How to-

Migrate Data from One Elasticsearch to Another (Cross Account)

elasticsearch

Steps to Follow-

Create a role with Elasticsearch's permission. You may also use the existing role with the following trust relationship,

```
{
  "Effect": "Allow",
  "Principal": {
      "Service": "es.amazonaws.com"
  },
  "Action": "sts:AssumeRole"
}
```

Provide the **iam:PassRole** for the IAM user whose access/secret keys will be used to take a snapshot.

```
{
  "Version": "2012-10-17",
  "Statement": {
      "Effect": "Allow",
      "Action": "iam:PassRole",
      "Resource": "arn:aws:iam::accountID:role/TheServiceRole"
  }
}
```

Change the access & secret key, host, region, path, and payload in the below code and execute it.

```
import requests
from requests_aws4auth import AWS4Auth

AWS_ACCESS_KEY_ID=''
AWS_SECRET_ACCESS_KEY=''
region = 'us-west-1'
service = 'es'

awsauth = AWS4Auth(AWS_ACCESS_KEY_ID, AWS_SECRET_ACCESS_KEY, region, service)
host = 'https://elasticsearch-domain.us-west-1.es.amazonaws.com/' # include
https:// and trailing /

# REGISTER REPOSITORY
path = '_snapshot/my-snapshot-repo' # the Elasticsearch API endpoint
url = host + path

payload = {
  "type": "s3",
  "settings": {
      "bucket": "s3-bucket-name",
      "region": "us-west-1",
      "role_arn": "arn:aws:iam::accountID:role/TheServiceRole"
  }
}

headers = {"Content-Type": "application/json"}
r = requests.put(url, auth=awsauth, json=payload, headers=headers) #
requests.get, post, put, and delete all have similar syntax
print(r.text)
```

To take the snapshot and store it in the S3

```
path = '_snapshot/my-snapshot-repo/my-snapshot'
url = host + path
r = requests.put(url, auth=awsauth)
print(r.text)
```

Now the snapshot is ready. Share this snapshot to another account and use the same code with new account keys and endpoint to restore it using the below code snippet.

To restore all indices from the snapshot

```
path = '_snapshot/my-snapshot-repo/my-snapshot/_restore'
url = host + path
r = requests.post(url, auth=awsauth)
print(r.text)
```

To restore single index from the snapshot

```
path = '_snapshot/my-snapshot-repo/my-snapshot/_restore'
url = host + path
payload = {"indices": "my-index"}
headers = {"Content-Type": "application/json"}
r = requests.post(url, auth=awsauth, json=payload, headers=headers)
print(r.text)
```

Also, there are two EC2 instances (currently in stopped mode) available in the account -

Both the server have script.py on the root directory with all the permissions assigned. Modify the script.py in order to create a snapshot of the indices and restore it to the other account. I Will share the ec2 key such that you can access the servers and use the contents.

curl -XPUT '<source ElasticSearch host name > / _ snapshot/my-snapshot-repo/<name of snapshot>

Use the above command on the source EC2 server to create a snapshot after running the script, to create a snapshot and store it in the destination bucket as specified by the script.