subcarpeta  
  
## escribir comentario en pantalla  
echo comentario  
  
## listar ficheros  
aws s3 ls s3://bucket  
aws s3 website s3://paraborrar123/ --index-document index.html --error-document error.html  
aws s3api get-bucket-policy --bucket paraborrar123
```

What are EBS Volumes?

Volumes behave like raw, unformatted block devices, with user supplied device names and a block device interface. You can create a file system on top of Amazon EBS volumes, or use them in any other way you would use a block device (like a hard drive).

What are Snapshots?

Snapshots are a point-in-time of a volume, which are persisted to Amazon S3. These snapshots can be used as the starting point for new Amazon EBS volumes, and protect data for long-term durability. The same snapshot can be used to instantiate as many volumes as you wish. These snapshots can be copied across AWS regions, making it easier to leverage multiple AWS regions for geographical expansion, data center migration and disaster recovery.

Example 1: To describe a snapshot

The following describe-snapshots example describes the specified snapshot.

```
aws ec2 describe-snapshots \
--snapshot-id snap-1234567890abcdef0
```

Output:

```
{
  "Snapshots": [
    {
      "Description": "This is my snapshot",
      "Encrypted": false,
      "VolumeId": "vol-049df61146c4d7901",
      "State": "completed",
      "VolumeSize": 8,
      "StartTime": "2014-02-28T21:28:32.000Z",
      "Progress": "100%",
      "OwnerId": "012345678910",
      "SnapshotId": "snap-1234567890abcdef0"
    }
  ]
}
```

Example 2: To describe snapshots using filters

The following `describe-snapshots` example describes all snapshots owned by the specified AWS account that are in the pending state.

```
aws ec2 describe-snapshots \
--owner-ids 012345678910 \
--filters Name=status,Values=pending
```

Output:

```
{
  "Snapshots": [
    {
      "Description": "This is my copied snapshot",
      "Encrypted": true,
      "VolumeId": "vol-1234567890abcdef0",
      "State": "pending",
      "VolumeSize": 8,
      "StartTime": "2014-02-28T21:37:27.000Z",
      "Progress": "87%",
      "OwnerId": "012345678910",
      "SnapshotId": "snap-066877671789bd71b"
    }
  ]
}
```

Example 3: To describe tagged snapshots and filter the output

The following `describe-snapshots` example describes all snapshots that have the tag `Group=Prod`. The output is filtered to display only the snapshot IDs and the time the snapshot was started.

```
aws ec2 describe-snapshots \
  --filters Name>tag:Group,Values=Prod \
  --query "Snapshots[*].{ID:SnapshotId,Time:StartTime}"
```

Output:

```
[  
  {  
    "ID": "snap-1234567890abcdef0",  
    "Time": "2014-08-04T12:48:18.000Z"  
  }  
]
```

When you make periodic snapshots of a volume, the snapshots are incremental, and only the blocks on the device that have changed since your last snapshot are saved in the new snapshot. When you delete a snapshot, only the data not needed for any other snapshot is removed. So regardless of which prior snapshots have been deleted, all active snapshots will have access to all the information needed to restore the volume.

You cannot delete a snapshot of the root device of an EBS volume used by a registered AMI. You must first de-register the AMI before you can delete the snapshot.

delete-snapshot --snapshot-id <value> [--dry-run | --no-dry-run] [--cli-input-json <value>] [--generate-cli-skeleton <value>]
--snapshot-id (string)

The ID of the EBS snapshot.

--dry-run | --no-dry-run (boolean)

Checks whether you have the required permissions for the action, without actually making the request, and provides an error response. If you have the required permissions, the error response is DryRunOperation . Otherwise, it is UnauthorizedOperation .

--cli-input-json (string)

Performs service operation based on the JSON string provided. The JSON string follows the format provided by --generate-cli-skeleton.. It is not possible to pass arbitrary binary values using a JSON-provided value as the string will be taken literally.

--generate-cli-skeleton (string)

Prints a JSON skeleton to standard output without sending an API request.

```
aws ec2 delete-snapshot --snapshot-id snap-1234567890abcdef0
```

EBS Volumes available to remove

Use this command to list of all the volumes available to remove

```
ec2-describe-volumes --region eu-west-1 | grep available | awk '{print $2}' | tr '\n' ' '
```

All Snapshots

Use this command to list of all snapshots

```
ec2-describe-snapshots --region eu-west-1 | grep SNAPSHOT | awk '{print $2}' | sort | uniq
```

All Snapshots in use

Use this command to list all snapshots in use

```
ec2-describe-images --region eu-west-1 | grep BLOCKDEVICEMAPPING | awk '{print $3}' | sort | uniq
```

All Snapshots not in use by any AMI's

Use this command to list all snapshots not associated with an AMI

```
$(ec2-describe-snapshots --region eu-west-1 | grep SNAPSHOT | awk '{print $2}' | sort | uniq) | tr ' ' '\n')  
  
$(ec2-describe-images --region eu-west-1 | grep BLOCKDEVICEMAPPING | awk '{print $3}' | sort | uniq) | tr ' ' '\n') | tr '\n' ''
```

How to delete all EBS Volumes which are unattached

Use this command to delete all volumes which are unattached

```
// delete all volumes which are unattached  
ec2-delete-volume --region eu-west-1 $(ec2-describe-volumes --region eu-west-1 | grep available | awk '{print $2}' | tr '\n' '')
```

How to delete all Snapshots which are not in use

Use this command to delete all snapshots not in use

```
// run as a single command
for s in $(comm -23 <(echo $($aws ec2 describe-snapshots --region eu-west-1 | grep SNAPSHOT | awk '{print $2}' | sort | uniq) | tr ' ' '\n')
        <($aws ec2 describe-images --region eu-west-1 | grep BLOCKDEVICEMAPPING | awk '{print $3}' | sort | uniq) | tr ' ' '\n') | tr '\n' ' ')
do
    echo Deleting snapshot $s
    $aws ec2 delete-snapshot --region eu-west-1 $s
done
```

This script can help you find and remove unused AWS snapshots and volumes.

There is hardcoded list of regions that it searches, adjust the value to suit your needs.

Use `snapshot.py snapshot-report` to generate `report.csv` containing information about all snapshots.

`snapshot.py snapshot-cleanup` lets you interactively delete snapshot if it finds it is referencing unexisting resources.

```
./snapshots.py --help
Usage: snapshots.py [OPTIONS] COMMAND [ARGS]...

    Helper commands for Snapshots management.

Options:
    --help  Show this message and exit.

Commands:
    snapshot-cleanup  Find and delete unreferenced snapshots.
    snapshot-delete   Delete single snapshot by id.
    snapshot-report   Find unreferenced snapshots.
    volume-cleanup    Find and delete unused volumes.
```

What is AWS CloudFormation?

AWS CloudFormation ofrece un lenguaje común para describir y provisionar los recursos de la infraestructura en el entorno de la nube. CloudFormation permite utilizar lenguajes de programación o un archivo de texto simple para modelar y aprovisionar, de una manera segura y automatizada, todos los recursos necesarios para las aplicaciones en todas las regiones y cuentas. Esto proporciona una única fuente de información para los recursos de AWS.

When you use AWS CloudFormation, you work with *templates* and *stacks*. You create templates to describe your AWS resources and their properties. Whenever you create a stack, AWS CloudFormation provisions the resources that are described in your template.

Topics

- [Templates](#)
- [Stacks](#)
- [Change Sets](#)

Las templates de CloudFormation ofrecen varias ventajas:

Seguir un formato conocido: una plantilla de CloudFormation es un archivo de texto con formato JSON (JavaScript Object Notation) o YAML que describe la infraestructura de AWS necesaria para ejecutar una aplicación o un servicio, junto con las interconexiones que existan entre ellos.

Administrar relaciones: las plantillas contienen una descripción concisa de las relaciones existentes entre los recursos, por ejemplo las instancias EC2 que deben asociarse con un balanceador de carga de Elastic Load Balancing, o la necesidad de que un volumen de EBS esté en la misma zona de disponibilidad de EC2 que la instancia a la que se conecta.

Uso constante: la utilización de los parámetros de las plantillas permite utilizar una única plantilla para muchas implementaciones de la infraestructura con diferentes valores de configuración, como cuántas instancias hay que implementar para la aplicación.

Obtener comentarios útiles: las plantillas también ofrecen propiedades de salida para comunicar al usuario los resultados de implementación o información relativa a la configuración. Por ejemplo, cuando se incluye en una instancia, una plantilla puede incluir la URL del punto de conexión de Elastic Load Balancing que el usuario debería utilizar para conectarse a la aplicación que acaba de incluir en una instancia.

Evitar conflictos: todos los recursos de AWS incluidos en una plantilla se identifican mediante nombres lógicos, lo que permite crear varias stacks a partir de una plantilla sin temor a encontrarse con conflictos de nomenclatura entre recursos de AWS.

Escribir y listo: utilice cualquier método para lanzar una stack sin tener que registrar previamente la plantilla en AWS CloudFormation.

Visualización de la stack: CloudFormation Designer permite ver las plantillas en un diagrama. La herramienta permite ver fácilmente los recursos de AWS y sus relaciones, y organizarlos espacialmente de modo tal que el diagrama resulte comprensible. Puede modificar las plantillas arrastrando los elementos con el ratón y con el editor de JSON integrado. Los cambios que haga en el diagrama se trasladarán automáticamente al código JSON de la plantilla.

Buscar recursos: AWS CloudFormation conserva una copia de la plantilla de stack para poder utilizar la consola de administración de AWS, las herramientas de línea de comandos o las API con objeto de consultar las configuraciones precisas de recursos que se aplicaron durante la creación de la stack.

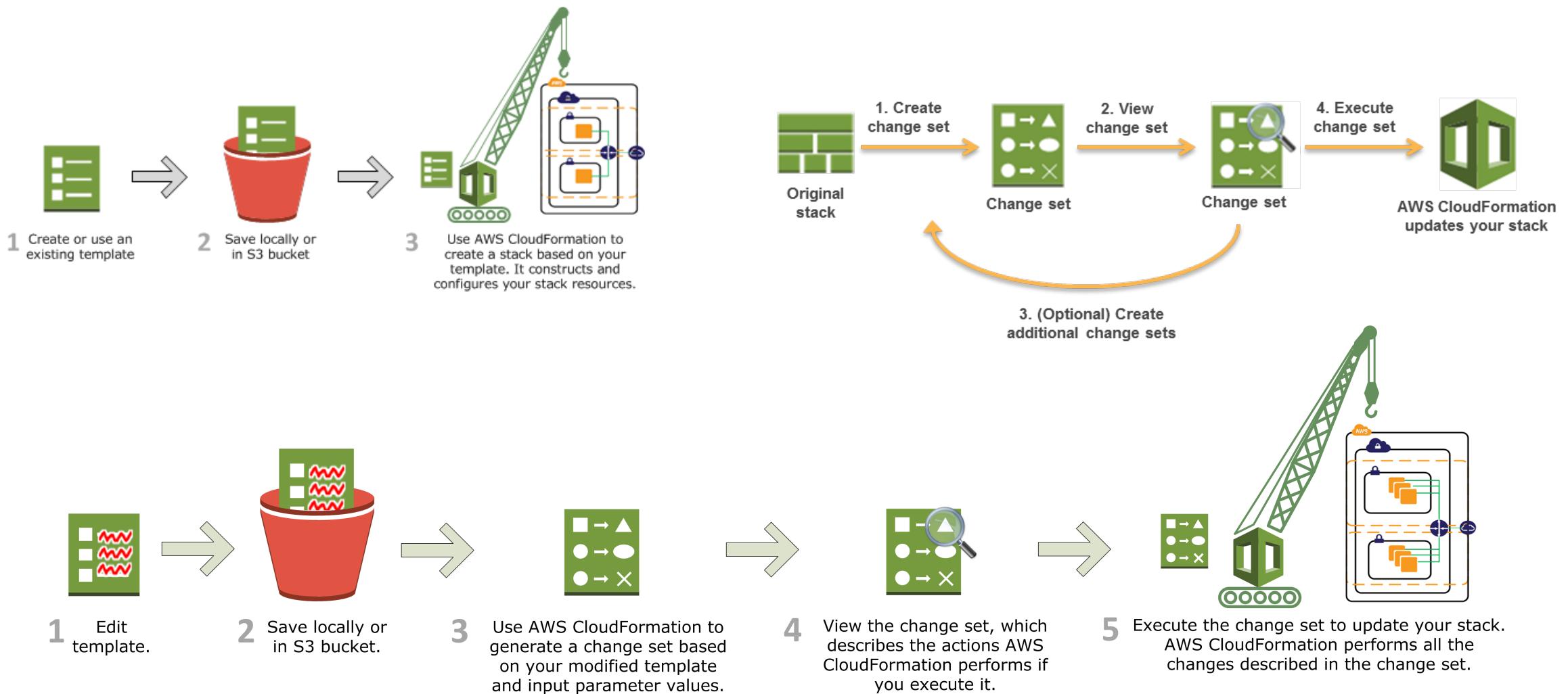
Automatizar: tiene la opción de automatizar la generación de plantillas con la utilización de un lenguaje de programación o una herramienta de su elección. También puede automatizar la creación de stacks a partir de plantillas mediante la API de CloudFormation, los SDK de AWS o la CLI de AWS.

Una stack es un conjunto de recursos que se producen como resultado de la inclusión de una plantilla en una instancia. Para crear una , es necesario proporcionar una plantilla y todos los parámetros necesarios para AWS CloudFormation. Según la plantilla y las dependencias especificadas en ella, AWS CloudFormation determina qué recursos de AWS deben crearse y en qué orden.

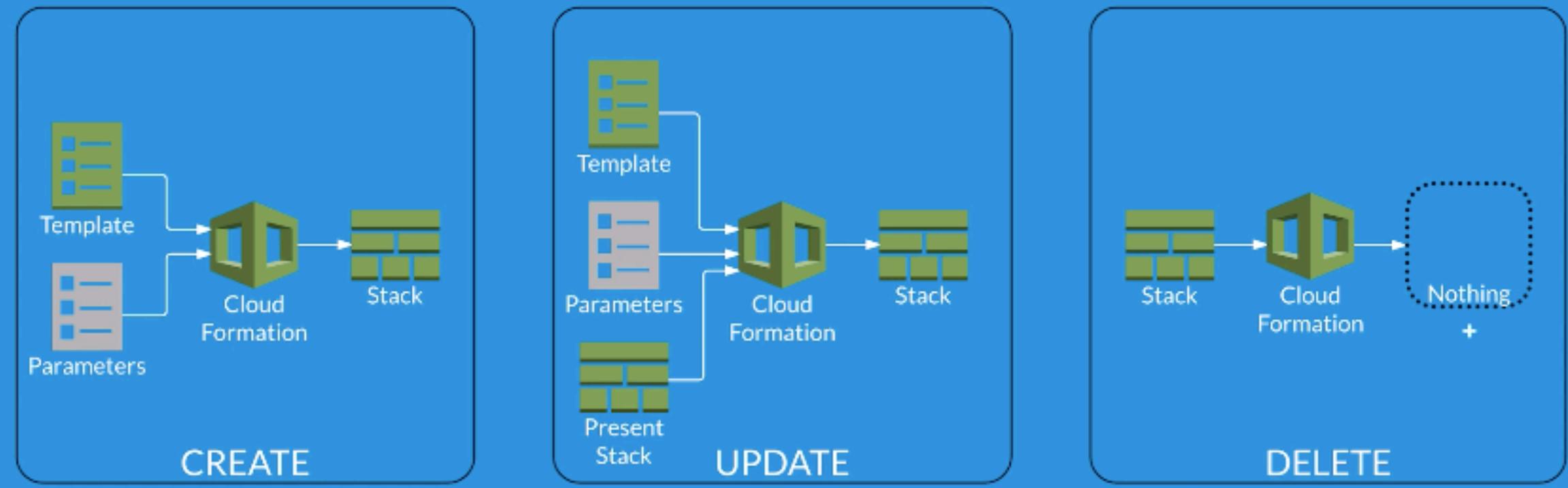
Para actualizar una stack, debe proporcionar una plantilla con la configuración deseada de todos los recursos de la stack. Puede modificar las propiedades de los recursos existentes de su stack para que reaccionen a los cambios del entorno o a los nuevos requisitos de las aplicaciones. Por ejemplo, puede cambiar los umbrales de alarma de sus alarmas de AWS CloudWatch o actualizar la AMI que se ejecuta en una instancia de la stack. AWS CloudFormation se encarga de aplicar dichos cambios a los diferentes recursos de la stack. En muchos casos, los cambios se realizarán sin afectar a la aplicación en ejecución. No obstante, si un cambio no se puede realizar de forma dinámica (como la actualización de la AMI en una instancia EC2), AWS CloudFormation creará un recurso nuevo, lo reconectará a la stack y, una vez que el servicio ha determinado que la actualización completa se realizará correctamente, eliminará el recurso anterior.

AWS CloudFormation creará o actualizará una stack completa. Si una stack no se puede crear o actualizar completamente, AWS CloudFormation la desmontará. A efectos de depuración, la operación de restauración puede deshabilitarse y volver a intentar crear o actualizar la stack manualmente en otro momento.

AWS CloudFormation Designer también permite crear o modificar la plantilla de una stack y luego enviarla a AWS CloudFormation para crear o actualizar la stack. AWS CloudFormation Designer está disponible con la consola de administración de AWS.



Stack: State Machine



Stack Status Lifecycle

CREATE	
OK	ERROR
CREATE_IN_PROGRESS	
CREATE_COMPLETE	CREATE_FAILED ROLLBACK_IN_PROGRESS ROLLBACK_COMPLETE ** ROLLBACK_FAILED ** Rollback failure requires stack deletion! It is a fairly serious condition and may cause DELETE_FAILED too.

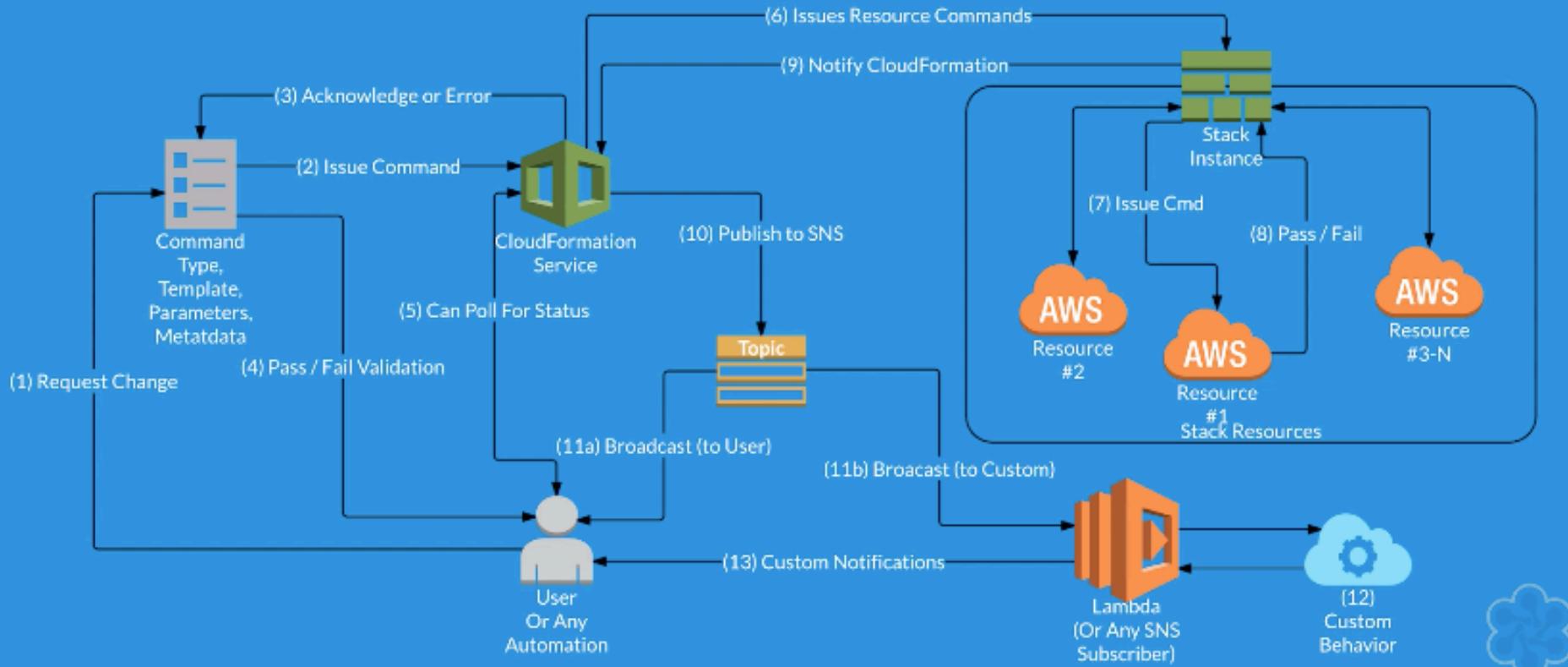
UPDATE	
OK	ERROR
UPDATE_IN_PROGRESS	
UPDATE_COMPLETE	UPDATE_FAILED UPDATE_ROLLBACK_IN_PROGRESS UPDATE_ROLLBACK_COMPLETE_CLEANUP_IN_PROGRESS UPDATE_ROLLBACK_COMPLETE ** UPDATE_ROLLBACK_FAILED ** Rollback failure requires stack deletion! It is a fairly serious condition and may cause DELETE_FAILED too.

DELETE	
OK	ERROR
DELETE_IN_PROGRESS	
DELETE_COMPLETE	** DELETE_FAILED ** Deletion failure is serious. This status requires either a retry on the part of the user, or if that fails again, a manual intervention from the CloudFormation team. Submit a support ticket to ask for hard delete of the Stack. This will only remove the Stack from CloudFormation, any remaining resources must be cleaned manually.

Resource: State Machine



CloudFormation Data Flow



Proceso de arranque de aplicaciones y gestión de actualizaciones

AWS CloudFormation ofrece un número de scripts de ayudante que se pueden implementar en sus instancias EC2. Estos scripts proporcionan una forma simple de leer metadatos de recursos de la stack y utilizarlos para configurar su aplicación, implementar paquetes y archivos en su instancia que aparezcan en su plantilla y reaccionar a las actualizaciones de la stack como, por ejemplo, en caso de cambios en la configuración o actualizaciones de la aplicación.

Se encuentran disponibles los siguientes scripts:

- `cfn-get-metadata`: recupere metadatos adjuntos a los recursos de la plantilla.
- `cfn-init`: descargue e instale paquetes y archivos descritos en la plantilla.
- `cfn-signal`: indique al flujo de trabajo de creación de la stack que su aplicación está en funcionamiento y lista para incorporar tráfico.
- `cfn-hup`: un demonio para escuchar las actualizaciones de stack que se iniciaron a través de la consola de AWS, las herramientas de línea de comandos o el API directamente y ejecutar los ganchos específicos de su aplicación para reaccionar frente a estos cambios.

JSON

El siguiente ejemplo muestra un fragmento de código de plantilla con formato JSON.

```
{  
  "AWSTemplateFormatVersion" : "version date",  
  "Description" : "JSON string",  
  "Metadata" : {  
    template metadata  
  },  
  "Parameters" : {  
    set of parameters  
  },  
  "Mappings" : {  
    set of mappings  
  },  
  "Conditions" : {  
    set of conditions  
  },  
  "Transform" : {  
    set of transforms  
  },  
  "Resources" : {  
    set of resources  
  },  
  "Outputs" : {  
    set of outputs  
  }  
}
```

AWSTemplateFormatVersion (opcional) identifica las capacidades de la plantilla. La última versión de formato de plantilla es **2010-09-09** y es en la actualidad el único valor válido.

JSON

```
"AWSTemplateFormatVersion" : "2010-09-09"
```

Description (opcional) le permite incluir comentarios acerca de su plantilla.

JSON

```
"Description" : "Here are some details about the template."
```

```
{  
    "AWSTemplateFormatVersion" : "version date",  
    "Description" : "JSON string",  
    "Metadata" : {  
        template metadata  
    },  
    "Parameters" : {  
        set of parameters  
    },  
    "Mappings" : {  
        set of mappings  
    },  
    "Conditions" : {  
        set of conditions  
    },  
    "Transform" : {  
        set of transforms  
    },  
    "Resources" : {  
        set of resources  
    },  
    "Outputs" : {  
        set of outputs  
    }  
}
```

Metadata (opcional) para incluir objetos JSON o YAML arbitrarios que proporcionan detalles sobre la plantilla..

importante

Durante la actualización de una stack, no puede actualizar la sección Metadata . Puede actualizarse solo cuando incluye cambios que añaden, modifican o eliminan recursos.

JSON

```
"Metadata" : {  
    "Instances" : {"Description" : "Information about the instances"},  
    "Databases" : {"Description" : "Information about the databases"}  
}
```

Claves de metadatos

Algunas características de AWS CloudFormation recuperan ajustes o información de configuración se define en la sección Metadata. Puede definir esta información en las siguientes claves de metadatos específicos de AWS CloudFormation:

AWS::CloudFormation::Designer

Describe cómo se presentan los recursos en AWS CloudFormation Designer (Designer).

AWS::CloudFormation::Interface

Para crear plantillas de creación de infra donde el usuario mete los datos

JSON

```
"Metadata" : {  
    "AWS::CloudFormation::Interface" : {  
        "ParameterGroups" : [  
            {  
                "Label" : { "default" : "Network Configuration" },  
                "Parameters" : [ "VPCID", "SubnetId", "SecurityGroupID" ]  
            },  
            {  
                "Label" : { "default": "Amazon EC2 Configuration" },  
                "Parameters" : [ "InstanceType", "KeyName" ]  
            }  
        ],  
        "ParameterLabels" : {  
            "VPCID" : { "default" : "Which VPC should this be deployed to?" }  
        }  
    }  
}
```



Parameters

Network Configuration

Which VPC should this be deployed to? Search by ID, or Name tag value
VpcId of your existing Virtual Private Cloud (VPC)

SubnetId Search by ID, or Name tag value
The list of SubnetIds in your Virtual Private Cloud (VPC)

SecurityGroupID Search by ID, name or Name tag value
VpcId of your existing Virtual Private Cloud (VPC)

Amazon EC2 Configuration

InstanceType m1.small Web

KeyName Search
Name of an existing EC2 KeyPair to enable SSH access to th

AWS::CloudFormation::Init

Define tareas de configuración para el script auxiliar cfn-init. Este script es útil para la configuración y la instalación de aplicaciones en instancias EC2.

Para instalar, configurar e iniciar automáticamente aplicaciones en instancias Amazon EC2. Esto le permite duplicar fácilmente implementaciones y actualizar instalaciones existentes sin conectarse directamente a la instancia, lo que permite ahorrar mucho tiempo y esfuerzo.

AWS CloudFormation incluye un conjunto de scripts auxiliares (cfn-init, cfn-signal, cfn-get-metadata y cfn-hup) que se basan en cloud-init. Se llama a estos scripts auxiliares desde las plantillas de AWS CloudFormation para instalar, configurar y actualizar aplicaciones en instancias Amazon EC2 que se encuentran en la misma plantilla.

AWS CloudFormation tiene estos scripts:

- **cfn-init:** Proceso que arranca el servicio. Permite instalar paquetes, actualizaciones y demás
- **cfn-signal:** A simple wrapper que avisa a AWS CloudFormation WaitCondition para esperar a que terminen de crearse otros recursos del stack. Por ejemplo IGW para salir a internet antes de descargar actualizaciones.
- **cfn-get-metadata:** Script para resolver metadata que se usan en funciones intrinsic. Sirve para que las plantillas sean dinámicas y resuelvan cosas. Por ejemplo DNS endpoint de RDS
- **cfn-hup:** Un daemon que actualiza los metadatos y que permite lanzar custom hooks cuando se detectan cambios.

Si la plantilla cfn-init llama al script, el script buscará la ruta raíz de metadatos de recursos en la clave de metadatos AWS::CloudFormation::Init.

El script auxiliar cfn-init lee metadatos de plantilla de la clave AWS::CloudFormation::Init y :

- Obtener y analizar metadatos de AWS CloudFormation
- Instalar paquetes
- Escribir archivos en el disco
- Habilitar/deshabilitar e iniciar/detener servicios

nota

Si utiliza cfn-init para actualizar un archivo existente, crea una copia de seguridad del archivo original en el mismo directorio con una extensión .bak. Por ejemplo, si actualiza */path/to/file_name*, la acción produce dos archivos: */path/to/file_name.bak* incluye el contenido original del archivo y */path/to/file_name* incluye los contenidos actualizados.

We recommend you use the latest version of the AWS CloudFormation bootstrap scripts when you launch an Amazon Linux AMI. To ensure that happens, add a `yum update` command to the user data script, as shown in the following example:

```
"MyInstance": {
    "Type": "AWS::EC2::Instance",
    "Metadata": {
        :
    },
    "Properties": {
        :
        "UserData" : { "Fn::Base64" : { "Fn::Join" : [ "", [
            "#!/bin/bash\n",
            "yum update -y aws-cfn-bootstrap\n",
            "/opt/aws/bin/cfn-init ",
            "    --stack ", { "Ref" : "AWS::StackName" },
            "    --resource MyInstance ",
            "    --region ", { "Ref" : "AWS::Region" }, "\n"
        ]]}}
    }
}
```

AWS::CloudFormation::Init

JSON

```
"Resources": {  
    "MyInstance": {  
        "Type": "AWS::EC2::Instance",  
        "Metadata" : {  
            "AWS::CloudFormation::Init" : {  
                "config" : {  
                    "packages" : {  
                        :  
                    },  
                    "groups" : {  
                        :  
                    },  
                    "users" : {  
                        :  
                    },  
                    "sources" : {  
                        :  
                    },  
                    "files" : {  
                        :  
                    },  
                    "commands" : {  
                        :  
                    },  
                    "services" : {  
                        :  
                    }  
                }  
            }  
        }  
    }  
},  
"Properties": {  
    :  
}  
}
```

Sintaxis

La configuración se divide en secciones. El siguiente fragmento de código de plantilla muestra cómo se pueden adjuntar metadatos de cfn-init a un recurso de instancia Amazon EC2 dentro de la plantilla.

Los metadatos se organizan en claves de configuración, que puede agrupar en configsets. Puede especificar un configset cuando llama a cfn-init en la plantilla. Si no especifica un configset, cfn-init buscará una clave de configuración única denominada *config*.

nota

El script auxiliar cfn-init procesa estas secciones de configuración en el siguiente orden: paquetes, grupos, usuarios, fuentes, archivos, comandos y, a continuación, servicios. Si necesita un orden diferente, separe las secciones en diferentes claves de configuración y, a continuación, utilice un configset que especifique el orden en el que deben procesarse las claves de configuración.

JSON

```
"AWS::CloudFormation::Init" : {
    "configSets" : {
        "ascending" : [ "config1" , "config2" ],
        "descending" : [ "config2" , "config1" ]
    },
    "config1" : {
        "commands" : {
            "test" : {
                "command" : "echo \\$CFNSTEST\\ > test.txt",
                "env" : { "CFNSTEST" : "I come from config1." },
                "cwd" : "~"
            }
        }
    },
    "config2" : {
        "commands" : {
            "test" : {
                "command" : "echo \\$CFNSTEST\\ > test.txt",
                "env" : { "CFNSTEST" : "I come from config2" },
                "cwd" : "~"
            }
        }
    }
}
```

Configsets

Si desea crear más de una clave de configuración y que cfn-init las procese en un orden específico, crear un configset que contenga las claves de configuración en el orden deseado.

Configset individual

El siguiente fragmento de código de plantilla crea configsets denominados ascending y descending que contienen cada uno dos claves de configuración.

Llamadas a cfn-init relacionadas

Las siguientes llamadas a cfn-init de ejemplo se refieren a los configsets de ejemplo anteriores. Las llamadas de ejemplo se abrevian para facilitar la lectura, consulte [cfn-init](#) para ver la sintaxis completa.

- Si una llamada a cfn-init especifica el configset ascending:

```
cfn-init -c ascending
```

el script procesa config1 y, a continuación, procesa config2 y el archivo test.txt contendrá el texto I come from config2.

- Si una llamada a cfn-init especifica el configset descending:

```
cfn-init -c descending
```

el script procesa config2 y, a continuación, procesa config1 y el archivo test.txt contendrá el texto I come from config1.

Comandos

Puede utilizar la clave de comandos para ejecutar comandos en la instancia EC2. Los comandos se procesan en orden alfabético por nombre.

Clave	Descripción
command	Obligatorio. Una matriz o bien una cadena que especifica el comando que se va a ejecutar. Si utiliza una matriz, no es necesario que encierre caracteres de espacio o incluya parámetros de comando entre comillas. No utilice la matriz para especificar varios comandos.
env	Opcional. Establece las variables de entorno para el comando. Esta propiedad sobrescribe, en lugar de anexar, el entorno existente.
cwd	Opcional. El directorio de trabajo
test	Opcional. Un comando de prueba que determina si cfn-init ejecuta comandos que se especifican en la clave del comando. Si se supera la prueba, cfn-init ejecuta los comandos. El script cfn-init script ejecuta la prueba en un intérprete de comandos, como Bash o cmd.exe. Que la prueba se supere depende del código de salida que el intérprete devuelve. En el caso de Linux, el comando de prueba debe devolver un código de salida de 0 para superar la prueba. En el caso de Windows, el comando de prueba debe devolver un %ERRORLEVEL% de 0.
ignoreErrors	Opcional. Un valor booleano que determina si cfn-init continúa ejecutándose si el comando contenido en la clave del comando falla (devuelve un valor distinto de cero). Establezca en true si desea que cfn-init continúe ejecutándose incluso si el comando falla. Establezca en false si desea que cfn-init deje de ejecutarse si el comando falla. El valor predeterminado es false.
waitAfterCompletion	Opcional. Para sistemas Windows solamente. Especifica el tiempo durante el que esperar (en segundos) después de que un comando ha terminado en caso de que el comando provoque un reinicio. El valor predeterminado es de 60 segundos y un valor de "forever" indica a cfn-init que salga y reanude la actividad solo después de que se haya completado el reinicio. Establezca este valor en 0 si no desea esperar cada comando.

Ejemplo

El siguiente fragmento de código de ejemplo llama al comando echo si el archivo ~/test.txt no existe.

JSON

```
"commands" : {
    "test" : {
        "command" : "echo \\$MAGIC\\ > test.txt",
        "env" : { "MAGIC" : "I come from the environment!" },
        "cwd" : "~",
        "test" : "test ! -e ~/test.txt",
        "ignoreErrors" : "false"
    },
    "test2" : {
        "command" : "echo \\$MAGIC2\\ > test2.txt",
        "env" : { "MAGIC2" : "I come from the environment!" },
        "cwd" : "~",
        "test" : "test ! -e ~/test2.txt",
        "ignoreErrors" : "false"
    }
}
```

Archivos

Puede utilizar la clave files para crear archivos en la instancia EC2. El contenido puede ser en línea en la plantilla o el contenido puede extraerse de una URL. Los archivos se escriben en el disco en orden lexicográfico. En la tabla siguiente se muestran las claves admitidas.

Clave	Descripción
content	<p>Una cadena o un objeto JSON con formato correcto. Si utiliza un objeto JSON como su contenido, el objeto JSON se escribirá en un archivo en el disco. Cualquier función intrínseca como, por ejemplo, Fn::GetAtt o Ref se evalúa antes de que el objeto JSON se escriba en el disco. Al crear un symlink, especifique el destino de symlink como el contenido.</p> <p>nota</p> <p>Si crea un symlink, el script auxiliar modifica los permisos del archivo de destino. Actualmente, no puede crear un symlink sin modificar los permisos del archivo de destino.</p>
origen	Una URL desde la que cargar el archivo. Esta opción no se puede especificar con la clave de contenido.
encoding	El formato de la codificación. Solo se utiliza si el contenido es una cadena. La codificación no se aplica si utiliza una fuente. Valores válidos: plain base64
grupo	El nombre del grupo propietario de este archivo. No es compatible con sistemas Windows.
owner	El nombre del usuario propietario de este archivo. No es compatible con sistemas Windows.
mode	Un valor octal de seis dígitos que representa el modo para este archivo. No es compatible con sistemas Windows. Utilice los tres primeros dígitos para symlinks y los últimos tres dígitos para la configuración de permisos. Para crear un symlink, especifique 120xxx , donde xxxdefine los permisos del archivo de destino. Para especificar los permisos de un archivo, utilice los tres últimos dígitos, como 000644 .
autenticación	El nombre de un método de autenticación que se debe utilizar. Esto anula cualquier autenticación predeterminada. Puede utilizar esta propiedad para seleccionar un método de autenticación definido con el recurso AWS::CloudFormation::Authentication .
context	Especifica un contexto para los archivos que se procesan como plantillas Mustache . Para utilizar esta clave, debe tener instalado aws-cfn-bootstrap 1.3-11 o posterior, así como pystache .

Ejemplos

El siguiente fragmento de código de ejemplo crea un archivo denominado setup.mysql como parte de una instalación de mayor tamaño.

ejemplo JSON

```
"files" : {
    "/tmp/setup.mysql" : {
        "content" : { "Fn::Join" : [ "", [
            "CREATE DATABASE ", { "Ref" : "DBName" }, ";\\n",
            "CREATE USER ''", { "Ref" : "DBUsername" }, "'@'localhost' IDENTIFIED BY ''",
            { "Ref" : "DBPassword" }, ";\\n",
            "GRANT ALL ON ", { "Ref" : "DBName" }, ".*" TO '', { "Ref" : "DBUsername" },
            "'@'localhost';\\n",
            "FLUSH PRIVILEGES;\\n"
        ] ] },
        "mode" : "000644",
        "owner" : "root",
        "group" : "root"
    }
}
```



Paquetes

Puede utilizar la clave de paquetes para descargar e instalar aplicaciones y componentes previamente empaquetados. En los sistemas Windows, la clave de paquetes solo admite el instalador MSI.

Formatos de paquetes admitidos

El script cfn-init admite actualmente los siguientes formatos de paquete: apt, msi, python, rpm, rubygems y yum. Los paquetes se procesan en el orden siguiente: rpm, yum/apt y, a continuación, rubygems y python. No existe ningún orden establecido entre rubygems y python, y los paquetes dentro de cada administrador de paquetes no tienen garantizada la instalación en ningún orden.

Especificación de versiones

Dentro de cada administrador de paquetes, cada paquete se especifica con un nombre de paquete y una lista de versiones. La versión puede ser una cadena, una lista de versiones o una cadena o lista vacía. Una cadena o lista vacía indica que se debe usar la versión más reciente. Para el administrador de rpm, la versión se especifica como una ruta a un archivo en el disco o una URL.

Si especifica una versión de un paquete, cfn-init intentará instalar esa versión incluso si ya hay una versión más reciente del paquete instalada en la instancia. Algunos administradores de paquetes admiten varias versiones, pero puede haber otros que no. Consulte la documentación del administrador de paquetes para obtener más información. Si no especifica una versión y ya hay una versión del paquete instalada, el script cfn-init no instalará una nueva versión; presupondrá que desea mantener y utilizar la versión existente.

El siguiente fragmento de código especifica una URL de versión para rpm, solicita las últimas versiones de yum y la versión 0.10.2 de chef de rubygems:

JSON

```
"rpm" : {  
    "epel" : "http://download.fedoraproject.org/pub/epel/5/i386/epel-release-5-4.noarch.rpm"  
},  
"yum" : {  
    "httpd" : [],  
    "php" : [],  
    "wordpress" : []  
},  
"rubygems" : {  
    "chef" : [ "0.10.2" ]  
}
```



El siguiente fragmento de código especifica una URL para un paquete MSI:

JSON

```
"msi" : {  
    "awscli" : "https://s3.amazonaws.com/aws-cli/AWSCLI64.msi"  
}
```

Servicios

Puede utilizar la clave de servicios para definir qué servicios deben habilitarse o deshabilitarse cuando se lanza la instancia. En sistemas Linux, esta clave es compatible con sysvinit. En sistemas Windows, es compatible con el administrador de servicio de Windows.

La clave de servicios también le permite especificar dependencias en orígenes, paquetes y archivos, de manera que, si es necesario realizar un reinicio debido a los archivos que se están instalado, cfn-init se encargará de reiniciar el servicio. Por ejemplo, si descarga el paquete del servidor HTTP Apache, la instalación del paquete comenzará automáticamente el servidor HTTP Apache durante el proceso de creación de la stack. Sin embargo, si la configuración del servidor HTTP Apache se actualiza más adelante en el proceso de creación de la stack, la actualización no surtirá efecto hasta que se reinicie el servidor Apache. Puede utilizar la clave de servicios para asegurarse de que se reinicia el servicio HTTP Apache.

Clave	Descripción
ensureRunning	<p>Establezca en true para garantizar que el servicio se ejecuta una vez que cfn-init termine.</p> <p>Establezca en false para garantizar que el servicio no se ejecuta una vez que cfn-init termine.</p> <p>Omita esta clave para no realizar ningún cambio en el estado del servicio.</p>
enabled	<p>Establezca en true para garantizar que el servicio comenzará automáticamente al arrancar.</p> <p>Establezca en false para garantizar que el servicio no comenzará automáticamente al arrancar.</p> <p>Omita esta clave para no realizar ningún cambio a esta propiedad.</p>
files	Una lista de archivos. Si cfn-init cambia uno directamente a través del bloque de archivos, este servicio se reinicia.
sources	Una lista de directorios. Si cfn-init amplía un archivo hacia uno de estos directorios, este servicio se reinicia.
packages	Una asignación del administrador de paquetes con la lista de nombres de paquetes. Si cfn-init instala o actualiza uno de estos paquetes, este servicio se reinicia.
commands	Una lista de nombres de comandos. Si cfn-init ejecuta el comando especificado, este servicio se reinicia.

Linux

El siguiente fragmento de código de Linux configura los servicios de la siguiente manera:

- El servicio nginx se reinicia si cfn-init modifica /etc/nginx/nginx.conf o /var/www/html.
- El servicio php-fastcgi se reiniciará si cfn-init instala o actualiza php o spawn-fcgi mediante yum.
- El servicio sendmail se interrumpirá y deshabilitará.

JSON

```
"services" : {  
    "sysvinit" : {  
        "nginx" : {  
            "enabled" : "true",  
            "ensureRunning" : "true",  
            "files" : ["/etc/nginx/nginx.conf"],  
            "sources" : ["/var/www/html"]  
        },  
        "php-fastcgi" : {  
            "enabled" : "true",  
            "ensureRunning" : "true",  
            "packages" : { "yum" : ["php", "spawn-fcgi"] }  
        },  
        "sendmail" : {  
            "enabled" : "false",  
            "ensureRunning" : "false"  
        }  
    }  
}
```

Windows

El siguiente fragmento de código de Windows inicia el servicio cfn-hup, lo establece en automático y reinicia el servicio si cfn-init modifica los archivos de configuración especificados:

JSON

```
"services" : {  
    "windows" : {  
        "cfn-hup" : {  
            "enabled" : "true",  
            "ensureRunning" : "true",  
            "files" : ["c:\\cfn\\cfn-hup.conf", "c:\\cfn\\hooks.d\\cfn-auto-reloader.conf"]  
        }  
    }  
}
```

Orígenes

Puede utilizar la clave de origen para descargar un archivo de almacenamiento y extraerlo en un directorio de destino en la instancia EC2. Esta clave es totalmente compatible con los sistemas Linux y Windows.

Formatos admitidos

Entre los formatos compatibles están star, tar+gzip, tar+bz2 y zip.

GitHub

Si utiliza GitHub como un sistema de control de origen, puede utilizar cfn-init y el mecanismo del paquete de origen para extraer una versión específica de su aplicación. GitHub le permite crear un zip o un tar a partir de una versión específica a través de una URL de la siguiente manera:



```
https://github.com/<your directory>/(<zipball|tarball>)/<version>
```

Por ejemplo, el siguiente fragmento de código extrae la versión maestra como un archivo .tar.

S3 Bucket

El siguiente ejemplo descarga un archivo zip de un bucket de Amazon S3 y lo desempaquetá en /etc/myapp:

JSON

```
"sources" : {  
    "/etc/myapp" : "https://s3.amazonaws.com/mybucket/myapp.tar.gz"  
}
```

Usuarios

Puede utilizar la clave de usuarios para crear usuarios de Linux/UNIX en la instancia EC2. La clave de usuarios no es compatible con sistemas Windows.

En la tabla siguiente se muestran las claves admitidas.

Clave	Descripción
uid	ID de usuario. El proceso de creación falla si existe un nombre de usuario con otro ID. Si el ID de usuario ya se ha asignado a un usuario existente, el sistema operativo podría rechazar la solicitud de creación.
groups	Lista de nombres de grupos. Se añadirá al usuario a cada grupo en la lista.
homeDir	Directorio de inicio del usuario.

Los usuarios se crean como usuarios de un sistema no interactivo con un shell de /sbin/nologin. Esto es así por diseño y no se puede modificar.

JSON

```
"users" : {  
    "myUser" : {  
        "groups" : [ "groupOne", "groupTwo" ],  
        "uid" : "50",  
        "homeDir" : "/tmp"  
    }  
}
```



AWS::CloudFormation::Authentication

para especificar las credenciales de autenticación de los archivos u orígenes indicados mediante el recurso [AWS::CloudFormation::Init](#).

Si desea incluir información de autenticación para un archivo u origen especificados con AWS::CloudFormation::Init, utilice la propiedad uris si el origen es un URI o la propiedad buckets si el origen es un bucket de Amazon S3.

También puede especificar información de autenticación para los archivos directamente en el recurso AWS::CloudFormation::Init. La clave de los archivos del recurso contiene una propiedad denominada authentication. Puede utilizar la propiedad authentication para asociar información de autenticación definida en un recurso AWS::CloudFormation::Authentication directamente a un archivo.

El siguiente fragmento de código de plantilla de ejemplo incluye tipos de autenticación *básica* y *S3*.

JSON

```
"AWS::CloudFormation::Authentication" : {
    "testBasic" : {
        "type" : "basic",
        "username" : { "Ref" : "UserName" },
        "password" : { "Ref" : "Password" },
        "uris" : [ "http://www.example.com/test" ]
    },
    "testS3" : {
        "type" : "S3",
        "accessKeyId" : { "Ref" : "AccessKeyID" },
        "secretKey" : { "Ref" : "SecretAccessKeyID" },
        "buckets" : [ "aws-s3-bucket1" ]
    }
}
```

Roles de IAM

El siguiente ejemplo muestra cómo utilizar los roles de IAM:

- myRole es un recurso [AWS::IAM::Role](#).
- La instancia Amazon EC2 que ejecuta cfn-init está asociada a myRole a través de un perfil de instancia.
- El ejemplo especifica la autenticación mediante la propiedad buckets, como en la autenticación de Amazon S3. También puede especificar la autenticación por nombre.

JSON

```
"AWS::CloudFormation::Authentication": {
    "rolebased" : {
        "type": "S3",
        "buckets": [ "myBucket" ],
        "roleName": { "Ref": "myRole" }
    }
}
```

```
{
  "AWSTemplateFormatVersion" : "version date",
  "Description" : "JSON string",
  "Metadata" : {
    template metadata
  },
  "Parameters" : {
    set of parameters
  },
  "Mappings" : {
    set of mappings
  },
  "Conditions" : {
    set of conditions
  },
  "Transform" : {
    set of transforms
  },
  "Resources" : {
    set of resources
  },
  "Outputs" : {
    set of outputs
  }
}
```

Parameters optional para personalizar sus plantillas. Los parámetros le permiten introducir valores personalizados a su plantilla cada vez que crea o actualiza una stack.

Definición de un parámetro en una plantilla

En el siguiente ejemplo se declara un parámetro denominado InstanceTypeParameter. Este parámetro le permite especificar el tipo de instancia Amazon EC2 para la pila que se usará al crear o actualizar la pila.

Tenga en cuenta que InstanceTypeParameter tiene un valor predeterminado de t2.micro. Este es el valor que AWS CloudFormation usa para aprovisionar la pila a menos que se proporcione otro valor.

JSON

```
"Parameters" : {
  "InstanceTypeParameter" : {
    "Type" : "String",
    "Default" : "t2.micro",
    "AllowedValues" : ["t2.micro", "m1.small", "m1.large"],
    "Description" : "Enter t2.micro, m1.small, or m1.large. Default is t2.micro."
  }
}
```

Referencia a un parámetro dentro de una plantilla

Puede usar la función intrínseca Ref para hacer referencia a un parámetro, mientras que AWS CloudFormation utiliza el valor del parámetro para aprovisionar la pila. Puede hacer referencia a los parámetros en las secciones Resources y Outputs de la misma plantilla.

En el siguiente ejemplo, la propiedad InstanceType del recurso de instancia EC2 hace referencia al valor del parámetro InstanceTypeParameter:

JSON

```
"Ec2Instance" : {
  "Type" : "AWS::EC2::Instance",
  "Properties" : {
    "InstanceType" : { "Ref" : "InstanceTypeParameter" },
    "ImageId" : "ami-0ff8a91507f77f867"
  }
}
```

```
{  
  "AWSTemplateFormatVersion" : "version date",  
  "Description" : "JSON string",  
  "Metadata" : {  
    template metadata  
  },  
  "Parameters" : {  
    set of parameters  
  },  
  "Mappings" : {  
    set of mappings  
  },  
  "Conditions" : {  
    set of conditions  
  },  
  "Transform" : {  
    set of transforms  
  },  
  "Resources" : {  
    set of resources  
  },  
  "Outputs" : {  
    set of outputs  
  }  
}
```

Mappings opcional hace coincidir una clave con el conjunto correspondiente de valores identificados. Por ejemplo, si desea establecer valores en función de una región, puede crear un mapeo que utiliza el nombre de una región como clave y contiene los valores que desea especificar para cada región específica. Puede utilizar la función intrínseca Fn::FindInMap para recuperar valores en una asignación.

Mapeo básico

El siguiente ejemplo muestra la sección Mappings con una asignación RegionMap, que contiene cinco claves que se asignan a pares nombre-valor que contienen valores de una única cadena. Las claves son nombres de regiones. Cada par nombre-valor es el ID de AMI de la AMI HVM64 en la región representada por la clave.

Los pares nombre-valor tienen un nombre (HVM64 en el ejemplo) y un valor. Al nombrar los valores, puede asignar más de un conjunto de valores a una clave.

JSON

```
"Mappings" : {  
  "RegionMap" : {  
    "us-east-1"      : { "HVM64" : "ami-0ff8a91507f77f867"},  
    "us-west-1"      : { "HVM64" : "ami-0bdb828fd58c52235"},  
    "eu-west-1"      : { "HVM64" : "ami-047bb4163c506cd98"},  
    "ap-southeast-1" : { "HVM64" : "ami-08569b978cc4dfa10"},  
    "ap-northeast-1" : { "HVM64" : "ami-06cd52961ce9f0d85"}  
  }  
}
```

```
{  
  "AWSTemplateFormatVersion" : "version date",  
  "Description" : "JSON string",  
  "Metadata" : {  
    template metadata  
  },  
  "Parameters" : {  
    set of parameters  
  },  
  "Mappings" : {  
    set of mappings  
  },  
  "Conditions" : {  
    set of conditions  
},  
  "Transform" : {  
    set of transforms  
  },  
  "Resources" : {  
    set of resources  
  },  
  "Outputs" : {  
    set of outputs  
  }  
}
```

Conditions contiene declaraciones que definen las circunstancias por las que se crean o configuran entidades. Por ejemplo, puede crear una condición y, a continuación, asociarla a un recurso o salida de manera que AWS CloudFormation solo cree el recurso o salida si la condición es true.

Es posible utilizar condiciones cuando desea reutilizar una plantilla que puede crear recursos en diferentes contextos, como, por ejemplo, un entorno de pruebas frente a un entorno de producción. En su plantilla, puede añadir un parámetro de entrada EnvironmentType, que acepte **prod** o **test** como entradas. Para el entorno de producción, podría incluir instancias Amazon EC2 con determinadas capacidades; sin embargo, para el entorno de prueba sugerimos utilizar capacidades reducidas para ahorrar dinero. Con las condiciones, puede definir qué recursos se crean y cómo se configuran para cada tipo de entorno.

Utilización condiciones

En función de la entidad que desee crear o configurar condicionalmente, debe incluir declaraciones en las siguientes secciones de plantilla:

Sección Parameters

Defina las entradas que desee que sus condiciones evalúen. Las condiciones se evalúan en true o false en función de los valores de estos parámetros de entrada. Si desea que sus condiciones evalúen pseudoparámetros, no es necesario definir los pseudoparámetros de esta sección; AWS CloudFormation predefine los pseudoparámetros.

Sección Conditions

Defina condiciones mediante las funciones de condiciones intrínsecas. Las condiciones determinan cuándo crea AWS CloudFormation los recursos asociados.

Secciones Resources y Outputs

Asocie condiciones con los recursos o salidas que desee crear condicionalmente. AWS CloudFormation crea entidades que se asocian a una condición true y pasa por alto las entidades que están asociadas a una condición false. Utilice la clave Condition y el ID lógico de una condición para asociarlo a un recurso o salida. Para especificar condicionalmente una propiedad, utilice la función Fn::If. Para obtener más información, consulte Funciones de condiciones.

Sintaxis

La sección Conditions consta del nombre de clave Conditions. Cada declaración de condición incluye un ID lógico y funciones intrínsecas que se evalúan cuando crea o actualiza una stack. La siguiente pseudoplantilla muestra la sección Conditions:

JSON

```
"Conditions" : {  
    "Logical ID" : {Intrinsic function}  
}
```

Funciones intrínsecas de condiciones

Puede utilizar las siguientes funciones intrínsecas para definir condiciones:

- Fn::And
- Fn::Equals
- Fn::If
- Fn::Not
- Fn::Or

```
{  
    "AWSTemplateFormatVersion" : "version date",  
    "Description" : "JSON string",  
    "Metadata" : {  
        template metadata  
    },  
    "Parameters" : {  
        set of parameters  
    },  
    "Mappings" : {  
        set of mappings  
    },  
    "Conditions" : {  
        set of conditions  
    },  
    "Transform" : {  
        set of transforms  
    },  
    "Resources" : {  
        set of resources  
    },  
    "Outputs" : {  
        set of outputs  
    }  
}
```

La sección Transform opcional especifica una o varias macros que AWS CloudFormation utiliza para procesar su plantilla. La sección Transform se basa en el lenguaje sencillo y declarativo de AWS CloudFormation con un potente sistema de macros.

AWS CloudFormation también admite las transformaciones AWS::Serverless y AWS::Include, que son macros alojadas por AWS CloudFormation. AWS CloudFormation trata estas transformaciones de la misma manera que cualquier macro que cree en términos de orden de ejecución y ámbito.

- Una transformación de AWS::Serverless especifica la versión de AWS Serverless Application Model (AWS SAM) que debe utilizarse. Este modelo define la sintaxis AWS SAM que puede utilizar y cómo la procesa AWS CloudFormation. (Para obtener más información acerca de las aplicaciones sin servidor y AWS SAM, consulte la sección sobre [implementación de aplicaciones basadas en Lambda](#) en la [AWS Lambda Developer Guide](#)).
- Una transformación AWS::Include funciona con fragmentos de plantillas almacenados independientemente de la plantilla de AWS CloudFormation principal. Puede insertar estos fragmentos de código en su plantilla principal cuando [Creación de un conjunto de cambios](#) o [Actualización de stacks con conjuntos de cambios](#).

Por ejemplo, en el ejemplo de plantilla siguiente, AWS CloudFormation evalúa MyMacro y, a continuación, AWS::Serverless, los cuales pueden procesar el contenido de toda la plantilla porque están incluidos en la sección Transform.



```
// Start of processable content for MyMacro and AWS::Serverless
AWSTemplateFormatVersion: 2010-09-09
Transform: [MyMacro, AWS::Serverless]
Resources:
  WaitCondition:
    Type: AWS::CloudFormation::WaitCondition
  MyBucket:
    Type: 'AWS::S3::Bucket'
    Properties:
      BucketName: MyBucket
      Tags: [{"key": "value"}]
      CorsConfiguration:[]
  MyEc2Instance:
    Type: 'AWS::EC2::Instance'
    Properties:
      ImageID: "ami-123"
// End of processable content for MyMacro and AWS::Serverless
```

```
{  
    "AWSTemplateFormatVersion" : "version date",  
    "Description" : "JSON string",  
    "Metadata" : {  
        template metadata  
    },  
    "Parameters" : {  
        set of parameters  
    },  
    "Mappings" : {  
        set of mappings  
    },  
    "Conditions" : {  
        set of conditions  
    },  
    "Transform" : {  
        set of transforms  
    },  
    "Resources" : {  
        set of resources  
    },  
    "Outputs" : {  
        set of outputs  
    }  
}
```

La sección Resources declara los recursos de AWS que desea incluir

Campos de recursos

- **ID lógico** El ID lógico tiene que ser alfanumérico (A-Za-z0-9) y único dentro de la plantilla. Utilice el nombre lógico para hacer referencia al recurso en otras partes de la plantilla. Además de los IDs lógicos, determinados recursos también tienen un ID físico, que es el nombre asignado real de dicho recurso, como un ID de instancia EC2 o un nombre de bucket de S3. Utilice los ID físicos para identificar recursos fuera de las plantillas de AWS CloudFormation, pero solo después de que se hayan creado los recursos.
- **Tipo de recurso** El tipo de recurso identifica el tipo de recurso que está declarando. Por ejemplo, AWS::EC2::Instance declara una instancia EC2. [Referencia de tipos de recursos y propiedades de AWS](#).

Propiedades de recursos Las propiedades de recursos son opciones adicionales que puede especificar para un recurso. Por ejemplo, para cada instancia EC2, debe especificar un ID de Imagen de máquina de Amazon (AMI) para dicha instancia.

Sintaxis

La sección Resources consta del nombre de clave Resources. La siguiente pseudoplantilla muestra la sección Resources:

JSON

```
"Resources" : {  
    "Logical ID" : {  
        "Type" : "Resource type",  
        "Properties" : {  
            Set of properties  
        }  
    }  
}
```

El siguiente ejemplo muestra una declaración de recursos. Define dos recursos. El recurso MyInstance incluye el recurso MyQueue como parte de su propiedad UserData:

JSON

```
"Resources" : {
    "MyInstance" : {
        "Type" : "AWS::EC2::Instance",
        "Properties" : {
            "UserData" : {
                "Fn::Base64" : {
                    "Fn::Join" : [ "", [ "Queue=", { "Ref" : "MyQueue" } ] ]
                }
            },
            "AvailabilityZone" : "us-east-1a",
            "ImageId" : "ami-0ff8a91507f77f867"
        }
    },
    "MyQueue" : {
        "Type" : "AWS::SQS::Queue",
        "Properties" : {
        }
    }
}
```

```
{  
    "AWSTemplateFormatVersion" : "version date",  
    "Description" : "JSON string",  
    "Metadata" : {  
        template metadata  
    },  
    "Parameters" : {  
        set of parameters  
    },  
    "Mappings" : {  
        set of mappings  
    },  
    "Conditions" : {  
        set of conditions  
    },  
    "Transform" : {  
        set of transforms  
    },  
    "Resources" : {  
        set of resources  
    },  
    "Outputs" : {  
        set of outputs  
    }  
}
```

La sección Outputs opcional declara valores de salida que puede [importar a otras stack](#) (para [crear referencias cruzadas de stacks](#)), devolver en respuesta (para describir las llamadas a la stacks), o [ver en la consola de AWS CloudFormation](#). Por ejemplo, puede declarar la salida para el nombre del bucket de S3 para una stack para que sea más fácil encontrar el bucket.

```
"Outputs" : {  
    "Logical ID" : {  
        "Description" : "Information about the value",  
        "Value" : "Value to return",  
        "Export" : {  
            "Name" : "Value to export"  
        }  
    }  
}
```

Puede utilizar funciones intrínsecas para personalizar el valor Name de una exportación. Los siguientes ejemplos utilizan las funciones Fn::Join.

JSON

```
"Export" : {  
    "Name" : {  
        "Fn::Join" : [ ":", [ { "Ref" : "AWS::StackName" }, "AccountVPC" ] ]  
    }  
}
```



Salida de la pila

En el siguiente ejemplo, la salida denominada BackupLoadBalancerDNSName devuelve el nombre de DNS para el recurso con el ID lógico BackupLoadBalancer solo cuando la condición CreateProdResources es true. (La segunda salida muestra cómo especificar múltiples salidas.)

JSON

```
"Outputs" : {  
    "BackupLoadBalancerDNSName" : {  
        "Description": "The DNSName of the backup load balancer",  
        "Value" : { "Fn::GetAtt" : [ "BackupLoadBalancer", "DNSName" ]},  
        "Condition" : "CreateProdResources"  
    },  
    "InstanceID" : {  
        "Description": "The Instance ID",  
        "Value" : { "Ref" : "EC2Instance" }  
    }  
}
```



Información general de la interfaz AWS CloudFormation Designer

The screenshot illustrates the AWS CloudFormation Designer interface, which allows users to visualize and edit CloudFormation templates.

Top Bar: Includes standard file operations (New, Open, Save, Close) and a toolbar with icons for download, refresh, and other functions.

Left Panel (Resource types): A tree view of available AWS services and their corresponding resource types. Services listed include ApplicationAutoScaling, ApiGateway, AutoScaling, CertificateManager, CloudFormation, CloudFront, CloudTrail, CloudWatch, CodeBuild, CodeCommit, CodeDeploy, CodePipeline, Config, and DataPipeline. The "Resource types" section is highlighted with a yellow border.

Diagram Area: Displays a visual representation of the CloudFormation stack. It includes the following resources and their connections:

- ElasticLoadBalancer:** A load balancer resource.
- LaunchConfiguration:** A launch configuration resource.
- WebServerAutoScalingGroup:** An Auto Scaling group.
- InstanceSecurityGroup:** A security group.
- LoadBalancerSecurityGroup:** A security group.

Connections show the Launch Configuration pointing to the Auto Scaling group, the Auto Scaling group pointing to the Web Server, the Web Server pointing to the Instance Security Group, and the Instance Security Group pointing to the Load Balancer Security Group.

Right Panel (Template Editor): Shows the template content for "template1".

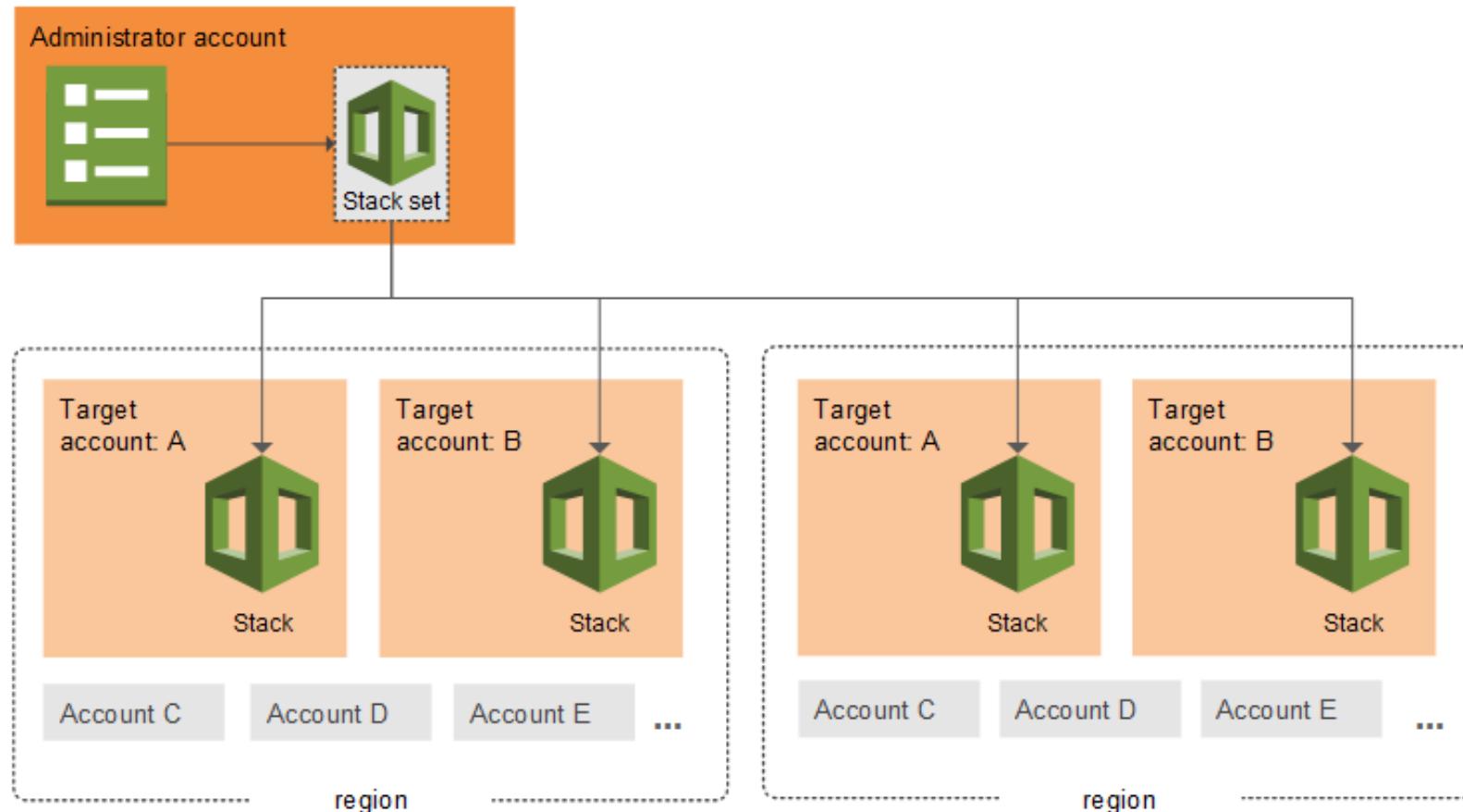
```
template1
1 AWSTemplateFormatVersion: 2010-09-09
2 Description: >-
3   AWS CloudFormation Sample Template VPC_AutoScaling_and_ElasticLoadBalancer:
4   Create a load balanced, Auto Scaled sample website in an existing Virtual
5   Private Cloud (VPC). This example creates an Auto Scaling group behind a load
6   balancer with a simple health check using a basic getting start AMI that has a
7   simple Apache Web Server-based PHP page. The web site is available on port 80,
8   however, the instances can be configured to listen on any port (8888 by
9   default). **WARNING** This template creates one or more Amazon EC2 instances
10  and an Elastic Load Balancer. You will be billed for the AWS resources used if
11  you create a stack from this template.
12  Parameters:
13  VpcId:
14    Type: 'AWS::EC2::VPC::Id'
15    Description: VpcId of your existing Virtual Private Cloud (VPC)
16    ConstraintDescription: must be the VPC Id of an existing Virtual Private Cloud.
17  Subnets:
```

Bottom Right (Messages): Displays a message indicating the template was successfully converted to YAML.

Bottom Navigation: Buttons for "Components" and "Template".

AWS CloudFormation StackSets

AWS CloudFormation StackSets amplía la funcionalidad de las stacks al permitirle crear, actualizar o eliminar stacks de varias cuentas y regiones con una sola operación. Utilizando una cuenta de administrador se define y se administra una plantilla de AWS CloudFormation y la plantilla se utiliza como base para aprovisionar stacks en las cuentas de destino seleccionadas en las regiones especificadas.



Referencia de tipos de recursos y propiedades de AWS

Esta sección contiene información de referencia para todos los tipos de recursos y propiedades de AWS que AWS CloudFormation admite.

Los identificadores del tipo de recursos siempre tienen esta forma:

service-provider::service-name::data-type-name

CloudFormation Referencia de tipos de recursos

Tipos de recurso

- [AWS::CloudFormation::CustomResource](#)
- [AWS::CloudFormation::Macro](#)
- [AWS::CloudFormation::Stack](#)
- [AWS::CloudFormation::WaitCondition](#)
- [AWS::CloudFormation::WaitConditionHandle](#)

AWS::CloudFormation::CustomResource

En una plantilla de CloudFormation, utilizará el tipo de recurso **AWS::CloudFormation::CustomResource** o **Custom::String** para especificar recursos personalizados.

Los recursos personalizados son una forma de escribir la lógica de aprovisionamiento personalizada en una plantilla de CloudFormation y de hacer que CloudFormation la ejecute durante una operación de stack como, por ejemplo, al crear, actualizar o eliminar una stack.

Uso de una función de AWS Lambda en un recurso personalizado

Con los recursos personalizados y las funciones Lambda, puede ejecutar código personalizado en respuesta a eventos de pila (crear, actualizar y eliminar). El siguiente recurso personalizado invoca una función Lambda y envía la propiedad `StackName` como entrada. La función utiliza esta propiedad para obtener salidas de la pila correspondiente.

JSON

```
"MyCustomResource" : {
  "Type" : "Custom::TestLambdaCrossStackRef",
  "Properties" : {
    "ServiceToken": { "Fn::Join": [ "", [ "arn:aws:lambda:", { "Ref": "AWS::Region" }, ":" , { "Ref": "NetworkStackName" } ] ] }
  }
}
```

AWS::CloudFormation::WaitCondition

Puede utilizar una condición de espera para situaciones como las siguientes:

- Para coordinar la creación de recursos de la stack con las acciones de configuración que son externas a la creación de la stack
- Para realizar un seguimiento del estado de un proceso de configuración

WaitCondition que espera el número deseado de instancias en un grupo de servidores web

JSON

```
"WebServerGroup" : {  
    "Type" : "AWS::AutoScaling::AutoScalingGroup",  
    "Properties" : {  
        "AvailabilityZones" : { "Fn::GetAZs" : "" },  
        "LaunchConfigurationName" : { "Ref" : "LaunchConfig" },  
        "MinSize" : "1",  
        "MaxSize" : "5",  
        "DesiredCapacity" : { "Ref" : "WebServerCapacity" },  
        "LoadBalancerNames" : [ { "Ref" : "ElasticLoadBalancer" } ]  
    },  
},  
  
"WaitHandle" : {  
    "Type" : "AWS::CloudFormation::WaitConditionHandle"  
},  
  
"WaitCondition" : {  
    "Type" : "AWS::CloudFormation::WaitCondition",  
    "DependsOn" : "WebServerGroup",  
    "Properties" : {  
        "Handle" : { "Ref" : "WaitHandle" },  
        "Timeout" : "300",  
        "Count" : { "Ref" : "WebServerCapacity" }  
    },  
}
```

AWS::CloudFormation:: WaitConditionHandle

El tipo AWS::CloudFormation::WaitConditionHandle no tiene propiedades. Cuando hace referencia al recurso WaitConditionHandle mediante la función Ref, AWS CloudFormation devuelve una URL prefijada. Puede transferir esta URL a aplicaciones o scripts que se están ejecutando en sus instancias Amazon EC2 para enviar señales a esa URL. Un recurso AWS::CloudFormation::WaitCondition asociado comprueba la presencia del número necesario de señales de éxito o una señal de error en la URL.

importante

Siempre que añada un recurso WaitCondition durante una actualización de la stack o actualice un recurso con una condición de espera, debe asociar la condición de espera con un nuevo recurso WaitConditionHandle. No reutilice un identificador de condición de espera antiguo que ya se haya definido en la plantilla. Si reutiliza un identificador de condición de espera, la condición de espera podría evaluar señales antiguas de un comando anterior de creación o actualización de la stack.

nota

Las actualizaciones no son compatibles con este recurso.

JSON

```
{  
  "Type" : "AWS::CloudFormation::WaitConditionHandle",  
  "Properties" : {  
  }  
}
```

Referencia de atributos personalizados

Esta sección detalla los atributos que puede añadir a un recurso para controlar relaciones y comportamientos adicionales.

Temas

- [Atributo CreationPolicy](#)
- [Atributo DeletionPolicy](#)
- [Atributo DependsOn](#)
- [Atributo Metadata](#)
- [Atributo UpdatePolicy](#)
- [Atributo UpdateReplacePolicy](#)

Atributo CreationPolicy

Asocie el atributo `CreationPolicy` con un recurso para impedir que alcance el estado de creación completada hasta que AWS CloudFormation reciba un número específico de señales o se supere el período de tiempo de espera. Para señalar un recurso, puede utilizar el script auxiliar [cfn-signal](#) o la API [SignalResource](#). AWS CloudFormation publica señales válidas a los eventos de la pila para que pueda realizar un seguimiento del número de señales enviadas.

La política de creación se invoca solo cuando AWS CloudFormation crea el recurso asociado.

Sintaxis

JSON

```
"CreationPolicy" : {  
    "AutoScalingCreationPolicy" : {  
        "MinSuccessfulInstancesPercent" : Integer  
    },  
    "ResourceSignal" : {  
        "Count" : Integer,  
        "Timeout" : String  
    }  
}
```

Atributo DeletionPolicy

Con el atributo `DeletionPolicy` se define que acción se toma con los recursos cuando se eliminan. Opciones:

- backup de un recurso (snapshot y luego delete)
- Conservar (**Retain**)
- Eliminar (**Delete**)

Cada recurso tiene una `DeletionPolicy`. Si un recurso no tiene ningún atributo `DeletionPolicy`, AWS CloudFormation elimina el recurso de forma predeterminada.

JSON

```
{  
    "AWSTemplateFormatVersion" : "2010-09-09",  
    "Resources" : {  
        "myS3Bucket" : {  
            "Type" : "AWS::S3::Bucket",  
            "DeletionPolicy" : "Retain"  
        }  
    }  
}
```

Atributo DependsOn

Con el atributo DependsOn puede especificar que a la creación de un recurso específico le sigue otra: Cuando añade un atributo DependsOn a un recurso, ese recurso se crea solo tras la creación del recurso especificado en el atributo DependsOn.

importante

Las stacks dependientes también tienen dependencias implícitas. Por ejemplo, si las propiedades de un recurso A utilizan una !Ref al recurso B, se aplican las siguientes reglas:

- El recurso B se crea antes que el recurso A.
- El recurso A se elimina antes que el recurso B.

Puede utilizar el atributo DependsOn con cualquier recurso

- Determine cuándo entra en vigor una condición de espera
- Declare dependencias para recursos que deben crearse o eliminarse en un orden específico.
- Invalidación de paralelismo predeterminado al crear, actualizar o eliminar recursos. AWS CloudFormation crea, actualiza y elimina los recursos en paralelo en la medida de lo posible. Puede utilizar DependsOn para especificar explícitamente dependencias, lo cual invalida el paralelismo predeterminado e indica a CloudFormation que opere en esos recursos en un orden especificado.

JSON

```
"service": {  
    "Type": "AWS::ECS::Service",  
    "DependsOn": [  
        "ECSAutoScalingGroup"  
    ],  
    "Properties" : {  
        "Cluster": {  
            "Ref": "ECSCluster"  
        },  
        "DesiredCount": "1",  
        "LoadBalancers": [  
            {  
                "ContainerName": "simple-app",  
                "ContainerPort": "80",  
                "LoadBalancerName" : {  
                    "Ref" : "EcsElasticLoadBalancer"  
                }  
            }  
        ],  
        "Role" : {  
            "Ref": "ECSServiceRole"  
        },  
        "TaskDefinition" : {  
            "Ref": "taskdefinition"  
        }  
    }  
}
```

Atributo UpdatePolicy

Utilice el atributo `UpdatePolicy` para especificar cómo gestiona AWS CloudFormation las actualizaciones en los recursos [AWS::AutoScaling::AutoScalingGroup](#), [AWS::Lambda::Alias](#) o [AWS::ElastiCache::ReplicationGroup](#).

- Para los recursos `AWS::AutoScaling::AutoScalingGroup`, AWS CloudFormation; invoca una de las tres políticas de actualización en función del tipo de cambio que realice o de si una acción programada está asociada al grupo de Auto Scaling.
 - Las políticas `AutoScalingReplacingUpdate` y `AutoScalingRollingUpdate` se aplican *únicamente* al realizar una o varias de las siguientes acciones:
 - Cambiar [AWS::AutoScaling::LaunchConfiguration](#) del grupo de Auto Scaling.
 - Cambiar la propiedad `VPCZoneIdentifier` del grupo de Auto Scaling.
 - Cambiar la propiedad `LaunchTemplate` del grupo de Auto Scaling.
 - Actualizar un grupo de Auto Scaling que contenga instancias que no coincidan con la `LaunchConfiguration` actual.
 - Si se especifican las dos políticas, `AutoScalingReplacingUpdate` y `AutoScalingRollingUpdate`, al establecer la propiedad `WillReplace` en `true`, `AutoScalingReplacingUpdate` tiene preferencia.
 - La política `AutoScalingScheduledAction` se aplica al actualizar una pila que incluya un grupo de Auto Scaling con una acción programada asociada.
- Para los recursos `AWS::Lambda::Alias`, AWS CloudFormation realiza una implementación de `CodeDeploy` cuando la versión cambia en el alias. Para obtener más información, consulte [Política CodeDeployLambdaAliasUpdate](#).
- Para los recursos `AWS::ElastiCache::ReplicationGroup`, AWS CloudFormation puede modificar las particiones del grupo de replicación mediante la incorporación o eliminación de particiones, en lugar de sustituir todo el recurso. Para obtener más información, consulte [Política UseOnlineResharding](#).

Política AutoScalingReplacingUpdate

Para especificar la forma en que AWS CloudFormation administra las actualizaciones de sustitución para un grupo de Auto Scaling, utilice la política AutoScalingReplacingUpdate. Esta política le permite especificar si AWS CloudFormation sustituye a un grupo de Auto Scaling por uno nuevo o sustituye solo las instancias del grupo de Auto Scaling.

importante

Antes de realizar una actualización, asegúrese de que dispone de suficiente capacidad de Amazon EC2 tanto para los grupos de Auto Scaling antiguos como los nuevos.

Sintaxis

JSON

```
"UpdatePolicy" : {  
    "AutoScalingReplacingUpdate" : {  
        "WillReplace" : Boolean  
    }  
}
```



Política AutoScalingRollingUpdate

Para especificar la forma en que AWS CloudFormation administra las actualizaciones acumulativas para un grupo de Auto Scaling, utilice la política AutoScalingRollingUpdate. Las actualizaciones acumulativas le permiten especificar si AWS CloudFormation actualiza las instancias que se encuentran en un grupo de Auto Scaling en lotes o todas a la vez.

importante

Durante una actualización acumulativa, algunos procesos de Auto Scaling podrían realizar cambios en el grupo de Auto Scaling antes de que AWS CloudFormation termine la actualización acumulativa. Estos cambios pueden hacer que falle la actualización acumulativa. Para evitar que Auto Scaling ejecute procesos durante una actualización acumulativa, utilice la propiedad SuspendProcesses. Para obtener más información, consulte [¿Cuáles son algunas de las prácticas recomendadas para realizar actualizaciones acumulativas del grupo de Auto Scaling?](#)

Tenga en cuenta que, durante las operaciones de restauración de actualización de la pila, CloudFormation utiliza la configuración UpdatePolicy especificada en la plantilla antes de la operación de actualización de la pila actual. Por ejemplo, supongamos que ha actualizado MaxBatchSize en su UpdatePolicy de plantilla de pila de 1 a 10. A continuación, realiza una actualización de la pila, la actualización falla y CloudFormation inicia una operación de restauración de la actualización. En este caso, CloudFormation utilizará 1 como el tamaño máximo del lote, en lugar de 10. Por este motivo, recomendamos que haga los cambios a la configuración UpdatePolicy en una actualización de la pila independiente y antes de cualquier actualización del recurso `AWS::AutoScaling::AutoScalingGroup` que podría desencadenar actualizaciones continuas.

Política AutoScalingScheduledAction

Para especificar la forma en que AWS CloudFormation administra las actualizaciones de las propiedades `MinSize`, `MaxSize` y `DesiredCapacity` cuando el recurso `AWS::AutoScaling::AutoScalingGroup` tiene una acción programada asociada, utilice la política `AutoScalingScheduledAction`.

Con las acciones programadas, las propiedades de tamaño de un grupo de Auto Scaling pueden cambiar en cualquier momento. Al actualizar una pila con un grupo de Auto Scaling y una acción programada, AWS CloudFormation siempre establece los valores de la propiedad del tamaño del grupo de Auto Scaling en los valores que se definen en el recurso `AWS::AutoScaling::AutoScalingGroup` de la plantilla, incluso si se está en vigor una acción programada.

Si no desea que AWS CloudFormation cambie cualquiera de los valores de la propiedad de tamaño del grupo cuando hay en vigor una acción programada, utilice la política de actualización `AutoScalingScheduledAction` y configure `IgnoreUnmodifiedGroupSizeProperties` en `true` para evitar que AWS CloudFormation cambie las propiedades `MinSize`, `MaxSize` o `DesiredCapacity` a menos que haya modificado estos valores en la plantilla.

Política UseOnlineResharding

Para modificar las particiones del grupo de replicación añadiendo o quitando particiones, en lugar de sustituir todo el recurso [AWS::ElastiCache::ReplicationGroup](#), utilice la política de actualización de UseOnlineResharding.

Si UseOnlineResharding está establecido en true, puede actualizar las propiedades NumNodeGroups y NodeGroupConfiguration del recurso AWS::ElastiCache::ReplicationGroup y CloudFormation actualizará esas propiedades sin interrupción. Cuando UseOnlineResharding se establece en false o no se especifica, la actualización de las propiedades NumNodeGroups y NodeGroupConfiguration tendrá como resultado que CloudFormation sustituya todo el recurso AWS::ElastiCache::ReplicationGroup.

La política de actualización UseOnlineResharding no tiene propiedades.

Atributo UpdateReplacePolicy

Filter View: All ▾

Utilice el atributo `UpdateReplacePolicy` para conservar o, en ciertos casos, hacer una copia de seguridad de la instancia física existente de un recurso cuando se sustituye durante una operación de actualización de la pila.

Cuando inicie la actualización de una pila, AWS CloudFormation actualiza los recursos en función de las diferencias entre lo que envía y la plantilla y los parámetros actuales de la pila. Si actualiza una propiedad de recurso que requiere la [sustitución](#) del recurso, AWS CloudFormation recrea el recurso durante la actualización. La recreación del recurso genera un nuevo ID físico. AWS CloudFormation crea primero el recurso de sustitución y, a continuación, cambia las referencias de otros recursos dependientes para que apunten al recurso de sustitución. De forma predeterminada, AWS CloudFormation elimina seguidamente el recurso antiguo. Mediante el uso del atributo `UpdateReplacePolicy`, puede especificar que AWS CloudFormation conserve o, en ciertos casos, cree una instantánea del recurso antiguo.

Para los recursos que admiten instantáneas, como por ejemplo `AWS::EC2::Volume`, puede especificar `Snapshot` para que AWS CloudFormation cree una instancia del recurso antiguo.

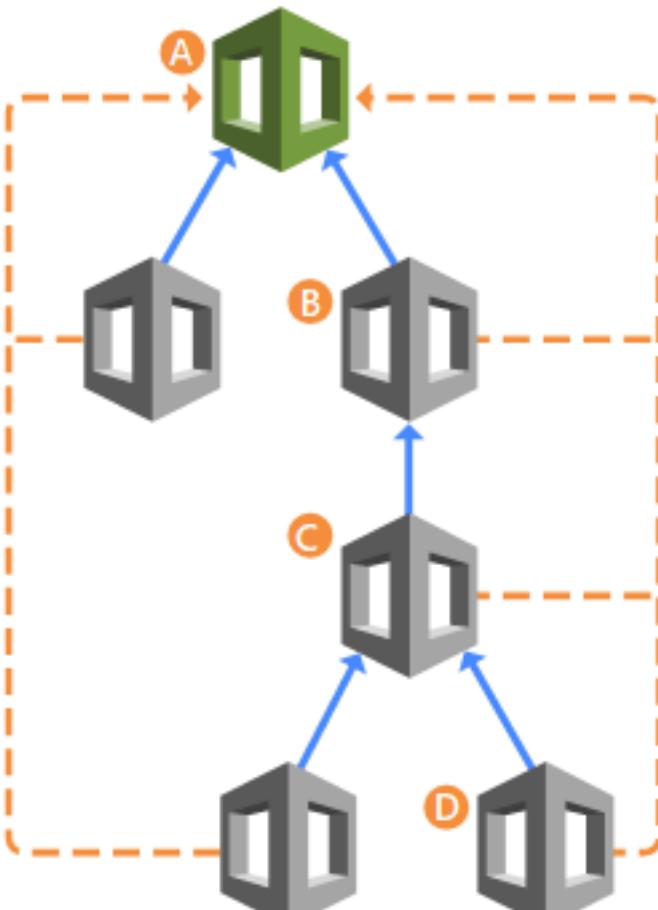
Referencia de función intrínseca

AWS CloudFormation ofrece varias funciones integradas que le ayudan a administrar sus pilas. Utilice las funciones intrínsecas en las plantillas para asignar valores a las propiedades que no están disponibles hasta que el tiempo de ejecución.

Topics

- [Fn::Base64](#)
- [Fn::Cidr](#)
- [Condition Functions](#)
- [Fn::FindInMap](#)
- [Fn::GetAtt](#)
- [Fn::GetAZs](#)
- [Fn::ImportValue](#)
- [Fn::Join](#)
- [Fn::Select](#)
- [Fn::Split](#)
- [Fn::Sub](#)
- [Fn::Transform](#)
- [Ref](#)

Working with Nested Stacks



Nested stacks are created as part of other stacks using the [AWS::CloudFormation::Stack](#) resource.

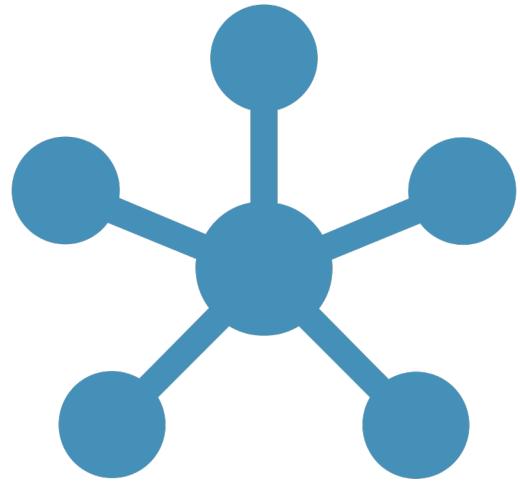
For example, imagine you have a configuration for a load balancer that you use for most of your stacks. Instead of copying and pasting the same configurations into each template, you can create a dedicated template for the load balancer. Then, you can reference this template from other templates.

Nested stacks can themselves contain other stacks, resulting in a stack hierarchy, as shown in the diagram below.

The *root stack* is the stack at the highest level that contains all other stacks. Additionally, each nested stack has a *primary stack* directly beneath it. For the first level of nested stacks, the root stack is also the primary stack. In the diagram above, for example:

- Stack A is the root of all other nested stacks in the hierarchy.
- For stack B, stack A is both its primary stack and the root stack.
- For stack D, stack C is its primary stack, just as stack C is for stack B.
- For stack C, stack B is its primary stack.





LAB