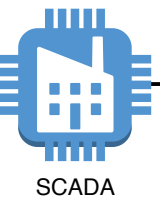
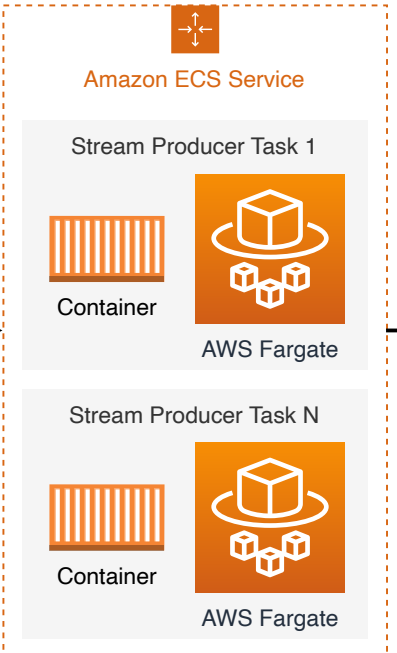


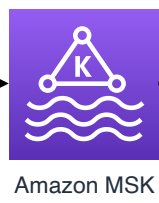
Data Pipeline with Amazon EMR Serverless and Amazon MSK



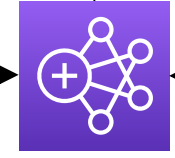
Read Stream



Publishes messages



Consumes messages



EMR reads Spark job scripts from Bootstrap bucket



S3 EMR Log



CloudWatch



S3 Bootstrap



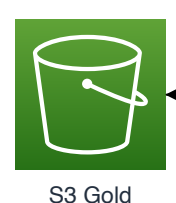
CloudFormation



S3 Silver



CloudWatch Dashboard



S3 Gold



QueryResults

EMR Serverless CloudWatch Dashboard

The CloudWatch Dashboard provides an overview of **pre-initialized capacity vs. OnDemand** as well as drill-down metrics for CPU, memory, and disk usage for Spark Drivers and Executors.

Pre-initialized capacity is an optional feature of EMR Serverless that keeps driver and workers pre-initialized and ready to respond in seconds and this dashboard can help understand if pre-initialized capacity is being used effectively.

In addition, user can see job-level metrics for the state of jobs on a per-application basis

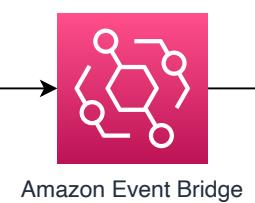


Systems Manager Parameter Store

Parameter Store is used by most AWS services to retrieve information about the following parameters: 'kafka-servers', 'kafka-topics', 'sns-topic', 'emr-app-id', S3 bucket locations.

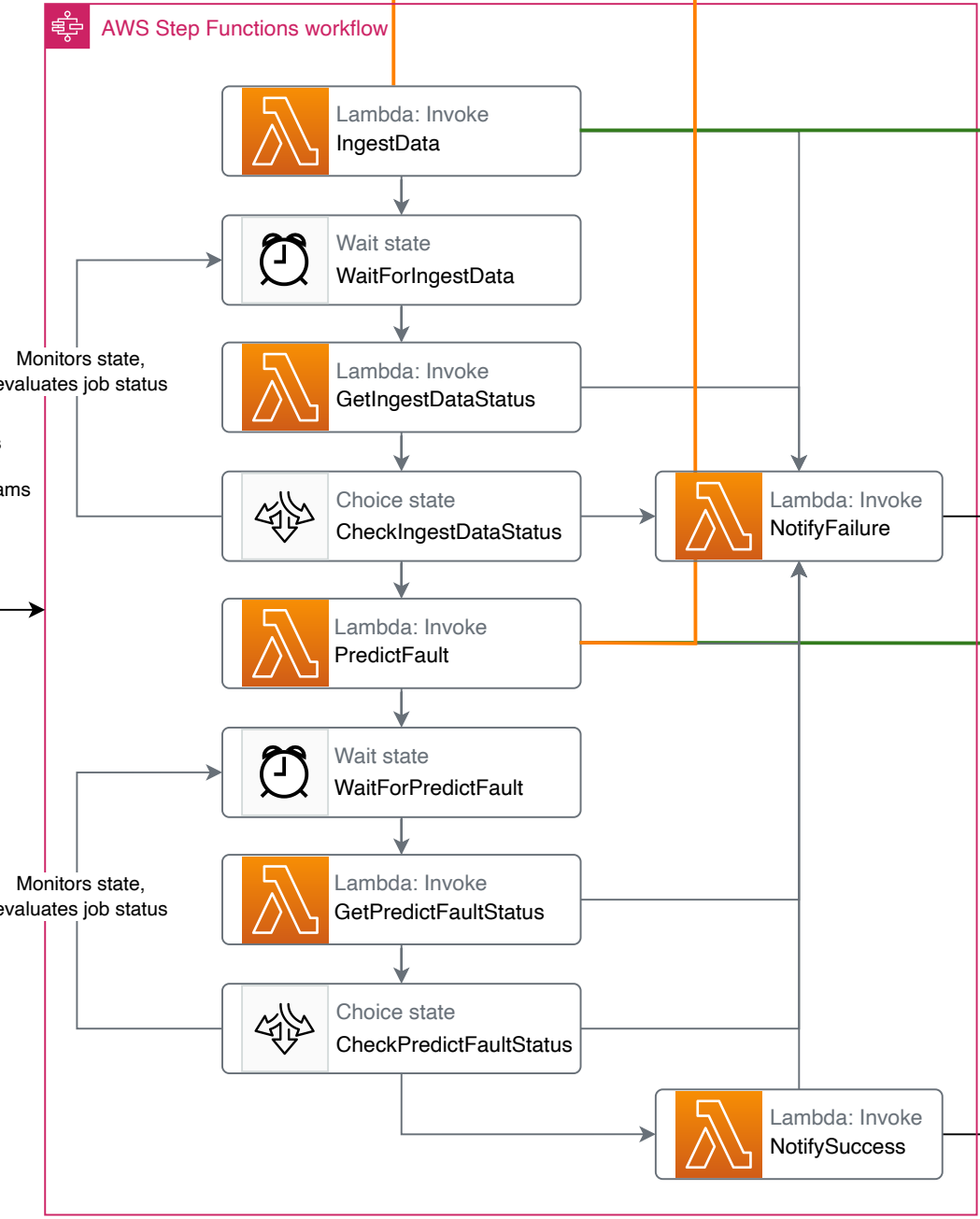


IAM Role



Amazon Event Bridge

Schedules execution + passes params



StepFunctions workflow

The Step Functions state machine has 10 states, each performing a specific task or decision.

The **IngestData** task state submits an EMR job to ingest data from a Kafka cluster and retries up to 6 times if errors occur.

WaitForIngestData pauses the state machine to avoid polling the EMR job status too often.

The state machine then uses **GetIngestDataStatus** and **CheckIngestDataStatus** to monitor job state and evaluate its status.

A similar pattern of 4 tasks applies to the next step, started with the **PredictFault** task.

Finally, **NotifySuccess** and **NotifyFailure** states send SNS notifications based on the success or failure of the pipeline.

Runs 'ingest_data.py' job from S3 bootstrap bucket on EMR Serverless cluster

Runs 'predict_fault.py' Spark job, loads intermediate file, feeds into pre-trained LeNet, gets its prediction classes

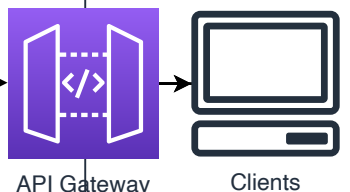
EMR stores and loads intermediate file to/from S3 Silver bucket

Writes intermediate results

Stores result in S3 gold bucket

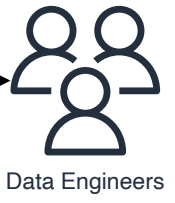
Notifies in case of any failure

Notifies on successful completion of the Spark jobs



API Gateway

Clients



Data Engineers

CloudFormation

We use AWS CLI and CloudFormation to create some of the infrastructure for the pipeline: S3 buckets, parameters in Systems Manager Parameter Store, tasks for Amazon EventBridge

CloudFormation also deploys CloudWatch Dashboard for Spark applications on EMR Serverless.

Hidden resources

Multiple required support resources are not shown on the diagram such as Amazon Virtual Private Clouds (VPCs), private and public subnets, multiple AWS Identity and Access Management (IAM) Roles and Policies, Security Groups and Route Tables, and a VPC Gateway Endpoint for S3