

Capítulo 3

Servicios de computación en AWS

1. Introducción

Detrás de cualquier proyecto en la nube, la cuestión de la computación –es decir, la forma en que se ejecuta el código y se asignan los recursos informáticos– sigue siendo fundamental.

Tanto si es desarrollador, arquitecto o simplemente tiene curiosidad por las tecnologías de AWS, la elección de la solución de computación adecuada influye en el rendimiento, la fiabilidad e incluso en el coste de su aplicación.

En este capítulo, exploramos los principales servicios de computación que ofrece AWS: EC2, ECS, EKS y Lambda. Cada una de estas opciones tiene su propia lógica de uso: por un lado, máquinas virtuales configurables (EC2); por otro, un entorno "sin servidor" en el que no hay que ocuparse de la gestión de los servidores (Lambda). Entre estos dos extremos se encuentran los contenedores gestionados a través de ECS u orquestados mediante Kubernetes (EKS), que ofrecen un equilibrio interesante para la implementación de microservicios portables y escalables.

Comenzaremos con una visión general y, a continuación, profundizaremos en Amazon EC2 y sus buenas prácticas. A continuación, analizaremos el escalado mediante Autoscaling y los equilibradores de carga. Veremos cómo elegir la solución de contenedores adecuada entre ECS, EKS y Fargate y descubriremos el universo "serverless" de AWS Lambda. Por último, una comparativa detallada le ayudará a seleccionar el servicio que mejor se adapte a tus proyectos.

Tanto si está descubriendo AWS como si desea profundizar en sus conocimientos, este capítulo tiene como objetivo ofrecerte una visión general y consejos prácticos.

2. Presentación de las opciones de computación (EC2, ECS, EKS, Lambda)

2.1 Una breve historia de la computación en AWS

Lanzado al público en 2006, Amazon EC2 (*Elastic Compute Cloud*) se convirtió rápidamente en el "escaparate" de la nube, permitiendo alquilar potencia de cálculo por horas, que hoy en día se factura por segundos para la mayoría de las instancias de Linux (con un mínimo de 60 segundos), mientras que Windows y algunas distribuciones comerciales siguen facturándose por horas. Esta flexibilidad supuso una ruptura con la compra y la gestión de servidores físicos tradicionales.

Las necesidades han ido evolucionando progresivamente hacia una mayor abstracción, especialmente a través de los contenedores Docker. AWS respondió introduciendo Amazon ECS (*Elastic Container Service*) y, posteriormente, un servicio gestionado de Kubernetes, Amazon EKS (*Elastic Kubernetes Service*). Más tarde, apareció el servicio AWS Lambda para permitir la ejecución de código sin necesidad de gestionar servidores, según un modelo "serverless".

Estos cuatro pilares de la computación se pueden resumir de la siguiente manera:

- **EC2** proporciona máquinas virtuales (VM) que ofrecen un control total.
- **ECS** implementa contenedores Docker gestionados por AWS.
- **EKS** orquesta contenedores de Kubernetes, pero en AWS.
- **Lambda** ejecuta funciones sin necesidad de gestionar infraestructuras.

La elección depende principalmente del grado de control que se desee sobre el sistema operativo, del tipo de aplicación, de la cultura empresarial (Kubernetes o no) y del esfuerzo que suponga gestionar los servidores y el sistema operativo.

2.2 Los diferentes modelos de ejecución

En el modelo IaaS (EC2), el usuario es responsable de todo lo que se encuentra por encima de la virtualización. AWS proporciona los servidores físicos, la capa de hipervisor y la puesta a disposición de la instancia, mientras que usted debe instalar y mantener el sistema operativo, las dependencias y su código. Este grado de libertad tiene como contrapartida la necesidad de aplicar parches y supervisar sus máquinas, así como de diseñar tus estrategias de escalabilidad.

En el modelo CaaS (ECS o EKS), la aplicación se empaqueta en un contenedor Docker. El orquestador (ECS o Kubernetes a través de EKS) planifica y gestiona estos contenedores. El aislamiento es similar al de un minisistema operativo y varios contenedores se pueden ejecutar en la misma máquina o estar dispersos. La ventaja es poder centrarse más en el código y la implementación, al tiempo que se acepta que sigue habiendo una capa de infraestructura que hay que administrar (o delegar, a través de Fargate).

Por último, el modelo FaaS (Lambda) corresponde a un entorno "serverless" en el que solo se escriben funciones que se invocan bajo demanda. Solo se paga por el tiempo de ejecución, sin tener que gestionar instancias de forma continua. Es una solución ideal para tareas puntuales o basadas en eventos, con una facturación granular por milisegundos, pero sujeta a ciertos límites (duración, memoria, acceso al disco).

¿Cuándo es mejor optar por una u otra?

EC2 es una buena opción para migraciones *lift & shift* (trasladar una aplicación tal cual, sin reescribirla) o cuando es imprescindible tener control sobre el sistema operativo. ECS/EKS son adecuados para quienes ya utilizan Docker o Kubernetes y buscan una implementación flexible. Lambda resulta especialmente adecuado para eventos, cargas de trabajo de corta duración y contextos en los que se desea reducir el mantenimiento del sistema.

3. Amazon EC2: funcionamiento y buenas prácticas

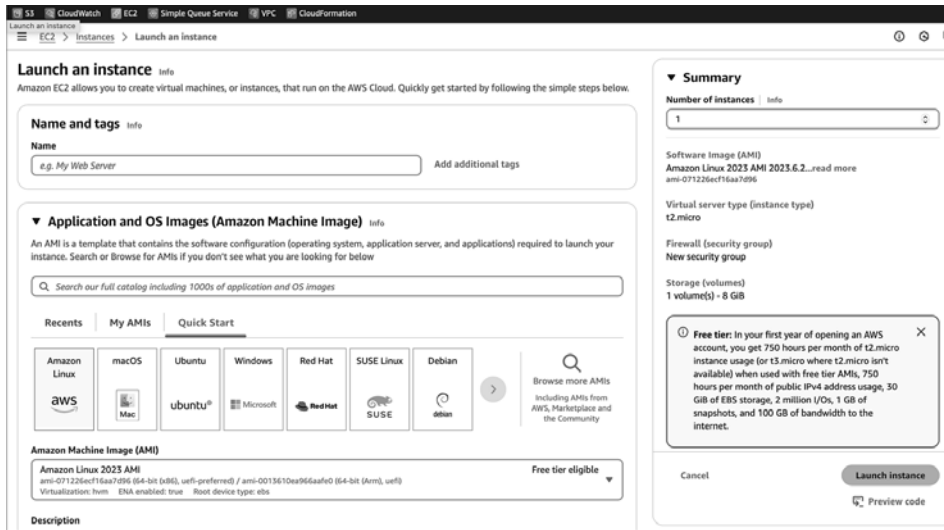
EC2 en pocas palabras

Amazon EC2 (*Elastic Compute Cloud*) permite implementar y configurar máquinas virtuales (instancias) en la nube de AWS. Cada instancia corresponde a unos recursos seleccionados (vCPU, RAM, almacenamiento, ancho de banda de red), que se pueden adaptar entre numerosas categorías (T, M, C, R, G, P, etc.). La facturación se realiza en función del uso y es posible cambiar el tamaño, detener o iniciar las instancias a voluntad.

Históricamente, EC2 es uno de los servicios estrella de AWS, ya que ofrece la posibilidad de hacer de todo en una instancia, a cambio de asumir la carga relacionada con el sistema operativo (parches, seguridad, escalabilidad).

3.1 Funcionamiento

Para crear una instancia, se selecciona una AMI (*Amazon Machine Image*) que define el sistema operativo y los paquetes iniciales. A continuación, se elige el tipo de instancia (t2.micro, m5.large, etc.), los parámetros de red (VPC, subred, IP), así como las opciones de seguridad (grupo de seguridad, claves SSH, contraseñas). Tras el lanzamiento, se puede iniciar sesión y configurar libremente la instancia.



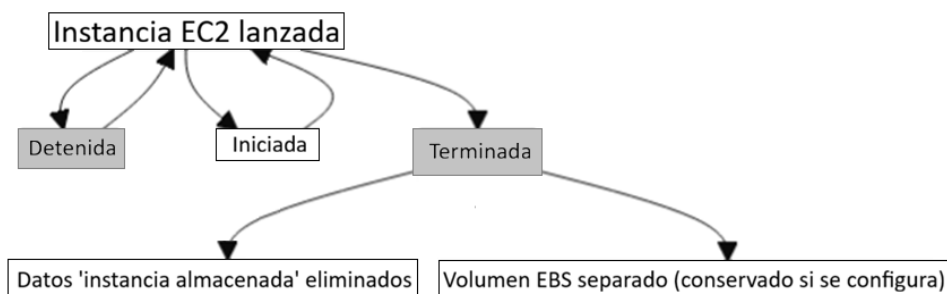
Pantalla de creación de una instancia EC2 en AWS

Volúmenes EBS

Por defecto, se suele utilizar un volumen EBS (*Elastic Block Store*) como disco principal. Se trata de un almacenamiento persistente y cifrable cuyo tamaño y rendimiento varían según los tipos: gp3 (uso general), io2 (uso intensivo) o st1/sc1 (HDD más económico).

Ciclo de vida de las instancias

Una instancia se puede **iniciar**, **detener**, **reiniciar** o **terminar**. Cuando está detenida, ya no se paga por la instancia, pero se sigue pagando por el volumen EBS. En caso de terminación, la instancia se elimina y los datos de tipo "instance store" desaparecen, mientras que un volumen EBS desacoplado se puede conservar.



Ciclo de vida de las instancias EC2

Buenas prácticas

Se recomienda añadir etiquetas a sus instancias (que indiquen, por ejemplo, el equipo responsable o el entorno) para simplificar la gestión. Proteja sus instancias mediante un rol de IAM (*Instance Profile*) con permisos mínimos y aplique regularmente los parches de seguridad. Supervise también las métricas con CloudWatch, programe copias de seguridad (instantáneas de EBS) y no dude en cambiar de tipo de instancia para ajustar sus recursos.

Ejemplo de uso

Para un pequeño servidor web, se puede iniciar una instancia **t2.micro**, instalar en ella un servidor web (Apache, Nginx), asignarle una IP pública y abrir el tráfico HTTP/HTTPS a través de un grupo de seguridad. Este tipo de instancia tiene un coste reducido si permanece encendida de forma continua. Para una configuración más robusta, se pueden crear varias instancias en un auto scaling group distribuidas en varias zonas de disponibilidad, con conmutación automática en caso de fallo.

3.2 Autoescalado y equilibrio de carga: gestionar el aumento de la carga

El principio del "scale-out" y el "scale-in"

Una de las grandes ventajas de la nube es la capacidad de ajustar dinámicamente los recursos en función de la carga. Se pueden añadir instancias (*scale-out*) cuando se produce un pico y, posteriormente, retirarlas (*scale-in*) para reducir los costes cuando la actividad disminuye. Este mecanismo se basa en métricas como el uso de la CPU o el número de solicitudes HTTP.

Las políticas de escalado (simple, por pasos o seguimiento de objetivos) permiten definir los umbrales y las reacciones deseadas. Con el "*target tracking*", se busca mantener un porcentaje objetivo (por ejemplo, un 50 % de CPU); el grupo de escalado automático (ASG) crea o elimina instancias para mantenerse cerca de ese objetivo.

Elastic Load Balancing (ELB)

Un load balancer distribuye el tráfico entre varias instancias y supervisa su estado. AWS ofrece varios tipos de equilibradores de carga: el Application Load Balancer (capa 7), el Network Load Balancer (capa 4) y el Gateway Load Balancer (para seguridad y filtrado). Combinar un ELB con un ASG permite dirigir el tráfico únicamente a las instancias que funcionan correctamente e iniciar o detener instancias automáticamente según sea necesario.

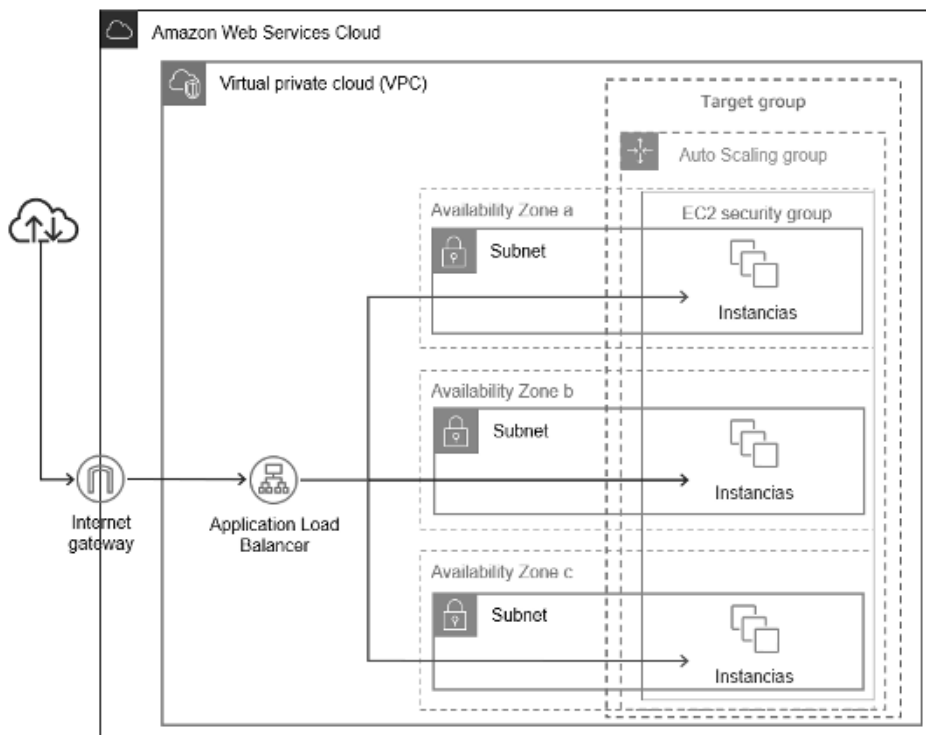
Estrategias de autoescalado

El escalado puede ser horizontal (añadir instancias) o vertical (cambiar el tipo de instancia por una configuración más potente). El escalado horizontal es más flexible y más tolerante a los fallos, mientras que el escalado vertical está limitado por los tamaños disponibles y a menudo requiere un reinicio.

Dentro de un ASG, se define un número mínimo (para garantizar la alta disponibilidad), un número deseado para la carga actual y un número máximo para evitar costes desmesurados. También se configura un periodo de "cooldown" para evitar oscilaciones demasiado frecuentes.

Ejemplo concreto

En el caso de una web de comercio electrónico cuya carga varía considerablemente, puede resultar contraproducente dimensionar un clúster para el pico de fin de año. Al optar por un ASG, se mantiene una base de dos instancias en todo momento (una en cada zona de disponibilidad) y, si la CPU supera el 70% durante unos minutos, el ASG pone en marcha una instancia adicional (hasta un límite máximo establecido). Cuando la carga disminuye, el ASG retira instancias para reducir los costes.



Infraestructura de AWS que aprovecha el autoescalado

Este esquema ilustra el funcionamiento combinado de un load balancer y un auto scaling group en un entorno de AWS.