
Usecase Datalake

Beschreibung

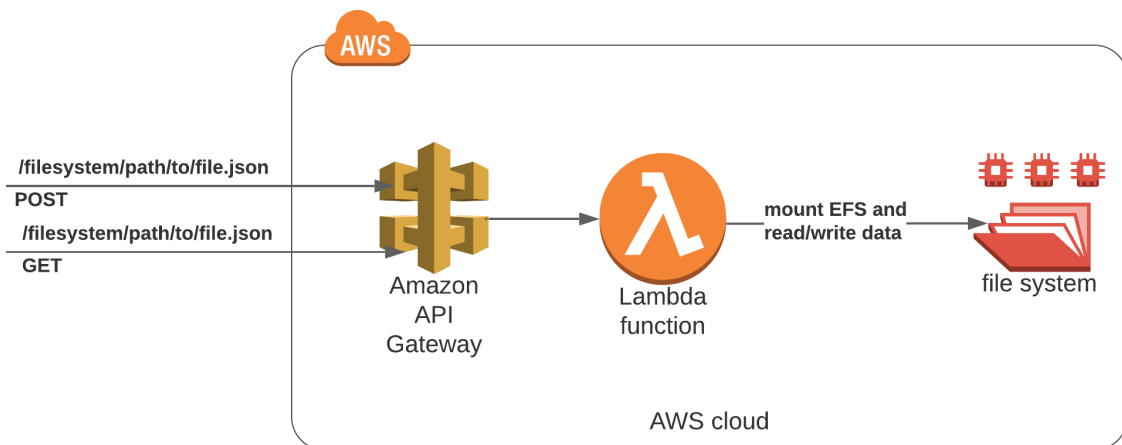
Ziel ist es ein mittels serverless Funktionen einen Datalake zu erstellen, der strukturierte und unstrukturierte Datensätze speichern kann.

User Stories

Als Datawarehouse Admin möchte ich
die Möglichkeit Datensätze aus Quellsystemen egal welcher Form zwischen zu speichern,
damit die Rohdaten quasi archiviert werden.

Als Datawarehouse Admin möchte ich
das die Lösung Serverless umgesetzt wird,
damit eventuelle Lastspitzen abgefangen werden ohne das Server ausgelastet werden.

Architektur



Es wird ein API Gateway erstellt welches den Pfad `/filesystem` öffentlich zugänglich macht. Alle weiteren Pfade dahinter dienen für das Dateisystem als Speicherpfad. Das Speichern und Lesen wird hierbei von einer Lambda funktion übernommen, damit Lastspitzen gut abgefangen werden können.

Aufsetzen

Source Code Compilieren

Nur notwendig wenn Änderungen am Code vorgenommen wurden.

Es muss Go auf dem entsprechenden System installiert sein. Installationsanweisungen können hier gefunden werden. <https://golang.org/doc/install>

Nun muss in den Ordner `./cc3-aws/datalake/writereadfs` navigiert werden. Dort wird der folgende Befehl ausgeführt:

```
1 env GOOS=linux GOARCH=amd64 go build -o main ./main.go && zip -r main.zip main
```

Dieser baut aus dem vorhandenen Sourcecode eine Binary für die Zielplattform `linux`. Diese Binary mit dem Namen `main` wird als `main.zip` Archiv verpackt.

File System (EFS)

Als erstes muss ein neues Dateisystem erstellt werden, welches dann in die Lambda Funktion gemounted wird. Ein Dateisystem kann unter EFS erstellt werden. Dabei muss ein Name vergeben werden und ein VPC festgelegt werden. Die spätere Lambda muss in dem gleichen VPC und in der gleichen Region sein!

Dateisystem erstellen

Erstellen Sie ein EFS-Dateisystem mit den vom Service empfohlenen Einstellungen.
[Weitere Informationen](#)

Name - (optional)
Benennen Sie Ihr Dateisystem.

datalake-fs

Der Name darf nicht länger als 256 Zeichen sein und darf nur Buchstaben, Zahlen und die folgenden Zeichen enthalten: + - = . _ : /

Virtual Private Cloud (VPC)
Wählen Sie die VPC aus, in der EC2-Instances eine Verbindung zum Dateisystem herstellen sollen.
[Weitere Informationen](#)

vpc-f95b8693
Standard

Verfügbarkeit und Beständigkeit
Wählen Sie Regional (empfohlen) aus, um ein Dateisystem mit regionalen Speicherklassen zu erstellen.
Wählen Sie One Zone, um ein Dateisystem mit One Zone-Speicherklassen zu erstellen. [Weitere Informationen](#)

☒ **Regional**
Speichert Daten
redundant in mehreren
AZs

☐ **One Zone**
Speichert Daten
redundant innerhalb
einer einzigen AZ

Abbrechen

Anpassen

Erstellen

Sobald diese erstellt ist muss ein Zugriffspunkt für die Lambdas erstellt werden. Hierzu wird das Wurzelverzeichnis verwendet. Da das Wurzelverzeichnis von root erstellt wird muss der Nutzer 0 der Gruppe 0 angegeben werden.

Details

Dateisystem

Wählen Sie das Dateisystem aus, dem der Zugriffspunkt zugeordnet ist.

 fs-f6a463ad

Name - (optional)

/

Maximal 256 Unicode-Buchstaben, Leerzeichen und Zahlen sowie +- =. _:/


Stammverzeichnispfad - (optional)

Verbindungen verwenden den angegebenen Pfad als virtuelles Stammverzeichnis des Dateisystems. [Weitere Informationen](#)

/

Beispiel: „/foo/bar“

POSIX-Benutzer - (optional)

Die vollständige POSIX-Identität auf dem Zugriffspunkt, der für alle Dateioperationen von NFS-Clients verwendet wird. [Informationen](#) 

Benutzer-ID

POSIX-Benutzer-ID für alle Dateisystemoperationen, die diesen Zugriffspunkt verwenden.

0

Akzeptiert Werte von 0 bis 4294967295


Gruppen-ID

POSIX-Gruppen-ID für alle Dateisystemoperationen, die diesen Zugriffspunkt verwenden.

0

Akzeptiert Werte von 0 bis 4294967295

Berechtigungen zum Erstellen des Stammverzeichnisses - (optional)

EFS erstellt automatisch das angegebene Stammverzeichnis mit diesen Berechtigungen, wenn das Verzeichnis nicht bereits existiert. [Weitere Informationen](#) 

Benutzer-ID des Besitzers

Benutzer-ID des Besitzers für das Stammverzeichnis des Zugriffspunkts, wenn das Verzeichnis nicht bereits existiert.

Akzeptiert Werte von 0 bis 4294967295

Gruppen-ID des Besitzers

Gruppen-ID des Besitzers für das Stammverzeichnis des Zugriffspunkts, wenn das Verzeichnis nicht bereits existiert.

Akzeptiert Werte von 0 bis 4294967295

POSIX-Berechtigungen, die auf den Stammverzeichnispfad angewendet werden

Eine Oktalzahl, welche die Modus-Bits der Datei darstellt.

Das Dateisystem ist hiermit eingerichtet.

Lambda Funktion

Als nächstes muss eine Lambda Funktion angelegt werden. Bei dieser muss die Go Runtime festgelegt werden und das VPC eingerichtet werden. Das VPC, die Subnetze und die Sicherheitsgruppe kann unter [Erweiterte Einstellungen](#) beim erstellen der Lambda eingerichtet werden.

Grundlegende Informationen

Funktionsname

Geben Sie einen Namen zur Beschreibung Ihrer I

datalake-ex

Verwenden Sie nur Buchstaben, Zahlen, Bindestr

Laufzeit [Info](#)

Wählen Sie die Sprache aus, die Sie zum Schreib

Go 1.x

Netzwerk

Um Netzwerkzugriff für Ihre Lambda-Funktion bereitzustellen, geben Sie eine Virtual Private Cloud (VPC), VPC-Subnetze und VPC-Sicherheitssgruppen an, die Sie eine VPC konfigurieren.

VPC - optional [Info](#)

Wählen Sie eine VPC aus, auf die Ihre Funktion zugreifen kann.

vpc-f95b8693 (172.31.0.0/16)

Subnetze

Wählen Sie die VPC-Subnetze für Lambda aus, die zum Einrichten Ihrer VPC-Konfiguration verwendet werden sollen.

Subnetze auswählen

subnet-33f07459 (172.31.16.0/20) eu-central-1a ✕

subnet-a9c231d5 (172.31.32.0/20) eu-central-1b ✕

Sicherheitsgruppen

Wählen Sie die VPC-Sicherheitsgruppen aus, die Lambda für das Einrichten der VPC-Konfiguration verwenden sollte. Die Tabelle unten be

Sicherheitsgruppen auswählen

sg-63175f1a (default) ✕
default VPC security group

Danach muss noch der Handler von der Lambda umgestellt werden, da die Go Binary main heißt.

Runtime settings Info		Edit
Runtime Go 1.x	Handler Info main	

Dann kann unter [Konfiguration](#) > [Dateisysteme](#) > [Dateisystem hinzufügen](#) das erstellte EFS eingebunden werden. Dabei wird der erstellte Zugriffspunkt genutzt und als mount-Pfad wird `/mnt/datalake` genutzt.

Für einen anderen Pfad müsste der Code angepasst werden.

Dateisystem

Sie können Ihrer Funktion ein vorhandenes Amazon Elastic File System (EFS) zuordnen. Besuchen Sie die Amazon EFS-Konsole, um [ein neues Dateisystem zu erstellen](#).

EFS-Dateisystem
Wählen Sie ein vorhandenes EFS für die Verwendung mit Ihrer Lambda-Funktion aus.

datalake-fs
arn:aws:elasticfilesystem:eu-central-1:196645549633:file-system/fs-f6a463ad
Owner: 196645549633 Throughput Mode: bursting

fs-f6a463ad ▼

Zugriffspunkt
Ein Zugriffspunkt, der für das Mounten eines Netzwerkdateisystems konfiguriert ist und in IAM integriert wird, um den Zugriff zu steuern.

/
arn:aws:elasticfilesystem:eu-central-1:196645549633:access-point/fsap-0765b720fe0b3ce00
POSIX uid: 0 POSIX gid: 0 Remote path: /

fsap-0765b720fe0b3ce00 ▼

Lokaler Mount-Pfad
Es werden nur absolute Pfade unterstützt.

/mnt/datalake

Danach kann in der Lambdafunktion unter dem Reiter [Code](#) mit dem Button [Hochladen](#) von die erstellte / vorhandene ZIP-Datei hochgeladen werden. Damit ist die Lambda funktion eingerichtet.

API Gateway

Als letztes muss ein API Gateway vom Typen [REST API](#) angelegt werden. Dort muss dann eine neue Ressource erstellt werden, welche den Pfad `/filesystem` an Anfang beinhaltet. Zusätzlich wird diese Ressource als Proxyressource definiert, dass heißt alle Subpfade führen zum gleichen Endpunkt. Alles nach `/filesystem` dient dann als Pfad zum speichern des Datensatzes.

Neue untergeordnete Ressource

Erstellen Sie anhand dieser Seite eine neue untergeordnete Ressource für Ihre -Ressource. 

Als [Proxy-Ressource konfigurieren](#)

☐ 

Ressourcenname*

filesystem

Ressourcenpfad*

/ filesystem

Mithilfe von Klammern können Sie Pfadparameter hinzufügen. Der Ressourcenpfad **{username}** stellt beispielsweise einen Pfadparameter mit der Bezeichnung "username" dar. Wenn Sie `/ {proxy+}` als Proxy-Ressource konfigurieren, werden alle Anforderungen für dessen Unterressourcen abgefangen. Dies funktioniert beispielsweise für GET-Anforderungen an `/foo`. Um Anforderungen an `/` zu verarbeiten, müssen Sie eine neue ANY-Methode zur Ressource `/` hinzufügen.

API Gateway CORS aktivieren

☐ 

Neue untergeordnete Ressource

Erstellen Sie anhand dieser Seite eine neue untergeordnete Ressource für Ihre -Ressource. 

Als [Proxy-Ressource konfigurieren](#)

☒ 

Ressourcenname*

proxy

Ressourcenpfad*

/filesystem/ {proxy+}

Mithilfe von Klammern können Sie Pfadparameter hinzufügen. Der Ressourcenpfad **{username}** stellt beispielsweise einen Pfadparameter mit der Bezeichnung "username" dar. Wenn Sie `/filesystem/ {proxy+}` als Proxy-Ressource konfigurieren, werden alle Anforderungen für dessen Unterressourcen abgefangen. Dies funktioniert beispielsweise für GET-Anforderungen an `/filesystem/foo`. Um Anforderungen an `/filesystem` zu verarbeiten, müssen Sie eine neue ANY-Methode zur Ressource `/filesystem` hinzufügen.

API Gateway CORS aktivieren

☐ 

Diese Proxy Ressource legt eine **ANY** Method an und muss dann mit dem erstellten Lambda verbunden werden.

/filesystem/{proxy+} - ANY - Setup

API Gateway konfiguriert Ihre ANY-Methode als Proxy-Integration. Proxy-Integrationen können mit HTTP-Endpunkten und Lambda-Funktionen Gateway sendet die gesamte Anforderung einschließlich Ressourcenpfad, Headern, Parametern von Abfragezeichenfolgen und Text an HTTP-E Lambda-Integrationen wendet API Gateway eine Standardzuweisung an, um alle Informationen aus der Anforderung zu senden. Antworten erfo Standardschnittstelle. Weitere Informationen finden Sie in der [Dokumentation](#).

Integrationstyp ☒ Lambda-Funktions-Proxy 

☐ HTTP-Proxy 

☐ VPC-Link 

Lambda-Region

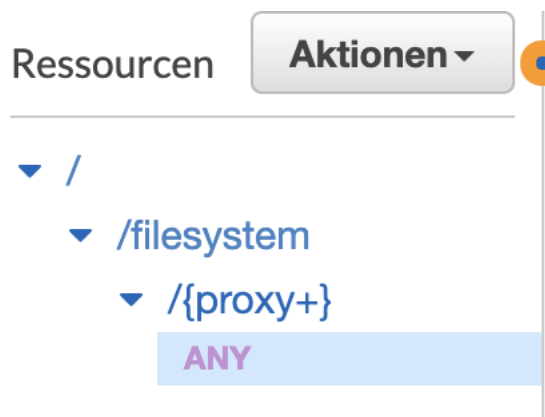
eu-central-1 

Lambda-Funktion

datalake-ex 

Standardzeitüberschreitung verwenden ☒ 

Am Ende sollte der Pfadbaum so aussehen:



Das API Gateway kann dann z.B. mit der Stufe `test` bereitgestellt werden.

Testen

Folgender Curl Befehl schreibt im unter `/test/test.json` eine Datei mit dem JSON

```
1 {  
2   "test": 123  
3 }
```

Es sollte ein Response mit dem Statuscode 200 und dem Text `File written to filesystem.` zurück kommen.

```
1 curl -X POST https://mpb01j9z52.execute-api.eu-central-1.amazonaws.com/  
   test/filesystem/test/test.json -d {"test":123}
```

Dieser Befehl lädt die gerade gespeicherten Daten.

```
1 curl -X GET https://mpf01i9z12.execute-api.eu-central-1.amazonaws.com/  
   test/filesystem/test/test.json
```

Probleme

Zugriffspfad Wurzelverzeichnis

Wenn das Wurzelverzeichnis als Zugriffspfad bei einem EFS verwendet wird müssen die Permissions und Users so gewählt werden, dass es sich um root handelt. Das liegt daran, dass das Root-Verzeichnis vom root nutzer erstellt wurde.

Subpfade abfangen

Subpfade können nur mit einem `{proxy+}` abgefangen werden. Dies nimmt einen aber die Möglichkeit Lambdas für einzelne Request-Methoden zu schreiben, da immer eine `ANY` Methode angelegt wird.

Zonen, Regionen und VPC

Die Lambda muss im gleichen VPC / in den gleichen Regionen wie das EFS liegen. Sollte das EFS nur für die Zone A existieren, die Lambda allerdings in allen drei Zonen dann kann diese das Dateisystem nicht mounten. Das liegt daran, dass die Lambda theoretisch in Zone B gestartet werden kann, wo sie kein Zugriff auf das EFS hätte.

Erweiterungspotential

Absichern der API mittels Cognito

Dadurch wird sichergestellt, dass nur verifizierte Nutzer Zugriff auf die Daten haben.

Anbinden weiterer Lambdas und Speichermöglichkeiten

Nach dem Pfad oder intelligent in den Lambdas könnte erkannt werden ob es sich bei den Daten z.B. um strukturierte Datensätze handelt. Dadurch könnte der Speicherort dynamisch gewählt werden.

JSON -> DynamoDB

Tabellen Daten -> RDS

große Dateien -> S3

kleine Dateien -> ES