Data Migrations from OldProd to Substrate

Document Level Classification

300

Introduction

This document covers data migrations for databases and other data services when migrating from OldProd to Substrate. Explore the sections below for use cases.

Database Migrations

If you are migrating your service from OldProd to Substrate and your service requires access to a database, a database migration will need to be performed. This section serves as guidance as to what is required to perform the migration as well as who to contact to perform the database migration.

Service Network

A Service Network connection is required to perform database migrations. Substrate environments should already have a Service Network connection set up to OldProd. However, If there is no Service Network connection set up, reach out to the #cloud-ops-support-ext slack channel.

MongoDB

MongoDB is currently managed entirely by the DBA team. If a MongoDB migration is required, a request can be made by reaching out to the **#data-ops-support-ext** slack channel.

For NON LIVE, some services may be using DocumentDB in AWS. In this case service owners should reach out to devops at the **#cloud-embedded-devops-ext slack** channel for assistance.

Elasticache for Redis

For best practices for migrating a Redis cluster from one account to another, OldProd to Substrate, refer to What are best practices for migrating my Elasticache for Redis cluster?

If assistance is needed reach out to devops at the **#cloud-embedded-devops-ext slack** slack channel.

Kafka

Kafka for Dev environments are managed by a 3rd party vendor called Confluent Cloud. In order to access your Kafka database from Substrate, a private link connection to your Substrate VPC would need to be created. Documentation for logging into Confluent Cloud and creating a cluster can be found here How to create Confluent KAFKA Cluster. This documentation also contains instructions for setting up the private link connection.

For Dev environments of Kafka using Amazon MSK, Apache Kafka Mirror Maker can be used to migrate data between clusters.

If addition help is needed reach out to the **#cloud-embedded-devops-ext slack** slack channel

RDS

Basic RDS Migrations

For basic RDS migrations with minimal uptime or cutover requirements, one can follow this official migration document from AWS: https://aws.amazon.com/premiumsupport/knowledge-center/account-transfer-rds/

To summarize here:

1. Create a DB snapshot.
2. Share the snapshot with the target account.
3. In the target account, create a new DB instance by restoring the DB snapshot.

This migration is adequate for services that can be brought down for the duration of the migration or have little traffic. Once the snapshot is created, no new writes to the old database will be included in the migration or snapshot.

Advanced RDS Migrations

For migrations with more stringent uptime or cutover requirements the biggest question becomes how to handle network connectivity between the accounts to allow for prolonged replication between the accounts as well as connectivity to the old and new databases from services in either account. These migrations will require a connection to the Service Network. Refer to the Service Network section.

DynamoDB

For basic DynamoDB migrations from OldProd to Substrate AWS documentation provides 2 scenarios for moving tables from one account to another. For documentation reference Migrating a DynamoDB table from one account to another

S3 Migrations

If you have data in an S3 bucket and that data needs to be migrated from OldProd to Substrate, S3 Replication along with S3 Batch Replication can be used for this purpose and will allow for transfer of existing objects and continuous replication. The pattern for migrating data from an S3 bucket

in OldProd to an S3 bucket in Substrate within the same region can be found below. The documentation is based on the AWS tutorial here Replicate Existing Objects in your Amazon S3 Buckets with Amazon S3 Batch Replication.

Note: While DataSync can be used for this purpose as well, there are cost implications and would only be used as a last resort. Using S3 Replication eliminates the data transfer cost as you don't pay for data transfer if it is within the same region. For example transferring from OldProd us-east-1 to a Substrate account in us-east-1. For more information on S3 pricing refer to Amazon S3 pricing

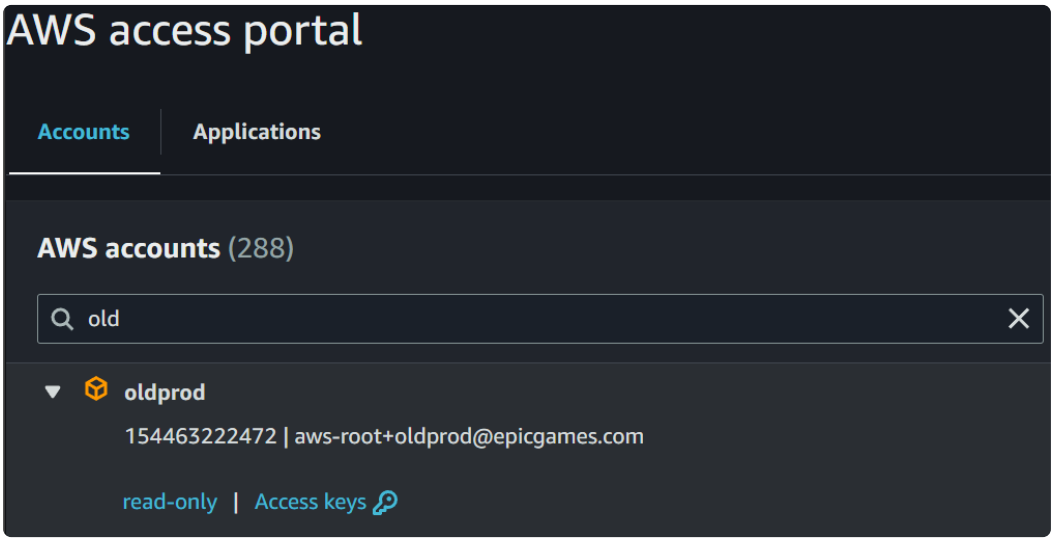
Click > to expand

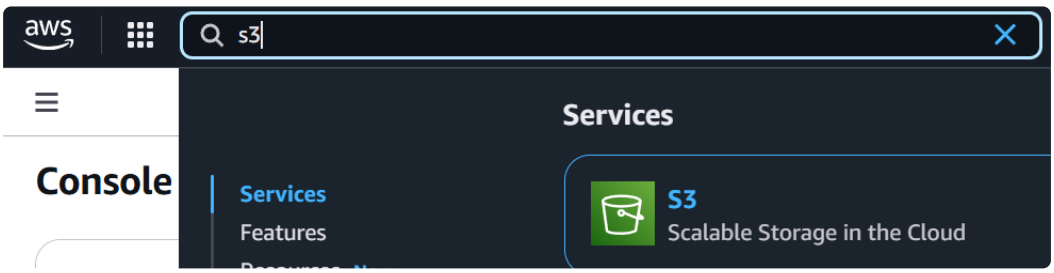
S3 Data Migration Using S3 Batch Operations (Manual Setup)

These are the steps to migrate data from an S3 Bucket in OldProd to an S3 Bucket in Substrate. In this process you first set up a replication rule. When it is created, replication will be live and replicate only new objects from source to destination bucket. The second part of the process is creating an S3 Batch Operations job that will replicate existing objects (backfill) from the source to destination bucket.

1. Create an S3 bucket in your substrate account if you have not already.
2. Enable versioning on both the source bucket in OldProd and the destination bucket in Substrate.
    1. This can be done either in your Terraform configuration or in the console.
    2. Link to Terraform example here
    3. In the AWS console this can be enabled on the **Properties** tab of the bucket.

3. Make sure the source and destination buckets are properly tagged.
    1. Reference the Cloud Governance Wiki for proper tagging guidelines.
4. Login to the Okta Dashboard.
5. Click on the **AWS** tile in the dashboard.
    - 

6. When the AWS access portal loads click on the **oldprod** account and choose the profile which you want to log in with. (Ex. **read-only**, **infra-support**, **devops**, etc...)
    - 

7. When the console loads, use the search bar to search for **S3.** Then click on **S3.**
    - 

8. From the list of buckets click on the name the bucket that will be your source bucket.

9. Click the **Management** tab.
10. In Management click **Create Replication Rule** and use the parameters below.
    1. **Replication Rule Configuration**
        1. **Replication rule name:** Enter a unique name for the rule
        2. **Status:** Enabled
    2. **Source Bucket**
        1. **Choose a rule scope**: Apply to all object in the bucket
    3. **Destination**
        1. **Specify a bucket in another account:** Selected
        2. **Account ID**: The AWS account ID for your Substrate account
        3. **Bucket name:** The name of the s3 bucket in your Substrate account.
        4. **Change object ownership to destination bucket owner:** Checked
    4. **IAM Role**
        1. **Create new role:** Selected
    5. **Additional replication options**
        1. **Delete marker replication:** Checked
    6. Leave all other page options at their defaults and click **Save**
    7. You will receive a popup asking to Replicate Existing Objects? For now choose **No, do not replicate existing objects.**
    8. Make note of the **IAM role** before moving to the next step.
    9. At this point in the process, since the status of the rule is **Enabled,** replication of new objects is live. However, in the next steps changes will be made to both the OldProd source IAM role and the bucket policy of the destination bucket in Substrate so replication will be successful.
11. Modify the IAM Role created by the configuration wizard.
    1. Open the AWS Search bar and search for IAM and then click on **IAM** in the search results.
    2. Click on **Roles** then search for the role created in step 10 and click on the role name.
    3. On the **Permissions** tab click **Add Permissions**->**Create Inline Policy**
    4. Click JSON in the policy editor to show in JSON format.

5. Copy the following policy into the editor, making sure to replace the tags <> with your source OldProd and destination Substrate bucket names.
6.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "s3:GetBucketObjectLockConfiguration",
            "Resource": [
                "arn:aws:s3:::<substrate-destination-bucket>"
            ]
        },
        {
            "Action": [
                "s3:GetReplicationConfiguration",
                "s3:PutInventoryConfiguration"
            ],
            "Effect": "Allow",
            "Resource": [
                "arn:aws:s3:::<oldprod-source-bucket>"
            ]
        },
        {
            "Effect": "Allow",
            "Action": "s3:PutObjectLegalHold",
            "Resource": [
                "arn:aws:s3:::<substrate-destination-bucket>/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:PutObject",
                "s3:InitiateReplication",
                "s3:GetObject",
```

```
                "s3:GetObjectVersion"
            ],
            "Resource": [
                "arn:aws:s3:::<oldprod-source-bucket>/*"
            ]
        }
    ]
}
```

7. Click **Next**
8. Enter a name for the policy then click **Create Policy.**
9. This will bring you back to the policy.

12. Modify the Trust Policy for the role created by the configuration wizard.
    1. Click on **Trust Relationship**
    2. Click **Edit trust policy**
    3. Overwrite the policy with the following policy.
    4.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": [
                    "s3.amazonaws.com",
                    "batchoperations.s3.amazonaws.com"
                ]
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

    5. Click **Update Policy.**

13. Modify the Destination Bucket Policy.
    1. In another tab, follow steps 4 through 7 above, but this time log into your destination Substrate account and go to S3.
    2. Locate and click on your destination bucket.
    3. Click **Permissions,** scroll down to **Bucket Policy,** then click **Edit.**
    4. Copy the following policy into the editor, making sure to replace the tags <> with your source OldProd bucket, destination Substrate bucket, and the name of the IAM role that was created in step 10.
    5.

```
{
    "Version": "2012-10-17",
    "Id": "",
    "Statement": [
        {
            "Sid": "Set-permissions-for-objects",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::154463222472:role/service-
            },
            "Action": [
                "s3:ReplicateObject",
                "s3:ReplicateDelete"
            ],
            "Resource": "arn:aws:s3:::<substrate-destination-bu
        },
        {
            "Sid": "Set permissions on bucket",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::154463222472:role/service-
            },
            "Action": [
                "s3:GetBucketVersioning",
                "s3:PutBucketVersioning"
            ],
```

```
            "Resource": "arn:aws:s3:::<substrate-destination-bu
        }
    ]
}
```

14. Go back to OldProd and go to your source S3 bucket then click **Management.**
15. Under **Replication Rules** click **View replication configuration** then click **Create Replication Job.** Use the following options for the job.
    1. **Manifest**
        1. **Create manifest using S3 Replication Configuration:** Selected
        2. **Destination bucket**: Bucket in another AWS Account
            1. **Account ID**: The AWS account ID for your Substrate account
            2. **Bucket name:** The name of the s3 bucket in your Substrate account.
    2. **Batch Operations Manifest**
        1. **Save Batch Operations Manifest:** Checked
        2. **Destination bucket**: Bucket in another AWS Account
            1. **Account ID**: The AWS account ID for your Substrate account
            2. **Bucket name:** The name of the s3 bucket in your Substrate account.
    3. **Operation**
        1. **Replicate:** Selected
    4. **Completion report**
        1. **Generate completion report:** Checked
        2. **Completion report scope**: All tasks
        3. **Path to completion report destination**: Your OldProd bucket
    5. **Permissions**
        1. **Choose from existing IAM roles:** Choose the role that was created in step 10.
    6. Leave all other page options at their defaults and click **Create Job**

16. Run the Job
    1. After clicking **Create Job** in the last step you will be brought to the **Batch Operations** home page.
    2. Next to the job you created the status should be **Awaiting your confirmation to run.** Click on the Job ID in the list of jobs.
    3. On the Job page, click **Run Job** to start the Batch Operations replication job.
    4. After clicking **Run Job** you will be brought to the **Batch Operations** home page
    5. Select the Job ID of your new job to review the job configuration or to check the status of the job.

S3 Data Migration Using S3 Batch Operations (Terraform Module)

A Terraform Module can be utilized to create the necessary configuration in OldProd. Additionally it provides instructions on how to create the necessary bucket policy needed in Substrate to give access to the bucket in OldProd to replicate objects. One caveat to using this module is there is currently no way to trigger the job from Terraform. To run the job the manual instructions to Run the Job are in the manual section above.

The Terraform Module can be found in the following repo https://github.ol.epicgames.net/terraform/s3-replication-module.

SQS Migrations

If you have messages in an SQS queue in OldProd that you need to be moved to another SQS queue Substrate, AWS provides documentation for a basic use case on using a Lambda function to consume the messages in one queue and move them to another. Documentation can be found here Tutorial: Using a cross-account Amazon SQS queue as an event source

SNS Migrations

Generally you wouldn't migrate SNS from one account to another. SNS topics deliver messages to subscribers such as an email address or an

SQS queue. After successful delivery the message is no longer available in SNS. For some background, publishers publish messages to SNS topics and then deliver the message to queue subscribers. For example if there are SQS queues subscribed to a topic, they are fanned out to the SQS queues where they are delivered (AKA pub/sub). Then messages in the queue are consumed by consumers. To migrate SNS messages that were delivered to a queue in OldProd to a new queue in Substrate, you would have to create new SNS topics in Substrate and subscribe your new SQS queues in Substrate to the new SNS topics. From there you could use the pattern in the **SQS Migrations** section above to move messages to Substrate.

Additionally, you could also set up a Dead Letter Queue (DLQ) for an SNS topic to hold messages that are undeliverable. For example, if you were to unsubscribe your SQS queues from the SNS Topic in OldProd, they would no longer be delivered to the queues. This would result in messages being undeliverable. To avoid those SNS messages being unavailable after retries, you could set up a Dead Letter Queue for the undeliverable messages to be stored. From there you would use the pattern in the **SQS Migrations** section above to move the messages to Substrate.