Author 1

Author 2

CrayonData KBZ Migration

AWS Account to Account – Migration

**Version 1.0**

**20-Feb -2023**

**CloudWayZ Team**

**Version history**

|  |  |  |  |
| --- | --- | --- | --- |
| Version Number | Date Updated | Updated by | Notes |
| 1.0 | 19-Feb -2023 | Siddhartha Sarnobat | Initial Version |

[1. Summary 3](#_Toc1305269336)

[1.1 Introduction 3](#_Toc1207686302)

[1.1 Objective 4](#_Toc1890137735)

[2. Scope 4](#_Toc918074762)

[2.1 Scope as Impact on existing infrastructure 5](#_Toc1223704533)

[2.2 Assumptions & Risks 5](#_Toc939711894)

[2.3 Clarifications / Questions 6](#_Toc1057998165)

[3. Implementation - Technical Design 7](#_Toc138174248)

[3.1 Logical Architecture 8](#_Toc278851905)

[3.2 Request Flow 9](#_Toc1230926225)

[3.3 Components 9](#_Toc1252429153)

[3.3.1 AIRFLOW 9](#_Toc1334006106)

[1. Airflow to SFTP: 9](#_Toc774920336)

[2. Airflow to Jupyter: 10](#_Toc398568501)

[3. Airflow to S3: 10](#_Toc815669418)

[4. S3 to Airflow: 10](#_Toc1723370718)

[5. Airflow to EMR: 10](#_Toc1112414085)

[6. OBS to Airflow: 10](#_Toc432982554)

[3.3.2 JUPYTER 10](#_Toc1817547895)

[3.3.3 SFTP 10](#_Toc358473426)

[1. Airflow to SFTP: 10](#_Toc1041365088)

[3.3.4 EMR 11](#_Toc2083692259)

[1. EMR to S3: 11](#_Toc1188818374)

[2. S3 to EMR: 11](#_Toc2131635095)

[3.3.5 Application Flow (Hasura to Application) 11](#_Toc1124811563)

[3.4 Implementation Strategy 11](#_Toc2070516295)

[3.4.1 Creating S3 Buckets and Copying Data in New KBZ Account B 12](#_Toc868421877)

[3.4.2 Creating AWS EC2 in Account A and Account B 15](#_Toc1069470413)

[3.4.3 DB Snapshot Copy from Account A to Account B 18](#_Toc1598381418)

[3.4.4 DAG Script Changes 19](#_Toc1499305148)

[4. Deployment Strategy 21](#_Toc768757241)

[4.1 Cutover Plan 21](#_Toc1144605741)

[4.1.1 Prerequisites 21](#_Toc1832586421)

[4.1.2 DAY 1 21](#_Toc387689769)

[4.1.3 DAY 2 22](#_Toc424419303)

[4.1.4 Observations/Conclusion/Success Criteria 22](#_Toc1215945450)

[4.1.5 Rollback Plan 22](#_Toc1325151938)

[5. Non-Functional requirements 22](#_Toc887987275)

[5.1 Additional Cost Estimates 22](#_Toc464385046)

[5.2 Essentials 23](#_Toc316851917)

[6. References 23](#_Toc492227068)

# 1. Summary

## 1.1 Introduction

CrayonData has partnered with CloudWayZ to migrate AWS resources from CrayonData account to KBZ Account. This migration will involve spinning up new resources in the KBZ Account. KBZ has provisioned a New AWS Account specially.

To achieve this AWS Account migration CloudWayZ team needs to come up with a plan that involves smooth cutover making sure we are maintaining two environment blue and green that will make sure incase failures seen in green environment we can have a smooth transition to the live-blue environment without any data loss.

As a part of this setup this document will list out the whole plan which will act as a runbook while we execute this project plan.

## 1.1 Objective

Objective of this document is to layout overall technical design and implementation which will be used as our end-to-end migration runbook

Document will include scope, impact, logical architecture, configurations, Infrastructure changes and other non-functional requirements.

# 2. Scope

Following is requirement scope for implementation of this change.

This scope considers only Crayon AWS Resources that are hosted in the CrayonDta AWS Account.

1. Migration will involve the below of these resources
   1. AWS S3 datalake
   2. AWS EC2 – SFTP Server
   3. AWS EC2 – Airflow
   4. AWS EC2 – Juptyer
   5. AWS EC2 – Bastion Server
   6. AWS RDS – Tableau DB
   7. IAM Role and Policies
   8. VPC, Subnets, RouteTables

## 2.1 Scope as Impact on existing infrastructure

Existing infrastructure will not take part of this migration as the process followed for this migration will involve moving backups/AMI’s/Snapshots from Crayon AWS Account A to KBZ AWS Account B.

We will also make sure that no testing/disruption is encountered whilst migration resources from Account A to Account B.

Some resources like Jupyter, Airflow and Bastion server’s will be newly launched(version v2) to avoid any interactions with current/live version v1 this will reduce the blast radius and allow us to test the end-to-end flow making sure cross-account data is populated and network connectivity is UP.

## 2.2 Assumptions & Risks

Following are assumptions taken for this implementation,

As per the latest updated SoW and migration plan sent by Crayon Team below are the assumptions

|  |  |
| --- | --- |
| **Sr. No.** | **Assumption** |
| 1 | CloudWayZ team to create Network Layer(VPC/Subnets/RouteTables) |
| 2 | CloudWayZ team to create Cross-Account Roles(IAM) in Account A and Account B |
| 3 | CloudWayZ team to create AWS Resources only which were part of Crayon AWS Account A in AWS Account B |
| 4 | CloudWayZ team to do the AWS S3 datalake copy to new S3 buckets in AWS Account B |

Following are risks for post migration,

|  |  |  |  |
| --- | --- | --- | --- |
| **Sr. No.** | **Risk Type** | **Dependency** | **Description** |
| 1 | General | KBZ | Data Syncing not working between New Datalake S3 and Deepsee Data S3 |
| 2 | General | KBZ | EMR v2 unable to read/write tables <->Application DB |
| 3 | General | KBZ | Tableau Server v2 unable to read data from Tableau DB v2 (KBZ needs to update Tableau DB endpoint) – If Endpoint is hardcoded |

## 2.3 Clarifications / Questions

|  |  |  |
| --- | --- | --- |
| **Sr. No.** | **Question / Clarification** | **Response** |
| 1 | CloudWayZ will handle creation of the Network Layer(VPC/Subnets/RouteTables) or KBZ |  |
| 2 | VPC Peering between Crayon Data AWS Account and KBZ New Account B to be done by CloudWayZ or KBZ? |  |
| 3 | Will KBZ also migrate their part of resources (eg : Deepsee Data/Application DB/Tableau Server etc) from existing their Account to New KBZ AWS Account B |  |
| 4 | If no, will they have a VPC between their New KBZ AWS Account B and their existing KBZ AWS Account |  |
| 5 | Athena Database – Creating new tables/database based on new S3 buckets will be done by CloudWayZ/KBZ/Crayon |  |
| 6 | How is data sync done between DataLake S3 and Deepsee S3 (Flow Number 8) – Which buckets are syncing data ? |  |
| 7 | Jupyter EC2 in Crayon Account A has endpoint of Tableau DB hardcoded or we are using a DNS (Domain Name) |  |
| 8 | EMR connecting to Application DB – Are we connecting via DNS or endpoint is hardcoded in EMR ? |  |
| 9 | Airflow UI is accessed by UI for Airflow V2 we will need a new DNS Name/ Sub-domain |  |
| 10 | CrayonData to provide their SCM Repository access to store CloudFormation Scripts |  |
| 11 | Does the Airflow setup uses Docker ? |  |
| 12 | For EMR what is the number of master and slave nodes |  |

# 

# 3. Implementation - Technical Design

## 3.1 Logical Architecture

Following existing & Proposed diagram represents high-level logical architecture of Payment gateway core components with integrations

**CURRENT ARCHITECTURE**

|  |
| --- |
| **PROPOSED ARCHITECTURE** |
|  |

## 3.2 Request Flow

|  |  |
| --- | --- |
| **FLOW NUMBER** | **DESCRIPTION** |
| 1 | Crayon users connect to the Bastion EC2 instance through remote connection |
| 2 | Triggers the Jupyter notebook from the airflow machine |
| 3 | Airflow to SFTP: File sensors are triggered from Airflow based on files received in SFTP. Reports are uploaded to SFTP out path for sharing with client.  SFTP to Airflow: Data transferred from SFTP to airflow for archival process. |
| 4 | Airflow to S3: Data sync between prod and dev buckets using shell script.  S3 to Airflow: Data extracted from S3 to airflow for generation of reports. |
| 5 | Trigger EMR clusters using DAGs |
| 6 | Load real time sqoop extracted data into S3 buckets |
| 7 | Read/write data from/to Mysql RDS database |
| 8 | Data sync between prod/dev tables and deepsee tables |
| 9 | Load source files into S3 |
| 10 | Read/write data to S3 from EMR during data processing |
| 11 | Read/write final tables from application database |
| 12 | Read data from OBS cloud |
| 13 | Data extracted from Mysql database to be viewed in dashboards |
| 14 | Real/near real time data extracted from Oracle database using sqoop jobs |
| 15 | Tableau dashboards can be accessed by end users |
| 16 | Hasura is used to connect with different RDS instances for graphql querying |
| 17 | Directly integrating graphQL queries to frontend instead of using API calls. |
| 18 | Application makes use of load balancer for performance enhancement |
| 19 | The KBZ application’s frontend technology is ReactJS, so we are using graphQL NPM package to use hasura graphQL queries in the frontend. |
| 20 | Data visualization and dashboard preparation |
| 21 | Glue database created over the underlying S3 data |
| 22 | Athena queries are executed over the underlying S3 data |
| 23 | Deepsee application accessed by end users |
| 24 | Application accessed by end users |

## 3.3 Components

### AIRFLOW

#### Airflow to SFTP:

* File sensors are triggered from Airflow based on files received in SFTP.
* Reports are uploaded to SFTP out path for sharing with client.
* Data transferred from SFTP to airflow for the archival process. In the next phase, we are planning to archive the files directly from SFTP to S3.

#### Airflow to Jupyter:

* Triggers Jupyter notebooks from airflow machine

#### Airflow to S3:

* Data sync between prod and dev buckets using shell script.

#### S3 to Airflow:

* Data extracted from S3 to airflow for generation of reports.

#### Airflow to

* Trigger EMR clusters using DAGs

#### OBS to Airflow:

* Read data from OBS cloud

### JUPYTER

1. Oracle DB to Jupyter:

* Real/near real time data extracted from Oracle database using sqoop jobs

2. Tableau RDS (Crayon side) to Jupyter:

* Read data from Mysql RDS database (Crayon side)

3. Jupyter to Tableau RDS (Crayon side):

* Write data to Mysql RDS database (Crayon side)

4. Jupyter to S3:

* Load real time sqoop extracted data into S3 buckets

5. Airflow to Jupyter:

* Triggers Jupyter notebooks from airflow machine

### SFTP

#### 1. Airflow to SFTP:

* File sensors are triggered from Airflow based on files received in SFTP.
* Reports are uploaded to SFTP out path for sharing with client.

2. SFTP to Airflow:

* Data transferred from SFTP to airflow for archival process.

3. SFTP to S3:

* Load source files into S3

### 3.3.4 EMR

#### 1. EMR to S3:

* Write data to S3 from EMR during data processing

#### 2. S3 to EMR:

* Read data from S3 to EMR during data processing

3. Airflow to EMR:

* Trigger EMR clusters using DAGs

4. EMR to application RDS (KBZ side):

* Write final tables to application database (KBZ side)

5. Application RDS (KBZ side) to EMR:

* Read final tables from application database (KBZ side)

6. EMR to Tableau RDS (Crayon side):

* Write final tables to Mysql RDS database (Crayon side)

### 3.3.5 Application Flow (Hasura to Application)

1. The KBZ applications frontend technology is ReactJS, so we are using graphQL NPM package to use hasura graphQL queries in the frontend.
2. Here we are directly integrating graphQL queries to frontend instead of using API calls.

## 3.4 Implementation Strategy

### 3.4.1 Creating S3 Buckets and Copying Data in New KBZ Account B

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Stage** | **Description** | **Priority** | **Action Items** | **Comments** |
| 1 | Create a New repo in Client’s SCM repository |  | KBZ/CrayonData | Infrastructure Repository will be hosted  In this SCM repo |
| 2 | Creating Network layer Components for Jenkins |  | CloudWayZ | 1 Management VPC – CIDR block  6 Subnets – 3 Public , 3 Private  2 Route Tables – 1 Public, 1 Private  1 IGW  1 Security Group  1 Key Pair |
| 3 | Create a Jenkins in New KBZ Account B |  | CloudWayZ | Team to create to a Jenkins instance  T3.medium that will orchestrate the  Creation of S3 buckets.  Connect to a SCM repository hosted by  CrayonData/KBZ (Require access to their  SCM) |
| 4 | Jenkins Authentication and Authorization |  | CloudWayZ | Gitlab : Jenkins will be authenticate using  SSH keys with gitlab and will be able  To clone a new infra repository  S3 : Jenkins EC2 will have a S3 Create/Read  /Update access to create S3 Buckets |
| 5 | Once Jenkins is setup start creating cloudformation scripts for the buckets  The script will create S3 buckets from Cloudformation  Further Creation/Updation of S3 Buckets can be orchestrated using Cloudformation |  | CloudWayZ | **Buckets v1**  kbz-dev-dl-app-zone  kbz-dl-app-zone  kbz-dev-dl-raw-zone  kbz-dl-raw-zone kbz-dev-dl-stage-zone  kbz-dl-stage-zone  kbz-dl-archived-zone  emr-prod-logs-sg kbz-mm-emr-data  kbz-edp-data  **Buckets v2**  kbz-dev-dl-app-zone-v2 kbz-dl-app-zone-v2 kbz-dev-dl-raw-zone-v2 kbz-dl-raw-zone-v2 kbz-dev-dl-stage-zone-v2 kbz-dl-stage-zone-v2 kbz-dl-archived-zone-v2 emr-prod-logs-sg-v2 kbz-mm-emr-data-v2  kbz-edp-data-v2 |
| 6 | Data copy using S3P  Utility  Once the S3 Buckets in Account B are created.  Launch a t3.xlarge Ubuntu/EC2-linux server in Management VPC  Download the S3P utility and start data copy between v1 and v2 buckets  The data entire data copy should take less than a day |  | CloudWayZ | S3P (list/copy/sync/compare S3 buckets 5x-50)  Faster  <https://github.com/generalui/s3p>  Make sure the EC2 has suffiecient IAM  Permissions to COPY between buckets  Cross Account Role and Permission  References :  <https://medium.com/tensult/copy-s3-bucket-objects-across-aws-accounts-e46c15c4b9e1>  <https://blog.vizuri.com/how-to-copy/move-objects-from-one-s3-bucket-to-another-between-aws-accounts>  Note: You can use the below features  S3P supports many useful operations right out of the box:   * **compare**: Compare two buckets and produce a summary of their differences. * **list**: List all matching files. * **list-buckets:** List all your S3 buckets. * **summarize:** Scan all items in one bucket and produce a summary of all the items. * **sync**: Only copy files which do not exist in the target bucket.   S3P Costs  S3P uses two S3 operations: list-objects and copy-object. Currently both operations cost $.005 per 1000 requests, or 200,000 requests per $1 (US-East, June 2020).  Note : *S3 pricing is based on how much you use,*not how fast you use it*.* |

### 3.4.2 Creating AWS EC2 in Account A and Account B

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Stage** | **Description** | **Priority** | **Action Items** | **Comments** |
| 1 | Launch Airflow v2 in Crayon Account A |  | CloudWayZ | * Create Airflow v2 from a clone/AMI of Airflow v1 make sure the network configurations are same as Airflow v1 * Create new key pair, new security group * Create a cross-account IAM role to access data lake reference links   <https://repost.aws/knowledge-center/s3-instance-access-bucket>  <https://dev.to/aws-builders/explore-cross-accounts-access-with-iam-roles-from-ec2-to-s3-with-custom-managed-key-5bje>   * Launch Airflow v2 make sure to turn off the DAG’s and cron jobs until all the setup is complete * OBS Bucket   obs://kbzpay-market-analysis  obs://kbz-payroll-search   * Make sure above buckets are reachable and you can read * Make changes to DAG scripts and make sure all the DAG’s pointing to the new S3 Buckets * Check all the parameters/env variables make sure that all the parameter/env’s point to V2 env |
| 2 | Create Cross Account IAM Role |  | CloudWayZ | * Create Cross Account IAM role that has permissions for following * Read/Write access to S3 Buckets   kbz-dev-dl-app-zone-v2 kbz-dl-app-zone-v2 kbz-dev-dl-raw-zone-v2 kbz-dl-raw-zone-v2 kbz-dev-dl-stage-zone-v2 kbz-dl-stage-zone-v2 kbz-dl-archived-zone-v2 emr-prod-logs-sg-v2 kbz-mm-emr-data-v2  kbz-edp-data-v2   * Creating EMR cluster having Cross Account IAM Role Access   <https://aws.amazon.com/premiumsupport/knowledge-center/cross-account-access-iam/> |
| 3 | Create a Jupyter v2 instance in Account A |  | CloudWayZ | * Create Jupyter v2 from a clone/AMI of Airflow v1 make sure the network configurations are same as Airflow v1 * Create new key pair, new security group * Create a cross-account IAM role to access data lake reference links   <https://repost.aws/knowledge-center/s3-instance-access-bucket>  <https://dev.to/aws-builders/explore-cross-accounts-access-with-iam-roles-from-ec2-to-s3-with-custom-managed-key-5bje>   * Launch the Jupyter v2 and make sure sqoop jobs/cron are turned off * Make changes to Sqoop jobs make sure they are pointing to V2 variables |
| 4 | Create Cross Account IAM Role |  | CloudWayZ | * Create Cross Account IAM role that has permissions for following * Read/Write access to S3 Buckets   kbz-dev-dl-app-zone-v2 kbz-dl-app-zone-v2 kbz-dev-dl-raw-zone-v2 kbz-dl-raw-zone-v2 kbz-dev-dl-stage-zone-v2 kbz-dl-stage-zone-v2 kbz-dl-archived-zone-v2 emr-prod-logs-sg-v2 kbz-mm-emr-data-v2  kbz-edp-data-v2 |
| 5 | Jupyter EC2 v2 Connectivity |  | CloudWayZ | * Make sure VPC peering is established between Account A and New Account B * Jupyter v2 should be able to connect Tableau DB(Security Grp) – Make sure endpoint is shared and reachable * Jupyter v2 should be able to connect Oracle DB DB(Security Grp) – Make sure endpoint is shared and reachable |
| 6 | Account B  Network Components |  | CloudWayZ/KBZ | * VPC * Subnets * RouteTables * IGW * NAT Gateway * VPC Peering |
| 7 | Launch SFTP v2 in Account B |  | CloudWayZ | * Create SFTP v2 from a clone/AMI of SFTP v1 * Share this clone with Account B * Launch this AMI in new VPC, Private subnet * Create Security Group * Create New Key Pair * Size to be checked * Check if files are loaded in S3 * Check if Files are store in Airflow Archival * Check if DAG in Airflow is triggered and reports are store in SFTP v2 output folder |
| 8 | Launch Bastion EC2 v2 in Account.A |  | CloudWayZ | * Make sure VPC peering is established between Account A and New Account B * Try to connect from Bastion in Account A to SFTP v2 in Account B |

### 3.4.3 DB Snapshot Copy from Account A to Account B

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Stage** | **Description** | **Priority** | **Action Items** | **Comments** |
| 1 | Create DB snapshot of Tableau DB |  | CloudWayZ | * Create Latest DB snapshot of Tableau DB * If the KMS is default make sure to create a custom KMS key and encrypt it and take the snapshot * Follow stage 2 to copy snapshot from one Account to Another |
| 2 | Move the DB snapshot from one AWS account to another |  | CloudWayZ | * To copy the encrypted snapshot from AWS Account to another follow the below reference   Reference : https://aws.amazon.com/premiumsupport/knowledge-center/rds-snapshots-share-account/ |
| 3 | Launch DB  Snapshot |  | CloudWayZ | * Make sure VPC peering is established between Account B and Account A * Allow port and sg/ip of Jupyter from Jupyter V2 * Share endpoint with Jupyter v2 |

### 3.4.4 DAG Script Changes

Total no of DAG scripts are 109.

At runtime these DAG scripts will call the parameter store and fetch variables.

These params will have default encryption enabled since they store credentials.

We will AWS SSM Parameter as a secrets backend.

Note : Reachability Tests and Permission Issues to be resolved by CloudWayZ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Stage** | **Description** | **Priority** | **Action Items** | **Comments** |
| 1 | Create SSM param for DAG scripts |  | CloudWayZ | * Create a SSM parameter store key in JSON format that will store Key and Value pairs in JSON format * Make sure Airflow v2 IAM Role has ability to fetch SSM params(List/Read) permissions |
| 2 | Launch a Snapshot of Application DB (Test Application DB) |  | CloudWayZ | * Take a latest snapshot of Application DB(Assuming this is Application DB from KBZ side) * Create new SG for this db * Launch this snapshot in the same Subnet Group and using the same Parameter Group/Option Group/Logging Levels etc * Note this endpoint and provide to crayon when Crayon Team start with DAG changes   **Reachability/Permission Tests**   * EMR can successfully connect to Test Application DB |
| 2 | DAG changes |  | CrayonData | * CrayonData Team will make changes in Airflow V2 in a test DAG and check if we can successfully fetch values from SSM parameter store.   References : <https://aws.plainenglish.io/aws-backed-variables-and-connections-in-airflow-d55080263e10>  https://airflow.apache.org/docs/apache-airflow-providers-amazon/stable/secrets-backends/aws-ssm-parameter-store.html |
| 3 | Testing DAG changes |  | CrayonData & CloudWayZ | * Once DAG changes are done on a test DAG * We can then start making changes in all the DAG’s in Airflow V2   **Reachability/Permission Tests**   * EC2 Role has enough permissions to read/write in S3 * EC2 Role has enough permissions to read from SSM * Check if S3 is reachable IAM Role/Policy * Check SFTP v2 is reachable read/write operation (vice-versa) * Check EMR is launching properly IAM Role/Permissions * Check EMR network reachability over port * Check if OBS Bucket is accessible * Network Connectivity between Airlfow and Jupyter(SG) |
| 4 | Sqoop/Cron Job (40-50)  Changes |  | CrayonData & CloudWayZ | * Once DAG changes are over we will apply the same changes in CRON jobs   We can AWS CLI commands of SSM param store if we want to use the SSM params   * We will need to make sure all the endpoints point to the V2 env and ascertain that cross-account role is working and data is being populate in S3 v2   **Reachability/Permission Tests**   * EC2 Role has enough permissions to read/write in S3 * EC2 Role has enough permissions to read from SSM * CloudWayZ team to check if S3 is reachable IAM Role/Policy * CloudWayZ team to check SFTP v2 is reachable read/write operation (vice-versa) * CloudWayz team to if EMR is launching properly IAM Role/Permissions * Check EMR network reachability over port |
| 5 | Juptyer Notebooks(2-3) |  | CrayonData & CloudWayZ | * Jupyter Python Notebooks make the env variables changes   **Reachability/Permission Tests**   * EC2 Role has enough permissions to read/write in S3 * EC2 Role has enough permissions to read from SSM * CloudWayZ team to check if S3 is reachable IAM Role/Policy * CloudWayZ team to check Tableau Application DB reachability on port 3306 * CloudWayz team to check if Oracle DB is reachable over 1880 |

# 4. Deployment Strategy

## 4.1 Cutover Plan

### 4.1.1 Prerequisites

* EMR is pointing to Main Application DB DNS Endpoint

### 4.1.2 DAY 1

1. Make sure DAG scripts/Cron Jobs in Airflow v2 are turned OFF
2. All S3 Buckets in Airflow v2 are cleaned UP – NO Files should be present
3. Cross-Check all Security Group/IAM Role and Permissions/Resource Policies on S3
4. Start the S3P Operation for all the S3 Buckets
5. Create a latest snapshot of Tableau DB and share between Crayon Account A and Crayon Account B
6. Launch the snapshot in KBZ Account B (same network configurations like application DB) – make sure security group is different
7. Change endpoint in Jupyter v2 or change DNS to point to Tableau DB v2 – Make sure TTL settings are short duration
8. Share the latest Tableau DB v2 endpoint/change DNS with KBZ so they can make changes at their end
9. Make changes in Jupyter V2 scripts to point to latest Tableau DB V2

### 4.1.3 DAY 2

Day 2 will be where we will want to take a downtime of roughly 7-8 Hours incase we want to rollback also

1. Stop all the DAG jobs/Scoop Jobs/Cron Jobs in Airflow/Jupyter v1
2. Stop putting files in SFTP v1
3. Re-sync S3 Buckets using S3P between Crayon Account A and KBZ Account B
4. One-by-One start the DAG scripts in Airflow v2
5. Start the Sqoop jobs in Jupyter V2
6. Start the Cron Jobs in Airflow v2

### 4.1.4 Observations/Conclusion/Success Criteria

* Data should sync from DataLake to Deepsee S3 Buckets
* EMR should successfully run from the DAG scripts and read/write to Application DB
* Jupyter Notebooks should be able to read/write from Tableau DB and like in S3
* Sqoop Jobs(Jupyter V2) should able to read from Oracle DB
* Tableau server should be able to query the Tableau DB v2 and get latest data
* SFTP v2 files put should populate in Airflow and likewise in S3
* Bastion Host should be able to read/write files in SFTP v2
* Airflow should be able to Launch EMR cluster

### 4.1.5 Rollback Plan

**If unable to meet the Success Criteria we will Rollback to V1 version as follows**

1. Turn off DAG/Sqoop/Cron v2
2. If Delta data is found re-sync from S3 V2 to S3 V1 make sure commands are ready
3. Once done start the DAG/Sqoop/Cron v1

# 5. Non-Functional requirements

## 5.1 Additional Cost Estimates

* We will use an additional EC2 ~ t3.xlarge to do the S3 sync operation
* We will run two parallel env’s(blue and green) atleast for 2 months
* Jenkins server will be launched to orchestrate Infrastructure as a Code (t3.medium/small) this will run only for a month and can be shutdown later

## 5.2 Essentials

Following are the essentials are to make sure the migration is successful

1. Data Copy estimated data is 90 TB – S3P Tool to be used estimated time of copy is within a day (POC is done)
2. Cross Account Roles – Making sure Crayon AWS Account A has enough permissions and privileges to access resources in KBZ AWS Account B
3. VPC Peering – Peering connection between Crayon Account A and KBZ Account B this will make sure we can use private IP’s to connect to resources in KBZ Account B
4. Security Group – Making sure only required ports are opened from Account A to Account B
5. Cloudformation Infrastructure as a Code – S3 buckets(DataLake) future updates if any should be done via Infrastructure as a Code this will make sure releases to S3 are monitored and carefully released in production – Orchestrated by Jenkins
6. SSM/S3 Encrypted Env Variables – We will use a Secrets Backed this is one of the industry best practice to store env variables (POC is pending)

# 6. References

Following are the references for this plan

* <https://airflow.apache.org/docs/apache-airflow/stable/administration-and-deployment/security/secrets/secrets-backend/index.html#secrets-backend-configuration>
* <https://aws.plainenglish.io/aws-backed-variables-and-connections-in-airflow-d55080263e10>
* <https://www.genui.com/open-source/s3p-massively-parallel-s3-copying>
* <https://dev.to/aws-builders/explore-cross-accounts-access-with-iam-roles-from-ec2-to-s3-with-custom-managed-key-5bje>
* https://aws.amazon.com/premiumsupport/knowledge-center/rds-snapshots-share-account/