Come costruire un URL shortener completamente Serverless utilizzando solo una manciata di servizi AWS

9 Giugno 2022 - 5 min. read

Amazon CloudFront | Amazon Route 53 | AWS Lambda@Edge | Edge Computing | Serverless

Al giorno d'oggi, sfruttare a pieno le potenzialità del cloud è essenziale per realizzare progetti efficienti e di successo.

Uno dei paradigmi che più ha rivoluzionato, e sta tutt'ora rivoluzionando il settore è sicuramente **serverless**.

Sfruttando i principi del serverless computing è possibile costruire applicazioni in cui la componente infrastrutturale scala in modo automatico seguendo perfettamente la curva delle richieste. Una buona infrastruttura serverless permette di **abbattere i costi infrastrutturali e di manutenzione** riducendo l'area di competenza del cliente. L'intera infrastruttura, infatti, risulta essere in **alta disponibilità, scalabile ed elastica by design**.

In questo articolo, vogliamo presentare una soluzione completamente serverless per creare un URL shortener. Nei paragrafi seguenti ripercorremo uno use case completo partendo dai requisiti e arrivando fino alla progettazione di una soluzione semplice basata su servizi AWS completamente gestiti e serverless.

I requirements

Lo shortener dovrà fornire un'API per la creazione dei link accorciati.

L'API dovrà rispondere a una richiesta GET così composta:
https://example.com/api/v1/action/shorten?url=[URI]

La risposta dovrà essere in text/plain ed il corpo deve contenere l'URL abbreviato
https://example.com/f/7427

Quando un utente visiterà l'URL abbreviato, il sistema dovrà reindirizzare il visitatore all'URL completo.

Non è necessaria alcuna autorizzazione né autenticazione per questa PoC.

Lo shortener dovrà obbligare l'utilizzo di HTTPS per tutte le redirect fornite.

La nostra soluzione

Cercheremo di non utilizzare la soluzione ovvia di esporre semplicemente un'API RESTFul pubblica utilizzando API Gateway, Lamba e DynamoDB perché non sarebbe divertente :) e, cosa più importante, non è la soluzione più efficiente in termini di costi e scalabilità.

Combineremo invece i seguenti servizi AWS:

- **Amazon S3** come Object storage per i link e come motore di reindirizzamento.
- **Amazon CloudFront** come CDN e punto di ingresso unico per tutte le richieste.
- **AWS Lambda@Edge** per l'elaborazione back-end
- **Amazon Route53** per gestire le voci DNS.
- **AWS Amazon Certificate Manager** per generare e rinnovare automaticamente i certificati HTTPS.

L'idea è quella di creare una semplice API sfruttando Lambda@Edge.

Lambda@Edge è una feature di Amazon CloudFront che permette di renderlo il **punto di ingresso** del codice ed eseguire Lambda **più vicino agli utenti** che effettuano le richieste. Ciò **migliora le prestazioni e riduce la latenza**; ha anche **funzionalità di caching integrata** sfruttando l'infrastruttura di CloudFront. Lambda@Edge consente di creare un'applicazione davvero globale senza gestire l'infrastruttura in più region in tutto il mondo.

Quando l'utente chiama l'API dello shortneer, la richiesta raggiunge il miglior nodo CloudFront per l'utente in base al traffico ed alla zona del mondo di origine. CloudFront avvierà quindi un lambda nella posizione più vicina.

La logica del backend su lambda dovrebbe essere molto semplice; fondamentalmente calcola **k**, una stringa univoca di 8 caratteri composta da lettere maiuscole, lettere minuscole e numeri. Il numero totale di collegamenti che possono essere contemporaneamente attivi è, quindi, 218.340.105.584.896 (218+ collegamenti Tera).

Ora abbiamo bisogno di **un posto dove archiviare la nostra chiave univoca e l'URL corrispondente**, e DynamoDB sembrerebbe la scelta più ovvia.

C'è però una caratteristica di Amazon S3 quasi sconosciuta, che può essere utilizzata per diminuire ulteriormente il costo della soluzione, mantenendo prestazioni e scalabilità eccellenti.

Reindirizzare le richieste ad un oggetto S3

È possibile reindirizzare le richieste di un oggetto a un altro oggetto o a un URL arbitrario impostando la posizione di reindirizzamento nei metadati dell'oggetto. L'aggiunta della proprietà x-amz-website-redirect-location ai metadati dell'oggetto S3 è tutto ciò che serve per ottenere un reindirizzamento 301 ogni volta che si richiede la chiave dell'oggetto. Inoltre, l'oggetto può essere vuoto (0 byte).

Possiamo sfruttare questa funzione utilizzando S3 come archivio dati integrato con CloudFront come nostro database key-value, creando oggetti di 0 byte denominati utilizzando k come chiave e impostando x-amz-website-redirect-location sull' URL originale.

Tornando al nostro scenario: Lambda@edge archivia un oggetto vuoto in un bucket Amazon S3. La chiave dell'oggetto è impostata a k e la proprietà x-amz-website-redirect-location è imposta sull'URL originale.

Gli oggetti contenuti nel Bucket S3 vengono utilizzati per causare il reindirizzamento HTTP quando l'utente segue il collegamento tramite CloudFront.

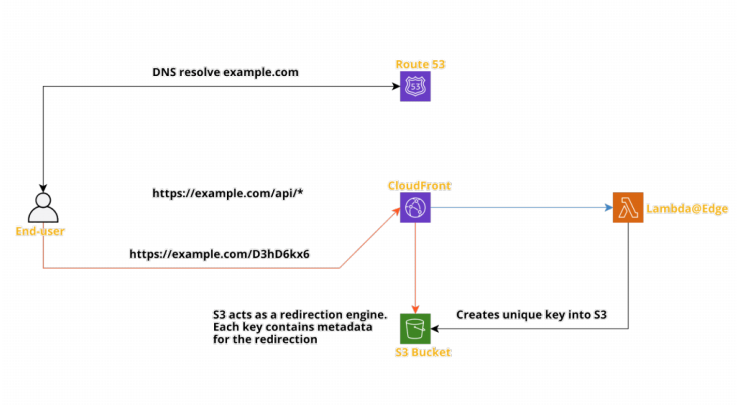
Dobbiamo configurare la nostra distribuzione CloudFront per utilizzare il bucket S3 come origine.

Questa soluzione è economica e veloce, grazie alla distribuzione CloudFront. Utilizza solo servizi serverless e può scalare in modo rapido e affidabile fino a milioni di impressioni.

È anche possibile configurare un meccanismo per eliminare i vecchi collegamenti utilizzando le lifecycle policies di S3.

Infrastruttura AWS

Quello che segue è lo schema infrastrutturale della soluzione.



Il punto di ingresso del sistema è sempre CloudFrontCDN; integra sia Lambda@Edge per l'API che Amazon S3 per i reindirizzamenti dei link. Il routing è basato sul percorso.

La CDN può o meno memorizzare nella cache i risultati dell'API; pertanto, puoi configurare CloudFront per memorizzare nella cache le risposte API, riducendo il costo di calcolo della soluzione. Se abiliti la memorizzazione nella cache, le chiamate successive per abbreviare lo stesso URL otterranno la stessa versione abbreviata e non eseguiranno una funzione Lambda. In caso contrario, ogni richiesta attiverà Lambda e produrrà collegamenti abbreviati diversi e univoci.

Si noti che per questa soluzione, il servizio VPC non viene utilizzato affatto, in quanto la soluzione sfrutta i servizi completamente gestiti nello spazio di rete AWS.

Motore di reindirizzamento

Il motore di reindirizzamento viene realizzato utilizzando Amazon S3.

Il Bucket è configurato come Web hosting statico pubblico, ciò è richiesto dal motore di reindirizzamento per funzionare. Ogni oggetto ha metadati per istruire l'hosting web a reindirizzare l'utente con un codice di stato HTTP 301.

Questo bucket viene utilizzato come una delle origini per la CDN, pertanto la redirect viene servita tramite CDN, non direttamente da S3. Possiamo configurare S3 in modo che il contenuto possa essere richiesto solo tramite CloudFront, riducendo il rischio di accessi S3 indesiderati e i relativi costi.

Il reindirizzamento per ogni chiave univoca viene memorizzato nella cache nella rete CDN per risparmiare sui costi e prestazioni.

Logging e Dubbing

Lambda è configurato per eseguire lo streaming dei log in AWS CloudWatch ed è possibile impostare il periodo di conservazione per ciascun gruppo di log.

Poiché i log di Lambda@Edge vengono trasmessi in streaming alla regione AWS più vicina, i log vengono archiviati nella stessa regione in cui viene creato il collegamento.

Wrap-up

Quindi, per concludere, questa è una soluzione multi-region, completamente serverless ed ad alta disponibilità per creare un semplice URL shortener. Un ottimo punto di partenza per ulteriori affinamenti e discussioni e per soluzioni più complesse basate sugli stessi principi!

Avete realizzato soluzioni simili? Quali servizi avete utilizzato? Fatecelo sapere!

Alessio Gandini

Cloud-native Development Line Manager @ beSharp, DevOps Engineer e AWS expert.Computer geek da quando avevo 6 anni, appassionato di informatica ed elettronica a tutto tondo. Ultimamente sto esplorando l'esperienza utente vocale e il mondo dell'IoT.Appassionato di cinema e grande consumatore di serie TV, videogiocatore della domenica.
