Introduction

The objective is to extract and transform incoming data from a third party in XML to json. When the file is created (i.e. dumped) in the S3 bucket, the S3 service sends an event to EventBridge, and the ETL process begins.

Ideally, the application should be run on AWS lambda (see lambda version in projects) because of the data size (see lambda version [i.e. xml_parser_service_lambda]). However, per the requirement, the application is wrapped in docker and expected to be run on an EKS cluster.

The pipeline is reactive, and the reason is for other teams to get real-time data when they arrive.

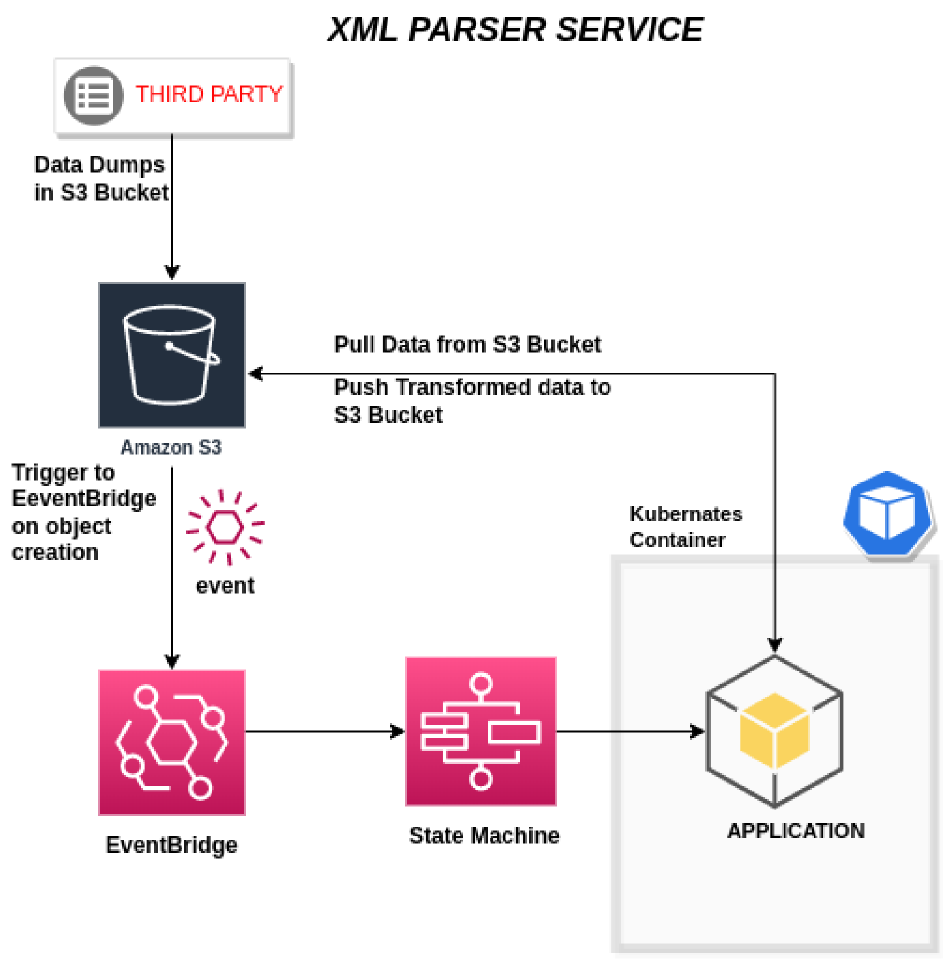
In this pipeline, we do not employ data quality checks; therefore, we are not tracking the files processed or caching tags of IDs (future implementation). Further, it is assumed; the Third-party provider built checkers that guarantee the following;

- eight files are dumped every day at various intervals as agreed upon.
- They perform data quality checks before dumps.

Upon the S3 bucket receiving a file, we trigger an event (containing the bucket and key) to EventBridge; the EventBridge starts a state machine, and the state machine starts a job on the cluster.

In the cluster, the data file dumped is loaded from S3, transformed and pushed back to another bucket.

Flow Diagram



Schema

- To receive an event either as an environment variable or an argument.
- Event

  Eg. of events

```
{"Records": [{"eventVersion": "2.0", "eventSource":
"aws:s3", "awsRegion": "us-east-1", "eventTime":
"1970-01-01T00:00:00.000Z", "eventName":
"ObjectCreated:Put", "userIdentity": {"principalId":
"EXAMPLE"}, "requestParameters": {"sourceIPAddress":
"127.0.0.1"}, "responseElements": {"x-amz-request-id":
"EXAMPLE123456789", "x-amz-id-2":
"EXAMPLE123/5678abcdefghijklambdaisawesome/mnopqrstuvwxyzABC
DEFGH"}, "s3": {"s3SchemaVersion": "1.0", "configurationId":
"testConfigRule", "bucket": {"name": "test-emile-dev",
"ownerIdentity": {"principalId": "EXAMPLE"}, "arn":
"arn:aws:s3:::example-bucket"}, "object": {"key":
```

```
"sample.xml", "size": 1024, "eTag":
"0123456789abcdef0123456789abcdef", "sequencer":
"0A1B2C3D4E5F678901"}}}]}
```

- Returns - N/A

Services

- Implement S3 event notification on the S3 bucket.
- Define an EventBridge rule
- Define a State Machine using ASL
- Implement an EKS cluster and endpoint.

Observability and monitoring

- The application contains logs; logs are saved on AWS Cloudwatch.
- We can employ tools like Datadog in the future for monitoring and observability.