Présentation du dossier Projet

Pour la certification Rncp-36061
TP - Administrateur système DevOps

Bertrand chatelet

Plan, de la présentation du projet

→ Quels choix technologiques, quelles actions ?

pour les 3 thèmes de la certification dans ce projet

-Automatiser le déploiement d'une infrastructure dans le cloud

-Déployer en continu une application

-Superviser les services déployés

Sommaire

- Présentation du Projet
- Choix: du Cloud, du provisionning, premiers déploiements de MongoDB, d'un Cluster Spark sur Aws avec Terraform
- Vers la définition d'un schéma d'architecture lié au stockage des données pour le dataware : tests ETL
- Sécurisation: security groups, Iam, credentials et clefs accès
- Ci/Cd : Solutions possibles, GitHub Actions
- Gestion logs et supervision : OpenSearch/ CloudWatch, Prometheus
- Conclusion

Présentation du projet

GoldenLine est une société commerciale avec 3 millions de clients.

L'analyse de ses données clientèle volumineuses (Big data) est faite

par le prestataire *DataPro.*

En tant que devops de DataPro, pour ce projet, on demande de mettre en place une *architecture Cloud* pour:

→ répondre aux besoins de l'équipe Analytics

-en déployant avec l'**Iac** (Infrastucture as code)
-en installant une solution de **CI/CD** : livraison, intégration continues.
-en **supervisant** la production

Le CIO impose les choix suivants:
→ un cluster de serveurs **Apache Spark**, et une bd **MongoDb.**

Choix des environnements de travail

Le choix d'un *fournisseur de Cloud*, a été guidé par la possibilité de se familiariser avec l'environnement qui sera le plus demandé dans les offres d'emploi → **Aws** (part de marché par exemple 3 fois plus importante que Google par exemple).
Ce dernier choix va conditionner aussi le **schéma d'architecture**.

Les outils de *provisioning* ont été examinés. **Terraform** est retenu.
Les autres solutions les plus répandues ont des inconvénients :

-Ansible est plus difficile d'emploi (playbooks, avec yaml). Pas de notion de state (traces des modifications manuelles de l'infrastrucure). Toutefois intérêt des modules avec Ansible galaxy.

-Cloud Formation pas de state.

Premiers déploiements automatisés

- **Apache-Spark,** est installé avec le service **Emr-Aws,** et l'aide de Terraform : 2 serveurs esclave (EC2), 1 serveur maître.
  La configuration utilise des modules, affectés à domaine de la configuration : par exemple module « eip", pour une adresse ip élastique. On a aussi fait un déploiement sans utiliser le service Emr.

- **Base MongoDb,** est installée dans un premier temps avec serveur EC2 et un script d'installation : liste d'instructions « inline », d'un provisioner "remote-exec" de Terraform. On a aussi fait un déploiement à l'aide du service **Ecs-Aws**, qui installe des containers.

Stockage des données.

Le **stockage** des données volumineuses pour le bigdata est spécifique (temps accès).
Les solutions les plus répandues reposent sur:
-les bases de données Nosql: ***MongoDb***,*Cassendra*
- le système de fichiers distribués Hadoop *HDFS, Habase* bdd distribuée, non relationnelle, au-dessus de HDFS
-S3 stockage d'objets de Aws

Comme le **service Emr de Aws** propose l'installation de Hadoop (avec Hdfs) et Spark à la fois,

→ le choix du stockage des données sera: **s3, Hdfs ou MongoDb**.

Architecture proposée par Aws

Schéma proposé par Aws :

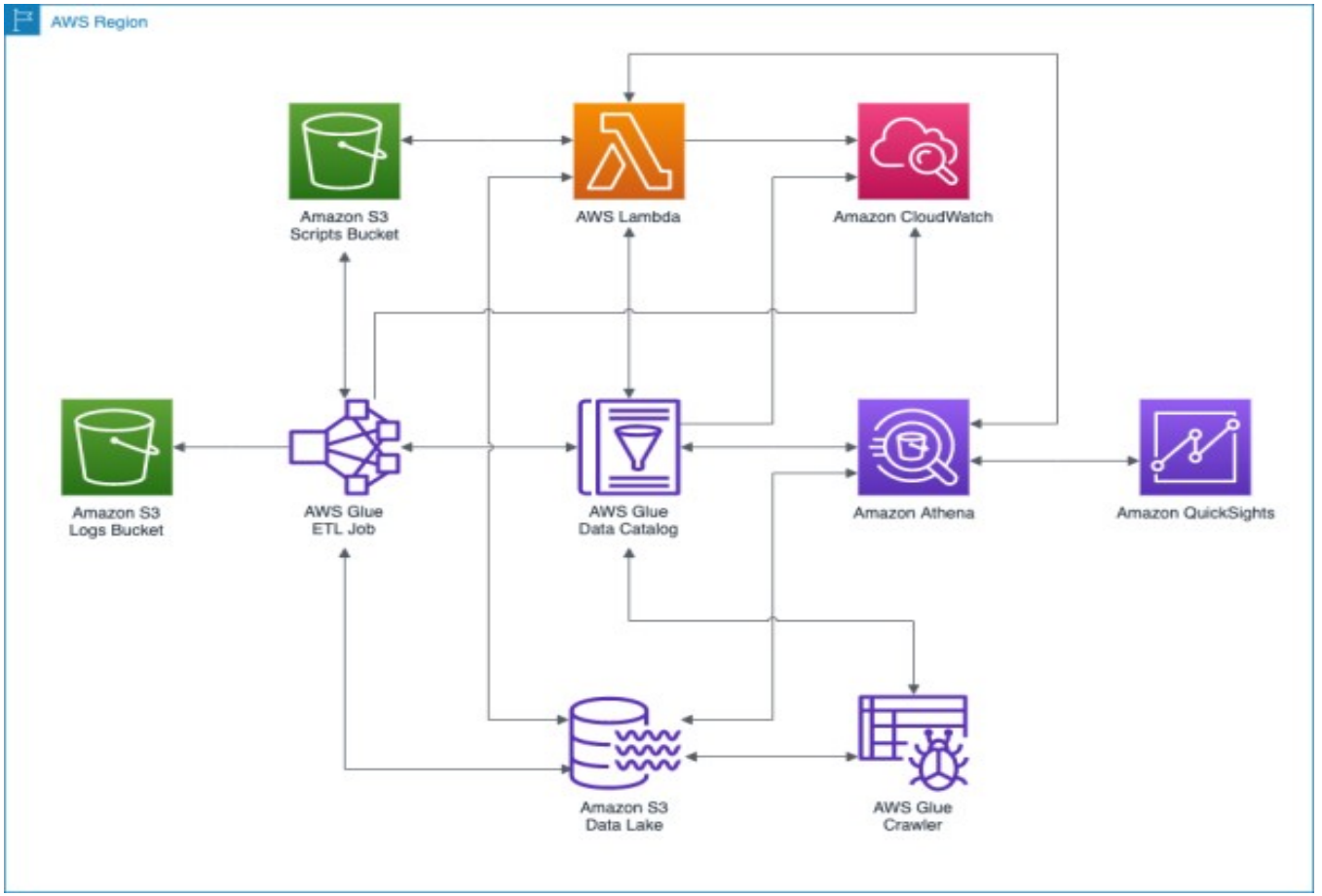
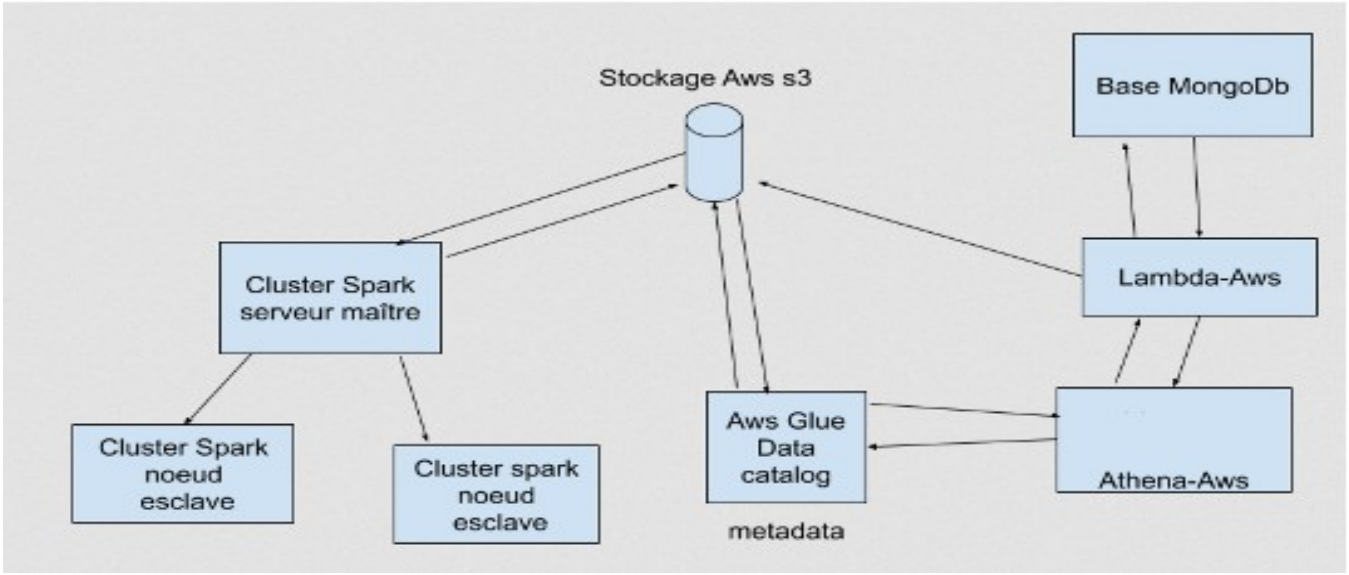


Schéma d’architecture

Proposition



3 Tests d'intégration de données Csv ->Architecture

Fichier csv,(400 Meg), de 6 millions de lignes de paiement extrait de Kaggle, placé sur un bucket s3

-import-csv.py, serveur EC2 → base mongodb: qq minutes si csv fait 4 Meg, plante le serveur avec 400 Meg

-"**spark-submit** .. script.py": Csv → Fichiers parquets sur s3 en 3 minutes

-**Service Glue**: Csv →Glue Catalog
-csv-to-json.py: Glue Catalog →json sur s3, qq minutes
-json-to-mongo.py: json de s3 → MongoDb, qq minutes
Schéma de traitement préconisé par Aws →Intéressant de déployer Glue dans l'**architecture** (et Athena, si SQL avec le bigdata).

ETL

Les types de fichiers compatibles avec Spark
-format binaire: orc, parquet (temps accès rapides)
-ascii: csv, json, text

Les données étant le plus souvent hétérogènes, on propose de compléter l'architecture avec:

-**Glue-Aws (**Pour création Data-calog), pour des insertions efficaces de données dans MongoDb
-Éventuellement : **Athena-Aws,** Glue si on voulait utiliser des outils analytiques basés sur Sql.
-**Lambda-Aws:** peut être trop complexe pour un premier niveau d'utilisation.
-Spark cependant peut lui même être utilisé faire des conversions vers des formats optimisés pour le bigdata (parquet, orc).

Sécuriser

- **Security groups**: des groupes de sécurité permettent de déclarer des ports ouverts sur des serveurs EC2-Aws. Par défaut tous les ports sont fermés.
- **Credentials Aws**: l'association Access_key/secret_access_key générée depuis la console permet la connexion à Aws depuis Terraform ou GitHub Actions.
- **Clefs d'accès** : paire clef pub/privée pour une connexion sur chaque serveur (copié sur chaque image générique de serveur EC2)

  Autres services Aws : Iam, Vpc, subnet, chiffrement sur S3, CloudTrail, CloudWatch..

CI/CD, GitHub Actions

- Diversité des solutions

  -Jenkins, Gitlab, **GitHubActions**, CircleCI ..
  -Pour les fournisseurs de cloud : Azure Devops, Aws-CodePipeline, Google CodeBuild..

- Mise en œuvre de GitHub Actions
  -Assez proche de Git

  -CI : Peut être mis en œuvre assez facilement par exemple en Python avec des outils de tests unitaires Pytest, d'intégration Selenium..

  -Mise en œuvre de CD (continuous delivery) avec un des projets d'automation Terraform, présenté.

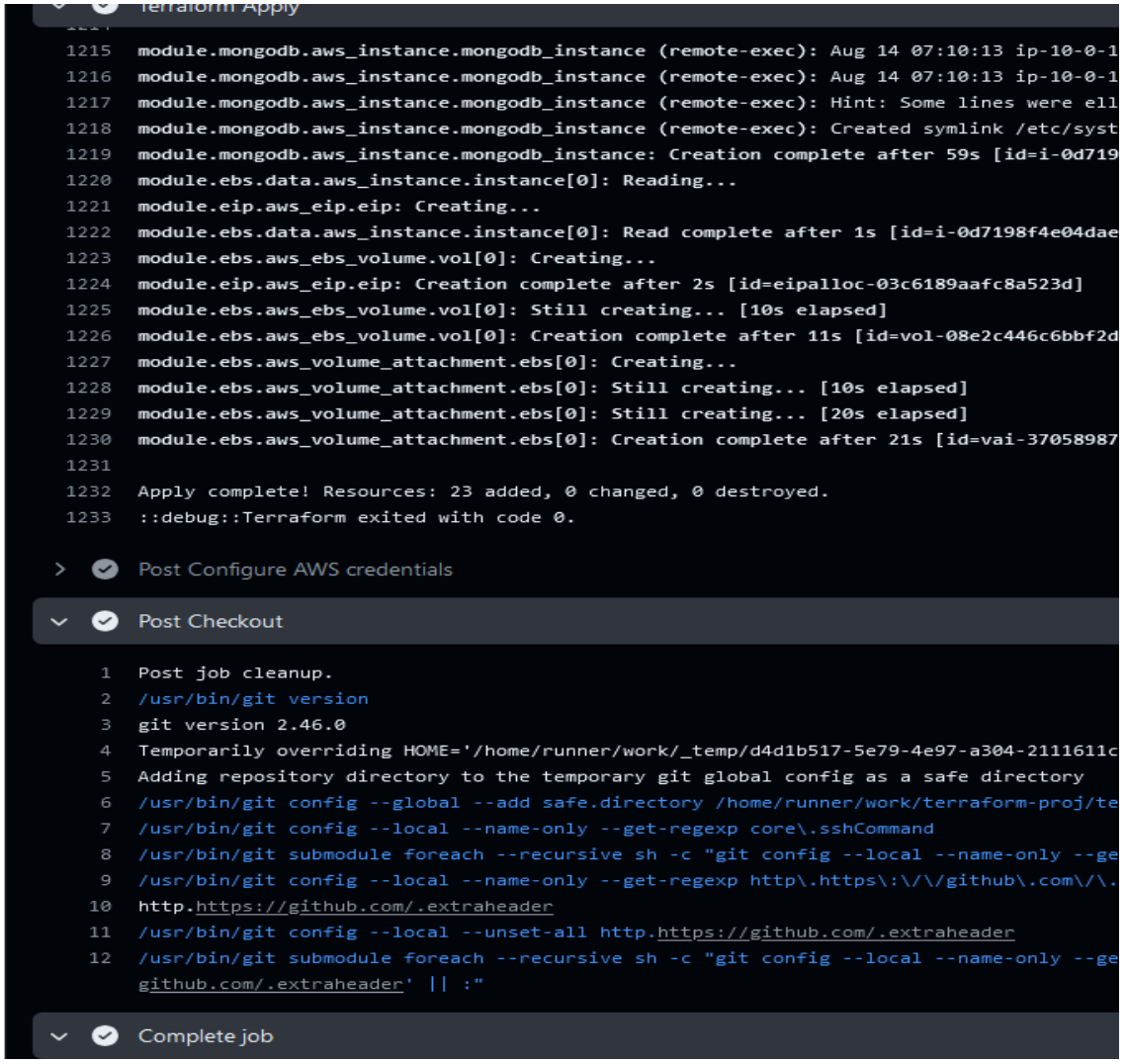
CD, avec GitHub Actions

Terraform

Run details

Usage

Workflow file



Stats de services

- Nombreuses solutions possibles
  -propriétaires : Datalog, Splunk
  -open-sources : Logstash (Elastic search)
  On a testé **OpenSearch-Aws** : service facilitant la mise en place d'Elastic Search.

- Statistiques du cluster Spark, proposées par Hadoop

Supervision

- Solutions open-source : Prometheus, Nagios, Inciga, Centreon

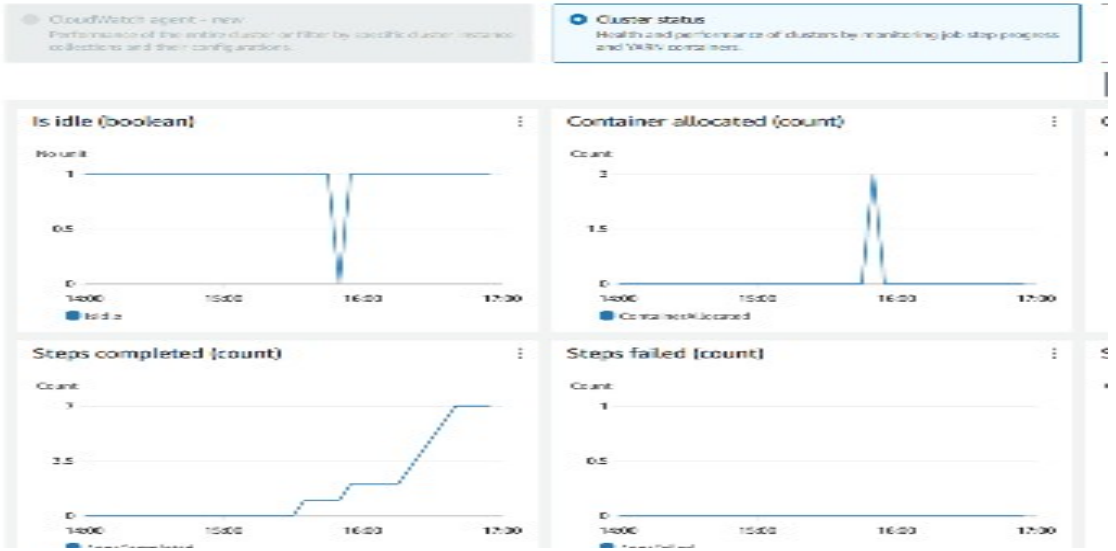
- Ou propriétaires : CloudWatch (Aws), Micosoft Cloud monitoring, Datadog ..

→ Préconisation pour **Prometheus/Grafana** : métriques très détaillées, facilité d'installations sur Aws

→ Mise en œuvre de CloudWatch

Exemple de supervision avec CloudWatch

- Possibilité de créer des alarmes (CloudWatch Alarm), et de les lier au service SNS-Aws (emails).
- Suivi du cluster Spark :



Conclusion

Le projet a permis d’aborder les différentes notions liées à, la certification Devops:

-gestion du Cloud (Aws en particulier)
-déploiement automatisé (avec Terraform)
-les outils de production (supervision,sécurité)
-Le CI/CD

Dans le contexte particulier du Bigdata: volume et hétérogénéité des données.