**Steps for Lambda pulling data from AWS OpenSearch and Uploading Data to S3**

1. **Account B (Bucket Owner):**
   1. **Create an S3 Bucket**: If you haven't already, create an S3 bucket in Account B where you want to allow uploads from Account A.
   2. **Define Bucket Policy**: Create a bucket policy for the S3 bucket in Account B that allows the Lambda function from Account A to upload objects. The policy should include a **Principal** element referencing the IAM role in Account A.

**Example**: {

"Version": "2012-10-17",

"Statement": [

{

"Effect": "Allow",

"Principal": {

"AWS": "arn:aws:iam::ACCOUNT-A-ID:role/lambda-role-in-account-a"

},

"Action": "s3:PutObject",

"Resource": "arn:aws:s3:::bucket-in-account-b/\*"

}

]

}

**Note:** Replace ACCOUNT-A-ID with the AWS account ID of Account A and lambda-role-in-account-a with the IAM role name in Account A.

1. **Account C (OpenSearch):**
   1. **OpenSearch Access Policy:** Configure the access policy for the OpenSearch domain in Account C to allow the Lambda function in Account A to access and perform actions on the OpenSearch domain.

**Example:** {

"Version": "2012-10-17",

"Statement": [

{

"Effect": "Allow",

"Principal": {

"AWS": "arn:aws:iam::ACCOUNT-A-ID:role/lambda-role-in-account-a"

},

"Action": [

"es:ESHttpGet”

],

"Resource": "arn:aws:es:us-east-1:ACCOUNT-C-ID:domain/your-opensearch-domain/\*"

}

]

}

**Note:** Please update below details in above Access Policy.

1. "**AWS":** "arn:aws:iam::ACCOUNT-A-ID:role/lambda-role-in-account-a": Replace ACCOUNT-A-ID with the AWS account ID of Account A and lambda-role-in-account-a with the IAM role name in Account A.
2. **"Resource":** "arn:aws:es:us-east-1:ACCOUNT-C-ID:domain/your-opensearch-domain/\*": Replace ACCOUNT-C-ID with the AWS account ID of Account C and your-opensearch-domain with the name of your OpenSearch domain.
3. **"Action":** The list of OpenSearch actions that the Lambda function is allowed to perform. You can adjust this list based on the actions your Lambda function needs to perform.
4. **Account A (Lambda Executor):**
   1. **Create IAM Role**: Create an IAM role in Account A that the Lambda function will assume to perform the S3 upload. This role should have permissions to assume the cross-account role in Account B.
   2. **Assume Cross-Account Role**: Attach a trust policy to the IAM role in Account A that allows it to assume the cross-account role in Account B.

**Example**: {

"Version": "2012-10-17",

"Statement": [

{

"Effect": "Allow",

"Principal": {

"Service": "lambda.amazonaws.com"

},

"Action": "sts:AssumeRole",

"Condition": {

"StringEquals": {

"sts:ExternalId": "external-id"

}

}

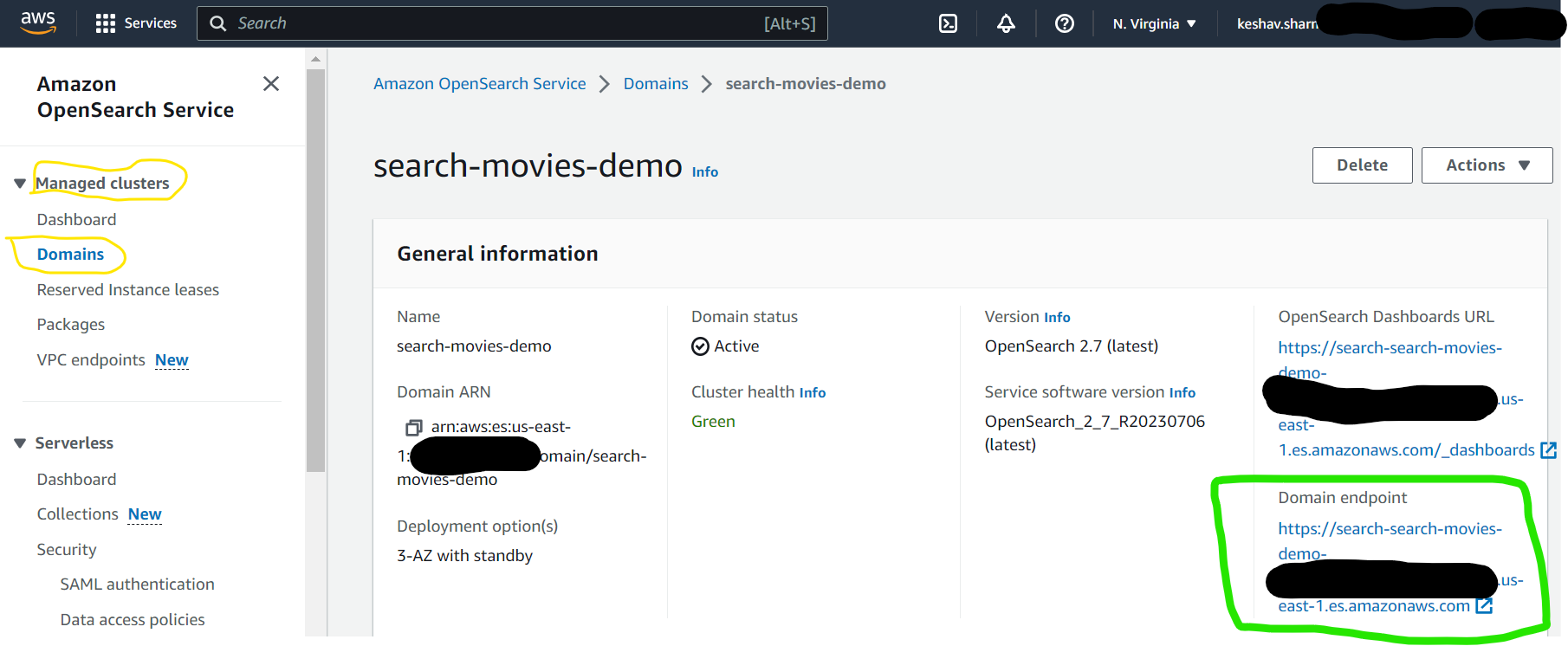
}

]

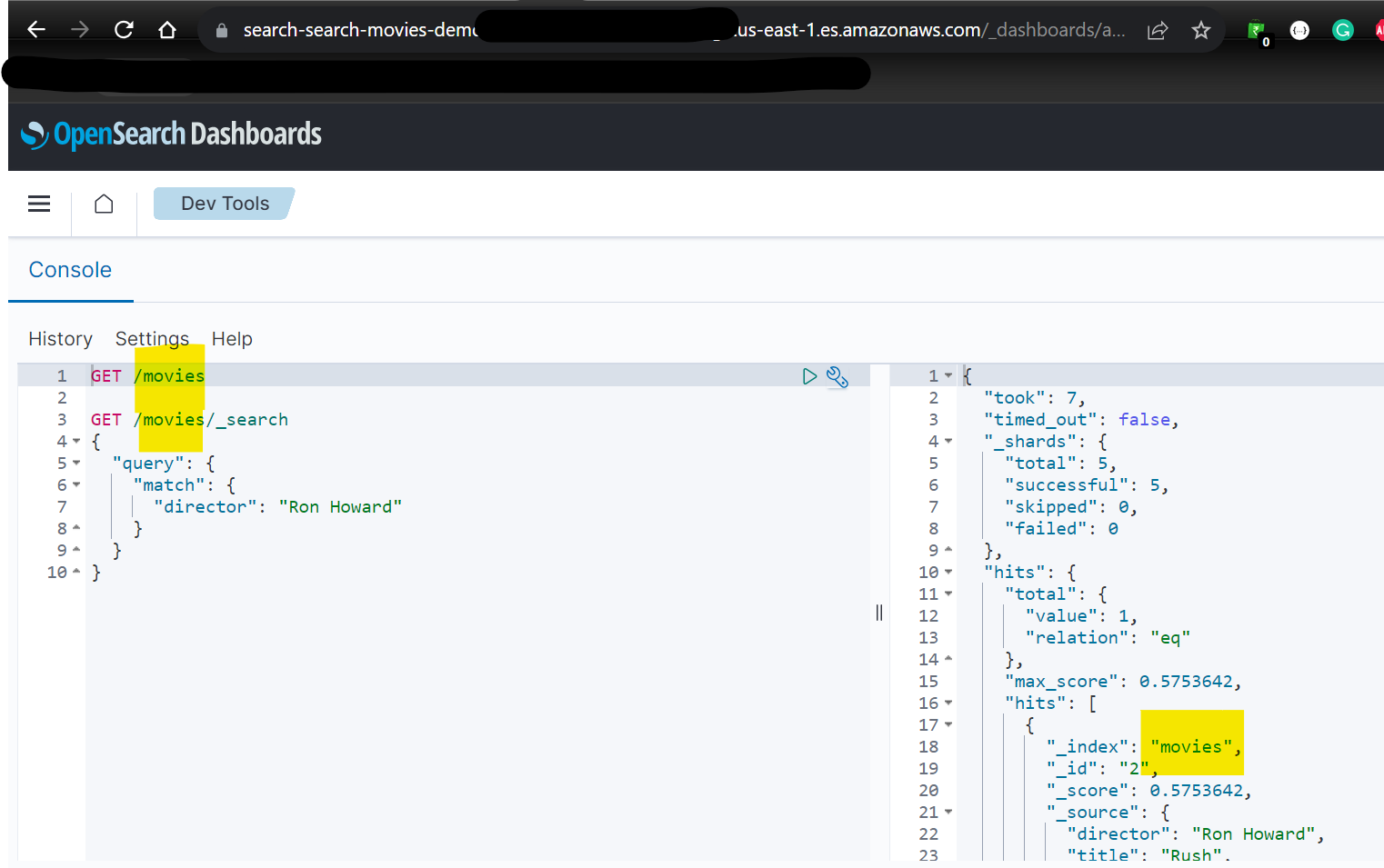
}

**Note:** Replace **"external-id"** with the external ID that you set when you created the cross-account role in Account B.

1. **Lambda Code property changes**
   1. **BeanConfiguration.java:**
      1. aws.opensearch.domain.endpoint //Endpoint from Account C OpenSearch Service Domain endpoint **Example:** search-search-movies-demo-{uniqueId}.us-east-1.es.amazonaws.com for below setup.



* + 1. aws.region // Region in which Account C OpenSearch Service is deployed. **Example:** us-east-1 for above setup.
  1. **OpenSearchService.java:**
     1. aws.opensearch.index.name



* 1. **S3Service.java:**
     1. aws.s3.opensearch.data.file.name // Expected filename in S3

1. **Deploy Lambda to AWS Account A**