AWS Guida decisionale

AWS Fargate o AWS Lambda?

AWS Fargate o AWS Lambda?: AWS Guida decisionale

Table of Contents

AWS Fargate oppure AWS Lambda?

Comprendi le differenze e scegli quella più adatta a te

Scopo	Per scoprire se AWS Fargate o AWS Lambda soddisfare le tue esigenze di un servizio di elaborazione serverless.
Ultimo aggiornamento	15 novembre 2024
Servizi coperti	• AWS Fargate • AWS Lambda

Introduzione

Prima di iniziare a valutare se scegliere AWS Lambda o utilizzare AWS Fargate come servizio di elaborazione serverless, probabilmente avete preso in considerazione la gamma più ampia di servizi di AWS elaborazione (illustrata nella guida alla scelta di un servizio di AWS elaborazione) e l'avete ristretta a queste due scelte perché forniscono:

- Sovraccarico operativo ridotto: sia Lambda che Fargate riducono la gestione dei server, riducendo la necessità di patch, manutenzione e pianificazione della capacità.
- Pay-per-use prezzi: paghi solo per le risorse di elaborazione effettivamente utilizzate, con una potenziale riduzione dei costi per carichi di lavoro variabili.
- Implementazione più rapida: in genere offre tempi di implementazione più rapidi rispetto al provisioning e alla configurazione delle istanze. EC2
- Alta disponibilità integrata: entrambi i servizi gestiscono automaticamente la ridondanza dell'infrastruttura.
- Conformità semplificata: la superficie di attacco ridotta e le funzionalità di sicurezza integrate possono facilitare gli sforzi di conformità.
- Concentrarsi sul codice: gli sviluppatori possono concentrarsi maggiormente sulla scrittura del codice applicativo piuttosto che sulla gestione dell'infrastruttura.

Sebbene Lambda e Fargate siano entrambe opzioni serverless, esistono differenze significative tra loro:

AWS Fargateè un motore di elaborazione serverless per container, utilizzato principalmente con Amazon ECS. Gestisce automaticamente l'infrastruttura, consentendoti di concentrarti sulla distribuzione e sulla scalabilità di applicazioni containerizzate. Fargate è ideale per applicazioni a esecuzione prolungata, microservizi o elaborazione in batch, in cui è necessario un controllo granulare sull'allocazione delle risorse (CPU, memoria) ed evitare di gestire i server sottostanti.

AWS Lambdaè un servizio di elaborazione serverless che esegue automaticamente il codice in risposta agli eventi e gestisce le risorse di elaborazione sottostanti. È ideale per applicazioni basate sugli eventi, come l'elaborazione di file caricati su Amazon S3, la risposta a richieste HTTP o l'esecuzione di attività pianificate. Lambda è ideale anche per applicazioni di elaborazione di flussi ed elaborazione dati grazie alla sua capacità di scalare automaticamente in risposta agli eventi e gestire elevati volumi di dati in tempo reale. Lambda può elaborare flussi di dati da fonti come Amazon Kinesis o Amazon DynamoDB, permettendo trasformazioni, filtri e analisi dei dati efficienti e senza server senza dover gestire l'infrastruttura. Lambda è progettata per attività di breve durata (fino a 15 minuti) e viene fatturata in base al numero di richieste e al tempo di esecuzione, il che la rende conveniente per carichi di lavoro sporadici.

Se il tuo progetto prevede attività basate su eventi, di breve durata o carichi di lavoro imprevedibili, potrebbe essere la soluzione migliore. AWS Lambda Se hai bisogno di eseguire applicazioni containerizzate con esigenze di risorse specifiche (o hai bisogno di processi persistenti), sarebbe più appropriato. AWS Fargate

La tabella seguente fornisce una panoramica più dettagliata di alcune delle differenze tra questi servizi per iniziare.

Funzionalità	AWS Fargate	AWS Lambda
Modello di esecuzione	Elaborazione senza server basata su container	Funzioni serverless basate sugli eventi
Lingue supportate	Qualsiasi linguaggio che può essere eseguito in un contenitore	Linguaggi supportati: Node.js, Python, Java, C#, Go, Ruby e. PowerShell Puoi anche creare un runtime personalizzato per implementare una AWS

		Lambda funzione nella lingua che preferisci.
Caso d'uso	Applicazioni containerizzate a lunga durata	Attività di breve durata e basate su eventi
Dimensionamento	Ridimensionamento automatico in base al numero di attività desiderato	Ridimensionamento automatico per richiesta
Avvio a freddo	Da 35 secondi a 2 minuti	Da 100 ms a 2 secondi
Limite di tempo di esecuzione	Nessun limite rigido	15 minuti al massimo
Allocazione della memoria	Fino a 120 GiB	Fino a 10 GiB
Allocazione della CPU	Fino a 16 vCPU	Proporzionale alla memoria, fino a 6 vCPU
Rete	Funziona in VPC, può essere utilizzato ENIs	Può essere eseguito in un VPC AWS gestito o collegato a un VPC gestito dal cliente utilizzando Hyperplane AWS
Gestione dello stato	I contenitori di Fargate possono mantenere lo stato tra le richieste finché il contenitore è in esecuzione, rendendo possibile la gestione delle sessioni, la cache dei dati o il mantenimento dello stato in memoria senza bisogno di storage esterno. L'archiviazione esterna è consigliata per i dati critici.	Stateless by design (lo stato deve essere gestito esternamente, ad esempio Amazon S3, Amazon DynamoDB, Amazon EFS)
Supporto container	Supporta contenitori	Supporto limitato ai container (tramite implementazioni di immagini dei container)

Orchestrazione	Integrato con Amazon ECS	Non è richiesta alcuna orchestrazione
Modello tariffario	Fatturazione al secondo per vCPU e memoria utilizzate	Per chiamata e durata (GB al secondo)
Limiti di concorrenza	In base alla capacità del cluster	1000 esecuzioni simultanee per impostazione predefinita (possono essere aumentate)
Chiamata basata sugli eventi	Richiede una configurazione aggiuntiva	Supporto nativo per varie fonti di AWS eventi
Mitigazione dell'avviamento a freddo	Il caricamento lento delle immagini con Seekable OCI può velocizzare l'avvio delle attività di Fargate	Concorrenza fornita disponibi le
Limite di dimensione del pacchetto	Nessun limite specifico (dimensione del contenito re limitata dallo storage temporaneo configurato, massimo 200 GiB)	250 MB decompressi, livelli inclusi, 10 GB per le implementazioni di immagini in container

Differenze tra Fargate e Lambda

Esplora le differenze tra Fargate e Lambda in diverse aree chiave.

Languages supported

Fargate: AWS Fargate è un servizio di orchestrazione di container, il che significa che supporta qualsiasi linguaggio di programmazione o ambiente di runtime che può essere impacchettato in un contenitore Docker. Questa flessibilità consente agli sviluppatori di utilizzare praticamente qualsiasi linguaggio, framework o libreria adatto alle loro esigenze applicative. Che stiate utilizzando Python, Java, Node.js, Go, .NET, Ruby, PHP o anche linguaggi e ambienti personalizzati, Fargate può eseguirli purché siano incapsulati in un contenitore. Questo ampio supporto linguistico rende Fargate ideale per l'esecuzione di diverse applicazioni, inclusi sistemi legacy, microservizi multilingue e moderne applicazioni native per il cloud.

Lambda: AWS Lambda offre supporto nativo per un set di lingue più limitato rispetto a Fargate, progettato specificamente per le funzioni basate sugli eventi. A partire da ora, Lambda supporta ufficialmente le seguenti lingue:

- Node.js
- Python
- Java
- Go
- Ruby
- C#
- PowerShell

Lambda supporta anche i runtime personalizzati, che consentono di utilizzare il proprio linguaggio o ambiente di runtime, ma ciò richiede una maggiore configurazione e gestione rispetto all'utilizzo delle opzioni supportate nativamente. Se scegli di distribuire la tua funzione Lambda da un'immagine del contenitore, puoi scrivere la funzione in Rust utilizzando AWS un'immagine di base solo per il sistema operativo e includendo il client di runtime Rust nell'immagine. Se utilizzi un linguaggio che non dispone di un client di interfaccia di runtime AWS fornito, devi crearne uno personalizzato.

Event-driven invocation

Lambda è intrinsecamente progettata per l'elaborazione basata sugli eventi. Le funzioni Lambda vengono attivate in risposta a una serie di eventi, tra cui modifiche dei dati, azioni dell'utente o attività pianificate. Si integra perfettamente con molte applicazioni Servizi AWS, come Amazon S3 (ad esempio, richiamando una funzione quando viene caricato un file), DynamoDB (ad esempio, attivazione degli aggiornamenti dei dati) e API Gateway (ad esempio, gestione delle richieste HTTP). L'architettura Lambda event-driven è ideale per le applicazioni che devono rispondere immediatamente agli eventi senza richiedere risorse di elaborazione persistenti.

Fargate non è nativamente basato sugli eventi, ma con una logica standard aggiuntiva, può integrarsi con sorgenti di eventi come Amazon SQS e Kinesis. Sebbene Lambda gestisca la maggior parte di questa logica di integrazione per te, dovrai implementare tu stesso questa integrazione utilizzando APIs per questi servizi.

Runtime/use cases

Fargate è progettato per eseguire applicazioni containerizzate, fornendo un ambiente di runtime flessibile in cui è possibile definire le impostazioni di CPU, memoria e rete per i contenitori. Poiché Fargate opera su un modello basato su container, supporta processi a esecuzione prolungata, servizi persistenti e applicazioni con requisiti di runtime specifici. I container di Fargate possono funzionare all'infinito, poiché non vi sono limiti rigidi al tempo di esecuzione, il che li rende ideali per le applicazioni che devono essere attive e funzionanti continuamente.

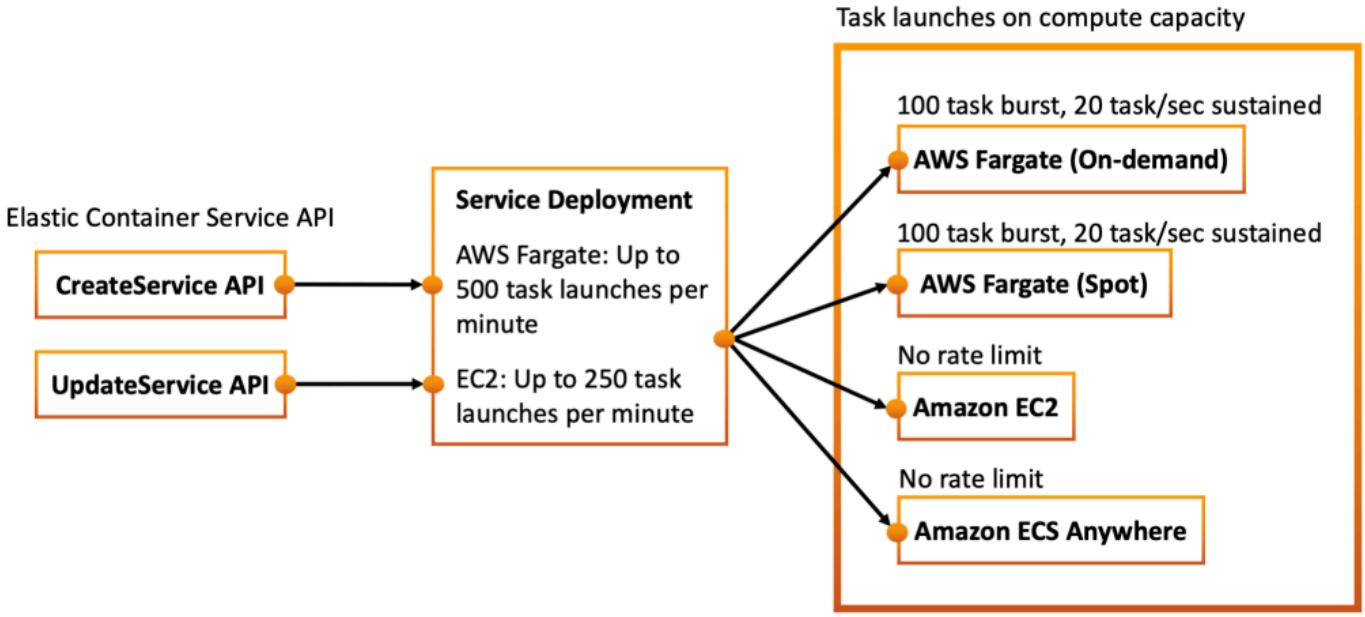
Lambda, d'altra parte, è ottimizzata per attività di breve durata e basate su eventi. Le funzioni Lambda vengono eseguite in un ambiente senza stato in cui il tempo massimo di esecuzione è limitato a 15 minuti. Ciò rende Lambda ideale per scenari come l'elaborazione di file, lo streaming di dati in tempo reale e la gestione delle richieste HTTP, in cui le attività sono brevi e non richiedono processi di lunga durata.

In Lambda, l'ambiente di runtime è più astratto e il controllo sull'infrastruttura sottostante è minore. La natura stateless di Lambda significa che ogni chiamata di funzione è indipendente e qualsiasi stato o dato che deve persistere tra le chiamate deve essere gestito esternamente (ad esempio, nei database o nei servizi di storage).

Scaling

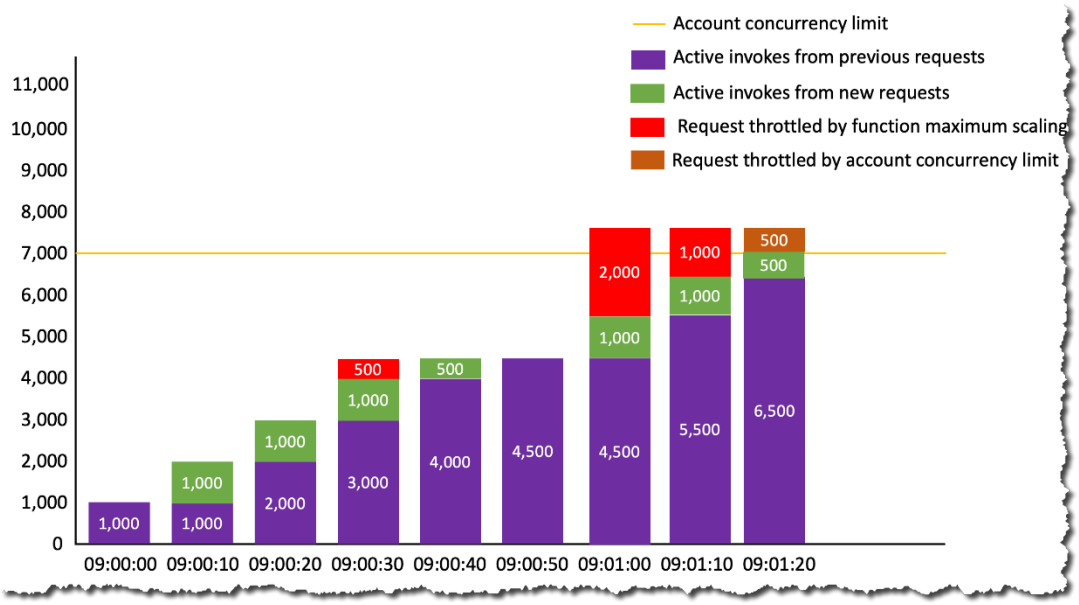
Fargate scala regolando il numero di container in esecuzione in base allo stato desiderato definito nel servizio di orchestrazione dei container (Amazon ECS). Questo ridimensionamento può essere eseguito manualmente o automaticamente tramite Amazon EC2 Auto Scaling. Questo post sul blog offre ulteriori dettagli su come.

In Fargate, ogni container viene eseguito nel proprio ambiente isolato e la scalabilità prevede il lancio di contenitori aggiuntivi o il loro arresto in base al carico. Lo strumento di pianificazione dei servizi Amazon ECS è in grado di avviare fino a 500 attività in meno di un minuto per servizio per il Web e altri servizi a lunga durata.



Per Lambda, la concorrenza è il numero di richieste in corso che la AWS Lambda funzione gestisce contemporaneamente. Ciò differisce dalla concorrenza in Fargate, in cui ogni attività Fargate può gestire richieste simultanee purché siano disponibili risorse di calcolo e di rete. Per ogni richiesta simultanea, Lambda fornisce un'istanza separata del tuo ambiente di esecuzione. Man mano che le funzioni ricevono più richieste, Lambda gestisce automaticamente il dimensionamento del numero di ambienti di esecuzione fino al raggiungimento del limite di simultaneità dell'account. Per impostazione predefinita, Lambda fornisce al tuo account un limite totale di 1.000 esecuzioni simultanee in tutte le funzioni di un account e Regione AWS, se necessario, puoi richiedere un aumento della quota.

Per ogni funzione Lambda in una regione, la velocità di scalabilità simultanea è di 1.000 istanze di esecuzione ogni 10 secondi, fino alla concorrenza massima dell'account. Come spiegato in questo blog, se il numero di richieste in un periodo di 10 secondi supera 1.000, le richieste aggiuntive verranno limitate. Il grafico seguente mostra come funziona la scalabilità Lambda presupponendo una concorrenza di 7000 account.



Cold start and cold-start mitigation

Lambda può subire avviamenti a freddo, che si verificano quando una funzione viene richiamata dopo essere rimasta inattiva per qualche tempo. Durante un avvio a freddo, il servizio Lambda deve inizializzare un nuovo ambiente di esecuzione, incluso il caricamento del runtime, delle dipendenze e del codice della funzione. Questo processo può introdurre latenza, in particolare per i linguaggi con tempi di inizializzazione più lunghi (ad esempio, Java o C#). Gli avvii a freddo possono influire sulle prestazioni delle applicazioni, in particolare quelle che richiedono risposte a bassa latenza.

Per mitigare gli avviamenti a freddo in Lambda, è possibile utilizzare diverse strategie:

- Riduci al minimo le dimensioni della funzione: la riduzione delle dimensioni del pacchetto di funzioni e delle relative dipendenze può ridurre il tempo necessario per l'inizializzazione.
- Aumento dell'allocazione di memoria: allocazioni di memoria più elevate aumentano la capacità della CPU, riducendo potenzialmente i tempi di inizializzazione.
- Mantieni calde le funzioni: richiamare periodicamente le funzioni Lambda (ad esempio, CloudWatch utilizzando Events) può mantenerle attive e ridurre la probabilità di avviamenti a freddo.
- Lambda SnapStart: utilizza le funzioni Lambda SnapStart for Java per ridurre i tempi di avvio.
- Concorrenza fornita: questa funzionalità mantiene un determinato numero di istanze di funzione attive e pronte a soddisfare le richieste, riducendo la latenza di avvio a freddo. Tuttavia, aumenta i costi in quanto si pagano le istanze fornite anche se non gestiscono attivamente le richieste.

Fargate non è generalmente influenzata dagli avviamenti a freddo allo stesso modo della Lambda. Il tempo necessario per avviare un'attività Fargate è direttamente correlato al tempo necessario per estrarre le immagini del contenitore definite nell'operazione dal registro delle immagini. Fargate supporta anche il caricamento lento delle immagini dei container che sono state indicizzate con Seekable OCI (SOCI). Il caricamento lento delle immagini dei container con SOCI riduce il tempo necessario per avviare le attività di Amazon ECS su Fargate. Fargate gestisce container che rimangono attivi per tutto il tempo necessario, il che significa che sono sempre pronti a gestire le richieste. Tuttavia, se è necessario avviare nuovi contenitori in risposta a eventi di scalabilità, potrebbe verificarsi un certo ritardo durante l'inizializzazione dei contenitori, ma in genere questo è meno significativo rispetto agli avvii a freddo Lambda.

Memory and CPU options

Fargate offre un controllo granulare sulle risorse di memoria e CPU per le applicazioni containerizzate. Quando si avvia un'attività in Fargate, è possibile specificare i requisiti esatti di CPU e memoria in base alle esigenze dell'applicazione. Le allocazioni di CPU e memoria sono indipendenti e consentono di scegliere le combinazioni più adatte al carico di lavoro. Ad esempio, è possibile selezionare valori della CPU compresi tra 0,25 v CPUs e 16 v CPUs e la memoria da 0,5 GB a 120 GB per contenitore, a seconda della configurazione.

Questa flessibilità è ideale per l'esecuzione di applicazioni che richiedono caratteristiche prestazionali specifiche, come database a uso intensivo di memoria o attività di calcolo legate alla CPU. Fargate consente di ottimizzare l'allocazione delle risorse per bilanciare costi e prestazioni in modo efficace.

In Lambda, memoria e CPU sono collegate, con la CPU allocata automaticamente in proporzione alla quantità di memoria selezionata. Puoi scegliere allocazioni di memoria tra 128 MB e 10 GB, con incrementi di 1 MB. La CPU è scalabile in base alla memoria, fino a 6 vCPU, il che significa che impostazioni di memoria più elevate comportano una maggiore potenza della CPU, ma non hai il controllo diretto sull'allocazione della CPU stessa.

Questo modello è progettato per essere semplice e consente agli sviluppatori di regolare rapidamente le impostazioni della memoria senza dover gestire le configurazioni della CPU. Tuttavia, potrebbe essere meno flessibile per i carichi di lavoro che richiedono un equilibrio specifico tra CPU e risorse di memoria. Il modello Lambda è adatto per attività in cui si desidera una scalabilità semplice in base alle esigenze di memoria, ma potrebbe non essere ottimale per applicazioni con richieste di risorse complesse o altamente specifiche.

Networking

Quando distribuisci le attività in Fargate, queste vengono eseguite in un Amazon VPC (Amazon Virtual Private Cloud), offrendoti il pieno controllo sull'ambiente di rete. Ciò include la configurazione di gruppi di sicurezza, elenchi di controllo degli accessi alla rete (ACLs) e tabelle di routing. Ogni attività Fargate dispone di una propria interfaccia di rete, con un indirizzo IP privato dedicato e, se necessario, può essere assegnato un indirizzo IP pubblico.

Fargate supporta funzionalità di rete avanzate come il bilanciamento del carico (tramite AWS ELB), il peering VPC e l'accesso diretto ad altri elementi all'interno del VPC. Servizi AWS È inoltre possibile utilizzare Servizi AWS Supported AWS PrivateLink per una connettività privata e sicura, senza dover attraversare Internet.

Per impostazione predefinita, le funzioni Lambda vengono eseguite in un ambiente di rete gestito senza controllo diretto sulle interfacce di rete o sugli indirizzi IP. Tuttavia, Lambda può essere collegata a un VPC gestito dal cliente AWS utilizzando Hyperplane, consentendoti di controllare l'accesso alle risorse all'interno del tuo VPC.

Quando le funzioni Lambda sono collegate a un VPC gestito dal cliente, ereditano i gruppi di sicurezza e le configurazioni di sottorete del VPC, consentendo loro di interagire in modo sicuro con altri (come i database RDS) all'interno dello stesso VPC. Servizi AWS

Il servizio Lambda utilizza una piattaforma di virtualizzazione delle funzioni di rete per fornire funzionalità NAT dal VPC Lambda al cliente. VPCs Ciò configura le interfacce di rete elastiche richieste (ENIs) nel punto in cui le funzioni Lambda vengono create o aggiornate. Consente ENIs inoltre la condivisione del tuo account tra più ambienti di esecuzione, il che consente a Lambda di utilizzare in modo più efficiente una risorsa di rete limitata quando le funzioni sono scalabili.

Poiché ENIs si tratta di una risorsa esauribile ed è previsto un limite minimo di 250 ENIs per regione, è necessario monitorare l'utilizzo dell'elastic network interface se si configurano le funzioni Lambda per l'accesso al VPC. Le funzioni Lambda nella stessa AZ e nello stesso gruppo di sicurezza possono essere condivise. ENIs In genere, se si aumentano i limiti di simultaneità in Lambda, è necessario valutare se è necessario aumentare l'interfaccia di rete elastica. Se viene raggiunto il limite, le invocazioni delle funzioni Lambda abilitate per VPC vengono limitate.

Pricing model

I prezzi di Fargate si basano sulle risorse allocate ai container, in particolare sulla vCPU e sulla memoria selezionate per ciascuna attività. La fatturazione viene calcolata al secondo, con un costo minimo di un minuto, per la CPU e la memoria utilizzate dai container. I costi sono direttamente legati alle risorse utilizzate dall'applicazione, il che significa che si paga per ciò

che viene fornito, indipendentemente dal fatto che l'applicazione stia elaborando attivamente le richieste. Fargate è ideale per carichi di lavoro prevedibili in cui sono necessarie configurazioni di risorse specifiche e può ottimizzare i costi regolando le risorse allocate. Inoltre, potrebbero essere applicati costi aggiuntivi per i servizi correlati, come il trasferimento di dati, l'archiviazione e il networking (ad esempio, VPC, ELB).

Lambda ha una struttura dei prezzi diversa basata sugli eventi e. pay-per-execution I costi vengono addebitati in base al numero di richieste ricevute dalle funzioni e alla durata di ciascuna esecuzione, misurata in millisecondi. Lambda tiene conto anche della quantità di memoria allocata alla funzione, con una scalabilità dei costi in base alla memoria utilizzata e al tempo di esecuzione. Il modello di prezzo include un piano gratuito, che offre 1 milione di richieste gratuite e 400.000 GB di tempo di elaborazione al mese, il che rende Lambda particolarmente conveniente per carichi di lavoro sporadici e a basso volume.

Il modello di prezzo Lambda è ideale per le applicazioni con schemi di traffico imprevedibili o intensi, in quanto si paga solo per le chiamate effettive delle funzioni e il tempo di esecuzione, senza la necessità di fornire o pagare la capacità inattiva.

Utilizzo

Ora che hai letto i criteri per scegliere tra AWS Fargate e AWS Lambda, puoi selezionare il servizio che soddisfa le tue esigenze e utilizzare le seguenti informazioni per iniziare a utilizzare ciascuno di essi.

AWS Fargate

- Scopri come creare un'attività Amazon ECS Linux per il tipo di lancio Fargate

  Inizia a usare Amazon ECS usando il tipo AWS Fargate di avvio Fargate per le tue attività su Linux.

  Esplora la guida
- Scopri come creare un'attività Amazon ECS Windows per il tipo di lancio Fargate

  Inizia a usare Amazon ECS usando il tipo AWS Fargate di avvio Fargate per le tue attività Windows.

  Esplora la guida
- Guida introduttiva a Fargate e Amazon EKS

Questa guida descrive come iniziare a utilizzare i tuoi pod AWS Fargate con il tuo cluster Amazon EKS.

Esplora la guida

- AWS Fargate prezzi

  Usa questa guida per capire in che modo le configurazioni di vCPU, memoria, storage e sistema operativo influiscono sui prezzi. AWS Fargate

  Esplora la guida

- AWS Fargate domande frequenti

  Ottieni risposte alle domande più comuni sulle AWS Fargate funzionalità e sulle migliori pratiche di implementazione.

  Esplora la guida

AWS Lambda

- Crea un'app di elaborazione file senza server

  Una step-by-step procedura dettagliata per configurare e utilizzare Amazon SNS. Tratta argomenti come la creazione di un argomento, la sottoscrizione degli endpoint a un argomento, la pubblicazione di messaggi e la configurazione delle autorizzazioni di accesso.

  Esplora la guida

- Guida per gli sviluppatori serverless

  Questa guida ti aiuta a sviluppare una migliore comprensione concettuale dello sviluppo di applicazioni serverless e di come le diverse applicazioni Servizi AWS si combinino per creare modelli di applicazione che costituiscono il nucleo delle tue applicazioni cloud.

  Esplora la guida

- Terra senza server

  Questo sito raccoglie le informazioni più recenti, i blog, i video, il codice e le risorse di apprendimento per AWS Serverless. Impara a utilizzare e creare app scalabili automaticamente su un'architettura serverless a basso costo e completamente gestita.

Esplora il sito

- AWS Lambda prezzi

  Utilizza questa guida per stimare le spese e ottimizzare i costi in base all'utilizzo e alla configurazione delle funzioni. Include un calcolatore dei prezzi per calcolare i costi tuoi AWS Lambda e dell'architettura in un'unica stima.

  Esplora la guida

- AWS Lambda domande frequenti

  Ottieni risposte alle domande più comuni sulle AWS Lambda funzionalità e sulle migliori pratiche di implementazione.

  Esplora la guida

Cronologia dei documenti per AWS Fargate o AWS Lambda?

La tabella seguente descrive le modifiche importanti a questa guida decisionale. Per ricevere notifiche sugli aggiornamenti di questa guida, puoi iscriverti a un feed RSS.

Modifica	Descrizione	Data
Versione iniziale	Versione iniziale della guida decisionale.	15 novembre 2024

Le traduzioni sono generate tramite traduzione automatica. In caso di conflitto tra il contenuto di una traduzione e la versione originale in Inglese, quest'ultima prevarrà.