Metadata Migration

Design Document

Version 1.0

10/16/2024

# Version History

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Version Number | Implemented By | Revision Date | Approved By | Approval Date | Description of Change |
| 1.0 | Eran Rosenberg |  |  |  |  |

Table of Contents

[Version History 2](#_Toc182554495)

[1 Introduction 4](#_Toc182554496)

[1.1 Purpose of this document 4](#_Toc182554497)

[1.2 Background 4](#_Toc182554498)

[1.3 Scope 4](#_Toc182554499)

[1.4 Assumptions 5](#_Toc182554500)

[1.5 Constraints 5](#_Toc182554501)

[2 Architecture Design 5](#_Toc182554502)

[2.1 Hardware Architecture 5](#_Toc182554503)

[2.2 Software Architecture 5](#_Toc182554504)

[3 Software Design 5](#_Toc182554505)

[3.1 Introduction 5](#_Toc182554506)

[3.2 Detailed Design 5](#_Toc182554507)

[3.2.1 APIs 5](#_Toc182554508)

[3.2.2 Scheduled Tasks 6](#_Toc182554509)

[3.3 User Interface Design 9](#_Toc182554510)

[3.4 System Integration 9](#_Toc182554511)

[4 Database Design 9](#_Toc182554512)

[4.1 Table Design 9](#_Toc182554513)

[4.1.1 HPC\_DATA\_MIGRATION\_TASK 9](#_Toc182554514)

[4.1.2 HPC\_DATA\_MIGRATION\_TASK\_RESULT 9](#_Toc182554515)

# Introduction

## Purpose of this document

The purpose of this design document is to specify the process, API and Scheduled Tasks changes to support metadata migration following completion of archive migration by the storage team

## Background

DME provides a capability to migrate data objects from their current S3 archive to a new S3 archive location. Migration between all supported S3 storage providers (AWS, Cloudian, Cleversafe, etc) is supported.

3 APIs provide capability to migrate a single data object, a collection of data objects, a list of data objects, or a list of collections. In all cases, once a migration request is validated and received, a migration task ID is returned.

The transfer of the data is performed by DME servers utilizing the streaming method. In common usage, large number of files, many of which are very large in size are being migrated together, putting a significant load on DME servers and network resources.

There are other ways, much more efficient to migrate large amount of data between S3 storages that can be performed externally to DME.

## Scope

This new process of data migration begins w/ the storage team copying data from current S3 archive to the new desired S3 storage.

Once the data is copied, the DME team will call a new DME API to create a DME data migration task that will complete the migration. This migration process will identify the files that got copied by the storage team to the new S3 storage, and then update the relevant system metadata to ‘repoint’ DME data management to the new archive location of the files that migrated.

## Assumptions

1. Storage team will keep the same bucket/object key between the source and the destination archives as they copy data from current S3 storage to the new S3 storage
2. Storage team will copy the metadata on each data-object (uuid and user\_id) when transferring data from source to destination S3 archives.
3. The data has correctly copied from source to destination by the storage team.

## Constraints

1. Linking of DME metadata of data object will be performed utilizing AWS API (via the AWS S3 Java SDK) which is implemented by the vendor.

# Architecture Design

## Hardware Architecture

## Software Architecture

# Software Design

## Introduction

## Detailed Design

### APIs

The following changes and additions to APIs are proposed

#### Migrate Metadata

This is a new API with the purpose to request DME to complete migration for files that the storage team externally copied from current to new S3 archive location. DME will create a migration task in which system metadata is updated to reflect the new archive location

##### Request

POST /migrateMetadata

{

"fromS3ArchiveConfigurationId": "<from S3 Archive ID>",

"toS3ArchiveConfigurationId": "<to S3 Archive ID>",

"archiveFileContainerId": "<S3 file container ID (bucket)>",

"archiveFileIdPattern": "<Pattern to match object IDs. '%' match all files>"

}

##### Response

The response will return a migration task ID that will perform the metadata migration.

{

"taskId": "<task-id>"

}

##### Implementation

* All fields in the Request JSON except ‘archiveFileIdPattern‘ are mandatory.
* ‘archiveFileIdPattern’ will be defaulted to ‘%’ if not provided, to match all files in the bucket.
* The API will perform input validation and record a migration task in HPC\_DATA\_MIGRATION\_TASK table
* The column ‘TYPE’ will contain a new value for this migration task – BULK\_METADATA\_UPDATE
* New Columns will be added to store the values of ‘archiveFileContainerId’ and ‘archiveFileIdPattern’ received in the request JSON.

### Scheduled Tasks

The following changes are needed in the scheduled tasks to support metadata migration

#### processBulkMetadataUpdatetMigrationReceived()

This is a new scheduled task that is responsible to process a bulk metadata update task. The purpose of this schedule task is to identify all the files that are included in this bulk metadata update request and create a migration task for them

##### Implementation

* A metadata query will be performed to identify all the files that are included in this bulk metadata update request. Sample query JSON:  
  {

"compoundQuery": {

"operator": "AND",

"compoundQueries": [

{

"operator": "AND",

"queries": [

{

"attribute": "s3\_archive\_configuration\_id",

"value": "90020ee7-175b-4d20-bced-f39f3f637064",

"operator": "EQUAL"

},

{

"attribute": "archive\_file\_container\_id",

"value": "dme-dev-bucket",

"operator": "EQUAL"

},

{

"attribute": "archive\_file\_id",

"value": "%",

"operator": "LIKE"

}

]

}

]

},

"page": 1,

"totalCount": **true**

}

* For each data object found in the search, a migration task will be created to perform the actual update of the metadata for that data object
  + The ‘TYPE’ of the migration task will be DATA\_OBJECT\_METADATA\_UPDATE
  + All individual data object metadata update tasks will be linked to the bulk metadata update task
* Note that the search for data objects to be included in a metadata migration is performed against the materialized views. As a result, the materialized views need to be refreshed if the same request is sent consequently.

#### processDataObjectMetadataUpdatetMigrationReceived()

This is a new scheduled task. The purpose of this task is to perform the metadata update for a single data object.

##### Implementation

1. Read data object metadata that was set in the new archive by the storage team in the new archive.
2. Update the iRODs system metadata w/ new file location
3. A thread pool (task executor) will be used to enable concurrent execution of this task. The following property will control the pool size:  
   hpc.bus.dataObjectMetadataUpdateTaskExecutorThreadPoolSize

#### completeBulkMigrationInProgressTask()

This is an existing task that completes bulk migration tasks. A code change is required to include the bulk metadata update tasks (an additional query call) for checking progress through completion.

## User Interface Design

## System Integration

# Database Design

## Table Design

### HPC\_DATA\_MIGRATION\_TASK

New Columns will be added to store the values of ‘archiveFileContainerId’ and ‘archiveFileIdPattern’ received in the request JSON.

* METADATA\_ARCHIVE\_FILE\_CONTAINER\_ID
* METADATA\_ARCHIVE\_FILE\_ID\_PATTERN

### HPC\_DATA\_MIGRATION\_TASK\_RESULT

Same changes as HPC\_DATA\_MIGRATION\_TASK