Da monolite infrastrutturale a Cloud-Native: la nuova vita di un vecchio Load Balancer

24 Settembre 2025 - 8 min. read

Amazon CloudFront | *AWS Lambda@Edge* | *Cloud Migration*

“Dobbiamo guardare le cose in modo diverso.” Robin Williams (John Keating) – L’attimo fuggente

Ahh, scomporre un monolite! Il classico esercizio da sviluppatore che tutti hanno affrontato almeno una volta nella loro carriera.

Ma come smantellare un monolite infrastrutturale? Come rendere agile e modulare un sistema grande e complesso?

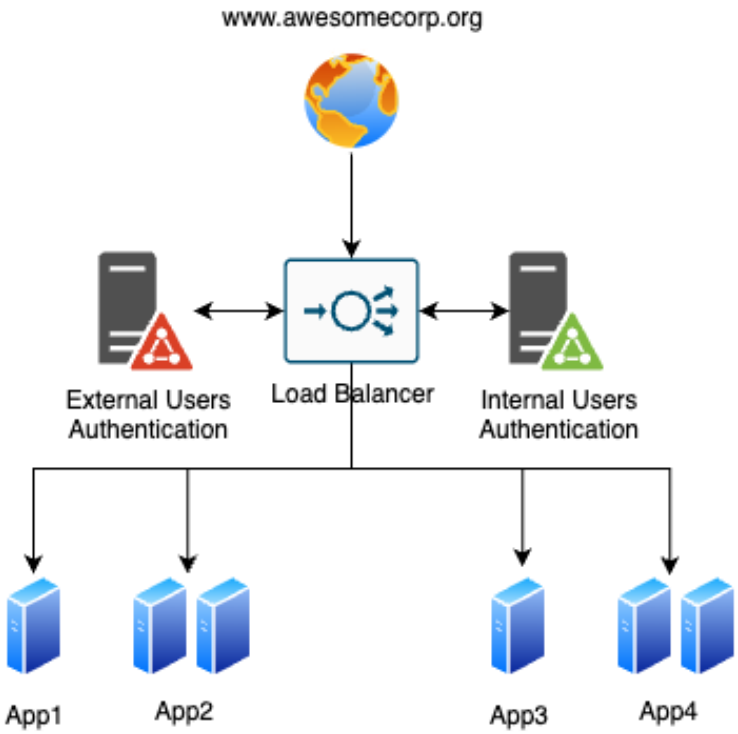
Per questo articolo, parliamo di un sistema che ha visto passare generazioni di sysadmin che hanno aggiunto configurazioni secondo il proprio gusto personale. Ci siamo trovati ad avere a che fare con un Load Balancer che ha superato upgrade di kernel, virtualizzazione, cambiamenti del linguaggio di configurazione e migrazioni al punto che nessuno voleva più toccarlo, nemmeno con un bastone lungo 5 metri!

Ma a noi piacciono le sfide, quindi quando ci hanno chiesto di partecipare a questo progetto di migrazione quasi impossibile, da sistemi legacy on-premise a servizi gestiti cloud-native AWS mantenendo il sito operativo, abbiamo quasi festeggiato.

In questo articolo vedremo la parte più importante del percorso di migrazione, mettendo in evidenza l’approccio e il pattern usato per realizzare questa impresa che sembrava impossibile.

Il mostro

Immaginate un sito aziendale nato a metà degli anni ’90 e che ha visto l’intera trasformazione di Internet. Da un semplice sito statico con tag <blink> e gif animate, è diventato un capolavoro di design e il punto centrale per tutto ciò che riguarda il business: meeting, calendari, intranet pubblica e privata (per consulenti e staff interno con diversi Identity provider e regole di autenticazione personalizzate), micro-siti marketing con molteplici installazioni dell’onnipresente WordPress. Tutto ovviamente instradato e gestito utilizzando una coppia di load balancer in alta affidabilità e un singolo dominio DNS, senza sapere quali parti del sito fossero ancora utilizzate. Diversi identity provider sono usati, uno per lo staff interno e uno per i contractor, aggiungendo complessità a uno scenario già complesso.



L’approccio

Uno dei miei pattern preferiti per smantellare i monoliti è lo “Strangler Fig Pattern” di Martin Fowler. Utilizzando questo approccio, si sostituisce un sistema vecchio gradualmente, riscrivendo i componenti in modo incrementale e mettendoli subito in produzione, mantenendo il sistema originale finché serve.

Nel nostro caso, il sistema è composto da tre componenti principali:

- Backend Load Balancing

- Un portale di autenticazione che instrada le richieste ai server di autenticazione appropriati in base al dominio dell'utente e usando protocolli diversi (LDAP Active Directory, OIDC, SAML)
- Routing applicativo per la selezione del backend, basato su subpath

A questo punto, abbiamo compreso che ci serviva qualcosa che potesse instradare le richieste permettendoci di lavorare dietro le quinte e sostituire i componenti in silenzio; la stessa tecnologia doveva permetterci anche di implementare un'architettura Proof-of-Concept per validare le nostre ipotesi. Fortunatamente, esiste Amazon CloudFront.

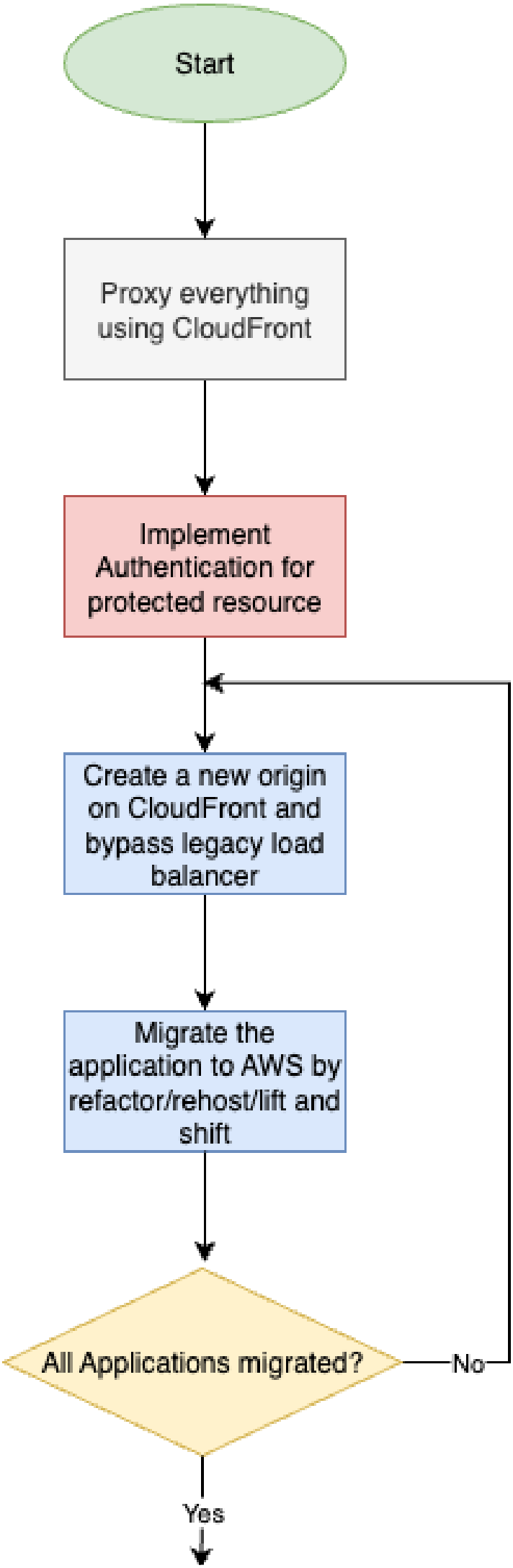
CloudFront come strangler fig

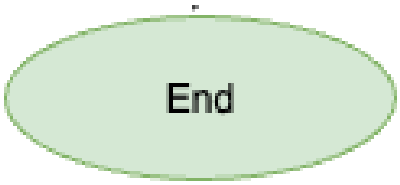
Amazon CloudFront è una CDN globale il cui comportamento è facilmente personalizzabile per soddisfare diverse esigenze. Ad esempio, utilizzando CloudFront Functions, è possibile modificare le richieste HTTP prima che raggiungano il server o alterare le risposte prima di servirle al client.

Come inserirsi nel flusso del load balancer

Per sostituire il load balancer per prima cosa occorre creare una distribuzione Amazon CloudFront "temporanea" con il nome di dominio attuale per testare che tutto funzioni.

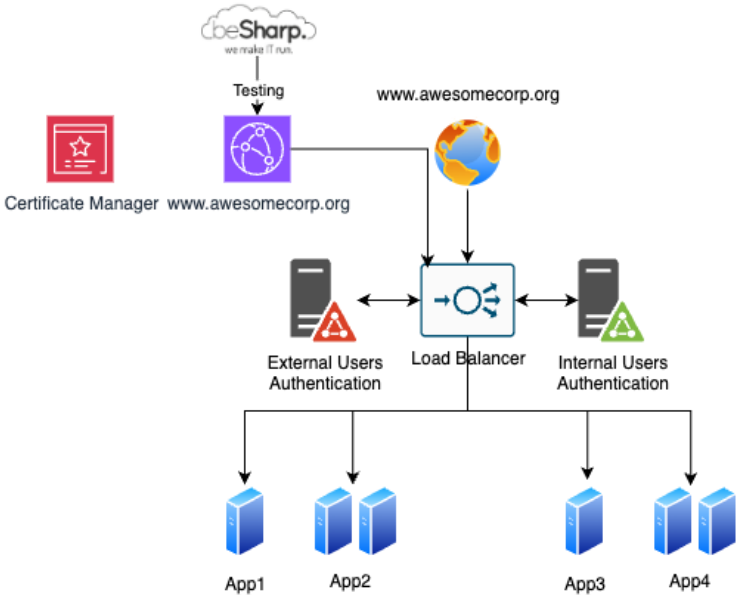
È possibile importare il certificato esistente in Amazon Certificate Manager (ACM) e configurare il load balancer legacy come origin; poiché il record DNS non viene modificato, il sito non subisce malfunzionamenti: per testare il setup, infatti, basta modificare il file hosts puntando agli IP della distribuzione CloudFront.


Start
Proxy everything using CloudFront
Implement Authentication for protected resource
Create a new origin on CloudFront and bypass legacy load balancer
Migrate the application to AWS by refactor/rehost/lift and shift
All Applications migrated?
No
Yes



Una volta testato il corretto funzionamento anche con l'inserimento di CloudFront, siamo riusciti ad attaccare la parte più critica non disponibile nativamente su CloudFront: **l'autenticazione**.

Implementando l'autenticazione, possiamo fare replatform o lift-and-shift dei server nel Cloud e scomporre le applicazioni senza toccare il load balancer originale.



Flusso di autenticazione

Il load balancer legacy ospita un backend interno custom (auth.awesomecorp.org) che gestisce l'autenticazione.

Ogni richiesta viene intercettata, controllando JWT, Cookie e altri meccanismi di autenticazione. Se l'utente accede a una risorsa protetta senza essere autenticato, viene mostrata una pagina ospitata dal load balancer.

Dopo l'inserimento delle credenziali, il load balancer stabilisce il backend di autenticazione utente corretto (in base al formato dello username) e procede con l'autenticazione. Terminata l'autenticazione, viene servito un cookie con contenuti diversi a seconda del path dell'applicazione.

Come possiamo implementare questo meccanismo solo con servizi cloud-native?

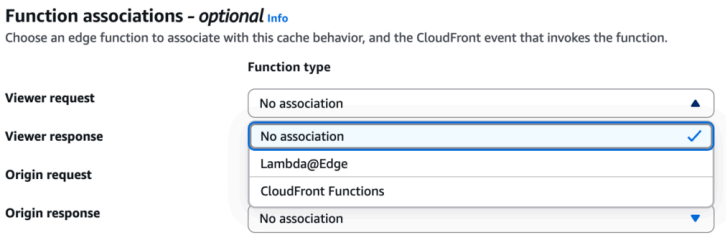
CloudFront intercetta richieste e risposte nelle edge location e attiva eventi diversi a seconda dello stato della richiesta HTTP: viewer request, origin request, origin

response, viewer response.

Tipi di evento e uso tipico

Di seguito una tabella riassuntiva delle opzioni disponibili e del loro utilizzo:

Tipo di Evento	Punto di Attivazione	Caso d'uso Tipico
Viewer Request	Quando CloudFront riceve la richiesta	URL rewrite, autenticazione
Origin Request	Prima dell'inoltro all'origine	Selezione dinamica dell'origine
Origin Response	Dopo la risposta dell'origine	Header di sicurezza, gestione errori
Viewer Response	Prima della risposta al viewer	Cookie injection, personalizzazione



Come regola generale, le CloudFront Functions possono essere utilizzate per modificare semplici testi. Quando però la logica di implementazione diventa più complessa e c'è bisogno di interrogare servizi esterni, Lambda@Edge diventa la scelta ideale.

Meccanismo di autenticazione serverless

Abbiamo scritto il flusso di autenticazione per utilizzare solo componenti serverless, con una Lambda custom per autenticare gli utenti. Questo ci permette di integrare ogni meccanismo usato, partendo dal più semplice e implementando man mano i meccanismi più complessi.

Quando un utente non autenticato accede alla distribuzione CloudFront, una Viewer Request CloudFront Function verifica la presenza di un token JWT.

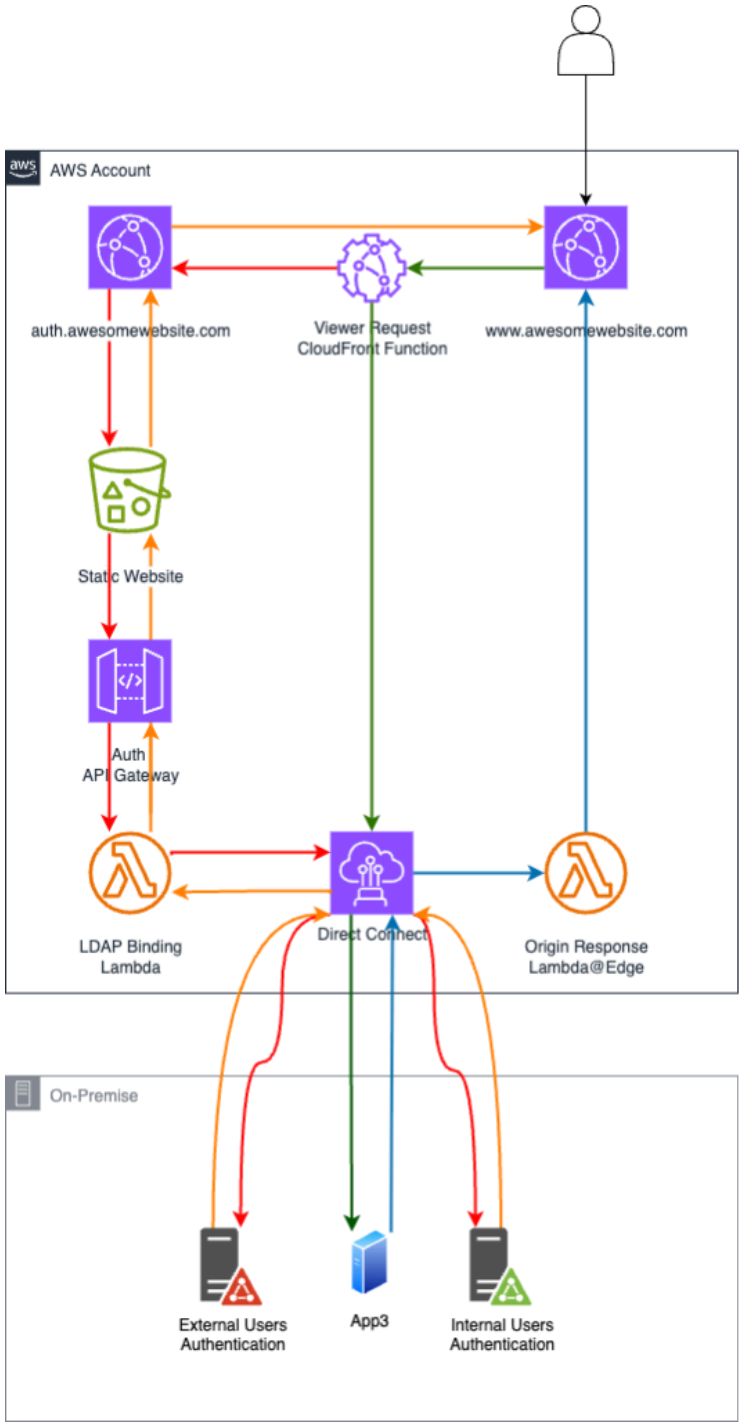
Se non è valido, la funzione reindirizza l'utente a una pagina di login personalizzata ospitata su una distribuzione CloudFront separata con backend su Amazon S3.

Le credenziali a questo punto vengono inviate a un API Gateway che inoltra la richiesta a una Lambda. La Lambda esegue l'autenticazione (LDAP binding, chiamate OIDC, SAML, ecc.) e restituisce un token JWT firmato. Questo token viene poi salvato come cookie HTTP-Only e restituito all'utente tramite una Viewer Response Lambda@Edge.

Da quel momento, ogni richiesta viene intercettata da una CloudFront Function Viewer Request, che verifica il token JWT e, se valido, instrada l'utente al backend corretto.

Il flusso si semplifica per gli utenti autenticati: la CloudFront Function Viewer Request valida il token JWT esistente e inoltra la richiesta al servizio backend includendo l'header di autorizzazione per la verifica dell'utente.

Le risposte tornano attraverso una Lambda@Edge Origin Response che monitora eventuali fallimenti di autenticazione (con status code 401) e invalida automaticamente il token quando necessario, costringendo l'utente a ri-autenticarsi se la sessione risulta non più valida.



Descrizione del flusso:

1. Utente non autenticato

1. [Frecce verdi] L'utente esterno contatta la distribuzione CloudFront.
2. [Frecce verdi] La distribuzione CloudFront esegue la Viewer Request CloudFront Function, che verifica se il token JWT è stato inviato nella richiesta.
3. [Frecce rosse] Essendo un utente non autenticato, la Viewer Request CloudFront Function risponderà con un reindirizzamento verso la pagina di login personalizzata.
4. [Frecce rosse] La pagina di login personalizzata è servita dalla distribuzione CloudFront di auth.awesomecorp.com, che punta al Bucket S3 che ospita il sito

web statico.

5. [Frecce rosse] L'utente inserisce le proprie credenziali e clicca sul pulsante di login. La pagina di login personalizzata invia una richiesta all'API Gateway.
6. [Frecce rosse] L'API Gateway inoltra la richiesta di login alla Lambda LDAP Binding.
7. [Frecce rosse] La Lambda LDAP Binding seleziona il backend di autenticazione appropriato ed esegue una richiesta LDAP Binding verso il Domain Controller Active Directory selezionato.
8. [Frecce arancioni] Il Domain Controller Active Directory risponde alla richiesta con il risultato dell'operazione LDAP Binding.
9. [Frecce arancioni] Se l'LDAP Binding ha successo, la Lambda LDAP Binding risponde all'API Gateway con uno status code 200.
10. [Frecce arancioni] L'API Gateway inoltra la risposta alla pagina di login personalizzata.
11. [Frecce arancioni] La pagina di login personalizzata inoltra la risposta alla distribuzione CloudFront di auth.awesomecorp.com, reindirizzando l'utente alla distribuzione www.awesomecorp.com e impostando il JWT Token.

L'utente è ora autenticato e seguirà il flusso descritto nella sezione successiva.

2. Utente autenticato

1. [Frecce verdi] L'utente esterno contatta la distribuzione CloudFront di www.awesomecorp.com
2. [Frecce verdi] La distribuzione CloudFront esegue la Viewer Request CloudFront Function, che verifica se il JWT Token è stato inviato nella richiesta.
3. [Frecce verdi] Essendo un utente autenticato, la Viewer Request CloudFront Function inoltra la richiesta al backend host, che procederà con la verifica dell'utente.
4. [Frecce blu] Il servizio backend risponde alla richiesta, inviandola alla distribuzione CloudFront. La Lambda@Edge Origin Response verifica che la risposta abbia uno status code diverso da 401.
5. [Frecce blu] La Lambda@Edge Origin Response inoltra la risposta all'utente se lo status code è diverso da 401; altrimenti, invalida il JWT Token, rendendo l'utente non autenticato.

Next Step

Una volta implementato, questo flusso può essere esteso e modificato, permettendoci di spostare e riscrivere i servizi backend in modo sicuro e senza impattare sulla disponibilità del sito web. Anche in caso di migrazioni più lunghe, è possibile mettere una sottosezione in “modalità manutenzione” ospitando una pagina web statica in un bucket S3.

Smantellare un monolite infrastrutturale non è solo possibile, ma può essere fatto in modo elegante. Usando pattern architetturali collaudati e strumenti potenti come Amazon CloudFront, CloudFront Functions e Lambda@Edge, è possibile trasformare un sistema legacy in una piattaforma moderna, scalabile e manutenibile.

A volte, per innovare, basta guardare le cose da un’altra prospettiva.

Hai a che fare con vecchi load balancer? Hai in mente percorsi o pattern di migrazione alternativi? Raccontacelo nei commenti!

About Proud2beCloud

Proud2beCloud è il blog di beSharp, APN Premier Consulting Partner italiano esperto nella progettazione, implementazione e gestione di infrastrutture Cloud complesse e servizi AWS avanzati. Prima di essere scrittori, siamo Solutions Architect che, dal 2007, lavorano quotidianamente con i servizi AWS. Siamo innovatori alla costante ricerca della soluzione più all'avanguardia per noi e per i nostri clienti. Su Proud2beCloud condividiamo regolarmente i nostri migliori spunti con chi come noi, per lavoro o per passione, lavora con il Cloud di AWS. Partecipa alla discussione!

Damiano Giorgi

Ex sistemista on-prem, pigro e incline all'automazione di task noiosi. Alla ricerca costante di novità tecnologiche e quindi passato al cloud per trovare nuovi stimoli.L'unico hardware a cui mi dedico ora è quello del mio basso; se non mi trovate in ufficio o in sala prove provate al pub o in qualche aeroporto!
