CONTINUOUS DELIVERY DI UN SITO WEB STATICO CON AWS CODEBUILD

AWS CodeBuild | CI/CD | Continuous Delivery

beSharp | 28 Giugno 2019

Questo articolo è il primo di una breve serie volta a illustrare vari possibili utilizzi dei servizi di Continuous Integration (CI) e Continuous Delivery (CD) offerti da AWS.

Una delle applicazioni più comuni di uno stack di Continuous Delivery è senza dubbio l'automatizzazione del deploy di un sito web statico o del front-end JavaScript di una web app. Per l'hosting dei contenuti statici, AWS offre una soluzione estremamente scalabile e potente: i file HTML e JavaScript del sito web possono essere caricati direttamente su S3, l'object storage service di AWS, e distribuiti usando un URL fornita dalla stessa Amazon. È ovviamente anche possibile usare un un domain name personale, andando semplicemente ad indicare degli Alias con Route 53, il servizio DNS di AWS.

Distribuire i contenuti statici di un sito web tramite S3 offre molti vantaggi rispetto al tradizionale hosting su un web server, in particolare:

- Alta affidabilità: i file salvati su S3 vengono ridondati su più dischi distribuiti nella Region nella quale si è scelto di creare il bucket. Questo garantisce una latenza di accesso ai file molto bassa, oltre ad un'altissima durabilità e accessibilità dei contenuti memorizzati (rispettivamente 99.999999999% e 99.99%, su base annua)
- Scalabilità: i contenuti possono essere distribuiti su scala globale in modo totalmente automatico tramite CloudFront, la Content Delivery Network di AWS, garantendo un tempo di latenza molto basso per l'accesso ai file, indipendentemente dalla provenienza geografica dei visitatori del sito.

Tuttavia, caricare manualmente gli oggetti su un Bucket S3 può risultare inefficiente e poco pratico per sviluppi relativamente grandi, come ad esempio per il front-end di una web app complessa, dove più programmatori collaborano allo stesso progetto.

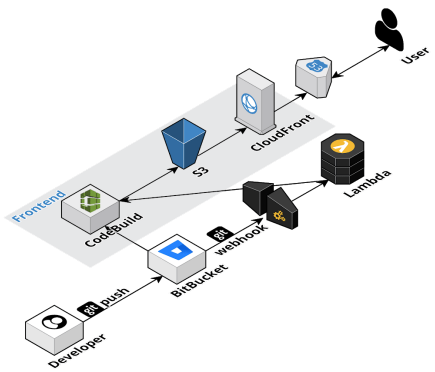
La soluzione universalmente adottata per ovviare a questo tipo di inconveniente è l'utilizzo di una pipeline di Continuous Integration/Delivery, che viene attivata in modo automatico nel momento in cui uno degli sviluppatori esegue un commit del codice da un version control system (e.g. Git,

Subversion etc...). Nel caso di contenuti statici, la pipeline è particolarmente semplice, dato che non è necessario compilare codice o gestire il deploy su un server: l'unica operazione consiste semplicemente nel copiare i file dal repository al bucket S3. De fosse necessario, è anche possibile eseguire dei test automatici sul codice per accertarsi del corretto funzionamento delle pagine web prima di aggiornare la versione presente su S3, per esempio utilizzando la libreria Selenium.

Di seguito verrà analizzata la pipeline per il deploy di un sito statico partendo da un repository Git su Bitbucket. Questa pipeline si basa su due degli strumenti messi a disposizione da AWS: CodeBuild e Lambda Functions.

Lo schema concettuale della pipeline è il seguente:

- Il developer esegue il push di un commit su Bitbucket;
- Bitbucket viene configurato in modo da eseguire una chiamata POST ad un endpoint, per notificare che un push del codice è stato eseguito con successo;
- Una funzione Lambda viene eseguita (tramite AWS ApiGateway) in risposta alla ricezione della chiamata POST effettuata di BitBucket, ed avvia l'esecuzione di un progetto creato su AWS CodeBuild;
- CodeBuild esegue un pull del codice da Bitbucket, esegue i test (se necessario) e la build del codice ed infine salva il codice del front-end su S3.



Abbiamo scelto di lanciare il processo di build usando una Lambda invece di usare il servizio di CodePipeline offerto da AWS: CodePipeline non ha infatti dei trigger pre-configurati per Bitbucket e dunque sarebbe stato necessario, sempre tramite una Lambda, scaricare i file da Bitbucket su un bucket S3 versionato che avremmo dovuto usare nello stadio di Source della pipeline. Tutto questo avrebbe allungato i tempi di esecuzione della Pipeline, in particolare in presenza di molti file piccoli, come spesso è il caso per i progetti di front-end.

I servizi di hosting di repository Git offerti da AWS (CodeCommit) e da GitHub hanno invece dei trigger pre-configurati per CodePipeline e, quindi, se si decide di avvalersi di uno di questi servizi per il processo di deploy dei file su S3, questo viene triggerato in modo automatico senza il bisogno di utilizzare una Lambda.

Se la nostra soluzione vi ha incuriositi e vorreste implementarla nel vostro flusso di lavoro, lasciate un commento o contattateci! Saremo felici di rispondere alle vostre domande, di raccogliere i vostri feedback e di aiutarvi a trarre il massimo vantaggio da questa applicazione.

beSharp

Dal 2011 beSharp guida le aziende italiane sul Cloud. Dalla piccola impresa alla grande multinazionale, dal manifatturiero al terziario avanzato, aiutiamo le realtà più all'avanguardia a realizzare progetti innovativi in campo IT.

Get in touch

beSharp.it
proud2becloud@besharp.it