



Immersion Day

Serverless Data Lake with AWS

December, 2018

Table of Contents

Table of Contents	2
Revision History.....	3
Disclaimer	3
Overview.....	4
Lab Solution Architecture	4
Lab Pre-Requisites	5
Expected Costs	5
Section I - Data Ingestion & Storage Layers.....	6
Lab 1.1 - Create simulated data in real-time data with KDG	6
Lab 1.2 - Create Kinesis Firehose Delivery Stream to load Stream data on S3	11
Summary & Next Steps	18
Deleting Lab Resources	18
Appendix A – Lab Prerequisites.....	19
Appendix A.1 - User Permissions:	19
Appendix A.2 - Service Role Permissions: CloudFormation Template for IAM Roles	21
References	36

Revision History

Revision	Date	By	Change Log
PA6	03-Dec-2018	@akirmak	First 3 labs reer reviewed by @baya
PA7	13-Dec-2018	@akirmak	Initial draft, ready for internal joint review.
PA8	14-Dec-2018	@akirmak	Joint reviewed by @halilb and @serdarn
PA9	16-Dec-2018	@akirmak	Joint review feedback incorporated. Fixed an error in Lab 2.4 instructions. Shortened Lab 3.1

Disclaimer

The work is provided “as is” without warranties or conditions of any kind, either express or implied, including warranties.

Overview

A **data lake** is a centralized repository for both structured and unstructured data where you store data as-is, in open-source file formats to enable direct analytics. Implementing a Data Lake architecture requires a broad set of tools and technologies to serve an increasingly diverse set of applications and use cases. Refer to <https://aws.amazon.com/tr/big-data/datalakes-and-analytics/what-is-a-data-lake/>

Lambda architecture (should not be confused with the AWS Lambda compute service.) is a data-processing design pattern to handle massive quantities of data and integrate batch and real-time processing within a single framework. Various AWS services are used as building blocks of this unified architectural pattern that unifies stream (real-time) and batch processing. In this lab, you will focus on the batch processing layer. Refer to whitepaper “Lambda Architecture for Batch and Stream Processing (October 2018)” <https://d0.awsstatic.com/whitepapers/lambda-architecture-on-for-batch-aws.pdf>

Lab Solution Architecture

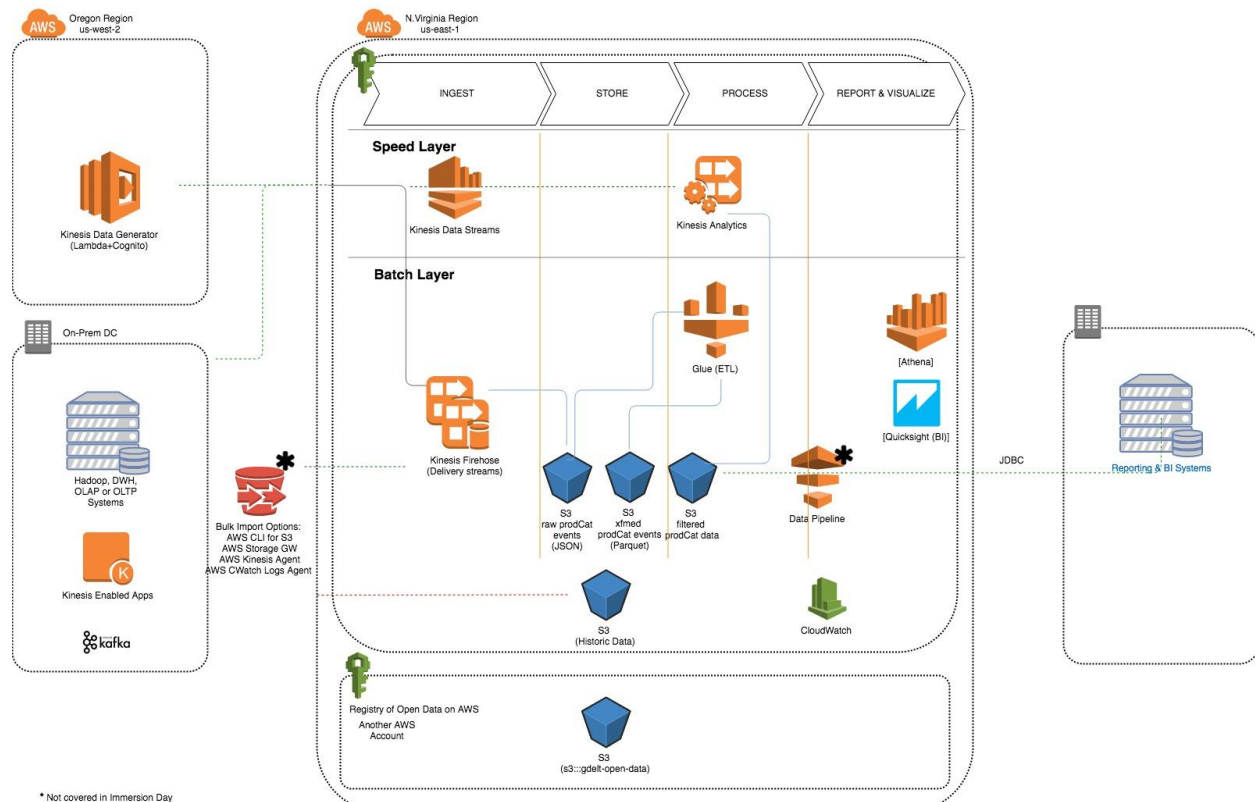


Figure 1: Serverless Data Lake Lab Solution Architecture

This lab guide is prepared to assist you ingest, store, transform, create insights on unstructured data using AWS serverless services. Most of the demos make use of AWS Console, however all the labs can be automated via Cloudformation templates, AWS CLI or AWS API.

Lab Pre-Requisites

In order to complete this immersion day lab, please follow the preparation steps in the Serverless Data LakeDay Lab Preparation Guide (provided to you prior to the event).

Expected Costs

- You are responsible for the cost of the AWS services used while running this lab. As of the date of publication, the baseline cost for running this solution as-is should be around:
 - Kinesis Firehose: < 1\$
 - S3 < 1\$
- Cost Management advice: Whenever you are creating a cloud resource, tag it. Try setting following tag fields during the labs:
 - project: serverlessdatalake
 - costcenter: awsimmersionday

Section I - Data Ingestion & Storage Layers

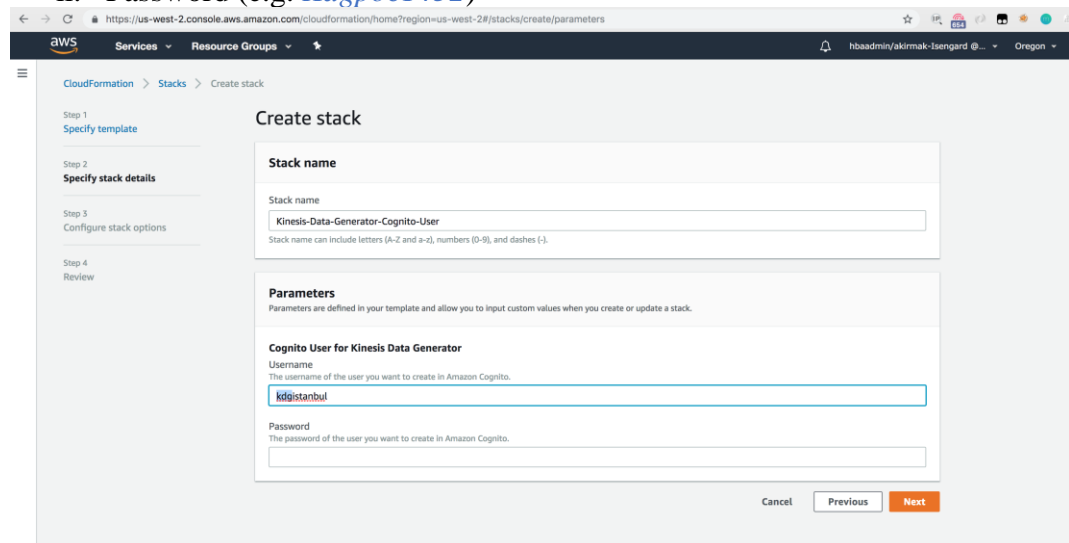
Lab 1.1 - Create simulated data in real-time data with KDG

In lab 1, you will deploy an open-source serverless real-time data generator tool to simulate data in real-time and ingest into AWS Kinesis Firehose. We will simulate product catalogue data with random values. The data will be used in upcoming labs to demonstrate storage, processing, reporting and visualization.

The tool is called Kinesis Data Generator (KDG), and is available on GitHub under:
<https://github.com/aws-labs/amazon-kinesis-data-generator>

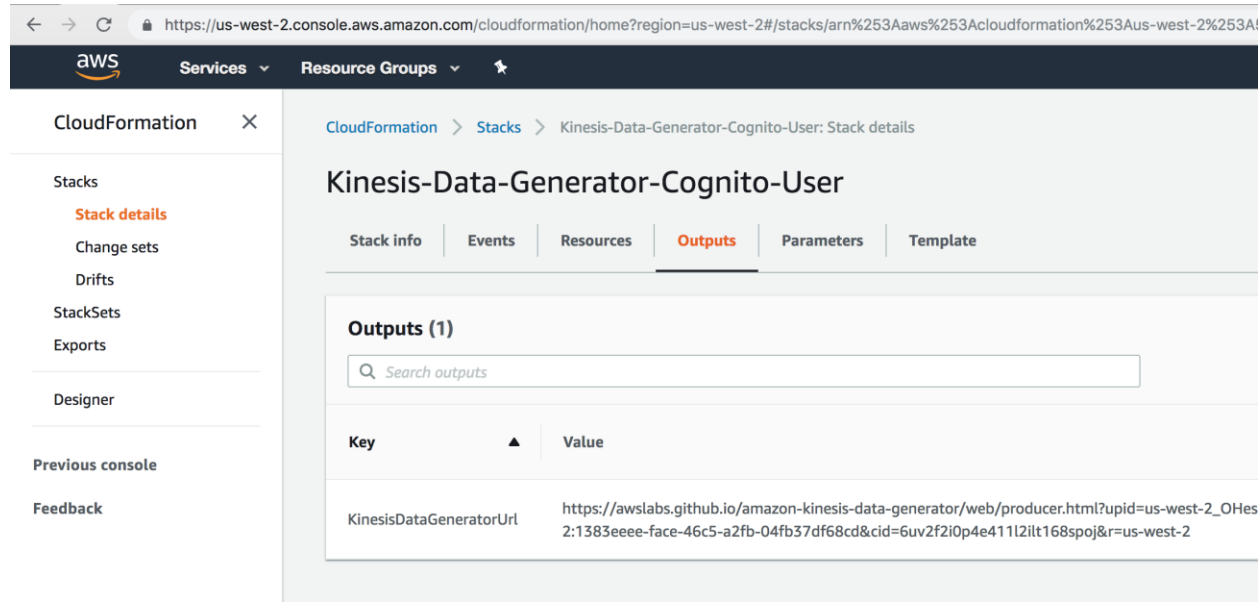
KDG is a UI that simplifies how you send test data to Amazon Kinesis Streams or Amazon Kinesis Firehose. Using the Amazon Kinesis Data Generator, you can create templates for your data, create random values to use for your data, and save the templates for future use.

1. Follow the KDG Guide on <https://awslabs.github.io/amazon-kinesis-data-generator/web/help.html> to install and configure the KDG on your AWS account. Because the project is a collection of static HTML and JavaScript, No servers are required.
 - a. Deploy Cloudformation Template from the link in the install guide.
 - i. Username: *kdgpoc*
 - ii. Password (e.g. *Kdgpoc1432*)



- b. The stack creates a user password in a User DataBase (called User Pool) using the Amazon Cognito service.
 - c. Select output tab of the CloudFormation Stack, which contains a URL generated for your KDG. Copy this URL to your browser and login with the user/password you created during CloudFormation stack creation.

Serverless Data Lake Immersion Day
Tame Your Big Data with AWS Kinesis Firehose, S3, Glue



2. Login to KDG, and select your region.
3. At this stage, there's no stream we can push data towards. Let's define the structure of our simulated data using the parametrized template. Copy and paste it to the template tabs. Template to use:

```
{
  "productName": "{{commerce.productName}}",
  "color": "{{commerce.color}}",
  "department": "{{commerce.department}}",
  "product": "{{commerce.product}}",
  "imageUrl": "{{image.imageUrl}}",
  "dateSoldSince": "{{date.past}}",
  "dateSoldUntil": "{{date.future}}",
  "price": {{random.number(
    {
      "min":10,
      "max":150
    }
  )}},
  "campaign": "{{random.arrayElement(
    ["BlackFriday","10Percent","NONE"]
  )}}"
}
```

This template generates data such as (each record will be random).

```
{  
  "productName": "Generic Granite Ball",  
  "color": "teal",  
  "department": "Beauty",  
  "product": "Cheese",  
  "imageUrl": "http://lorempixel.com/640/480",  
  "dateSoldSince": "Mon Dec 04 2017 19:19:50 GMT+0300 (GMT+03:00)",  
  "dateSoldUntil": "Fri May 17 2019 08:56:56 GMT+0300 (GMT+03:00)",  
  "price": 64,  
  "campaign": "10Percent"  
}
```


Serverless Data Lake Immersion Day

Tame Your Big Data with AWS Kinesis Firehose, S3, Glue


https://awslabs.github.io/amazon-kinesis-data-generator/web/producer.html?upid=us-west-2_OHesoJbFS&ipid=us-west-2:1383eeee-face-46c5-a2fb-0


Amazon Kinesis Data Generator

Region: us-east-1

Stream/delivery stream: tamebda-rta-kinesisfh-prodcat

Records per second: 100

Compress Records  ☐

Record template 

stockExchange-dynamic Template 2 Taming-BDA-Immersion Template 4 Template 5

Taming-BDA-Immersion

```
{
  "productName" : "{{commerce.productName}}",
  "color" : "{{commerce.color}}",
  "department" : "{{commerce.department}}",
  "product" : "{{commerce.product}}",
  "imageUrl" : "{{image.imageUrl}}",
  "dateSoldSince" : "{{date.past}}",
  "dateSoldUntil" : "{{date.future}}",
  "price": {{random.number(
    {
      "min":10,
      "max":150
    }
  )}},
  "campaign": "{{random.arrayElement(
    [ "BlackFriday", "10Percent", "NONE" ]
  )}}"
}
```

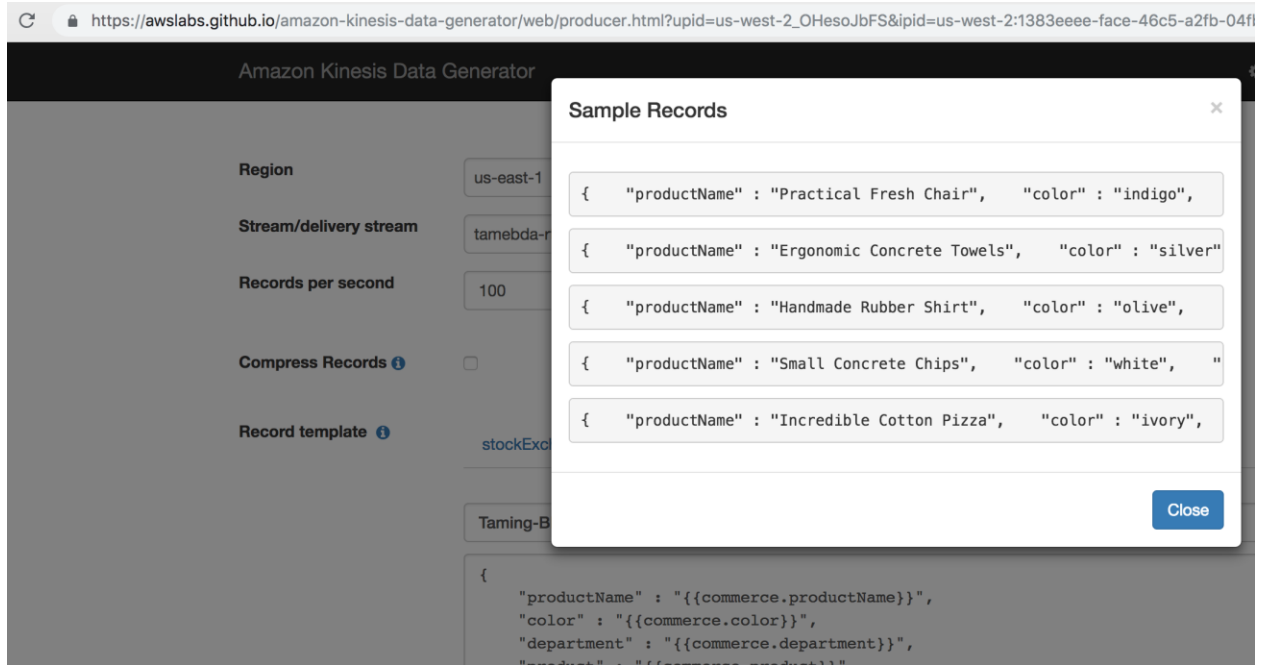
Send data Test template

Hint: KDG makes use of an open-source toolkit called Faker.js (available in GitHub: <https://github.com/marak/Faker.js/>). You can customize the template as you wish. Check Faker.js documentation for details.

4. Press *Test Template* to generate a few records.

Serverless Data Lake Immersion Day

Tame Your Big Data with AWS Kinesis Firehose, S3, Glue



You have successfully created KDG, and are ready to ingest data to the AWS Kinesis Firehose. Please proceed to the next lab.

Lab 1.2 - Create Kinesis Firehose Delivery Stream to load Stream data on S3

In this lab, you will create a Firehose delivery stream from the AWS Management Console, configure it with a few clicks to store incoming stream data into S3, and start sending data to the stream Kinesis Data Generator (created in Lab 1) as data source.

Amazon Kinesis Firehose is a serverless service used to reliably load streaming data into data stores and analytics tools. It can capture, transform, and load streaming data into Amazon S3, Amazon Redshift, Amazon Elasticsearch Service, and 3rd Party services (e.g. Splunk), enabling near real-time analytics.

Amazon Kinesis Firehose benefits are:

- no explicit provisioning or scaling,
 - extremely simple APIs and console,
 - guaranteed data capture and reliable delivery.
 - It is a fully managed service that automatically scales to match the throughput of your data and requires no ongoing administration.
 - It can also batch, compress, transform, and encrypt the data before loading it, minimizing the amount of storage used at the destination and increasing security.
1. Create a bucket with exactly the naming convention and press Create: bucket name: (name: <**your initials**>-**tame-bda-immersion**. For instance **fs-tame-bda-immersion** for Frank Sinatra) (Important, the CloudFormation template creates IAM policies according to a naming convention for simplicity. Therefore, please use the exact naming convention).

The screenshot shows the AWS S3 'Create bucket' wizard. The first step, 'Name and region', is selected. The 'Bucket name' field is populated with 'fs-tame-bda-immersion'. The 'Region' dropdown is set to 'US East (N. Virginia)'. Below these fields, there is a section for 'Copy settings from an existing bucket' with a dropdown menu showing 'Select bucket (optional)' and '74 Buckets'. At the bottom left, there is a 'Create' button.

2. Create a Firehose delivery stream and configure as follows:
 - a. Stream name: **tamebda-rta-kinesisfh-prodcat**
 - b. Source: **direct put**
 - c. Transformation: **disabled**
 - d. Conversion: **disabled**
 - e. Destination: S3
 - i. (name: *<your initials>*-**tame-bda-immersion**. For instance **hba-tame-bda-immersion**)
 - ii. prefix: **raw/** (Important: Don't forget the trailing "/" character. It shall be raw/ and not raw)
 - f. buffer: **1 MB, 60 seconds**
 - g. compression: **gzip**

Serverless Data Lake Immersion Day
Tame Your Big Data with AWS Kinesis Firehose, S3, Glue

← → ↻ https://console.aws.amazon.com/firehose/home?region=us-east-1#/wizard/nameAndSource

aws Services Resource Groups

Kinesis Firehose - Create delivery stream

Step 1: Name and source

Step 2: Process records

Step 3: Choose destination

Step 4: Configure settings

Step 5: Review

New delivery stream

Delivery streams load data, automatically and continuously, to the destinations that you specify. Kinesis Firehose resources are not covered under the [AWS Free Tier](#), and **usage-based charges apply**. For more information, see [Kinesis Firehose pricing](#).

Delivery stream name*

Acceptable characters are uppercase and lowercase letters, numbers, underscores, hyphens, and periods.

Choose source

Choose how you would prefer to send records to the delivery stream.

Firehose data flow overview

```
graph LR; Source[Source] --> Firehose[Firehose delivery stream]; subgraph Firehose; SourceRecords[Source records]; ProcessedRecords[Processed records]; end; Firehose --> Destination[Destination];
```

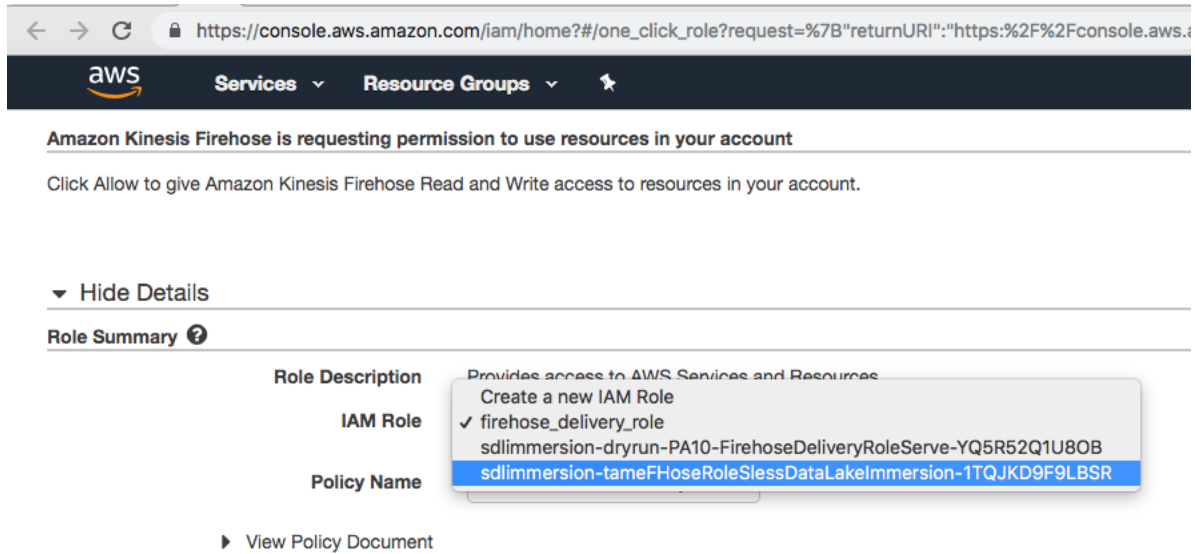
----- Optional

Source* ☒ Direct PUT or other sources
Choose this option to send records directly to the delivery stream, or to send records from AWS IoT, CloudWatch Logs, or CloudWatch Events.

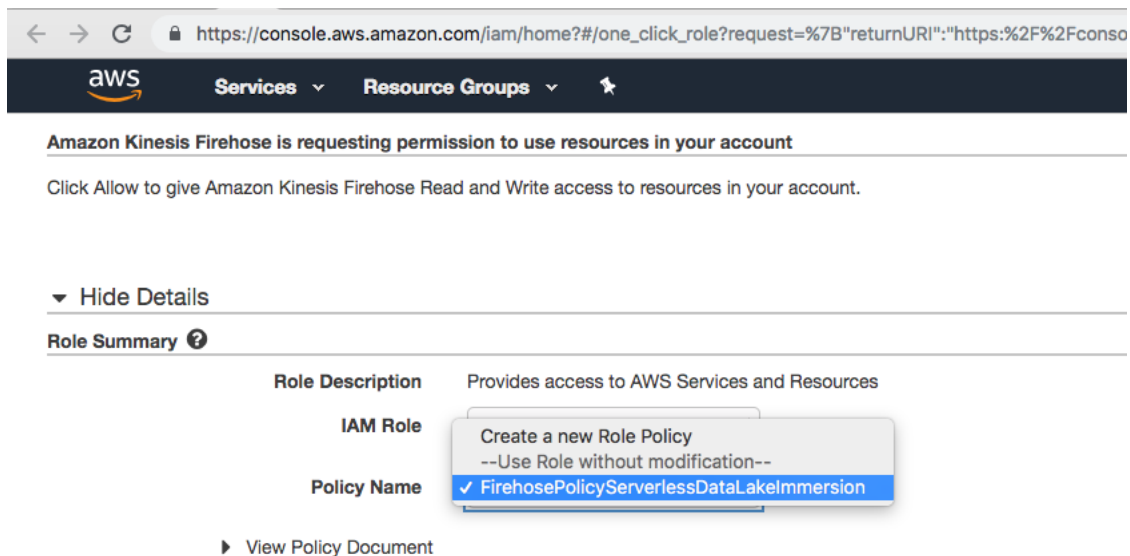
☐ Kinesis stream

- h. IAM:
- IAM role: Select the role created by the CloudFormation template (AWS Console -> CloudFormation -> Stacks -> sldimmersion-> Outputs. The IAM role has a name like “**sdlimmersion-tameFHoseRoleSlessDataLakeImmersion-<UNIQUE ID>**”

Serverless Data Lake Immersion Day
Tame Your Big Data with AWS Kinesis Firehose, S3, Glue

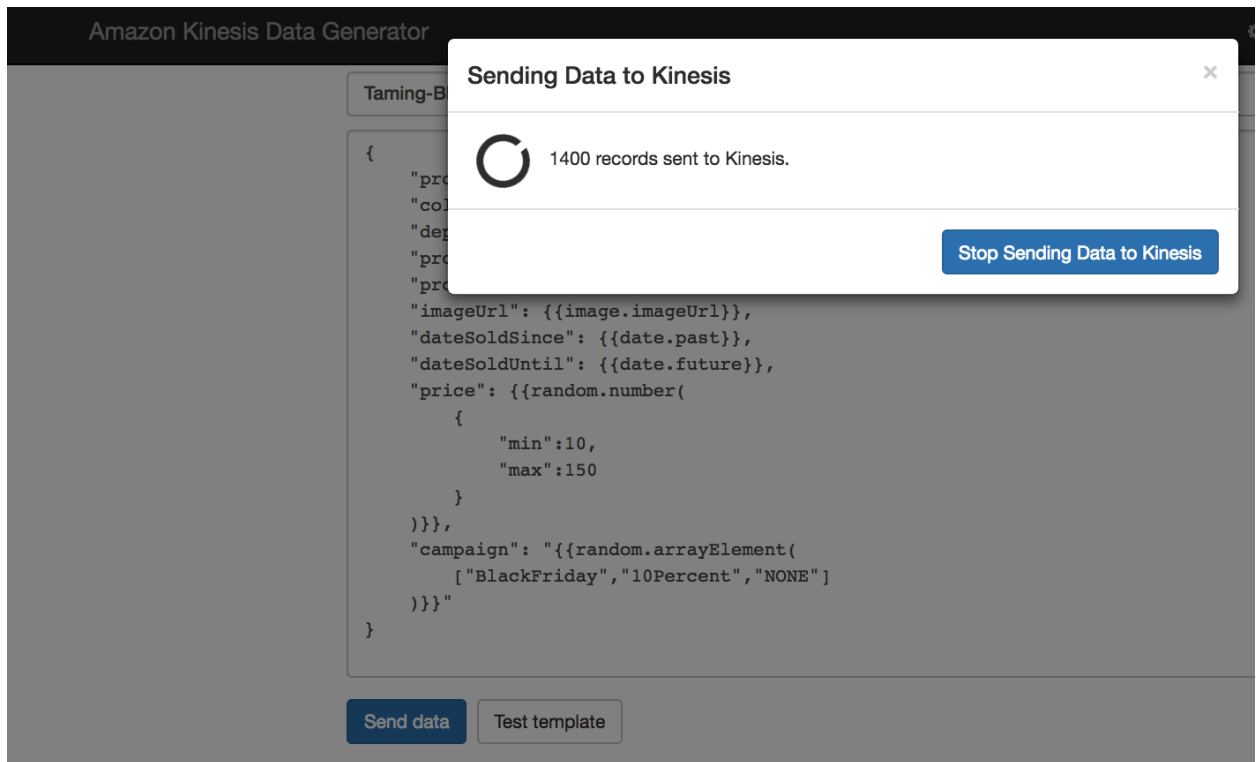


- ii. Policy: Select the role created by the CloudFormation template. The IAM policy has a name like: “**tameFHosePolicySlessDataLakeImmersion**”



3. Return to Kinesis Data Generator tool, select the region you created Firehose Delivery stream (us-east-1), and select your stream. Start sending simulated data at 1000 messages

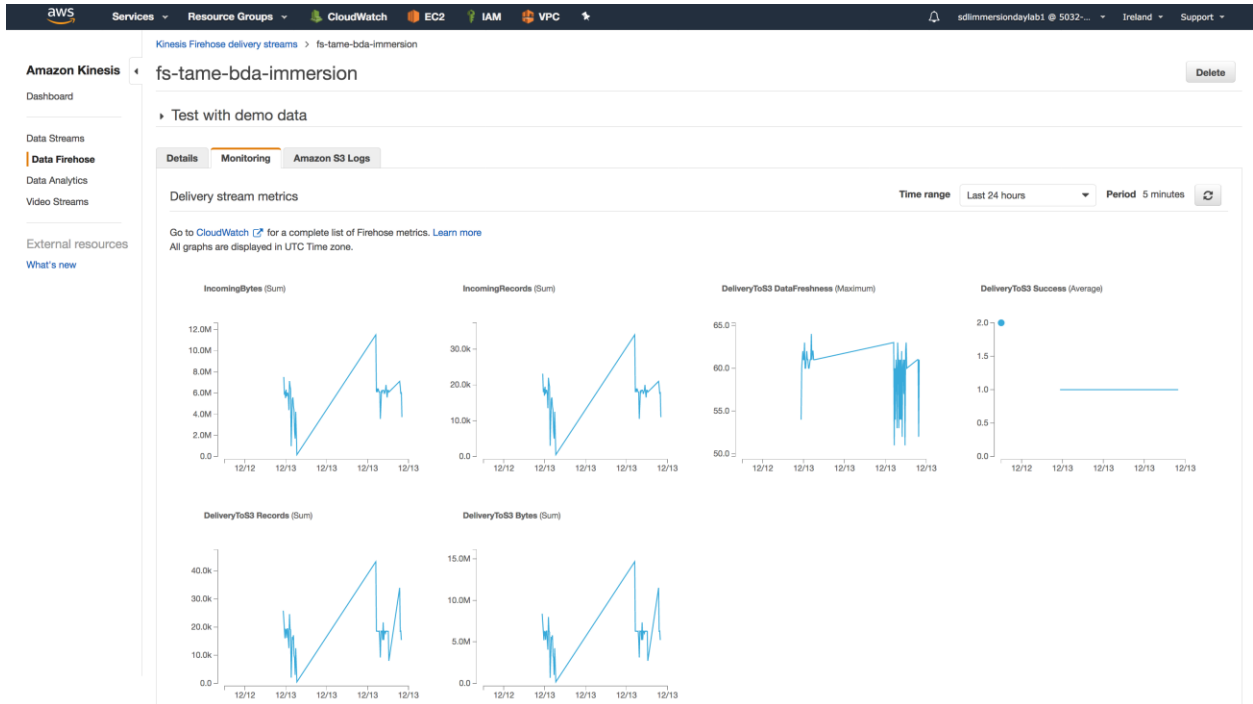
per second rate (If the delivery stream doesn't appear in KDG, simply logout and login to the KDG to refresh).



4. Wait a few minutes. From AWS Console -> Firehose -> Monitoring tab, check activity on your delivery stream.

Serverless Data Lake Immersion Day

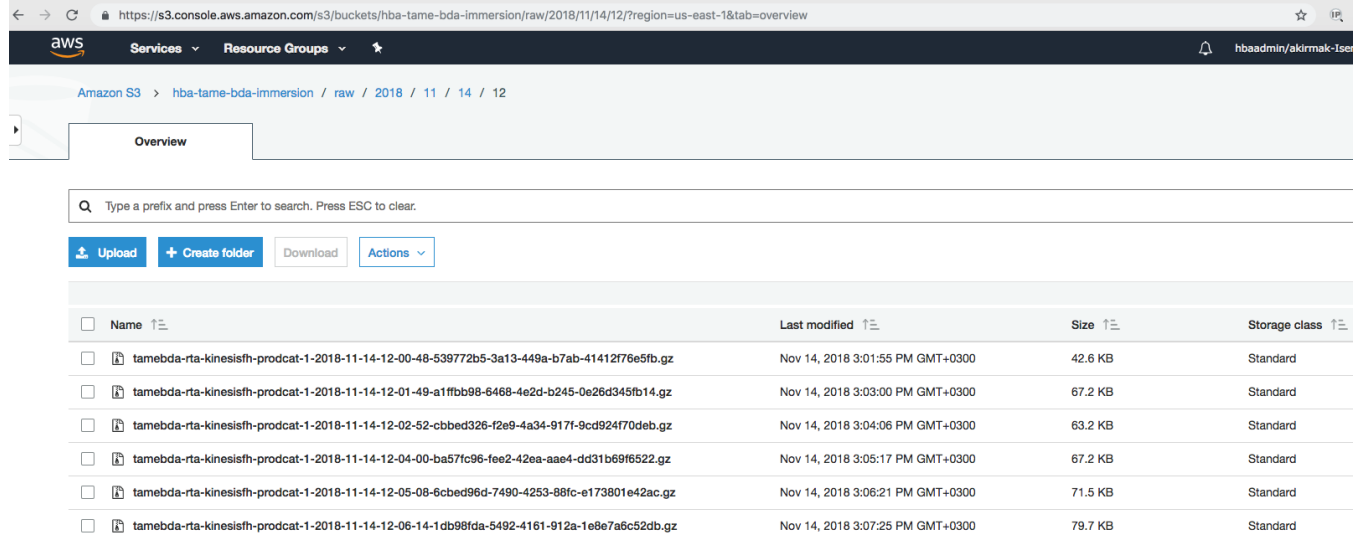
Tame Your Big Data with AWS Kinesis Firehose, S3, Glue



5. From AWS console -> S3, go to the S3 folder you created. You should see data in `s3://<bucket-name>/<prefix>/<year>/<month>/<day>/<hour>` format. For instance:
s3://<your initials>-tame-bda-immersion/raw/2018/11/14/12.
 - a. Hour is in UTC format.
 - b. Depending on your buffering configuration, Firehose writes data to S3 either when size limit (1 ML in our case) or time limit (60 seconds in our case) is reached, whichever condition is satisfied first triggers data delivery to Amazon S3.
 - c. Download one file and view its content.
6. **Hint:** Don't stop sending events from KDG. Let it run while you are working on the other labs. Since the KDG is serverless, there are no servers to pay when there is no traffic. Kinesis Firehose is charged per GB, and during the whole day, the scenario above will not ingest data more than a few hundred MBs.

Serverless Data Lake Immersion Day

Tame Your Big Data with AWS Kinesis Firehose, S3, Glue



The screenshot shows the AWS S3 console interface. The breadcrumb navigation indicates the path: Amazon S3 > hba-tame-bda-immersion / raw / 2018 / 11 / 14 / 12. The 'Overview' tab is selected. Below the navigation bar, there is a search bar and action buttons: Upload, Create folder, Download, and Actions. A table lists the contents of the folder, showing file names, last modified dates, sizes, and storage classes.

<input type="checkbox"/>	Name ↑	Last modified ↑	Size ↑	Storage class ↑
<input type="checkbox"/>	tamebda-rta-kinesisfh-prodcat-1-2018-11-14-12-00-48-539772b5-3a13-449a-b7ab-41412f76e5fb.gz	Nov 14, 2018 3:01:55 PM GMT+0300	42.6 KB	Standard
<input type="checkbox"/>	tamebda-rta-kinesisfh-prodcat-1-2018-11-14-12-01-49-a1ffb98-6468-4e2d-b245-0e26d345fb14.gz	Nov 14, 2018 3:03:00 PM GMT+0300	67.2 KB	Standard
<input type="checkbox"/>	tamebda-rta-kinesisfh-prodcat-1-2018-11-14-12-02-52-cbbed326-f2e9-4a34-917f-9cd924f70deb.gz	Nov 14, 2018 3:04:06 PM GMT+0300	63.2 KB	Standard
<input type="checkbox"/>	tamebda-rta-kinesisfh-prodcat-1-2018-11-14-12-04-00-ba57fc96-fee2-42ea-aae4-dd31b68f6522.gz	Nov 14, 2018 3:05:17 PM GMT+0300	67.2 KB	Standard
<input type="checkbox"/>	tamebda-rta-kinesisfh-prodcat-1-2018-11-14-12-05-08-6cbcd96d-7490-4253-88fc-e173801e42ac.gz	Nov 14, 2018 3:06:21 PM GMT+0300	71.5 KB	Standard
<input type="checkbox"/>	tamebda-rta-kinesisfh-prodcat-1-2018-11-14-12-06-14-1db98fda-5492-4161-912a-1e8e7a6c52db.gz	Nov 14, 2018 3:07:25 PM GMT+0300	79.7 KB	Standard

Summary & Next Steps

Congratulations. You have successfully created an ingestion pipeline without servers, which is ready to process huge amounts of data.

So, what's next? In the next lab, you will create the processing pipeline to convert, transform the data from the ingestion layer..

Deleting Lab Resources

If you won't proceed to the next lab, make sure you terminate the resources below to avoid bills.

- S3:
 - Delete the buckets and the data inside.
- Firehose:
 - Delete the delivery stream
- IAM
 - **Security Warning:** For the sake of simplicity, some of the permissions in the examples below are a bit too permissive (for example, the IAM role defined for the sagemaker notebook has **s3FullAccess permissions. This is a configuration issue we will fix in the next version of the CloudFormation template**). The permissions shall either be removed for the user after the immersion day, or they shall be turned into more fine-grained permissions.

Appendix A – Lab Prerequisites

Appendix A.1 - User Permissions:

Note: These IAM settings are required if you do not have an AWS user with Administration permissions. If you have administrative rights (AWS Role: AdministrativeAccess), they are not necessary.

Note: It is not recommended to do the immersion from your AWS production account/user.

1. Ask your AWS security administrator in your company to create a user for you. We won't specify the procedure in details. Here's an basic overview:
 - a. IAM -> Users -> Add User
 - i. User name: **sdlimmersionlab**
 - ii. Access type: Select "AWS Management console access"
 - iii. Console password: Autogenerated password.
 - iv. Require password reset: Unselect this box.
 - b. Set permissions: Select "attach existing policies directly"
 - i. Add following AWS managed permissions:
 1. AmazonKinesisFirehoseFullAccess
 2. AmazonAthenaFullAccess
 3. AWSGlueConsoleFullAccess
 4. AWSGlueConsoleSageMakerNotebookFullAccess
 5. AmazonSageMakerFullAccess
 6. The custom policy you create in the next step (called **tameSDLImmersionUserPolicy**)
 - ii. Select Create Policy
 1. Select JSON tab & paste the policy JSON below

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "labUserIamPolicy",
      "Effect": "Allow",
      "Action": [
        "iam:CreatePolicy",
        "iam:DetachRolePolicy",
        "iam>DeleteRolePolicy",
        "iam:CreateRole",
        "iam:AttachRolePolicy",
        "iam:PutRolePolicy",
        "iam:ListPolicies"
      ]
    }
  ]
}
```

```
],
  "Resource": [
    "arn:aws:iam::*:role/*SlessData*",
    "arn:aws:iam::*:role/*Kinesis-Data-Generator*",
    "arn:aws:iam::*:policy/*SlessData*"
  ]
},
{
  "Sid": "labUserS3Policy",
  "Effect": "Allow",
  "Action": [
    "s3:PutObject",
    "s3:GetObject",
    "s3:CreateBucket",
    "s3:ListBucket",
    "s3:DeleteObject",
    "s3:DeleteBucket"
  ],
  "Resource": [
    "arn:aws:s3:::*tame-bda-immersion/*"
  ]
},
{
  "Sid": "labUserCloudFormationPolicy",
  "Effect": "Allow",
  "Action": [
    "cloudformation:CreateUploadBucket",
    "cloudformation:CreateStack",
    "cloudformation:DeleteStack",
    "cloudformation:ListStacks",
    "cloudformation:DescribeStackEvents"
  ],
  "Resource": "*"
},
{
  "Sid": "labUserSageMakerPolicy",
  "Effect": "Allow",
  "Action": [
    "sagemaker:ListNotebookInstances",
    "sagemaker:CreateNotebookInstanceLifecycleConfig",
    "sagemaker:ListTags"
  ],
  "Resource": "*"
}
]
```

2. Set policy name as: **tameSDLImmersionUserPolicy**
3. Add this to the user you created in the previous step.

Security Warning: The above permission set is too permissive for the sake of simplicity. The user shall either be deleted after immersion day, or it shall be turned into a more fine-grained policy. It is not recommended for production accounts AS IS.

Appendix A.2 - Service Role Permissions: CloudFormation Template for IAM Roles

Create a stack with the CloudFormation template below.

```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Description": "PA16 2018-12-13 - @akirmak - RevHist: PA16: sagemaker notebook role type fixed. PA15:- (parameters added for AcctId and S3 bucket's name initials)",
  "Parameters": {
    "yourInitials": {
      "Description": "Your Initials to be used in the s3-bucket created. All in small letters pls. e.g. It shall be 'fs' for Frank Sinatra",
      "Type": "String",
      "MinLength": "2",
      "MaxLength": "5"
    }
  },
  "Metadata": {
    "AWS::CloudFormation::Designer": {
      "38df71f9-6e6c-4cb3-bc01-40d0988453b1": {
        "size": {
          "width": 60,
          "height": 60
        },
        "position": {
          "x": 540,
          "y": 180
        },
        "z": 1,
        "embeds": []
      },
      "13e1ef2d-884d-4875-8b55-27a4843db116": {
        "size": {
          "width": 60,
          "height": 60
        },
        "position": {
```

```
    "x": 540,
    "y": 60
  },
  "z": 1,
  "embeds": []
},
"c599e0d5-d036-4fa1-9503-59cebc8349d1": {
  "size": {
    "width": 60,
    "height": 60
  },
  "position": {
    "x": 540,
    "y": 270
  },
  "z": 1,
  "embeds": []
},
"e058c21b-ec9d-4936-95eb-2c492465d87b": {
  "size": {
    "width": 60,
    "height": 60
  },
  "position": {
    "x": 360,
    "y": 270
  },
  "z": 1,
  "embeds": [],
  "isassociatedwith": [
    "c599e0d5-d036-4fa1-9503-59cebc8349d1"
  ]
},
"ed799633-3378-4ef8-8cbe-37b6c7ad5181": {
  "size": {
    "width": 60,
    "height": 60
  },
  "position": {
    "x": 330,
    "y": 180
  },
  "z": 1,
  "embeds": [],
  "isassociatedwith": [
    "38df71f9-6e6c-4cb3-bc01-40d0988453b1"
  ]
}
```

```
},
  "e517d929-5247-426f-9c9e-2c8b7c9a37c6": {
    "size": {
      "width": 60,
      "height": 60
    },
    "position": {
      "x": 320,
      "y": 80
    },
    "z": 0,
    "embeds": [],
    "isassociatedwith": [
      "13e1ef2d-884d-4875-8b55-27a4843db116"
    ]
  }
},
"Resources": {
  "tameGlueRoleSlessDataLakeImmersion": {
    "Type": "AWS::IAM::Role",
    "Properties": {
      "AssumeRolePolicyDocument": {
        "Version": "2012-10-17",
        "Statement": [
          {
            "Effect": "Allow",
            "Principal": {
              "Service": [
                "glue.amazonaws.com"
              ]
            },
            "Action": [
              "sts:AssumeRole"
            ]
          }
        ]
      },
      "ManagedPolicyArns": [
        "arn:aws:iam::aws:policy/service-role/AWSGlueServiceRole"
      ]
    },
    "Metadata": {
      "AWS::CloudFormation::Designer": {
        "id": "38df71f9-6e6c-4cb3-bc01-40d0988453b1"
      }
    }
  }
}
```

```
},
"tameGluePolicySlessDataLakeImmersion": {
  "Type": "AWS::IAM::Policy",
  "Properties": {
    "PolicyName": "AWSGlueServicePolicyServerlessDataLakeImmersion",
    "PolicyDocument": {
      "Version": "2012-10-17",
      "Statement": [
        {
          "Sid": "VisualEditor0",
          "Effect": "Allow",
          "Action": [
            "s3:PutObject",
            "s3:GetObject",
            "s3:DeleteObject"
          ],
          "Resource": {
            "Fn::Join": [
              "",
              [
                "arn:aws:s3:::",
                {
                  "Ref": "yourInitials"
                },
                "-tame-bda-immersion/*"
              ]
            ]
          }
        },
        {
          "Effect": "Allow",
          "Action": [
            "s3:GetObject",
            "s3:PutObject"
          ],
          "Resource": {
            "Fn::Join": [
              "",
              [
                "arn:aws:s3:::",
                {
                  "Ref": "yourInitials"
                },
                "-tame-bda-immersion/compressed-parquet*"
              ]
            ]
          }
        }
      ]
    }
  }
}
```



```
    }
  ]
},
"Roles": [
  {
    "Ref": "tameGlueRoleSlessDataLakeImmersion"
  }
]
},
"Metadata": {
  "AWS::CloudFormation::Designer": {
    "id": "ed799633-3378-4ef8-8cbe-37b6c7ad5181"
  }
}
},
"tameFHoseRoleSlessDataLakeImmersion": {
  "Type": "AWS::IAM::Role",
  "Properties": {
    "AssumeRolePolicyDocument": {
      "Version": "2012-10-17",
      "Statement": [
        {
          "Sid": "",
          "Effect": "Allow",
          "Principal": {
            "Service": "firehose.amazonaws.com"
          },
          "Action": "sts:AssumeRole",
          "Condition": {
            "StringEquals": {
              "sts:ExternalId": {
                "Ref": "AWS::AccountId"
              }
            }
          }
        }
      ]
    }
  },
  "ManagedPolicyArns": [
    "arn:aws:iam::aws:policy/CloudWatchLogsFullAccess"
  ]
},
"Metadata": {
  "AWS::CloudFormation::Designer": {
    "id": "13e1ef2d-884d-4875-8b55-27a4843db116"
  }
}
```

```
},
"tameFHosePolicySlessDataLakelImmersion": {
  "Type": "AWS::IAM::Policy",
  "Properties": {
    "PolicyName": "FirehosePolicyServerlessDataLakelImmersion",
    "PolicyDocument": {
      "Version": "2012-10-17",
      "Statement": [
        {
          "Sid": "",
          "Effect": "Allow",
          "Action": [
            "glue:GetTableVersions"
          ],
          "Resource": "*"
        },
        {
          "Sid": "",
          "Effect": "Allow",
          "Action": [
            "s3:AbortMultipartUpload",
            "s3:GetBucketLocation",
            "s3:GetObject",
            "s3:ListBucket",
            "s3:ListBucketMultipartUploads",
            "s3:PutObject"
          ],
          "Resource": [
            {
              "Fn::Join": [
                "",
                [
                  "arn:aws:s3:::",
                  {
                    "Ref": "yourInitials"
                  },
                  "-tame-bda-immersion"
                ]
              ]
            }
          ],
          "Condition": {
            "Fn::Join": [
              "",
              [
                "arn:aws:s3:::",
                {
                  "Ref": "yourInitials"
                }
              ]
            ]
          }
        }
      ]
    }
  }
}
```

```
    },
    "-tame-bda-immersion/*"
  ]
]
},
"arn:aws:s3:::%FIREHOSE_BUCKET_NAME%",
"arn:aws:s3:::%FIREHOSE_BUCKET_NAME%/*"
]
},
{
  "Sid": "",
  "Effect": "Allow",
  "Action": [
    "lambda:InvokeFunction",
    "lambda:GetFunctionConfiguration"
  ],
  "Resource": {
    "Fn::Join": [
      "",
      [
        "arn:aws:lambda:",
        {
          "Ref": "AWS::Region"
        },
        ":",
        {
          "Ref": "AWS::AccountId"
        },
        ":function:%FIREHOSE_DEFAULT_FUNCTION%:%FIREHOSE_DEFAULT_VERSION%"
      ]
    ]
  }
},
{
  "Sid": "",
  "Effect": "Allow",
  "Action": [
    "logs:PutLogEvents"
  ],
  "Resource": [
    {
      "Fn::Join": [
        "",
        [
          "arn:aws:logs:",
          {
            "Ref": "AWS::Region"
          },
          ":",
          {
            "Ref": "AWS::AccountId"
          },
          ":",
          "log-group:/aws-logs-%FIREHOSE_DEFAULT_FUNCTION%-%FIREHOSE_DEFAULT_VERSION%"
        ]
      ]
    }
  ]
}
```

```
    },
    ":",
    {
      "Ref": "AWS::AccountId"
    },
    "log-group:/aws/kinesisfirehose/tamebda-rta-kinesisfh-prodcat:log-stream:*"
  ]
]
}
]
},
{
  "Sid": "",
  "Effect": "Allow",
  "Action": [
    "kinesis:DescribeStream",
    "kinesis:GetShardIterator",
    "kinesis:GetRecords"
  ],
  "Resource": {
    "Fn::Join": [
      "",
      [
        "arn:aws:kinesis:",
        {
          "Ref": "AWS::Region"
        },
        ":",
        {
          "Ref": "AWS::AccountId"
        },
        ":stream/%FIREHOSE_STREAM_NAME%"
      ]
    ]
  }
},
{
  "Effect": "Allow",
  "Action": [
    "kms:Decrypt"
  ],
  "Resource": [
    "arn:aws:kms:region:accountid:key/%SSE_KEY_ARN%"
  ],
  "Condition": {
    "StringEquals": {
      "kms:ViaService": "kinesis.%REGION_NAME%.amazonaws.com"
    }
  }
}
```

```
    },
    "StringLike": {
      "kms:EncryptionContext:aws:kinesis:arn": {
        "Fn::Join": [
          "",
          [
            "arn:aws:kinesis:%REGION_NAME%:",
            {
              "Ref": "AWS::AccountId"
            },
            ":stream/%FIREHOSE_STREAM_NAME%"
          ]
        ]
      }
    }
  },
  "Roles": [
    {
      "Ref": "tameFHoseRoleSlessDataLakeImmersion"
    }
  ],
  "Metadata": {
    "AWS::CloudFormation::Designer": {
      "id": "e517d929-5247-426f-9c9e-2c8b7c9a37c6"
    }
  },
  "tameSageMakerNBookRoleSlessDataLake": {
    "Type": "AWS::IAM::Role",
    "Properties": {
      "AssumeRolePolicyDocument": {
        "Version": "2012-10-17",
        "Statement": [
          {
            "Effect": "Allow",
            "Principal": {
              "Service": "sagemaker.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
          }
        ]
      }
    }
  },
  "ManagedPolicyArns": [
```

```
    "arn:aws:iam::aws:policy/AmazonS3FullAccess",
    "arn:aws:iam::aws:policy/AmazonAthenaFullAccess"
  ]
},
"Metadata": {
  "AWS::CloudFormation::Designer": {
    "id": "c599e0d5-d036-4fa1-9503-59cebc8349d1"
  }
},
"tameSageMakerNBookPolicySlessDataLake": {
  "Type": "AWS::IAM::Policy",
  "Properties": {
    "PolicyName": "SageMakerNotebookPolicyServerlessDataLake",
    "PolicyDocument": {
      "Version": "2012-10-17",
      "Statement": [
        {
          "Sid": "VisualEditor0",
          "Effect": "Allow",
          "Action": [
            "logs:CreateLogStream",
            "logs:DescribeLogStreams",
            "s3:ListBucket",
            "logs:PutLogEvents"
          ],
          "Resource": [
            {
              "Fn::Join": [
                "",
                [
                  "arn:aws:s3:::aws-glue-jes-prod-",
                  {
                    "Ref": "AWS::Region"
                  },
                  "-assets"
                ]
              ]
            }
          ],
        },
        {
          "Fn::Join": [
            "",
            [
              "arn:aws:logs:",
              {
                "Ref": "AWS::Region"
              },
              ":",
            ]
          ],
          "Effect": "Allow",
          "Action": "logs:CreateLogStream",
          "Resource": "*"
        }
      ]
    }
  }
}
```

```
        "Ref": "AWS::AccountId"
      },
      "log-group:/aws/sagemaker/*"
    ]
  ],
  {
    "Fn::Join": [
      "",
      [
        "arn:aws:logs:",
        {
          "Ref": "AWS::Region"
        },
        ":",
        {
          "Ref": "AWS::AccountId"
        },
        ":log-group:/aws/sagemaker/*:log-stream:aws-glue-*"
      ]
    ]
  }
],
{
  "Sid": "VisualEditor1",
  "Effect": "Allow",
  "Action": "s3:GetObject",
  "Resource": {
    "Fn::Join": [
      "",
      [
        "arn:aws:s3:::aws-glue-jes-prod-",
        {
          "Ref": "AWS::Region"
        },
        "-assets*"
      ]
    ]
  }
},
{
  "Sid": "VisualEditor2",
  "Effect": "Allow",
  "Action": [
```

```
"s3:PutAnalyticsConfiguration",
"s3:GetObjectVersionTagging",
"s3:CreateBucket",
"s3:ReplicateObject",
"s3:GetObjectAcl",
"s3:DeleteBucketWebsite",
"s3:PutLifecycleConfiguration",
"s3:GetObjectVersionAcl",
"s3:PutObjectTagging",
"s3:DeleteObject",
"s3:DeleteObjectTagging",
"s3:GetBucketPolicyStatus",
"s3:GetBucketWebsite",
"s3:PutReplicationConfiguration",
"s3:DeleteObjectVersionTagging",
"s3:GetBucketNotification",
"s3:PutBucketCORS",
"s3:GetReplicationConfiguration",
"s3:ListMultipartUploadParts",
"s3:GetObject",
"s3:PutBucketNotification",
"s3:PutObject",
"s3:PutBucketLogging",
"s3:GetAnalyticsConfiguration",
"s3:GetObjectVersionForReplication",
"s3:GetLifecycleConfiguration",
"s3:ListBucketByTags",
"s3:GetBucketTagging",
"s3:GetInventoryConfiguration",
"s3:PutAccelerateConfiguration",
"s3:DeleteObjectVersion",
"s3:GetBucketLogging",
"s3:ListBucketVersions",
"s3:ReplicateTags",
"s3:RestoreObject",
"s3:GetAccelerateConfiguration",
"s3:ListBucket",
"s3:GetBucketPolicy",
"s3:PutEncryptionConfiguration",
"s3:GetEncryptionConfiguration",
"s3:GetObjectVersionTorrent",
"s3:AbortMultipartUpload",
"s3:GetBucketRequestPayment",
"s3:PutBucketTagging",
"s3:GetObjectTagging",
"s3:GetMetricsConfiguration",
"s3:DeleteBucket",
```



```
    "s3:PutBucketVersioning",
    "s3:GetBucketPublicAccessBlock",
    "s3:ListBucketMultipartUploads",
    "s3:PutMetricsConfiguration",
    "s3:PutObjectVersionTagging",
    "s3:GetBucketVersioning",
    "s3:GetBucketAcl",
    "s3:PutInventoryConfiguration",
    "s3:GetObjectTorrent",
    "s3:PutBucketRequestPayment",
    "s3:PutBucketWebsite",
    "s3:GetBucketCORS",
    "s3:GetBucketLocation",
    "s3:GetObjectVersion",
    "s3:ReplicateDelete"
  ],
  "Resource": {
    "Fn::Join": [
      "",
      [
        "arn:aws:s3:::aws-athena-query-results-",
        {
          "Ref": "AWS::AccountId"
        },
        {
          "Ref": "AWS::Region"
        }
      ]
    ]
  }
},
{
  "Sid": "VisualEditor3",
  "Effect": "Allow",
  "Action": [
    "s3:GetAccountPublicAccessBlock",
    "s3:ListAllMyBuckets",
    "s3:HeadBucket"
  ],
  "Resource": "*"
},
{
  "Sid": "VisualEditor4",
  "Effect": "Allow",
  "Action": "logs:CreateLogGroup",
  "Resource": [
```

```
{
  "Fn::Join": [
    "",
    [
      "arn:aws:logs:",
      {
        "Ref": "AWS::Region"
      },
      ":",
      {
        "Ref": "AWS::AccountId"
      },
      ":log-group:/aws/sagemaker/*"
    ]
  ],
  {
    "Fn::Join": [
      "",
      [
        "arn:aws:logs:",
        {
          "Ref": "AWS::Region"
        },
        ":",
        {
          "Ref": "AWS::AccountId"
        },
        ":log-group:/aws/sagemaker/*:log-stream:aws-glue-*"
      ]
    ]
  }
],
{
  "Sid": "VisualEditor5",
  "Effect": "Allow",
  "Action": [
    "glue:GetDevEndpoints",
    "glue:UpdateDevEndpoint",
    "glue:GetDevEndpoint"
  ],
  "Resource": {
    "Fn::Join": [
      "",
      [
        "arn:aws:glue:",
```

```
{
  "Ref": "AWS::Region"
},
{
  "Ref": "AWS::AccountId"
},
":devEndpoint/gj-tame-bda-kdg-raw2parquet-devEndpoint*"
]
}
}
]
},
"Roles": [
  {
    "Ref": "tameSageMakerNBookRoleSlessDataLake"
  }
]
},
"Metadata": {
  "AWS::CloudFormation::Designer": {
    "id": "e058c21b-ec9d-4936-95eb-2c492465d87b"
  }
}
}
}
```

References

1. Website, AWS Data Lake, <https://aws.amazon.com/tr/big-data/datalakes-and-analytics/what-is-a-data-lake/>
2. Whitepaper, “Lambda Architecture for Batch and Stream Processing, AWS, October 2018, <https://d0.awsstatic.com/whitepapers/lambda-architecure-on-for-batch-aws.pdf>