¿Qué es AWS Lambda?

Lambda es un servicio informático que permite ejecutar código sin aprovisionar ni administrar servidores. Lambda ejecuta el código en una infraestructura informática de alta disponibilidad y realiza todas las tareas de administración de los recursos informáticos, incluido el mantenimiento del servidor y del sistema operativo, el aprovisionamiento de capacidad y el escalado automático, así como la monitorización del código y las funciones de registro

Organice su código en Funciones de Lambda. Lambda ejecuta su función solo cuando es necesario y escala de manera automática, **desde unas pocas solicitudes por día hasta miles por segundo**. **Solo pagará por el tiempo informático que consuma, no se aplican cargos cuando el código no se está ejecutando**.

**Puede invocar sus funciones de Lambda utilizando la API de Lambda, o Lambda puede ejecutar las funciones en respuesta a eventos de otros servicios de AWS**. Por ejemplo, puede utilizar Lambda para:

* Crear desencadenadores de procesamiento de datos para servicios de AWS como Amazon Simple Storage Service (Amazon S3) y Amazon DynamoDB.
* Procesar datos de streaming almacenados en Amazon Kinesis.
* Crear su propio backend que funcione a escala de AWS, rendimiento y seguridad.

## Función

Una función es un recurso que puede invocar para ejecutar el código en Lambda. Una función tiene código para procesar los eventos que pasa a la función o que otros AWS servicios envían a la función.

## Desencadenador

Un desencadenador es un recurso o configuración que invoca una función de Lambda. Los desencadenadores incluyen los servicios de AWS que puede configurar para invocar una función y mapeos de fuentes de eventos.

**Tiempo de ejecución**

El *tiempo de ejecución* proporciona un entorno específico del lenguaje que se ejecuta en el entorno de ejecución. El tiempo de ejecución transmite eventos de invocación, información de contexto y respuestas entre Lambda y la función. Puede usar tiempos de ejecución que Lambda proporcione, o crear los suyos propios. Si empaqueta el código como un archivo de archivo .zip, debe configurar su función para utilizar un tiempo de ejecución que coincida con su lenguaje de programación.

## Capa

Una capa de Lambda es un archivo .zip que puede contener código u otros datos adicionales. Una capa puede contener bibliotecas, un tiempo de ejecución personalizado, datos o archivos de configuración.

Las capas proporcionan una forma conveniente de empaquetar bibliotecas y otras dependencias que puede usar con las funciones de Lambda. El uso de capas reduce el tamaño de los archivos de implementación cargados y acelera la implementación de su código. Las capas también promueven el uso compartido de código y la separación de responsabilidades para que pueda iterar más rápido al escribir la lógica empresarial.

## Simultaneidad

La simultaneidad es el número de solicitudes que la función atiende en un momento dado. Cuando se invoca la función, Lambda aprovisiona una instancia para procesar el evento. Cuando el código de la función termina de ejecutarse, puede encargarse de otra solicitud. Si la función se invoca de nuevo mientras se sigue procesando una solicitud, se aprovisiona otra instancia, lo que aumenta la simultaneidad de la función.

La simultaneidad está sujeta a cuotas en el nivel de región de AWS. Puede configurar funciones individuales para limitar su simultaneidad o para permitirles alcanzar un nivel específico de simultaneidad.

Características de Lambda

**Escalado**

Lambda administra la infraestructura que ejecuta el código, escalándola automáticamente en respuesta a las solicitudes entrantes. Cuando la función se invoca a mayor velocidad de la que una sola instancia de su función puede procesar eventos, Lambda amplía la capacidad ejecutando instancias adicionales. Cuando el tráfico disminuye, las instancias inactivas se bloquean o detienen. Solo paga por el tiempo que la función inicia o procesa eventos.

**Controles de simultaneidad**

Utilice la configuración de simultaneidad para asegurarse de que sus aplicaciones de producción tengan alta disponibilidad y capacidad de respuesta. Para evitar que una función utilice demasiada simultaneidad y para reservar una parte de la simultaneidad disponible de su cuenta para una función, utilice la *simultaneidad reservada*. La simultaneidad reservada divide el grupo de simultaneidad disponible en subconjuntos. Una función con simultaneidad reservada solo usa simultaneidad desde su grupo dedicado.


        Simultaneidad reservada asignada a dos funciones.
      

**URL de funciones**

Lambda ofrece compatibilidad integrada para puntos de conexión HTTP(S) a través de *URL de funciones*. Con las URL de funciones, puede asignar un punto de conexión HTTP dedicado a la función de Lambda. Cuando la URL de función esté configurada, puede utilizarla para invocar la función a través de un navegador web, curl, Postman o cualquier cliente HTTP.

Puede agregar una URL de función a una función existente o crear una nueva función con una URL de función.

## Invocación asincrónica

Al invocar una función, puede optar por invocarla de forma síncrona o asíncrona. Con [invocación síncrona](https://docs.aws.amazon.com/es_es/lambda/latest/dg/invocation-sync.html), espere la función para procesar el evento y devolver una respuesta. Con invocación asíncrona, Lambda pone en cola el evento para su procesamiento y devuelve una respuesta inmediatamente.

Para las invocaciones asíncronas, Lambda maneja reintentos si la función devuelve un error o está limitada. Para personalizar este comportamiento, puede configurar las opciones de gestión de errores en una función, versión o alias. También puede configurar Lambda para que envíe eventos que hayan fallado en el procesamiento a una cola de mensajes fallidos o para que envíe un registro de cualquier invocación a un [destino](https://docs.aws.amazon.com/es_es/lambda/latest/dg/gettingstarted-features.html#gettingstarted-features-destinations).

¿Qué es Amazon API Gateway?

Amazon API Gateway es un servicio de AWS para la creación, la publicación, el mantenimiento, el monitoreo y la protección de las API REST, HTTP y de WebSocket a cualquier escala. Los desarrolladores de API pueden crear API que obtengan acceso a AWS o a otros servicios web, así como los datos almacenados en la [nube de AWS](http://aws.amazon.com/what-is-cloud-computing/). Como desarrollador de API de API Gateway, puede crear API para su uso en sus propias aplicaciones de cliente. También puede ofrecer sus API a otros desarrolladores de aplicaciones externos.

API Gateway crea API RESTful que:

* Se basan en HTTP.
* Habilitan la comunicación entre cliente y servidor sin estado.
* Implementan métodos HTTP estándar como, por ejemplo, GET, POST, PUT, PATCH y DELETE.

API Gateway crea API de WebSocket que:

* Cumplen el protocolo WebSocket, que permite la comunicación entre el cliente y el servidor de dúplex completo con estado.
* Dirigen mensajes entrantes en función del contenido de los mensajes.

Características de API Gateway

* Compatibilidad con las API con estado ([WebSocket](https://docs.aws.amazon.com/es_es/apigateway/latest/developerguide/apigateway-websocket-api.html)) y las API ([HTTP](https://docs.aws.amazon.com/es_es/apigateway/latest/developerguide/http-api.html) y [REST](https://docs.aws.amazon.com/es_es/apigateway/latest/developerguide/apigateway-rest-api.html)).
* Mecanismos de [autenticación](https://docs.aws.amazon.com/es_es/apigateway/latest/developerguide/apigateway-control-access-to-api.html) eficaces y flexibles, como políticas de AWS Identity and Access Management, funciones de autorizador de Lambda y grupos de usuarios de Amazon Cognito.
* [Portal para desarrolladores](https://docs.aws.amazon.com/es_es/apigateway/latest/developerguide/apigateway-developer-portal.html) para la publicación de sus API.
* [Implementaciones de la versión Canary](https://docs.aws.amazon.com/es_es/apigateway/latest/developerguide/canary-release.html) para el despliegue de cambios de forma segura.
* Registro de [CloudTrail](https://docs.aws.amazon.com/es_es/apigateway/latest/developerguide/cloudtrail.html) y monitoreo del uso y de los cambios en las API.
* Registro de acceso y registro de ejecución de CloudWatch, que incluye la posibilidad de establecer alarmas. Para obtener más información, consulte [Monitoreo de la ejecución de la API de REST con métricas de Amazon CloudWatch](https://docs.aws.amazon.com/es_es/apigateway/latest/developerguide/monitoring-cloudwatch.html) y [Monitoreo de la ejecución de la API de WebSocket con métricas de CloudWatch](https://docs.aws.amazon.com/es_es/apigateway/latest/developerguide/apigateway-websocket-api-logging.html).

¿Qué es Amazon Elastic Container Service?

Amazon Elastic Container Service (Amazon ECS) es un servicio de administración de contenedores muy escalable y rápido. Puede utilizarlo para ejecutar, detener y administrar contenedores en un clúster. Con Amazon ECS, los contenedores se definen en una definición de tareas que utiliza para ejecutar tareas individuales o tareas dentro de un servicio. En este contexto, un servicio es una configuración que puede usar para ejecutar y mantener un número determinado de tareas simultáneamente en un clúster. Las tareas y los servicios se pueden ejecutar en una infraestructura sin servidor administrada por AWS Fargate.

Amazon ECS ofrece las siguientes características:

* **Una opción sin servidor con AWS Fargate.** Con AWS Fargate, no necesita administrar servidores, gestionar la planificación de la capacidad ni aislar las cargas de trabajo de contenedores por seguridad. Fargate se encarga de los aspectos de administración de la infraestructura de su carga de trabajo por usted. Puede programar la ubicación de los contenedores en el clúster en función de las necesidades de recursos, las políticas de aislamiento y los requisitos de disponibilidad.
* **Integración con AWS Identity and Access Management (IAM).** Puede asignar permisos granulares para cada uno de sus contenedores. Esto permite un alto nivel de aislamiento al crear sus aplicaciones. En otras palabras, puede lanzar sus contenedores con los niveles de seguridad y cumplimiento que espera de AWS.
* **Orquestación de contenedores administrados de AWS.** Como servicio totalmente administrado, Amazon ECS incluye configuración y prácticas recomendadas operativas integradas de AWS. Esto también significa que no necesita administrar el plano de control, nodos o complementos. Está integrado tanto con Alexa Web Information Service como con herramientas de terceros, como Amazon Elastic Container Registry y Docker. Esta integración facilita a los equipos centrarse en crear las aplicaciones, no en el entorno.
* **Implementación e integración continuas (CI/CD).** Este es un proceso común para arquitecturas de microservicios que se basan en contenedores Docker. Puede crear una canalización de CI/CD que realice las siguientes acciones:
  + Monitoriza los cambios en un repositorio de código fuente
  + Crea una nueva imagen de Docker desde dicho origen
  + Inserta la imagen en un repositorio de imágenes como Amazon ECR o Docker Hub
  + Actualiza los servicios de Amazon ECS para utilizar la nueva imagen en la aplicación
* **Compatibilidad con la detección de servicios.** Este es un componente clave de los sistemas más distribuidos y las arquitecturas orientadas a servicios. Con la detección de servicios, los componentes de microservicios se detectan automáticamente a medida que se crean y se terminan en una infraestructura dada.
* **Compatibilidad para el envío de información de registro de instancias de contenedor a CloudWatch Logs**. Después de enviar esta información a Amazon CloudWatch, puede ver los registros desde sus instancias de contenedor en una ubicación cómoda. Esto evita que los registros de contenedor ocupen espacio en disco en las instancias de contenedor.

## Tipos de lanzamiento

Hay dos modelos que puede utilizar para ejecutar los contenedores:

* Tipo de lanzamiento de Fargate: esta es una opción de servicio de pago por uso sin servidor. Puede ejecutar contenedores sin necesidad de administrar su infraestructura.
* Tipo de lanzamiento de EC2: configure e implemente instancias de EC2 en el clúster para ejecutar los contenedores.

El tipo de lanzamiento de Fargate es adecuado para las siguientes cargas de trabajo:

* Grandes cargas de trabajo que deben optimizarse para reducir la sobrecarga
* Pequeñas cargas de trabajo que tienen ráfagas ocasionales
* Cargas de trabajo pequeñas
* Cargas de trabajo en lotes

El tipo de lanzamiento de EC2 es adecuado para las siguientes cargas de trabajo:

* Cargas de trabajo que requieren un uso constante de memoria y núcleo de CPU
* Grandes cargas de trabajo que deben optimizarse por precio
* Sus aplicaciones necesitan acceder al almacenamiento persistente
* Debe administrar directamente su infraestructura

# **¿Qué es AWS Fargate?**

La tecnología AWS Fargate se puede utilizar en Amazon ECS para ejecutar [contenedores](http://aws.amazon.com/what-are-containers) sin tener que administrar servidores ni clústeres de instancias de Amazon EC2. Con Fargate, ya no tendrá que aprovisionar, configurar ni escalar clústeres de máquinas virtuales para ejecutar los contenedores. De esta manera, se elimina la necesidad de elegir tipos de servidores, decidir cuándo escalar los clústeres u optimizar conjuntos de clústeres.

Al ejecutar las tareas y los servicios de Amazon ECS con el tipo de lanzamiento de Fargate o un proveedor de capacidad de Fargate, la aplicación se empaqueta en contenedores, se especifican los requisitos de sistema operativo, CPU y de memoria, se definen las políticas de IAM y las redes, y se lanza la aplicación. Cada tarea de Fargate tiene su propio límite de aislamiento y no comparte el kernel subyacente, los recursos de CPU, los recursos de memoria ni la interfaz de red elástica con otra tarea.