

Elasticsearch

Best Practices

Best Practices

Cloud data import

Import data from Alibaba Cloud to Alibaba Cloud ES (offline)

Alibaba Cloud stores an abundance of cloud storage and database products. If you want to analyze and search for data in these products, visit **Data Integration**, which allows you to synchronize offline data to Elasticsearch every 5 minutes.

Supported data source

- Alibaba Cloud database (MySQL, PostgreSQL, SQL Server, PPAS, MongoDB, and HBase)
- Alibaba Cloud DRDS
- Alibaba Cloud MaxCompute (ODPS)
- Alibaba Cloud OSS
- Alibaba Cloud Table Store
- Self-developed HDFS, Oracle, FTP, DB2, and self-developed versions of the previous cloud databases

Note:

Data synchronization may produce public network traffic cost.

Procedure

Take the following steps to import offline data.

- Prepare an ECS instance that can interact with Elasticsearch within a VPC. This ECS instance will obtain data sources and execute a job to write ES data (the job is centrally issued by Data Integration).
- You need to activate the Data Integration service and register the ECS instance to the Data

Integration service as an executable job resource.

- Configure a data synchronization script and make it run periodically.

Steps

Buy an ECS instance that is in the same VPC as the Elasticsearch service. Allocate a public IP address to the ECS instance or enable the elastic IP address for the ECS instance. To lower costs, you can use an existing ECS instance. For how to buy an ECS instance, see [Buy an ECS instance](#).

Note:

CentOS 6, CentOS 7, and AliyunOS are recommended.

If the added ECS instance needs to run MaxCompute or synchronization tasks, verify whether the current Python version of the ECS instance is 2.6 or 2.7 (The Python version of CentOS 5 is 2.4 while those of other operating systems are later than 2.6).

- Ensure that the ECS instance has a public IP address.

Log on to the **Data Integration console** to open the workbench.

If Data Integration or DataWorks is not enabled, follow the instructions to activate the Data Integration service. This is a **paid service**, so check the quoted price against your budget.

Go to the **Project Management-Scheduling Resource Management** page of the Data Integration service to configure the ECS instance in the VPC as a scheduling resource. For more information, see [Scheduling resource configuration](#).

Configure the data synchronization script in the Data Integration service. For the configuration procedure, see [Data synchronization script configuration](#).

For the instructions on configuring Elasticsearch, see [ES Writer](#).

Note:

- The synchronization script configuration includes three parts: Reader is the configuration of upstream data source (cloud product ready for data synchronization), Writer is the configuration of ES, and setting refers to the synchronization configurations such as packet loss rate and maximum concurrency.

- The `accessId` and `accessKey` of ES Writer are the Elasticsearch user name and password, respectively.

After configuring the script, submit the data synchronization job. Set the job execution cycle and click **Ok**.

Note:

- If you are configuring a periodic scheduling, set the parameters such as Job Start Time, Execution Interval, and Job Lifecycle in this window.
- A periodic job is executed at 00:00 on the next day according to the rule you have configured.

After the submission, go to the **O&M Center-Task Scheduling** page to find the submitted job, and change its scheduling resource from default to the scheduling resource you have configured.

Import real-time data

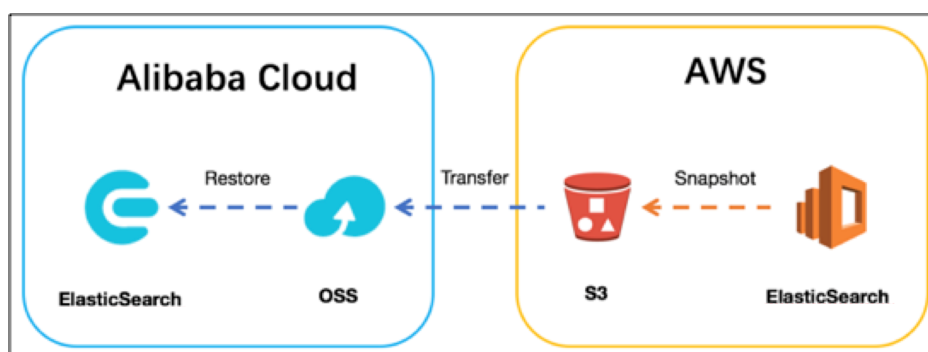
This function is currently under development and will become available in the future.

Elasticsearch index migration from AWS to Alibaba Cloud

Abstract

This document describes Elasticsearch (ES) index migration from AWS to Alibaba Cloud.

A reference architecture diagram is shown below:



Introduction

Concepts

- **Elasticsearch:** A distributed, RESTful search and analytics engine capable of solving a growing number of use cases. As the heart of the Elastic Stack, it centrally stores your data so you can discover the expected and uncover the unexpected.
- **Kibana:** Lets you visualize your Elasticsearch data and navigate the Elastic Stack.
- **Amazon Elasticsearch Service:** It's easy to deploy, secure, operate, and scale Elasticsearch for log analytics, full text search, application monitoring, and more. Amazon Elasticsearch Service is a fully managed service that delivers Elasticsearch' s easy-to-use APIs and real-time analytics capabilities alongside the availability, scalability, and security that production workloads require.
- **Alibaba Elasticsearch Service:** Alibaba Cloud' s Elasticsearch service. In this guide, we explain how to use Elasticsearch through our Alibaba Cloud China site. Have not onboard on International site.

Snapshot and Restore: You can store snapshots of individual indexes or an entire cluster in a remote repository like a shared file system, S3, or HDFS. These snapshots are great for backups because they can be restored relatively quickly. However, snapshots can only be restored to versions of Elasticsearch that can read the indexes:

- A snapshot of an index created in 5.x can be restored to 6.x.
- A snapshot of an index created in 2.x can be restored to 5.x.
- A snapshot of an index created in 1.x can be restored to 2.x.

Conversely, snapshots of indexes created in 1.x cannot be restored to 5.x or 6.x, and snapshots of indexes created in 2.x cannot be restored to 6.x.

Snapshots are incremental and can contain indexes created in various versions of Elasticsearch. If any indexes in a snapshot were created in an incompatible version, you will not be able restore the snapshot.

Solution overview

Elasticsearch (ES) indexes can be migrated with following steps:

Step 1: Create baseline indexes

1. Create a snapshot repository and associate it to an AWS S3 Bucket.

Create the first snapshot of the indexes to be migrated, which is a full snapshot.

The snapshot will be automatically stored in the AWS S3 bucket created in the first step.

3. Create an OSS Bucket on Alibaba Cloud, and register it to a snapshot repository of an Alibaba Cloud ES instance.
4. Use the OSSImport tool to pull the data from the AWS S3 bucket into the Alibaba Cloud OSS bucket.
5. Restore this full snapshot to the Alibaba Cloud ES instance.

Step 2: Periodic incremental snapshots

Repeat several incremental snapshot and restore.

Step 3: Final snapshot and service switchover

1. Stop services which can modify index data.
2. Create a final incremental snapshot of the AWS ES instance.
3. Transfer and restore the final incremental snapshot to an Alibaba Cloud ES instance.
4. Perform service switchover to the Alibaba Cloud ES instance.

Prerequisites

Elasticsearch service

- The version number of AWS ES is **5.5.2**, located in the Singapore region.
- The version number of Alibaba Cloud ES is **5.5.3**, located in Hangzhou.
- The demo index name is **movies**.

Manual Snapshot Prerequisites on AWS

Amazon ES takes daily automated snapshots of the primary index shards in a domain, and stores these automated snapshots in a preconfigured Amazon S3 bucket for 14 days at no additional charge to you. You can use these snapshots to restore the domain.

You cannot, however, use automated snapshots to migrate to new domains. Automated snapshots are read-only from within a given domain. **For migrations, you must use manual snapshots stored in**

your own repository (an S3 bucket). Standard S3 charges apply for manual snapshots.

To create and restore index snapshots manually, you must work with IAM and Amazon S3. Verify that you have met the following prerequisites before you attempt to take a snapshot.

Prerequisite	Description
S3 bucket	Stores manual snapshots for your Amazon ES domain.
IAM role	Delegates permissions to Amazon Elasticsearch Service. The trust relationship for the role must specify Amazon Elasticsearch Service in the Principal statement. The IAM role also is required to register your snapshot repository with Amazon ES. Only IAM users with access to this role may register the snapshot repository.
IAM policy	Specifies the actions that Amazon S3 may perform with your S3 bucket. The policy must be attached to the IAM role that delegates permissions to Amazon Elasticsearch Service. The policy must specify an S3 bucket in a Resource statement.

S3 bucket

You need an S3 bucket to store manual snapshots. Make a note of its Amazon Resource Name (ARN). You need it for the following:

- Resource statement of the IAM policy that is attached to your IAM role.
- Python client that is used to register a snapshot repository.

The following example shows an ARN for an S3 bucket:

```
arn:aws:s3:::eric-es-index-backups
```

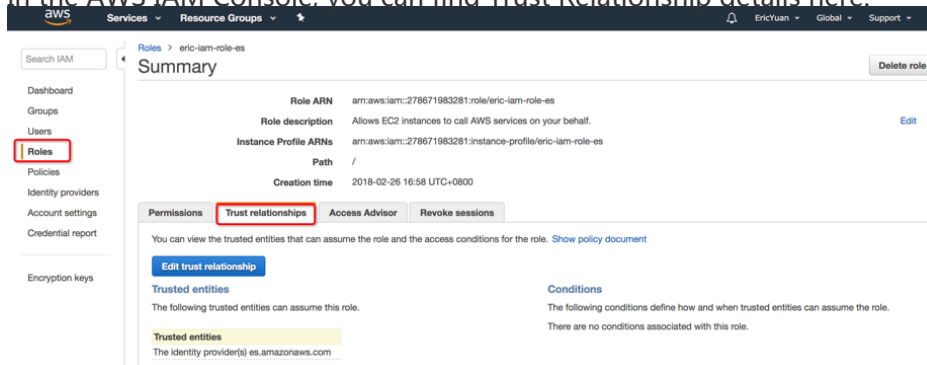
IAM role

You must have a role that specifies Amazon Elasticsearch Service, `es.amazonaws.com`, in a **ServiceStatement** in its trust relationship, as shown in the following example:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "",
      "Effect": "Allow",
      "Principal": {
        "Service": "es.amazonaws.com"
      },
    },
  ],
}
```

```
"Action": "sts:AssumeRole"
}
]
}
```

In the AWS IAM Console, you can find Trust Relationship details here:



Edit Trust Relationship

You can customize trust relationships by editing the following access control policy document.

Policy Document

```
1 {
2   "Version": "2012-10-17",
3   "Statement": [
4     {
5       "Sid": "",
6       "Effect": "Allow",
7       "Principal": {
8         "Service": "es.amazonaws.com"
9       },
10      "Action": "sts:AssumeRole"
11    }
12  ]
13 }
```

When you create an AWS service role by using the IAM Console, Amazon ES is not included in the **Select role type** list. However, you can still create the role by choosing **Amazon EC2**, following the steps to create the role, and then editing the role's trust relationships to `es.amazonaws.com` instead of `ec2.amazonaws.com`.

IAM Policy

You must attach an **IAM policy** to the **IAM role**. The policy specifies the S3 bucket that is used to store manual snapshots for your Amazon ES domain. The following example specifies the ARN of the `eric-es-index-backups` bucket:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
```



```

"Action": [
  "s3:ListBucket"
],
"Effect": "Allow",
"Resource": [
  "arn:aws:s3:::eric-es-index-backups"
]
},
{
  "Action": [
    "s3:GetObject",
    "s3:PutObject",
    "s3:DeleteObject"
  ],
  "Effect": "Allow",
  "Resource": [
    "arn:aws:s3:::eric-es-index-backups/*"
  ]
}
]
}

```

You need to paste it in here:

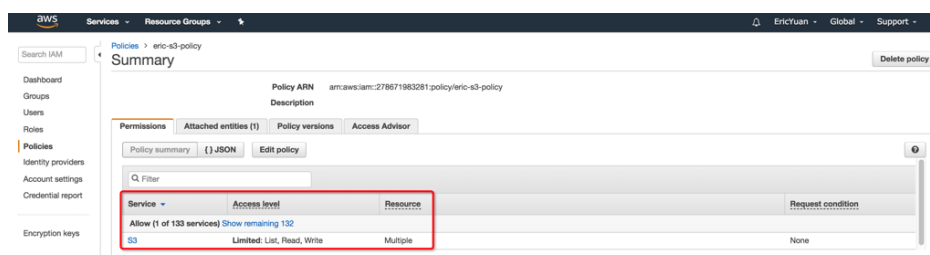
The screenshot shows the AWS IAM console interface. In the left-hand navigation menu, the 'Policies' option is highlighted. The main content area displays the 'Summary' page for a policy named 'eric-s3-policy'. The 'Permissions' tab is selected, showing the policy's JSON definition. The JSON is as follows:

```

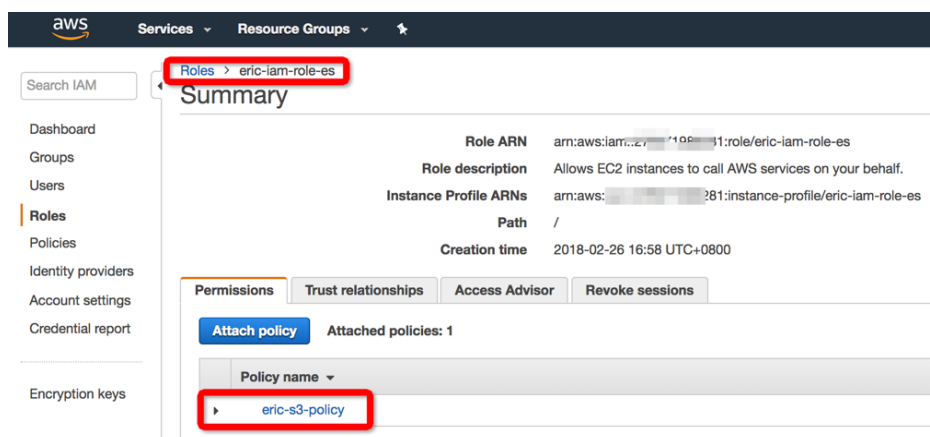
1 {
2   "Version": "2012-10-17",
3   "Statement": [
4     {
5       "Action": [
6         "s3:ListBucket"
7       ],
8       "Effect": "Allow",
9       "Resource": [
10        "arn:aws:s3:::eric-es-index-backups"
11      ]
12    },
13    {
14      "Action": [
15        "s3:GetObject",
16        "s3:PutObject",
17        "s3:DeleteObject"
18      ],
19      "Effect": "Allow",
20      "Resource": [
21        "arn:aws:s3:::eric-es-index-backups/*"
22      ]
23    }
24  ]
25 }

```

You can make sure the policy is correct by looking at the **Policy summary**, as follows:



Attach IAM Policy to IAM Role



Registering a manual snapshot directory

You must register the snapshot directory with Amazon Elasticsearch Service before you can take manual index snapshots. This one-time operation requires that you sign your AWS request with credentials for one of the users or roles specified in the IAM role's trust relationship, as described in [Section Manual snapshot prerequisites on AWS](#).

You can't use curl to perform this operation because it doesn't support AWS request signing. Instead, use the sample Python client to register your snapshot directory.

Modify sample python client

Download a copy of the file "Sample Python Client.docx", then modify the values in yellow in that document to match your real values. Copy the contents of "Sample Python Client.docx" into a Python file called "snapshot.py" after you've finished editing.

Sample Python Client.docx

Variable name	Description
region	AWS Region where you created the snapshot repository
host	Endpoint for your Amazon ES domain
aws_access_key_id	IAM credential
aws_secret_access_key	IAM credential

path	Name of the snapshot repository
data: bucket; region; role_arn	<p>Must include the name of the S3 bucket and the ARN for the IAM role that you created in Section Manual snapshot prerequisites on AWS . To enable server-side encryption with S3-managed keys for the snapshot repository, add "server_side_encryption": true to the settings JSON.</p> <p>Important</p> <p>If the S3 bucket is in the us-east-1 region, you need to use "endpoint": "s3.amazonaws.com" in place of "region": "us-east-1".</p>

Install Amazon Web Services Library boto-2.48.0

This sample Python client requires that you install version 2.x of the **boto** package on the computer where you register your snapshot repository.

```
# wget
https://pypi.python.org/packages/66/e7/fe1db6a5ed53831b53b8a6695a8f134a58833cadb5f2740802bc3730ac15/boto-2.48.0.tar.gz#md5=ce4589dd9c1d7f5d347363223ae1b970
# tar xzvf boto-2.48.0.tar.gz
# cd boto-2.48.0
# python setup.py install
```

Execute Python client to register snapshot directory

```
# python snapshot.py
```

Registering Snapshot Repository

Check result in **Kibana**->**Dev Tools** with request:

GET _snapshot



Snapshot and restore for the first time

Take a snapshot manually on AWS ES

The following commands are all performed on **Kibana->Dev Tools**, you can also perform them using **curl** from the Linux or Mac OSX command line.

- Take a snapshot with the name *snapshot_movies_1* only for the index **movies** in the repository *eric-snapshot-repository*.

```
PUT _snapshot/eric-snapshot-repository/snapshot_movies_1
{
  "indexes": "movies"
}
```

- Check snapshot status

```
GET _snapshot/eric-snapshot-repository/snapshot_movies_1
```

The screenshot shows the Kibana Dev Tools console. On the left, the 'Console' tab is active, displaying a series of commands: a PUT command to create a snapshot named 'snapshot_movies_1' for the 'movies' index, followed by a GET command to check its status. On the right, the response for the GET command is shown as a JSON object. The 'snapshot' field is highlighted with a red box, showing details like 'uid', 'version_id', and 'version'. The 'indices' field is also highlighted, showing the 'movies' index. The 'state' is 'SUCCESS', and the 'start_time_in_millis' is highlighted with a red box.

- Check snapshot files on the AWS S3 console

The screenshot shows the AWS S3 console for the bucket 'eric-es-index-backups'. The 'Overview' tab is selected, showing a list of objects. The objects include 'indices', 'incompatible-snapshots', 'index-0', 'index.latest', 'meta-BigKlVgoSpSgwBhbD4hTWg.dat', and 'snap-BigKlVgoSpSgwBhbD4hTWg.dat'. The 'snap-BigKlVgoSpSgwBhbD4hTWg.dat' object is highlighted with a red box, indicating it is the snapshot file.

Name	Last modified	Size	Storage class
indices	--	--	--
incompatible-snapshots	Feb 28, 2018 11:00:47 AM GMT+0800	29.0 B	Standard
index-0	Feb 28, 2018 11:00:47 AM GMT+0800	178.0 B	Standard
index.latest	Feb 28, 2018 11:00:47 AM GMT+0800	8.0 B	Standard
meta-BigKlVgoSpSgwBhbD4hTWg.dat	Feb 28, 2018 11:00:45 AM GMT+0800	337.0 B	Standard
snap-BigKlVgoSpSgwBhbD4hTWg.dat	Feb 28, 2018 11:00:47 AM GMT+0800	228.0 B	Standard

Pull snapshot data from AWS S3 to Alibaba Cloud OSS

In this step, you need to pull snapshot data from your AWS S3 bucket into Alibaba Cloud OSS.

Refer to [Migrate data from Amazon S3 to Alibaba Cloud OSS](#) After data transfer, check stored snapshot data from the OSS console:

eric-oss-aws-es-snapshot-s3				类型	标准存储	区域	华东 1	创建时间	2018-02-26 18:40
概览 文件管理 基础设置 域名管理 图片处理 事件通知 函数计算 基础数据 热点统计 API 统计 文件访问统计									
上传文件 新建目录 删除 设置 HTTP 头 碎片管理 刷新									
<input type="text" value="输入文件名前缀"/>									
<input type="checkbox"/>	文件名 (Object Name)	文件大小	存储类型	更新时间					
<input type="checkbox"/>	indices/								
<input type="checkbox"/>	incompatible-snapshots	0.028KB	标准存储	2018-02-26 11:06					
<input type="checkbox"/>	index-0	0.174KB	标准存储	2018-02-26 11:06					
<input type="checkbox"/>	index.latest	0.0080KB	标准存储	2018-02-26 11:06					
<input type="checkbox"/>	meta-BigKLVgoSpSgwBhbD4hTWg.dat	0.329KB	标准存储	2018-02-26 11:06					
<input type="checkbox"/>	snap-BigKLVgoSpSgwBhbD4hTWg.dat	0.223KB	标准存储	2018-02-26 11:06					

Restore snapshot to an Alibaba Cloud ES instance

Create snapshot repository

Perform the following request on **Kibana**->**Dev Tools** to create a snapshot repository with the same name: modify values as follows to match your real values.

```
PUT _snapshot/eric-snapshot-repository
{
  "type": "oss",
  "settings": {
    "endpoint": "http://oss-cn-hangzhou-internal.aliyuncs.com",
    "access_key_id": "Put your access key id here.",
    "secret_access_key": "Put your secret access key here.",
    "bucket": "eric-oss-aws-es-snapshot-s3",
    "compress": true
  }
}
```



After creating the snapshot directory, check the snapshot status for the snapshot named **snapshot_movies_1**, which was assigned in AWS ES manual snapshot step.

```
GET _snapshot/eric-snapshot-repository/snapshot_movies_1
```



Note: Please record the start time and end time of this snapshot operation: It will be used when you transfer incremental snapshot data with the Alibaba Cloud OSSimport tool. For example:

"start_time_in_millis" : 1519786844591

"end_time_in_millis" : 1519786846236

Restore snapshots

Perform the following request on **Kibana->Dev Tools**.

```

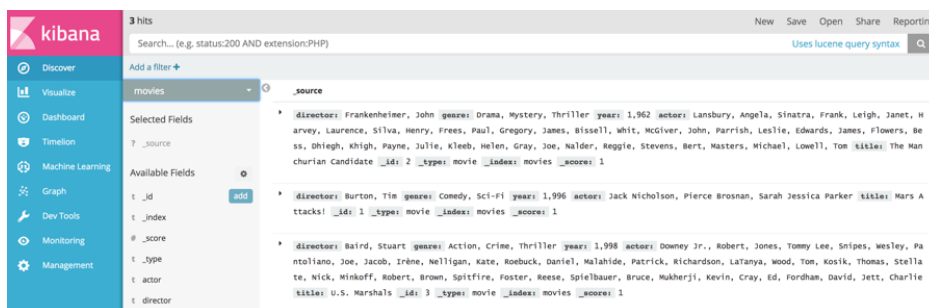
POST _snapshot/eric-snapshot-repository/snapshot_movies_1/_restore
{
  "indices": "movies"
}

GET movies/_recovery

```



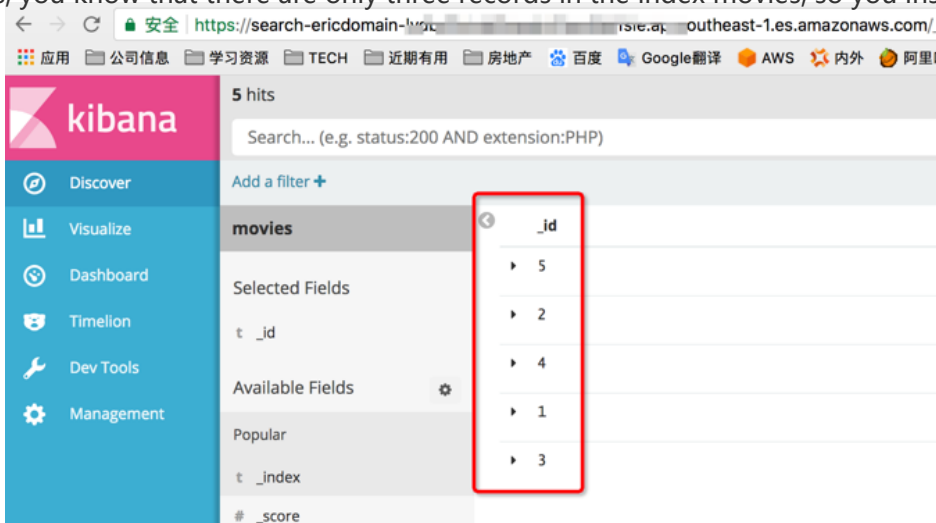
Check the availability of index movies on **Kibana->Dev Tools**, you can see there exist three records in the index movies, the number of records on your AWS ES instance.



Snapshot and restore for the last time

Create some sample data on AWS ES index movies

In the previous steps, you know that there are only three records in the index movies, so you insert



another two records.

You could also see the number of indexes using this request: GET movies/_count.



Take another snapshot manually

Refer to Section [Take a snapshot manually on AWS ES](#), then check the snapshot status:

```

1 get movies/_count
2
3 PUT _snapshot/eric-snapshot-repository/snapshot_movies_1
4 {
5   "indices": "movies"
6 }
7
8 GET _snapshot/eric-snapshot-repository/snapshot_movies_1
9
10 PUT _snapshot/eric-snapshot-repository/snapshot_movies_2
11 {
12   "indices": "movies"
13 }
14
15 GET _snapshot/eric-snapshot-repository/snapshot_movies_2
16
17
18 delete movies
19
20 GET _snapshot
21 GET _snapshot/eric-snapshot-repository
22 GET _snapshot/eric-snapshot-repository/snapshot_movies_1
23 GET _snapshot/eric-snapshot-repository/snapshot_movies_2
24
25 delete _snapshot/eric-es-index-backups
26 delete _snapshot/eric-snapshot-repository/snapshot_movies_1
27 delete _snapshot/eric-snapshot-repository/snapshot_movies_2

```

```

1 {
2   "snapshots": [
3     {
4       "snapshot": "snapshot_movies_2",
5       "uuid": "CWhIF7ShQZaKQJasPE70A",
6       "version_id": 5050299,
7       "version": "5.5.2",
8       "indices": [
9         "movies"
10      ],
11      "state": "SUCCESS",
12      "start_time": "2018-02-28T03:55:33.505Z",
13      "start_time_in_millis": 1519790133505,
14      "end_time": "2018-02-28T03:55:35.195Z",
15      "end_time_in_millis": 1519790135195,
16      "duration_in_millis": 1690,
17      "failures": [],
18      "shards": {
19        "total": 5,
20        "failed": 0,
21        "successful": 5
22      }
23    }
24  ]
25 }

```

Check the files listed in the S3 bucket:

Viewing 1 to 9				
<input type="checkbox"/> Name	Last modified	Size	Storage class	
<input type="checkbox"/> indices	--	--	--	
<input type="checkbox"/> snap-CWhIF7ShQZaKQJasPE70A.dat	Feb 28, 2018 11:55:36 AM GMT+0800	228.0 B	Standard	
<input type="checkbox"/> index.latest	Feb 28, 2018 11:55:36 AM GMT+0800	8.0 B	Standard	
<input type="checkbox"/> index-1	Feb 28, 2018 11:55:36 AM GMT+0800	274.0 B	Standard	
<input type="checkbox"/> meta-CWhIF7ShQZaKQJasPE70A.dat	Feb 28, 2018 11:55:34 AM GMT+0800	337.0 B	Standard	
<input type="checkbox"/> snap-BIqKLvgoSpSgwBhbD4hTWg.dat	Feb 28, 2018 11:00:47 AM GMT+0800	228.0 B	Standard	
<input type="checkbox"/> index-0	Feb 28, 2018 11:00:47 AM GMT+0800	178.0 B	Standard	
<input type="checkbox"/> incompatible-snapshots	Feb 28, 2018 11:00:47 AM GMT+0800	29.0 B	Standard	
<input type="checkbox"/> meta-BIqKLvgoSpSgwBhbD4hTWg.dat	Feb 28, 2018 11:00:45 AM GMT+0800	337.0 B	Standard	

If you check the folder **indexes**, you can also find some differences.

Pull incremental snapshot data from AWS S3 to Alibaba Cloud OSS

You can use the OSSImport tool to migrate data from S3 to OSS. Because there are 2 snapshot files stored in our S3 bucket now, we try to migrate only new files by modifying the value of **isSkipExistFile** in the configuration file **local_job.cfg**.

Filed	Meaning	Description
isSkipExistFile	Whether to skip the existing objects during data migration, a Boolean value.	<p>If it is set to true, the objects are skipped according to the size and LastModifiedTime.</p> <p>If it is set to false, the existing objects are overwritten. The default value is false. This option is invalid when jobType is set to audit.</p>

After the OSSImport migration job completes, you can see only 'new' files are migrated to OSS.

In your Alibaba Cloud OSS bucket:

文件名 (Object Name)	文件大小	存储类型	更新时间	操作
indices/				
incompatible-snapshots	0.028KB	标准存储	2018-02-28 11:06	设置
index-0	0.174KB	标准存储	2018-02-28 11:06	设置
index-1	0.268KB	标准存储	2018-02-28 14:06	设置
index_latest	0.0080KB	标准存储	2018-02-28 14:07	设置
meta-BlgKLvgoSpSgwBhbD4hTWg.dat	0.329KB	标准存储	2018-02-28 11:06	设置
meta-CWhIF7ShQZaKQJasPE70A.dat	0.329KB	标准存储	2018-02-28 14:07	设置
snap-BlgKLvgoSpSgwBhbD4hTWg.dat	0.223KB	标准存储	2018-02-28 11:06	设置
snap-CWhIF7ShQZaKQJasPE70A.dat	0.223KB	标准存储	2018-02-28 14:07	设置

Name	Last modified	Size	Storage class
indices	--	--	--
incompatible-snapshots	Feb 28, 2018 11:00:47 AM GMT+0800	29.0 B	Standard
index-0	Feb 28, 2018 11:00:47 AM GMT+0800	178.0 B	Standard
index-1	Feb 28, 2018 11:55:36 AM GMT+0800	274.0 B	Standard
index_latest	Feb 28, 2018 11:55:36 AM GMT+0800	8.0 B	Standard
meta-BlgKLvgoSpSgwBhbD4hTWg.dat	Feb 28, 2018 11:00:45 AM GMT+0800	337.0 B	Standard
meta-CWhIF7ShQZaKQJasPE70A.dat	Feb 28, 2018 11:55:34 AM GMT+0800	337.0 B	Standard
snap-BlgKLvgoSpSgwBhbD4hTWg.dat	Feb 28, 2018 11:00:47 AM GMT+0800	228.0 B	Standard
snap-CWhIF7ShQZaKQJasPE70A.dat	Feb 28, 2018 11:55:36 AM GMT+0800	228.0 B	Standard

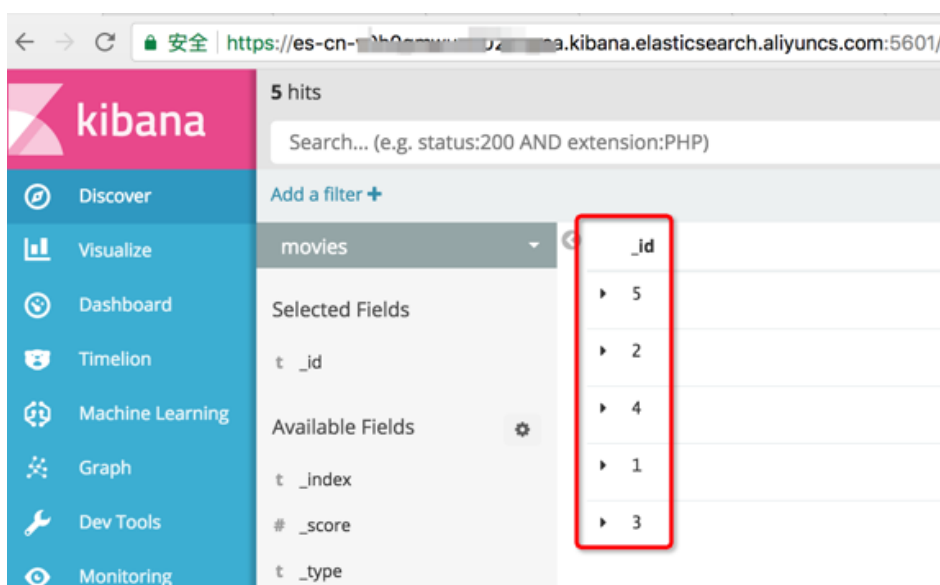
In our AWS S3 bucket:

Restore an incremental snapshot

You can follow along with the steps from Section **Restore snapshots**, but the index **movies** needs to be closed firstly, then you have to restore the snapshot, and open the index again after restore:

```
POST /movies/_close
GET movies/_stats
POST _snapshot/eric-snapshot-repository/snapshot_movies_2/_restore
{
  "indexes": "movies"
}
POST /movies/_open
```

After the restore procedure completes, you can see the count (5) of documents in the index **movies** is the same as it is in our AWS ES instance.



Conclusion

It is possible to migrate AWS Elasticsearch service data to Alibaba Cloud's Elasticsearch service by the snapshot and restore method.

This solution requires that the AWS ES instance is stopped first to prevent writes and requests during migration.

Further reading:

- <https://www.elastic.co/products/elasticsearch>
- <https://www.elastic.co/guide/en/elasticsearch/reference/current/modules-snapshots.html>
- <https://docs.aws.amazon.com/elasticsearch-service/latest/developerguide/es-manageddomains-snapshots.html>
- <https://www.alibabacloud.com/help/doc-detail/64919.htm?spm=a3c0i.l31815en.b99.105.67c25139Y3tgnc>
- <https://github.com/zhichen/elasticsearch-repository-oss/wiki/OSS%E5%BF%AB%E7%85%A7%E8%BF%81%E7%A7%BB?spm=a2c4g.11186623.2.3.2acd85>
- <https://github.com/zhichen/elasticsearch-repository-oss/wiki/OSS%E5%BF%AB%E7%85%A7%E8%BF%81%E7%A7%BB?spm=a2c4g.11186623.2.3.wfCX30>
- <https://www.alibabacloud.com/help/doc-detail/56990.htm?spm=a3c0i.o56990en.a1.4.5baa6605O1C9yZ>

Logstash deployment

Prepare the environment

Buy Alibaba Cloud ES instances and ECS instances that can access self-built clusters and Alibaba Cloud ES. If you already have ECS instances that meet the requirements, there is no need to purchase additional ECS instances. Prepare the JDK of version 1.8 or later.

The ECS instance on a classic network can be used as long as the ECS instance can access the Alibaba Cloud ES service within VPC through Classiclink.

Download Logstash v5.5.3.

Download the Logstash of the version matching Elasticsearch on the Elastic website (v5.5.3 is recommended).

Decompress the downloaded Logstash package.

```
tar -xzf logstash-5.5.3.tar.gz
# A stringent configuration file checking feature is added to Elasticsearch later than version 5.x.
```

Test cases

Create the user name and password for data access

Create a role

```
curl -u elastic:es-password -XPOST
http://***instanceId***.elasticsearch.aliyuncs.com:9200/_xpack/security/role/***role-name*** -d '{"cluster":
["manage_index_templates", "monitor"], "indices": [{"names": [ "logstash-*" ],
"privileges":["write", "delete", "create_index"]}]}'

# es-password is the Kibana logon password.
# ***instanceId*** is the ES instance ID.
# ***role-name*** is the role name.
# The default index name of Logstash is in the format of logstash-current date. Therefore, the read and write
permissions on the Logstash-* index must be assigned when you add a user role.
```

Create a user

```
curl -u elastic:es-password -XPOST
http://***instanceId***.elasticsearch.aliyuncs.com:9200/_xpack/security/user/***user-name*** -d '{"password" :
```

```

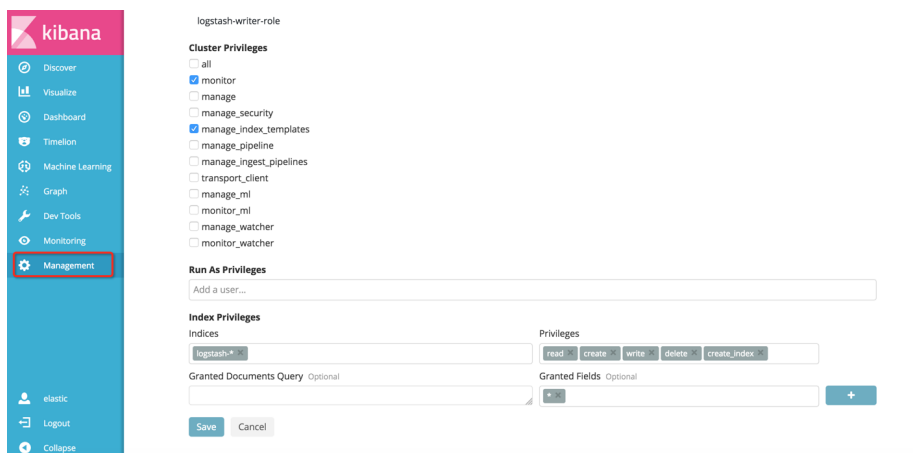
"***logstash-password***", "roles" : [ "***role-name***", "full_name" : "your full name" }'
# es-password is the Kibana logon password.
# ***instanceId*** is the ES instance ID.
# ***user-name*** is the user name for data access.
# ***logstash-password*** is the password for data access.

```

Note:

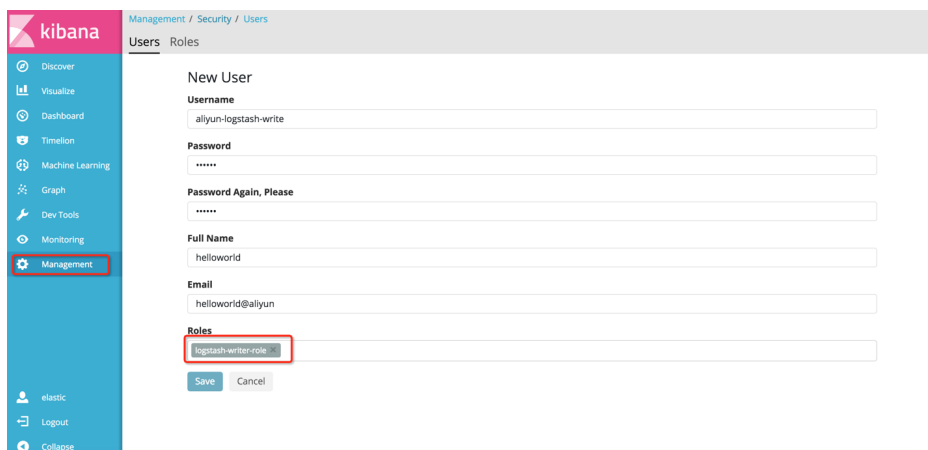
The role and user can also be created on the Kibana page.

Add a role



The screenshot shows the Kibana Management interface. On the left sidebar, the 'Management' tab is selected. The main content area displays the configuration for the 'logstash-writer-role'. Under 'Cluster Privileges', the 'monitor' and 'manage_index_templates' checkboxes are checked. The 'Run As Privileges' section has a text input field for 'Add a user...'. The 'Index Privileges' section shows a list of indices with a 'logstash-*' filter. The 'Privileges' section includes buttons for 'read', 'create', 'write', 'delete', and 'create_index'. The 'Granted Documents Query' and 'Granted Fields' sections are optional and currently empty. At the bottom, there are 'Save' and 'Cancel' buttons.

Add a user



The screenshot shows the Kibana Management interface, specifically the 'New User' form. The left sidebar shows the 'Management' tab selected. The main content area has a breadcrumb trail 'Management / Security / Users' and tabs for 'Users' and 'Roles'. The 'New User' form includes fields for 'Username' (filled with 'aliyun-logstash-write'), 'Password' (masked with dots), 'Password Again, Please' (masked with dots), 'Full Name' (filled with 'helloworld'), and 'Email' (filled with 'helloworld@aliyun'). The 'Roles' section shows a dropdown menu with 'logstash-writer-role' selected. At the bottom, there are 'Save' and 'Cancel' buttons.

Prepare the conf file.

For more information, see Configuration file structure.

Example

Create the test.conf file on the ECS instance and add the following configurations.

```
input {
  file {
    path => "/your/file/path/xxx"
  }
}
filter {
}
output {
  elasticsearch {
    hosts => ["http://***instanceId***.elasticsearch.aliyuncs.com:9200"]
    user => "***user-name***"
    password => "***logstash-password***"
  }
}
```

instanceId is the ES instance ID.
user-name is the user name for data access.
logstash-password is the password for data access.
Place the user name and password in quotation marks to prevent errors in Logstash startup caused by special characters.

Run

Run Logstash according to the conf file.

```
bin/logstash -f path/to/your/test.conf
```

Logstash provides many input, filter, and output plugins. Only simple configurations are required for data transfer. This example shows how to obtain file changes through Logstash and submit the changed data to the Elasticsearch cluster. All the new contents in the monitored file can be automatically indexed to the Elasticsearch cluster by Logstash.

FAQ

How to configure the index automatically created by the cluster?

To ensure security during users' data operations, Alibaba Cloud Elasticsearch does not allow automatic creation of indexes by default.

Logstash creates indexes by submitting data in data upload, instead of using the create index API. Therefore, before using Logstash to upload data, allow the automatic creation of indexes.

Note:

After the setting is changed and confirmed, the Alibaba ES cluster restarts.

No permissions to create indexes?

Check whether the role you created for data access has the write, delete, and create_index permissions.

Insufficient memory?

By default, Logstash has a 1 GB memory. If your requested ECS memory becomes insufficient, reduce the memory usage of Logstash by changing the memory settings in config/jvm.options.

No quotation marks added to the user name and password in test.conf configuration?

If the user name or password containing special characters in the test.conf file are not added to quotation marks, the previous error message is displayed.

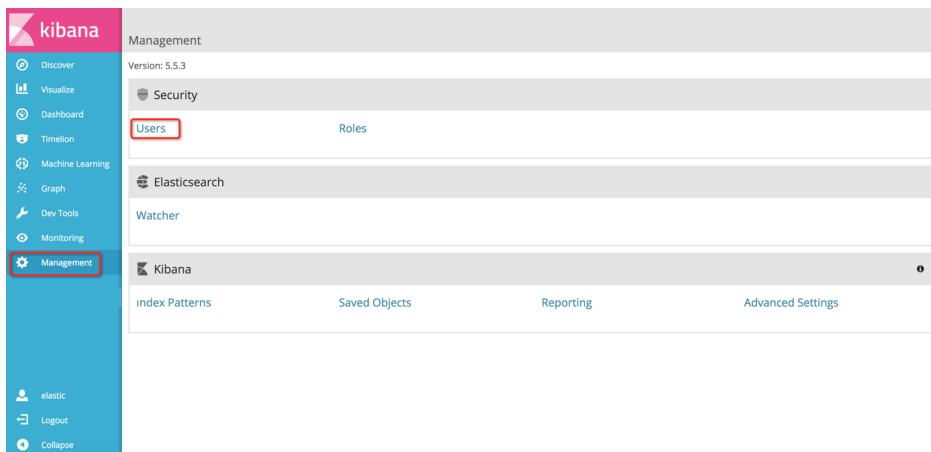
Additional information

To monitor the Logstash node and collect logs:

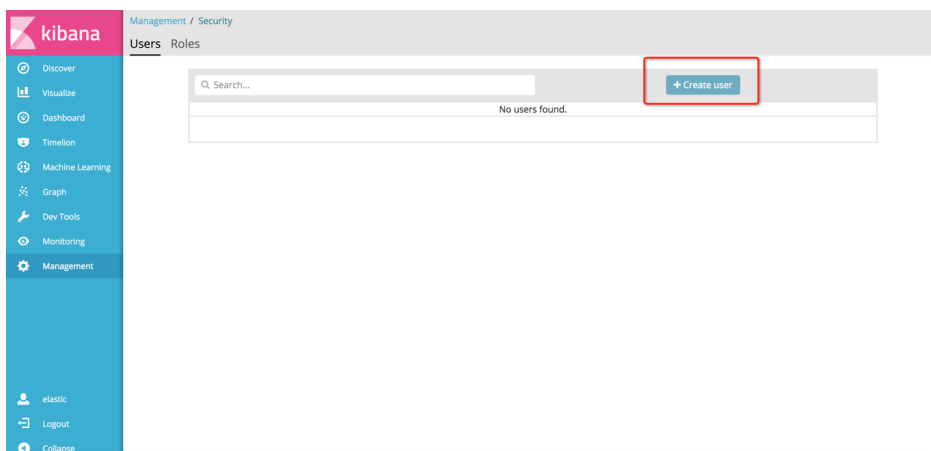
- Install the X-Pack plugin for Logstash. For more information, see [download link](#).
- Deploy the X-Pack after download.
- bin/logstash-plugin install file:///path/to/file/x-pack-5.5.3.zip.
- Add a Logstash monitor user. Alibaba Cloud Elasticsearch cluster disables the logstash_system user by default. You need to create a user with the role name logstash_system. The user name cannot be logstash_system. The user name can be changed. In this example, the user name is logstash_system_monitor. The following two methods are recommended for creating users:

Create a monitor user through the Kibana module

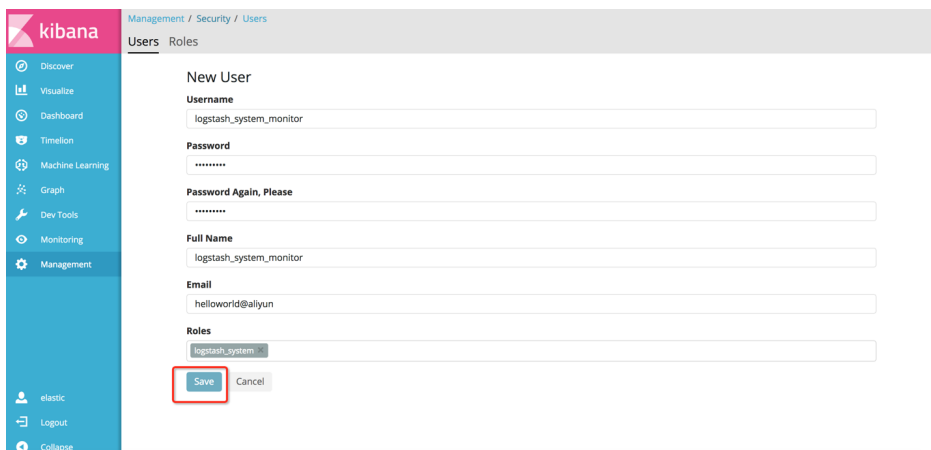
Log on to the Kibana management page, and perform the operations according to the following figure:



Click the **Create User** button.



Enter the required information. Save and submit the information.



Add a user through commands

```
curl -u elastic:es-password -XPOST
```

```
http://***instanceId***.elasticsearch.aliyuncs.com:9200/_xpack/security/user/logstash_system_monitor -d
'{"password" : "***logstash-monitor-password***", "roles" : ["logstash_system"], "full_name" : "your full name"}'

# es-password is the Kibana logon password.
# ***instanceId*** is the ES instance ID.
# ***logstash-monitor-password*** is the password of logstash_system_monitor.
```

ES self-built migration

Prerequisites

You must meet the following requirements to migrate data from a user-created Elasticsearch instance to an Alibaba Cloud Elasticsearch instance.

-The ECS instance that hosts the user-created Elasticsearch instance must be connected to a VPC network. **ECS instances connected to a VPC network through a ClassicLink are not supported. The ECS instance and your Alibaba Cloud Elasticsearch instance must be connected to the same VPC network.

-You can use an ECS instance to run the `reindex.sh` script. To perform this task, you must make sure that the ECS instance can access port 9200 on the user-created and Alibaba Cloud Elasticsearch instances.

-The VPC security group must allow all IP addresses in the IP whitelist to access the ECS instance and port 9200 must be open.

-The VPC security group must allow the IP addresses of all Elasticsearch instance nodes to access the ECS instance. You can view these IP addresses in the Kibana console.

-To check whether the ECS instance that runs the script can access port 9200 on the source and target Elasticsearch instances, run the `curl -XGET http://<host>:9200` command on the ECS instance.

Procedure

1. Create indexes.
2. Migrate data.

Create indexes

You must create indexes on the target Elasticsearch instance based on the indexes on the source cluster. You can also choose to enable dynamic index creation and dynamic mapping (not

recommended) to create indexes on the target cluster. You must enable auto index creation before you enable dynamic index creation.

The following section provides a Python script (indiceCreate.py). You can copy all the indexes from the source cluster to the target cluster. Only the number of shards and zero replica are configured. You need to configure the remaining settings.

Note:

If the following error occurs when you run the cURL command, add the -H "Content-Type: application/json" parameter to the command and run the command again.

```
{"error": "Content-Type header [application/x-www-form-urlencoded] is not supported", "status": 406}
```

```
// Obtain all the indexes on the source cluster. If you do not have the required permissions, remove the "-u
user:pass" parameter. Make sure that you have replaced oldClusterHost with the name of the ECS instance that
hosts the source cluster.
curl -u user:pass -XGET http://oldClusterHost/_cat/indices | awk '{print $3}'

// Based on the returned indexes, obtain the setting and mapping of the index that you need to migrate for the
specified user. Make sure that you have replaced indexName with the index name that you need to query.
curl -u user:pass -XGET http://oldClusterHost/indexName/_settings,_mapping?pretty=true

// Create a new index in the target cluster according to the _settings and _mapping settings that you have obtained
from the preceding step. You can set the number of index replicas to zero to accelerate the data synchronization
process, and change the number to one after the migration has completed.
// ewClusterHost indicates the ECS instance that hosts the target cluster, testindex indicates the name of the index
that you have created, and testtype indicates the type of the index.
curl -u user:pass -XPUT http://<newClusterHost>/<testindex> -d '{
  "testindex" : {
    "settings" : {
      "number_of_shards" : "5", //Set the number of shards for the corresponding index on the source cluster, for
example, 5
      "number_of_replicas" : "0" //Set the number of index replicas to zero
    }
  },
  "mappings" : { //Set the mapping for the index on the source cluster. For example, you can set the mapping as
follows
    "testtype" : {
      "properties" : {
        "uid" : {
          "type" : "long"
        },
        "name" : {
          "type" : "text"
        },
        "create_time" : {
          "type" : "long"
        }
      }
    }
  }
}
```

```
}  
}'
```

Accelerate the synchronization process

Note:

If the index is too large, you can set the number of replicas to 0 and the refresh interval to -1 before migration. After the data has been migrated, set the replicas and refresh settings to the previous values. This accelerates the synchronization process.

```
// You can set the number of index replicas to zero and disable refresh, to accelerate the migration process.  
curl -u user:password -XPUT 'http://<host:port>/indexName/_settings' -d' {  
  "number_of_replicas" : 0,  
  "refresh_interval" : "-1"  
'  
  
// After the data has been migrated, set the number of index replicas to `1` and the refresh interval to `1` (default  
value, which means 1 second).  
curl -u user:password -XPUT 'http://<host:port>/indexName/_settings' -d' {  
  "number_of_replicas" : 1,  
  "refresh_interval" : "1s"  
'
```

Data migration

To ensure data consistency after the migration, you must **stop the write operation on the source cluster**. You do not need to stop the read operation. After the migration process has been completed, switch the read and write operations to the target cluster. Data inconsistency may occur if you do not stop the write operation on the source cluster.

Note:-When using the following method to migrate data, if you access the source cluster using an IP address and a port, you must configure a reindex whitelist in the YML file of the target cluster, and add the IP address of the source cluster to the whitelist:reindex.remote.whitelist:
1.1.1.1:9200,1.2.*.*

-If you access the source cluster using a domain name, do not use the http://host:port/path format. The domain name must not contain the path.

1. Migrate small amounts of data

Run the reindex.sh script.

```

#!/bin/bash
# file:reindex.sh

indexName="The name of the index"
Newclusteruser = "The username that is used to log on to the target cluster"
Newclusterpass = "The password that is used to log on to the target cluster"
Newclusterhost = "The ECS instance that hosts the target cluster"
Oldclusteruser = "The username that is used to log on to the source cluster"
Oldclusterpass = "The password that is used to log on to the source cluster"
# Set oldClusterHost in the format of [scheme]://[host]:[port]. Example: http://10.37.1.1:9200.
Oldclusterhost = "The ECS instance that hosts the source cluster"

curl -u ${newClusterUser}:${newClusterPass} -XPOST "http://${newClusterHost}/_reindex? pretty" -H "Content-Type:
application/json" -d'{
"source": {
"remote": {
"host": "${oldClusterHost}",
"username": "${oldClusterUser}",
"password": "${oldClusterPass}"
},
"index": "${indexName}",
"query": {
"match_all": {}
}
},
"dest": {
"index": "${indexName}"
}
}'

```

2. Migrate large amounts of data without delete operations and with update time

If the amount of data is large without deletion operations, you can use rolling migration to minimize the time period during which your write operation is suspended. Rolling migration requires that your data schema has a time-series attribute that indicates the update time. You can stop the write operation after the data has been migrated, then migrate the incremental data. Switch the read and write operations to the target cluster.

```

#!/bin/bash
# file: circleReindex.sh
# CONTROLLING STARTUP:
# This script is used to remotely rebuild the index using the reindex operation. Requirements:
#1. You have created the index on the target cluster, or the target cluster supports automatic index creation and
dynamic mapping.
# 2. You must configure an IP whitelist in the YAML file of the target cluster: reindex.remote.whitelist: 172.16.123.
*:9200
#3. You need to specify the ECS instance address in the following format: [scheme]://[host]:[port].

USAGE="Usage: sh circleReindex.sh <count>"
count: The number of executions. A negative number indicates loop execution. You can set this parameter to
perform the reindex operation only once or multiple times.

```

Example:

```
sh circleReindex.sh 1
sh circleReindex.sh 5
sh circleReindex.sh -1"
```

```
indexName="The name of the index"
newClusterUser="The username that is used to log on to the target cluster"
newClusterPass="The password that is used to log on to the target cluster"
oldClusterUser="The username that is used to log on to the source cluster"
oldClusterPass="The password that is used to log on to the source cluster"
## http://myescluster.com
newClusterHost="The host of the target cluster"
# You need to address of the ECS instance that hosts the source cluster in the following format:
[scheme]://[host]:[port]. Example: http://10.37.1.1:9200
oldClusterHost="The ECS instance that hosts the source cluster"
timeField="The field that specifies the time window during which the incremental data is migrated"
```

```
reindexTimes=0
lastTimestamp=0
curTimestamp=`date +%s`
hasError=false
```

```
function reIndexOP() {
reindexTimes=$((reindexTimes) + 1)
curTimestamp=`date +%s`
```

```
ret=`curl -u ${newClusterUser}:${newClusterPass} -XPOST "${newClusterHost}/_reindex? pretty" -H "Content-Type:
application/json" -d '{
"source": {
"remote": {
"host": "${oldClusterHost}",
"username": "${oldClusterUser}",
"password": "${oldClusterPass}"
},
"index": "${indexName}",
"query": {
"range": {
"${timeField}": {
"gte": "${lastTimestamp}",
"lt": "${curTimestamp}"
}
}
}
},
"dest": {
"index": "${indexName}"
}
}'`
lastTimestamp=${curTimestamp}
echo "${reindexTimes} reindex operations have been performed. The last reindex operation is completed at
${lastTimestamp} Result:${ret}"
if [[ ${ret} == *error* ]]; then
hasError=true
echo "An unknown error occurred while performing this operation. All subsequent operations have been
suspended."
```

```

fi
}

function start() {
## A negative number indicates loop execution.
if [[ $1 -lt 0 ]]; then
while :
do
reIndexOP
done
elif [[ $1 -gt 0 ]]; then
k=0
while [[ k -lt $1 ]] && [[ ${hasError} == false ]]; do
reIndexOP
let ++k
done
fi
}

## main
if [ $# -lt 1 ]; then
echo "$USAGE"
exit 1
fi

echo "Start the reindex operation for index ${indexName}"
start $1
echo "You have performed ${reindexTimes} reindex operations"

```

3. Migrate large amounts of data without deletion operations or update time

When you need to migrate large amounts of data and no update time field is defined in the mapping, you must add a update time field to the code that is used to access the source cluster. After the field has been added, you can migrate the existing data, and then use rolling migration described in the preceding data migration plan to migrate the incremental data.

The following script shows how to migrate the existing data without the update time field.

```

#!/bin/bash
# file:miss.sh

indexName="The name of the index"
newClusterUser="The username that is used to log on to the target cluster"
newClusterPass="The password that is used to log on to the target cluster"
newClusterHost="The ECS instance that hosts the target cluster"
oldClusterUser="The username that is used to log on to the source cluster"
oldClusterPass="The password that is used to log on to the source cluster"
# The address of the ECS instance that hosts the source cluster must be in this format: [scheme]://[host]:[port].
Example: http://10.37.1.1:9200.
oldClusterHost="The ECS instance that hosts the source cluster"
timeField="updatetime"

```

```
curl -u ${newClusterUser}:${newClusterPass} -XPOST "http://${newClusterHost}/_reindex? pretty" -H "Content-Type: application/json" -d '{
  "source": {
    "remote": {
      "host": "${oldClusterHost}",
      "username": "${oldClusterUser}",
      "password": "${oldClusterPass}"
    },
    "index": "${indexName}",
    "query": {
      "bool": {
        "must_not": {
          "exists": {
            "field": "${timeField}"
          }
        }
      }
    }
  },
  "dest": {
    "index": "${indexName}"
  }
}'
```

4. Migrate data without suspending the write operation

This feature will soon be available.

Use the batch creation operation to replicate indexes from the source cluster

The following Python script shows how to replicate indexes from the source cluster to the target cluster. The default number of newly created index replicas is 0.

```
#!/usr/bin/python
# -*- coding: UTF-8 -*-
# File name: indiceCreate.py

import sys
import base64
import time
import httplib
import json

## The ECS instance that hosts the source cluster (ip+port)
oldClusterHost = "old-cluster.com"
# The username that is used to log on to the source cluster. The username field can be left empty
oldClusterUserName = "old-username"
## The password that is used to log on to the source cluster. The password field can be left empty
oldClusterPassword = "old-password"
```

```

## The ECS instance that hosts the target cluster (ip+port)
newClusterHost = "new-cluster.com"
## The username that is used to log on to the target cluster. The username field can be left empty
newClusterUser = "new-username"
## The password that is used to log on to the target cluster. The password field can be left empty
newClusterPassword = "new-password"

DEFAULT_REPLICAS = 0

def httpRequest(method, host, endpoint, params="", username="", password=""):
    conn = httplib.HTTPConnection(host)
    headers = {}
    if (username != "") :
        'Hello {name}, your age is {age} !'.format(name = 'Tom', age = '20')
        base64string = base64.encodestring('{username}:{password}'.format(username = username, password =
        password)).replace('\n', '')
        headers["Authorization"] = "Basic %s" % base64string;
    if "GET" == method:
        Content-Type: application/x-www-form-urlencoded
        conn.request(method=method, url=endpoint, headers=headers)
    else :
        Headers ["Content-Type"] = "application/JSON"
        conn.request(method=method, url=endpoint, body=params, headers=headers)

    response = conn.getresponse()
    res = response.read()
    return res

def httpGet(host, endpoint, username="", password=""):
    return httpRequest("GET", host, endpoint, "", username, password)

def httpPost(host, endpoint, params, username="", password=""):
    return httpRequest("POST", host, endpoint, params, username, password)

def httpPut(host, endpoint, params, username="", password=""):
    return httpRequest("PUT", host, endpoint, params, username, password)

def getIndices(host, username="", password=""):
    endpoint = "/_cat/indices"
    indicesResult = httpGet(oldClusterHost, endpoint, oldClusterUserName, oldClusterPassword)
    indicesList = indicesResult.split("\n")
    indexList = []
    for indices in indicesList:
        if (indices.find("open") > 0):
            indexList.append(indices.split()[2])

    return indexList

def getSettings(index, host, username="", password=""):
    endpoint = "/" + index + "/_settings"
    indexSettings = httpGet(host, endpoint, username, password)
    print index + " The original settings: \n" + indexSettings
    settingsDict = json.loads(indexSettings)
    ## The number of shards equals the number of indexes on the source cluster by default
    number_of_shards = settingsDict[index]["settings"]["index"]["number_of_shards"]
    ## The default number of replicas is 0

```

```

number_of_replicas = DEFAULT_REPLICAS
newSetting = "\"settings\": {\"number_of_shards\": %s, \"number_of_replicas\": %s}\" % (number_of_shards,
number_of_replicas)
return newSetting

def getMapping(index, host, username="", password=""):
    endpoint = "/" + index + "/_mapping"
    indexMapping = httpGet(host, endpoint, username, password)
    print index + "The original mappings: \n" + indexMapping
    mappingDict = json.loads(indexMapping)
    mappings = json.dumps(mappingDict[index]["mappings"])
    newMapping = "\"mappings\": " + mappings
    return newMapping

def createIndexStatement(oldIndexName):
    settingStr = getSettings(oldIndexName, oldClusterHost, oldClusterUserName, oldClusterPassword)
    mappingStr = getMapping(oldIndexName, oldClusterHost, oldClusterUserName, oldClusterPassword)
    createstatement = "{\n" + str(settingStr) + ",\n" + str(mappingStr) + "\n}"
    return createstatement

def createIndex(oldIndexName, newIndexName=""):
    if (newIndexName == "") :
        newIndexName = oldIndexName
    createstatement = createIndexStatement(oldIndexName)
    print "new index" + newIndexName + "settings and mappings: \n" + createstatement
    endpoint = "/" + newIndexName
    createResult = httpPut(newClusterHost, endpoint, createstatement, newClusterUser, newClusterPassword)
    print "new index" + newIndexName + "creation result:" + createResult

## main
indexList = getIndices(oldClusterHost, oldClusterUserName, oldClusterPassword)
systemIndex = []
for index in indexList:
    if (index.startswith(".")):
        systemIndex.append(index)
    else :
        createIndex(index, index)
if (len(systemIndex) > 0) :
    for index in systemIndex:
        print index + "It may be a system index that will not be recreated. Create the index based on your needs."

```

Note:

You can use Logstash to migrate data. For more information, see [Logstash deployment](#).