Cloud Computing (**LAB**)

HANDS ON

Amazon
Lambda

Giuseppe Bisicchia
giuseppe.bisicchia@phd.unipi.it

Dipartimento di Informatica, Università di Pisa

Cosa faremo?

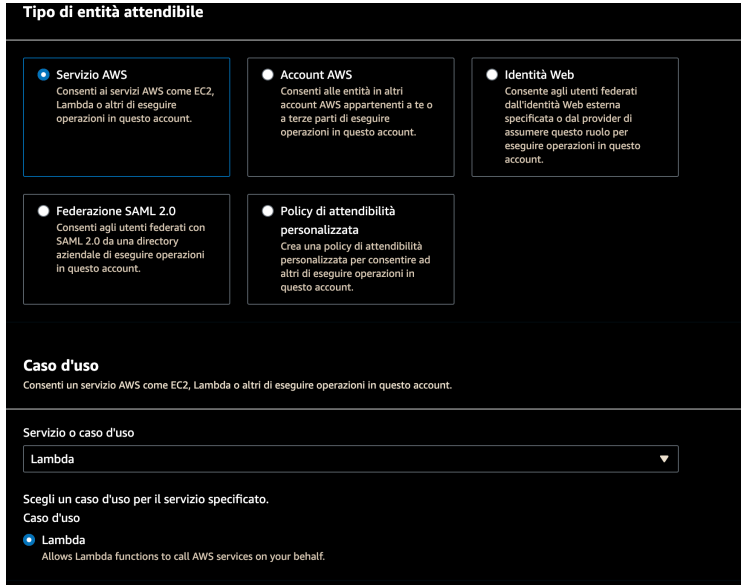
Implementeremo un'**AWS Lambda Function** per la cifratura di file testuali memorizzati su **S3**.

Cosa faremo?

- La **funzione** prende **un file di testo** all'interno di una **cartella** di un **bucket S3** e ne crea una **copia cifrata** in una cartella ***crypted*** all'interno dello **stesso bucket**. Al contempo **l'originale viene eliminata.** In un'ulteriore cartella ***decrypted*** ne viene depositata una **copia decifrata** per verificare che il processo di cifratura sia andato a buon fine.
- La funzione è già fornita e disponibile su **moodle** (`lambda_function.py`)

Step 1 – Creare Ruolo IAM

- Così come per EC2 anche per le **Lambda Functions** dobbiamo prima **concedere i permessi** per poter **interagire** con i nostri **buckets S3**.
- Il processo di creazione è analogo a quello per creare un ruolo per EC2.
- Questa volta però come **caso d'uso** scegliamo **Lambda**.
- Come policy scegliamo ***AmazonS3FullAccess***



Step 2 – Creare Bucket S3

- La nostra *Lambda Function* dovrà lavorare interagendo con un *bucket S3*.
- Quindi creiamone uno come fatto nel *Lab 1*.
- Questa volta creiamo all'interno del bucket **tre cartelle**
  - `crypted/`
  - `decrypted/`
  - `plain/`

Step 3 – Creare la Lambda Function

- Una volta pronto il ruolo IAM e il bucket S3 possiamo procedere con la **creazione** della *Lambda Function*.
- Specificare *«Autore da zero»*
- Come *«Runtime»* usare **Python 3.8**
- In *«Autorizzazione»* scegliere *«Utilizza un ruolo esistente»* e inserire il ruolo creato precedentemente
- Dopo aver creato la funzione inserire il **codice** presente nel file `lambda_function.py` disponibile su **moodle**.
- Dopo **ogni modifica** al codice ricordarsi di cliccare su ***Deploy***.

Step 4 – Testare la Lambda Function

- Dopo aver creato la *Lambda Function* provare a generare un **evento di test**.
- Al momento non ci interessa la struttura dell'evento, utilizzare quindi quella di **default**.
- La funzione restituisce un **errore**. Perché?

{

"errorMessage": "Unable to import module 'lambda_function': No module named 'cryptography'",

"errorType": "Runtime.ImportModuleError",

"stackTrace": []

}

Step 5 – Creare un livello in AWS Lambda

- In AWS Lambda quando creiamo una funzione dobbiamo specificare un Runtime da utilizzare. Questo però contiene solo **funzionalità e librerie di base**.
- Se si vogliono utilizzare ulteriori librerie è necessario creare dei ***Livelli Lambda*** che verranno utilizzati dalle nostre funzioni.
- Un Livello Lambda è «*un archivio di file con estensione .zip che può contenere codice o dati aggiuntivi. I livelli Lambda offrono un metodo pratico per impacchettare librerie e altre dipendenze che è possibile utilizzare insieme alle funzioni Lambda.*»
- Nel nostro caso la libreria ***cryptography*** non fa parte del *Runtime* e dobbiamo quindi inserirla tramite un livello.
- Creiamo quindi un **nuovo livello** utilizzando l'archivio "`python.zip`"disponibile su **moodle** che contiene già tutto il necessario.
  - **N.B.** il file zip non va rinominato, né modificato
  - **Importante:** nella creazione del livello bisogna specificare come runtime lo stesso della funzione (Python 3.8)

Step 6 – Aggiungere il Livello

- Dopo aver creato il Livello, possiamo aggiungerlo alla nostra funzione.
- Per farlo basta andare su *«Aggiungi un livello»*
- Quindi selezionare *«Livelli personalizzati»* e scegliere il livello appena creato e la versione di default.
- Riprovando ad eseguire il Test otteniamo ancora un errore, questa volta diverso.

{ "errorMessage": "'Records'", "errorType": "KeyError", "stackTrace": [" File \"/var/task/lambda_function.py\", line 42, in lambda_handler\n for record in event['Records']:\n"]}

- L'errore indica che la **funzione** riesce ad andare in **esecuzione** ma **l'input ricevuto** è **mal formattato**. Avevamo infatti usato un input di **default**. Ora è il momento di creare un **input** effettivamente **eseguibile** dalla funzione.

Esercizio 1 – Testare (nuovamente) la Lambda Function

- Le funzioni Lambda lavorano ricevendo in input e rispondendo in output oggetti JSON (https://en.wikipedia.org/wiki/JSON)
- Prendendo d'esempio un evento PUT di S3 (https://docs.aws.amazon.com/AmazonS3/latest/userguide/notification-content-structure.html) e analizzando il codice a disposizione, provare a capire che tipo di informazioni sono necessarie alla nostra funzione per lavorare correttamente.
- Quindi generare un possibile evento di test.

Step 7 – Creare un livello in AWS Lambda

- Come evento possiamo usare:

{ "Records": [{ "s3": { "bucket": { "name": "BUCKET" }, "object": { "key": "plain/FILENAME" } } }]}

- In cui sostituire a *BUCKET* il nome del nostro bucket e a *FILENAME* il nome di un file da cifrare (terminante in *.txt*) e precedentemente caricato nella cartella *plain/* del nostro bucket.

Step 8 – Aggiungere un Trigger

- È possibile automatizzare l'esecuzione delle *Lambda Functions* facendo in modo che vengano **lanciate automaticamente** ogni volta che avviene un **evento** particolare.
- Nel nostro caso vogliamo fare in modo che ogni volta che viene **caricato un file** nella cartella ***plain/*** del nostro **bucket** la *Lambda Function* viene **invocata automaticamente**.
- Per farlo cliccare su *«Aggiungi trigger»* e come origine scegliere **S3**.
- Selezionare dunque il bucket di interesse.
- Come *«Tipo di evento»* scegliere *«Tutti gli eventi di creazione oggetti».*
- Aggiungere come **prefisso** *plain/* per far in modo che la funzione venga chiamata solo quando viene caricato un **file nella cartella plain**.
- Infine **accettare** le condizioni sull'*Invocazione ricorsiva*.
- **ATTENZIONE**: Per evitare problemi durante l'esercizio, eliminate tutti i file già presenti nel bucket.
- **Caricare** quindi un file nella cartella *plain/* e verificare l'attivazione della funzione.

Esercizio 2 – Generare altri eventi di test

- Provate a generare altri eventi verificando anche casi limite o di errore (es., il file non è presente nel bucket o il bucket non è stato creato)

Esercizio 3 – Eliminazione automatica

- Analizzate il codice della funzione e provate ad eliminare quelle parti di codice che fanno sì che il file originale venga eliminato dopo la cifratura.

Esercizio 4 – Lambda & API Gateway

- Seguite questa guida per scoprire un possibile uso più avanzato delle *Lambda Functions*:
- https://docs.aws.amazon.com/it_it/lambda/latest/dg/services-apigateway-tutorial.html

Importante

Per non rischiare di sforare il piano gratuito verificate di aver eliminato ogni artefatto creato durante la lezione!

Controllate regolarmente il seguente link https://us-east-1.console.aws.amazon.com/billing/home#/freetier per accertarvi di non avere risorse attivite nascoste