

Cloud Search Service

User Guide

Issue 04
Date 2023-06-20



Copyright © Huawei Technologies Co., Ltd. 2023. All rights reserved.

No part of this document may be reproduced or transmitted in any form or by any means without prior written consent of Huawei Technologies Co., Ltd.

Trademarks and Permissions



HUAWEI and other Huawei trademarks are trademarks of Huawei Technologies Co., Ltd.

All other trademarks and trade names mentioned in this document are the property of their respective holders.

Notice

The purchased products, services and features are stipulated by the contract made between Huawei and the customer. All or part of the products, services and features described in this document may not be within the purchase scope or the usage scope. Unless otherwise specified in the contract, all statements, information, and recommendations in this document are provided "AS IS" without warranties, guarantees or representations of any kind, either express or implied.

The information in this document is subject to change without notice. Every effort has been made in the preparation of this document to ensure accuracy of the contents, but all statements, information, and recommendations in this document do not constitute a warranty of any kind, express or implied.

Contents

1 Overview.....	1
1.1 What Is Cloud Search Service?.....	1
1.2 Advantages.....	2
1.3 Product Components.....	3
1.4 Scenarios.....	4
1.5 Quotas.....	8
1.6 Constraints.....	8
1.7 Related Services.....	9
1.8 Basic Concepts.....	10
2 Getting Started.....	12
2.1 Getting Started with Elasticsearch.....	12
3 Permissions Management.....	18
3.1 Creating a User and Granting Permissions.....	18
4 Creating and Accessing a Cluster.....	20
4.1 Deploying a Cross-AZ Cluster.....	20
4.2 Clusters in Security Mode.....	22
4.3 Creating an Elasticsearch Cluster in Security Mode.....	27
4.4 Creating an Elasticsearch Cluster in Non-Security Mode.....	34
4.5 Accessing an Elasticsearch Cluster.....	41
4.6 Viewing Cluster Information.....	42
5 Scaling In/Out a Cluster.....	46
5.1 Overview.....	46
5.2 Scaling Out a Cluster.....	47
5.3 Changing Specifications.....	49
5.4 Scaling in a Cluster.....	51
5.5 Removing Specified Nodes.....	54
6 Importing Data to Elasticsearch.....	56
6.1 Using CDM to Import Data from OBS to Elasticsearch.....	56
6.2 Using DIS to Import Local Data to Elasticsearch.....	59
6.3 Using Logstash to Import Data to Elasticsearch.....	63
6.4 Using Kibana or APIs to Import Data to Elasticsearch.....	71

7 Managing Elasticsearch Clusters.....	75
7.1 Cluster and Storage Capacity Statuses.....	75
7.2 Introduction to the Cluster List.....	76
7.3 Index Backup and Restoration.....	77
7.4 Binding an Enterprise Project.....	83
7.5 Restarting a Cluster.....	84
7.6 Migrating Cluster Data.....	85
7.7 Deleting a Cluster.....	87
7.8 Managing Tags.....	87
7.9 Public Network Access.....	89
7.10 Managing Logs.....	91
7.11 Managing Plugins.....	93
7.12 Hot and Cold Data Storage.....	94
7.13 Configuring Parameters.....	95
7.14 VPC Endpoint Service.....	97
7.15 Kibana Public Access.....	99
8 Vector Retrieval.....	103
8.1 Description.....	103
8.2 Cluster Planning for Vector Retrieval.....	104
8.3 Creating a Vector Index.....	105
8.4 Querying Vectors.....	111
8.5 Optimizing the Performance of Vector Retrieval.....	115
8.6 (Optional) Pre-Building and Registering a Center Point Vector.....	116
8.7 Managing the Vector Index Cache.....	117
8.8 Sample Code for Vector Search on a Client.....	118
9 Working with Kibana.....	122
9.1 Logging In to Kibana.....	122
9.2 Creating a User and Granting Permissions by Using Kibana.....	122
9.3 Managing Index Statuses.....	128
9.3.1 Creating and Managing Indexes.....	128
9.3.2 Changing Policies.....	130
9.4 How Do I Connect a User-built Kibana with Elasticsearch?.....	131
9.5 Kibana Usage Restrictions.....	132
10 Elasticsearch SQL.....	133
11 Enhanced Features.....	138
11.1 Storage-Compute Decoupling.....	138
11.1.1 Context.....	138
11.1.2 Freezing an Index.....	138
11.1.3 Configuring Cache.....	146
11.1.4 Enhanced Cold Data Query Performance.....	148
11.1.5 Monitoring OBS Operations.....	151

11.2 Flow Control.....	153
11.2.1 Context.....	153
11.2.2 HTTP/HTTPS Flow Control.....	154
11.2.3 Memory Flow Control.....	156
11.2.4 Global Path Whitelist for Flow Control.....	159
11.2.5 Request Sampling.....	160
11.2.6 Flow Control.....	161
11.2.7 Access Logs.....	164
11.2.8 CPU Flow Control.....	166
11.2.9 One-click Traffic Blocking.....	168
11.3 Large Query Isolation.....	169
11.3.1 Context.....	169
11.3.2 Procedure.....	169
11.4 Index Monitoring.....	174
11.4.1 Context.....	174
11.4.2 Enabling Index Monitoring.....	174
11.4.3 Checking the Index Read and Write Traffic.....	175
11.5 Enhanced Monitoring.....	177
11.5.1 P99 Latency Monitoring.....	177
11.5.2 HTTP Status Code Monitoring.....	179
12 Monitoring.....	181
12.1 Supported Metrics.....	181
12.2 Configuring Cluster Monitoring.....	190
13 Auditing.....	193
13.1 Key Operations Recorded by CTS.....	193
13.2 Viewing Audit Logs.....	194
14 Best Practices.....	195
14.1 Cluster Migration.....	195
14.1.1 Migration Solution Overview.....	195
14.1.2 Migration from Elasticsearch.....	197
14.1.2.1 Migrating Cluster Data Using Logstash.....	197
14.1.2.2 Migrating Cluster Data Through Backup and Restoration.....	199
14.1.3 Migration from Kafka/MQ.....	202
14.1.4 Migration from a Database.....	203
14.2 Cluster Access.....	203
14.2.1 Overview.....	203
14.2.2 Accessing a Cluster Using cURL Commands.....	204
14.2.3 Accessing a Cluster Using Java.....	206
14.2.3.1 Accessing a Cluster Through the Rest High Level Client.....	206
14.2.3.2 Accessing a Cluster Through the Rest Low Level Client.....	215
14.2.3.3 Accessing the Cluster Through the Transport Client.....	229

14.2.4 Accessing a Cluster Using Python.....	230
14.2.5 Using ES-Hadoop to Read and Write Data in Elasticsearch Through Hive.....	233
14.3 Cluster Performance Tuning.....	238
14.3.1 Optimizing Write Performance.....	238
14.3.2 Optimizing Query Performance.....	241
14.4 Practices.....	244
14.4.1 Using CSS to Accelerate Database Query and Analysis.....	244
14.4.2 Using CSS to Build a Unified Log Management Platform.....	248
14.4.3 Configuring Query Scoring in an Elasticsearch Cluster.....	251
15 FAQs.....	257
15.1 General Consulting.....	257
15.1.1 What Are Regions and AZs?.....	257
15.1.2 How Does CSS Ensure Data and Service Security?.....	258
15.1.3 Which CSS Metrics Should I Focus On?.....	258
15.1.4 What Storage Options Does CSS Provide?.....	259
15.1.5 What Is the Maximum Storage Capacity of CSS?.....	259
15.1.6 How Can I Manage CSS?.....	259
15.1.7 What Can the Disk Space of a CSS Cluster Be Used For?.....	260
15.1.8 How Do I Check the Numbers of Shards and Replicas in a Cluster on the CSS Console?.....	260
15.1.9 What Data Compression Algorithms Does CSS Use?.....	260
15.2 Functions.....	261
15.2.1 Can Elasticsearch Data Be Migrated Between VPCs?.....	261
15.2.2 How Do I Migrate a CSS Cluster Across Regions?.....	261
15.2.3 How Do I Configure the Threshold for CSS Slow Query Logs?.....	262
15.2.4 How Do I Update the CSS Lifecycle Policy?.....	262
15.2.5 How Do I Set the Numbers of Index Copies to 0 in Batches?.....	264
15.2.6 Why All New Index Shards Are Allocated to the Same Node?.....	265
15.2.7 How Do I Query Snapshot Information?.....	266
15.2.8 Can I Upgrade a Cluster from an Earlier Version to a Later Version?.....	267
15.2.9 Can I Restore a Deleted Cluster?.....	268
15.2.10 Can I Modify the TLS Algorithm of an Elasticsearch Cluster?.....	269
15.2.11 How Do I Set the search.max_buckets Parameter for an ES Cluster?.....	269
15.2.12 Does the Value i of node.roles Indicate an Injest Node?.....	270
15.2.13 How Do I Create a Type Under an Index in an Elasticsearch 7.x Cluster?.....	270
15.3 Clusters in Security Mode.....	270
15.3.1 What Is the Relationship Between the Filebeat Version and Cluster Version?.....	270
15.3.2 How Do I Obtain the Security Certificate of CSS?.....	271
15.3.3 How Do I Convert the Format of a CER Security Certificate?.....	271
15.4 Resource Usage and Change.....	271
15.4.1 How Do I Clear Expired Data to Release Storage Space?.....	271
15.4.2 How Do I Configure a Two-Replica CSS Cluster?.....	272
15.4.3 How Do I Delete Index Data?.....	272

15.4.4 Can I Change the Number of Shards to Four with Two Replicas When There Is One Shard Set in the JSON File?.....	272
15.4.5 What Are the Impacts If an Elasticsearch Cluster Has Too Many Shards?.....	272
15.4.6 How Do I Set the Default Maximum Number of Records Displayed on a Page for an Elasticsearch Cluster.....	273
15.4.7 Why Does the Disk Usage Increase After the delete_by_query Command Was Executed to Delete Data?.....	273
15.4.8 How Do I Clear the Cache of a CSS Cluster?.....	273
15.4.9 The Average Memory Usage of an Elasticsearch Cluster Reaches 98%.....	274
15.5 Components.....	274
15.5.1 Can I Install Search Guard on CSS?.....	274
15.6 Kibana.....	275
15.6.1 Can I Export Data from Kibana?.....	275
15.6.2 How Do I Query Index Data on Kibana in an ES Cluster?.....	275
15.7 Clusters.....	276
15.7.1 Why Does My ECS Fail to Connect to a Cluster?.....	276
15.7.2 Can a New Cluster Use the IP Address of the Old Cluster?.....	277
15.7.3 Can I Associate My EIP If I Want to Access the Cluster from the Internet?.....	277
15.7.4 Can I Use x-pack-sql-jdbc to Access CSS Clusters and Query Data?.....	277
15.8 Ports.....	277
15.8.1 Do Ports 9200 and 9300 Both Open?.....	277
16 Change History.....	279

1 Overview

1.1 What Is Cloud Search Service?

CSS

Cloud Search Service (CSS) is a fully hosted distributed search service based on Elasticsearch. You can use it for structured and unstructured data search, and use AI vectors for combine search, statistics, and reports. CSS is a fully managed cloud service of the ELK Stack and is compatible with open-source Elasticsearch, Kibana, and Cerebro.

Elasticsearch is a distributed search engine that can be deployed in standalone or cluster mode. The heart of the ELK Stack, Elasticsearch clusters support multi-condition search, statistical analysis, and create visualized reports of structured and unstructured text. For details about Elasticsearch, see the [Elasticsearch: The Definitive Guide](#).

CSS can be automatically deployed, allowing you to quickly create Elasticsearch clusters. It provides the search engine optimization practices and does not require your O&M. Additionally, it has a robust monitoring system to present you key metrics, including clusters and query performance so that you can focus on the business logic.

Functions

- Compatible with Elasticsearch
Freely use native Elasticsearch APIs and other software in the ecosystem, such as Beats and Kibana.
- Support various data sources
A few simple configurations can allow you to smoothly connect to multiple data sources, such as FTP, OBS, HBase, and Kafka. No extra coding is required.
- One-click operation
One-click cluster application, capacity expansion, and restart from small-scale testing to large-scale rollout
- User-defined snapshot policies

Trigger backup snapshots manually or configure an automated schedule.

1.2 Advantages

CSS has the following features and advantages.

Efficient and Ease of Use

You can get insights from terabyte-scale data in milliseconds. In addition, you can use the visualized platform for data display and analysis.

Flexible and Scalable

You can request resources as needed and perform capacity expansion online with zero service interruption.

Easy O&M

CSS is a fully-managed, out-of-the-box service. You can start using it with several clicks, instead of managing clusters.

Kernel Enhancement

Speed up data import, decouple storage and computing resources, isolate read and write requests, and take advantage of our high-performance vector search engine – all these at an affordable price.

High Reliability

You can choose to trigger snapshots manually or on a periodic basis for backup and restore snapshots to the current or other clusters. Snapshots of a cluster can be restored to another cluster to implement cluster data migration.

- Automatic backup using snapshots
CSS provides the backup function. You can enable the automatic backup function on the CSS management console and set the backup period based on the actual requirements.
Automatic backup is to back up the index data of a cluster. Index backup is implemented by creating cluster snapshots. For backup of the first time, you are advised to back up all index data.
CSS allows you to store the snapshot data of Elasticsearch instances to OBS, thereby achieving cross-region backup with the cross-region replication function of OBS.
- Restoring data using snapshots
If data loss occurs or you want to retrieve data of a certain period, click **Restore** in the **Operation** column in the **Snapshots** area to restore the backup index data to the specified cluster by using existing snapshots.

High Security

CSS ensures secure running of data and services from the following aspects:

- Network isolation

The network is divided into two planes, service plane and management plane. The two planes are deployed and isolated physically to ensure the security of the service and management networks.

 - Service plane: refers to the network plane of the cluster. It provides service channels for users and delivers data definition, index, and search capabilities.
 - Management plane: refers to the management console. It is used to manage CSS.
 - VPC security groups or isolated networks ensure the security of hosts.
- Access control
 - Using the network access control list (ACL), you can permit or deny the network traffic entering and exiting the subnets.
 - Internal security infrastructure (including the network firewall, intrusion detection system, and protection system) can monitor all network traffic that enters or exits the VPC through the IPsec VPN.
 - User authentication and index-level authentication are supported. CSS also supports interconnection with third-party user management systems.
- Data security
 - In CSS, the multi-replica mechanism is used to ensure user data security.
 - Communication between the client and server can be encrypted using SSL.
- Operation audit

Cloud Trace Service (CTS) can be used to perform auditing on key logs and operations.

High Availability

To prevent data loss and minimize the cluster downtime in case of service interruption, CSS supports cross-AZ cluster deployment. When creating a cluster, you can select two or three AZs in the same region. The system will automatically allocate nodes to these AZs. If an AZ is faulty, the remaining AZs can still run properly, significantly enhancing cluster availability and improving service stability.

1.3 Product Components

CSS supports Kibana and Cerebro.

Kibana

Kibana is an open-source data analytics and visualization platform that works with Elasticsearch. You can use Kibana to search for and view data stored in Elasticsearch indexes and display data in charts and maps. For details about Kibana, visit <https://www.elastic.co/guide/en/kibana/current/index.html>.

By default, the Elasticsearch cluster of CSS provides the access channel to Kibana. You can quickly access Kibana without installing it. CSS is compatible with Kibana visualizations and Elasticsearch statistical and analysis capabilities.

- Over 10 data presentation modes
- Nearly 20 data statistics methods
- Classification in various dimensions, such as time and tag

Cerebro

Cerebro is an open-source Elasticsearch web admin tool built using Scala, Play Framework, AngularJS, and Bootstrap. Cerebro allows you to manage clusters on a visualized page, such as executing REST requests, modifying Elasticsearch configurations, monitoring real-time disks, cluster loads, and memory usage.

By default, the Elasticsearch cluster of CSS provides the access channel to Cerebro. You can quickly access Cerebro without installing it. CSS is fully compatible with the open-source Cerebro and adapts to the latest 0.8.4 version.

- Elasticsearch visualized and real-time load monitoring
- Elasticsearch visualized data management

1.4 Scenarios

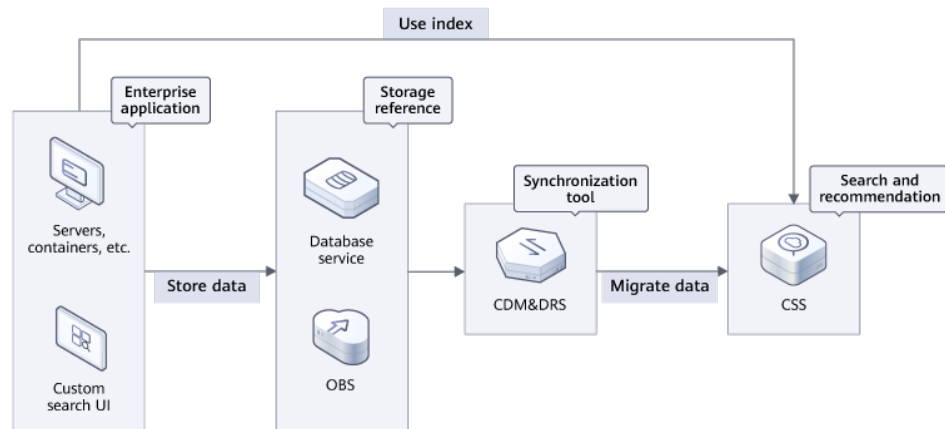
CSS can be used to build search boxes for websites and apps to improve user experience. You can also build a log analysis platform with it, facilitating data-driven O&M and business operations. CSS vector search can help you quickly build smart applications, such as AI-based image search, recommendation, and semantic search.

Site Search

CSS can be used to search for website content by keyword as well as search for and recommend commodities on e-commerce sites.

- Real-time search: When site content is updated, you can find the updated content in your search within minutes, or even just seconds.
- Categorized statistics: You can apply search filters to sort products by category.
- Custom highlight style: You can define how the search results are highlighted.

Figure 1-1 Site search

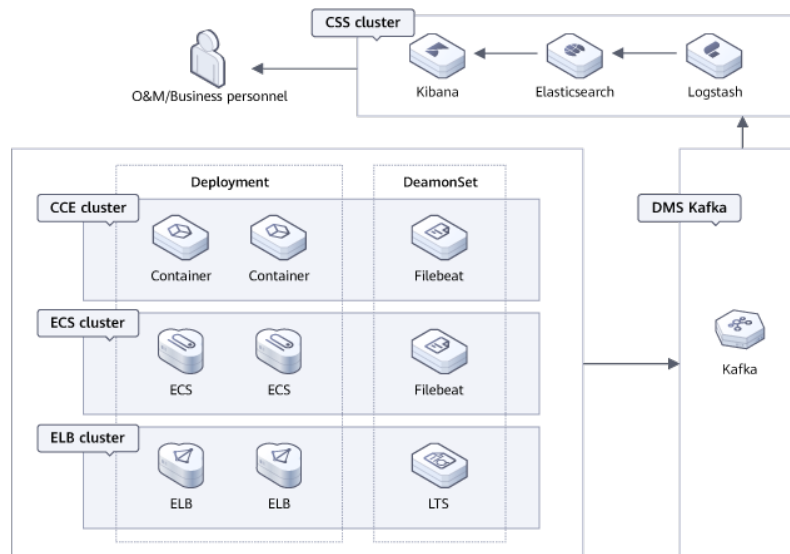


All-Scenario Log Analysis

Analyze the logs of Elastic Load Balance (ELB), servers, containers, and applications. In CSS, the Kafka message buffer queue is used to balance loads in peak and off-peak hours. Logstash is used for data extract, transform and load (ETL). Elasticsearch retrieves and analyzes data. The analysis results are visualized by Kibana and presented to you.

- High cost-effectiveness: CSS uses the Kunpeng computing power, separates cold and hot storage, and decouples computing and storage resources, achieving high performance and reducing costs by over 30%.
- Ease of use: Perform queries in a GUI editor. Easily create reports using drag-and-drop components.
- Powerful processing capability: CSS can import hundreds of terabytes of data per day, and can process petabytes of data.

Figure 1-2 All-scenario log analysis

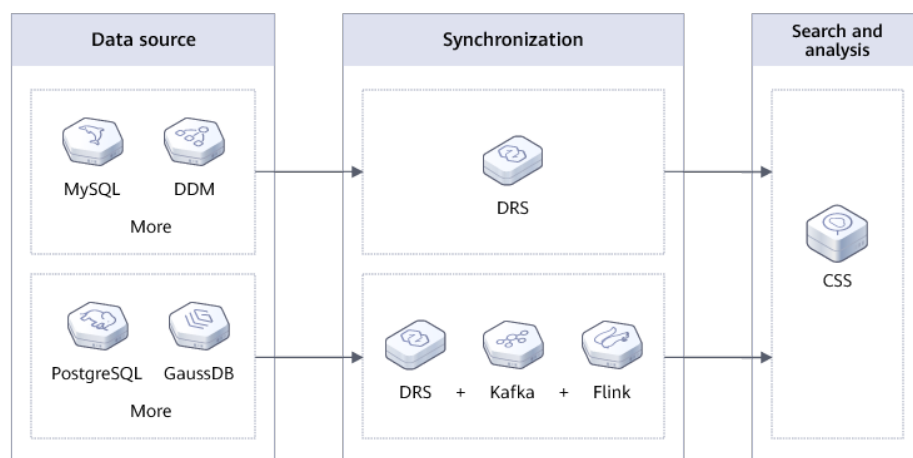


Database Query Acceleration

CSS can be used to accelerate database queries. E-commerce and logistics companies have to respond to a huge number of concurrent order queries within a short period of time. Relational databases, although having good transaction atomicity, are weak in transaction processing, and can rely on CSS to enhance OLTP and OLAP capabilities.

- High performance: Retrieve data from hundreds of millions of records within milliseconds. Text, time, numeric, and spatial data types are supported.
- High scalability: CSS can be scaled to have over 200 data nodes and over 1000 columns.
- Zero service interruption: The rolling restart and dual-copy mechanisms can avoid service interruption in case of specifications change or configuration update.

Figure 1-3 Database query acceleration

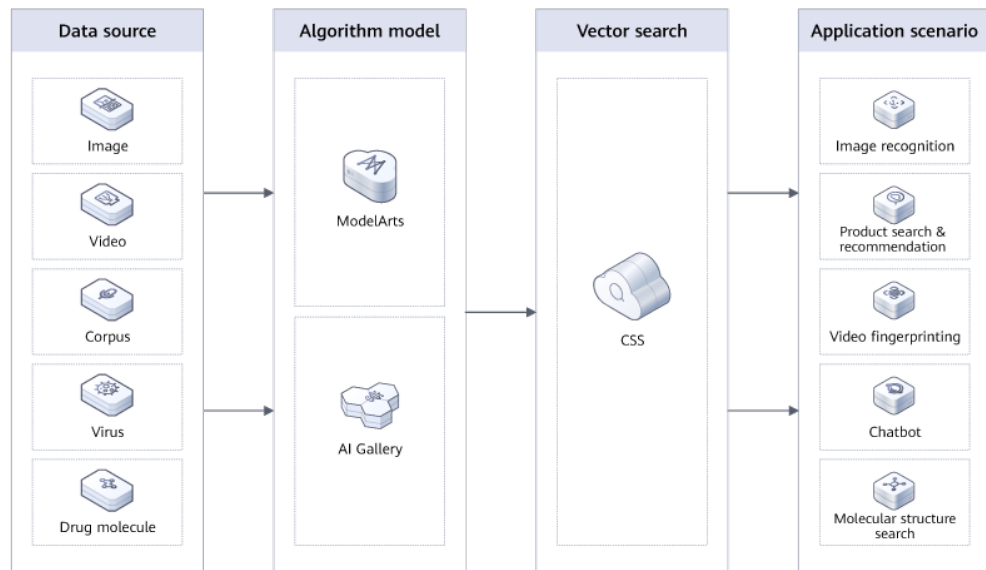


Vector Search

When you search for unstructured data, such as images, videos, and corpuses, the nearest neighbors or approximate nearest neighbors are searched based on feature vectors. This has the following advantages:

- Efficiency and reliability: The vector search engine provides ultimate search performance and distributed disaster recovery capabilities.
- Abundant indexes: Multiple indexing algorithms and similarity measurement methods are available and can meet diverse needs.
- Easy learning: CSS is fully compatible with the open-source Elasticsearch ecosystem.

Figure 1-4 Vector search



1.5 Quotas

CSS uses the following resources:

- Instance
- CPU
- Memory (GB)
- Disk quantity
- Disk size (GB)

1.6 Constraints

Restrictions on Clusters and Nodes

The following table describes restrictions on clusters and nodes in CSS.

Table 1-1 Restrictions on Elasticsearch clusters and nodes

Cluster and Node	Restriction
Maximum number of nodes in a cluster	Default: 32. Maximum: 200. To change the default value, contact technical support.
Minimum number of nodes in a cluster	1

Restrictions on Browsers

- You are advised to use the following browsers to access the CSS management console:
 - Google Chrome 36.0 or later
 - Mozilla Firefox 35.0 or later
- You are advised to use the following browsers to access Kibana integrated in CSS:
 - Google Chrome 36.0 or later
 - Mozilla Firefox 35.0 or later

1.7 Related Services

Figure 1-5 shows the relationships between CSS and other services.

Figure 1-5 Relationships between CSS and other services

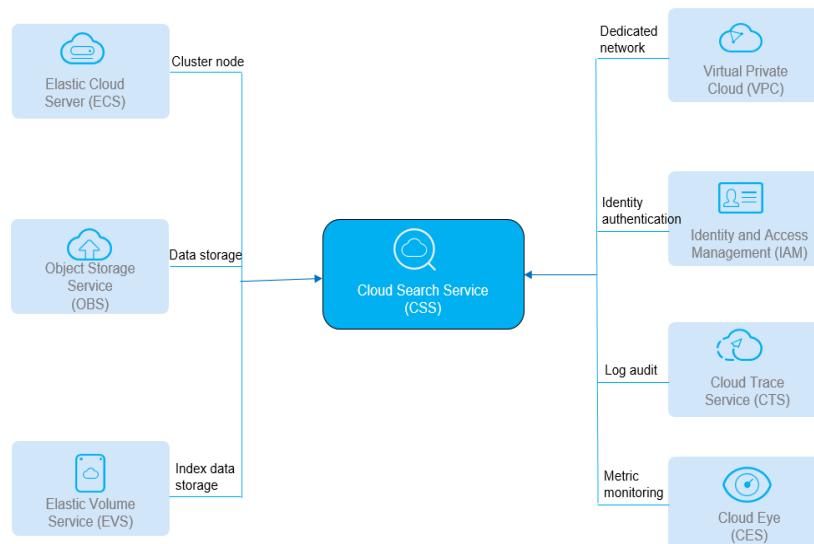


Table 1-2 Relationships between CSS and other services

Service	Description
Virtual Private Cloud (VPC)	CSS clusters are created in the subnets of a VPC. VPCs provide a secure, isolated, and logical network environment for your clusters.
Elastic Cloud Server (ECS)	In a CSS cluster, each node represents an ECS. When you create a cluster, ECSs are automatically created.
Elastic Volume Service (EVS)	CSS uses EVS to store index data. When you create a cluster, EVSs are automatically created for cluster data storage.
Object Storage Service (OBS)	Snapshots of CSS clusters are stored in OBS buckets.

Service	Description
Identity and Access Management (IAM)	IAM authenticates access to CSS.
Cloud Eye	CSS uses Cloud Eye to monitor cluster metrics in real time. The supported CSS metrics include the disk usage and cluster health status. You can learn about the disk usage of the cluster based on the disk usage metric. You can learn about the health status of a cluster based on the cluster health status metric.
Cloud Trace Service (CTS)	With CTS, you can record operations associated with CSS for query, audit, and backtracking operations.

1.8 Basic Concepts

Cluster

CSS provides functions on a per cluster basis. A cluster represents an independent search service that consists of multiple nodes.

Index

An index stores Elasticsearch data. It is a logical space in which one or more shards are grouped.

Shard

An index can potentially store a large amount of data that can exceed the hardware limits of a single node. To solve this problem, Elasticsearch provides the ability to subdivide your index into multiple pieces called shards. When you create an index, you can simply define the number of shards that you want. Each shard is in itself a fully-functional and independent "index" that can be hosted on any node in the cluster.

You need to specify the number of shards before creating an index and cannot change the number after the index is successfully created.

Replica

A replica is a copy of the actual storage index in a shard. It can be understood as a backup of the shard. Replicas help prevent single point of failures (SPOFs). You can increase or decrease the number of replicas based on your service requirements.

Document

An entity for Elasticsearch storage. Equivalent to the row in the RDB, the document is the basic unit that can be indexed.

Document Type

Similar to a table in the RDB, type is used to distinguish between different data.

In versions earlier than Elasticsearch 7.x, each index can contain multiple document types. Elasticsearch defines a type for each document.

Elasticsearch 7.x and later versions only support documents of the .doc type.

Mapping

A mapping is used to restrict the type of a field and can be automatically created based on data. It is similar to the schema in the database.

Field

The field is the minimum unit of a document. It is similar to the column in the database.

2 Getting Started

2.1 Getting Started with Elasticsearch

This section describes how to use Elasticsearch for product search. You can use the Elasticsearch search engine of CSS to search for data based on the scenario example. The basic operation process is as follows:

- [Step 1: Create a Cluster](#)
- [Step 2: Import Data](#)
- [Step 3: Search for Data](#)
- [Step 4: Delete the Cluster](#)

Scenario Description

A women's clothing brand builds an e-commerce website. It uses traditional databases to provide a product search function for users. However, due to an increase in the number of users and business growth, the traditional databases have slow response and low accuracy. To improve user experience and user retention, the e-commerce website plans to use Elasticsearch to provide the product search function for users.

This section describes how to use Elasticsearch to provide the search function for users.

Assume that the e-commerce website provides the following data:

```
{
  "products":[
    {"productName":"Latest art shirts for women in autumn 2017","size":"L"}
    {"productName":"Latest art shirts for women in autumn 2017","size":"M"}
    {"productName":"Latest art shirts for women in autumn 2017","size":"S"}
    {"productName":"Latest jeans for women in spring 2018","size":"M"}
    {"productName":"Latest jeans for women in spring 2018","size":"S"}
    {"productName":"Latest jeans for women in spring 2017","size":"L"}
    {"productName":"Latest casual pants for women in spring 2017","size":"S"}
  ]
}
```

Step 1: Create a Cluster

Create a cluster using Elasticsearch as the search engine. In this example, suppose that you create a cluster named **Sample-ESCluster**. This cluster is used only for getting started with Elasticsearch. For this cluster, you are advised to select **ess.spec-4u8g** for **Node Specifications**, **High I/O** for **Node Storage Type**, and **40 GB** for **Node Storage Capacity**. For details, see [Creating an Elasticsearch Cluster in Non-Security Mode](#).

Create a cluster using Elasticsearch as the search engine. In this example, suppose that you create a cluster named **Sample-ESCluster**. This cluster is used only for getting started with Elasticsearch. For this cluster, you are advised to select **ess.spec-4u8g** for **Node Specifications**, **High I/O** for **Node Storage Type**, and **40 GB** for **Node Storage Capacity**. For details, see [Creating an Elasticsearch Cluster in Security Mode](#) or [Creating an Elasticsearch Cluster in Non-Security Mode](#).

After you create the cluster, switch to the cluster list to view the created cluster. If the **Status** of the cluster is **Available**, the cluster is created successfully. See the following figure.

Figure 2-1 Creating a cluster

Name/ID	Cluster Status	Task Status	Version	Created	Enterprise Project	Private Network	Billing Mode	Operation
css-jkl-switchAZ-... Seabd430-1e8f-4...	Available	Upgrade failed	7.10.2 elasticsearch	Dec 22, 2022 17:...	default		Pay-per-use	Access Kibana More

Step 2: Import Data

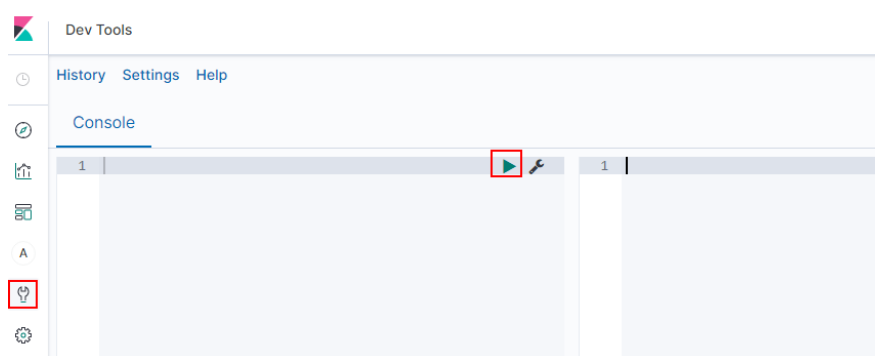
CSS supports importing data to Elasticsearch using Logstash, Kibana, or APIs. Kibana lets you visualize your Elasticsearch data. The following procedure illustrates how to import data to Elasticsearch using Kibana.

- On the **Clusters** page, locate the target cluster and click **Access Kibana** in the **Operation** column to go to the Kibana login page.
 - Non-security cluster: The Kibana console is displayed.
 - Security cluster: Enter the username and password on the login page and click **Log In** to go to the Kibana console. The default username is **admin** and the password is the one specified during cluster creation.

- In the navigation pane of Kibana on the left, choose **Dev Tools**.

The text box on the left is the input box. The triangle icon in the upper right corner of the input box is the command execution button. The text box on the right area is the result output box.

Figure 2-2 Console page



 **NOTE**

The Kibana UI varies depending on the Kibana version.

3. On the **Console** page, run the following command to create index named **my_store**:

(Versions later than 7.x)

```
PUT /my_store
{
  "settings": {
    "number_of_shards": 1
  },
  "mappings": {
    "properties": {
      "productName": {
        "type": "text",
        "analyzer": "ik_smart"
      },
      "size": {
        "type": "keyword"
      }
    }
  }
}
```

The command output is similar to the following:

```
{
  "acknowledged" : true,
  "shards_acknowledged" : true,
  "index" : "my_store"
}
```

4. On the **Console** page, run the following command to import data to index named **my_store**:

(Versions later than 7.x)

```
POST /my_store/_doc/_bulk
{"index":{}}
{"productName":"Latest art shirts for women in autumn 2017","size":"L"}
{"index":{}}
{"productName":"Latest art shirts for women in autumn 2017","size":"M"}
{"index":{}}
{"productName":"Latest art shirts for women in autumn 2017","size":"S"}
{"index":{}}
{"productName":"Latest jeans for women in spring 2018","size":"M"}
{"index":{}}
{"productName":"Latest jeans for women in spring 2018","size":"S"}
{"index":{}}
{"productName":"Latest casual pants for women in spring 2017","size":"L"}
{"index":{}}
{"productName":"Latest casual pants for women in spring 2017","size":"S"}
```

If the value of the **errors** field in the command output is **false**, the data is imported successfully.

Step 3: Search for Data

- **Full-text search**

If you access the e-commerce website and want to search for commodities whose names include "spring jeans", enter "spring jeans" to begin your search. The following example shows the command to be executed on Kibana and the command output.

Command to be executed on Kibana:

(Versions later than 7.x)

```
GET /my_store/_search
{
  "query": {"match": {
    "productName": "spring jeans"
  }}
}
```

The command output is similar to the following:

```
{
  "took" : 3,
  "timed_out" : false,
  "_shards" : {
    "total" : 1,
    "successful" : 1,
    "skipped" : 0,
    "failed" : 0
  },
  "hits" : {
    "total" : {
      "value" : 4,
      "relation" : "eq"
    },
    "max_score" : 1.7965372,
    "hits" : [
      {
        "_index" : "my_store",
        "_type" : "_doc",
        "_id" : "9xf6VHIBfCl6SDjw7H5",
        "_score" : 1.7965372,
        "_source" : {
          "productName": "Latest jeans for women in spring 2018",
          "size" : "M"
        }
      },
      {
        "_index" : "my_store",
        "_type" : "_doc",
        "_id" : "-Bf6VHIBfCl6SDjw7H5",
        "_score" : 1.7965372,
        "_source" : {
          "productName": "Latest jeans for women in spring 2018",
          "size" : "S"
        }
      },
      {
        "_index" : "my_store",
        "_type" : "_doc",
        "_id" : "-Rf6VHIBfCl6SDjw7H5",
        "_score" : 0.5945667,
        "_source" : {
          "productName": "Latest casual pants for women in spring 2017",
          "size" : "L"
        }
      },
      {
        "_index" : "my_store",
        "_type" : "_doc",
        "_id" : "-hf6VHIBfCl6SDjw7H5",
        "_score" : 0.5945667,
        "_source" : {
          "productName": "Latest casual pants for women in spring 2017",
          "size" : "S"
        }
      }
    ]
  }
}
```

- Elasticsearch supports word segmentation. The preceding command segments "spring jeans" into "spring" and "jeans".

- Elasticsearch supports full-text search. The preceding command searches for the information about all commodities whose names include "spring" or "jeans".
 - Unlike traditional databases, Elasticsearch can return results in milliseconds by using inverted indexes.
 - Elasticsearch supports sorting by score. In the command output, information about the first two commodities contains both "spring" and "jeans", while that about the last two products contain only "spring". Therefore, the first two commodities rank prior to the last two due to high keyword match.
- **Aggregation result display**

The e-commerce website provides the function of displaying aggregation results. For example, it classifies commodities corresponding to "spring" based on the size so that you can collect the number of products of different sizes. The following example shows the command to be executed on Kibana and the command output.

Command to be executed on Kibana:

(Versions later than 7.x)

```
GET /my_store/_search
{
  "query": {
    "match": { "productName": "spring" }
  },
  "size": 0,
  "aggs": {
    "sizes": {
      "terms": { "field": "size" }
    }
  }
}
```

The command output is similar to the following:

(Versions later than 7.x)

```
{
  "took" : 3,
  "timed_out" : false,
  "_shards" : {
    "total" : 1,
    "successful" : 1,
    "skipped" : 0,
    "failed" : 0
  },
  "hits" : {
    "total" : {
      "value" : 4,
      "relation" : "eq"
    },
    "max_score" : null,
    "hits" : [ ]
  },
  "aggregations" : {
    "sizes" : {
      "doc_count_error_upper_bound" : 0,
      "sum_other_doc_count" : 0,
      "buckets" : [
        {
          "key" : "S",
          "doc_count" : 2
        },
        {
          "key" : "L",
```

```
    "doc_count" : 1
  },
  {
    "key" : "M",
    "doc_count" : 1
  }
]
}
```

Step 4: Delete the Cluster

Once you understand the process and method of using Elasticsearch, you can perform the following steps to delete the cluster you created for the example and its data to avoid resource wastage.

NOTE

After you delete a cluster, its data cannot be restored. Exercise caution when deleting a cluster.

1. Log in to the CSS management console. In the navigation pane on the left, choose **Clusters > Elasticsearch**.
2. Locate the row that contains cluster **Sample-ESCluster** and click **More > Delete** in the **Operation** column.
3. In the displayed dialog box, enter the name of the cluster to be deleted and click **OK**.

3 Permissions Management

3.1 Creating a User and Granting Permissions

This section describes how to use a group to grant permissions to a user. [Figure 3-1](#) shows the process for granting permissions.

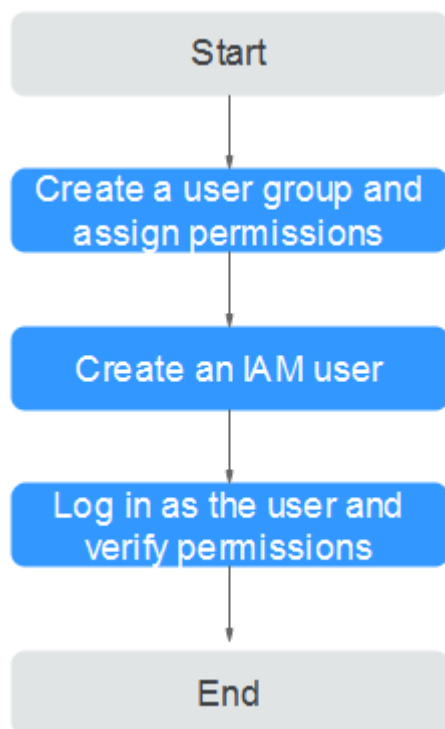
CSS has two types of user permissions: CSS administrator permission and read-only permission.

Prerequisites

Before assigning permissions to user groups, you have learned about the system policies listed in Permissions Management.

Process Flow

Figure 3-1 Process for granting CSS permissions



1. .Create a user group and assign permissions
Create a user group on the IAM console, and assign the CSS permission to the group.
2. Create a user and add it to a user group
Create a user on the IAM console and add the user to the group created in **1. Create a user group and assign permissions.**
3. Log in and verify permissions.
Log in to the CSS console as the created user, and verify that it only has read permissions for CSS.

4 Creating and Accessing a Cluster

4.1 Deploying a Cross-AZ Cluster

To prevent data loss and minimize the cluster downtime in case of service interruption, CSS supports cross-AZ cluster deployment. When creating a cluster, you can select two or three AZs in the same region. The system will automatically allocate nodes to these AZs.

Allocating Nodes

If you select two or three AZs when creating a cluster, CSS automatically enables the cross-AZ HA function and properly allocates nodes to different AZs. [Table 4-1](#) describes how the nodes are allocated.

 **NOTE**

- When creating a cluster, ensure that the number of selected nodes is no less than the number of AZs. Otherwise, cross-AZ deployment is not supported.
- If you enable master nodes when deploying a cross-AZ cluster, the master nodes will also be distributed to different AZs.
- The node quantity difference between any two AZs is no more than one.

Table 4-1 Number of nodes and AZ distribution

Nodes	Single AZ	Two AZs		Three AZs		
	AZ1	AZ1	AZ2	AZ1	AZ2	AZ3
1	1	Not supported		Not supported		
2	2	1	1	Not supported		
3	3	2	1	1	1	1
4	4	2	2	2	1	1
...

Setting Replicas

To maximize HA performance, properly determine the number of replicas.

- In two-AZ deployment, if one AZ becomes unavailable, the other AZ continues to provide services. In this case, at least one replica is required. Elasticsearch has one replica by default. You can retain the default value if you do not require higher read performance.
- In three-AZ deployment, if one AZ becomes unavailable, the other AZs continue to provide services. In this case, at least one replica is required. Elasticsearch has one replica by default. If you need more replicas to improve the cluster's ability to handle queries, modify `[<code>/topic/body/section/ul/li/p/idp:point/strong {<code>"}</code>] settings (strong) to change the number of replicas.`

You can run the following command to modify the number of index replicas:

```
curl -XPUT http://ip:9200/{index_name}/_settings -d
'{"number_of_replicas":2}'
```

Alternatively, run the following command to specify the number of replicas in the template:

```
curl -XPUT http://ip:9200/_template/templatename -d '{ "template": "*",
"settings": {"number_of_replicas": 2}}'
```

NOTE

- **ip**: private network address
- **index_name**: index name
- **number_of_replicas**: number of replicas after modification. The value in the preceding command indicates that two replicas are required.

Possible Service Interruptions

The following table describes the possible service interruptions when an AZ of a two- or three-AZ cluster is faulty.

Table 4-2 Possible service interruptions

AZs	Master Nodes	Service Interruption Analysis
2	0	<ul style="list-style-type: none"> • When the number of nodes is an even number: <ul style="list-style-type: none"> – If half of data nodes are faulty, replace one node in the faulty AZ before you select the master node. • When the number of nodes is an odd number: <ul style="list-style-type: none"> – If the faulty AZ contains one more node than the normal AZ, you need to replace one node in the faulty AZ before you select the master node. For details about how to replace nodes, contact technical support. – If the faulty AZ contains one less node than the normal AZ, services will not be interrupted and you can select the master node.

AZs	Master Nodes	Service Interruption Analysis
2	3	<p>There is a 50% possibility for service interruption. When two dedicated master nodes are allocated to one AZ and another master node is allocated to the other AZ:</p> <ul style="list-style-type: none"> • If service interruption happens in the AZ with one master node, you can select a master node from the AZ that has two dedicated master nodes. • If service interruption happens in the AZ with two dedicated master nodes, you have no choice in the remaining AZ, because it has only one dedicated master node. In this case, services will be interrupted and you need to contact technical support.
3	0	<p>If you configure four nodes in three AZs, each AZ will have at least one node. If the AZ with two nodes is faulty, the services will be interrupted. You are not advised to configure four nodes when selecting three AZs.</p> <p>Generally, service interruption will not occur.</p>
3	3	Service interruption does not occur.

4.2 Clusters in Security Mode

When creating an Elasticsearch cluster, you can enable the security mode for it. Identity authentication is required when users access a security cluster. You can also authorize and encrypt security clusters.

Context

You can create clusters in multiple security modes. For details about the differences between security modes, see [Table 4-3](#).

Table 4-3 Cluster security modes

Security Mode	Scenario	Advantage	Disadvantage
Non-Security Mode	Intranet services and test scenarios	Simple. Easy to access.	Poor security. Anyone can access such clusters.

Security Mode	Scenario	Advantage	Disadvantage
Security Mode + HTTP Protocol	User permissions can be isolated, which is applicable to scenarios sensitive to cluster performance.	Security authentication is required for accessing such clusters, which improves cluster security. Accessing a cluster through HTTP protocol can retain the high performance of the cluster.	Cannot be accessed from the public network.
Security Mode + HTTPS Protocol	Scenarios that require high security and public network access.	Security authentication is required for accessing such clusters, which improves cluster security. HTTPS protocol allows public network to access such clusters.	The performance of clusters using HTTPS is 20% lower than that of using HTTP.

Identity Verification

To access a security cluster, you need to enter the username and password. The identity verification is required for the following two types of users:

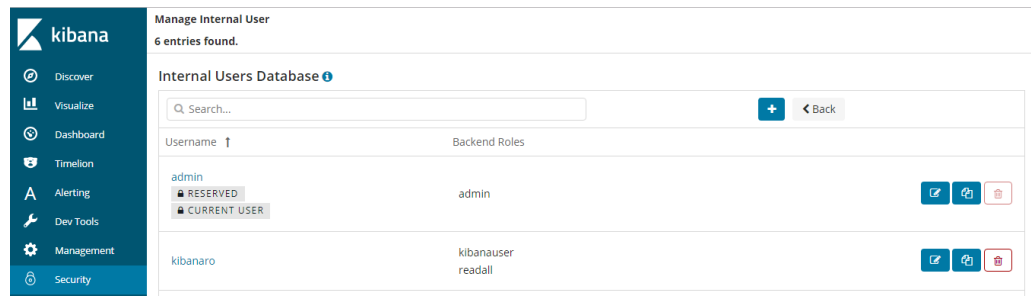
- Administrator: The default administrator username is **admin**, and the password is the one specified during cluster creation.
- Users: Enter the username and password created through Kibana.

Authorization

On the **Kibana** console, click **Security** to control user permissions in Elasticsearch clusters. You can configure hierarchical user permissions by cluster, index, document, and field. For details, see [Creating a User and Granting Permissions by Using Kibana](#).

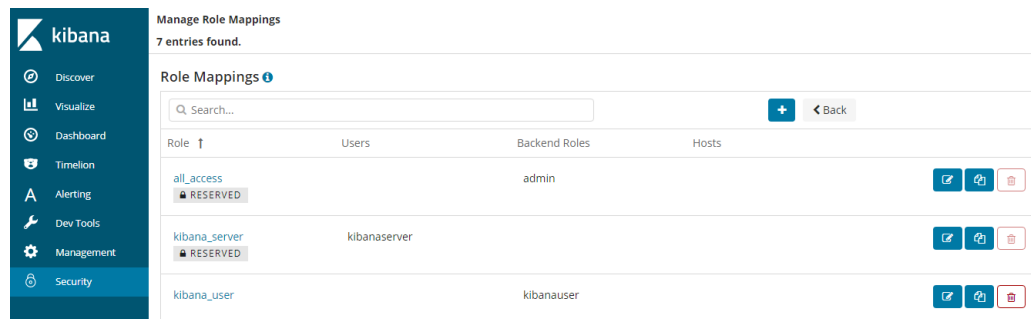
You can add or delete users, and map users to different roles for permissions control.

Figure 4-1 Configuring users



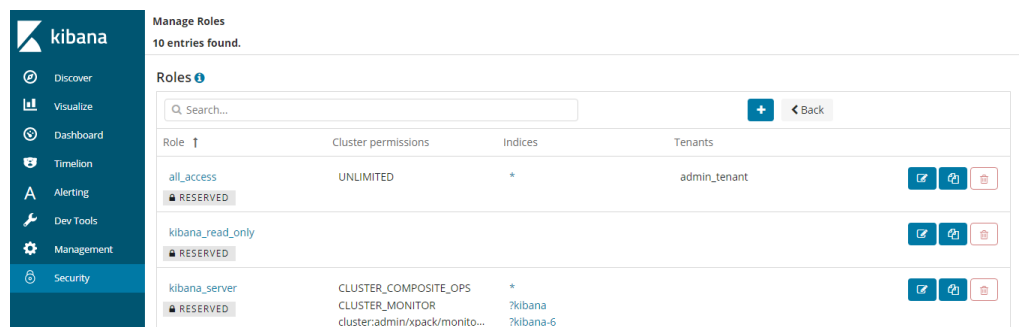
You can use role mapping to configure roles and map a user, backend role, and host name to a role.

Figure 4-2 Role mapping



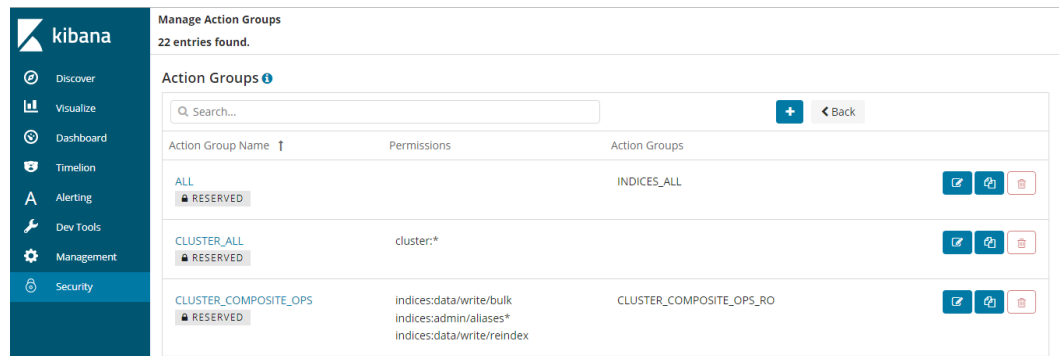
You can set permissions for each role to access clusters, indexes and documents and assign Kibana tenants different roles.

Figure 4-3 Configuring role permissions



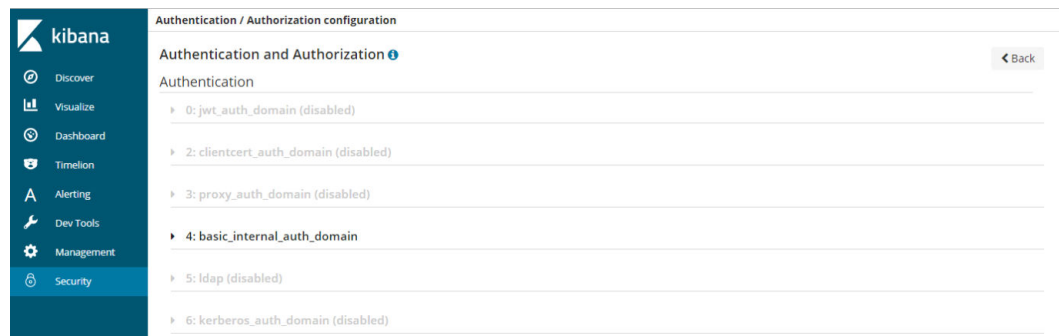
You can set action groups, assign the groups to roles, and configure the roles' permission for accessing indexes and documents.

Figure 4-4 Configuring action groups



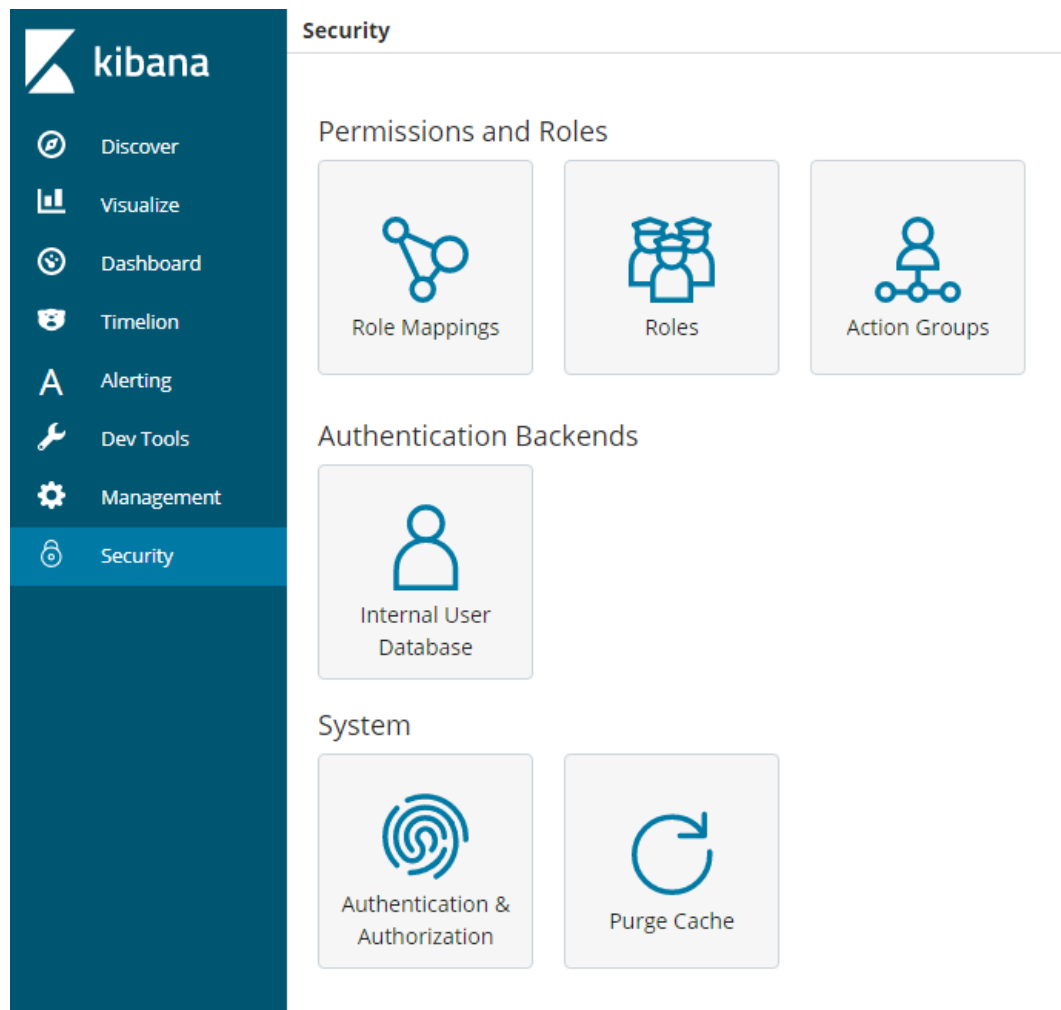
You can view the parameters of authentication and authorization for the current cluster. You can also run the **securityadmin** command to modify the configuration.

Figure 4-5 Viewing cluster parameters



You can also clear the security cache.

Figure 4-6 Clearing the security cache



Encryption

When key data is transferred between nodes or through the HTTP protocol, SSL/TLS encryption is used to ensure data security.

You can perform the preceding functions on Kibana, using **.yaml** files (not recommended), or by calling RESTful APIs. For more information about the security mode, see [Security](#).

Resetting the Administrator Password

If you want to change the administrator password of a security cluster or you have forgotten the password, reset the password.

1. On the **Clusters** page, locate the target cluster whose password you want to reset and click the cluster name. The **Cluster Information** page is displayed.
2. In the **Configuration** area, click **Reset** next to **Reset Password**.

 **NOTE**

- The password can contain 8 to 32 characters.
- The password must contain at least three of the following character types: uppercase letters, lowercase letters, digits, and special characters. The following special characters are supported: ~!@#\$\$%^&*()-_+=\|{}];;<.>/?
- Do not use the administrator name, or the administrator name spelled backwards.
- You are advised to change the password periodically.

4.3 Creating an Elasticsearch Cluster in Security Mode

This section describes how to create an Elasticsearch cluster in security mode.

 **NOTE**

- Public IP address access and Kibana public access can be used only after security mode is enabled.

Context

- When creating a cluster, the number of nodes that can be added varies according to the node type. For details, see [Table 4-4](#).

Table 4-4 Number of nodes in different types

Node Type	Number
ess	ess: 1-32
ess, ess-master	ess: 1-200 ess-master: an odd number ranging from 3 to 9
ess, ess-client	ess: 1-32 ess-client: 1-32
ess, ess-cold	ess: 1-32 ess-cold: 1-32
ess, ess-master, ess-client	ess: 1-200 ess-master: an odd number ranging from 3 to 9 ess-client: 1-32
ess, ess-master, ess-cold	ess: 1-200 ess-master: an odd number ranging from 3 to 9 ess-cold: 1-32
ess, ess-client, ess-cold	ess: 1-32 ess-client: 1-32 ess-cold: 1-32

Node Type	Number
ess, ess-master, ess-client, ess-cold	ess: 1-200 ess-master: an odd number ranging from 3 to 9 ess-client: 1-32 ess-cold: 1-32
Details about the four node types: <ul style="list-style-type: none"> ● ess: the default node type that is mandatory for cluster creation. The other three node types are optional. ● ess-master: master node ● ess-client: client node ● ess-cold: cold data node 	

Procedure

1. Log in to the CSS management console.
2. Click **Create Cluster** in the upper right corner. The **Create Cluster** page is displayed.
3. Specify **Region** and **AZ**.




Table 4-5 Parameter description for Region and AZ

Parameter	Description
Region	Select a region for the cluster from the drop-down list on the right.
AZ	Select AZs associated with the cluster region. You can select a maximum of three AZs. For details, see Deploying a Cross-AZ Cluster .

4. Configure basic cluster information.

Table 4-6 Description of basic parameters

Parameter	Description
Version	Select a cluster version from the drop-down list box.

Parameter	Description
Name	<p>Cluster name, which contains 4 to 32 characters. Only letters, numbers, hyphens (-), and underscores (_) are allowed and the value must start with a letter.</p> <p>NOTE After a cluster is created, you can modify the cluster name as required. Click the name of a cluster to be modified. On the displayed Basic Information page, click  next to the cluster name. After the modification is completed, click  to save the modification. If you want to cancel the modification, click .</p>

5. Configure cluster specifications.

Table 4-7 Parameter description

Parameter	Description
Nodes	<p>Number of nodes in a cluster.</p> <ul style="list-style-type: none"> • If neither a master node nor client node is enabled, the nodes specified by this parameter are used to serve as both the master node and client node. Nodes provide the cluster management, data storage, cluster access, and data analysis functions. To ensure data stability in the cluster, it is recommended that you set this parameter to a value no less than 3. • If only the master node function is enabled, nodes specified by this parameter are used to store data and provide functions of client nodes. • If both the master and client node functions are enabled, the nodes specified by this parameter are only used for storing data. • If only the client node function is enabled, nodes specified by this parameter are used to store data and provide functions of the master node.
CPU Architecture	Currently, x86 and Kunpeng are supported. The supported type is determined by the actual regional environment.
Node Specifications	Specifications of nodes in a cluster. You can select a specified specification based on your needs. Each cluster supports only one specification.
Node Storage Type	In the current version, the following options are available: Common I/O , High I/O , and Ultra-high I/O .
Node Storage Capacity	Storage space. Its value varies with node specifications. The node storage capacity must be a multiple of 20.

Parameter	Description
Master node	<p>The master node manages all nodes in the cluster. If more than 20 nodes are required to store and analyze a large amount of data, you are advised to enable the master node to ensure cluster stability. Otherwise, you are advised to set only the Nodes parameter and use the nodes as both master and client nodes.</p> <p>After enabling the master node, specify Node Specifications, Nodes, and Node Storage Type. The value of Nodes must be an odd number equal to or greater than 3. Up to nine nodes are supported. The value of Node Storage Capacity is fixed. You can select a storage type based on your needs.</p>
Client node	<p>The client node allows clients to access clusters and analyze data. If more than 20 nodes are required to store and analyze a large amount of data, you are advised to enable the client node to ensure cluster stability. Otherwise, you are advised to set only the Nodes parameter and use the nodes as both master and client nodes.</p> <p>After enabling the client node, specify Node Specifications, Nodes and Node Storage Type. The value of Nodes ranges from 1 to 32. The value of Node Storage Capacity is fixed. You can select a storage type based on your needs.</p>
Cold data node	<p>The cold data node is used to store historical data, for which query responses can be returned in minutes. If you do not require a quick query response, store historical data on cold data nodes to reduce costs.</p> <p>After enabling cold data node, configure Node Specifications, Nodes, Node Storage Type, and Node Storage Capacity. The value of Nodes ranges from 1 to 32. Select Node Storage Type and Node Storage Capacity as requirement.</p> <p>After the cold data node is enabled, CSS automatically adds cold and hot tags to related nodes.</p>

6. Set the enterprise project.
When creating a CSS cluster, you can bind an enterprise project to the cluster if you have enabled the enterprise project function. You can select an enterprise project created by the current user from the drop-down list on the right or click **View Project Management** to go to the **Enterprise Project Management** console and create a new project or view existing projects.
7. Click **Next: Configure Network**. Configure the cluster network.

Table 4-8 Network configuration parameters

Parameter	Description
VPC	<p>A VPC is a secure, isolated, and logical network environment.</p> <p>Select the target VPC. Click View VPC to enter the VPC management console and view the created VPC names and IDs. If no VPCs are available, create one.</p> <p>NOTE The VPC must contain CIDRs. Otherwise, cluster creation will fail. By default, a VPC will contain CIDRs.</p>
Subnet	<p>A subnet provides dedicated network resources that are isolated from other networks, improving network security.</p> <p>Select the target subnet. You can access the VPC management console to view the names and IDs of the existing subnets in the VPC.</p>
Security Group	<p>A security group is a collection of access control rules for ECSs that have the same security protection requirements and are mutually trusted in a VPC. To view more details about the security group, click View Security Group.</p> <p>NOTE</p> <ul style="list-style-type: none"> • For cluster access purposes, ensure that the security group contains port 9200. • If your cluster version is 7.6.2 or later, ensure that all the ports used for communication between nodes in the same security group are allowed. If such settings cannot be configured, ensure at least the access to port 9300 is allowed.
Security Mode	<p>After the security mode is enabled, communication will be encrypted and authentication required for the cluster.</p> <ul style="list-style-type: none"> • The default Administrator Username is admin. • Set and confirm the Administrator Password. This password will be required when you access this cluster.

Parameter	Description
HTTPS Access	<p>HTTPS access can be enabled only after the security mode of the cluster is enabled. After HTTPS access is enabled, communication is encrypted when you access the cluster.</p> <p>NOTE Security clusters use HTTPS for communication, which is much slower than non-security clusters that use HTTP for communication. If you want fast read performance and the permission provided by the security mode to isolate resources (such as indexes, documents, and fields), you can disable the HTTPS Access function. After HTTPS Access is disabled, HTTP protocol is used for cluster communication. In this case, data security cannot be ensured and public IP address cannot be used.</p>
Public IP Address	<p>If HTTPS Access is enabled, you can configure Public Network Access and obtain an IP address for public network access. This IP address can be used to access this security cluster through the public network. For details, see Public Network Access.</p>

8. Click **Next: Configure Advanced Settings**. Configure the automatic snapshot creation and other functions.
 - a. Configure **Cluster Snapshot**. Set basic configuration and snapshot configuration.

The cluster snapshot function is enabled by default. You can also disable this function as required. To store automatic snapshots in OBS, an agency will be created to access OBS. Additional cost will be incurred if snapshots are stored in standard storage.

Table 4-9 Cluster snapshot parameter

Parameter	Description
OBS bucket	<p>Select an OBS bucket for storing snapshots from the drop-down list box. You can also click Create Bucket on the right to create an OBS bucket.</p> <p>The created or existing OBS bucket must meet the following requirements:</p> <ul style="list-style-type: none"> • Storage Class is Standard.

Parameter	Description
Backup Path	<p>Storage path of the snapshot in the OBS bucket.</p> <p>The backup path configuration rules are as follows:</p> <ul style="list-style-type: none"> • The backup path cannot contain the following characters: \:*?"<> • The backup path cannot start with a slash (/). • The backup path cannot start or end with a period (. • The backup path cannot contain more than 1,023 characters.
IAM Agency	<p>IAM agency authorized by the current account to CSS access or maintain data stored in the OBS bucket. You can also click Create IAM Agency on the right to create an IAM agency.</p> <p>The created or existing IAM agency must meet the following requirements:</p> <ul style="list-style-type: none"> • Agency Type must be Cloud service. • Set Cloud Service to Elasticsearch or CSS. • The agency must have the Tenant Administrator permission for the OBS project in Global service.

Table 4-10 Automatic snapshot creation parameter

Parameter	Description
Snapshot Name Prefix	<p>The snapshot name prefix contains 1 to 32 characters and must start with a lowercase letter. Only lowercase letters, digits, hyphens (-), and underscores (_) are allowed. A snapshot name consists of a snapshot name prefix and a timestamp, for example, snapshot-1566921603720.</p>
Time Zone	<p>Time zone for the backup time, which cannot be changed. Specify Backup Started Time based on the time zone.</p>
Backup Start Time	<p>The time when the backup starts automatically every day. You can specify this parameter only in full hours, for example, 00:00 or 01:00. The value ranges from 00:00 to 23:00. Select a time from the drop-down list.</p>
Retention Period (days)	<p>The number of days that snapshots are retained in the OBS bucket. The value ranges from 1 to 90. You can specify this parameter as required. The system automatically deletes expired snapshots every hour at half past the hour.</p>

Figure 4-7 Setting parameters for automatic snapshot creation

Snapshot Name Prefix	<input type="text" value="snapshot"/>	
Time Zone	GMT+08:00	
Backup Start Time	<input type="text" value="00:00"/>	
Retention Period (days)	<input type="text" value="35"/>	

- b. Configure advanced settings for the cluster.
 - **Default:** The **Kibana Public Access**, and **Tag** functions are disabled by default. You can manually enable these functions after the cluster is created.
 - **Custom:** You can enable the **Kibana Public Access**, and **Tag** functions as required.

Table 4-11 Parameters for advanced settings

Parameter	Description
Kibana Public Access	You can configure this parameter only when security mode is enabled for a cluster. After enabling this function, you can obtain a public IP address for accessing Kibana. For details, see Kibana Public Access .
Tag	Adding tags to clusters can help you identify and manage your cluster resources. You can customize tags or use tags predefined by Tag Management Service (TMS). For details, see Managing Tags .

9. Click **Next: Confirm**. Check the configuration and click **Next** to create a cluster.
10. Click **Back to Cluster List** to switch to the **Clusters** page. The cluster you created is listed on the displayed page and its status is **Creating**. If the cluster is successfully created, its status will change to **Available**.

If the cluster creation fails, create the cluster again.

4.4 Creating an Elasticsearch Cluster in Non-Security Mode

This section describes how to create an Elasticsearch cluster in non-security mode.

Context

- When creating a cluster, the number of nodes that can be added varies according to the node type. For details, see [Table 4-12](#).

Table 4-12 Number of nodes in different types

Node Type	Number
ess	ess: 1-32
ess, ess-master	ess: 1-200 ess-master: an odd number ranging from 3 to 9
ess, ess-client	ess: 1-32 ess-client: 1-32
ess, ess-cold	ess: 1-32 ess-cold: 1-32
ess, ess-master, ess-client	ess: 1-200 ess-master: an odd number ranging from 3 to 9 ess-client: 1-32
ess, ess-master, ess-cold	ess: 1-200 ess-master: an odd number ranging from 3 to 9 ess-cold: 1-32
ess, ess-client, ess-cold	ess: 1-32 ess-client: 1-32 ess-cold: 1-32
ess, ess-master, ess-client, ess-cold	ess: 1-200 ess-master: an odd number ranging from 3 to 9 ess-client: 1-32 ess-cold: 1-32
Details about the four node types: <ul style="list-style-type: none"> ess: the default node type that is mandatory for cluster creation. The other three node types are optional. ess-master: master node ess-client: client node ess-cold: cold data node 	

Procedure




1. Log in to the CSS management console.
2. Click **Create Cluster** in the upper right corner. The **Create Cluster** page is displayed.
3. Specify **Region** and **AZ**.

Table 4-13 Parameter description for Region and AZ

Parameter	Description
Region	Select a region for the cluster from the drop-down list on the right.
AZ	Select AZs associated with the cluster region. You can select a maximum of three AZs. For details, see Deploying a Cross-AZ Cluster .

4. Configure basic cluster information.

Table 4-14 Description of basic parameters

Parameter	Description
Version	Select a cluster version from the drop-down list box.
Name	<p>Cluster name, which contains 4 to 32 characters. Only letters, numbers, hyphens (-), and underscores (_) are allowed and the value must start with a letter.</p> <p>NOTE After a cluster is created, you can modify the cluster name as required. Click the name of a cluster to be modified. On the displayed Basic Information page, click  next to the cluster name. After the modification is completed, click  to save the modification. If you want to cancel the modification, click .</p>

5. Configure cluster specifications.

Table 4-15 Parameter description

Parameter	Description
Nodes	<p>Number of nodes in a cluster.</p> <ul style="list-style-type: none"> • If neither a master node nor client node is enabled, the nodes specified by this parameter are used to serve as both the master node and client node. Nodes provide the cluster management, data storage, cluster access, and data analysis functions. To ensure data stability in the cluster, it is recommended that you set this parameter to a value no less than 3. • If only the master node function is enabled, nodes specified by this parameter are used to store data and provide functions of client nodes. • If both the master and client node functions are enabled, the nodes specified by this parameter are only used for storing data. • If only the client node function is enabled, nodes specified by this parameter are used to store data and provide functions of the master node.
CPU Architecture	Currently, x86 and Kunpeng are supported. The supported type is determined by the actual regional environment.
Node Specifications	Specifications of nodes in a cluster. You can select a specified specification based on your needs. Each cluster supports only one specification.
Node Storage Type	In the current version, the following options are available: Common I/O , High I/O , and Ultra-high I/O .
Node Storage Capacity	Storage space. Its value varies with node specifications. The node storage capacity must be a multiple of 20.
Master node	<p>The master node manages all nodes in the cluster. If more than 20 nodes are required to store and analyze a large amount of data, you are advised to enable the master node to ensure cluster stability. Otherwise, you are advised to set only the Nodes parameter and use the nodes as both master and client nodes.</p> <p>After enabling the master node, specify Node Specifications, Nodes, and Node Storage Type. The value of Nodes must be an odd number equal to or greater than 3. Up to nine nodes are supported. The value of Node Storage Capacity is fixed. You can select a storage type based on your needs.</p>

Parameter	Description
Client node	<p>The client node allows clients to access clusters and analyze data. If more than 20 nodes are required to store and analyze a large amount of data, you are advised to enable the client node to ensure cluster stability. Otherwise, you are advised to set only the Nodes parameter and use the nodes as both master and client nodes.</p> <p>After enabling the client node, specify Node Specifications, Nodes and Node Storage Type. The value of Nodes ranges from 1 to 32. The value of Node Storage Capacity is fixed. You can select a storage type based on your needs.</p>
Cold data node	<p>The cold data node is used to store historical data, for which query responses can be returned in minutes. If you do not require a quick query response, store historical data on cold data nodes to reduce costs.</p> <p>After enabling cold data node, configure Node Specifications, Nodes, Node Storage Type, and Node Storage Capacity. The value of Nodes ranges from 1 to 32. Select Node Storage Type and Node Storage Capacity as requirement.</p> <p>After the cold data node is enabled, CSS automatically adds cold and hot tags to related nodes.</p>

6. Set the enterprise project.

When creating a CSS cluster, you can bind an enterprise project to the cluster if you have enabled the enterprise project function. You can select an enterprise project created by the current user from the drop-down list on the right or click **View Project Management** to go to the **Enterprise Project Management** console and create a new project or view existing projects.

7. Set network specifications of the cluster.

Table 4-16 Parameter description

Parameter	Description
VPC	<p>A VPC is a secure, isolated, and logical network environment.</p> <p>Select the target VPC. Click View VPC to enter the VPC management console and view the created VPC names and IDs. If no VPCs are available, create one.</p> <p>NOTE The VPC must contain CIDRs. Otherwise, cluster creation will fail. By default, a VPC will contain CIDRs.</p>

Parameter	Description
Subnet	<p>A subnet provides dedicated network resources that are isolated from other networks, improving network security.</p> <p>Select the target subnet. You can access the VPC management console to view the names and IDs of the existing subnets in the VPC.</p>
Security Group	<p>A security group is a collection of access control rules for ECSs that have the same security protection requirements and are mutually trusted in a VPC. To view more details about the security group, click View Security Group.</p> <p>NOTE</p> <ul style="list-style-type: none"> For cluster access purposes, ensure that the security group contains port 9200. If your cluster version is 7.6.2 or later, ensure that all the ports used for communication between nodes in the same security group are allowed. If such settings cannot be configured, ensure at least the access to port 9300 is allowed.
Security Mode	Security mode is disabled.

8. Click **Next: Configure Advanced Settings**. Configure the automatic snapshot creation and other functions.

a. Configure **Cluster Snapshot**. Set basic configuration and snapshot configuration.

The cluster snapshot function is enabled by default. You can also disable this function as required. To store automatic snapshots in OBS, an agency will be created to access OBS. Additional cost will be incurred if snapshots are stored in standard storage.

Table 4-17 Cluster snapshot parameter

Parameter	Description
OBS bucket	<p>Select an OBS bucket for storing snapshots from the drop-down list box. You can also click Create Bucket on the right to create an OBS bucket.</p> <p>The created or existing OBS bucket must meet the following requirements:</p> <ul style="list-style-type: none"> Storage Class is Standard.

Parameter	Description
Backup Path	<p>Storage path of the snapshot in the OBS bucket.</p> <p>The backup path configuration rules are as follows:</p> <ul style="list-style-type: none"> • The backup path cannot contain the following characters: \:*?"<> • The backup path cannot start with a slash (/). • The backup path cannot start or end with a period (. • The backup path cannot contain more than 1,023 characters.
IAM Agency	<p>IAM agency authorized by the current account to CSS access or maintain data stored in the OBS bucket. You can also click Create IAM Agency on the right to create an IAM agency.</p> <p>The created or existing IAM agency must meet the following requirements:</p> <ul style="list-style-type: none"> • Agency Type must be Cloud service. • Set Cloud Service to Elasticsearch or CSS. • The agency must have the Tenant Administrator permission for the OBS project in Global service.

Table 4-18 Automatic snapshot creation parameter

Parameter	Description
Snapshot Name Prefix	<p>The snapshot name prefix contains 1 to 32 characters and must start with a lowercase letter. Only lowercase letters, digits, hyphens (-), and underscores (_) are allowed. A snapshot name consists of a snapshot name prefix and a timestamp, for example, snapshot-1566921603720.</p>
Time Zone	<p>Time zone for the backup time, which cannot be changed. Specify Backup Started Time based on the time zone.</p>
Backup Start Time	<p>The time when the backup starts automatically every day. You can specify this parameter only in full hours, for example, 00:00 or 01:00. The value ranges from 00:00 to 23:00. Select a time from the drop-down list.</p>
Retention Period (days)	<p>The number of days that snapshots are retained in the OBS bucket. The value ranges from 1 to 90. You can specify this parameter as required. The system automatically deletes expired snapshots every hour at half past the hour.</p>

Figure 4-8 Setting parameters for automatic snapshot creation

Snapshot Name Prefix	snapshot	?
Time Zone	GMT+08:00	
Backup Start Time	00:00	?
Retention Period (days)	- 35 +	?

- b. Configure advanced settings for the cluster.
 - **Default:** The **Kibana Public Access**, and **Tag** functions are disabled by default. You can manually enable these functions after the cluster is created.
 - **Custom:** You can enable the **Tag** functions as required.

Table 4-19 Parameters for advanced settings

Parameter	Description
Kibana Public Access	Clusters in non-security mode cannot access Kibana through the Internet.
Tag	Adding tags to clusters can help you identify and manage your cluster resources. You can customize tags or use tags predefined by Tag Management Service (TMS). For details, see Managing Tags .

9. Click **Next: Confirm**. Check the configuration and click **Next** to create a cluster.
10. Click **Back to Cluster List** to switch to the **Clusters** page. The cluster you created is listed on the displayed page and its status is **Creating**. If the cluster is successfully created, its status will change to **Available**.
If the cluster creation fails, create the cluster again.

4.5 Accessing an Elasticsearch Cluster

Elasticsearch clusters have built-in Kibana and Cerebro components. You can quickly access an Elasticsearch cluster through Kibana and Cerebro.

Access a Cluster Through Kibana

1. Log in to the CSS management console.
2. On the **Clusters** page, locate the target cluster and click **Access Kibana** in the **Operation** column to go to the Kibana login page.

- Non-security cluster: The Kibana console is displayed.
 - Security cluster: Enter the username and password on the login page and click **Log In** to go to the Kibana console. The default username is **admin** and the password is the one specified during cluster creation.
3. After the login is successful, you can access the Elasticsearch cluster through Kibana.

Accessing a Cluster Through Cerebro


1. Log in to the CSS management console.
2. On the **Clusters** page, locate the target cluster and click **More > Cerebro** in the **Operation** column to go to the Cerebro login page.
 - Non-security cluster: Click the cluster name on the Cerebro login page to go to the Cerebro console.
 - Security cluster: Click the cluster name on the Cerebro login page, enter the username and password, and click **Authenticate** to go to the Cerebro console. The default username is **admin** and the password is the one specified during cluster creation.
3. After the login is successful, you can access the Elasticsearch cluster through Cerebro.

4.6 Viewing Cluster Information

On the **Cluster Information** page, you can view the information about a cluster, including the private network address, public IP address, version, and node.

1. Log in to the CSS management console.
2. Choose **Clusters > Elasticsearch**. The cluster list page is displayed.
3. Click a cluster name to go to the **Cluster Information** page and view the basic information about the cluster.

Table 4-20 Parameters for configuring basic information

Type	Parameter	Description
Cluster Information	Name	User-defined cluster name. You can click  on the right to change the cluster name.
	ID	Unique ID of a cluster, which is automatically generated by the system. Each cluster in the same region has a unique ID.
	Version	Cluster version information.
	Cluster Status	Current status of a cluster

Type	Parameter	Description
	Task Status	Current task status of a cluster. If no task is in progress, -- is displayed.
	Created	Time when a cluster was created
	Cluster Storage Capacity (GB)	Storage capacity of a cluster
	Used Cluster Storage (GB)	Used storage capacity of a cluster
Configuration	Region	Region where a cluster is located
	AZ	AZ where a cluster is located
	VPC	VPC to which the cluster belongs
	Subnet	Subnet to which the cluster belongs
	Security Group	<p>Security group to which a cluster belongs.</p> <p>To change the security group of a cluster, click Change Security Group on the right.</p> <p>NOTICE</p> <p>Before changing the security group, ensure that the port 9200 required for service access has been enabled. Incorrect security group configuration may cause service access failures. Exercise caution when performing this operation.</p>
	Security Mode	<p>Security mode of a cluster.</p> <ul style="list-style-type: none"> Enabled: The current cluster is a security cluster. Disabled: The current cluster is a non-security cluster.

Type	Parameter	Description
	Reset Password	<p>This parameter is displayed only for security clusters.</p> <p>Click Reset to change the password of the administrator account admin of the security cluster.</p> <p>NOTE Requirements for administrator passwords:</p> <ul style="list-style-type: none"> • The password can contain 8 to 32 characters. • The password must contain at least three of the following character types: uppercase letters, lowercase letters, digits, and special characters. The following special characters are supported: ~!@#%&^*()-_+=+ []{};:;<.>/? • Do not use the administrator name, or the administrator name spelled backwards. • You are advised to change the password periodically.
	Enterprise Project	<p>Enterprise project to which a cluster belongs.</p> <p>You can click the project name to view the basic information about the enterprise project.</p>
	Access Control	<p>Whether to set access control for a cluster. This parameter is displayed only for clusters with public network access enabled.</p> <ul style="list-style-type: none"> • Enabled: Only IP addresses in the whitelist can access the cluster through the public network. • Disabled: Any IP address can access the cluster through the public network. <p>Click Set to configure the access control and the whitelist.</p>
	Bandwidth	<p>The bandwidth for public network access. This parameter is displayed only for clusters with public network access enabled.</p> <p>Click Edit to change the bandwidth size.</p>

Type	Parameter	Description
	HTTPS Access	<p>Whether to enable the HTTPS access protocol for a cluster.</p> <ul style="list-style-type: none"> • Disabled: The HTTP protocol is used for cluster access. • Enabled: The HTTPS protocol is used for cluster access. Only security clusters can enable this function. If HTTPS Access is enabled, you can click Download Certificate to obtain the CER security certificate for accessing the security cluster.
	Private Network Address	<p>Private IP address and port number of a cluster, which can be used to access the cluster. If the cluster has only one node, the IP address and port number of only one node are displayed, for example, 10.62.179.32:9200. If the cluster has multiple nodes, the IP addresses and port numbers of all nodes are displayed, for example, 10.62.179.32:9200,10.62.179.33:9200.</p>
Node	Node Specifications	Specifications of nodes in a cluster
	Node Storage Type	Storage capacity and storage type of nodes in a cluster
	Nodes	Number of nodes in a cluster

5 Scaling In/Out a Cluster

5.1 Overview

You can scale in or out a cluster and change cluster specifications. In this way, you can improve cluster efficiency and reduce O&M costs.

Scaling Out a Cluster

- If a data node (ess) processes many data writing and querying requests and responds slowly, you can expand its storage capacity to improve its efficiency. If some nodes turn unavailable due to the excessive data volume or misoperations, you can add new nodes to ensure the cluster availability.
- **Cold data nodes** (ess-cold) are used to share the workload of data nodes. To prevent cold data loss, you can expand the storage capacity of the cold data node or add new ones.

Changing Specifications

- If the allocation of new indexes or shards takes too long or the node coordination and scheduling are inefficient, you can change the master node (ess-master) specifications.
- If too many tasks need to be distributed or too many results have been aggregated, you can change the client node (ess-client) specifications.
- If the writing and query of a node suddenly become slow, you can change the data node (ess) specifications.
- If cold data query becomes slow, you can change the cold node (ess-cold) specifications.

Scaling in a Cluster

- If a cluster can process existing data without fully using its resources, you can scale in the cluster to reduce costs.

Removing Specified Nodes

- If a cluster can process existing data without fully using its nodes, you can remove one or more specified nodes from the cluster to reduce costs.

5.2 Scaling Out a Cluster

If the workloads on the data plane of a cluster change, you can scale out the cluster by increasing the number or capacity of its nodes. Services are not interrupted during cluster scale-out.

Prerequisites

- The target cluster is available and has no tasks in progress.
- The target cluster has sufficient quotas available.

Constraints

- Node specifications cannot be modified during cluster scale-out.
- If you change the number and storage capacity of a specified type of node, nodes in other types will not be changed.
- The quota of nodes in different types varies. For details, see [Table 5-1](#).

Table 5-1 Number of nodes in different types

Node Type	Number
ess	ess: 1-32
ess, ess-master	ess: 1-200 ess-master: an odd number ranging from 3 to 9
ess, ess-client	ess: 1-32 ess-client: 1-32
ess, ess-cold	ess: 1-32 ess-cold: 1-32
ess, ess-master, ess-client	ess: 1-200 ess-master: an odd number ranging from 3 to 9 ess-client: 1-32
ess, ess-master, ess-cold	ess: 1-200 ess-master: an odd number ranging from 3 to 9 ess-cold: 1-32
ess, ess-client, ess-cold	ess: 1-32 ess-client: 1-32 ess-cold: 1-32

Node Type	Number
ess, ess-master, ess-client, ess-cold	ess: 1-200 ess-master: an odd number ranging from 3 to 9 ess-client: 1-32 ess-cold: 1-32
Details about the four node types: <ul style="list-style-type: none"> ● ess: the default node type that is mandatory for cluster creation. The other three node types are optional. ● ess-master: master node ● ess-client: client node ● ess-cold: cold data node 	

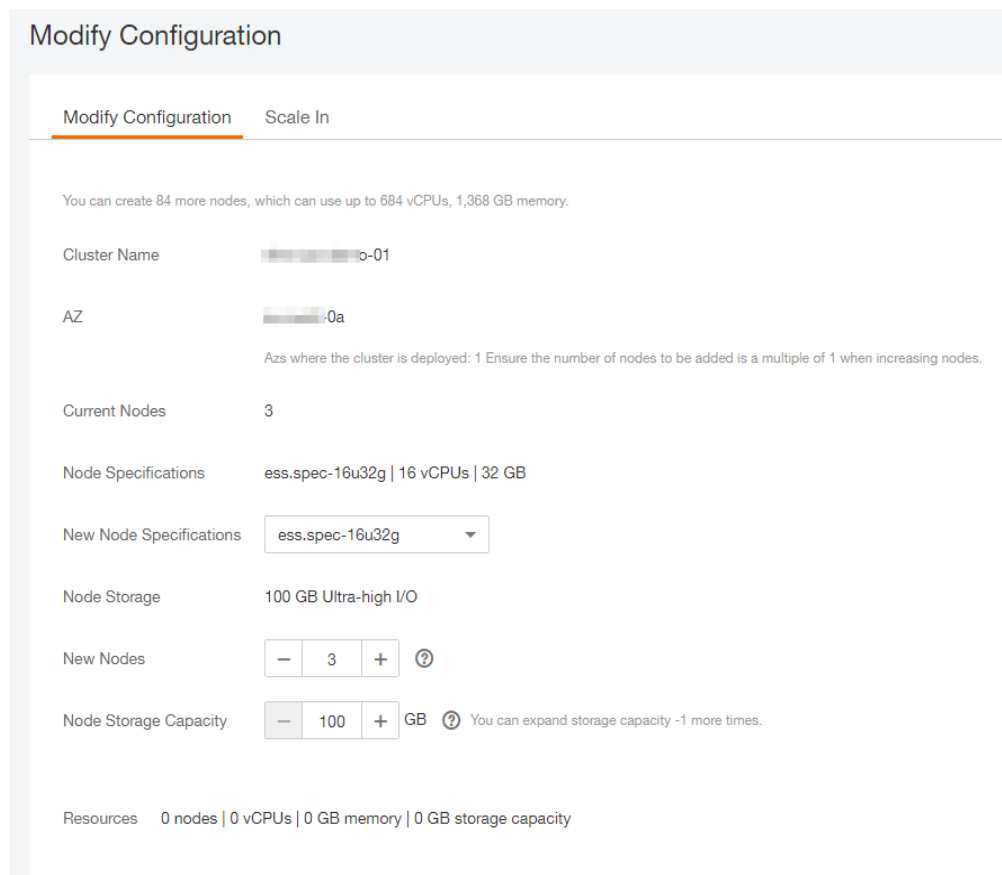
Procedure

1. Log in to the CSS management console.
2. In the navigation pane on the left, choose **Clusters > Elasticsearch**. On the displayed **Clusters** page, locate the target cluster and choose **More > Modify Configuration** in the **Operation** column.
3. On the **Modify Configuration** page, choose the **Scale Cluster** tab and click **Scale out** to set parameters.
 - **New Nodes**: The number of default nodes. For details about the quota, see [Table 5-1](#).
 - **Node Storage Capacity**: The storage capacity of the default nodes. The quota is determined by the **New Node Specifications**. The value must be a multiple of 20.

NOTE

If a cluster has master, client, or cold data nodes, you can change the number of these nodes and expand the storage capacity of cold data nodes. For details about the quotas of the master, client, and cold data node, see [Table 5-1](#).

Figure 5-1 Scaling out a cluster



4. Click **Next: Scale Now**.
5. Confirm the information and click **Submit**.
6. Click **Back to Cluster List** to switch to the **Clusters** page. The **Task Status** is **Scaling out**. When **Cluster Status** changes to **Available**, the cluster has been successfully scaled out.

5.3 Changing Specifications

If the workloads on the data plane of a cluster change, you can change its node specifications as needed.

Prerequisites

- The target cluster is available and has no tasks in progress.
- The target cluster has sufficient quotas available.
- When changing the node specifications, ensure that all service data has copies so the services will not be interrupted.

Run the **GET _cat/indices?v** command in Kibana. If the returned **rep** value is greater than **0**, the data has copies. If the returned **rep** value is **0**, the data has no copies. In this case, create snapshot for the cluster by referring to [Manually Creating a Snapshot](#).

- If the data volume is large, it may take long to modify the node specifications. You are advised to modify specifications during off-peak hours.

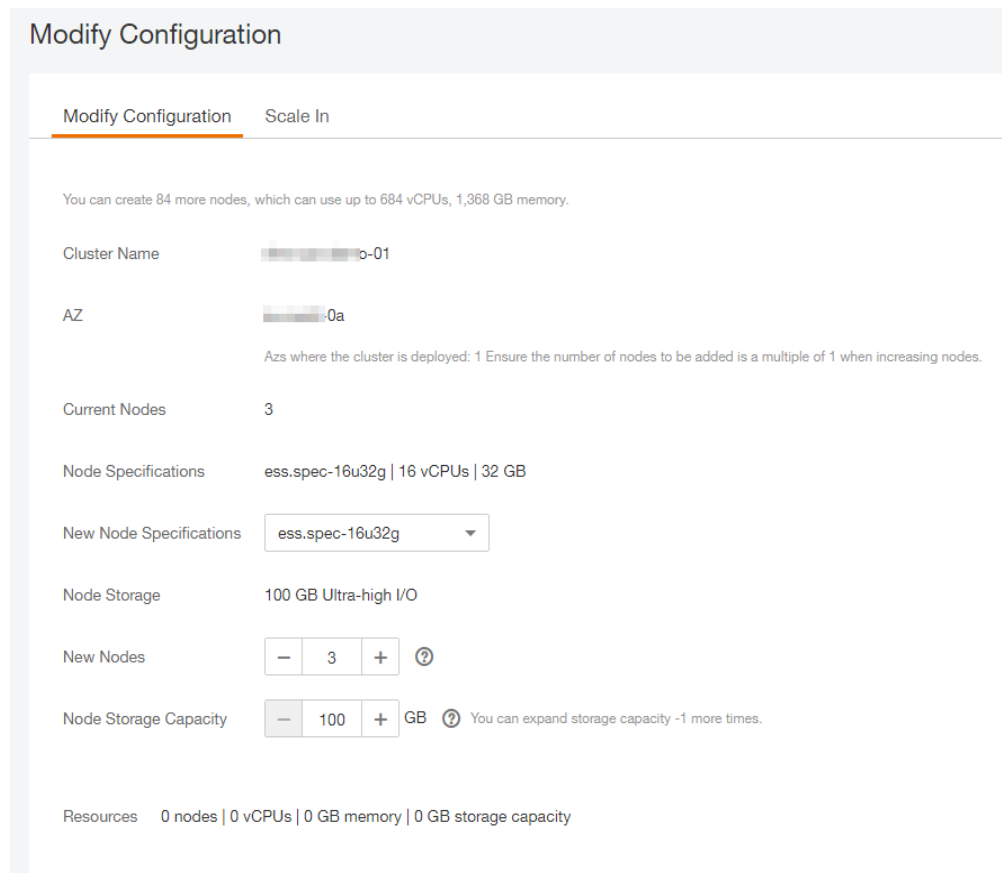
Constraints

- The node number and storage capacity cannot be modified when you change the node specifications.
- After decreasing cluster specifications, the cluster performance will deteriorate and service capabilities will be affected. Exercise caution when performing this operation.
- If a cluster has multiple node types, you can change the specifications of only one type at a time. After the change, nodes in other types still maintain their original specifications.
- Kibana is unavailable during specification change.
- During the specification modification, the nodes are stopped and restarted in sequence. It is a rolling process.

Procedure

1. Log in to the CSS management console.
2. In the navigation pane on the left, choose **Clusters > Elasticsearch**. On the displayed **Clusters** page, locate the target cluster and choose **More > Modify Configuration** in the **Operation** column.
3. On the **Modify Configuration** page, choose the **Scale Cluster** tab and click **Change Specifications** to set parameters.
 - **New Node Specifications** The specifications of the default data nodes. Select the specifications from the drop-down list as required.
 - If a cluster has master nodes, client nodes, or cold data nodes, you can change their specifications.

Figure 5-2 Changing cluster specifications



4. Click **Next: Scale Now**.
5. Confirm the information and click **Submit**.
6. In the displayed **Verify Index Copy** dialog box, select **Verify index copies** if you need. Click **OK**.
 - If you selected **Verify index copies** and the cluster has no master node, indexes must have at least one copy and the cluster must have at least three nodes.
 - If you selected **Verify index copies** and the cluster has no master node, indexes must have at least one copy.
7. Click **Back to Cluster List** to switch to the **Clusters** page. The **Cluster Status** is **Configuration modified**. When **Cluster Status** changes to **Available**, the cluster specifications have been successfully modified.

5.4 Scaling in a Cluster

If a cluster can process existing data without fully using its resources, you can scale in the cluster to reduce costs. Services are not interrupted during cluster scale-in.

Prerequisites

The target cluster is available and has no tasks in progress.

Constraints

- The node specifications and storage capacity cannot be modified during scale-in.
- If you change the number and storage capacity of a specified type of node, nodes in other types will not be changed.
- Ensure that the disk usage after scale-in is less than 80% and each AZ of each node type has at least one node.
- When scaling in a cluster, the data in the node to be deleted is migrated to other nodes. The timeout threshold for data migration is five hours. If data migration is not complete within 5 hours, the cluster scale-in fails. You are advised to perform scale-in for multiple times when the cluster has huge amounts of data.
- For a cluster without master nodes, the number of remaining data nodes (including cold data nodes and other types of nodes) after scale-in must be greater than half of the original node number, and greater than the maximum number of index replicas.
- The quota of nodes in different types varies. For details, see [Table 5-2](#).

Table 5-2 Number of nodes in different types

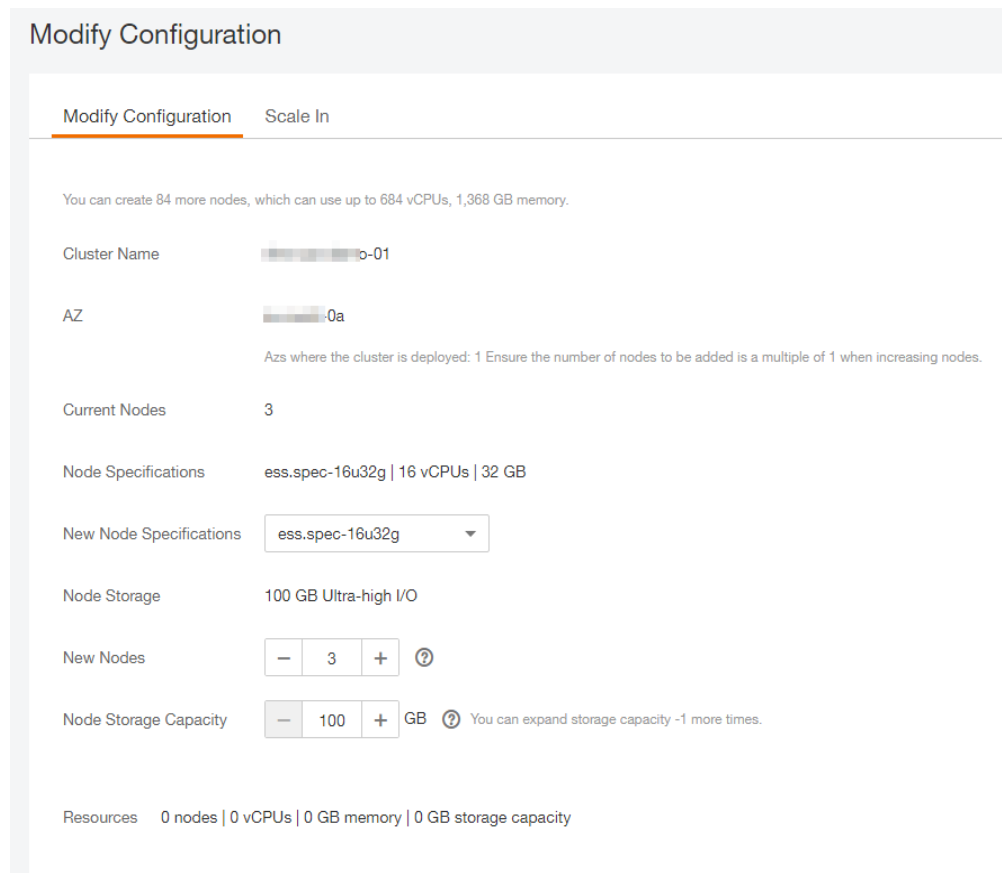
Node Type	Number
ess	ess: 1-32
ess, ess-master	ess: 1-200 ess-master: an odd number ranging from 3 to 9
ess, ess-client	ess: 1-32 ess-client: 1-32
ess, ess-cold	ess: 1-32 ess-cold: 1-32
ess, ess-master, ess-client	ess: 1-200 ess-master: an odd number ranging from 3 to 9 ess-client: 1-32
ess, ess-master, ess-cold	ess: 1-200 ess-master: an odd number ranging from 3 to 9 ess-cold: 1-32
ess, ess-client, ess-cold	ess: 1-32 ess-client: 1-32 ess-cold: 1-32

Node Type	Number
ess, ess-master, ess-client, ess-cold	ess: 1-200 ess-master: an odd number ranging from 3 to 9 ess-client: 1-32 ess-cold: 1-32
Details about the four node types: <ul style="list-style-type: none"> ● ess: the default node type that is mandatory for cluster creation. The other three node types are optional. ● ess-master: master node ● ess-client: client node ● ess-cold: cold data node 	

Procedure

1. Log in to the CSS management console.
2. In the navigation pane on the left, choose **Clusters > Elasticsearch**. On the displayed **Clusters** page, locate the target cluster and choose **More > Modify Configuration** in the **Operation** column.
3. On the **Modify Configuration** page, choose the **Scale Cluster** tab and click **Scale in** to set parameters.
 - **New Nodes**: The number of default nodes. For details about the quota, see [Table 5-2](#).
 - If a cluster has master nodes, client nodes, or cold data nodes, you can change the node number. For details about the quotas of the master, client, and cold data node, see [Table 5-2](#).

Figure 5-3 Scaling in a cluster



4. Click **Next: Scale Now**.
5. Confirm the information and click **Submit**.
6. Click **Back to Cluster List** to switch to the **Clusters** page. The **Task Status** is **Scaling in**. When **Cluster Status** changes to **Available**, the cluster has been successfully scaled in.

5.5 Removing Specified Nodes

If a cluster can process existing data without fully using its nodes, you can remove one or more specified nodes from the cluster to reduce costs. Services will not be interrupted during the removal of specified nodes.

Prerequisites

The target cluster is available and has no tasks in progress.

Constraints

- Ensure that the disk usage after scale-in is less than 80% and each AZ of each node type has at least one node.
- In a cross-AZ cluster, the difference between the numbers of the same type nodes in different AZs cannot exceed 1.
- For a cluster without master nodes, the number of removed data nodes and cold data nodes in a scale-in must be fewer than half of the original number.

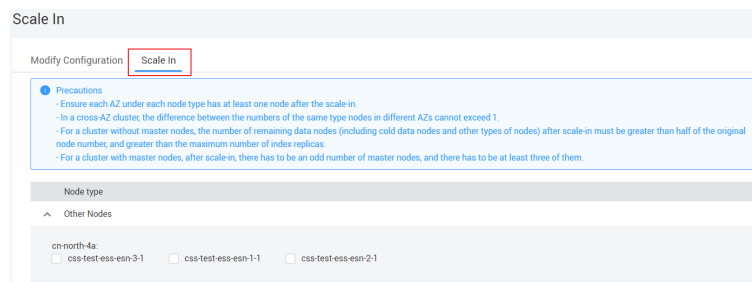
of data nodes and cold data nodes, and the number of remaining data nodes and cold data nodes after a scale-in must be greater than the maximum number of index replicas.

- For a cluster with master nodes, the number of removed master nodes in a scale-in must be fewer than half of the original master node number. After scale-in, there has to be an odd number of master nodes, and there has to be at least three of them.

Procedure

1. Log in to the CSS management console.
2. In the navigation pane on the left, choose **Clusters > Elasticsearch**. On the displayed **Clusters** page, locate the target cluster and choose **More > Modify Configuration** in the **Operation** column.
3. On the **Modify Configuration** page, click the **Scale In** tab.
4. Select the target nodes.

Figure 5-4 Scaling in a cluster by removing specific nodes



5. Click **Next: Scale Now**.
6. Confirm the information and click **Submit**.
7. Click **Back to Cluster List** to switch to the **Clusters** page. The **Task Status** is **Scaling in**. When **Cluster Status** changes to **Available**, the cluster has been successfully scaled in.

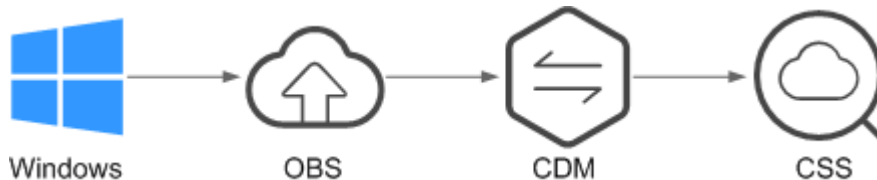
6 Importing Data to Elasticsearch

6.1 Using CDM to Import Data from OBS to Elasticsearch

You can use the CDM-provided wizard to import data stored in OBS to Elasticsearch in CSS. Data files can be in the JSON or CSV format.

Figure 6-1 shows the data transmission process.

Figure 6-1 Process of using CDM to import OBS data to Elasticsearch



Procedure

1. Log in to the OBS management console.
2. Create an OBS bucket for storing data.
For details, see "Creating a Bucket" in the *Object Storage Service Console Operation Guide*.
The OBS bucket must be in the same region as the cluster.
3. Upload the data file to the OBS bucket.
For details, see "Uploading a File" in the *Object Storage Service Console Operation Guide*.
For example, save the following data as a JSON file and upload the file to the OBS bucket.

```

{"productName":"Latest art shirts for women in autumn 2017","size":"L"}
{"productName":"Latest art shirts for women in autumn 2017","size":"M"}
{"productName":"Latest art shirts for women in autumn 2017","size":"S"}
{"productName":"Latest jeans for women in spring 2018","size":"M"}
{"productName":"Latest jeans for women in spring 2018","size":"S"}
{"productName":"Latest casual pants for women in spring 2017","size":"L"}
{"productName":"Latest casual pants for women in spring 2017","size":"S"}
  
```

4. Log in to the CSS management console.
5. In the navigation pane on the left, choose **Clusters > Elasticsearch** to switch to the **Clusters** page.
6. From the cluster list, locate the row that contains the cluster to which you want to import data, and click **Access Kibana** in the **Operation** column.
7. In the Kibana navigation pane on the left, choose **Dev Tools**.
8. On the **Console** page, run the related command to create an index for the data to be stored and specify a custom mapping to define the data type:

If there is an available index in the cluster where you want to import data, this step is not required. If there is no available index, create an index by referring to the following sample code.

For example, on the **Console** page, run the following command to create index **demo** and specify a user-defined mapping to define the data type:

Versions earlier than 7.x

```
PUT /demo
{
  "settings": {
    "number_of_shards": 1
  },
  "mappings": {
    "products": {
      "properties": {
        "productName": {
          "type": "text",
          "analyzer": "ik_smart"
        },
        "size": {
          "type": "keyword"
        }
      }
    }
  }
}
```

Versions later than 7.x

```
PUT /demo
{
  "settings": {
    "number_of_shards": 1
  },
  "mappings": {
    "properties": {
      "productName": {
        "type": "text",
        "analyzer": "ik_smart"
      },
      "size": {
        "type": "keyword"
      }
    }
  }
}
```

The command is successfully executed if the following information is displayed.

```
{
  "acknowledged" : true,
  "shards_acknowledged" : true,
  "index" : "demo"
}
```

9. Log in to the CDM management console.

10. Purchase a CDM cluster.
For details, see "Creating a Cluster" in the *Cloud Data Migration User Guide*.
11. Create a link between CDM and CSS.
For details, see "Creating a Link" in the *Cloud Data Migration User Guide*.
12. Create a link between CDM and OBS.
For details, see "Creating a Link" in the *Cloud Data Migration User Guide*.
13. Create a job on the CDM cluster and migrate the data in the OBS bucket to the target cluster in CSS.
For details, see "Table/File Migration" in the *Cloud Data Migration User Guide*.
14. On the **Console** page of Kibana, search for the imported data.
On the **Console** page of Kibana, run the following command to search for data. View the search results. If the searched data is consistent with the imported data, the data has been imported successfully.

```
GET demo/_search
```

The command is successfully executed if the following information is displayed.

```
{
  "took": 18,
  "timed_out": false,
  "_shards": {
    "total": 1,
    "successful": 1,
    "skipped": 0,
    "failed": 0
  },
  "hits": {
    "total": 7,
    "max_score": 1,
    "hits": [
      {
        "_index": "demo",
        "_type": "products",
        "_id": "g6UepnEBuvdFwWkRmn4V",
        "_score": 1,
        "_source": {
          "size": "L",
          "productName": "Latest art shirts for women in autumn 2017"
        }
      },
      {
        "_index": "demo",
        "_type": "products",
        "_id": "hKUepnEBuvdFwWkRmn4V",
        "_score": 1,
        "_source": {
          "size": "M",
          "productName": "Latest art shirts for women in autumn 2017"
        }
      },
      {
        "_index": "demo",
        "_type": "products",
        "_id": "haUepnEBuvdFwWkRmn4V",
        "_score": 1,
        "_source": {
          "size": "S",
          "productName": "Latest art shirts for women in autumn 2017"
        }
      }
    ]
  }
}
```

```
{
  "_index": "demo",
  "_type": "products",
  "_id": "hqUepnEBuvdFwWkRmn4V",
  "_score": 1,
  "_source": {
    "size": "M",
    "productName": "Latest jeans for women in autumn 2018"
  }
},
{
  "_index": "demo",
  "_type": "products",
  "_id": "h6UepnEBuvdFwWkRmn4V",
  "_score": 1,
  "_source": {
    "size": "S",
    "productName": "Latest jeans for women in autumn 2018"
  }
},
{
  "_index": "demo",
  "_type": "products",
  "_id": "iKUepnEBuvdFwWkRmn4V",
  "_score": 1,
  "_source": {
    "size": "L",
    "productName": "Latest casual pants for women in autumn 2017"
  }
},
{
  "_index": "demo",
  "_type": "products",
  "_id": "iaUepnEBuvdFwWkRmn4V",
  "_score": 1,
  "_source": {
    "size": "S",
    "productName": "Latest casual pants for women in autumn 2017"
  }
}
]
```

NOTE

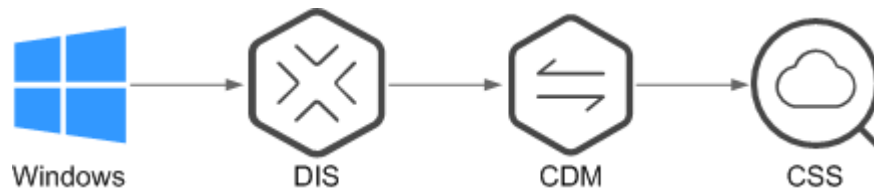
demo specifies the created index name. Set this parameter based on site requirements.

6.2 Using DIS to Import Local Data to Elasticsearch

You can use DIS to upload log data stored on the local Windows PC to the DIS queue and use CDM to migrate the data to Elasticsearch in CSS. In this way, you can efficiently manage and obtain logs through Elasticsearch. Data files can be in the JSON or CSV format.

Figure 6-2 shows the data transmission process.

Figure 6-2 Process of using DIS to import local data to Elasticsearch



Procedure

1. Log in to the DIS management console.
2. Purchase a DIS stream.
For details, see "Creating a DIS Stream" in the *Data Ingestion Service User Guide*.
3. Install and configure DIS Agent.
For details, see "Installing DIS Agent" and "Configuring DIS Agent" in *Data Ingestion Service User Guide*.
4. Start DIS Agent and upload the collected local data to the DIS queue.
For details, see "Starting DIS Agent" in the *Data Ingestion Service User Guide*.
For example, upload the following data to a DIS queue using the DIS Agent:

```

{"logName":"aaa","date":"bbb"}
{"logName":"ccc","date":"ddd"}
{"logName":"eee","date":"fff"}
{"logName":"ggg","date":"hhh"}
{"logName":"mmm","date":"nnn"}
  
```

5. Log in to the CSS management console.
6. In the navigation pane on the left, choose **Clusters > Elasticsearch** to switch to the **Clusters** page.
7. From the cluster list, locate the row that contains the cluster to which you want to import data, and click **Access Kibana** in the **Operation** column.
8. In the Kibana navigation pane on the left, choose **Dev Tools**.
9. On the **Console** page, run the related command to create an index for the data to be stored and specify a custom mapping to define the data type:
If there is an available index in the cluster where you want to import data, this step is not required. If there is no available index, create an index by referring to the following sample code.

For example, on the **Console** page, run the following command to create index **apache** and specify a custom mapping to define the data type:

Versions earlier than 7.x

```

PUT /apache
{
  "settings": {
    "number_of_shards": 1
  },
  "mappings": {
    "logs": {
      "properties": {
        "logName": {
          "type": "text",
          "analyzer": "ik_smart"
        },
        "date": {
  
```

```

        "type": "keyword"
      }
    }
  }
}

```

Versions later than 7.x

```

PUT /apache
{
  "settings": {
    "number_of_shards": 1
  },
  "mappings": {
    "properties": {
      "logName": {
        "type": "text",
        "analyzer": "ik_smart"
      },
      "date": {
        "type": "keyword"
      }
    }
  }
}

```

The command is successfully executed if the following information is displayed.

```

{
  "acknowledged" : true,
  "shards_acknowledged" : true,
  "index" : "apache"
}

```

10. Log in to the CDM management console.
11. Purchase a CDM cluster.
For details, see "Creating a Cluster" in the *Cloud Data Migration User Guide*.
12. Create a link between CDM and CSS.
For details, see "Creating a Link" in the *Cloud Data Migration User Guide*.
13. Create a link between CDM and DIS.
For details, see "Creating a Link" in the *Cloud Data Migration User Guide*.
14. Create a job on the purchased CDM cluster and migrate the data in the DIS queue to the target cluster in CSS.
For details, see "Table/File Migration" in the *Cloud Data Migration User Guide*.
15. On the **Console** page of Kibana, search for the imported data.
On the **Console** page of Kibana, run the following command to search for data. View the search results. If the searched data is consistent with the imported data, the data has been imported successfully.

```
GET apache/_search
```

The command is successfully executed if the following information is displayed.

```

{
  "took": 81,
  "timed_out": false,
  "_shards": {
    "total": 1,
    "successful": 1,
    "skipped": 0,

```

```

"failed": 0
},
"hits": {
  "total": 5,
  "max_score": 1,
  "hits": [
    {
      "_index": "apache",
      "_type": "logs",
      "_id": "txfbqnEBPuwwWJWL-qvP",
      "_score": 1,
      "_source": {
        "date": """"{"logName":"aaa"}""",
        "logName": """"date":"bbb"}""
      }
    },
    {
      "_index": "apache",
      "_type": "logs",
      "_id": "uBfbqnEBPuwwWJWL-qvP",
      "_score": 1,
      "_source": {
        "date": """"{"logName":"ccc"}""",
        "logName": """"date":"ddd"}""
      }
    },
    {
      "_index": "apache",
      "_type": "logs",
      "_id": "uRfbqnEBPuwwWJWL-qvP",
      "_score": 1,
      "_source": {
        "date": """"{"logName":"eee"}""",
        "logName": """"date":"fff"}""
      }
    },
    {
      "_index": "apache",
      "_type": "logs",
      "_id": "uhfbqnEBPuwwWJWL-qvP",
      "_score": 1,
      "_source": {
        "date": """"{"logName":"ggg"}""",
        "logName": """"date":"hhh"}""
      }
    },
    {
      "_index": "apache",
      "_type": "logs",
      "_id": "uxfbqnEBPuwwWJWL-qvP",
      "_score": 1,
      "_source": {
        "date": """"{"logName":"mmm"}""",
        "logName": """"date":"nnn"}""
      }
    }
  ]
}

```

 **NOTE**

apache specifies the created index name. Set this parameter based on site requirements.

6.3 Using Logstash to Import Data to Elasticsearch

You can use Logstash to collect data and migrate collected data to Elasticsearch in CSS. This method helps you effectively obtain and manage data through Elasticsearch. Data files can be in the JSON or CSV format.

Logstash is an open-source, server-side data processing pipeline that ingests data from a multitude of sources simultaneously, transforms it, and then sends it to Elasticsearch. For details about Logstash, visit the following website: <https://www.elastic.co/guide/en/logstash/current/getting-started-with-logstash.html>

The following two scenarios are involved depending on the Logstash deployment:

- [Importing Data When Logstash Is Deployed on the External Network](#)
- [Importing Data When Logstash Is Deployed on an ECS](#)

Prerequisites

- To facilitate operations, you are advised to deploy Logstash on a host that runs the Linux operating system (OS).
- To download Logstash, visit the following website: <https://www.elastic.co/downloads/logstash-oss>

NOTE

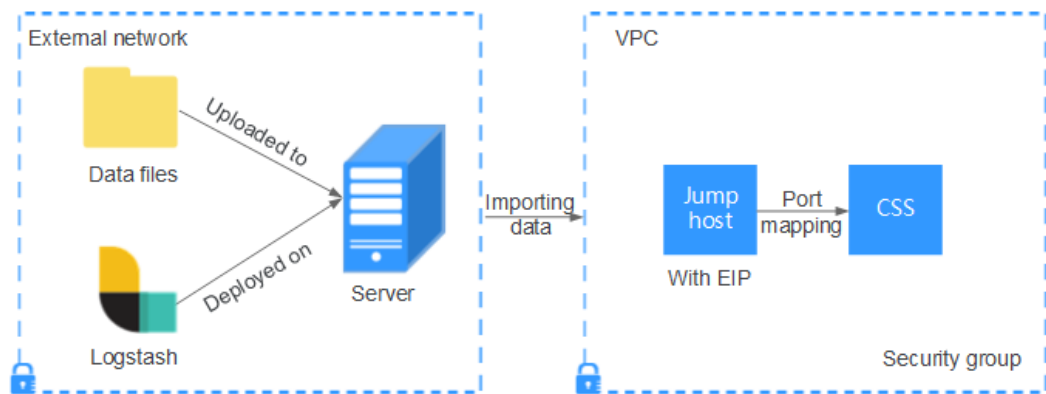
Logstash requires an OSS version same as the CSS version.

- After installing Logstash, perform the following steps to import data. For details about how to install Logstash, visit the following website: <https://www.elastic.co/guide/en/logstash/current/installing-logstash.html>
- The JDK must be installed before Logstash is installed. In Linux OS, you can run the `yum -y install java-1.8.0` command to install JDK 1.8.0. In Windows OS, you can download the required JDK version from the [official website of JDK](#), and install it by following the installation guide.
- In the [Importing Data When Logstash Is Deployed on an ECS](#) scenario, ensure that the ECS and the Elasticsearch cluster to which data is imported reside in the same VPC.

Importing Data When Logstash Is Deployed on the External Network

[Figure 6-3](#) illustrates how data is imported when Logstash is deployed on an external network.

Figure 6-3 Importing data when Logstash is deployed on an external network



1. Create a jump host and configure it as follows:
 - The jump host is an ECS running the Linux OS and has been bound with an EIP.
 - The jump host resides in the same VPC as the CSS cluster.
 - SSH local port forwarding is configured for the jump host to forward requests from a chosen local port to port **9200** on one node of the CSS cluster.
 - Refer to [SSH documentation](#) for the local port forwarding configuration.
2. Use PuTTY to log in to the created jump host with the EIP.
3. Run the following command to perform port mapping and transfer the request sent to the port on the jump host to the target cluster:


```
ssh -g -L <Local port of the jump host:Private network address and port number of a node> -N -f root@<Private IP address of the jump host>
```

NOTE

- In the preceding command, *<Local port of the jump host>* refers to the port obtained in **1**.
- In the preceding command, *<Private network address and port number of a node>* refers to the private network address and port number of a node in the cluster. If the node is faulty, the command execution will fail. If the cluster contains multiple nodes, you can replace the value of **<private network address and port number of a node>** with the private network address and port number of any available node in the cluster. If the cluster contains only one node, restore the node and execute the command again.
- Replace *<Private IP address of the jump host>* in the preceding command with the IP address (with **Private IP**) of the created jump host in the **IP Address** column in the ECS list on the ECS management console.

For example, port **9200** on the jump host is assigned external network access permissions, the private network address and port number of the node are **192.168.0.81** and **9200**, respectively, and the private IP address of the jump host is **192.168.0.227**. You need to run the following command to perform port mapping:

```
ssh -g -L 9200:192.168.0.81:9200 -N -f root@192.168.0.227
```

4. Log in to the server where Logstash is deployed and store the data files to be imported on the server.

For example, data file `access_20181029_log` needs to be imported, the file storage path is `/tmp/access_log/`, and the data file includes the following data:

 **NOTE**

Create the `access_log` folder if it does not exist.

All	Heap used for segments		18.6403	MB
All	Heap used for doc values		0.119289	MB
All	Heap used for terms		17.4095	MB
All	Heap used for norms		0.0767822	MB
All	Heap used for points		0.225246	MB
All	Heap used for stored fields		0.809448	MB
All	Segment count		101	
All	Min Throughput	index-append	66232.6	docs/s
All	Median Throughput	index-append	66735.3	docs/s
All	Max Throughput	index-append	67745.6	docs/s
All	50th percentile latency	index-append	510.261	ms

5. In the server where Logstash is deployed, run the following command to create configuration file `logstash-simple.conf` in the Logstash installation directory:

```
cd /<Logstash installation directory>/
vi logstash-simple.conf
```

6. Input the following content in `logstash-simple.conf`:

```
input {
  Location of data
}
filter {
  Related data processing
}
output {
  elasticsearch {
    hosts => "<EIP of the jump host>:<Number of the port assigned external network access
permissions on the jump host>"
  }
}
```

- The **input** parameter indicates the data source. Set this parameter based on the actual conditions. For details about the **input** parameter and parameter usage, visit the following website: <https://www.elastic.co/guide/en/logstash/current/input-plugins.html>
- The **filter** parameter specifies the mode in which data is processed. For example, extract and process logs to convert unstructured information into structured information. For details about the **filter** parameter and parameter usage, visit the following website: <https://www.elastic.co/guide/en/logstash/current/filter-plugins.html>
- The **output** parameter indicates the destination address of the data. For details about the **output** parameter and parameter usage, visit <https://www.elastic.co/guide/en/logstash/current/output-plugins.html>. Replace `<EIP address of the jump host>` with the IP address (with **EIP**) of the created jump host in the **IP Address** column in the ECS list on the ECS management console. `<Number of the port assigned external network access permissions on the jump host>` is the number of the port obtained in **1**, for example, **9200**.

Consider the data files in the `/tmp/access_log/` path mentioned in [4](#) as an example. Assume that data import starts from data in the first row of the data file, the filtering condition is left unspecified (indicating no data processing operations are performed), the public IP address and port number of the jump host are **192.168.0.227** and **9200**, respectively, and the name of

the target index is **myindex**. Edit the configuration file as follows, and enter **:wq** to save the configuration file and exit.

```
input {
  file {
    path => "/tmp/access_log/*"
    start_position => "beginning"
  }
}
filter {
}
output {
  elasticsearch {
    hosts => "192.168.0.227:9200"
    index => "myindex"
  }
}
```





 **NOTE**

If a license error is reported, set **ilm_enabled** to **false**.

If the cluster has the security mode enabled, you need to download a certificate first.

- a. Download a certificate on the **Basic Information** page of the cluster.

Figure 6-4 Downloading a certificate

Region	eu- 
VPC	nhc-  ps
Security Group	nhc-  group1
Security Mode	Enabled Download Certificate
Public IP Address	90.  :9200 Disassociate
Bandwidth	101 Mbit/s Edit
HTTPS Access	Enabled

- b. Store the certificate to the server where Logstash is deployed.
- c. Modify the **logstash-simple.conf** configuration file.

Consider the data files in the **/tmp/access_log/** path mentioned in **4** as an example. Assume that data import starts from data in the first row of the data file, the filtering condition is left unspecified (indicating no data processing operations are performed), and the public IP address and port number of the jump host are **192.168.0.227** and **9200**, respectively. The name of the index for importing data is **myindex**, and the certificate is stored in **/logstash/logstash6.8/config/CloudSearchService.cer**. Edit the configuration file as follows, and enter **:wq** to save the configuration file and exit.

```
input{
  file {
```

```
    path => "/tmp/access_log/*"
    start_position => "beginning"
  }
}
filter {
}
output{
  elasticsearch{
    hosts => ["https://192.168.0.227:9200"]
    index => "myindex"
    user => "admin"
    password => "*****"
    cacert => "/logstash/logstash6.8/config/CloudSearchService.cer"
  }
}
```

NOTE

password: password for logging in to the cluster

7. Run the following command to import the data collected by Logstash to the cluster:

```
./bin/logstash -f logstash-simple.conf
```

NOTE

This command must be executed in the directory where the **logstash-simple.conf** file is stored. For example, if the **logstash-simple.conf** file is stored in **/root/logstash-7.1.1/**, go to the directory before running the command.

8. Log in to the CSS management console.
9. In the navigation pane on the left, choose **Clusters > Elasticsearch** to switch to the **Clusters** page.
10. From the cluster list, locate the row that contains the cluster to which you want to import data and click **Access Kibana** in the **Operation** column.
11. In the Kibana navigation pane on the left, choose **Dev Tools**.
12. On the **Console** page of Kibana, search for the imported data.

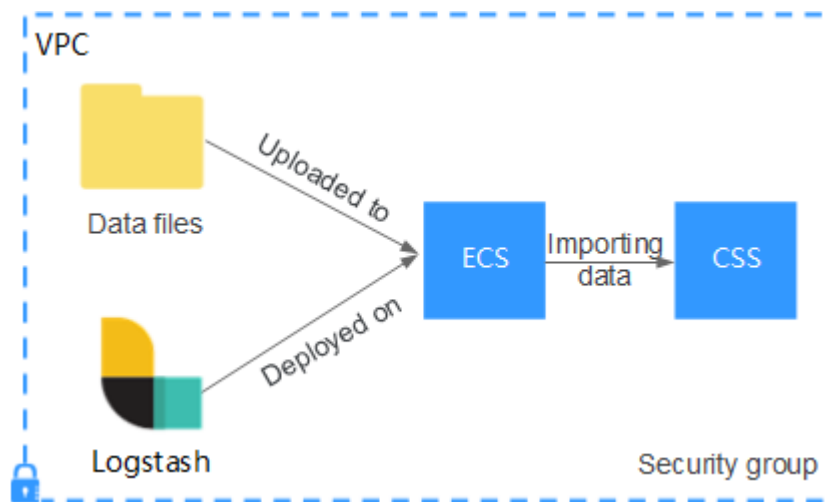
On the **Console** page of Kibana, run the following command to search for data. View the search results. If the searched data is consistent with the imported data, the data has been imported successfully.

```
GET myindex/_search
```

Importing Data When Logstash Is Deployed on an ECS

Figure 6-5 illustrates how data is imported when Logstash is deployed on an ECS that resides in the same VPC as the cluster to which data is to be imported.

Figure 6-5 Importing data when Logstash is deployed on an ECS



1. Ensure that the ECS where Logstash is deployed and the cluster to which data is to be imported reside in the same VPC, port **9200** of the ECS security group has been assigned external network access permissions, and an EIP has been bound to the ECS.
2. Use PuTTY to log in to the ECS.

For example, data file **access_20181029_log** is stored in the **/tmp/access_log/** path of the ECS, and the data file includes the following data:

All	Heap used for segments		18.6403	MB
All	Heap used for doc values		0.119289	MB
All	Heap used for terms		17.4095	MB
All	Heap used for norms		0.0767822	MB
All	Heap used for points		0.225246	MB
All	Heap used for stored fields		0.809448	MB
All	Segment count		101	
All	Min Throughput	index-append	66232.6	docs/s
All	Median Throughput	index-append	66735.3	docs/s
All	Max Throughput	index-append	67745.6	docs/s
All	50th percentile latency	index-append	510.261	ms

3. Run the following command to create configuration file **logstash-simple.conf** in the Logstash installation directory:

```
cd /<Logstash installation directory>/
vi logstash-simple.conf
```

Input the following content in **logstash-simple.conf**:

```
input {
  Location of data
}
filter {
  Related data processing
}
output {
  elasticsearch{
    hosts => "<Private network address and port number of the node>"
  }
}
```

- The **input** parameter indicates the data source. Set this parameter based on the actual conditions. For details about the **input** parameter and parameter usage, visit the following website: <https://www.elastic.co/guide/en/logstash/current/input-plugins.html>
- The **filter** parameter specifies the mode in which data is processed. For example, extract and process logs to convert unstructured information

into structured information. For details about the **filter** parameter and parameter usage, visit the following website: <https://www.elastic.co/guide/en/logstash/current/filter-plugins.html>

- The **output** parameter indicates the destination address of the data. For details about the **output** parameter and parameter usage, visit <https://www.elastic.co/guide/en/logstash/current/output-plugins.html>. *<private network address and port number of a node>* refers to the private network address and port number of a node in the cluster. If the cluster contains multiple nodes, you are advised to replace the value of *<Private network address and port number of a node>* with the private network addresses and port numbers of all nodes in the cluster to prevent node faults. Use commas (,) to separate the nodes' private network addresses and port numbers. The following is an example:

```
hosts => ["192.168.0.81:9200","192.168.0.24:9200"]
```

If the cluster contains only one node, the format is as follows:

```
hosts => "192.168.0.81:9200"
```

Consider the data files in the **/tmp/access_log/** path mentioned in [2](#) as an example. Assume that data import starts from data in the first row of the data file, the filtering condition is left unspecified (indicating no data processing operations are performed), the private network address and port number of the node in the cluster where data is to be imported are **192.168.0.81** and **9200**, respectively, and the name of the target index is **myindex**. Edit the configuration file as follows, and enter **:wq** to save the configuration file and exit.

```
input {
  file{
    path => "/tmp/access_log/*"
    start_position => "beginning"
  }
}
filter {
}
output {
  elasticsearch {
    hosts => "192.168.0.81:9200"
    index => "myindex"
  }
}
```

If the cluster has the security mode enabled, you need to download a certificate first.

- a. Download a certificate on the **Basic Information** page of the cluster.

Figure 6-6 Downloading a certificate

Region	eu-
VPC	nh-
Security Group	nh-
Security Mode	Enabled Download Certificate
Public IP Address	90.168.0.227:9200 Disassociate
Bandwidth	101 Mbit/s Edit
HTTPS Access	Enabled

- b. Store the certificate to the server where Logstash is deployed.
- c. Modify the **logstash-simple.conf** configuration file.

Consider the data files in the **/tmp/access_log/** path mentioned in 2 as an example. Assume that data import starts from data in the first row of the data file, the filtering condition is left unspecified (indicating no data processing operations are performed), the public IP address and port number of the jump host are **192.168.0.227** and **9200**, respectively. The name of the index for importing data is **myindex**, and the certificate is stored in **/logstash/logstash6.8/config/CloudSearchService.cer**. Edit the configuration file as follows, and enter **:wq** to save the configuration file and exit.

```
input{
  file {
    path => "/tmp/access_log/"
    start_position => "beginning"
  }
}
filter {
}
output{
  elasticsearch{
    hosts => ["https://192.168.0.227:9200"]
    index => "myindex"
    user => "admin"
    password => "*****"
    cacert => "/logstash/logstash6.8/config/CloudSearchService.cer"
  }
}
```

NOTE

password: password for logging in to the cluster

4. Run the following command to import the ECS data collected by Logstash to the cluster:
`./bin/logstash -f logstash-simple.conf`
5. Log in to the CSS management console.
6. In the navigation pane on the left, choose **Clusters > Elasticsearch** to switch to the **Clusters** page.

7. From the cluster list, locate the row that contains the cluster to which you want to import data and click **Access Kibana** in the **Operation** column.
8. In the Kibana navigation pane on the left, choose **Dev Tools**.
9. On the **Console** page of Kibana, search for the imported data.

On the **Console** page of Kibana, run the following command to search for data. View the search results. If the searched data is consistent with the imported data, the data has been imported successfully.

```
GET myindex/_search
```

6.4 Using Kibana or APIs to Import Data to Elasticsearch

You can import data in various formats, such as JSON and CSV, to Elasticsearch in CSS by using Kibana or APIs.

Importing Data Using Kibana

Before importing data, ensure that you can use Kibana to access the cluster. The following procedure illustrates how to use the **POST** command to import data.

1. Log in to the CSS management console.
2. In the navigation pane on the left, choose **Clusters > Elasticsearch** to switch to the **Clusters** page.
3. Choose **Clusters** in the navigation pane. Locate the target cluster and click **Access Kibana** in the **Operation** column to log in to Kibana.
4. Click **Dev Tools** in the navigation tree on the left.
5. (Optional) On the **Console** page, run the related command to create an index for storing data and specify a custom mapping to define the data type.

If there is an available index in the cluster where you want to import data, skip this step. If there is no available index, create an index by referring to the following sample code.

For example, on the **Console** page of Kibana, run the following command to create an index named **my_store** and specify a user-defined mapping to define the data type:

Versions earlier than 7.x

```
PUT /my_store
{
  "settings": {
    "number_of_shards": 1
  },
  "mappings": {
    "products": {
      "properties": {
        "productName": {
          "type": "text"
        },
        "size": {
          "type": "keyword"
        }
      }
    }
  }
}
```

Versions later than 7.x

```
PUT /my_store
{
  "settings": {
    "number_of_shards": 1
  },
  "mappings": {
    "properties": {
      "productName": {
        "type": "text"
      },
      "size": {
        "type": "keyword"
      }
    }
  }
}
```

- Run commands to import data. For example, run the following command to import a piece of data:

Versions earlier than 7.x

```
POST /my_store/products/_bulk
{"index":{}}
{"productName":"Latest art shirts for women in 2017 autumn","size":"L"}
```

Versions later than 7.x

```
POST /my_store/_bulk
{"index":{}}
{"productName":"Latest art shirts for women in 2017 autumn","size":"L"}
```

The command output is similar to that shown in [Figure 6-7](#). If the value of the **errors** field in the result is **false**, the data is successfully imported.

Figure 6-7 Response message

```
1  {
2    "took": 42,
3    "errors": false,
4    "items": [
5      {
6        "index": {
7          "_index": "my_store",
8          "_type": "products",
9          "_id": "AWTGbHt7BwpN-hb3LKau",
10         "_version": 1,
11         "result": "created",
12         "_shards": {
13           "total": 2,
14           "successful": 2,
15           "failed": 0
16         },
17         "created": true,
18         "status": 201
19       }
20     ]
21   }
22 }
```

Importing Data Using APIs

You can call the bulk API using the cURL command to import a JSON data file.

NOTE

You are advised to import a file smaller than 50 MB.

1. Log in to the ECS that you use to access the cluster.
2. Run the following command to import JSON data:

In the command, replace the value of *{Private network address and port number of the node}* with the private network address and port number of a node in the cluster. If the node fails to work, the command will fail to be executed. If the cluster contains multiple nodes, you can replace the value of *{Private network address and port number of the node}* with the private network address and port number of any available node in the cluster. If the cluster contains only one node, restore the node and execute the command again. **test.json** indicates the JSON file whose data is to be imported.

```
curl -X PUT "http://{Private network address and port number of the node} /_bulk" -H 'Content-Type: application/json' --data-binary @test.json
```

NOTE

The value of the **-X** parameter is a command and that of the **-H** parameter is a message header. In the preceding command, **PUT** is the value of the **-X** parameter and **'Content-Type: application/json' --data-binary @test.json** is the value of the **-H** parameter. Do not add **-k** between a parameter and its value.

Example: In this example, assume that you need to import data in the **testdata.json** file to an Elasticsearch cluster, where communication encryption is disabled and the private network address and port number of one node are **192.168.0.90** and **9200** respectively. The data in the **testdata.json** file is as follows:

Versions earlier than 7.x

```
{"index": {"_index": "my_store", "_type": "products"}}
{"productName": "Autumn new woman blouses 2019", "size": "M"}
{"index": {"_index": "my_store", "_type": "products"}}
{"productName": "Autumn new woman blouses 2019", "size": "L"}
```

Versions later than 7.x

```
{"index": {"_index": "my_store"}}
{"productName": "Autumn new woman blouse 2019", "size": "M"}
{"index": {"_index": "my_store"}}
{"productName": "Autumn new woman blouse 2019", "size": "L"}
```

Perform the following steps to import the data:

- a. Run the following command to create an index named **my_store**:

Versions earlier than 7.x

```
curl -X PUT http://192.168.0.90:9200/my_store -H 'Content-Type: application/json' -d '{
  "settings": {
    "number_of_shards": 1
  },
  "mappings": {
    "products": {
      "properties": {
        "productName": {
          "type": "text"
        },
        "size": {
          "type": "keyword"
        }
      }
    }
  }
}
```



```
}  
}  
}  
}'
```

Versions later than 7.x

```
curl -X PUT http://192.168.0.90:9200/my_store -H 'Content-Type: application/json' -d '  
{  
  "settings": {  
    "number_of_shards": 1  
  },  
  "mappings": {  
    "properties": {  
      "productName": {  
        "type": "text"  
      },  
      "size": {  
        "type": "keyword"  
      }  
    }  
  }  
}'
```

- b. Run the following command to import the data in the **testdata.json** file:
curl -X PUT "http://192.168.0.90:9200/_bulk" -H 'Content-Type: application/json' --data-binary @testdata.json

7 Managing Elasticsearch Clusters

7.1 Cluster and Storage Capacity Statuses

On the **Dashboard** page of the CSS management console, you can view information about the status and storage capacity of existing clusters.

Table 7-1 Cluster status description

Status	Description
Available	The cluster is running properly and is providing services.
Abnormal	The cluster creation failed or the cluster is unavailable. If a cluster is in the unavailable status, you can delete the cluster or use snapshots created when the cluster is available to restore data to other clusters. However, operations such as expanding cluster capacity, accessing Kibana, creating snapshots, and restoring snapshots to the cluster are not allowed. When a cluster is in the unavailable status, data importing is not recommended to avoid data loss. You can view the cluster metrics or restart the cluster. However, the operations may fail. If the operations fail, contact technical support in a timely manner.
Processing	The cluster is being restarted, scaled, backed up, or recovered.
Creating	The cluster is being created.

Table 7-2 Cluster storage capacity status description

Status	Description
Normal	The storage capacity usage of all nodes in a cluster is less than 50%.
Warning	The storage capacity usage of any node in a cluster is from 50% to less than 80%.
Danger	The storage capacity usage of any node in a cluster is greater than or equal to 80%. You are advised to increase the storage space of the cluster to achieve normal data search or analysis.
Abnormal	The cluster storage capacity usage is unknown. For example, if the status of a cluster is Abnormal due to faults, the storage space status of the cluster will be Abnormal .

7.2 Introduction to the Cluster List

The cluster list displays all CSS clusters. If there are a large number of clusters, these clusters will be displayed on multiple pages. You can view clusters of all statuses from the cluster list.

Clusters are listed in chronological order by default in the cluster list, with the most recent cluster displayed at the top. [Table 7-3](#) shows the cluster parameters.



In the upper right corner of the cluster list, you can enter the name or ID of a cluster and click  to search for a cluster. You can also click  in the upper right corner to refresh the cluster list.

Table 7-3 Cluster list parameter description

Parameter	Description
Name/ID	Name and ID of a cluster. You can click a cluster name to switch to the Basic Information page. The cluster ID is automatically generated by the system and uniquely identifies a cluster.
Cluster Status	Status of a cluster. For details about the cluster status, see Cluster and Storage Capacity Statuses .
Task Status	Status of a task, such as cluster restart, cluster capacity expansion, cluster backup, and cluster restoration.
Version	Elasticsearch version of the cluster.
Created	Time when the cluster is created.
Enterprise Project	Enterprise project that a cluster belongs to.

Parameter	Description
Private Network Address	Private network address and port number of the cluster. You can use these parameters to access the cluster. If the cluster has multiple nodes, the private network addresses and port numbers of all nodes are displayed.
Billing Mode	Billing mode of a cluster.
Operation	Operations that can be performed on a cluster, including accessing Kibana, checking metrics, restarting a cluster, and deleting a cluster. If an operation is not allowed, the button is gray.

7.3 Index Backup and Restoration

You can back up index data in clusters. If data loss occurs or you want to retrieve data of a specified duration, you can restore the index data. Index backup is implemented by creating cluster snapshots. When creating a backup for the first time, you are advised to back up data of all indexes.

- **Managing Automatic Snapshot Creation:** Snapshots are automatically created at a specified time each day according to the rules you create. You can enable or disable the automatic snapshot creation function and set the automatic snapshot creation policy.
- **Manually creating a snapshot:** You can manually create a snapshot at any time to back up all data or data of specified indexes.
- **Restoring data:** You can use existing snapshots to restore the backup index data to a specified cluster.
- **Deleting a snapshot:** Delete snapshots you do not require and release resources.

 NOTE

- Before creating a snapshot, you need to perform basic configurations, including configuring the OBS bucket for storing snapshots, the snapshot backup path, and IAM agency used for security authentication.
- If there are available snapshots in the snapshot list when you configure the OBS bucket for storing cluster snapshots for the first time, you cannot change the bucket for snapshots that are subsequently created automatically or manually. Therefore, exercise caution when you configure the OBS bucket.
- If you want to change the OBS bucket where there are snapshots, do as follows: Disable the snapshot function, enable it, and specify a new OBS bucket. After you disable the snapshot function, you cannot use previously created snapshots to restore the cluster.
- If a cluster is in the **Unavailable** status, you can use the cluster snapshot function only to restore clusters and view existing snapshot information.
- During backup and restoration of a cluster, you can perform only certain operations, including scaling out, accessing Kibana, viewing metric, and deleting other snapshots of clusters. However, you cannot perform the following operations: restarting or deleting the cluster, deleting a snapshot that is in the **Creating** or **Restoring** status, and creating or restoring another snapshot. If a snapshot is being created or restored for a cluster, any automatic snapshot creation task initiated for the cluster will be canceled.
- The first snapshot of a cluster is a full snapshot, and subsequent snapshots are incremental snapshots. CSS snapshot files depend on each other.
- If you restore a CSS cluster snapshot to another cluster, indexes with the same name in the destination cluster will be overwritten. If the snapshot and the destination cluster use different shards, the indexes with the same name will not be overwritten.

Prerequisites

To use the function of creating or restoring snapshots, the account or IAM user logging in to the CSS management console must have both of the following permissions:

- **Tenant Administrator** for project **OBS** in region **Global service**
- **Elasticsearch Administrator** in the current region

Managing Automatic Snapshot Creation



1. In the CSS navigation pane on the left, click **Clusters**.
2. On the **Clusters** page that is displayed, click the name of the target cluster. In the navigation pane on the left, choose **Cluster Snapshots**.
3. On the displayed **Cluster Snapshots** page, click the icon to the right of **Cluster Snapshot** to enable the cluster snapshot function.
4. Enable the cluster snapshot function. OBS buckets and IAM agencies are automatically created to store snapshots. The automatically created OBS bucket and IAM agency are displayed on the page. You can also click  on the right of **Basic Configuration** to edit the configuration.

Table 7-4 Cluster snapshot parameter

Parameter	Description
OBS bucket	<p>Select an OBS bucket for storing snapshots from the drop-down list box. You can also click Create Bucket on the right to create an OBS bucket.</p> <p>The created or existing OBS bucket must meet the following requirements:</p> <ul style="list-style-type: none"> • Storage Class is Standard. • CSS does not support encrypted OBS buckets. When selecting an OBS bucket for snapshot storage, do not enable bucket encryption.
Backup Path	<p>Storage path of the snapshot in the OBS bucket.</p> <p>The backup path configuration rules are as follows:</p> <ul style="list-style-type: none"> • The backup path cannot contain the following characters: \:*?"<> • The backup path cannot start with a slash (/). • The backup path cannot start or end with a period (.). • The backup path cannot contain more than 1,023 characters.
IAM Agency	<p>IAM agency authorized by the current account to CSS access or maintain data stored in the OBS bucket. You can also click Create IAM Agency on the right to create an IAM agency.</p> <p>The created or existing IAM agency must meet the following requirements:</p> <ul style="list-style-type: none"> • Agency Type must be Cloud service. • Set Cloud Service to Elasticsearch or CSS. • The agency must have the Tenant Administrator permission for the OBS project in Global service.

5. Enable the automatic snapshot creation function. The **Configure Automatic Snapshot Creation** dialog box is displayed. If the automatic snapshot creation function is enabled, you can click  on the right of **Automatic Snapshot Creation** to modify the snapshot policy.
 - **Snapshot Name Prefix:** Enter a maximum of 32 characters starting with a lowercase letter. Only lowercase letters, digits, hyphens (-), and underscores (_) are allowed. A snapshot name consists of a snapshot name prefix and a timestamp, for example, **snapshot-2018022405925**.
 - **Time Zone:** indicates the time zone for the backup time. Specify **Backup Start Time** based on the time zone.
 - **Index:** Enter an index name. You can select an index for backup. Use commas (,) to separate multiple indexes. Uppercase letters, spaces, and the following special characters are not allowed: "\<|>/? If you do not specify this parameter, data of all indexes in the cluster is backed up by default. You can use the asterisk (*) to back up data of certain indices.

For example, if you enter **index***, then data of indices with the name prefix of **index** will be backed up.

Run the **GET /_cat/indices** command in Kibana to query the names of all indexes in the cluster.

- **Backup Start Time:** indicates the time when the backup starts automatically every day. You can specify this parameter only in hours and not minutes, for example, **00:00** or **01:00**. The value ranges from **00:00** to **23:00**. Select the backup time from the drop-down list box.
- **Retention Period (days):** indicates the duration when snapshots are retained in the OBS bucket, in days. The value ranges from **1** to **90**. You can specify this parameter as required. The system automatically deletes snapshots that are retained over the specified retention period on the half hour.

6. Click **OK** to save the snapshot policy.

Snapshots that are automatically created according to the snapshot policy are displayed in the snapshot list, along with manually created snapshots. You can distinguish them by the **Snapshot Type** setting. In the upper right corner of the snapshot list, enter the keyword of the snapshot name or snapshot ID to search for the desired snapshots.

7. (Optional) Disable the automatic snapshot creation function.

After you disable the automatic snapshot creation function, the system stops automatic creation of snapshots. If the system is creating a snapshot based on the automatic snapshot creation policy and the snapshot is not yet displayed in the snapshot list, you cannot disable the automatic snapshot creation function. In this case, if you click the button next to **Automatic Snapshot Creation**, a message is displayed, indicating that you cannot disable the function. You are advised to disable the function after the system completes automatic creation of the snapshot, and the created snapshot is displayed in the snapshot list.

When disabling the automatic snapshot creation function, you can choose whether to delete the snapshots that have been automatically created by selecting **Delete automated snapshots** in the displayed dialog box. By default, automatically created snapshots are not deleted.

- If you do not select **Delete automated snapshots**, automatically created snapshots are not deleted when you disable the automatic snapshot creation function. You can manually delete them later. For details, see [Deleting a Snapshot](#). If you do not manually delete the automatically created snapshots and enable the automatic snapshot creation function again, then all snapshots with **Snapshot Type** set to **Automated** in the snapshot list of the cluster can only be automatically deleted by the system. Specifically, the system automatically deletes snapshots based on the snapshot policy configured when you enable the automatic snapshot creation function again. For example, if you set **Retention Period (days)** to **10**, the system will automatically delete the snapshots that have been retained for more than 10 days.
- If you select **Delete automated snapshots**, all snapshots with **Snapshot Type** set to **Automated** in the snapshot list will be deleted when you disable the automatic snapshot creation function.

Manually Creating a Snapshot


1. In the CSS navigation pane on the left, click **Clusters**.
2. On the **Clusters** page that is displayed, click the name of the target cluster. In the navigation pane on the left, choose **Cluster Snapshots**.
3. On the displayed **Cluster Snapshots** page, click the icon to the right of **Cluster Snapshot** to enable the cluster snapshot function.
4. Enable the cluster snapshot function. OBS buckets and IAM agencies are automatically created to store snapshots. The automatically created OBS bucket and IAM agency are displayed on the page. You can also click  on the right of **Basic Configuration** to edit the configuration.

Table 7-5 Cluster snapshot parameter

Parameter	Description
OBS bucket	Select an OBS bucket for storing snapshots from the drop-down list box. You can also click Create Bucket on the right to create an OBS bucket. The created or existing OBS bucket must meet the following requirements: <ul style="list-style-type: none"> • Storage Class is Standard.
Backup Path	Storage path of the snapshot in the OBS bucket. The backup path configuration rules are as follows: <ul style="list-style-type: none"> • The backup path cannot contain the following characters: \:*?"<> • The backup path cannot start with a slash (/). • The backup path cannot start or end with a period (. • The backup path cannot contain more than 1,023 characters.
IAM Agency	IAM agency authorized by the current account to CSS access or maintain data stored in the OBS bucket. You can also click Create IAM Agency on the right to create an IAM agency. The created or existing IAM agency must meet the following requirements: <ul style="list-style-type: none"> • Agency Type must be Cloud service. • Set Cloud Service to Elasticsearch or CSS. • The agency must have the Tenant Administrator permission for the OBS project in Global service.

5. After basic configurations are completed, click **Create**.
 - **Name** indicates the name of the manually created snapshot, which can contain 4 to 64 characters and must start with a lowercase letter. Only lowercase letters, digits, hyphens (-), and underscores (_) are allowed. For snapshots you create manually, you can specify the snapshot name. The system will not automatically add the time information to the snapshot name.

- **Index:** Enter an index name. You can select an index for backup. Use commas (,) to separate multiple indexes. Uppercase letters, spaces, and the following special characters are not allowed: "\<|>/?. If you do not specify this parameter, data of all indexes in the cluster is backed up by default. You can use the asterisk (*) to back up data of certain indices. For example, if you enter **index***, then data of indices with the name prefix of **index** will be backed up.
Run the **GET /_cat/indices** command in Kibana to query the names of all indexes in the cluster.
 - **Description:** indicates the description of the created snapshot. The value contains 0 to 256 characters, and certain special characters (<>) are not allowed.
6. Click **OK**.
- After the snapshot is created, it will be displayed in the snapshot list. The status **Available** indicates that the snapshot is created successfully. All automatically and manually created snapshots are displayed in the snapshot list. You can distinguish them by the **Snapshot Type** setting. In the upper right corner of the snapshot list, enter the keyword of the snapshot name or snapshot ID to search for the desired snapshots.

Restoring Data

You can use snapshots whose **Snapshot Status** is **Available** to restore cluster data. The stored snapshot data can be restored to other clusters.

Restoring data will overwrite current data in clusters. Therefore, exercise caution when restoring data.

1. In the **Snapshots** area, locate the row that contains the snapshot you want to restore and click **Restore** in the **Operation** column.
2. On the **Restore** page, set restoration parameters.

Index: Enter the name of the index you want to restore. If you do not specify any index name, data of all indexes will be restored. The value can contain 0 to 1,024 characters. Uppercase letters, spaces, and certain special characters (including "\<|>/?.) are not allowed.

Rename Pattern: Enter a regular expression. Indexes that match the regular expression are restored. The default value **index_(.+)** indicates restoring data of all indices. The value contains 0 to 1,024 characters. Uppercase letters, spaces, and certain special characters (including "\<|>/?.) are not allowed.

Rename Replacement: Enter the index renaming rule. The default value **restored_index_\$1** indicates that **restored_** is added in front of the names of all restored indexes. The value can contain 0 to 1,024 characters. Uppercase letters, spaces, and certain special characters (including "\<|>/?.) are not allowed. You can set **Rename Replacement** only if you have specified **Rename Pattern**.

Cluster: Select the cluster that you want to restore. You can select the current cluster or others. However, you can only restore the snapshot to clusters whose status is **Available**. If the status of the current cluster is **Unavailable**, you cannot restore the snapshot to the current cluster. If you choose to restore the snapshot to another cluster, ensure that the target cluster runs an Elasticsearch version that is not earlier than that of the current cluster. If you

select another cluster and two or more indexes in the cluster have the same name, data of all indices with the same name as the name you specify will be overwritten. Therefore, exercise caution when you set the parameters.

3. Click **OK**. If restoration succeeds, **Task Status** of the snapshot in the snapshot list will change to **Restoration succeeded**, and the index data is generated again according to the snapshot information.

Deleting a Snapshot

If you no longer need a snapshot, delete it to release storage resources. If the automatic snapshot creation function is enabled, snapshots that are automatically created cannot be deleted manually, and the system automatically deletes these snapshots on the half hour after the time specified by **Retention Period (days)**. If you disable the automatic snapshot creation function while retaining the automated snapshots, then you can manually delete them later. If you do not manually delete the automatically created snapshots and enable the automatic snapshot creation function again, then all snapshots with **Snapshot Type** set to **Automated** in the snapshot list of the cluster can only be automatically deleted by the system.

NOTE

After a snapshot is deleted, its data cannot be restored. Exercise caution when deleting a snapshot.

1. In the snapshot list, locate the snapshot that you want to delete.
2. Click **Delete** in the **Operation** column. In the dialog box that is displayed, confirm the snapshot information and click **OK**.

7.4 Binding an Enterprise Project

You can create enterprise projects based on your organizational structure. Then you can manage resources across different regions by enterprise project, add users and user groups to enterprise projects, and grant different permissions to the users and user groups. This section describes how to bind a CSS cluster to an enterprise project and how to modify an enterprise project.

Prerequisites

Before binding an enterprise project, you have created an enterprise project.

Binding an Enterprise Project

When creating a cluster, you can bind an existing enterprise project to the cluster, or click **View Enterprise Project** to go to the enterprise project management console and create a new project or view existing projects.

Modifying an Enterprise Project

For a cluster that has been created, you can modify its enterprise project based on the site requirements.

1. Log in to the CSS management console.
2. In the navigation tree on the left, choose **Clusters > Elasticsearch**.
3. In the cluster list on the displayed page, click the target cluster name to switch to the **Cluster Information** page.
4. On the **Cluster Information** page, click the enterprise project name on the right of **Enterprise Project**. The project management page is displayed.

Figure 7-1 Enterprise Project

AZ	eu-0a
Subnet	subnet-4adf-Ernest_BIGDATA_WORKSPACE ...
Enterprise Project	nhe-demo
Reset Password	Reset
Access Control	Disabled Set

5. On the **Resources** tab page, select the region of the current cluster, and select **CSS** for **Service**. In this case, the corresponding CSS cluster is displayed in the resource list.
6. Select the cluster whose enterprise project you want to modify and click **Remove**.
7. On the **Remove Resource** page, specify **Mode** and select **Destination Enterprise Project**, and click **OK**.
8. After the resource is removed, you can view the modified enterprise project information on the **Clusters** page.

7.5 Restarting a Cluster

If a cluster becomes faulty, you can restart it to check if it can run normally.

Prerequisites

- The target cluster is not frozen and has no task in progress.
- If a cluster is available, ensure that it has stopped processing service requests (such as importing data and searching for data). Otherwise, data may be lost when the cluster is restarted. You are advised to perform this operation during off-peak hours.

Context

CSS supports quick restart and rolling restart.

Quick Restart

- All clusters support this function.
- If you select a node type for quick restart, all nodes of the selected type will be restarted together.

- If you select a node name for quick restart, only the specified node will be restarted.
- The cluster is unavailable during quick restart.

Rolling Restart

- Rolling restart is supported only when a cluster has at least three nodes (including master nodes, client nodes, and cold data nodes).
- Rolling restart can be performed only by specifying node types. If you select a node type for rolling restart, the nodes of the selected type will be restarted in sequence.
- During the rolling restart, only the nodes that are being restarted are unavailable and other nodes can run normally.
- When the data volume is large, rolling restart will take a long time.

Quick Restart

1. Log in to the CSS management console.
2. In the navigation pane on the left, choose **Clusters > Elasticsearch**. On the displayed **Clusters** page, locate the target cluster and choose **More > Restart** in the **Operation** column.
3. On the **Restart Cluster** page, select **Quick Restart**.
You can quick restart nodes by **Node type** or **Node name**. If you select **Node type**, then you can select multiple node types and perform quick restart at the time. If you select **Node name**, you can perform quick restart only on one node at a time.
4. Refresh the page and check the cluster status. During the restart, the cluster status is **Processing**, and the task status is **Restarting**. If the cluster status changes to **Available**, the cluster has been restarted successfully.

Rolling Restart

1. Log in to the CSS management console.
2. In the navigation pane on the left, choose **Clusters > Elasticsearch**. On the displayed **Clusters** page, locate the target cluster and choose **More > Restart** in the **Operation** column.
3. On the **Restart Cluster** page, select **Rolling Restart**.
You can perform rolling restart by **Node type**. Select specific node types for restart.
4. Refresh the page and check the cluster status. During the restart, the cluster status is **Processing**, and the task status is **Restarting**. If the cluster status changes to **Available**, the cluster has been restarted successfully.

7.6 Migrating Cluster Data

You can migrate data from one cluster to another. In certain scenarios, for example, if you cannot get sufficient capacity by changing the specifications of the current cluster, you can create a cluster of higher specifications and migrate all data of the current cluster to the new one. Alternatively, you can merge indexes in two clusters to one cluster to meet your business needs. CSS enables you to

migrate cluster data by using the index backup and restoration function, specifically, by restoring the snapshot of a cluster to the target cluster.

In this section, assume that we need to migrate the data of cluster **Es-1** to cluster **Es-2**. Cluster **Es-2** runs a version later than that of cluster **Es-1** and the number of nodes in cluster **Es-2** is greater than half of that in cluster **Es-1**.

Prerequisites

- The source and target clusters are in the same region.
- The version of the target cluster is the same as or later than that of the source cluster.
- The number of nodes in the target cluster must be greater than half of the number of nodes in the source cluster.

Suggestions

- The number of nodes in the target cluster should be no less than the number of replicas in the source cluster.
- The CPU, memory, and disk configurations of the target cluster should be no less than those of the source cluster. This will minimize service loss after migration.

Migration Duration

The number of nodes or index shards in the source and destination clusters determines how long the data migration will take. Data migration consists of two phases: data backup and restoration. The backup duration is determined by the source cluster and the restoration duration is determined by the destination cluster. The formula for calculating the total migration duration is as follows:

- If the number of index shards is greater than the number of nodes:

Total duration (s) = (800 GB/40 MB/Number of source cluster nodes + 800 GB/40 MB/Number of destination cluster nodes) x Number of indexes

- If the number of index shards is smaller than the number of nodes:

Total duration (s) = (800 GB/40 MB/Number of shards of the source cluster index + 800 GB/40 MB/Number of shards of the destination cluster index) x Number of indexes

NOTE

The migration duration estimated using the formula is the minimal duration possible (if each node transmits data at the fastest speed, 40 MB/s). The actual duration also depends on factors such as the network and resources condition.

Procedure

1. On the **Clusters** page, click **Es-1**.
2. In the navigation pane on the left, choose **Cluster Snapshots**. Enable the cluster snapshot and set basic configuration. For details, see [Manually Creating a Snapshot](#).
3. Click **Create** to manually create a snapshot. In the displayed dialog box, enter the snapshot name and click **OK**.
4. In the snapshot list, locate the target snapshot and click **Restore** in the **Operation** column to restore data to cluster **Es-2**.

- In the text box next to **Index**, enter *, indicating that you want to restore data of all indexes in cluster **Es-1**.
 - From the **Cluster** drop-down list, select **Es-2**.
- Click **OK**.
5. After restoration is complete, data in cluster **Es-1** will be migrated to cluster **Es-2**.

7.7 Deleting a Cluster

You can delete clusters that you no longer need.

NOTE

- If you delete a cluster, the cluster service data will be cleared. Exercise caution when performing this operation.
- The snapshots of a cluster stored in OBS are not deleted with the cluster.

Procedure

1. Log in to the CSS management console.
2. In the navigation tree on the left, choose **Clusters > Elasticsearch**.
3. Locate the target cluster and click **More > Delete** in the **Operation** column.
4. In the displayed dialog box, enter the name of the cluster to be deleted and click **OK**.

7.8 Managing Tags

Tags are cluster identifiers. Adding tags to clusters can help you identify and manage your cluster resources.

You can add tags to a cluster when creating the cluster or add them on the details page of the created cluster.

Managing Tags of a New Cluster

1. Log in to the CSS management console.
2. Click **Create Cluster** in the upper right corner. The **Create Cluster** page is displayed.
3. On the **Create Cluster** page, set **Advanced Settings** to **Custom**. Add tags for a cluster.

You can select a predefined tag and set **Tag value** for the tag. You can click **View Predefined Tag** to switch to the TMS management console and view existing tags.

You can also create new tags by specifying **Tag key** and **Tag value**.

You can add a maximum of 10 tags for a CSS cluster. If the entered tag is incorrect, you can click **Delete** on the right of the tag to delete the tag.

Table 7-6 Naming rules for a tag key and value

Parameter	Description
Tag key	<p>Must be unique in a cluster.</p> <p>Can contain a maximum of 36 characters.</p> <p>Can only consist of digits, letters, hyphens (-), and underscores (_).</p>
Tag value	<p>Can contain a maximum of 43 characters.</p> <p>Can only consist of digits, letters, hyphens (-), and underscores (_).</p> <p>Cannot be left blank.</p>

Managing Tags of Existing Clusters

You can modify, delete, or add tags for a cluster.

1. Log in to the CSS management console.
2. On the **Clusters** page, click the name of a cluster for which you want to manage tags.

The **Basic Information** page is displayed.

3. In the navigation pane on the left, choose the **Tags** tab. You can add, modify, or delete tags.
 - View

On the **Tags** page, you can view details about tags of the cluster, including the number of tags and the key and value of each tag.
 - Add

Click **Add** in the upper left corner. In the displayed **Add Tag** dialog box, enter the key and value of the tag to be added, and click **OK**.
 - Modify

You can only change the value of an existing tag.

In the **Operation** column of a tag, click **Edit**. In the displayed **Edit Tag** page, enter a new tag value and click **OK**.
 - Delete

In the **Operation** column of the tag, click **Delete**. After confirmation, click **Yes** on the displayed **Delete Tag** page.

Searching for Clusters by Tag

1. Log in to the CSS management console.
2. On the **Clusters** page, click **Search by Tag** in the upper right corner of the cluster list.
3. Select or enter the tag key and tag value you want to search for, and click **Add** to add the tag to the search text box.

You can select a tag key or tag value from their drop-down lists. The system returns a list of clusters that exactly match the tag key or tag value. If you

enter multiple tags, the cluster that meets requirements of all the tags will be filtered.

You can add a maximum of 10 tags at one time.

4. Click **Search**.

The system searches for the target cluster by tag key and value.

7.9 Public Network Access

You can access a security cluster (clusters in version 6.5.4 or later support the security mode) that has the HTTPS access enabled through the public IP address provided by the system.

 **NOTE**

If public network access is enabled for CSS, then EIP and bandwidth resources will be used and billed.

Configuring Public Network Access

1. Log in to the CSS management console.
2. On the **Create Cluster** page, enable **Security Mode**. Set the administrator password and enable HTTPS access.
3. Select **Automatically assign** for **Public IP Address** and set related parameters.

Figure 7-2 Configuring public network access

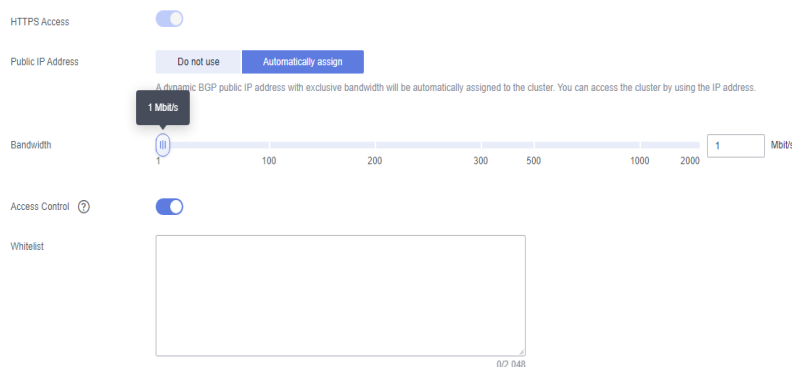


Table 7-7 Public network access parameters

Parameter	Description
Bandwidth	Bandwidth for accessing Kibana with the public IP address
Access Control	If you disable this function, all IP addresses can access the cluster through the public IP address. If you enable access control, only IP addresses in the whitelist can access the cluster through the public IP address.

Parameter	Description
Whitelist	IP address or IP address range allowed to access a cluster. Use commas (,) to separate multiple addresses. This parameter can be configured only when Access Control is enabled.

Managing Public Network Access

You can configure, modify, view the public network access of, or disassociate the public IP address from a cluster.

1. Log in to the CSS management console.
2. On the **Clusters** page, click the name of the target cluster. On the **Basic Information** page that is displayed, manage the public network access configurations.

Figure 7-3 Modifying public network access configurations

Region	eu-west-1	AZ	eu-west-1
VPC	nhe-vp-1	Subnet	subn-1
Security Group	nhe-sc-1	Enterprise Project	nhe-demo
Security Mode	Enabled Download Certificate	Reset Password	Reset
Public IP Address	90.61.100.15:9200 Disassociate	Access Control	Disabled Set
Bandwidth	101 Mbit/s Edit		
HTTPS Access	Enabled		
Private Network Address	192.168.0.110:9200, 192.168.0.100:9200, 192.168.0.245:9200		

- **Configuring public network access**
If you did not configure the public network access during cluster creation, you can configure it on the **Basic Information** page after configuring the cluster.
Click **Associate** next to **Public IP Address**, set the access bandwidth, and click **OK**.
If the association fails, wait for several minutes and try again.
- **Modifying public network access**
For a cluster for which you have configured public network access, you can click **Edit** next to **Bandwidth** to modify the bandwidth, or you can click **Set** next to **Access Control** to set the access control function and the whitelist for access.
- **Viewing public network access**
On the **Basic Information** page, you can view the public IP address associated with the current cluster.
- **Disassociating a public IP address from a cluster**
To disassociate the public IP address, click **Disassociate** next to **Public IP Address**.

Accessing a Cluster Through the Public IP Address

After configuring the public IP address, you can use it to access the cluster.

For example, run the following cURL commands to view the index information in the cluster. In this example, the public access IP address of one node in the cluster is **10.62.179.32** and the port number is **9200**.

- If the cluster you access does not have the security mode enabled, run the following command:

```
curl 'http://10.62.179.32:9200/_cat/indices'
```
- If the cluster you access has the security mode enabled, access the cluster using HTTPS and add the username, password and **-u** to the cURL command.

```
curl -u username:password -k 'https://10.62.179.32:9200/_cat/indices'
```

7.10 Managing Logs

CSS provides log backup and search functions to help you locate faults. You can back up cluster logs to OBS buckets and download required log files to analyze and locate faults.

Enabling Log Management

1. Log in to the CSS management console.
2. Choose **Clusters** in the navigation pane. On the **Clusters** page, click the name of the target cluster. The cluster information page is displayed.
3. Click the **Logs** tab and toggle on the **Log Management** switch.
4. In the **Edit Log Backup Configuration** dialog box, set the parameters.
In the displayed dialog box, **OBS Bucket**, **Backup Path**, and **IAM Agency** are automatically created for log backup. You can change the default value by referring to [Table 7-8](#).


If the **Log Management** function has been enabled for the cluster, you can click  on the right of **Log Backup Configuration** and modify the configuration in the displayed **Edit Log Backup Configuration** dialog box. For details, see [Table 7-8](#).

Table 7-8 Parameters for configuring log backup


Parameter	Description	Remarks
OBS Bucket	Select an OBS bucket from the drop-down list for storing logs. You can also click Create Bucket on the right to create an OBS bucket.	The OBS bucket and the cluster must be in the same region. NOTE <ul style="list-style-type: none"> • CSS does not support encrypted OBS buckets. When selecting an OBS bucket for log storage, do not enable bucket encryption. • To let an IAM user access an OBS bucket, you need to grant the GetBucketStoragePolicy, GetBucketLocation, ListBucket, and ListAllMyBuckets permissions to the user.
Backup Path	Storage path of logs in the OBS bucket	The backup path configuration rules are as follows: <ul style="list-style-type: none"> • The backup path cannot contain the following characters: \:*?"<> • The backup path cannot start with a slash (/). • The backup path cannot start or end with a period (.) • The total length of the backup path cannot exceed 1,023 characters.
IAM Agency	IAM agency authorized by the current account for CSS to access or maintain data stored in the OBS bucket. You can also click Create IAM Agency on the right to create an IAM agency.	The IAM agency must meet the following requirements: <ul style="list-style-type: none"> • Agency Type must be Cloud service. • Set Cloud Service to Elasticsearch or CSS. • Mandatory policies: Tenant Administrator

5. Back up logs.

- Automatically backing up logs

Click the icon on the right of **Auto Backup** to enable the auto backup function.

After the automatic backup function is enabled, set the backup start time in the **Configure Auto Backup** dialog box. When the scheduled time arrives, the system will back up logs automatically.

After the **Automatic Snapshot Creation** function is enabled, you can click  on the right of the parameter to change the backup start time.

- Manually backing up logs

On the **Log Backup** tab page, click **Back Up**. On the displayed page, click **Yes** to start backup.


If **Task Status** in the log backup list is **Successful**, the backup is successful.

 **NOTE**

All logs in the cluster are copied to a specified OBS path. You can view or download log files from the path of the OBS bucket.

6. Search for logs.

On the **Log Search** page, select the target node, log type, and log level, and

click  . The search results are displayed.

When you search for logs, the latest 10,000 logs are matched. A maximum of 100 logs are displayed.

Viewing Logs

After backing up logs, you can click **Backup Path** to go to the OBS console and view the logs.

Backed up logs mainly include deprecation logs, run logs, index slow logs, and search slow logs. [Table 7-9](#) lists the storage types of the OBS bucket.

Table 7-9 Log types

Log Name	Description
clustername_deprecation.log	Deprecation log
clustername_index_indexing_slowlog.log	Search slow log
clustername_index_search_slowlog.log	Index slow log
clustername.log	Elasticsearch run log
clustername_access.log	Access log
clustername_audit.log	Audit log

7.11 Managing Plugins

CSS clusters have default plugins. You can view the default plugin information on the console or Kibana.

Viewing Plugins on the Console

1. Log in to the CSS management console.

2. In the navigation pane, choose **Clusters**. Click the target cluster name and go to the **Basic Information** page of the cluster.
3. Click the **Plugins** tab.
4. On the **Default** page, view default plugins supported by the current version.

Viewing Plugins on the Kibana

1. Log in to the CSS management console.
2. In the navigation pane, choose **Clusters**. Locate the target cluster and click **Access Kibana** in the **Operation** column to log in to Kibana.
3. Go to **Dev Tools** and run the following command to view the cluster plugin information:

```
GET _cat/plugins?v
```

The following is an example of the response body:

name	component	version
css-test-ess-esn-1-1	analysis-dynamic-synonym	7.6.2-xx-ei-css-v1.0.1
css-test-ess-esn-1-1	analysis-icu	7.6.2-xx-ei-css-v1.1.6
css-test-ess-esn-1-1	analysis-ik	7.6.2-xx-ei-css-v1.0.1
.....		

name indicates the cluster node name, **component** indicates the plugin name, and **version** indicates the plugin version.

7.12 Hot and Cold Data Storage

CSS provides you with cold data nodes. You can store data that requires query response in seconds on high-performance nodes and store data that requires query response in minutes on cold data nodes with large capacity and low specifications.

NOTE

- Data nodes must be configured when you create a cluster. When you set a cold data node, the rest of the data nodes become hot nodes.
- You can enable the cold data node, master node, and client node functions at the same time.
- You can increase nodes and expand storage capacity of cold data nodes. The maximum storage capacity is determined by the node specifications. Local disks do not support storage capacity expansion.

Hot and Cold Data Node Switchover

After you enable the cold data node function, the cold data node is labeled with **cold**. In addition, data nodes are labeled with **hot** and become hot nodes. You can specify indices to distribute data to cold or hot nodes.

You can configure a template to store indices on the specified cold or hot node.

The following figure shows this process. Log in to the **Kibana Console** page of the cluster, modify the template by configuring the index starting with **myindex**, and store the indices on the cold node. In this case, the **myindex*** data is stored on the cold data node by modifying the template.

Run the following command to create a template:

```
PUT _template/test
{
  "order": 1,
  "index_patterns": "myindex*",
  "settings": {
    "refresh_interval": "30s",
    "number_of_shards": "3",
    "number_of_replicas": "1",
    "routing.allocation.require.box_type": "cold"
  }
}
```

You can perform operations on the created index.

```
PUT myindex/_settings
{
  "index.routing.allocation.require.box_type": "cold"
}
```

You can cancel the configurations of hot and cold data nodes.

```
PUT myindex/_settings
{
  "index.routing.allocation.require.box_type": null
}
```

7.13 Configuring Parameters

You can modify the `elasticsearch.yml` file.

Modifying Parameter Configurations

1. Log in to the CSS management console.
2. Choose **Clusters** in the navigation pane. On the **Clusters** page, click the name of the target cluster. The cluster information page is displayed.
3. Click **Parameter Configurations** and click **Edit** to modify module parameters as required.

Table 7-10 Module parameters

Module Name	Parameter	Description
Cross-domain Access	http.cors.allow-credentials	Whether to return the Access-Control-Allow-Credentials of the header during cross-domain access Value: true or false Default value: false
	http.cors.allow-origin	Origin IP address allowed for cross-domain access, for example, 122.122.122.122:9200
	http.cors.max-age	Cache duration of the browser. The cache is automatically cleared after the time range you specify. Unit: s Default value: 1,728,000

Module Name	Parameter	Description
	http.cors.allow-headers	Headers allowed for cross-domain access, including X-Requested-With , Content-Type , and Content-Length . Use commas (,) and spaces to separate headers.
	http.cors.enabled	Whether to allow cross-domain access Value: true or false Default value: false
	http.cors.allow-methods	Methods allowed for cross-domain access, including OPTIONS , HEAD , GET , POST , PUT , and DELETE . Use commas (,) and spaces to separate methods.
Reindexing	reindex.remote.whitelist	Configured for migrating data from the current cluster to the target cluster through the reindex API. The example value is 122.122.122.122:9200.
Custom Cache	indices.queries.cache.size	Cache size in the query phase Value range: 1 to 100 Unit: % Default value: 10%
Queue Size in a Thread Pool	thread_pool.bulk.queue_size	Queue size in the bulk thread pool. The value is an integer. Default value: 200
	thread_pool.write.queue_size	Queue size in the write thread pool. The value is an integer. Default value: 200
	thread_pool.force_merge.size	Queue size in the force merge thread pool. The value is an integer. Default value: 1
Customize	You can add parameters based on your needs.	Customized parameters NOTE <ul style="list-style-type: none"> Enter multiple values in the format as [value1, value2, value3...]. Separate values by commas (,) and spaces. Colons (:) are not allowed.

- After the modification is complete, click **Submit**. In the displayed **Submit Configuration** dialog box, select the box indicating "I understand that the modification will take effect after the cluster is restarted." and click **Yes**.

If the **Status** is **Succeeded** in the parameter modification list, the modification has been saved. Up to 20 modification records can be displayed.

5. Return to the cluster list and choose **More > Restart** in the **Operation** column to restart the cluster and make the modification take effect.
 - You need to restart the cluster after modification, or **Configuration unupdated** will be displayed in the **Task Status** column on the **Clusters** page.
 - If you restart the cluster after the modification, and **Task Status** displays **Configuration error**, the parameter configuration file fails to be modified.

7.14 VPC Endpoint Service

If the VPC endpoint service is enabled, you can use a private domain name or node IP address generated by the endpoint to access the cluster. When the VPC endpoint service is enabled, a VPC endpoint will be created by default. You can select **Private Domain Name Creation** as required.

CAUTION

The public IP address access and VPC endpoint service share a load balancer. If you have configured a public access whitelist, public and private IP addresses that access the cluster through VPCEP are restricted because the public IP address access shares the load balancer with the VPC endpoint service. In this case, you need to add IP address **198.19.128.0/17** to the public access whitelist to allow traffic through VPCEP.

Enabling the VPC Endpoint Service

1. Log in to the CSS management console.
2. Click **Create Cluster** in the upper right corner.
3. On the **Create Cluster** page, set **Advanced Settings** to **Custom**. Enable the VPC endpoint service.
 - **Private Domain Name Creation:** If you enable this function, the system automatically creates a private domain name for you, which you can use to access the cluster.
 - **VPC Endpoint Service Whitelist:** You can add an authorized account ID to the VPC endpoint service whitelist. Then you can access the cluster using the domain name or the node IP address.
 - You can click **Add** to add multiple accounts.
 - Click **Delete** in the **Operation** column to delete the accounts that are not allowed to access the cluster.

NOTE

- If the authorized account ID is set to *, all users are allowed to access the cluster.
- You can view authorized account IDs on the **My Credentials** page.

Managing VPC Endpoint Service

You can enable the VPC endpoint service while creating a cluster, and also enable it by performing the following steps after cluster creation.

1. Log in to the CSS management console.
2. Choose **Clusters** in the navigation pane. On the **Clusters** page, click the name of the target cluster.
3. Click the **VPC Endpoint Service** tab, and turn on the button next to **VPC Endpoint Service**.

In the displayed dialog box, you can determine whether to enable the private domain name. Click **Yes** to enable the VPC endpoint service.

NOTE

- If the VPC endpoint service is enabled, you can use a private domain name or node IP address generated by the VPC endpoint to access the cluster. For details, see [Accessing the Cluster Using the Private Domain Name or Node IP Address](#).
 - If you disable the VPC endpoint service, none of the users can access the cluster using the private domain name.
4. (Optional) Click **Modify** next to **VPC Endpoint Service Whitelist** to update the existing whitelist.
 5. Manage VPC endpoints.

The **VPC Endpoint Service** page displays all VPC endpoints connected to the current VPC endpoint service.

Figure 7-4 Managing VPC endpoints



VPC Endpoint ID	Status	Max. Connections	Owner	Created	Operation
2522a05c-8076-437f-aeca-888...	Accepted	3000	c4b2d06aa90b4b7ba8439b28be65410f	Jul 08, 2022 15:15:14 GMT+08:00	Accept Reject Modification

- Click **Accept** or **Reject** in the **Operation** column to change the node status. If you reject the connection with a VPC endpoint, you cannot access the cluster through the private domain name generated by that VPC endpoint.
- Click **Modify** in the **Operation** column to change the maximum number of connections of the current node.

Accessing the Cluster Using the Private Domain Name or Node IP Address

1. Obtain the private domain name or node IP address.
Log in to the CSS console, click the target cluster name and go to the **Cluster Information** page. Click the **VPC Endpoint Service** tab and view the private domain name.
2. Run the cURL command to execute the API or call the API by using a program before accessing the cluster. For details about Elasticsearch operations and APIs, see the [Elasticsearch Reference](#).

The ECS must meet the following requirements:

- Sufficient disk space is allocated for the ECS.
- The ECS and the cluster must be in the same VPC. After enabling the VPC endpoint service, you can access the cluster from the ECS even when the cluster is not in the same VPC as the ECS.
- The security group of the ECS must be the same as that of the cluster.
If this requirement is not met, modify the ECS security group or configure the inbound and outbound rules of the ECS security group to allow the ECS security group to be accessed by all security groups of the cluster. For details, see [Configuring Security Group Rules](#).
- Configure security group rule settings of the target CSS cluster. Set **Protocol** to **TCP** and **Port Range** to **9200** or a port range including port **9200** for both the outbound and inbound directions.

For example, run the following cURL command to view the index information in the cluster. In this example, the private network address is **vpcep-7439f7f6-2c66-47d4-b5f3-790db4204b8d.region01.huaweicloud.com** and port **9200** is used to access the cluster.

For example, run the following cURL command to view the index information in the cluster. In this example, the private network address is **vpcep-7439f7f6-2c66-47d4-b5f3-790db4204b8d.region01.xxxx.com** and port **9200** is used to access the cluster.

- If the cluster you access does not have the security mode enabled, run the following command:

```
curl 'http://vpcep-7439f7f6-2c66-47d4-b5f3-790db4204b8d.region01.xxxx.com:9200/_cat/indices'
```

- If the cluster you access has the security mode enabled, access the cluster using HTTPS and add the username, password and **-u** to the cURL command.

```
curl -u username:password -k 'https://vpcep-7439f7f6-2c66-47d4-b5f3-790db4204b8d.region01.xxxx.com:9200/_cat/indices'
```

7.15 Kibana Public Access

For CSS clusters that have security mode enabled, you can enable Kibana public access. After the configuration is complete, an IP address will be provided to access Kibana of this cluster over the Internet.

You can configure Kibana public access during cluster creation, or after a cluster in security mode is created.

NOTE

- You can enable **Security Mode** for clusters of version 6.5.4 and later versions.
- Kibana public access cannot be configured for clusters created in security mode before this function was rolled out (before June 2020).
- The whitelist for Kibana public network access depends on the ELB whitelist. After you updated the whitelist, the new settings take effect immediately for new connections. For existing persistent connections using the IP addresses that have been removed from the whitelist, the new settings take effect about 1 minute after these connections are stopped.

Configuring Kibana Public Access When Creating a Cluster

1. Log in to the CSS management console.
2. Click **Create Cluster** in the upper right corner. The **Create Cluster** page is displayed.
3. On the **Create Cluster** page, enable **Security Mode**.
4. Set **Advanced Settings** to **Custom**, enable **Kibana Public Access**, and set parameters.

Table 7-11 Kibana public access parameters

Parameter	Description
Bandwidth	Bandwidth for accessing Kibana with the public IP address Value range: 1 to 100 Unit: Mbit/s
Access Control	If you disable this function, all IP addresses can access Kibana through the public IP address. If you enable this function, only IP addresses or IP address in the whitelist can access Kibana through the public IP address.
Whitelist	IP address or IP address range allowed to access a cluster. Use commas (,) to separate multiple addresses. This parameter can be configured only when Access Control is enabled. You are advised to enable this function.

After the cluster is created, click the cluster name to go to the **Basic Information** page. On the **Kibana Public Access** page, you can view the Kibana public IP address.

Configuring Kibana Public Access for an Existing Cluster

You can enable, disable, modify, and view Kibana public access for an existing cluster that has security mode enabled.

1. Log in to the CSS management console.
2. Choose **Clusters** in the navigation pane. On the **Clusters** page, click the name of the target cluster.
3. Click the **Kibana Public Access** tab. Turn on the **Kibana Public Access** switch to enable the Kibana public access function.
4. On the displayed page, set parameters.

Table 7-12 Kibana public access parameters

Parameter	Description
Bandwidth	Bandwidth for accessing Kibana with the public IP address Value range: 1 to 100 Unit: Mbit/s
Access Control	If you disable this function, all IP addresses can access Kibana through the public IP address. If you enable this function, only IP addresses or IP address in the whitelist can access Kibana through the public IP address.
Whitelist	IP address or IP address range allowed to access a cluster. Use commas (,) to separate multiple addresses. This parameter can be configured only when Access Control is enabled. You are advised to enable this function.

5. After you set the parameters, click **OK**.

Modifying Kibana Public Access

For clusters configured Kibana public access, you can modify its bandwidth and access control or disable this function.

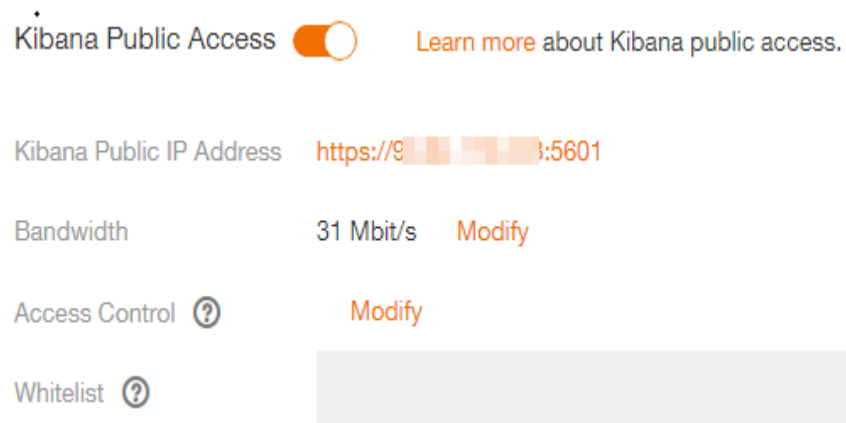
1. Log in to the CSS management console.
2. Choose **Clusters** in the navigation pane. On the **Clusters** page, click the name of the target cluster.
3. Click the **Kibana Public Access** tab to modify the Kibana public access configuration.
 - Modifying bandwidth
Click **Modify** on the right of **Bandwidth**. On the **Modify Bandwidth** page, modify the bandwidth and click **OK**.
 - Modifying access control
Click **Modify** on the right of **Access Control**. On the **Modify Access Control** page, set **Access Control** and **Whitelist**, and click **OK**.
 - Disabling Kibana public access
Toggle off the **Kibana Public Access** switch.

Accessing Kibana with the Public IP Address

After configuring Kibana public access, you will obtain a public IP address that you can use to access Kibana of this cluster.

1. Log in to the CSS management console.
2. Choose **Clusters** in the navigation pane. On the **Clusters** page, click the name of the target cluster.
3. Click the **Kibana Public Access** tab to obtain the Kibana public IP address.

Figure 7-5 Obtaining the Kibana public IP address



4. Use this IP address to access Kibana of this cluster through the Internet.

8 Vector Retrieval

8.1 Description

Large-scale image recognition and retrieval, video search, and personalized recommendation impose high requirements on the latency and accuracy of high-dimensional space vector retrieval. To facilitate large-scale vector search, CSS integrates the vector search feature powered by vector search engine and the Elasticsearch plug-in mechanism.

Principles

Vector search works in a way similar to traditional search. To improve vector search performance, we need to:

- **Narrow down the matched scope**
Similar to traditional text search, vector search use indexes to accelerate the search instead of going through all data. Traditional text search uses inverted indexes to filter out irrelevant documents, whereas vector search creates indexes for vectors to bypass irrelevant vectors, narrowing down the search scope.
- **Reduce the complexity of calculating a single vector**
The vector search method can quantize and approximate high dimensional vectors first. By doing this, you can acquire a smaller and more relevant data set. Then more sophisticated algorithms are applied to this smaller data set to perform computation and sorting. This way, complex computation is performed on only part of the vectors, and efficiency is improved.

Vector search means to retrieve the k-nearest neighbors (KNN) to the query vector in a given vector data set by using a specific measurement method. Generally, we only focus on Approximate Nearest Neighbor (ANN), because a KNN search requires excessive computational resources.

Functions

Our engine integrates a variety of vector indexes, such as brute-force search, Hierarchical Navigable Small World (HNSW) graphs, product quantization, and

IVF-HNSW. It also supports multiple similarity calculation methods, such as Euclidean, inner product, cosine, and Hamming. The recall rate and retrieval performance of the engine are better than those of open-source engines. It can meet the requirements for high performance, high precision, low costs, and multi-modal computation.

The search engine also supports all the capabilities of the native Elasticsearch, including distribution, multi-replica, error recovery, snapshot, and permission control. The engine is compatible with the native Elasticsearch ecosystem, including the cluster monitoring tool Cerebro, the visualization tool Kibana, and the real-time data ingestion tool Logstash. Several client languages, such as Python, Java, Go, and C++, are supported.

Constraints

- Only clusters of version 7.6.2 and 7.10.2 support vector search.
- The vector search plug-in performs in-memory computing and requires more memory than common indexes do. You are advised to use memory-optimized computing specifications.

8.2 Cluster Planning for Vector Retrieval

Off-heap memory is used for index construction and query in vector retrieval. Therefore, the required cluster capacity is related to the index type and off-heap memory size. You can estimate the off-heap memory required by full indexing to select proper cluster specifications.

There are different methods for estimating the size of off-heap memory required by different types of indexes. The calculation formulas are as follows:

- **Graph Index**

$$mem_needs = (dim \times dim_size + neighbors \times 4) \times num + delta$$

NOTE

If you need to update indexes in real time, consider the off-heap memory overhead required for vector index construction and automatic merge. The actual size of required **mem_needs** is at least 1.5 to 2 times of the original estimation.

- **PQ Index**

$$mem_needs = frag_num \times frag_size \times num + delta$$

- **FALT and IVF Indexes**

$$mem_needs = dim \times dim_size \times num + delta$$

Table 8-1 Parameter description

Parameter	Description
dim	Vector dimensions

Parameter	Description
neighbors	Number of neighbors of a graph node. The default value is 64 .
dim_size	Number of bytes required by each dimension. The default value is four bytes in the float type.
num	Total number of vectors
delta	Metadata size. This parameter can be left blank.
frag_num	Number of vector segments during quantization and coding. If this parameter is not specified when an index is created, the value is determined by the vector dimension dim . if dim <= 256: frag_num = dim / 4 elif dim <= 512: frag_num = dim / 8 else : frag_num = 64
frag_size	Size of the center point during quantization and coding. The default value is 1 . If the value of frag_num is greater than 256 , the value of frag_size is 2 .

These calculation methods can estimate the size of off-heap memory required by a complete vector index. To determine cluster specifications, you also need to consider the heap memory overhead of each node.

Heap memory allocation policy: The size of the heap memory of each node is half of the node physical memory, and the maximum size is **31 GB**.

For example, if you create a Graph index for the SIFT10M dataset, set **dim** to **128**, **dim_size** to **4**, **neighbors** to default value **64**, and **num** to **10 million**, the off-heap memory required by the Graph index is as follows:

$$mem_needs = (128 \times 4 + 64 \times 4) \times 10000000 \approx 7.5GB$$

Considering the overhead of heap memory, a single server with **8 vCPUs** and **16 GB memory** is recommended. If real-time write or update is required, you need to apply for larger memory.

8.3 Creating a Vector Index

Prerequisites

- A cluster of version 7.6.2 or 7.10.2 has been created by referring to [Cluster Planning for Vector Retrieval](#).
- Cluster advanced settings have been configured as required by referring to [Advanced Cluster Configurations](#).

Creating a Vector Index

1. Log in to the CSS management console.
2. Choose **Clusters** in the navigation pane. On the **Clusters** page, locate the target cluster and click **Access Kibana** in the **Operation** column.
3. Click **Dev Tools** in the navigation tree on the left and run the following command to create a vector index.

Create an index named **my_index** that contains a vector field **my_vector** and a text field **my_label**. The vector field creates the graph index and uses Euclidean distance to measure similarity.

```
PUT my_index
{
  "settings": {
    "index": {
      "vector": true
    }
  },
  "mappings": {
    "properties": {
      "my_vector": {
        "type": "vector",
        "dimension": 2,
        "indexing": true,
        "algorithm": "GRAPH",
        "metric": "euclidean"
      },
      "my_label": {
        "type": "text"
      }
    }
  }
}
```

Table 8-2 Parameters for creating an index

Type	Parameter	Description
Index settings parameters	vector	To use a vector index, set this parameter to true .
Field mappings parameters	type	Field type, for example, vector .
	dimension	Vector dimension. Value range: [1, 4096]

Type	Parameter	Description
	indexing	<p>Whether to enable vector index acceleration.</p> <p>The value can be:</p> <ul style="list-style-type: none"> • false: disables vector index acceleration. If this parameter is set to false, vector data is written only to docvalues, and only ScriptScore and Rescore can be used for vector query. • true: enables vector index acceleration. If this parameter is set to true, an extra vector index is created. The index algorithm is specified by the algorithm field and VectorQuery can be used for data query. <p>Default value: false</p>

Type	Parameter	Description
	algorithm	<p>Index algorithm. This parameter is valid only when indexing is set to true.</p> <p>The value can be:</p> <ul style="list-style-type: none"> • FLAT: brute-force algorithm that calculates the distance between the target vector and all vectors in sequence. The algorithm relies on sheer computing power and its recall rate reaches 100%. You can use this algorithm if you require high recall accuracy. • GRAPH: Hierarchical Navigable Small Worlds (HNSW) algorithm for graph indexes. This algorithm is mainly used in scenarios where high performance and precision are required and the data records of a single shard is fewer than 10 million. • GRAPH_PQ: combination of the HNSW algorithm and the PQ algorithm. The PQ algorithm reduces the storage overhead of original vectors, so that HNSW can easily search for data among hundreds of millions of records. • IVF_GRAPH: combination of IVF and HNSW. The entire space is divided into multiple cluster centroids, which makes search much faster but slightly inaccurate. You can use this algorithm if you require high performance when searching for data among hundreds of millions of records. • IVF_GRAPH_PQ: combination of the PQ algorithm with the IVF or HNSW algorithm to further improve the system capacity and reduce the system overhead. This algorithm is applicable to scenarios where there are more than 1 billion files in shards and high retrieval performance is required. <p>Default value: GRAPH</p> <p>NOTE If IVF_GRAPH or IVF_GRAPH_PQ is specified, you need to pre-build and register a central point index. For details, see (Optional) Pre-Building and Registering a Center Point Vector.</p>
	Table 8-3	If Indexing is set to true , CSS provides optional parameters for vector search to achieve higher query performance or precision.

Type	Parameter	Description
	metric	Method of calculating the distance between vectors. The value can be: <ul style="list-style-type: none"> • euclidean: Euclidean distance • inner_product: inner product distance • cosine: cosine distance • hamming: Hamming distance Default value: euclidean

Table 8-3 Optional parameters

Type	Parameter	Description
Graph index configuration parameters	neighbors	Number of neighbors of each vector in a graph index. The default value is 64 . A larger value indicates higher query precision. A larger index results in a slower build and query speed. Value range: [10, 255]
	shrink	Cropping coefficient during HNSW build. The default value is 1.0f . Value range: (0.1, 10)
	scaling	Scaling ratio of the upper-layer graph nodes during HNSW build. The default value is 50 . Value range: (0, 128]
	efc	Queue size of the neighboring node during HNSW build. The default value is 200 . A larger value indicates a higher precision and slower build speed. Value range: (0, 100000]
	max_scan_num	Maximum number of nodes that can be scanned. The default value is 10000 . A larger value indicates a higher precision and slower indexing speed. Value range: (0, 1000000]
PQ index configuration parameters	centroid_num	Number of cluster centroids of each fragment. The default value is 255 . Value range: (0, 65535]
	fragment_num	Number of fragments. The default value is 0 . The plugin automatically sets the number of fragments based on the vector length. Value range: [0, 4096]

Importing Vector Data

Run the following command to import vector data. When writing vector data to the **my_index** index, you need to specify the vector field name and vector data.

- If the input vector data is an array of floating-point numbers separated by commas (,):

```
POST my_index/_doc
{
  "my_vector": [1.0, 2.0]
}
```

- If the input vector data is a Base64 string encoded using little endian:

For a high dimensional vector has a large number of valid bits, the Base64 encoding format is efficient for data transmission and parsing.

```
POST my_index/_doc
{
  "my_vector": "AACAPwAAAEA="
}
```

- To write a large amount of data, bulk operations are recommended.

```
POST my_index/_bulk
{"index": {}}
{"my_vector": [1.0, 2.0], "my_label": "red"}
{"index": {}}
{"my_vector": [2.0, 2.0], "my_label": "green"}
{"index": {}}
{"my_vector": [2.0, 3.0], "my_label": "red"}
```

Advanced Cluster Configurations

- When importing data offline, you are advised to set **refresh_interval** of indexes to **-1** to disable automatic index refreshing and improve batch write performance.
- You are advised to set **number_of_replicas** to **0**. After the offline data import is complete, you can modify the parameter value as needed.
- The parameters of other advanced functions as follows:

Table 8-4 Cluster parameters

Parameter	Description
native.cache.circuit_breaker.enabled	Whether to enable the circuit breaker for off-heap memory. Default value: true
native.cache.circuit_breaker.cpu.limit	Upper limit of off-heap memory usage of the vector index. For example, if the overall memory of a host is 128 GB and the heap memory occupies 31 GB, the default upper limit of the off-heap memory usage is 43.65 GB, that is, (128 - 31) x 45%. If the off-heap memory usage exceeds its upper limit, the circuit breaker will be triggered. Default value: 45%

Parameter	Description
native.cache.expire.enabled	Whether to enable the cache expiration policy. If this parameter is set to true , some cache items that have not been accessed for a long time will be cleared. Value: true or false Default value: false
native.cache.expire.time	Expiration time. Default value: 24h
native.vector.index_threads	Number of threads used for creating underlying indexes. Each shard uses multiple threads. Set a relatively small value to avoid resource preemption caused by the build queries of too many threads. Default value: 4

8.4 Querying Vectors

Standard Query

Standard vector query syntax is provided for vector fields with vector indexes. The following command will return *n* (specified by **size/topk**) data records that are most close to the query vector.

```
POST my_index/_search
{
  "size":2,
  "_source": false,
  "query": {
    "vector": {
      "my_vector": {
        "vector": [1, 1],
        "topk":2
      }
    }
  }
}
```

Table 8-5 Parameters for standard query

Parameter	Description
vector (the first one)	Indicates that the query type is VectorQuery .
my_vector	Indicates the name of the vector field you want to query.
vector (the second one)	Indicates the vector value you want to query, which can be an array or a Base64 string
topk	Same as the value of size generally.

Parameter	Description
Table 8-6	Indicates optional query parameters. You can adjust the vector index parameters to achieve higher query performance or precision.

Table 8-6 Optional query parameters

Type	Parameter	Description
Graph index configuration parameters	ef	Queue size of the neighboring node during the query. A larger value indicates a higher query precision and slower query speed. The default value is 200 . Value range: (0, 100000]
	max_scan_number	Maximum number of scanned nodes. A larger value indicates a higher query precision and slower query speed. The default value is 10000 . Value range: (0, 1000000]
IVF index configuration parameters	nprobe	Number of center points. A larger value indicates a higher query precision and slower query speed. The default value is 100 . Value range: (0, 100000]

Compound Query

Vector search can be used together with other Elasticsearch subqueries, such as Boolean query and post-filtering, for compound query.

In the following two examples, top 10 (**topk**) results closest to the query vector are queried first. **filter** retains only the results whose **my_label** field is **red**.

- Example of a Boolean query

```
POST my_index/_search
{
  "size": 10,
  "query": {
    "bool": {
      "must": {
        "vector": {
          "my_vector": {
            "vector": [1, 2],
            "topk": 10
          }
        }
      }
    },
    "filter": {
      "term": { "my_label": "red" }
    }
  }
}
```

- Example of post-filtering

```
GET my_index/_search
{
  "size": 10,
  "query": {
    "vector": {
      "my_vector": {
        "vector": [1, 2],
        "topk": 10
      }
    }
  },
  "post_filter": {
    "term": { "my_label": "red" }
  }
}
```

ScriptScore Query

You can use **script_score** to perform Nearest Neighbor Search (NSS) on vectors. The query syntax is provided below.

The pre-filtering condition can be any query. **script_score** traverses only the pre-filtered results, calculates the vector similarity, and sorts and returns the results. The performance of this query depends on the size of the intermediate result set after the pre-filtering. If the pre-filtering condition is set to **match_all**, brute-force search is performed on all data.

```
POST my_index/_search
{
  "size": 2,
  "query": {
    "script_score": {
      "query": {
        "match_all": {}
      },
      "script": {
        "source": "vector_score",
        "lang": "vector",
        "params": {
          "field": "my_vector",
          "vector": [1.0, 2.0],
          "metric": "euclidean"
        }
      }
    }
  }
}
```

Table 8-7 script_score parameters

Parameter	Description
source	Script description. Its value is vector_score if the vector similarity is used for scoring.
lang	Script syntax description. Its value is vector .
field	Vector field name
vector	Vector data to be queried

Parameter	Description
metric	Measurement method, which can be euclidean , inner_product , cosine , and hamming . Default value: euclidean

Re-Score Query

If the **GRAPH_PQ** or **IVF_GRAPH_PQ** index is used, the query results are sorted based on the asymmetric distance calculated by PQ. CSS supports re-scoring and sorting of query results to improve the recall rate.

Assuming that **my_index** is a PQ index, an example of re-scoring the query results is as follows:

```
GET my_index/_search
{
  "size": 10,
  "query": {
    "vector": {
      "my_vector": {
        "vector": [1.0, 2.0],
        "topk": 100
      }
    }
  },
  "rescore": {
    "window_size": 100,
    "vector_rescore": {
      "field": "my_vector",
      "vector": [1.0, 2.0],
      "metric": "euclidean"
    }
  }
}
```

Table 8-8 Rescore parameter description

Parameter	Description
window_size	Vector retrieval returns <i>topk</i> search results and sorts the first <i>window_size</i> results.
field	Vector field name
vector	Vector data to be queried
metric	Measurement method, which can be euclidean , inner_product , cosine , and hamming . Default value: euclidean

8.5 Optimizing the Performance of Vector Retrieval

Optimizing Write Performance

- To reduce the cost of backup, disable the backup function before data import and enable it afterwards.
- Set **refresh_interval** to **120s** or a larger value. Larger segments can reduce the vector index build overhead caused by merging.
- Increase the value of **native.vector.index_threads** (the default value is **4**) to increase the number of threads for vector index build.

```
PUT _cluster/settings
{
  "persistent": {
    "native.vector.index_threads": 8
  }
}
```

Optimizing Query Performance

- After importing data in batches, you can run the **forcemerge** command to improve the query efficiency.
- If the off-heap memory required by the vector index exceeds the circuit breaker limit, index entry swap-in and swap-out occur, which affects the query performance. In this case, you can increase the circuit breaker threshold of off-heap memory.

```
POST index_name/_forcemerge?max_num_segments=1
```

```
PUT _cluster/settings
{
  "persistent": {
    "native.cache.circuit_breaker.cpu.limit": "75%"
  }
}
```

- If the end-to-end latency is greater than the **took** value in the returned result, you can configure **_source** to reduce the **fdt** file size and reduce the **fetch** overhead.

```
PUT my_index
{
  "settings": {
    "index": {
      "vector": "true"
    },
    "index.soft_deletes.enabled": false
  },
  "mappings": {
    "_source": {
      "excludes": ["my_vector"]
    },
    "properties": {
      "my_vector": {
        "type": "vector",
        "dimension": 128,
        "indexing": true,
        "algorithm": "GRAPH",
        "metric": "euclidean"
      }
    }
  }
}
```

8.6 (Optional) Pre-Building and Registering a Center Point Vector

When you perform operations in [Creating a Vector Index](#), if **IVF_GRAPH** and **IVF_GRAPH_PQ** index algorithms are selected, you need to pre-build and register the center point vector.

Context

The vector index acceleration algorithms **IVF_GRAPH** and **IVF_GRAPH_PQ** are suitable for ultra-large-scale computing. These two algorithms allow you to narrow down the query range by dividing a vector space into subspaces through clustering or random sampling. Before pre-build, you need to obtain all center point vectors by clustering or random sampling.

Then, pre-construct and register the center point vectors to create the **GRAPH** or **GRAPH_PQ** index and register them with the Elasticsearch cluster. All nodes in the cluster can share the index file. Reuse of the center index among shards can effectively reduce the training overhead and the number of center index queries, improving the write and query performance.

Procedure

1. On the **Clusters** page, locate the target cluster, and click **Access Kibana** in the **Operation** column.
2. Click **Dev Tools** in the navigation tree on the left.
3. Create a center point index table.
 - For example, if the created index is named **my_dict**, **number_of_shards** of the index must be set to **1**. Otherwise, the index cannot be registered.
 - If you want to use the **IVF_GRAPH** index, set **algorithm** of the center point index to **GRAPH**.
 - If you want to use the **IVF_GRAPH_PQ** index, set **algorithm** of the center point index to **GRAPH_PQ**.

```
PUT my_dict
{
  "settings": {
    "index": {
      "vector": true
    },
    "number_of_shards": 1,
    "number_of_replicas": 0
  },
  "mappings": {
    "properties": {
      "my_vector": {
        "type": "vector",
        "dimension": 2,
        "indexing": true,
        "algorithm": "GRAPH",
        "metric": "euclidean"
      }
    }
  }
}
```

4. Write the center point vector to the created index.
Write the center point vector obtained through sampling or clustering into the created **my_dict** index by referring to [Importing Vector Data](#).

5. Call the registration API.
Register the created **my_dict** index with a **Dict** object with a globally unique identifier name (**dict_name**).

```
PUT _vector/register/my_dict
{
  "dict_name": "my_dict"
}
```

6. Create an **IVF_GRAPH** or **IVF_GRAPH_PQ** index.

You do not need to specify the dimension and metric information. Simply specify the registered dictionary name.

```
PUT my_index
{
  "settings": {
    "index": {
      "vector": true
    }
  },
  "mappings": {
    "properties": {
      "my_vector": {
        "type": "vector",
        "indexing": true,
        "algorithm": "IVF_GRAPH",
        "dict_name": "my_dict",
        "offload_ivf": false
      }
    }
  }
}
```

Table 8-9 Field mappings parameters

Parameter	Description
dict_name	Specifies the name of the depended central point index. The vector dimension and measurement metric of the index are the same as those of the Dict index.
offload_ivf	Unloads the IVF inverted index implemented by the underlying index to Elasticsearch. In this way, the use of non-heap memory and the overhead of write and merge operations are reduced. However, the query performance also deteriorates. You can use the default value. Value: true or false Default value: false

8.7 Managing the Vector Index Cache

The vector retrieval engine is developed in C++ and uses off-heap memory. You can use the following APIs to manage the index cache.

- **View cache statistics.**

```
GET /_vector/stats
```

In the implementation of the vector plug-in, the vector index is the same as other types of Lucene indexes. Each segment constructs and stores an index file. During query, the index file is loaded to the non-heap memory. The plug-in uses the cache mechanism to manage the non-heap memory. You can use this API to query the non-heap memory usage, number of cache hits, and number of loading times.

- **Preload the vector index.**

```
PUT /_vector/warmup/{index_name}
```

You can use this API to preload the vector index specified by **index_name** to the off-heap memory for query.

- **Clear the cache.**

```
PUT /_vector/clear/cache
```

```
PUT /_vector/clear/cache/index_name
```

The caching mechanism limits the non-heap memory usage when vector indexes are used. When the total index size exceeds the cache size limit, index entry swap-in and swap-out occur, which affects the query performance. You can use this API to clear unnecessary index cache to ensure the query performance of hot data indexes.

8.8 Sample Code for Vector Search on a Client

Elasticsearch provides standard REST APIs and clients developed using Java, Python, and Go.

Based on the open-source dataset **SIFT1M** (<http://corpus-texmex.irisa.fr/>) and Python Elasticsearch client, this section provides a code snippet for creating a vector index, importing vector data, and querying vector data on the client.

Prerequisites

The Python dependency package has been installed on the client. If it is not installed, run the following commands to install it:

```
pip install numpy
pip install elasticsearch==7.6.0
```

Sample Code

```
import numpy as np
import time
import json

from concurrent.futures import ThreadPoolExecutor, wait
from elasticsearch import Elasticsearch
from elasticsearch import helpers

endpoint = 'http://xxx.xxx.xxx.xxx:9200/'

# Construct an Elasticsearch client object
es = Elasticsearch(endpoint)

# Index mapping information
index_mapping = ""
{
  "settings": {
```

```
"index": {
  "vector": "true"
}
},
"mappings": {
  "properties": {
    "my_vector": {
      "type": "vector",
      "dimension": 128,
      "indexing": true,
      "algorithm": "GRAPH",
      "metric": "euclidean"
    }
  }
}
}
}
"""

# Create an index.
def create_index(index_name, mapping):
    res = es.indices.create(index=index_name, ignore=400, body=mapping)
    print(res)

# Delete an index.
def delete_index(index_name):
    res = es.indices.delete(index=index_name)
    print(res)

# Refresh indexes.
def refresh_index(index_name):
    res = es.indices.refresh(index=index_name)
    print(res)

# Merge index segments.
def merge_index(index_name, seg_cnt=1):
    start = time.time()
    es.indices.forcemerge(index=index_name, max_num_segments=seg_cnt, request_timeout=36000)
    print(f" Complete the merge within {time.time() - start} seconds")

# Load vector data.
def load_vectors(file_name):
    fv = np.fromfile(file_name, dtype=np.float32)
    dim = fv.view(np.int32)[0]
    vectors = fv.reshape(-1, 1 + dim)[: , 1:]
    return vectors

# Load the ground_truth data.
def load_gts(file_name):
    fv = np.fromfile(file_name, dtype=np.int32)
    dim = fv.view(np.int32)[0]
    gts = fv.reshape(-1, 1 + dim)[: , 1:]
    return gts

def partition(ls, size):
    return [ls[i:i + size] for i in range(0, len(ls), size)]

# Write vector data.
def write_index(index_name, vec_file):
    pool = ThreadPoolExecutor(max_workers=8)
    tasks = []

    vectors = load_vectors(vec_file)
    bulk_size = 1000
```

```
partitions = partition(vectors, bulk_size)

start = time.time()
start_id = 0
for vecs in partitions:
    tasks.append(pool.submit(write_bulk, index_name, vecs, start_id))
    start_id += len(vecs)
wait(tasks)
print(f" Complete the writing within {time.time() - start} seconds")

def write_bulk(index_name, vecs, start_id):
    actions = [
        {
            "_index": index_name,
            "my_vector": vecs[j].tolist(),
            "_id": str(j + start_id)
        }
        for j in range(len(vecs))
    ]
    helpers.bulk(es, actions, request_timeout=3600)

# Query an index.
def search_index(index_name, query_file, gt_file, k):
    print("Start query! Index name: " + index_name)

    queries = load_vectors(query_file)
    gt = load_gts(gt_file)

    took = 0
    precision = []
    for idx, query in enumerate(queries):
        hits = set()
        query_json = {
            "size": k,
            "_source": False,
            "query": {
                "vector": {
                    "my_vector": {
                        "vector": query.tolist(),
                        "topk": k
                    }
                }
            }
        }
        res = es.search(index=index_name, body=json.dumps(query_json))

        for hit in res['hits']['hits']:
            hits.add(int(hit['_id']))
        precision.append(len(hits.intersection(set(gt[idx, :k]))) / k)
        took += res['took']

    print("precision: " + str(sum(precision) / len(precision)))
    print(f" Complete the retrieval within {took / 1000:.2f} seconds; average took size is {took / len(queries):.2f} ms")

if __name__ == "__main__":
    vec_file = r"./data/sift/sift_base.fvecs"
    qry_file = r"./data/sift/sift_query.fvecs"
    gt_file = r"./data/sift/sift_groundtruth.ivecs"

    index = "test"
    create_index(index, index_mapping)
    write_index(index, vec_file)
    merge_index(index)
    refresh_index(index)
```

```
search_index(index, qry_file, gt_file, 10)
```


9 Working with Kibana

9.1 Logging In to Kibana

Prerequisites

A CSS cluster has been created.

Procedure

- Logging in to the console
 - a. Log in to the CSS management console.
 - b. On the **Clusters** page, locate the target cluster and click **Access Kibana** in the **Operation** column to go to the Kibana login page.
 - Non-security cluster: The Kibana console is displayed.
 - Security cluster: Enter the username and password on the login page and click **Log In** to go to the Kibana console. The default username is **admin** and the password is the one specified during cluster creation.
 - c. After the login is successful, you can access the Elasticsearch cluster through Kibana.
- Logging in using an EIP

If you have enabled Kibana public access during cluster creation, you can use the Kibana public IP address to log in to the cluster. For details, see .

9.2 Creating a User and Granting Permissions by Using Kibana

Prerequisites

The security mode has been enabled for the cluster.

Description

Table 9-1 Parameters for creating a user and assigning permissions on Kibana

Parameter	Description
Permission	Single action, for example, creating an index (for example, indices:admin/create)
Action group	A group of permissions. For example, the predefined SEARCH action group grants roles permissions to use _search and _msearchAPI .
Role	A role is a combination of permissions and action groups, including operation permissions on clusters, indexes, documents, or fields.
Backend role	(Optional) Other external roles from the backend such as LDAP/Active Directory
User	A user can send operation requests to the Elasticsearch cluster. The user has credentials such as username and password, zero or more backend roles, and zero or more custom attributes.
Role mapping	A user will be assigned a role after successful authentication. Role mapping is to map a role to a user (or a backend role). For example, the mapping from kibana_user (role) to jdoh (user) means that John Doe obtains all permissions of kibana_user after being authenticated by kibana_user . Similarly, the mapping from all_access (role) to admin (backend role) means that any user with the backend role admin (from the LDAP/Active Directory server) has all the permissions of role all_access after being authenticated. You can map each role to multiple users or backend roles.

Procedure

NOTE

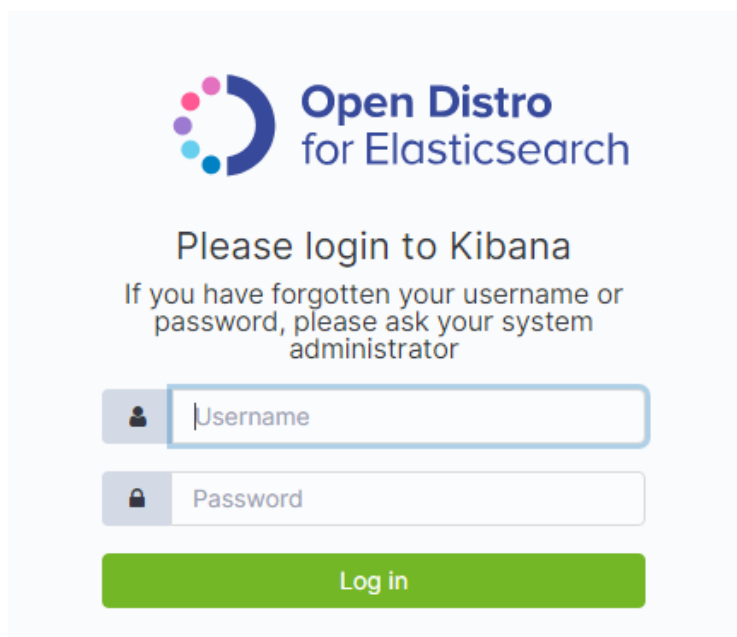
- The Kibana UI varies depending on the Kibana version, but their operations are similar. This section takes Kibana 7.6.2 as an example to describe the procedure.
- You can customize the username, role name, and tenant name in Kibana.

1. Log in to the CSS management console.
2. Choose **Clusters** in the navigation pane. On the **Clusters** page, locate the target cluster and click **Access Kibana** in the **Operation** column.

Enter the administrator username and password to log in to Kibana.

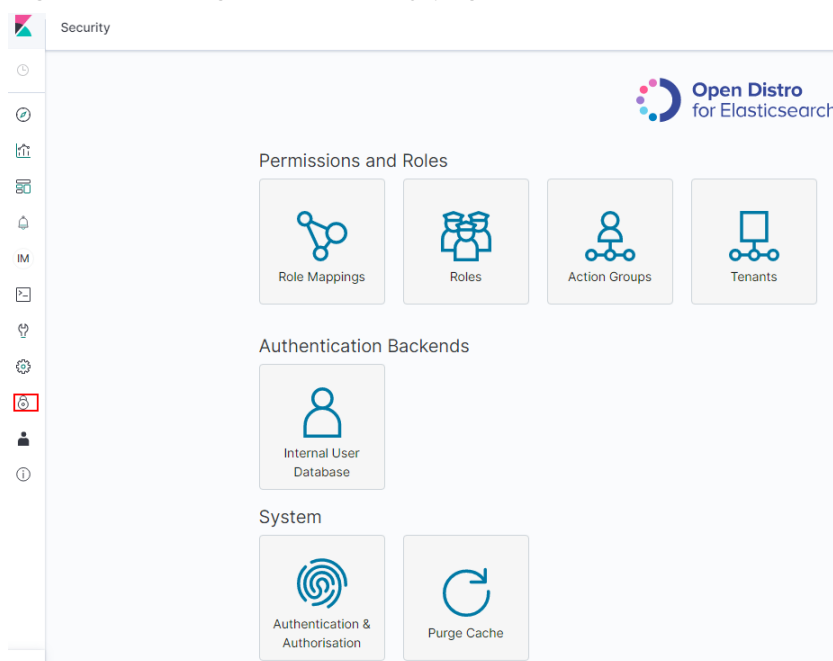
- Username: admin (default administrator account name)
- Password: Enter the administrator password you set when creating the cluster in security mode.

Figure 9-1 Login page



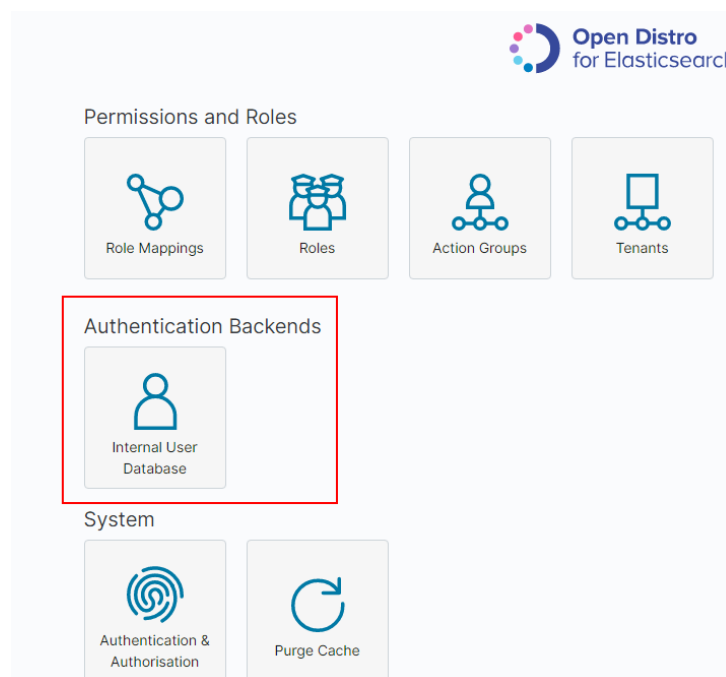
3. Click the **Security** icon on the Kibana operation page.

Figure 9-2 Going to the Security page



4. Create a user.
 - a. Choose **Authentication Backends > Internal Users Database**.

Figure 9-3 Adding a user (1)




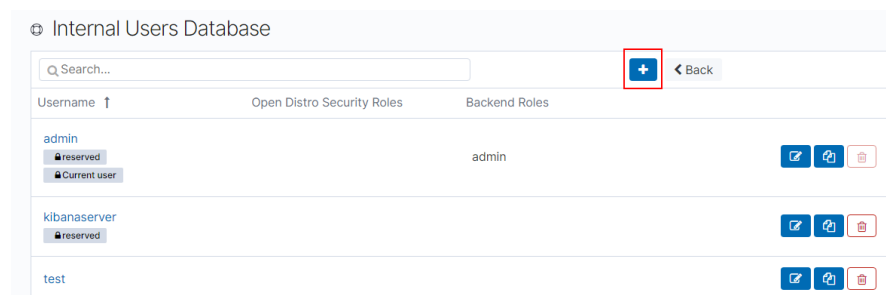
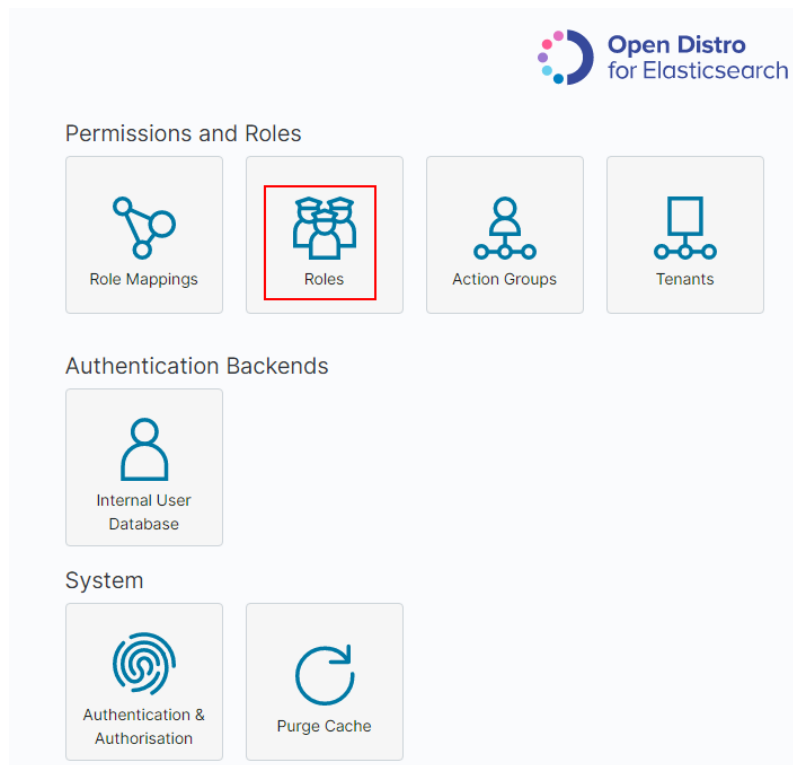
- b. On the **Internal Users Database** page, choose  . The page for adding user information is displayed.

Figure 9-4 Adding a user (2)



- c. On the user creation page, configure **Username** and **Password**, and click **Submit**.
The user will be displayed in the user list.
- 5. Configure roles and permissions for the user.
 - a. Click **Roles**.

Figure 9-5 Adding a role




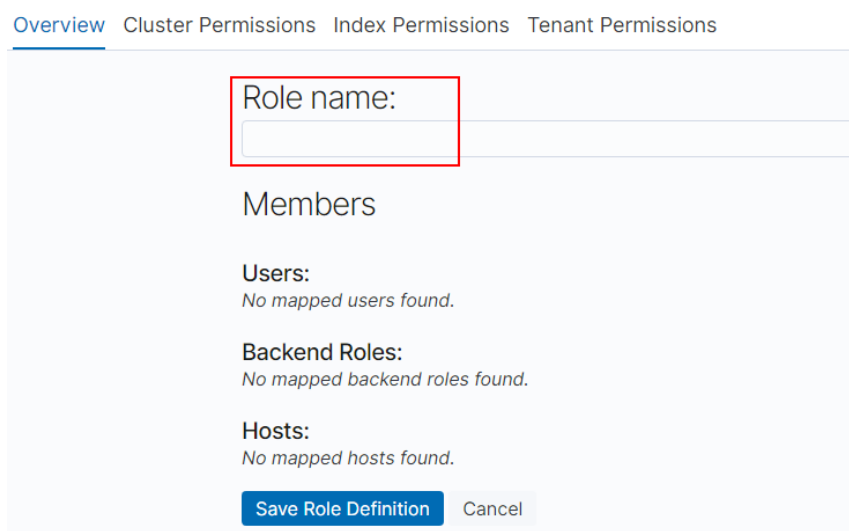
- b. On the **Open Distro Security Roles** page, click  .
 - i. Enter a role name on the **Overview** page.

Figure 9-6 Entering a role name



- ii. Configure CSS cluster permissions on the **Cluster Permissions** page. You can skip this step.
- iii. Configure index permissions on the **Index Permissions** page.
Index patterns: Set this parameter to the name of the index whose permission needs to be configured. For example, **my_store**.

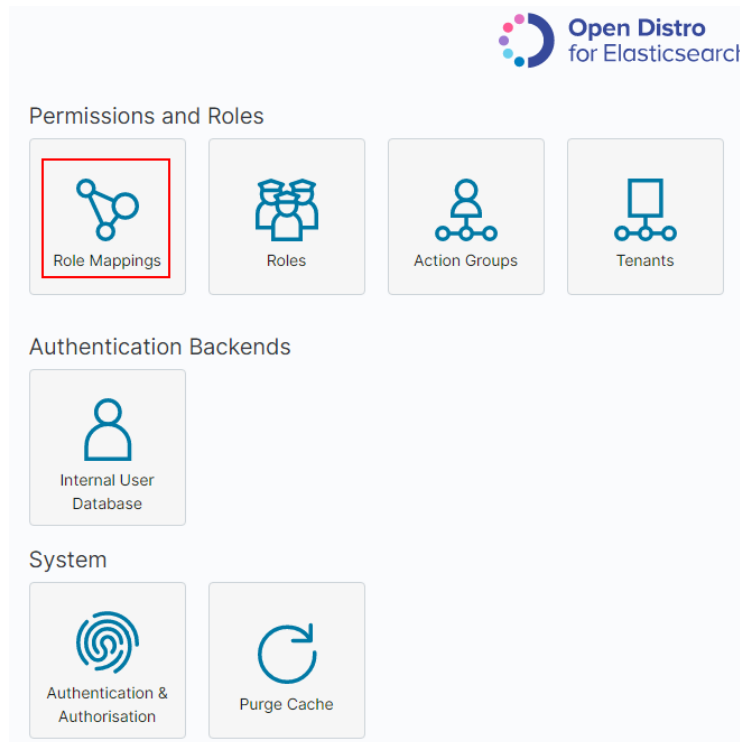
 **NOTE**

Use different names for the index and the user.

Configure **Permissions: Action Groups** as required, for example, select the read-only permission **Search**.

- iv. Retain the default settings on the **Tenant Permissions** page.
After the configuration is complete, the role will be displayed.
- 6. Assign a role to the user.
 - a. Click **Role Mappings**.

Figure 9-7 Role mapping




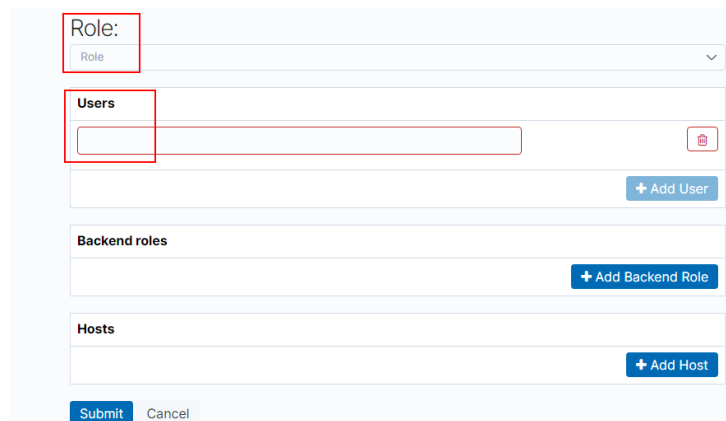
- b. Click  to add the mapping between users and roles.

Figure 9-8 Users and roles



- c. Click **Submit**.
7. Verify that the configuration takes effect in Kibana.

9.3 Managing Index Statuses

9.3.1 Creating and Managing Indexes

Clusters of version 7.6.2 or later support index status management. ISM is a plugin that allows you to automate periodic and administrative operations based on changes on the index age, index size, or number of documents. When using the ISM plug-in, you can define policies that automatically handle index rollovers or deletions based on your needs.

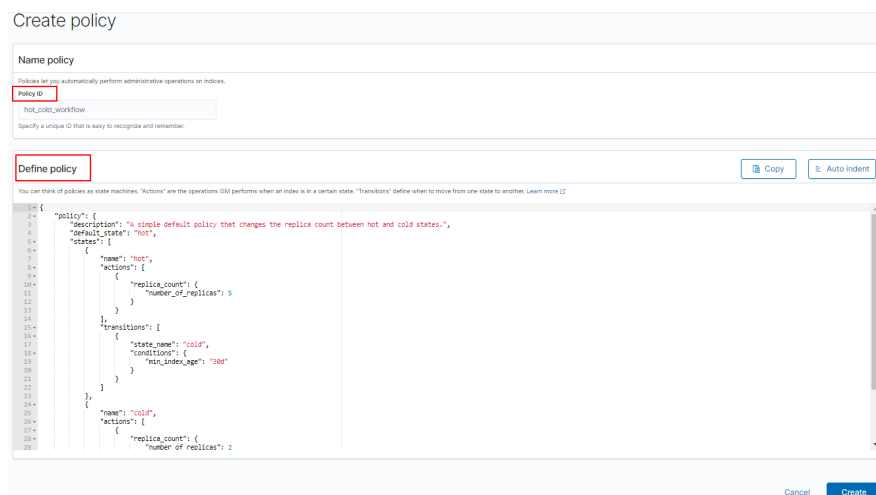
NOTE

The following procedure uses Kibana 7.6.2 as an example. The Kibana UI varies depending on the Kibana version, but their operations are similar.

Creating an Index Policy

1. Log in to Kibana and choose **IM** or **Index Management** on the left. The **Index Management** page is displayed.
2. Click **Create policy** to create an index policy.
3. Enter a policy ID in the **Policy ID** text box and enter your policy in the **Define policy** text box.

Figure 9-9 Configuring a policy



4. Click **Create**.

Attaching a Policy to an Index

You can attach a policy to one or more indexes and add the policy ID to an index template. When you create indexes using that index template pattern, the policy will be attached to all created indexes.

- **Method 1: Kibana commands**

On the **Dev Tools** page of Kibana, run the following command to associate a policy ID with an index template:

```
PUT _template/<template_name>
{
  "index_patterns": ["index_name-*"],
  "settings": {
    "opendistro.index_state_management.policy_id": "policy_id"
  }
}
```

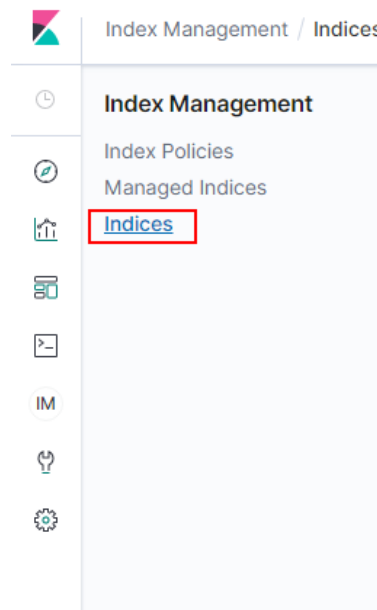
- **<template_name>**: Replace it with the name of a created index template.
- **policy_id**: Replace it with a custom policy ID.

For details about how to create an index template, see [Index Template](#).

- **Method 2: Kibana console**

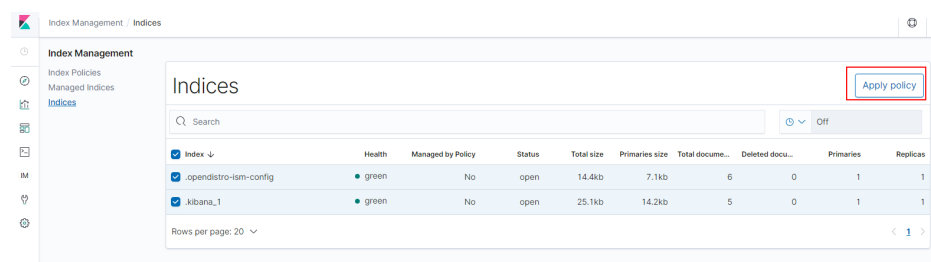
- On the **Index Management** page of Kibana, choose **Indices**.

Figure 9-10 Choosing Indices



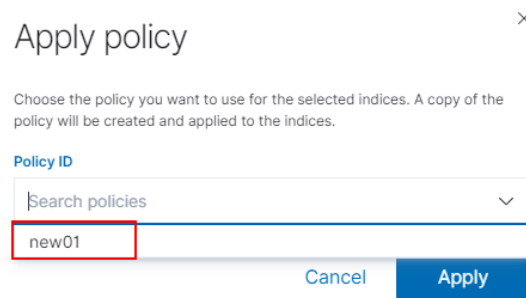
- In the **Indices** list, select the target index to which you want to attach a policy.
- Click **Apply policy** in the upper right corner.

Figure 9-11 Adding a policy



- d. Select the policy you created from the **Policy ID** drop-down list.

Figure 9-12 Selecting a policy



- e. Click **Apply**.
After you attach a policy to an index, ISM creates a job that runs every 5 minutes by default, to execute the policy, check conditions, and convert the index to different statuses.

Managing Index Policies

1. Click **Managed Indices**.
2. If you want to change the policy, click **Change policy**. For details, see [Changing Policies](#).
3. To delete a policy, select your policy, and click **Remove policy**.
4. To retry a policy, select your policy, and click **Retry policy**.

For details, see [Index State Management](#).

9.3.2 Changing Policies

You can change any managed index policy. ISM has constraints to ensure that policy changes do not break indexes.

If an index is stuck in its current status, never proceeding, and you want to update its policy immediately, make sure that the new policy includes the same status (same name, action, and order) as the old policy. In this case, ISM applies the new policy even if the policy is being executed.

If you update the policy without including an identical status, ISM updates the policy only after all actions in the current status finish executing. Alternatively, you can select a specific status in the old policy and have the new policy take effect.

To change a policy using Kibana, do the following:

1. Under **Managed Indices**, select the indices to which you want to attach the new policy.
2. Click **Change policy** in the upper right corner. The **Choose managed indices** page is displayed. Configure parameters required for changing a policy.

Table 9-2 Parameters required for changing a policy

Parameter	Description
Managed indices	Select the indexes to which you want to attach the new policy. Multiple indices can be selected.
State filters	Select an index status. When a status is selected, the new policy is attached to an index in this status.
New policy	Select a new policy.

3. After configuration is complete, click **Change**.

9.4 How Do I Connect a User-built Kibana with Elasticsearch?

To connect the self-built Kibana with Elasticsearch, the following conditions must be met:

1. The local environment must support access from external networks.
2. Kibana is built using ECS in the same VPC as Elasticsearch. Kibana can be accessed from the local public network.

Example of a Kibana configuration file:

Security mode:

```
elasticsearch.username: "****"
elasticsearch.password: "****"
elasticsearch.ssl.verificationMode: none
server.ssl.enabled: false
server.rewriteBasePath: false
server.port: 5601
logging.dest: /home/Ruby/log/kibana.log
pid.file: /home/Ruby/run/kibana.pid
server.host: 192.168.25.226
elasticsearch.hosts: https://10.0.0.207:9200
elasticsearch.requestHeadersWhitelist: ["securitytenant","Authorization"]
opendistro_security.multitenancy.enabled: true
opendistro_security.multitenancy.tenants.enable_global: true
opendistro_security.multitenancy.tenants.enable_private: true
opendistro_security.multitenancy.tenants.preferred: ["Private", "Global"]
opendistro_security.multitenancy.enable_filter: false
```

NOTE

- In security mode, the **opendistro_security_kibana** plug-in must be installed. For details, see: <https://opendistro.github.io/for-elasticsearch-docs/docs/kibana/plugins/>
- The version of the installed plug-in must be the same as that of the cluster. To check the version of the plug-in version, run the **GET _cat/plugins** command.

Non-security mode

```
server.port: 5601
logging.dest: /home/Ruby/log/kibana.log
pid.file: /home/Ruby/run/kibana.pid
server.host: 192.168.25.226
elasticsearch.hosts: http://10.0.0.207:9200
```

9.5 Kibana Usage Restrictions

You can customize the username, role name, and tenant name in Kibana.

10 Elasticsearch SQL

For Elasticsearch 6.5.4 and later versions, Open Distro for Elasticsearch SQL lets you write queries in SQL rather than in the Elasticsearch query domain-specific language (DSL).

If you are already familiar with SQL and do not want to learn query DSL, this feature is a great option.

Basic Operations

- Kibana (recommended)
 - Log in to Kibana and send requests using request parameters or request body to **_opendistro/_sqlURI** in the **Dev Tools** page.

```
GET _opendistro/_sql?sql=select * from my-index limit 50
POST _opendistro/_sql
{
  "query": "SELECT * FROM my-index LIMIT 50"
}
```

- By default, the result is returned in the JSON structure. If you want the result to be returned in the CSV format, run the following command:

```
POST _opendistro/_sql?format=csv
{
  "query": "SELECT * FROM my-index LIMIT 50"
}
```

When data is returned in the CSV format, each row corresponds to a document and each column corresponds to a field.

- cURL commands
You can also run cURL commands in ECS to execute SQL statements.

```
curl -XPOST https://localhost:9200/_opendistro/_sql -u username:password -k -d '{"query": "SELECT * FROM kibana_sample_data_flights LIMIT 10"}' -H 'Content-Type: application/json'
```

Supported Operations

Open Distro for Elasticsearch supports the following SQL operations: statements, conditions, aggregations, include and exclude fields, common functions, joins, and show.

- Statements

Table 10-1 Statements

Statement	Example
Select	SELECT * FROM my-index
Delete	DELETE FROM my-index WHERE _id=1
Where	SELECT * FROM my-index WHERE ['field']='value'
Order by	SELECT * FROM my-index ORDER BY _id asc
Group by	SELECT * FROM my-index GROUP BY range(age, 20,30,39)
Limit	SELECT * FROM my-index LIMIT 50 (default is 200)
Union	SELECT * FROM my-index1 UNION SELECT * FROM my-index2
Minus	SELECT * FROM my-index1 MINUS SELECT * FROM my-index2

 **NOTE**

As with any complex query, large UNION and MINUS statements can strain or even crash your cluster.

- Conditions

Table 10-2 Conditions

Condition	Example
Like	SELECT * FROM my-index WHERE name LIKE 'j%'
And	SELECT * FROM my-index WHERE name LIKE 'j%' AND age > 21
Or	SELECT * FROM my-index WHERE name LIKE 'j%' OR age > 21
Count distinct	SELECT count(distinct age) FROM my-index
In	SELECT * FROM my-index WHERE name IN ('alejandro', 'carolina')
Not	SELECT * FROM my-index WHERE name NOT IN ('jane')
Between	SELECT * FROM my-index WHERE age BETWEEN 20 AND 30
Aliases	SELECT avg(age) AS Average_Age FROM my-index
Date	SELECT * FROM my-index WHERE birthday='1990-11-15'
Null	SELECT * FROM my-index WHERE name IS NULL

- Aggregations

Table 10-3 Aggregations

Aggregation	Example
avg()	SELECT avg(age) FROM my-index
count()	SELECT count(age) FROM my-index
max()	SELECT max(age) AS Highest_Age FROM my-index
min()	SELECT min(age) AS Lowest_Age FROM my-index
sum()	SELECT sum(age) AS Age_Sum FROM my-index

- Include and exclude fields

Table 10-4 Include and exclude fields

Pattern	Example
include()	SELECT include('a*'), exclude('age') FROM my-index
exclude()	SELECT exclude('*name') FROM my-index

- Functions

Table 10-5 Functions

Function	Example
floor	SELECT floor(number) AS Rounded_Down FROM my-index
trim	SELECT trim(name) FROM my-index
log	SELECT log(number) FROM my-index
log10	SELECT log10(number) FROM my-index
substring	SELECT substring(name, 2,5) FROM my-index
round	SELECT round(number) FROM my-index
sqrt	SELECT sqrt(number) FROM my-index
concat_ws	SELECT concat_ws(' ', age, height) AS combined FROM my-index

Function	Example
/	SELECT number / 100 FROM my-index
%	SELECT number % 100 FROM my-index
date_format	SELECT date_format(date, 'Y') FROM my-index

 **NOTE**

You must enable fielddata in the document mapping for most string functions to work properly.

- Joins

Table 10-6 Joins

Join	Example
Inner join	SELECT s.firstname, s.lastname, s.gender, sc.name FROM student s JOIN school sc ON sc.name = s.school_name WHERE s.age > 20
Left outer join	SELECT s.firstname, s.lastname, s.gender, sc.name FROM student s LEFT JOIN school sc ON sc.name = s.school_name
Cross join	SELECT s.firstname, s.lastname, s.gender, sc.name FROM student s CROSS JOIN school sc

For details about the restrictions, see [Joins](#).

- Show

Show commands display indexes and mappings that match an index pattern. You can use * or % for wildcards.

Table 10-7 Show

Show	Example
Show tables like	SHOW TABLES LIKE logs-*

Joins

Open Distro for Elasticsearch SQL supports inner joins, left outer joins and cross joins. Joins have the following constraints:

- You can only join two indexes.
- You must use an alias for an index (for example, people p).
- In an ON clause, you can only use the AND conditions.
- In a WHERE statement, do not combine trees that contain multiple indexes.
For example, the following statement will work:
`WHERE (a.type1 > 3 OR a.type1 < 0) AND (b.type2 > 4 OR b.type2 < -1)`
- The following statement will not work:
`WHERE (a.type1 > 3 OR b.type2 < 0) AND (a.type1 > 4 OR b.type2 < -1)`
- You cannot use GROUP BY or ORDER BY to obtain results.
- LIMIT with OFFSET (for example, LIMIT 25 OFFSET 25) is not supported.

JDBC Driver

The Java Database Connectivity (JDBC) driver allows you to integrate Open Distro for Elasticsearch with your business intelligence (BI) applications.

For details about how to download and use JAR files, see [GitHub Repositories](#).

11 Enhanced Features

11.1 Storage-Compute Decoupling

11.1.1 Context

You can store hot data on SSD to achieve the optimal query performance, and store historical data in OBS to reduce data storage costs.

Application Scenarios

A large volume of data is written to and stored in SSDs. If historical data is no longer updated (is turned into cold data) and its QPS decreases, you can call CSS APIs to dump hot data from SSDs to OBS buckets. This operation freezes indexes, decoupling compute from storage.

Constraints

- Currently, only versions 7.6.2 and 7.10.2 support storage-compute decoupling.
- The storage-compute decoupling feature depends on OBS. Therefore, you must comply with the restrictions on OBS bandwidth and QPS. For details, see OBS Restrictions. If these restrictions are violated, the performance of queries on OBS will deteriorate. For example, the speed of restoring shards and querying data will become slow.

11.1.2 Freezing an Index

Precautions

- Before freezing an index, ensure no data is being written to it. The index will be set to read only before being frozen, and data write will fail.
- After an index is frozen,
 - It becomes read-only.
 - The index data will be dumped to OBS. This process occupies network bandwidth.

- The query latency of a dumped index will increase. During aggregation, the latency of processing complex queries and reading a large volume of data is long.
- It cannot be unfrozen. That is, a read-only index cannot be changed to writable.
- After the freezing is complete, the index data in your local disks will be deleted.

Procedure

1. Log in to the CSS management console.
2. Choose **Clusters** in the navigation pane. On the **Clusters** page, locate the target cluster and click **Access Kibana** in the **Operation** column.
3. Click **Dev Tools** in the navigation tree on the left.
4. Run the following command to freeze a specified index and dump it to OBS:
POST `/${index_name}/_freeze_low_cost`

Table 11-1 Parameter description

Parameter	Description
index_name	Name of the index to be frozen.

Information similar to the following is displayed:

```
{
  "freeze_uuid": "pdsRgUtStymVDWR_HoTGFw"
}
```

Table 11-2 Response parameter

Parameter	Description
freeze_uuid	After an index freezing request is submitted, an asynchronous job will be started. The request returns the asynchronous job ID, which can be used to query the progress of the asynchronous job.

NOTE

After an index freezing request is submitted, data cannot be written to the index. During the index freezing, query requests are not affected. After the freezing is complete, the index is closed and then opened. During this period, the index cannot be queried, and the cluster may be in the **red** status for a short time. The index is restored after being opened.

5. Run the following command to check the freezing task progress:
GET `_freeze_low_cost_progress/${freeze_uuid}`

Table 11-3 Parameter description

Parameter	Description
freeze_uuid	Asynchronous task ID, which is obtained in 4 .

Information similar to the following is displayed:

```
{
  "stage": "STARTED",
  "shards_stats": {
    "INIT": 0,
    "FAILURE": 0,
    "DONE": 0,
    "STARTED": 3,
    "ABORTED": 0
  },
  "indices": {
    "data1": [
      {
        "uuid": "70S-G1-tRke2jHZPlckexg",
        "index": {
          "name": "data1",
          "index_id": "4b5PHXJITLaS6AurImfQ9A",
          "shard": 2
        },
        "start_ms": 1611972010852,
        "end_ms": -1,
        "total_time": "10.5s",
        "total_time_in_millis": 10505,
        "stage": "STARTED",
        "failure": null,
        "size": {
          "total_bytes": 3211446689,
          "finished_bytes": 222491269,
          "percent": "6.0%"
        },
        "file": {
          "total_files": 271,
          "finished_files": 12,
          "percent": "4.0%"
        },
        "rate_limit": {
          "paused_times": 1,
          "paused_nanos": 946460970
        }
      }
    ],
    {
      "uuid": "70S-G1-tRke2jHZPlckexg",
      "index": {
        "name": "data1",
        "index_id": "4b5PHXJITLaS6AurImfQ9A",
        "shard": 0
      },
      "start_ms": 1611972010998,
      "end_ms": -1,
      "total_time": "10.3s",
      "total_time_in_millis": 10359,
      "stage": "STARTED",
      "failure": null,
      "size": {
        "total_bytes": 3221418186,
        "finished_bytes": 272347118,
        "percent": "8.0%"
      },
      "file": {
```

```

    "total_files" : 372,
    "finished_files" : 16,
    "percent" : "4.0%"
  },
  "rate_limit" : {
    "paused_times" : 5,
    "paused_nanos" : 8269016764
  }
},
{
  "uuid" : "7OS-G1-tRke2jHZPlckexg",
  "index" : {
    "name" : "data1",
    "index_id" : "4b5PHXJITLaS6AurlmfQ9A",
    "shard" : 1
  },
  "start_ms" : 1611972011021,
  "end_ms" : -1,
  "total_time" : "10.3s",
  "total_time_in_millis" : 10336,
  "stage" : "STARTED",
  "failure" : null,
  "size" : {
    "total_bytes" : 3220787498,
    "finished_bytes" : 305789614,
    "percent" : "9.0%"
  },
  "file" : {
    "total_files" : 323,
    "finished_files" : 14,
    "percent" : "4.0%"
  },
  "rate_limit" : {
    "paused_times" : 3,
    "paused_nanos" : 6057933087
  }
}
]
}
}

```

Table 11-4 Response parameters

Parameter	Description
stage	Status. Its value can be: <ul style="list-style-type: none"> ● INIT: The instance has just started or is being initialized. ● FAILURE: failed ● DONE: complete ● STARTED: started ● ABORTED: Canceled. This field is reserved.
shards_stats	Numbers of shards in each state.
indices	Index status details.

Table 11-5 Return values of **indices**

Parameter	Description
uuid	UUID of the freezing operation
index	Index and shard information
start_ms	Start time
end_ms	End time. If no end time is specified, the value -1 is displayed.
total_time	Time spent
total_time_in_millis	Time spent, in milliseconds
stage	Status of the current shard.
failure	Failure cause. If no failure occurs, null is displayed.
size.total_bytes	Size of files to be frozen, in bytes
size.finished_bytes	Frozen bytes
size.percent	Percentage of frozen bytes
file.total_bytes	Number of files to be frozen
file.finished_bytes	Number of frozen files
file.percent	Percentage of frozen files
rate_limit.paused_times	Number of times that freezing is suspended due to rate limit
rate_limit.paused_nanos	Duration of freezing task suspension due to rate limit, in nanoseconds

The following parameters are added to a frozen index. For details, see [Table 11-6](#).

Table 11-6 Frozen index parameters

Parameter	Description
index.frozen_low_cost	Whether an index is frozen. The value is true .
index.blocks.write	Whether data writing is denied in a frozen index. The value is true .
index.store.type	Storage type of an index. The value is obs .

- After an index is frozen, its data will be cached. Run the following command to check the current cache status: For details about the cache, see [Configuring Cache](#).

```
GET _frozen_stats
GET _frozen_stats/${node_id}
```

Table 11-7 Parameter description

Parameter	Description
node_id	Node ID, which can be used to obtain the cache status of a node.

Information similar to the following is displayed:

```
{
  "_nodes" : {
    "total" : 3,
    "successful" : 3,
    "failed" : 0
  },
  "cluster_name" : "css-zzz1",
  "nodes" : {
    "7uwKO38RRoaON37YsXhCYw" : {
      "name" : "css-zzz1-ess-esn-2-1",
      "transport_address" : "10.0.0.247:9300",
      "host" : "10.0.0.247",
      "ip" : "10.0.0.247",
      "block_cache" : {
        "default" : {
          "type" : "memory",
          "block_cache_capacity" : 8192,
          "block_cache_blocksize" : 8192,
          "block_cache_size" : 12,
          "block_cache_hit" : 14,
          "block_cache_miss" : 0,
          "block_cache_eviction" : 0,
          "block_cache_store_fail" : 0
        }
      }
    },
    "obs_stats" : {
      "list" : {
        "obs_list_count" : 17,
        "obs_list_ms" : 265,
        "obs_list_avg_ms" : 15
      },
      "get_meta" : {
        "obs_get_meta_count" : 79,
        "obs_get_meta_ms" : 183,
        "obs_get_meta_avg_ms" : 2
      },
      "get_obj" : {
        "obs_get_obj_count" : 12,
        "obs_get_obj_ms" : 123,
        "obs_get_obj_avg_ms" : 10
      },
      "put_obj" : {
        "obs_put_obj_count" : 12,
        "obs_put_obj_ms" : 2451,
        "obs_put_obj_avg_ms" : 204
      },
      "obs_op_total" : {
        "obs_op_total_ms" : 3022,
        "obs_op_total_count" : 120,
        "obs_op_avg_ms" : 25
      }
    },
    "reader_cache" : {
      "hit_count" : 0,

```

```
"miss_count" : 1,
"load_success_count" : 1,
"load_exception_count" : 0,
"total_load_time" : 291194714,
"eviction_count" : 0
}
},
"73EDpEqoQES749umJqxOzQ" : {
"name" : "css-zzz1-ess-esn-3-1",
"transport_address" : "10.0.0.201:9300",
"host" : "10.0.0.201",
"ip" : "10.0.0.201",
"block_cache" : {
"default" : {
"type" : "memory",
"block_cache_capacity" : 8192,
"block_cache_blocksize" : 8192,
"block_cache_size" : 12,
"block_cache_hit" : 14,
"block_cache_miss" : 0,
"block_cache_eviction" : 0,
"block_cache_store_fail" : 0
}
},
"obs_stats" : {
"list" : {
"obs_list_count" : 17,
"obs_list_ms" : 309,
"obs_list_avg_ms" : 18
},
"get_meta" : {
"obs_get_meta_count" : 79,
"obs_get_meta_ms" : 216,
"obs_get_meta_avg_ms" : 2
},
"get_obj" : {
"obs_get_obj_count" : 12,
"obs_get_obj_ms" : 140,
"obs_get_obj_avg_ms" : 11
},
"put_obj" : {
"obs_put_obj_count" : 12,
"obs_put_obj_ms" : 1081,
"obs_put_obj_avg_ms" : 90
},
"obs_op_total" : {
"obs_op_total_ms" : 1746,
"obs_op_total_count" : 120,
"obs_op_avg_ms" : 14
}
},
"reader_cache" : {
"hit_count" : 0,
"miss_count" : 1,
"load_success_count" : 1,
"load_exception_count" : 0,
"total_load_time" : 367179751,
"eviction_count" : 0
}
},
"EF8WoLCUQbqJl1Pkqo9-OA" : {
"name" : "css-zzz1-ess-esn-1-1",
"transport_address" : "10.0.0.18:9300",
"host" : "10.0.0.18",
"ip" : "10.0.0.18",
"block_cache" : {
"default" : {
"type" : "memory",
"block_cache_capacity" : 8192,
```

```
"block_cache_blocksize" : 8192,
"block_cache_size" : 12,
"block_cache_hit" : 14,
"block_cache_miss" : 0,
"block_cache_eviction" : 0,
"block_cache_store_fail" : 0
}
},
"obs_stats" : {
  "list" : {
    "obs_list_count" : 17,
    "obs_list_ms" : 220,
    "obs_list_avg_ms" : 12
  },
  "get_meta" : {
    "obs_get_meta_count" : 79,
    "obs_get_meta_ms" : 139,
    "obs_get_meta_avg_ms" : 1
  },
  "get_obj" : {
    "obs_get_obj_count" : 12,
    "obs_get_obj_ms" : 82,
    "obs_get_obj_avg_ms" : 6
  },
  "put_obj" : {
    "obs_put_obj_count" : 12,
    "obs_put_obj_ms" : 879,
    "obs_put_obj_avg_ms" : 73
  },
  "obs_op_total" : {
    "obs_op_total_ms" : 1320,
    "obs_op_total_count" : 120,
    "obs_op_avg_ms" : 11
  }
},
"reader_cache" : {
  "hit_count" : 0,
  "miss_count" : 1,
  "load_success_count" : 1,
  "load_exception_count" : 0,
  "total_load_time" : 235706838,
  "eviction_count" : 0
}
}
}
```

7. Run the following command to reset the cache status:
POST `_frozen_stats/reset`

Information similar to the following is displayed:

```
{
  "_nodes" : {
    "total" : 1,
    "successful" : 1,
    "failed" : 0
  },
  "cluster_name" : "Es-0325-007_01",
  "nodes" : {
    "mqTdk2YRSPyOSXfesREFSg" : {
      "result" : "ok"
    }
  }
}
```

NOTE

This command is used to debug performance issues. If you reset the cache status and run this command, you can check the cache command status. You do not need to run this command during service running.

- Run the following command to check all the frozen indexes:

```
GET _cat/freeze_indices
```

Information similar to the following is displayed:

```
green open data2 0bNtxWDtRbOSkS4JYaUgMQ 3 0 5 0 7.9kb 7.9kb
green open data3 oYMLvw31QnyasqUNuyP6RA 3 0 51 0 23.5kb 23.5kb
```

 **NOTE**

The parameters and return values of this command are the same as those of `_cat/indices` of Elasticsearch.

11.1.3 Configuring Cache

After data is dumped to OBS, some data is cached to reduce access to OBS and improve Elasticsearch query performance. Data that is requested for the first time is obtained from OBS. The obtained data is cached in the memory. In subsequent queries, the system searches for data in the cache first. Data can be cached in memory or files.

Elasticsearch accesses different files in different modes. The cache system supports multi-level cache and uses blocks of different sizes to cache different files. For example, a large number of small blocks are used to cache .fdx and .tip files, and a small number of large blocks are used to cache .fdt files.

Table 11-8 Cache configurations

Parameter	Type	Description
low_cost.obs.blockcache.names	Array	The cache system supports multi-level cache for data of different access granularities. This configuration lists the names of all caches. If this parameter is not set, the system has a cache named default . To customize the configuration, ensure there is a cache named default . Default value: default
low_cost.obs.blockcache.<NAME>.type	ENUM	Cache type, which can be memory or file . If it is set to memory , certain memory will be occupied. If it is set to file , cache will be stored in disks. You are advised to use ultra-high I/O disks to improve cache performance. Default value: memory
low_cost.obs.blockcache.<NAME>.blockshift	Integer	Size of each block in the cache. Its value is the number of bytes shifted left. For example, if this parameter is set to 16 , the block size is 2¹⁶ bytes, that is, 65536 bytes (64 KB). Default value: 13 (8 KB)
low_cost.obs.blockcache.<NAME>.bank.count	Integer	Number of cache partitions. Default value: 1

Parameter	Type	Description
low_cost.obs.blockcache.<NAME>.number.blocks.perbank	Integer	Number of blocks included in each cache partition. Default value: 8192
low_cost.obs.blockcache.<NAME>.exclude.file.types	Array	Extensions of files that are not cached. If the extensions of certain files are neither in the exclude list nor in the include list, they are stored in the default cache.
low_cost.obs.blockcache.<NAME>.file.types	Array	Extensions of cached files. If the extensions of certain files are neither in the exclude list nor in the include list, they are stored in the default cache.

The following is a common cache configuration. It uses two levels of caches, **default** and **large**. The **default** cache uses 64 KB blocks and has a total of 30 x 4096 blocks. It is used to cache files except .fdt files. The **large** cache uses 2 MB blocks and contains 5 x 1000 blocks. It is used to cache .fdx, .dvd, and .tip files.

```
low_cost.obs.blockcache.names: ["default", "large"]
low_cost.obs.blockcache.default.type: file
low_cost.obs.blockcache.default.blockshift: 16
low_cost.obs.blockcache.default.number.blocks.perbank: 4096
low_cost.obs.blockcache.default.bank.count: 30
low_cost.obs.blockcache.default.exclude.file.types: ["fdt"]

low_cost.obs.blockcache.large.type: file
low_cost.obs.blockcache.large.blockshift: 21
low_cost.obs.blockcache.large.number.blocks.perbank: 1000
low_cost.obs.blockcache.large.bank.count: 5
low_cost.obs.blockcache.large.file.types: ["fdx", "dvd", "tip"]
```

Table 11-9 Other parameters

Parameter	Type	Description
index.frozen.obs.max_bytes_per_sec	String	Maximum rate of uploading files to OBS during freezing. It takes effect immediately after you complete configuration. Default value: 150MB
low_cost.obs.index.upload.threshold.use.multipart	String	If the file size exceeds the value of this parameter during freezing, the multipart upload function of OBS is used. Default value: 1GB

Parameter	Type	Description
index.frozen.reader.cache.expire.duration.seconds	Integer	Timeout duration. To reduce the heap memory occupied by frozen indexes, the reader caches data for a period of time after the index shard is started, and stops caching after it times out. Default value: 300s
index.frozen.reader.cache.max.size	Integer	Maximum cache size. Default value: 100

11.1.4 Enhanced Cold Data Query Performance

Context

When you query data on the **Discover** page of Kibana for the first time, all data needs to be obtained from OBS because there is no cache. If a large number of documents are returned, it takes a long time to obtain the corresponding time fields and file metadata from OBS. To accelerate queries the first time they run on the **Discover** page, you can cache data locally.

Prerequisites

This feature is available in clusters of versions 7.6.2 and 7.10.2 created after February 2023.

API for Querying Cold Data from Local Cache

This API can be used to query the cold data from local cache.

Example request:

```
GET /_frozen_stats/local_cache
GET /_frozen_stats/local_cache/{nodeld}
```

Response example:

```
{
  "_nodes" : {
    "total" : 1,
    "successful" : 1,
    "failed" : 0
  },
  "cluster_name" : "elasticsearch",
  "nodes" : {
    "6by3lPy1R3m55Dcq3liK8Q" : {
      "name" : "node-1",
      "transport_address" : "127.0.0.1:9300",
      "host" : "127.0.0.1",
      "ip" : "127.0.0.1",
      "local_cache" : {
        "get_stats" : {
          "get_total_count" : 562,
          //Total number of times data was retrieved from the local
          cold data cache.
        }
      }
    }
  }
}
```


Configuring Parameters

Configuration Item	Type	Unit	Value Range	Scope	Can Be Dynamically Modified	Description
low_cost.local_cache.max.capacity	Integer	-	The value ranges from 10 to 5000. The default value is 500 .	node	Yes	<p>Maximum number of available cold data caches on a node. Each shard corresponds to a cache object.</p> <p>NOTE</p> <ul style="list-style-type: none"> If the heap memory usage remains high, decrease the value. If the value of load_overflow_count keeps increasing rapidly, increase the value.
index.low_cost.local_cache.threshold	Integer	%	The value ranges from 0 to 100. The default value is 50 .	index	Yes	<p>Threshold for enabling the local cache of cold data.</p> <p>NOTE</p> <ul style="list-style-type: none"> If the percentage of date fields is less than the value of this parameter, the cold data of the date type will be cached locally. Otherwise, this parameter is not used. If the date fields of the current index occupy most of the data volume of the current index, you are not advised to use this function.

Configuration Item	Type	Unit	Value Range	Scope	Can Be Dynamically Modified	Description
index.low_cost.local_cache.evict_time	String	Days	The value ranges from 1d to 365d. The default value is 30d .	index	Yes	<p>Wait time before cold data is deleted from local cache. The value is determined based on index.frozen_date (time when the freezing is successful).</p> <p>NOTE</p> <ul style="list-style-type: none"> For indexes that have been frozen in old clusters and do not have index.frozen_date specified, the value of this parameter is determined based on the index creation time. You are advised to adjust the deletion time based on the disk usage to avoid high disk usage.

Modifying Parameters

- Run the following command to modify **low_cost.local_cache.max.capacity**:

```
PUT _cluster/settings
{
  "persistent": {
    "low_cost.local_cache.max.capacity":1000
  }
}
```

- Run the following command to modify **index.low_cost.local_cache.threshold**:

```
PUT es_write_pref2-00000000021/_settings
{
  "index.low_cost.local_cache.threshold":20
}
```

- Run the following command to modify **index.low_cost.local_cache.evict_time**:

```
PUT es_write_pref2-00000000021/_settings
{
  "index.low_cost.local_cache.evict_time":"7d"
}
```

11.1.5 Monitoring OBS Operations

To clearly display the operations of the storage and compute decoupling plugin in OBS, the real-time OBS rate metric is added to CSS and recorded in the system index.

Prerequisite

This feature is available in clusters of versions 7.6.2 and 7.10.2 created after March 2023.

Description

- The [GET_frozen_stats/obs_rate](#) API is used to query the real-time rate of OBS operations.
- The system index [.freeze_obs_rate-YYYY.mm.dd](#) is added to store the real-time OBS operation rate and OBS operation data, helping you monitor the OBS operations.
- The [low_cost.obs_rate_index.evict_time](#) parameter is added to control the storage duration of the [.freeze_obs_rate-YYYY.mm.dd](#) index

GET_frozen_stats/obs_rate API

- Calculation method: The average OBS operation rate in the last 5 seconds is calculated every 5 seconds.

- Example request:

```
GET_frozen_stats/obs_rate
GET_frozen_stats/obs_rate/{nodeId}
```

{nodeId} indicates the ID of the node whose OBS operation rate you want to query.

- Example response:

```
{
  "_nodes" : {
    "total" : 1,
    "successful" : 1,
    "failed" : 0
  },
  "cluster_name" : "elasticsearch",
  "nodes" : {
    "df1DvcSwTJ-fki1LT2zE3A" : {
      "name" : "node-1",
      "transport_address" : "127.0.0.1:9300",
      "host" : "127.0.0.1",
      "ip" : "127.0.0.1",
      "update_time" : 1671777600482,           // Time when the current statistics are
      "obs_rate" : {                          updated.
        "list_op_rate" : 0.0,                 // Rate of OBS list operations. Unit: times/s.
        "get_meta_op_rate" : 0.0,            // Rate of OBS get meta operations. Unit: times/s.
        "get_obj_op_rate" : 0.0,            // Rate of OBS get operations. Unit: times/s.
        "put_op_rate" : 0.0,                // Rate of OBS put operations. Unit: times/s.
        "obs_total_op_rate" : 0.0,          // Rate of all OBS operations. The unit is times/s.
        "obs_upload_rate" : "0.0 MB/s",     // Data upload rate of OBS, in MB/s.
        "obs_download_rate" : "0.0 MB/s"   // Data download rate of OBS, in MB/s.
      }
    }
  }
}
```

System Index

- System index name: [.freeze_obs_rate-YYYY.mm.dd](#).
- Example: [.freeze_obs_rate-2023.01.23](#)

 NOTE

The default retention period of indexes is 30 days.

Configuration Item

Configuration Item	Type	Scope	Can Be Dynamically Modified	Description
low_cost.obs_rate_index.evict_time	String	node	Yes	The retention period of the .freeze_obs_rate-YYYY.mm.dd index. <ul style="list-style-type: none"> Value range: 1d to 365d Default value: 30d Unit: day

For example, run the following command to modify the retention period of the **.freeze_obs_rate-YYYY.mm.dd** index:

```
PUT _cluster/settings
{
  "persistent": {
    "low_cost.obs_rate_index.evict_time": "7d"
  }
}
```

11.2 Flow Control

11.2.1 Context

Feature Description

CSS can control traffic at the node level. You can configure the blacklist and whitelist, the maximum concurrent HTTP connections, and the maximum HTTP connections for a node. You can also configure the maximum heap memory used by specific request paths, the maximum CPU usage, and block access in one click, and collect statistics on node access IP addresses and URIs. Each function has an independent control switch, which is disabled by default. To restore default values of parameters, set them to **null**.

If flow control is enabled, requests will be blocked at the entry, which relieves the cluster pressure in high-concurrency scenario and avoids unavailability issues.

- **HTTP/HTTPS flow control:**
 - You can control client IP address access by setting IP addresses and subnets in HTTP/HTTPS blacklist or whitelist. If an IP address is in the

blacklist, the client is disconnected and all its request are rejected. Whitelist rules take precedence over blacklist rules. If a client IP address exists in both the blacklist and whitelist, the client request will not be rejected.

- HTTP/HTTPS concurrent connection flow control limits the total number of HTTP connections to a node per second.
- HTTP/HTTPS new connection flow control limits the number of new connections to a node.
- Memory flow control limits request paths based on the node heap memory. You can configure memory flow control whitelist, global memory usage threshold, and heap memory threshold for a single path. Global memory flow control threshold takes precedence over the memory threshold of a single path. Paths in the whitelist will not be blocked in memory flow control.
- You can configure the global path whitelist for flow control as required when you need to use custom plug-ins.
- Request sampling can record the number of access requests from client IP addresses and the request paths of sampled users. Based on the statistics, you can identify the access traffic of client IP addresses and analyze the access traffic of request paths.
- Flow control provides an independent API for viewing traffic statistics and records the number of times the API is triggered. You can evaluate the flow control threshold and analyze the cluster load based on the statistics.
- Access logs record the URLs and bodies of HTTP/HTTPS requests received by nodes within a period of time. You can analyze the current traffic pressure based on the access logs.
- You can configure the node CPU usage threshold to limit the accessed traffic on a single node.
- One-click access blocking can block all the access traffic of a node, excluding the traffic from Kibana and Elasticsearch monitor APIs.

Constraints

- Currently, only versions 7.6.2 and 7.10.2 support the flow control feature.
- Flow control may affect the performance of some nodes.
- If flow control is enabled, user requests that exceed the flow control threshold will be rejected.
- Memory flow control and CPU flow control are based on request paths. The length and number of paths cannot be too large, or the cluster performance will be affected.

11.2.2 HTTP/HTTPS Flow Control

Context

You can run commands in Kibana to enable or disable HTTP/HTTPS flow control for your cluster. The command parameters are as follows.

Table 11-10 HTTP/HTTPS flow control parameters

Parameter	Type	Description
flowcontrol.http.enabled	Boolean	Whether to enable HTTP/HTTPS flow control. This function is disabled by default. Enabling it may affect node access performance. Value: true or false Default value: false
flowcontrol.http.allow	List<String>	IP address whitelist. It can contain multiple IP addresses and masks, or an IP address list. Use commas (,) to separate multiple values. Example: <i>xx.xx.xx.xx/24,xx.xx.xx.xx/24</i> , or <i>xx.xx.xx.xx.xx,xx.xx.xx</i> . The default value is null.
flowcontrol.http.deny	List<String>	IP address blacklist. Multiple IP addresses and masks or an IP address list can be configured. Use commas (,) to separate multiple IP addresses and masks. The default value is null.
flowcontrol.http.concurrent	Integer	Maximum concurrent HTTP/HTTPS connections. Default value: Number of available cores on a node x 400
flowcontrol.http.newconnect	Integer	Maximum new connections that can be created for HTTP/HTTPS requests per second. Default value: Number of available cores on a node x 200
flowcontrol.http.warmup_period	Integer	Time required for the HTTP/HTTPS connection setup speed to reach the maximum. If flowcontrol.http.newconnect is set to 100 and flowcontrol.http.warmup_period is set to 5000ms , it indicates the system can set up 100 connections per second in 5 seconds. Value range: 0-10000 Unit: ms Default value: 0

Procedure

1. Log in to the CSS management console.
2. Choose **Clusters** in the navigation pane. On the **Clusters** page, locate the target cluster and click **Access Kibana** in the **Operation** column.
3. In the navigation pane on the left, choose **Dev Tools** and run commands to enable or disable HTTP/HTTPS flow control.

- Enabling HTTP/HTTPS flow control for a node

```
PUT /_cluster/settings
{
  "persistent": {
    "flowcontrol.http.enabled": true,
    "flowcontrol.http.allow": ["192.168.0.1/24", "192.168.2.1/24"],
    "flowcontrol.http.deny": "192.168.1.1/24",
    "flowcontrol.http.concurrent": 1000,
    "flowcontrol.http.newconnect": 1000,
    "flowcontrol.http.warmup_period": 0
  }
}
```

NOTE

If all parameters are set to **null**, they will be restored to default values.

- Disabling HTTP/HTTPS flow control for a node

```
PUT /_cluster/settings
{
  "persistent": {
    "flowcontrol.http.enabled": false
  }
}
```

11.2.3 Memory Flow Control

Context

Elasticsearch provides a circuit breaker, which will terminate requests if the memory usage exceeds its threshold. However, Elasticsearch does not check the heap memory usage when an API is called, and does not allow users to configure the threshold for a single request. In this case, memory usage can only be calculated during request processing, which may lead to frequent circuit breaking and cannot avoid heap memory waste. To solve this problem, CSS checks the heap memory usage when receiving REST requests, blocking excess API requests and protecting nodes. You can configure global memory flow control, or configure the request path and heap memory threshold for a specific request path. Before a request is processed, the system checks the configured heap memory threshold. If the threshold is exceeded, the request path will be blocked.

NOTE

- Memory flow control may affect request processing performance.
- If the memory flow control is enabled, some Kibana search requests may fail.
- If memory flow control is enabled in Elasticsearch 5.5.1, `_mget` requests will be blocked and Kibana access will be abnormal. You can add `_mget` requests to the request whitelist to avoid this problem.

The following table describes memory flow control parameters.

Table 11-11 Memory flow control parameters

Parameter	Type	Description
flowcontrol.memory.enabled	Boolean	Whether to enable memory flow control. This function is disabled by default. Enabling memory flow control may slightly affect node performance. Value: true or false Default value: false
flowcontrol.memory.allow_path	List<String >	Request path whitelist for memory flow control. Whitelisted paths are blocked in memory flow control. Wildcard characters are supported. By default, query APIs controlled by the cluster are not blocked in memory flow control. This prevents the failure to query cluster information when the memory usage reaches the threshold. Example: <ul style="list-style-type: none"> • "flowcontrol.memory.allow_path": "/index/_search", • "flowcontrol.memory.allow_path": "/index*/_search", • "flowcontrol.memory.allow_path": ["/index/_search", "/index1/_bulk"], A maximum of 10 paths can be configured. A path can contain up to 32 characters. The default value is null.
flowcontrol.memory.heap_limit	String	Maximum global heap memory usage of a node. The value cannot be less than 10% of the heap memory. Value range: 10%–100% Default value: 90%

Parameter	Type	Description
flowcontrol.memory.*.filter_path	String	<p>Paths under memory flow control. The default value is **, indicating all paths. If flowcontrol.memory.heap_limit is configured and flowcontrol.memory.*.filter_path is not, it indicates that all the paths, except those in the whitelist, are under control. The whitelist takes precedence over the single-path rule. If a path is specified in both flowcontrol.memory.allow_path and flowcontrol.memory.*.filter_path, the requests from the path will be allowed.</p> <p>For example, if flowcontrol.memory.allow_path and flowcontrol.memory.*.filter_path are both set to abc/_search, then abc/_search will not be under flow control.</p> <p>Maximum length: 32 characters</p>
flowcontrol.memory.*.heap_limit	String	<p>Heap memory usage threshold of request paths. If the heap memory usage exceeds the threshold, flow control will be triggered.</p> <p>Value range: 0–100%</p> <p>Default value: 90%</p>

Procedure

1. Log in to the CSS management console.
2. Choose **Clusters** in the navigation pane. On the **Clusters** page, locate the target cluster and click **Access Kibana** in the **Operation** column.
3. In the navigation pane on the left, choose **Dev Tools** and run commands to enable or disable memory flow control.

- Enabling memory flow control

```
PUT /_cluster/settings
{
  "persistent": {
    "flowcontrol.memory.enabled": true,
    "flowcontrol.memory.allow_path": "/index/_search",
    "flowcontrol.memory.heap_limit": "85%"
  }
}
```

- Enabling memory flow control for a request path

Configure the heap memory usage threshold for a request path. You can configure the priorities of such threshold rules.

```
PUT /_cluster/settings
{
```

```
"persistent": {
  "flowcontrol.memory.enabled": true,
  "flowcontrol.memory": {
    "flowcontrol_search": {
      "filter_path": "index1/_search",
      "heap_limit": "50%"
    },
    "flowcontrol_bulk": {
      "filter_path": "index*/_bulk",
      "heap_limit": "50%"
    }
  }
}
```

- Deleting the memory flow control configuration of a request path

PUT /_cluster/settings

```
{
  "persistent": {
    "flowcontrol.memory.enabled": true,
    "flowcontrol.memory": {
      "flowcontrol_search": {
        "filter_path": null,
        "heap_limit": null
      }
    }
  }
}
```

- Disabling cluster memory flow control

PUT /_cluster/settings

```
{
  "persistent": {
    "flowcontrol.memory.enabled": false
  }
}
```

11.2.4 Global Path Whitelist for Flow Control

Context

The following table describes the global path whitelist parameters for flow control.

Table 11-12 Global path whitelist parameters for flow control

Parameter	Type	Description
flowcontrol.path.white_list	List<String>	<p>Paths that are not under flow control. These paths are not affected by memory flow control, CPU flow control, or one-click blocking; but are under IP address-based flow control.</p> <p>A maximum of 10 paths can be configured. A path can contain up to 32 characters.</p> <p>The default value is null.</p> <p>NOTE You are advised not to configure this parameter, unless required by plug-ins.</p>

Procedure

1. Log in to the CSS management console.
2. Choose **Clusters** in the navigation pane. On the **Clusters** page, locate the target cluster and click **Access Kibana** in the **Operation** column.
3. In the navigation tree on the left, choose **Dev Tools**. Run the following command to configure the global path whitelist for flow control:

```
PUT _cluster/settings
{
  "persistent": {
    "flowcontrol.path.white_list": "xxxx"
  }
}
```

11.2.5 Request Sampling

Context

Request sampling can record the access IP addresses, the number of accessed nodes, request paths, request URLs, and request bodies, which can be used to trace the IP addresses and paths of clients that have sent a large number of access requests.

The following table describes request sampling parameters.

Table 11-13 Request sampling parameters

Parameter	Type	Description
flowcontrol.statics.enabled	Boolean	Whether to enable request sampling. Request sampling may affect node performance. Value: true or false Default value: false
flowcontrol.statics.threshold	Integer	Number of recent access requests whose statistics are collected. The value 100 indicates that statistics will be collected on the 100 IP addresses and 100 URLs that are most frequently accessed. Minimum value: 10 Maximum value: 1000 Default value: 100
flowcontrol.statics.sample_frequency	Integer	Path sampling frequency. If this parameter is set to 100 , samples are collected from every 100 requests. Minimum value: 50 Default value: 100

NOTE

- The IP address statistics and URL sampling statistics are cached based on their access time. If the cache space reaches the threshold (**flowcontrol.statics.threshold**), the records of the earliest access will be deleted.
- In URL sampling, an access path is uniquely identified by its URL hash.

Procedure

1. Log in to the CSS management console.
2. Choose **Clusters** in the navigation pane. On the **Clusters** page, locate the target cluster and click **Access Kibana** in the **Operation** column.
3. In the navigation pane on the left, choose **Dev Tools** and run commands to enable or disable sampling.

– Enabling sampling

```
PUT /_cluster/settings
{
  "persistent": {
    "flowcontrol.statics.enabled": true,
    "flowcontrol.statics.threshold": 100,
    "flowcontrol.statics.sample_frequency": 50
  }
}
```

– Disabling sampling

```
PUT /_cluster/settings
{
  "persistent": {
    "flowcontrol.statics.enabled": false
  }
}
```

11.2.6 Flow Control

Flow control can be implemented via an independent API.

1. Log in to the CSS management console.
2. Choose **Clusters** in the navigation pane. On the **Clusters** page, locate the target cluster and click **Access Kibana** in the **Operation** column.
3. In the navigation pane on the left, choose **Dev Tools** and run the commands to query traffic control information.

– Check the traffic control status of all nodes.

```
GET /_nodes/stats/filter
```

– View the traffic control status of a specific node.

```
GET /_nodes/{nodeId}/stats/filter
```

{nodeId} indicates the ID of the node you want to check.

Example response:

```
{
  "_nodes" : {
    "total" : 1,
    "successful" : 1,
    "failed" : 0
  },
  "cluster_name" : "css-flowcontroller",
  "nodes" : {
    "E1BRNCMbTj6L1C-Wke-Dnw" : {
      "name" : "css-flowcontroller-ess-esn-1-1",
      "host" : "10.0.0.133",
```



```

"timestamp" : 1613979513747,
"flow_control" : {
  "transport" : {
    "concurrent_req" : 0,
    "rejected_concurrent" : 0,
    "rejected_new" : 0,
    "rejected_deny" : 0
  },
  "http" : {
    "concurrent_req" : 0,
    "rejected_concurrent" : 0,
    "rejected_new" : 0,
    "rejected_deny" : 0
  },
  "memory" : {
    "memory_allow" : 41,
    "memory_rejected" : 0
  },
  "cpu" : {
    "rejected_cpu" : 0
  }
},
"ip_address" : [
  {
    "ip" : "/10.0.0.198",
    "count" : 453
  },
  {
    "ip" : "/198.19.49.1",
    "count" : 42
  }
],
"url_sample" : [
  {
    "url" : "/*/_search?pretty=true",
    "method" : "GET",
    "remote_address" : "/10.0.0.198:16763",
    "count" : 1
  }
]
}
}

```

In the response, the information of each node is separated. The **http** field records the numbers of concurrent connections and new connections. The **memory** records memory flow control statistics. The **ip_address** field records the recent client IP addresses that are accessed most recently. The **url_sample** field records the recent URLs that are requested most frequently. The **cpu** field records CPU flow control statistics.

Table 11-14 Response parameters

Parameter	Description
concurrent_req	Number of TCP connections of a node, which is recorded no matter whether flow control is enabled. This value is similar to the value of current_open of the GET /_nodes/stats/http API but is smaller, because whitelisted IP addresses and internal node IP addresses are not counted.
rejected_concurrent	Number of concurrent connections rejected during HTTP flow control. This value is not cleared when HTTP flow control is disabled.

Parameter	Description
rejected_new	Number of new connections rejected during HTTP flow control. This value is not cleared when HTTP flow control is disabled.
rejected_deny	Number of requests rejected based on the blacklist during HTTP flow control. This value is not cleared when HTTP flow control is disabled.
memory_allow	Number of allowed requests during memory flow control. This parameter takes effect when memory flow control is enabled, and its value is not cleared after memory flow control is disabled. The requests from the paths in the allow_path whitelist are not recorded. If allow_path is set to **, no requests are recorded.
memory_rejected	Number of rejected requests during memory flow control. This parameter takes effect when memory flow control is enabled, and its value is not cleared after memory flow control is disabled. The requests from the paths in the allow_path whitelist are not recorded. If allow_path is set to **, no requests are recorded.
rejected_cpu	Number of requests rejected when the CPU flow control threshold is exceeded. This parameter takes effect when CPU flow control is enabled, and its value is not cleared after CPU flow control is disabled.
ip_address	IP addresses and the number of requests. For details, see Table 11-15 .
url_sample	Request path sampling. The number of URLs of a request are collected based on the configured time and sampling interval. For details, see Table 11-16 .

Table 11-15 ip_address

Parameter	Description
ip	Source IP address for accessing the node.
method	Number of access requests from an IP address.

Table 11-16 url_sample

Parameter	Description
url	Request URL
method	Method corresponding to the request path
remote_address	Source IP address and port number of the request
count	How many times a path is sampled

11.2.7 Access Logs

Context

You can check access logs in either of the following ways:

- Enable and check access logs via an independent API. Configure the API parameters to record the access log time and size. The access log content is returned through a REST API.
- Print access logs. Your access logs are printed as files in backend logs.

Enabling the access log function may affect cluster performance.

The following table describes access log parameters.

Table 11-17 Access log parameters

Parameter	Type	Description
duration_limit	String	Duration recorded in an access log. Value range: 10 to 120 Unit: s Default value: 30
capacity_limit	String	Size of an access log. After access logging is enabled, the size of recorded requests is checked. If the size exceeds the value of this parameter, the access logging stops. Value range: 1 to 5 Unit: MB Default value: 1

NOTE

Access logging stops if either **duration_limit** or **capacity_limit** reaches the threshold.

Procedure

1. Log in to the CSS management console.
2. Choose **Clusters** in the navigation pane. On the **Clusters** page, locate the target cluster and click **Access Kibana** in the **Operation** column.
3. In the navigation pane on the left, choose **Dev Tools** and run commands to enable or disable access logs.
 - Enabling access logs for all nodes in a cluster
PUT `/_access_log?duration_limit=30s&capacity_limit=1mb`
 - Enabling access logs for a node in a cluster
PUT `/_access_log/{nodeId}?duration_limit=30s&capacity_limit=1mb`
{nodeId} indicates the ID of the node where you want to enable access logs.
4. Use APIs to check access logs.
 - API for checking the access logs of all nodes in a cluster
GET `/_access_log`
 - API for checking the access logs of a node in a cluster
GET `/_access_log/{nodeId}`
{nodeId} indicates the ID of the node where you want to enable access logs.

Example response:

```
{
  "_nodes": {
    "total": 1,
    "successful": 1,
    "failed": 0
  },
  "cluster_name": "css-flowcontroller",
  "nodes": {
    "8x-ZHu-wTemBQwpcGivFKg": {
      "name": "css-flowcontroller-ess-esn-1-1",
      "host": "10.0.0.98",
      "count": 2,
      "access": [
        {
          "time": "2021-02-23 02:09:50",
          "remote_address": "/10.0.0.98:28191",
          "url": "/_access/security/log?pretty",
          "method": "GET",
          "content": ""
        },
        {
          "time": "2021-02-23 02:09:52",
          "remote_address": "/10.0.0.98:28193",
          "url": "/_access/security/log?pretty",
          "method": "GET",
          "content": ""
        }
      ]
    }
  }
}
```

Table 11-18 Response parameters

Parameter	Description
name	Node name

Parameter	Description
host	Node IP address
count	Number of node access requests in a statistical period
access	Details about node access requests in a statistical period For details, see Table 11-19 .

Table 11-19 access

Parameter	Description
time	Request time
remote_address	Source IP address and port number of the request
url	Original URL of the request
method	Method corresponding to the request path
content	Request content

5. Enable or disable the access log function.

All user access operation can be logged. By default, logs are recorded in the **access_log.log** file in the background. The maximum size of a log file is 250 MB, and there can be a maximum of five log files. You can back up access log files to OBS.

- Enabling access logs

```
PUT /_cluster/settings
{
  "persistent": {
    "flowcontrol.accesslog.enabled": true
  }
}
```

- Disabling access logs

```
PUT /_cluster/settings
{
  "persistent": {
    "flowcontrol.accesslog.enabled": false
  }
}
```

11.2.8 CPU Flow Control

Context

CPU flow control can be implemented based on the CPU usage of a node.

You can configure the CPU usage threshold of a node to prevent the node from breaking down due to heavy traffic. You can determine the CPU usage threshold based on the traffic threshold. If the CPU usage of a node exceeds the configured threshold, CPU flow control discards excess node requests to protect the cluster.

Traffic within the node or passing through Elasticsearch monitoring APIs are not affected.

The following table describes CPU flow control parameters.

Table 11-20 CPU flow control parameters

Parameter	Type	Description
flowcontrol.cpu.enabled	Boolean	Whether to enable CPU flow control. If this function is enabled, the node access performance may be affected. Value: true or false Default value: false
flowcontrol.cpu.percent_limit	Integer	Maximum CPU usage of a node. Value range: 0-100 Default value: 90
flowcontrol.cpu.allow_path	List	Path whitelist for CPU flow control. The paths specified in the allow_path whitelist are not under CPU flow control. The default value is null. A path can contain up to 32 characters. A maximum of 10 request paths can be configured. Wildcard characters are supported. For example, if this parameter is set to auto_*/_search , all the search requests of the indexes prefixed with auto_ are not under the flow control.
flowcontrol.cpu.*.filter_path	String	Paths under CPU flow control. Maximum length: 32 characters Example: "flowcontrol.cpu.search.filter_path": "/index/_search", "flowcontrol.cpu.search.limit": 60, The default value is ** , indicating all paths. If limit is configured and filter_path is not, it indicates that all the paths, except those in the whitelist, are under control. The whitelist takes precedence over the single-path rule. If a path is specified in both allow_path and filter_path , the requests from the path will be allowed. For example, if both filter_path and allow_path both set to abc/_search , then abc/_search will not be under flow control.

Parameter	Type	Description
flowcontrol.cpu.*.limit	Integer	CPU threshold of request paths. If the CPU usage exceeds the threshold, flow control will be triggered. Value range: 0-100 Default value: 90

Procedure

1. Log in to the CSS management console.
2. Choose **Clusters** in the navigation pane. On the **Clusters** page, locate the target cluster and click **Access Kibana** in the **Operation** column.
3. In the navigation pane on the left, choose **Dev Tools** and run commands to enable or disable memory flow control.

- Enabling CPU flow control

```
PUT /_cluster/settings
{
  "persistent": {
    "flowcontrol.cpu.enabled": true,
    "flowcontrol.cpu.percent_limit": 80,
    "flowcontrol.cpu.allow_path": ["index/_search"]
  }
}
```

- Disabling CPU flow control

```
PUT /_cluster/settings
{
  "persistent": {
    "flowcontrol.cpu.enabled": false
  }
}
```

11.2.9 One-click Traffic Blocking

You can block all traffic in one click, except the traffic that passes through O&M APIs, to handle unexpected traffic burst and quickly recover your cluster.

1. Log in to the CSS management console.
2. Choose **Clusters** in the navigation pane. On the **Clusters** page, locate the target cluster and click **Access Kibana** in the **Operation** column.
3. In the navigation pane on the left, choose **Dev Tools** and run commands to enable or disable one-click traffic blocking.

- Enabling one-click traffic blocking

```
PUT /_cluster/settings
{
  "persistent": {
    "flowcontrol.break.enabled": true
  }
}
```

- Disabling one-click traffic blocking

```
PUT /_cluster/settings
{
  "persistent": {
    "flowcontrol.break.enabled": false
  }
}
```

```
}  
}
```

11.3 Large Query Isolation

11.3.1 Context

The large query isolation feature allows you to separately manage large queries. You can isolate query requests that consume a large amount of memory or take a long period of time. If the heap memory usage of a node is too high, the interrupt control program will be triggered. The program will interrupt a large query based on the policies you configured and cancel the running query tasks of the query.

You can also configure a global query timeout duration. Long queries will be intercepted.

NOTE

Currently, only versions 7.6.2 and 7.10.2 support large query isolation.

11.3.2 Procedure

The large query isolation and global timeout features are disabled by default. If you enable them, the configuration will take effect immediately. Perform the following steps to configure the features:

1. Log in to the CSS management console.
2. Choose **Clusters** in the navigation pane. On the **Clusters** page, locate the target cluster, and click **Access Kibana** in the **Operation** column.
3. In the navigation pane of Kibana on the left, choose **Dev Tools**. Run the following command to enable large query isolation and global timeout features:

```
PUT _cluster/settings  
{  
  "persistent": {  
    "search.isolator.enabled": true,  
    "search.isolator.time.enabled": true  
  }  
}
```

The two features each has an independent switch and the following parameters.

Table 11-21 Parameters for large query isolation and global timeout duration

Switch	Parameter	Description
search.isolator.enabled	search.isolator.memory.task.limit search.isolator.time.management	Thresholds of a shard query task. A query task exceeding one of these thresholds is regarded as a large query task.

Switch	Parameter	Description
	search.isolator.memory.pool.limit search.isolator.memory.heap.limit search.isolator.count.limit	Resource usage thresholds in the isolation pool. If the resource usage of a query task exceeds one of these thresholds, the task will be intercepted. NOTE search.isolator.memory.heap.limit defines the limit on the heap memory consumed by write, query, and other operations of a node. If the limit is exceeded, large query tasks in the isolation pool will be interrupted.
	search.isolator.strategy search.isolator.strategy.ratio	Policy for selecting a query task in the isolation pool.
search.isolator.time.enabled	search.isolator.time.limit	Global timeout interval of query tasks.

4. Configure the large query isolation and global timeout duration separately.
 - Configure the thresholds of a shard query task. A query task exceeding one of these thresholds is regarded as a large query task.

```
PUT _cluster/settings
{
  "persistent": {
    "search.isolator.memory.task.limit": "50MB",
    "search.isolator.time.management": "10s"
  }
}
```

Table 11-22 Parameter description

Parameter	Data Type	Description
search.isolator.memory.task.limit	String	Threshold of the memory requested by a query task to perform aggregation or other operations. If the requested memory exceeds the threshold, the task will be isolated and observed. Value range: 0b to the maximum heap memory of a node Default value: 50MB NOTE You can run the following command to query the current heap memory and the maximum heap memory of a cluster: GET _cat/nodes?&h=id,ip,port,r,ramPercent,ramCurrent,heapMax,heapCurrent

Parameter	Data Type	Description
search.isolator.time.management	String	Threshold of the duration of a query. (started when cluster resources are used for query). If the duration of a query exceeds the threshold, it will be isolated and observed. Value range: \geq 0ms Default value: 10s

- Configure the resource usage thresholds in the isolation pool. If the resource usage of a query task exceeds one of these thresholds, the task will be intercepted.

PUT _cluster/settings

```
{
  "persistent": {
    "search.isolator.memory.pool.limit": "50%",
    "search.isolator.memory.heap.limit": "90%",
    "search.isolator.count.limit": 1000
  }
}
```

Table 11-23 Parameter description

Parameter	Data Type	Description
search.isolator.memory.pool.limit	String	Threshold of the heap memory percentage of the current node. If the total memory requested by large query tasks in the isolation pool exceeds the threshold, the interrupt control program will be triggered to cancel one of the tasks. Value range: 0.0 to 100.0% Default value: 50%
search.isolator.memory.heap.limit	String	Heap memory threshold of the current node. If the heap memory of the node exceeds the threshold, the interrupt control program will be triggered to cancel a large query task in the isolation pool. Value range: 0.0 to 100.0% Default value: 90%

Parameter	Data Type	Description
search.isolator.count.limit	Integer	<p>Threshold of the number of large query tasks in the current node isolation pool. If the number of observed query tasks exceeds the threshold, the interrupt control program will be triggered to stop accepting new large queries. New large query requests will be directly canceled.</p> <p>Value range: 10–50000</p> <p>Default value: 1000</p>

 NOTE

In addition to **search.isolator.memory.pool.limit** and **search.isolator.count.limit** parameters, you can configure **search.isolator.memory.task.limit** and **search.isolator.time.management** to control the number of query tasks that enter the isolation pool.

- Policy for selecting a query task in the isolation pool.

```
PUT _cluster/settings
{
  "persistent": {
    "search.isolator.strategy": "fair",
    "search.isolator.strategy.ratio": "0.5%"
  }
}
```

Parameter	Data Type	Description
search.isolator.strategy	String	<p>Policy for selecting large queries when the interrupt control program is triggered. The selected query will be interrupted.</p> <p>NOTE The large query isolation pool is checked every second until the heap memory is within the safe range.</p> <p>Values: fair, mem-first, or time-first</p> <ul style="list-style-type: none"> • mem-first: The query task that uses the most heap memory in the isolation pool is interrupted. • time-first: The query task that has been running for the longest time in the isolation pool is interrupted. • fair: If the difference between the heap memory of shard queries is smaller than <i>Maximum_heap_memory</i> x search.isolator.strategy.ratio, the query that takes the longest time should be interrupted. Otherwise, the query that uses the most heap memory is interrupted. <p>Default value: fair</p>
search.isolator.strategy.ratio	String	<p>Threshold of the fair policy. This parameter takes effect only if search.isolator.strategy is set to fair. If the difference between the memory usage of large query tasks does not exceed the threshold, the query that takes the longest time should be interrupted. If the difference between the memory usage of large query tasks exceeds the threshold, the query that uses the most memory is interrupted.</p> <p>Value range: 0.0 to 100.0% Default value: 1%</p>

- Configure the global timeout duration of query tasks.

```
PUT _cluster/settings
{
  "persistent": {
    "search.isolator.time.limit": "120s"
  }
}
```

Parameter	Data Type	Description
search.isolator.time.limit	String	Global query timeout duration. If this function is enabled, all the query tasks that exceed the specified duration will be canceled. Value range: $\geq 0\text{ms}$ Default value: 120s

11.4 Index Monitoring

11.4.1 Context

CSS monitors various metrics of the running status and change trend of cluster indexes to measure service usage and handle potential risks in a timely manner, ensuring that clusters can run stably.

During index monitoring, the **stats** information about indexes is collected and saved to the monitoring index (**monitoring-eye-css-[yyyy-mm-dd]**) of the cluster, and retained for one week by default.

Currently, only clusters of the version 7.6.2 and 7.10.2 support index monitoring.

11.4.2 Enabling Index Monitoring

1. Log in to the CSS management console.
2. Choose **Clusters** in the navigation pane. On the **Clusters** page, locate the target cluster and click **Access Kibana** in the **Operation** column.
3. Choose **Dev Tools** in the navigation pane on the left and run the following command to enable index monitoring:

```
PUT _cluster/settings
{
  "persistent": {
    "css.monitoring.index.enabled": "true"
  }
}
```

4. (Optional) To monitor a specific index, run the following command on the **Dev Tools** page of Kibana:

```
PUT _cluster/settings
{
  "persistent": {
    "css.monitoring.index.enabled": "true",
    "css.monitoring.index.interval": "30s",
    "css.monitoring.index.indices": ["index_name"],
    "css.monitoring.history.duration": "3d"
  }
}
```

Table 11-24 Parameter description

Parameter	Data Type	Description
css.monitoring.index.enabled	Boolean	Whether to enable index monitoring. If this parameter is set to true , the monitoring will be enabled. Default value: false
css.monitoring.index.interval	Time	Interval for collecting index monitoring data. Minimum value: 1s Default value: 10s
css.monitoring.index.indices	String	Name of an index to be monitored. By default, all indexes are monitored. You can configure specific indexes or a type of indexes to monitor. Example: <ul style="list-style-type: none"> "css.monitoring.index.indices": ["index_name"] indicates only <i>index_name</i> is monitored. "css.monitoring.index.indices": ["log_*"] indicates that only indexes starting with log_ are monitored. "css.monitoring.index.indices": ["index1", "index2"] indicates that index1 and index2 are monitored. Default value: * (indicating that all indexes are monitored)
css.monitoring.history.duration	Time	Retention period of monitoring data storage. The default period is a week. Minimum value: 1d Default value: 7d

NOTICE

Indexes starting with **monitoring-eye-css-*** are regarded as monitoring indexes and will not be monitored.

11.4.3 Checking the Index Read and Write Traffic

You can call an API to query the index read and write traffic within a period of time.

Prerequisites

A cluster has been created and **index monitoring** has been enabled.

Procedure

1. Log in to the CSS management console.
2. Choose **Clusters** in the navigation pane. On the **Clusters** page, locate the target cluster, and click **Access Kibana** in the **Operation** column.
3. Choose **Dev Tools** in the navigation pane on the left and run the following commands to query the index read and write traffic:
 - Check read and write traffic of all the indexes.
GET /_cat/monitoring
 - Check read and write traffic of a specific index.
GET /_cat/monitoring/{indexName}

{indexName} indicates the name of the index whose read and write traffic you want to check.
 - Check the read and write traffic of indexes for different periods.
GET _cat/monitoring?begin=1650099461000
GET _cat/monitoring?begin=2022-04-16T08:57:41
GET _cat/monitoring?begin=2022-04-16T08:57:41&end=2022-04-17T08:57:41

Table 11-25 Parameter description

Parameter	Mandatory	Description
begin	No	Start time (UTC time) of the monitoring data you want to view. Time format: strict_date_optional_time epoch_millis The default start time is five minutes before the current time.
end	No	End time (UTC time) of the monitoring data you want to view. Time format: strict_date_optional_time epoch_millis The default end time is the current time.

NOTE

These parameters cannot be used for system indexes, whose names start with a dot (.).

Information similar to the following is displayed:

```
index  begin          end          status pri rep init unassign docs.count docs.deleted store.size
pri.store.size delete.rate indexing.rate search.rate
test 2022-03-25T09:46:53.765Z 2022-03-25T09:51:43.767Z yellow 1 1 0 1 9 0
5.9kb 5.9kb 0/s 0/s 0/s
```

Table 11-26 Parameters in the returned information

Parameter	Description
index	Index name

Parameter	Description
begin	Start time of the monitoring data you queried.
end	End time of the monitoring data you queried.
status	Index status within the queried monitoring interval.
pri	The number of index shards within the queried monitoring interval.
rep	The number of index replicas within the queried monitoring interval.
init	The number of initialized indexes within the queried monitoring interval.
unassign	The number of unallocated indexes within the queried monitoring interval.
docs.count	The number of documents within the queried monitoring interval.
docs.deleted	The number of deleted documents within the queried monitoring interval.
store.size	Index storage size within the queried monitoring interval.
pri.store.size	Size of the primary index shard within the queried monitoring interval.
delete.rate	Number of indexes deleted per second within the queried monitoring interval.
indexing.rate	Number of indexes wrote per second within the queried monitoring interval.
search.rate	Number of indexes queried per second within the queried monitoring interval.

11.5 Enhanced Monitoring

11.5.1 P99 Latency Monitoring

Context

The Elasticsearch community only discusses how to monitor the average latency of search requests, which cannot reflect the actual search performance of a cluster. To enhance monitoring, CSS allows you to monitor the P99 latency of search requests in clusters.

Prerequisites

Currently, only clusters of version 7.6.2 and 7.10.2 support P99 latency monitoring.

Obtaining Monitoring Information

1. Log in to the CSS management console.
2. Choose **Clusters** in the navigation pane. On the **Clusters** page, locate the target cluster and click **Access Kibana** in the **Operation** column.
3. In the navigation tree on the left, choose **Dev Tools** and run the following command to check the P99 latency of the current cluster:

```
GET /search/stats/percentile
```

Example response:

```
{
  "overall" : {
    "1.0" : 2.0,
    "5.0" : 2.0,
    "25.0" : 6.5,
    "50.0" : 19.5,
    "75.0" : 111.0,
    "95.0" : 169.0,
    "99.0" : 169.0,
    "max" : 169.0,
    "min" : 2.0
  },
  "last_one_day" : {
    "1.0" : 2.0,
    "5.0" : 2.0,
    "25.0" : 6.5,
    "50.0" : 19.5,
    "75.0" : 111.0,
    "95.0" : 169.0,
    "99.0" : 169.0,
    "max" : 169.0,
    "min" : 2.0
  },
  "latest" : {
    "1.0" : 26.0,
    "5.0" : 26.0,
    "25.0" : 26.0,
    "50.0" : 26.0,
    "75.0" : 26.0,
    "95.0" : 26.0,
    "99.0" : 26.0,
    "max" : 26.0,
    "min" : 26.0
  }
}
```

NOTE

- In the response, **overall** indicates all the statistics that have been collected since the cluster startup, **last_one_day** indicates the statistics collected in the last day, and **latest** indicates the statistics that have been collected since the last reset.
- The calculated P99 latency is an estimation. It is more precise than the P50 latency.
- The P99 latency of a cluster is cleared and recalculated if the cluster is restarted.

Other Operations

- Define percentage.

You can run the following command to specify the percentage:

```
GET /search/stats/percentile
{
  "percents": [1, 50, 90]
}
```

- Reset the **latest** statistics.

You can run the following command to reset the **latest** statistics:

```
POST /search/stats/reset
```

Example response:

```
{
  "nodes" : {
    "css-c9c8-ess-esn-1-1" : "ok"
  }
}
```

11.5.2 HTTP Status Code Monitoring

Context

When an external system accesses Elasticsearch through the HTTP protocol, a response and the corresponding status code are returned. The open-source Elasticsearch server does not collect the status code, so users cannot monitor Elasticsearch APIs status or cluster request status. CSS allows you to monitor the HTTP status codes of clusters.

Prerequisites

Currently, only clusters of versions 7.6.2 and 7.10.2 support HTTP status code monitoring.

Obtaining Status Codes

1. Log in to the CSS management console.
2. Choose **Clusters** in the navigation pane. On the **Clusters** page, locate the target cluster and click **Access Kibana** in the **Operation** column.
3. In the navigation tree on the left, choose **Dev Tools**.
4. On the console page of **Dev Tools**, run commands based on the cluster version.

- For clusters of version 7.6.2, run the following command to obtain the status code statistics:

```
GET /_nodes/http_stats
```

Example response:

```
{
  "_nodes" : {
    "total" : 1,
    "successful" : 1,
    "failed" : 0 },
  "cluster_name" : "css-8362",
  "nodes" : {
    "F9IFdQPARaOJI7oL7HOXtQ" : {
      "http_code" : {
        "200" : 114,
        "201" : 5,
        "429" : 0,
        "400" : 7,
        "404" : 0,
        "405" : 0
      }
    }
  }
}
```

- For clusters of version 7.10.2, run the following command to obtain the status code statistics:

GET _nodes/stats/http

Example response:

```
{
// ...
"cluster_name" : "css-2985",
"nodes" : {
// ...
"omvR9_W-TsGApraMApREjA" : {
// ...
"http" : {
"current_open" : 4,
"total_opened" : 37,
"http_code" : {
"200" : 25,
"201" : 7,
"429" : 0,
"400" : 3,
"404" : 0,
"405" : 0
}
}
}
}
}
```

12 Monitoring

12.1 Supported Metrics

Function

This section describes CSS metrics that can be monitored by Cloud Eye as well as their namespaces and dimensions. You can use the management console or APIs provided by Cloud Eye to view the monitoring metrics and alarms generated for CSS.

Namespace

SYS.ES

Monitoring Metrics

- [Table 12-1](#) describes the monitoring metrics of CSS clusters.
- Monitored object: CSS cluster
- Monitoring period (original metric): 1 minute

NOTE

Accumulated value: The value is accumulated from the time when a node is started. After the node is restarted, the value is reset to zero and accumulated again.

Table 12-1 CSS metrics

Metric ID	Metric	Description	Value Range
status	Cluster Health Status	Health status of the monitored object	0,1,2,3 0: All primary and replica shards are allocated. Your cluster is 100% operational. 1: All primary shards are allocated, but at least one replica is missing. No data is missing, so search results will still be complete. However, high availability is compromised to some degree. If more shards disappear, you might lose data. Think of this status as a warning that should prompt investigation. 2: Data is missing and the cluster fails to work. 3: The cluster status is not obtained.
disk_util	Disk Usage	Disk usage of the monitored object. Unit: %	0-100%
max_jvm_heap_usage	Max. JVM Heap Usage	Maximum JVM heap usage of nodes in a CSS cluster. Unit: %	0-100%
max_jvm_young_gc_time	Max. JVM Young GC Duration	Maximum accumulated JVM Young GC duration of nodes in a CSS cluster. Unit: ms	≥ 0 ms

Metric ID	Metric	Description	Value Range
max_jvm_young_gc_count	Max. JVM Young GC Count	Maximum accumulated JVM Young GC count of nodes in a CSS cluster.	≥ 0
max_jvm_old_gc_time	Max. JVM Old GC Duration	Maximum accumulated JVM Old GC duration of nodes in a CSS cluster. Unit: ms	≥ 0 ms
max_jvm_old_gc_count	Max. JVM Old GC Count	Maximum accumulated JVM Old GC count of nodes in a CSS cluster.	≥ 0
total_fs_size	Total Size of File Systems	Total size of file systems in a CSS cluster. Unit: byte	≥ 0 bytes
free_fs_size	Available Size of File Systems	Available size of file systems in a CSS cluster. Unit: byte	≥ 0 bytes
max_cpu_usage	Max. CPU Usage	Maximum node CPU usage in a CSS cluster. Unit: %	0-100%
max_cpu_time_of_jvm_process	Max. CPU Time of JVM Process	Maximum accumulated CPU usage duration of node JVM processes in a CSS cluster. Unit: ms	≥ 0 ms
max_virtual_memory_size_of_jvm_process	Max. Virtual Memory Size of JVM Process	Maximum virtual memory size of node JVM processes in a CSS cluster. Unit: byte	≥ 0 bytes

Metric ID	Metric	Description	Value Range
max_current_ope ned_http_count	Current Max. Opened HTTP Connections	Maximum number of HTTP connections that are currently open for nodes in a CSS cluster.	≥ 0
max_total_opene d_http_count	Total Max. Opened HTTP Connections	Maximum number of HTTP connections that were open for nodes in a CSS cluster.	≥ 0
indices_count	Indexes	Number of indexes in a CSS cluster	≥ 0
total_shards_cou nt	Shards	Number of shards in a CSS cluster	≥ 0
primary_shards_c ount	Primary Shards	Number of primary shards in a CSS cluster	≥ 0
docs_count	Documents	Number of documents in a CSS cluster	≥ 0
docs_deleted_cou nt	Deleted Documents	Number of documents deleted in a CSS cluster	≥ 0
nodes_count	Nodes	Number of nodes in a CSS cluster	≥ 0
data_nodes_coun t	Data Nodes	Number of data nodes in a CSS cluster	≥ 0
coordinating_nod es_count	Coordinating Nodes	Number of coordinating nodes in a CSS cluster	≥ 0
master_nodes_co unt	Master Nodes	Number of master nodes in a CSS cluster	≥ 0
ingest_nodes_cou nt	Client Nodes	Number of client nodes in a CSS cluster	≥ 0

Metric ID	Metric	Description	Value Range
max_load_average	Max. Node Load	Maximum number of average queuing tasks per minute on nodes in a cluster.	≥ 0
avg_cpu_usage	Avg. CPU Usage	Average node CPU usage in a CSS cluster. Unit: %	0-100%
avg_load_average	Avg. Node Load	Average number of queuing tasks per minute on nodes in a CSS cluster.	≥ 0
avg_jvm_heap_usage	Avg. JVM Heap Usage	Average node JVM heap usage in a CSS cluster. Unit: %	0-100%
max_open_file_descriptors	Max. Open File Descriptors	Maximum number of node file descriptors that are currently open in a CSS cluster.	≥ 0
avg_open_file_descriptors	Avg. Open File Descriptors	Average number of node file descriptors that are currently open in a CSS cluster.	≥ 0
sum_max_file_descriptors	Max. Allowed File Descriptors	Maximum number of allowed node file descriptors in a CSS cluster.	≥ 0
sum_open_file_descriptors	Open File Descriptors	Number of node file descriptors that are currently open in a cluster.	≥ 0
sum_thread_pool_write_queue	Tasks in Write Queue	Number of job queues in a write thread pool	≥ 0
sum_thread_pool_search_queue	Tasks in Search Queue	Total number of queuing tasks in the search thread pools of nodes in the CSS cluster.	≥ 0

Metric ID	Metric	Description	Value Range
sum_thread_pool_force_merge_queue	Tasks in ForceMerge Queue	Total number of queuing tasks in the force merge thread pools of nodes in the CSS cluster.	≥ 0
sum_thread_pool_write_rejected	Rejected Tasks in Write Queue	Total number of rejected tasks in the write thread pools of nodes in the CSS cluster.	≥ 0
sum_thread_pool_search_rejected	Rejected Tasks in Search Queue	Total number of rejected tasks in the search thread pools of nodes in the CSS cluster.	≥ 0
sum_thread_pool_force_merge_rejected	Rejected Tasks in ForceMerge Queue	Total number of rejected tasks in the force merge thread pools of nodes in the CSS cluster.	≥ 0
max_thread_pool_search_queue	Max. Tasks in Search Queue	Maximum number of queuing tasks in the search thread pools of nodes in a CSS cluster.	≥ 0
max_thread_pool_force_merge_queue	Max. Tasks in ForceMerge Queue	Maximum number of queuing tasks in the force merge thread pools of nodes in a CSS cluster.	≥ 0
sum_thread_pool_write_threads	Size of Write Thread Pool	Total size of the write thread pools of nodes in the CSS cluster.	≥ 0
sum_thread_pool_search_threads	Size of Search Thread Pool	Total size of the search thread pools of nodes in the CSS cluster.	≥ 0
sum_thread_pool_force_merge_threads	Size of ForceMerge Thread Pool	Total size of the force merge thread pools of nodes in the CSS cluster.	≥ 0

Metric ID	Metric	Description	Value Range
avg_thread_pool_write_queue	Avg. Tasks in Write Queue	Average number of queuing tasks in the write thread pools of nodes in a CSS cluster.	≥ 0
avg_thread_pool_search_queue	Avg. Tasks in Search Queue	Average number of queuing tasks in the search thread pools of nodes in a CSS cluster.	≥ 0
avg_thread_pool_force_merge_queue	Avg. Tasks in ForceMerge Queue	Average number of queuing tasks in the force merge thread pools of nodes in the CSS cluster.	≥ 0
avg_thread_pool_search_threads	Avg. Size of Search Thread Pool	Average size of the search thread pool of a node in a CSS cluster.	≥ 0
avg_thread_pool_write_threads	Avg. Size of Write Thread Pool	Average size of the write thread pool of a node in a CSS cluster.	≥ 0
avg_thread_pool_force_merge_threads	Avg. Size of ForceMerge Thread Pool	Average size of the force merge thread pool of a node in a CSS cluster.	≥ 0
avg_thread_pool_write_rejected	Avg. Rejected Tasks in Write Queue	Average number of rejected tasks in the write thread pool of a node in a CSS cluster.	≥ 0
min_free_fs_size	Min. Available Storage Space	Minimum available storage space of nodes in a CSS cluster. Unit: byte	≥ 0 bytes
avg_jvm_old_gc_count	Avg. GCs of Old-Generation JVM	Average number of old-generation garbage collections of nodes in a CSS cluster.	≥ 0

Metric ID	Metric	Description	Value Range
avg_jvm_old_gc_time	Avg. GC Duration of Old-Generation JVM	Average old-generation garbage collection duration of nodes in a CSS cluster. Unit: ms	≥ 0 ms
avg_jvm_young_gc_count	Avg. GCs of Young-Generation JVM	Average number of young-generation garbage collections of nodes in a CSS cluster.	≥ 0
avg_jvm_young_gc_time	Avg. GC Duration of Young-Generation JVM	Average young-generation garbage collection duration of nodes in a CSS cluster. Unit: ms	≥ 0 ms
avg_max_file_descriptors	Avg. Maximum Allowed File Descriptors	Average value of the maximum number of allowed file descriptors on each node in a CSS cluster.	≥ 0
avg_mem_free_in_bytes	Avg. Available Memory	Average unused memory capacity of nodes in a CSS cluster. Unit: byte	≥ 0 bytes
avg_mem_free_percent	Avg. Available Memory Percentage	Average percentage of unused memory of nodes in a CSS cluster. Unit: %	0-100%
avg_mem_used_in_bytes	Avg. Used Memory	Average used memory of nodes in a CSS cluster. Unit: byte	≥ 0 bytes
avg_mem_used_percent	Avg. Used Memory Percentage	Average percentage of used memory of nodes in a CSS cluster. Unit: %	0-100%

Metric ID	Metric	Description	Value Range
max_mem_free_in_bytes	Max. Available Memory	Maximum unused memory of nodes in a CSS cluster. Unit: byte	≥ 0 bytes
max_mem_free_percent	Max. Available Memory Percentage	Maximum percentage of unused memory of nodes in a CSS cluster. Unit: %	0-100%
max_mem_used_in_bytes	Max. Used Memory	Maximum used memory of nodes in a CSS cluster. Unit: byte	≥ 0 bytes
max_mem_used_percent	Max. Used Memory Percentage	Maximum percentage of used memory of nodes in a CSS cluster. Unit: %	0-100%
sum_jvm_old_gc_count	Total GCs of Old-Generation JVM	Number of old-generation garbage collections of nodes in a CSS cluster.	≥ 0
sum_jvm_old_gc_time	Total GC Duration of Old-Generation JVM	Total old-generation garbage collection duration of nodes in the CSS cluster. Unit: ms	≥ 0ms
sum_jvm_young_gc_count	Total GCs of Young-Generation JVM	Number of young-generation garbage collections of nodes in a CSS cluster.	≥ 0
sum_jvm_young_gc_time	Total GC Duration of Young-Generation JVM	Total young-generation garbage collection duration of nodes in the CSS cluster. Unit: ms	≥ 0 ms
sum_current_opened_http_count	Currently Open HTTP Connections	Number of HTTP connections that are open on nodes in a CSS cluster.	≥ 0

Metric ID	Metric	Description	Value Range
sum_total_opene d_http_count	Historical Open HTTP Connections	Number of HTTP connections that were open on nodes in a CSS cluster.	≥ 0
IndexingLatency	Average Index Latency	Average time required for a shard to complete an index operation. Unit: ms	≥ 0 ms
IndexingRate	Average Index Rate	Average number of index operations per second in a cluster. Unit: s	≥ 0s
SearchLatency	Average Search Latency	Average time required for a segment to complete the search operation. Unit: ms	≥ 0 bytes
SearchRate	Average QPS	Average queries per second (QPS) in a cluster. Unit: s	≥ 0/s

Dimension

Table 12-2 Dimension description

Key	Value
cluster_id	CSS cluster

12.2 Configuring Cluster Monitoring

You can use Cloud Eye to monitor the created clusters. After configuring the cluster monitoring, you can log in to the Cloud Eye management console to view cluster metrics.

The procedure for configuring cluster monitoring:

1. **Creating Alarm Rules:** Customize alarm rules for the monitoring metrics. Once a metric exceeds the threshold, the system will notify you by sending emails or HTTP/HTTPS requests.

2. **Configuring Monitoring Metrics:** Configure monitoring metrics for a cluster or a node in the cluster.
3. **Viewing Monitoring Metrics:** View the statistics of the monitoring metrics in specific periods.

Prerequisites

- The cluster is in the **Available** or **Processing** status.
- The cluster has been running properly for more than 10 minutes.

Recommended Monitoring Metrics

- Cluster CPU and JVM usage. You are advised to configure the following monitoring metrics: average JVM heap usage, maximum JVM heap usage, average CPU usage, and maximum CPU usage.
- Cluster write and query latency and throughput. You are advised to configure the following monitoring metrics: average index latency, average index rate, average search latency, and average QPS.
- Cluster write and query queue and rejected tasks. You are advised to configure the following monitoring metrics: tasks in write queue, tasks in search queue, rejected tasks in write queue, and rejected tasks in search queue.

Creating Alarm Rules

1. Log in to the Cloud Eye console.
2. In the navigation pane on the left, choose **Alarm Management > Alarm Rules**.
3. In the **Resource Type** column, select **Cloud Search Service** as criteria to search for alarm rules that meet the requirements.

If no alarm rules are available, create one by referring to *Creating an Alarm Rule and Notification*. For details about how to set **Resource Type** and **Dimension**, see [Table 12-3](#).

Table 12-3 Alarm rule configuration parameter

Parameter	Description	Remark
Resource Type	Type of the resource that the alarm rule is created for	Select Cloud Search Service .
Dimension	Metric dimension of the selected resource type	<p>CSS supports two dimensions. Select a dimension as required.</p> <ul style="list-style-type: none"> • CSS Clusters: Alarm rules are specified by cluster. • CSS Clusters - CSS Instances: Alarm rules are specified by node in a cluster.

Configuring Monitoring Metrics

1. Create a monitoring panel. For details, see *Creating a Monitoring Panel*. If an available monitoring panel has been created, skip this step.
2. Add CSS monitoring graphs. For details, see *Adding a Graph*.

For details about how to set **Resource Type** and **Dimension**, see [Table 12-4](#).

Table 12-4 Graph configuration parameter

Parameter	Description	Remark
Resource Type	Type of the resource to be monitored	Select Cloud Search Service .
Dimension	Metric dimension	<p>CSS supports two dimensions. Select a dimension as required.</p> <ul style="list-style-type: none"> ● CSS Clusters: Monitoring is executed by cluster. ● CSS Clusters - CSS Instances: Monitoring is executed by node in a cluster.

Viewing Monitoring Metrics

1. Log in to the CSS management console.
2. Choose **Clusters**. Locate the target cluster and choose **More > View Metric** in the **Operation** column.
3. Select a time range.
4. View the monitoring metrics.

13 Auditing

13.1 Key Operations Recorded by CTS

With CTS, you can record operations associated with CSS for later query, audit, and backtrack operations.

Prerequisites

CTS has been enabled.

Key Operations Recorded by CTS

Table 13-1 Key operations recorded by CTS



Operation	Resource Type	Event Name
Creating a cluster	cluster	createCluster
Deleting a cluster	cluster	deleteCluster
Expanding the cluster capacity	cluster	roleExtendCluster
Restarting a cluster	cluster	rebootCluster
Performing basic configurations for a cluster snapshot	cluster	updateSnapshotPolicy
Setting the automatic snapshot creation policy	cluster	updateAutoSnapshotPolicy
Upgrading a cluster	cluster	upgradeCluster
Retrying the upgrade	cluster	retryAction
Manually creating a snapshot	snapshot	createSnapshot

Operation	Resource Type	Event Name
Restoring a snapshot	snapshot	restoreSnapshot
Deleting a snapshot	snapshot	deleteSnapshot

13.2 Viewing Audit Logs

After you enable CTS, it starts recording operations related to CSS. The CTS management console stores the last seven days of operation records. This section describes how to query the last seven days of operation records on the CTS management console.

Procedure

1. Log in to the CTS management console.
2. Click  in the upper left corner and select a region.
3. In the navigation pane on the left, click **Trace List**.
4. You can use filters to query traces. The following four filter criteria are available:
 - **Trace Source, Resource Type, and Search By**
Select a filter criterion from the drop-down list.
When you select **Trace name** for **Search By**, select a specific trace name.
When you select **Resource ID** for **Search By**, enter a specific resource ID.
When you select **Resource name** for **Search By**, select or enter a specific resource name.
 - **Operator**: Select a specific operator (at user level rather than tenant level).
 - **Trace Status**: Available options include **All trace statuses, normal, warning, and incident**. You can only select one of them.
 - **Time Range**: You can query traces generated during any time range of the last seven days.
5. Click  on the left of a trace to expand its details.
6. Click **View Trace** in the **Operation** column. In the displayed **View Trace** dialog box, the trace structure details are displayed.
For details about the key fields in the CTS trace structure, see the *Cloud Trace Service User Guide*.

14 Best Practices

14.1 Cluster Migration

14.1.1 Migration Solution Overview

You can migrate data to a Elasticsearch cluster from another Elasticsearch cluster, a user-built Elasticsearch cluster, or a third-party Elasticsearch cluster. This section describes the solutions for data migration from different clusters.

Scenarios

The migration solution varies depending on the data source.

- Migration from an Elasticsearch cluster
You can use Logstash, CDM, OBS backup and restoration, ESM, or cross-cluster replication plug-ins to migrate data in an Elasticsearch cluster.
 - Logstash: an official data cleaning tool provided by Elasticsearch. It is a part of the Elk ecosystem and provides powerful functions. It can migrate data between different data sources and Elasticsearch, and clean and process data. For details, see [Migrating Cluster Data Using Logstash](#).
 - CDM: a cloud migration tool provided by cloud to implement cluster migration between different cloud services. For details, see .
 - Backup and restoration: Elasticsearch provides backup and restoration capabilities. You can back up the data of a cluster to OBS, and restore the data to another cluster. For details, see [Migrating Cluster Data Through Backup and Restoration](#).
- [Migration from Kafka/MQ](#)
- [Migration from a Database](#)

Solutions

CSS supports migration by backup and restoration, by using the Reindex API or Logstash+ESM, or by data source synchronization. For details, see [Table 14-1](#).

Data source synchronization has fewer constraints and higher performance than the other three solutions. Data source synchronization allows cutover anytime after the synchronization completed, which is more convenient and flexible.

Table 14-1 Migration solutions

Solution	Description	Constraint	Performance
Backup and restoration	Prepare shared storage that supports the S3 protocol, for example, an OBS bucket. Create a snapshot to back up the data of the source Elasticsearch cluster, synchronize the snapshot to the target cluster, and restore data to the target cluster.	<ul style="list-style-type: none"> • Target Elasticsearch version \geq Source Elasticsearch version • Number of candidate master nodes of the target Elasticsearch cluster $>$ Half of the number of candidate master nodes of the source Elasticsearch cluster • Incremental data synchronization is not supported. You need to stop update before backing up or restoring data. 	The data migration rate is configurable. Ideally, the data migration rate is the same as the file copy rate.
Reindex API	Configure mutual trust between the source and target Elasticsearch clusters, and then migrate data using the Reindex API.	<ul style="list-style-type: none"> • _source must be enabled for indexes. • Real-time synchronization of incremental data is not supported. You need to stop the update and then call the API. 	Batch read and write are supported, but concurrent slicing synchronization is not supported.

Solution	Description	Constraint	Performance
Logstash +ESM	Apply for an ECS, deploy and configure Logstash on it, and then start data migration.	<ul style="list-style-type: none"> • <code>_source</code> must be enabled for indexes. • Real-time synchronization of incremental data is not supported. You need to stop the update and then start Logstash. 	Batch read and write are supported, and concurrent slicing synchronization is supported.
Data source synchronization	Inventory data is migrated using Logstash, and incremental data is automatically synchronized through traffic replication or data links.	None	The inventory migration rate is the same as that of Logstash. An existing tool is reused for incremental migration.

14.1.2 Migration from Elasticsearch

14.1.2.1 Migrating Cluster Data Using Logstash

Logstash is an official data migration tool provided by Elasticsearch.

Step 1 Apply for an ECS, preferably with at least 8 vCPUs and 16 GB memory.

Step 2 Install Logstash on the ECS.

1. Install JDK, because Logstash depends on Java. Run the following command to install JDK using **yum**:

```
yum install java
yum install python
```

2. Download Logstash. Choose a Logstash version close to the Elasticsearch version. They do not have to use exactly the same version.

Logstash 7.10.2 OSS is recommended. You can download it from <https://www.elastic.co/downloads/past-releases/logstash-oss-7-10-2>

3. Run the following command to install Logstash using **yum**:

```
yum install logstash-oss-7.10.0-x86_64.rpm
```

Replace `logstash-oss-7.10.0-x86_64.rpm` with the actual Logstash installation package name.

Step 3 Modify the JVM configuration of Logstash to improve the cluster data migration efficiency.

Run the following command to modify the JVM configuration. The default heap memory of Logstash is 1 GB. You are advised to change the heap memory to half of the cluster node memory.

```
vim /etc/logstash/jvm.options
-Xms4g
-Xmx4g
```

Step 4 Modify the **conf** configuration file of Logstash and configure cluster migration settings.

1. Go to the **/etc/logstash/conf.d/** directory where the Logstash configuration file is stored.

```
cd /etc/logstash/conf.d/
```

2. Create the **logstash-es-es-all.conf** file.

```
vim logstash-es-es-all.conf
```

3. Add the following content to the **logstash-es-es-all.conf** file and save the file.

Modify the **hosts**, **user**, **password**, **index** fields as needed.

```
input{
  elasticsearch{
    #IP address of the source cluster
    hosts => ["http://172.16.xxx.xxx:9200", "http://172.16.xxx.xxx:9200"]
    # #For a security cluster, configure the username and password for cluster login. For a non-
security cluster, you can use the number sign (#) to comment out the user and password fields.
    # user => "xxx"
    # password => "xxx"
    # #List of indexes to be migrated. Multiple indexes are separated by commas (,). Set this
parameter based on the actual host information. -* indicates that indexes starting with a period (.)
are excluded.
    index => "abmau_edi*,business_test,goods_deploy*, -*"
    # Retain the default values of the following three items, including the number of threads, the
size of migrated data, and Logstash JVM configurations.
    docinfo=>true
    # Retain the default value. To increase the migration speed, you can increase the values of the
following two parameters, but to a proper extent.
    slices => 3
    size => 3000
  }
}

filter {
  # Delete some fields added by Logstash.
  mutate {
    remove_field => ["@timestamp", "@version"]
  }
}

output{
  elasticsearch{
    # Destination cluster address.
    hosts => ["http://10.100.xx.xx:9200", "http://10.100.xx.xx:9200"]
    # Username and password for logging in to the target cluster. If you do not need to configure
them, use the number sign (#) to comment them out.
    user => "admin"
    password => "*****"
    # Index name of the target cluster. The following configurations must be the same as that of
the source cluster.
    index => "%{[@metadata][_index]}"
    # #Index type of the target cluster. The following configurations must be the same as that of
the source cluster.
    document_type => "%{[@metadata][_type]}"
    # _id of the target data. If the original _id does not need to be retained, you can delete it. After
the deletion, the cluster performance can be better.
    document_id => "%{[@metadata][_id]}"
    ilm_enabled => false
    manage_template => false
  }
  # Debugging information. You are advised to delete this information before migration.
```

```
# stdout { codec => rubydebug { metadata => true }}
}
```

Step 5 Start Logstash to migrate cluster data.

1. Run the following command to start Logstash:
`/usr/share/logstash/bin/logstash --path.settings /etc/logstash`
2. View the Logstash log file to check the task progress. The Logstash log directory is `/var/log/logstash/`.
3. Wait until the data migration is complete.

----End

14.1.2.2 Migrating Cluster Data Through Backup and Restoration

- To migrate data between Elasticsearch clusters, follow the instructions in [Migrating Cluster Data](#).
- To migrate data from a user-built or third-party Elasticsearch cluster to a Elasticsearch cluster, perform the steps in this section.

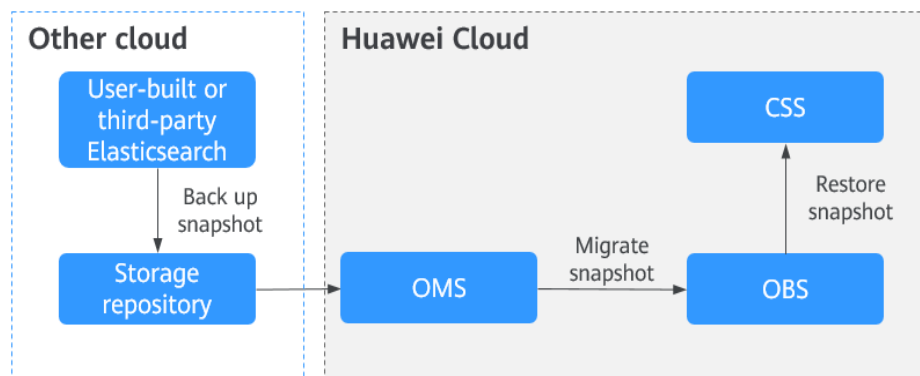
Prerequisites

- Before using backup and restoration, ensure that:
 - Target Elasticsearch version \geq Source Elasticsearch version
 - Number of candidate master nodes of the target Elasticsearch cluster $>$ Half of the number of candidate master nodes of the source Elasticsearch cluster
- Backup and restoration do not support incremental data synchronization. You need to stop data update before backing up data.
- The target Elasticsearch cluster has been created in CSS.

Migration Process

The following figure shows the cluster migration process when the source is a user-built or third-party Elasticsearch cluster, and the target is an Elasticsearch cluster of CSS.

Figure 14-1 Migration through backup and restoration



Procedure

- Step 1** Create a shared repository that supports the S3 protocol, for example.
- Step 2** Create a snapshot backup repository in the user-built or third-party Elasticsearch cluster to store Elasticsearch snapshot data.

For example, create a backup repository named **my_backup** in Elasticsearch and associate it with the repository OSS.

```
PUT _snapshot/my_backup
{
  # Repository type.
  "type": "oss",
  "settings": {
    # # Private network domain name of the repository in step 1.
    "endpoint": "http://oss-xxx.xxx.com",
    # User ID and password of the repository.
    "access_key_id": "xxx",
    "secret_access_key": "xxx",
    # Bucket name of the repository created in step 1.
    "bucket": "patent-esbak",
    # # Whether to enable snapshot file compression.
    "compress": false,
    # If the size of the uploaded snapshot data exceeds the value of this parameter, the data will be
    # uploaded as blocks to the repository.
    "chunk_size": "1g",
    # Start position of the repository. The default value is the root directory.
    "base_path": "snapshot/"
  }
}
```

- Step 3** Create a snapshot for the user-built or third-party Elasticsearch cluster.

- Create a snapshot for all indexes.

For example, create a snapshot named **snapshot_1**.

```
PUT _snapshot/my_backup/snapshot_1?wait_for_completion=true
```

- Create a snapshot for specified indexes.

For example, create a snapshot named **snapshot_test** that contains indexes **patent_analyse** and **patent**.

```
PUT _snapshot/my_backup/snapshot_test
{
  "indices": "patent_analyse,patent"
}
```

- Step 4** View the snapshot creation progress of the cluster.

- Run the following command to view information about all snapshots:

```
GET _snapshot/my_backup/_all
```

- Run the following command to view information about **snapshot_1**:

```
GET _snapshot/my_backup/snapshot_1
```

- Step 5** Migrate snapshot data from the repository to OBS.

The Object Storage Migration Service (OMS) supports data migration from multiple cloud vendors to OBS.

- Step 6** Create a repository in the Elasticsearch cluster of CSS and associate it with OBS. This repository will be used for restoring the snapshot data of the user-built or third-party Elasticsearch cluster.

For example, create a repository named **my_backup_all** in the cluster and associate it with the destination OBS.

```
PUT _snapshot/my_backup_all/
{
  "type" : "obs",
  "settings" : {
    # Private network domain name of OBS
    "endpoint" : "obs.xxx.xxx.com",
    "region" : "xxx",
    # Username and password for accessing OBS
    "access_key" : "xxx",
    "secret_key" : "xxx",
    # OBS bucket name, which must be the same as the destination OBS bucket name in the previous step
    "bucket" : "esbak",
    "compress" : "false",
    "chunk_size" : "1g",
    #Note that there is no slash (/) after snapshot.
    "base_path" : "snapshot",
    "max_restore_bytes_per_sec" : "100mb",
    "max_snapshot_bytes_per_sec" : "100mb"
  }
}
```

Step 7 Restore the snapshot data to the Elasticsearch cluster of CSS.

1. Check information about all snapshots.

```
GET _snapshot
```

2. Restore a snapshot

- Restore all the indexes from a snapshot. For example, to restore all the indexes from **snapshot_1**, run the following command:

```
POST _snapshot/my_backup_all/snapshot_1/_restore?wait_for_completion=true
```

- Restores some indexes from a snapshot. For example, in the snapshot named **snapshot_1**, restore only the indexes that do not start with a period (.).

```
POST _snapshot/my_backup/snapshot_1/_restore
{"indices":"*,-.monitoring*,-.security*,-.kibana*","ignore_unavailable":"true"}
```

- Restore a specified index from a snapshot and renames the index. For example, in **snapshot_1**, restore **index_1** to **restored_index_1** and **index_2** to **restored_index_2**.

```
POST /_snapshot/my_backup/snapshot_1/_restore
{
  # Restore only indexes index_1 and index_2 and ignore other indexes in the snapshot.
  "indices": "index_1,index_2"
  # Search for the index that is being restored. The index name must match the provided
  template.
  "rename_pattern": "index_(.+)",
  # Rename the found index.
  "rename_replacement": "restored_index_$1"
}
```

Step 8 View the snapshot restoration result.

- Run the following command to view the restoration results of all snapshots:

```
GET /_recovery/
```

- Run the following command to check the snapshot restoration result of a specified index:

```
GET {index_name}/_recovery
```

----End

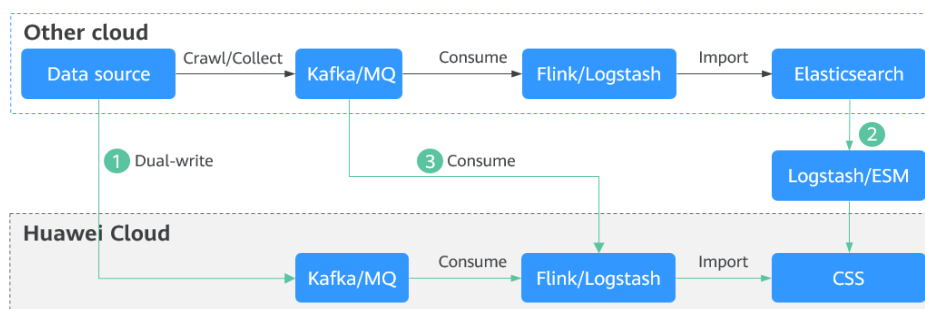
14.1.3 Migration from Kafka/MQ

Process

In industries dealing with a large amount of data, such as IoT, news, public opinion analysis, and social networking, message middleware such as Kafka and MQ is used to balance traffic in peak and off-peak hours. The tools such as Flink and Logstash are then used to consume data, preprocess data, and import data to the search engine, providing the search service for external systems.

The following figure shows the process of migrating data from a Kafka or MQ cluster.

Figure 14-2 Migration from a Kafka or MQ cluster



This migration solution is convenient and flexible.

- Convenient: Once the data of the two ES clusters becomes consistent, a cutover can be performed at any time.
- Flexible: Data can be added, deleted, modified, and queried on both sides.

Procedure

- Step 1** Subscribe to incremental data. Create a consumer group in Kafka or MQ, and subscribe to incremental data.
- Step 2** Synchronize inventory data. Use a tool such as Logstash to migrate data from the source Elasticsearch cluster to the CSS cluster. If Logstash is used for data migration, see [Migrating Cluster Data Using Logstash](#).
- Step 3** Synchronize incremental data. After the inventory data is synchronized, enable the incremental consumer group. Based on the idempotence of Elasticsearch operations on data, when the new consumer group catches up with the previous consumer group, the data on both sides will be consistent.

NOTE

For log migration, data in the source Elasticsearch cluster does not need to be migrated, and you can skip the inventory data synchronization. After the incremental data synchronization is complete, synchronize the data for a period of time (for example, three or seven days), and then directly perform cutover.

----End

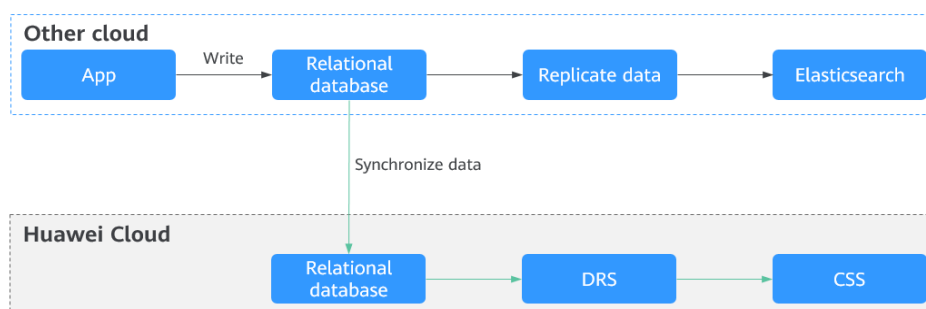
14.1.4 Migration from a Database

Process

Elasticsearch supports full-text search and ad hoc queries. It is often used as a supplement to relational databases, such as MySQL and GaussDB(for MySQL), to improve the full-text search and high-concurrency ad hoc query capabilities of databases.

The following figure shows the process of migrating data from a database.

Figure 14-3 Migration from a database



This migration solution is convenient and flexible.

- Convenient: You can start a cutover while the CSS synchronizes incremental data.
- Flexible: Data can be added, deleted, modified, and queried on both sides.

Procedure

Data Replication Service (DRS) can be used to migrate and synchronize data between relational databases, such as MySQL databases. For details about the supported database types, see .

- Step 1** Set up the data synchronization link. Set up a synchronization link from a database to CSS.
- Step 2** Synchronize data from the database. Use a third-party tool, such as DRS or DataX, to synchronize data to CSS.

----End

14.2 Cluster Access

14.2.1 Overview

Elasticsearch clusters support multiple connection modes. You can determine how to access an Elasticsearch cluster based on the programming language used for your services. For more information about the clients used for CSS clusters in different security modes (non-security mode, security mode+HTTP, and security mode+HTTPS), see [Table 14-2](#).

- CSS provides visualized Kibana and Cerebro APIs for monitoring and operating clusters. On the CSS console, you can quickly access the Kibana and Cerebro of an Elasticsearch cluster.
- You can access Elasticsearch clusters by using cURL commands, Java clients, and Python clients. You can also use Hadoop clients to develop complex applications. Elasticsearch provides Java clients, including Rest High Level Client, Rest Low Level Client, and Transport Client. To avoid compatibility issues, use the Java client that matches your Elasticsearch cluster version.

Table 14-2 Support for access from different clients

Client	Cluster in Non-Security Mode	Cluster in Security Mode + HTTP	Cluster in Security Mode + HTTPS
Kibana	Supported by clusters in all the three modes. To log in to Kibana from a cluster in security mode, enter the username and password for authentication. For details, see .		
Cerebro	Supported by clusters in all the three modes. To log in to Cerebro from a cluster in security mode, enter the username and password for authentication. For details, see .		
cURL	Supported by clusters in all the three modes. For details about the commands used for each mode, see Accessing a Cluster Using cURL Commands .		
Java (Rest High Level Client)	Supported by clusters in all the three modes. For details about the commands used for each mode, see Accessing a Cluster Through the Rest High Level Client .		
Java (Rest Low Level Client)	Supported by clusters in all the three modes. For details about the commands used for each mode, see Accessing a Cluster Through the Rest Low Level Client .		
Java (Transport Client)	Only clusters in non-security mode are supported. For details, see Accessing the Cluster Through the Transport Client .	Not supported	Not supported
Python	Supported by clusters in all the three modes. For details about the commands used for each mode, see Accessing a Cluster Using Python .		
ES-Hadoop	Supported by clusters in all the three modes. For details about the commands used for each mode, see Using ES-Hadoop to Read and Write Data in Elasticsearch Through Hive .		

14.2.2 Accessing a Cluster Using cURL Commands

If the CSS cluster and ECS are in the same VPC, you can run cURL commands on the ECS to directly access the Elasticsearch cluster. This method is mainly used to

check whether the client that accesses the cluster can be connected to Elasticsearch nodes.

Prerequisites

- The CSS cluster is available.
- An ECS that meets the following requirements is available:
 - The ECS and the CSS cluster must be in the same VPC to ensure network connectivity.
 - The security group of the ECS must be the same as that of the CSS cluster.

If they are different, change the ECS security group, or configure the inbound and outbound rules of the group to allow access from all the security groups of the cluster. For details, see the *Virtual Private Cloud User Guide*.

For details about how to use the ECS, see Elastic Cloud Server User Guide.

Procedure

1. Obtain the private network address of the cluster. It is used to access the cluster.
 - a. In the navigation pane on the left, choose **Clusters**.
 - b. In the cluster list, select a cluster, and obtain and record its **Private Network Address**. Format: `<host>:<port>` or `<host>:<port>,<host>:<port>`
 If the cluster has only one node, the IP address and port number of only one node are displayed, for example, **10.62.179.32:9200**. If the cluster has multiple nodes, the IP addresses and port numbers of all nodes are displayed, for example, **10.62.179.32:9200,10.62.179.33:9200**.
2. Run one of the following commands on the ECS to access the cluster. The access command varies according to the security mode of the cluster.
 - Cluster in non-security mode
`curl "http://<host>:<port>"`
 - Cluster in security mode + HTTP
`curl -u <user>:<password> "http://<host>:<port>"`
 - Cluster in security mode + HTTPS
`curl -u <user>:<password> -k "https://<host>:<port>"`

Table 14-3 Variables

Variable	Description
<host>	IP address of each node in the cluster. If the cluster contains multiple nodes, there will be multiple IP addresses. You can use any of them.
<port>	Port number for accessing a cluster node. Generally, the port number is 9200.
<user>	Username for accessing the cluster.
<password>	Password of the user.

An access example is as follows:

```
curl "http://10.62.176.32:9200"
```

Information similar to the following is displayed:

```
HTTP/1.1 200 OK
content-type: application/json; charset=UTF-8
content-length: 513

{
  "name" : "xxx-1",
  "cluster_name" : "xxx",
  "cluster_uuid" : "xxx_uuid",
  "version" : {
    "number" : "7.10.2",
    "build_flavor" : "oss",
    "build_type" : "tar",
    "build_hash" : "unknown",
    "build_date" : "unknown",
    "build_snapshot" : true,
    "lucene_version" : "8.7.0",
    "minimum_wire_compatibility_version" : "6.7.0",
    "minimum_index_compatibility_version" : "6.0.0-beta1"
  },
  "tagline" : "You Know, for Search"
}
```

NOTE

For more commands, see the [Elasticsearch documentation](#).

14.2.3 Accessing a Cluster Using Java

14.2.3.1 Accessing a Cluster Through the Rest High Level Client

Elasticsearch provides SDK (Rest High Level Client) for connecting to a cluster. This client encapsulates Elasticsearch APIs. You only need to construct required structures to access the Elasticsearch cluster. For details about how to use the Rest Client, see the official document at <https://www.elastic.co/guide/en/elasticsearch/client/java-api-client/master/index.html>.

This section describes how to use the Rest High Level Client to access the CSS cluster. The Rest High Level Client can be connected to the cluster in any of the following ways:

- **Connecting to a Non-Security Cluster Through the Rest High Level Client:** applicable to clusters in non-security mode
- **Connecting to a Security Cluster Through Rest High Level Client (Without Security Certificates):** applicable to clusters in security mode+HTTP, and to clusters in security mode+HTTPS (without using certificates)
- **Connecting to a Security Cluster Through Rest High Level Client (With Security Certificates):** applicable to clusters in security mode+HTTPS

Precautions

You are advised to use the Rest High Level Client version that matches the Elasticsearch version. For example, use Rest High Level Client 7.6.2 to access the

Elasticsearch cluster 7.6.2. If your Java Rest High Level Client version is later than the Elasticsearch cluster and incompatible with a few requests, you can use **RestHighLevelClient.getLowLevelClient()** to obtain Low Level Client and customize the Elasticsearch request content.

Prerequisites

- The CSS cluster is available.
- Ensure that the server running Java can communicate with the CSS cluster.
- Install JDK 1.8 on the server. You can download JDK 1.8 from: <https://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html>
- Declare Java dependencies.

7.6.2 indicates the version of the Elasticsearch Java client.

– Maven mode:

```
<dependency>
  <groupId>org.elasticsearch.client</groupId>
  <artifactId>elasticsearch-rest-high-level-client</artifactId>
  <version>7.6.2</version>
</dependency>
<dependency>
  <groupId>org.elasticsearch</groupId>
  <artifactId>elasticsearch</artifactId>
  <version>7.6.2</version>
</dependency>
```

– Gradle mode:

```
compile group: 'org.elasticsearch.client', name: 'elasticsearch-rest-high-level-client', version:
'7.6.2'
```

Connecting to a Non-Security Cluster Through the Rest High Level Client

You can use the Rest High Level Client to connect to a non-security cluster and check whether the **test** index exists. The sample code is as follows:

```
import org.apache.http.HttpHost;
import org.elasticsearch.client.RequestOptions;
import org.elasticsearch.client.RestClient;
import org.elasticsearch.client.RestClientBuilder;
import org.elasticsearch.client.RestHighLevelClient;
import org.elasticsearch.client.indices.GetIndexRequest;

import java.io.IOException;
import java.util.Arrays;
import java.util.List;

/**
 * Use Rest Hive Level to connect to a non-security cluster.
 */
public class Main {
    public static void main(String[] args) throws IOException {
        List<String> host = Arrays.asList("x.x.x.x", "x.x.x.x");
        RestClientBuilder builder = RestClient.builder(constructHttpHosts(host, 9200, "http"));
        final RestHighLevelClient client = new RestHighLevelClient(builder);
        GetIndexRequest indexRequest = new GetIndexRequest("test");
        boolean exists = client.indices().exists(indexRequest, RequestOptions.DEFAULT);
        System.out.println(exists);
        client.close();
    }
}

/**
 * Use the constructHttpHosts function to convert the node IP address list of the host cluster.
```

```
*/
public static HttpHost[] constructHttpHosts(List<String> host, int port, String protocol) {
    return host.stream().map(p -> new HttpHost(p, port, protocol)).toArray(HttpHost[]::new);
}
}
```

host indicates the IP address list of each node in the cluster. If there are multiple IP addresses, separate them with commas (,). *test* indicates the index name to be queried.

Connecting to a Security Cluster Through Rest High Level Client (Without Security Certificates)

You can connect to a cluster in security mode+HTTP or a cluster in security mode + HTTPS (without using certificates).

The sample code is as follows:

```
import org.apache.http.HttpHost;
import org.apache.http.auth.AuthScope;
import org.apache.http.auth.UsernamePasswordCredentials;
import org.apache.http.client.CredentialsProvider;
import org.apache.http.impl.client.BasicCredentialsProvider;
import org.apache.http.impl.nio.client.HttpAsyncClientBuilder;
import org.apache.http.nio.conn.ssl.SSLIOStrategy;
import org.apache.logging.log4j.LogManager;
import org.apache.logging.log4j.Logger;
import org.elasticsearch.action.admin.cluster.health.ClusterHealthRequest;
import org.elasticsearch.action.admin.cluster.health.ClusterHealthResponse;
import org.elasticsearch.client.RequestOptions;
import org.elasticsearch.client.RestClient;
import org.elasticsearch.client.RestClientBuilder;
import org.elasticsearch.client.RestHighLevelClient;
import org.elasticsearch.client.indices.GetIndexRequest;
import org.elasticsearch.common.Nullable;

import java.io.IOException;
import java.security.KeyManagementException;
import java.security.NoSuchAlgorithmException;
import java.security.SecureRandom;
import java.security.cert.CertificateException;
import java.security.cert.X509Certificate;
import java.util.Arrays;
import java.util.List;
import java.util.Objects;

import javax.net.ssl.HostnameVerifier;
import javax.net.ssl.SSLContext;
import javax.net.ssl.SSLSession;
import javax.net.ssl.TrustManager;
import javax.net.ssl.X509TrustManager;

/**
 * Connect to a security cluster through Rest High Level (without using certificates).
 */
public class Main {
    /**
     * Create a class for the client. Define the create function.
     */
    public static RestHighLevelClient create(List<String> host, int port, String protocol, int connectTimeout,
int connectionRequestTimeout, int socketTimeout, String username, String password) throws IOException{
        final CredentialsProvider credentialsProvider = new BasicCredentialsProvider();
        credentialsProvider.setCredentials(AuthScope.ANY, new UsernamePasswordCredentials(username,
password));
        SSLContext sc = null;
        try {
            sc = SSLContext.getInstance("SSL");

```

```
        sc.init(null, trustAllCerts, new SecureRandom());
    } catch (KeyManagementException | NoSuchAlgorithmException e) {
        e.printStackTrace();
    }
    SSLIOSessionStrategy sessionStrategy = new SSLIOSessionStrategy(sc, new NullHostNameVerifier());
    SecuredHttpClientConfigCallback httpClientConfigCallback = new
SecuredHttpClientConfigCallback(sessionStrategy,
        credentialsProvider);

    RestClientBuilder builder = RestClient.builder(constructHttpHosts(host, port, protocol))
        .setRequestConfigCallback(requestConfig -> requestConfig.setConnectTimeout(connectTimeout))
        .setConnectionRequestTimeout(connectionRequestTimeout)
        .setSocketTimeout(socketTimeout)
        .setHttpClientConfigCallback(httpClientConfigCallback);
    final RestHighLevelClient client = new RestHighLevelClient(builder);
    logger.info("es rest client build success {}", client);

    ClusterHealthRequest request = new ClusterHealthRequest();
    ClusterHealthResponse response = client.cluster().health(request, RequestOptions.DEFAULT);
    logger.info("es rest client health response {}", response);
    return client;
}

/**
 * Use the constructHttpHosts function to convert the node IP address list of the host cluster.
 */
public static HttpHost[] constructHttpHosts(List<String> host, int port, String protocol) {
    return host.stream().map(p -> new HttpHost(p, port, protocol)).toArray(HttpHost[]::new);
}

/**
 * Configure trustAllCerts to ignore the certificate configuration.
 */
public static TrustManager[] trustAllCerts = new TrustManager[] {
    new X509TrustManager() {
        @Override
        public void checkClientTrusted(X509Certificate[] chain, String authType) throws
CertificateException {
        }

        @Override
        public void checkServerTrusted(X509Certificate[] chain, String authType) throws
CertificateException {
        }

        @Override
        public X509Certificate[] getAcceptedIssuers() {
            return null;
        }
    }
};

private static final Logger logger = LogManager.getLogger(Main.class);

static class SecuredHttpClientConfigCallback implements RestClientBuilder.HttpClientConfigCallback {
    @Nullable
    private final CredentialsProvider credentialsProvider;
    /**
     * The {@link SSLIOSessionStrategy} for all requests to enable SSL / TLS encryption.
     */
    private final SSLIOSessionStrategy sslStrategy;
    /**
     * Create a new {@link SecuredHttpClientConfigCallback}.
     *
     * @param credentialsProvider The credential provider, if a username/password have been supplied
     * @param sslStrategy The SSL strategy, if SSL / TLS have been supplied
     * @throws NullPointerException if {@code sslStrategy} is {@code null}
     */
}
```



```
SecuredHttpClientConfigCallback(final SSLIOStrategy sslStrategy,
    @Nullable final CredentialsProvider credentialsProvider) {
    this.sslStrategy = Objects.requireNonNull(sslStrategy);
    this.credentialsProvider = credentialsProvider;
}
/**
 * Get the {@link CredentialsProvider} that will be added to the HTTP client.
 *
 * @return Can be {@code null}.
 */
@Nullable
CredentialsProvider getCredentialsProvider() {
    return credentialsProvider;
}
/**
 * Get the {@link SSLIOStrategy} that will be added to the HTTP client.
 *
 * @return Never {@code null}.
 */
SSLIOStrategy getSSLStrategy() {
    return sslStrategy;
}
/**
 * Sets the {@linkplain HttpAsyncClientBuilder#setDefaultCredentialsProvider(CredentialsProvider)
credential provider},
 *
 * @param httpClientBuilder The client to configure.
 * @return Always {@code httpClientBuilder}.
 */
@Override
public HttpAsyncClientBuilder customizeHttpClient(final HttpAsyncClientBuilder httpClientBuilder) {
    // enable SSL / TLS
    httpClientBuilder.setSSLStrategy(sslStrategy);
    // enable user authentication
    if (credentialsProvider != null) {
        httpClientBuilder.setDefaultCredentialsProvider(credentialsProvider);
    }
    return httpClientBuilder;
}

public static class NullHostNameVerifier implements HostnameVerifier {
    @Override
    public boolean verify(String arg0, SSLSession arg1) {
        return true;
    }
}

/**
 * The following is an example of the main function. Call the create function to create a client and check
whether the test index exists.
 */
public static void main(String[] args) throws IOException {
    RestHighLevelClient client = create(Arrays.asList("x.x.x.x", "x.x.x.x"), 9200, "https", 1000, 1000, 1000,
"username", "password");
    GetIndexRequest indexRequest = new GetIndexRequest("test");
    boolean exists = client.indices().exists(indexRequest, RequestOptions.DEFAULT);
    System.out.println(exists);
    client.close();
}
}
```

Table 14-4 Variables

Parameter	Description
host	List of the IP addresses of Elasticsearch nodes (or independent Client node). Multiple IP addresses are separated using commas (,).
port	Access port of the Elasticsearch cluster. The default value is 9200 .
protocol	Connection protocol, which can be http or https .
connectTimeout	Socket connection timeout period.
connectionRequestTimeout	Timeout period of a socket connection request.
socketTimeout	Timeout period of a socket request.
username	Username for accessing the cluster.
password	Password of the user.

Connecting to a Security Cluster Through Rest High Level Client (With Security Certificates)

You can use a security certificate to connect to a cluster in security mode + HTTPS.

1. Obtain the security certificate **CloudSearchService.cer**.
 - a. Log in to the CSS management console.
 - b. In the navigation pane, choose **Clusters**. The cluster list is displayed.
 - c. Click the name of a cluster to go to the cluster details page.
 - d. On the **Configuration** page, click **Download Certificate** next to **HTTPS Access**.

2. Convert the security certificate **CloudSearchService.cer**. Upload the downloaded security certificate to the client and use keytool to convert the .cer certificate into a .jks certificate that can be read by Java.

- In Linux, run the following command to convert the certificate:
keytool -import -alias *newname* -keystore ./truststore.jks -file ./CloudSearchService.cer
- In Windows, run the following command to convert the certificate:
keytool -import -alias *newname* -keystore .\truststore.jks -file .\CloudSearchService.cer

In the preceding command, *newname* indicates the user-defined certificate name.

After this command is executed, you will be prompted to set the certificate password and confirm the password. Securely store the password. It will be used for accessing the cluster.

3. Access the cluster. The sample code is as follows:

```
import org.apache.http.HttpHost;
import org.apache.http.auth.AuthScope;
```

```
import org.apache.http.auth.UsernamePasswordCredentials;
import org.apache.http.client.CredentialsProvider;
import org.apache.http.conn.ssl.NoopHostnameVerifier;
import org.apache.http.impl.client.BasicCredentialsProvider;
import org.apache.http.impl.nio.client.HttpAsyncClientBuilder;
import org.apache.http.nio.conn.ssl.SSLIOStrategy;
import org.apache.logging.log4j.LogManager;
import org.apache.logging.log4j.Logger;
import org.elasticsearch.action.admin.cluster.health.ClusterHealthRequest;
import org.elasticsearch.action.admin.cluster.health.ClusterHealthResponse;
import org.elasticsearch.client.RequestOptions;
import org.elasticsearch.client.RestClient;
import org.elasticsearch.client.RestClientBuilder;
import org.elasticsearch.client.RestHighLevelClient;
import org.elasticsearch.client.indices.GetIndexRequest;
import org.elasticsearch.common.Nullable;

import java.io.File;
import java.io.FileInputStream;
import java.io.IOException;
import java.io.InputStream;
import java.security.KeyStore;
import java.security.SecureRandom;
import java.security.cert.CertificateException;
import java.security.cert.X509Certificate;
import java.util.Arrays;
import java.util.List;
import java.util.Objects;

import javax.net.ssl.SSLContext;
import javax.net.ssl.TrustManager;
import javax.net.ssl.TrustManagerFactory;
import javax.net.ssl.X509TrustManager;

/**
 * Use Rest Hive Level to connect to a security cluster (using an HTTPS certificate).
 */
public class Main {
    public static RestHighLevelClient create(List<String> host, int port, String protocol, int
connectTimeout, int connectionRequestTimeout, int socketTimeout, String username, String password,
String cerFilePath,
String cerPassword) throws IOException {

        final CredentialsProvider credentialsProvider = new BasicCredentialsProvider();
        credentialsProvider.setCredentials(AuthScope.ANY, new
UsernamePasswordCredentials(username, password));
        SSLContext sc = null;
        try {
            TrustManager[] tm = {new MyX509TrustManager(cerFilePath, cerPassword)};
            sc = SSLContext.getInstance("SSL", "SunJSSE");
            //You can also use SSLContext sslContext = SSLContext.getInstance("TLSv1.2");
            sc.init(null, tm, new SecureRandom());
        } catch (Exception e) {
            e.printStackTrace();
        }

        SSLIOStrategy sessionStrategy = new SSLIOStrategy(sc, new
NoopHostnameVerifier());
        SecuredHttpClientConfigCallback httpClientConfigCallback = new
SecuredHttpClientConfigCallback(sessionStrategy,
credentialsProvider);

        RestClientBuilder builder = RestClient.builder(constructHttpHosts(host, port, protocol))
            .setRequestConfigCallback(requestConfig ->
requestConfig.setConnectTimeout(connectTimeout)
                .setConnectionRequestTimeout(connectionRequestTimeout)
                .setSocketTimeout(socketTimeout))
            .setHttpClientConfigCallback(httpClientConfigCallback);
        final RestHighLevelClient client = new RestHighLevelClient(builder);
    }
}
```

```
logger.info("es rest client build success {}", client);

ClusterHealthRequest request = new ClusterHealthRequest();
ClusterHealthResponse response = client.cluster().health(request, RequestOptions.DEFAULT);
logger.info("es rest client health response {}", response);
return client;
}

/**
 * Use the constructHttpHosts function to convert the node IP address list of the host cluster.
 */
public static HttpHost[] constructHttpHosts(List<String> host, int port, String protocol) {
    return host.stream().map(p -> new HttpHost(p, port, protocol)).toArray(HttpHost[]::new);
}

/**
 * SecuredHttpClientConfigCallback class definition
 */
static class SecuredHttpClientConfigCallback implements
RestClientBuilder.HttpClientConfigCallback {
    @Nullable
    private final CredentialsProvider credentialsProvider;

    private final SSLIOSessionStrategy sslStrategy;

    SecuredHttpClientConfigCallback(final SSLIOSessionStrategy sslStrategy,
        @Nullable final CredentialsProvider credentialsProvider) {
        this.sslStrategy = Objects.requireNonNull(sslStrategy);
        this.credentialsProvider = credentialsProvider;
    }

    @Nullable
    CredentialsProvider getCredentialsProvider() {
        return credentialsProvider;
    }

    SSLIOSessionStrategy getSSLStrategy() {
        return sslStrategy;
    }

    @Override
    public HttpAsyncClientBuilder customizeHttpClient(final HttpAsyncClientBuilder
httpClientBuilder) {
        httpClientBuilder.setSSLStrategy(sslStrategy);
        if (credentialsProvider != null) {
            httpClientBuilder.setDefaultCredentialsProvider(credentialsProvider);
        }
        return httpClientBuilder;
    }
}

private static final Logger logger = LogManager.getLogger(Main.class);

public static class MyX509TrustManager implements X509TrustManager {
    X509TrustManager sunJSSEX509TrustManager;

    MyX509TrustManager(String cerFilePath, String cerPassword) throws Exception {
        File file = new File(cerFilePath);
        if (!file.isFile()) {
            throw new Exception("Wrong Certification Path");
        }
        System.out.println("Loading KeyStore " + file + "...");
        InputStream in = new FileInputStream(file);
        KeyStore ks = KeyStore.getInstance("JKS");
        ks.load(in, cerPassword.toCharArray());
        TrustManagerFactory tmf = TrustManagerFactory.getInstance("SunX509", "SunJSSE");
        tmf.init(ks);
        TrustManager[] tms = tmf.getTrustManagers();
        for (TrustManager tm : tms) {
```

```

        if (tm instanceof X509TrustManager) {
            sunJSSEX509TrustManager = (X509TrustManager) tm;
            return;
        }
        throw new Exception("Couldn't initialize");
    }

    @Override
    public void checkClientTrusted(X509Certificate[] chain, String authType) throws
CertificateException {

    }

    @Override
    public void checkServerTrusted(X509Certificate[] chain, String authType) throws
CertificateException {

    }

    @Override
    public X509Certificate[] getAcceptedIssuers() {
        return new X509Certificate[0];
    }
}

/**
 * The following is an example of the main function. Call the create function to create a client and
 * check whether the test index exists.
 */
public static void main(String[] args) throws IOException {
    RestHighLevelClient client = create(Arrays.asList("xxx.xxx.xxx.xxx", "xxx.xxx.xxx.xxx"), 9200,
"https", 1000, 1000, 1000, "username", "password", "cerFilePath", "cerPassword");
    GetIndexRequest indexRequest = new GetIndexRequest("test");
    boolean exists = client.indices().exists(indexRequest, RequestOptions.DEFAULT);
    System.out.println(exists);
    client.close();
}
}

```

Table 14-5 Function parameters

Name	Description
host	List of the IP addresses of Elasticsearch nodes (or independent Client node). Multiple IP addresses are separated using commas (,).
port	Access port of the Elasticsearch cluster. The default value is 9200 .
protocol	Connection protocol. Set this parameter to https .
connectTimeout	Socket connection timeout period.
connectionRequestTimeout	Timeout period of a socket connection request.
socketTimeout	Timeout period of a socket request.
username	Username for accessing the cluster.

Name	Description
password	Password of the user.
cerFilePath	Certificate path.
cerPassword	Certificate password.

14.2.3.2 Accessing a Cluster Through the Rest Low Level Client

The high-level client is encapsulated based on the low-level client. If the method calls (such as `.search` and `.bulk`) in the high-level client cannot meet the requirements or has compatibility issues, you can use the low-level client. You can even use `HighLevelClient.getLowLevelClient()` to directly obtain a low-level client. A low-level client allows you to define the request structure, which is more flexible and supports all the request formats of Elasticsearch, such as GET, POST, DELETE, and HEAD.

This section describes how to use the Rest Low Level Client to access the CSS cluster. The methods are as follows. For each method, you can directly create a REST low-level client, or create a high-level client and then invoke `getLowLevelClient()` to obtain a low-level client.

- **Connecting to a Non-Security Cluster Through the Rest Low Level Client:** applicable to clusters in non-security mode
- **Connecting to a Security Cluster Through Rest Low Level Client (Without Security Certificates):** applicable to clusters in security mode+HTTP, and to clusters in security mode+HTTPS (without using certificates)
- **Connecting to a Security Cluster Through Rest Low Level Client (With Security Certificates):** applicable to clusters in security mode+HTTPS

Precautions

You are advised to use the Rest Low Level Client version that matches the Elasticsearch version. For example, use Rest Low Level Client 7.6.2 to access the Elasticsearch cluster 7.6.2.

Prerequisites

- The CSS cluster is available.
- Ensure that the server running Java can communicate with the CSS cluster.
- Install JDK 1.8 on the server. You can download JDK 1.8 from: <https://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html>
- Declare the Apache version in Maven mode. The following code uses version 7.6.2 as an example.

7.6.2 indicates the version of the Elasticsearch Java client.

```
<dependency>
  <groupId>org.elasticsearch.client</groupId>
  <artifactId>elasticsearch-rest-client</artifactId>
  <version>7.6.2</version>
</dependency>
```

```
<dependency>
  <groupId>org.elasticsearch</groupId>
  <artifactId>elasticsearch</artifactId>
  <version>7.6.2</version>
</dependency>
```

Connecting to a Non-Security Cluster Through the Rest Low Level Client

- **Method 1: Directly create a Rest Low Level Client.**

```
import org.apache.http.HttpHost;
import org.elasticsearch.client.Request;
import org.elasticsearch.client.Response;
import org.elasticsearch.client.RestClient;
import org.elasticsearch.client.RestClientBuilder;

import java.io.IOException;
import java.util.Arrays;
import java.util.List;

public class Main {

    public static void main(String[] args) throws IOException {
        List<String> host = Arrays.asList("xxx.xxx.xxx.xxx", "xxx.xxx.xxx.xxx");
        RestClientBuilder builder = RestClient.builder(constructHttpHosts(host, 9200, "http"));
        /**
         * Create a Rest Low Level Client.
         */
        RestClient lowLevelClient = builder.build();
        /**
         * Check whether the test index exists. If the index exists, 200 is returned. If the index does not
         exist, 404 is returned.
         */
        Request request = new Request("HEAD", "/test");
        Response response = lowLevelClient.performRequest(request);
        System.out.println(response.getStatusLine().getStatusCode());
        lowLevelClient.close();
    }

    /**
     * Use the constructHttpHosts function to convert the node IP address list of the host cluster.
     */
    public static HttpHost[] constructHttpHosts(List<String> host, int port, String protocol) {
        return host.stream().map(p -> new HttpHost(p, port, protocol)).toArray(HttpHost[]::new);
    }
}
```

- **Method 2: Create a high-level client and then call `getLowLevelClient()` to obtain a low-level client.**

```
import org.apache.http.HttpHost;
import org.elasticsearch.client.Request;
import org.elasticsearch.client.Response;
import org.elasticsearch.client.RestClient;
import org.elasticsearch.client.RestClientBuilder;
import org.elasticsearch.client.RestHighLevelClient;

import java.io.IOException;
import java.util.Arrays;
import java.util.List;

public class Main {

    public static void main(String[] args) throws IOException {
        List<String> host = Arrays.asList("xxx.xxx.xxx.xxx", "xxx.xxx.xxx.xxx");
        RestClientBuilder builder = RestClient.builder(constructHttpHosts(host, 9200, "http"));
        final RestHighLevelClient restHighLevelClient = new RestHighLevelClient(builder);
        /**
         * Create a high-level client and then call getLowLevelClient() to obtain a low-level client.
         The code differs from the client creation code only in the following line:
         */
    }
}
```

```
    final RestClient lowLevelClient = restHighLevelClient.getLowLevelClient();
    /**
     * Check whether the test index exists. If the index exists, 200 is returned. If the index does not
    exist, 404 is returned.
     */
    Request request = new Request("HEAD", "/" + test);
    Response response = lowLevelClient.performRequest(request);
    System.out.println(response.getStatusLine().getStatusCode());
    lowLevelClient.close();
}

/**
 * Use the constructHttpHosts function to convert the node IP address list of the host cluster.
 */
public static HttpHost[] constructHttpHosts(List<String> host, int port, String protocol) {
    return host.stream().map(p -> new HttpHost(p, port, protocol)).toArray(HttpHost[]::new);
}
}
```

host indicates the IP address list of each node in the cluster. If there are multiple IP addresses, separate them with commas (,). *test* indicates the index name to be queried.

Connecting to a Security Cluster Through Rest Low Level Client (Without Security Certificates)

- **Method 1: Directly create a Rest Low Level Client.**

```
import org.apache.http.HttpHost;
import org.apache.http.HttpResponse;
import org.apache.http.auth.AuthScope;
import org.apache.http.auth.UsernamePasswordCredentials;
import org.apache.http.client.CredentialsProvider;
import org.apache.http.impl.client.BasicCredentialsProvider;
import org.apache.http.impl.client.DefaultConnectionKeepAliveStrategy;
import org.apache.http.impl.nio.client.HttpAsyncClientBuilder;
import org.apache.http.nio.conn.ssl.SSLIOStrategy;
import org.apache.http.protocol.HttpContext;
import org.apache.logging.log4j.LogManager;
import org.apache.logging.log4j.Logger;
import org.elasticsearch.client.Request;
import org.elasticsearch.client.Response;
import org.elasticsearch.client.RestClient;
import org.elasticsearch.client.RestClientBuilder;
import org.elasticsearch.common.Nullable;

import java.io.IOException;
import java.security.KeyManagementException;
import java.security.NoSuchAlgorithmException;
import java.security.SecureRandom;
import java.security.cert.CertificateException;
import java.security.cert.X509Certificate;
import java.util.Arrays;
import java.util.List;
import java.util.Objects;
import java.util.concurrent.TimeUnit;

import javax.net.ssl.HostnameVerifier;
import javax.net.ssl.SSLContext;
import javax.net.ssl.SSLSession;
import javax.net.ssl.TrustManager;
import javax.net.ssl.X509TrustManager;

public class Main {

    /**
     * Create a class for the client. Define the create function.
     */
    public static RestClient create(List<String> host, int port, String protocol, int connectTimeout, int
    connectionRequestTimeout, int socketTimeout, String username, String password) throws
```



```
IOException {
    final CredentialsProvider credentialsProvider = new BasicCredentialsProvider();
    credentialsProvider.setCredentials(AuthScope.ANY, new
UsernamePasswordCredentials(username, password));
    SSLContext sc = null;
    try {
        sc = SSLContext.getInstance("SSL");
        sc.init(null, trustAllCerts, new SecureRandom());
    } catch (KeyManagementException | NoSuchAlgorithmException e) {
        e.printStackTrace();
    }
    SSLIOSessionStrategy sessionStrategy = new SSLIOSessionStrategy(sc, new
NullHostNameVerifier());
    SecuredHttpClientConfigCallback httpClientConfigCallback = new
SecuredHttpClientConfigCallback(sessionStrategy,
credentialsProvider);

    RestClientBuilder builder = RestClient.builder(constructHttpHosts(host, port, protocol))
        .setRequestConfigCallback(requestConfig ->
requestConfig.setConnectTimeout(connectTimeout)
        .setConnectionRequestTimeout(connectionRequestTimeout)
        .setSocketTimeout(socketTimeout))
        .setHttpClientConfigCallback(httpClientConfigCallback);
    final RestClient client = builder.build();
    logger.info("es rest client build success {}", client);
    return client;
}

/**
 * Use the constructHttpHosts function to convert the node IP address list of the host cluster.
 */
public static HttpHost[] constructHttpHosts(List<String> host, int port, String protocol) {
    return host.stream().map(p -> new HttpHost(p, port, protocol)).toArray(HttpHost[]::new);
}

/**
 * Configure trustAllCerts to ignore the certificate configuration.
 */
public static TrustManager[] trustAllCerts = new TrustManager[] {
    new X509TrustManager() {
        @Override
        public void checkClientTrusted(X509Certificate[] chain, String authType) throws
CertificateException {
        }

        @Override
        public void checkServerTrusted(X509Certificate[] chain, String authType) throws
CertificateException {
        }

        @Override
        public X509Certificate[] getAcceptedIssuers() {
            return null;
        }
    }
};

/**
 * The CustomConnectionKeepAliveStrategy function is used to set the connection keepalive time when
there are a large number of short connections or when the number of data requests is small.
 */
public static class CustomConnectionKeepAliveStrategy extends
DefaultConnectionKeepAliveStrategy {
    public static final CustomConnectionKeepAliveStrategy INSTANCE = new
CustomConnectionKeepAliveStrategy();

    private CustomConnectionKeepAliveStrategy() {
        super();
    }
}
```

```
/**
 * Maximum keep alive time (minutes)
 * The default value is 10 minutes. You can set it based on the number of TCP connections in
TIME_WAIT state. If there are too many TCP connections, you can increase the value.
 */
private final long MAX_KEEP_ALIVE_MINUTES = 10;

@Override
public long getKeepAliveDuration(HttpResponse response, HttpContext context) {
    long keepAliveDuration = super.getKeepAliveDuration(response, context);
    // <0 indicates that the keepalive period is unlimited.
    // Change the period from unlimited to a default period.
    if (keepAliveDuration < 0) {
        return TimeUnit.MINUTES.toMillis(MAX_KEEP_ALIVE_MINUTES);
    }
    return keepAliveDuration;
}
}

private static final Logger logger = LogManager.getLogger(Main.class);

static class SecuredHttpClientConfigCallback implements
RestClientBuilder.HttpClientConfigCallback {
    @Nullable
    private final CredentialsProvider credentialsProvider;
    /**
     * The {@link SSLIOStrategy} for all requests to enable SSL / TLS encryption.
     */
    private final SSLIOStrategy sslStrategy;
    /**
     * Create a new {@link SecuredHttpClientConfigCallback}.
     *
     * @param credentialsProvider The credential provider, if a username/password have been
supplied
     * @param sslStrategy The SSL strategy, if SSL / TLS have been supplied
     * @throws NullPointerException if {@code sslStrategy} is {@code null}
     */
    SecuredHttpClientConfigCallback(final SSLIOStrategy sslStrategy,
        @Nullable final CredentialsProvider credentialsProvider) {
        this.sslStrategy = Objects.requireNonNull(sslStrategy);
        this.credentialsProvider = credentialsProvider;
    }
    /**
     * Get the {@link CredentialsProvider} that will be added to the HTTP client.
     *
     * @return Can be {@code null}.
     */
    @Nullable
    CredentialsProvider getCredentialsProvider() {
        return credentialsProvider;
    }
    /**
     * Get the {@link SSLIOStrategy} that will be added to the HTTP client.
     *
     * @return Never {@code null}.
     */
    SSLIOStrategy getSSLStrategy() {
        return sslStrategy;
    }
    /**
     * Sets the {@linkplain
HttpAsyncClientBuilder#setDefaultCredentialsProvider(CredentialsProvider) credential provider},
     *
     * @param httpClientBuilder The client to configure.
     * @return Always {@code httpClientBuilder}.
     */
    @Override
    public HttpAsyncClientBuilder customizeHttpClient(final HttpAsyncClientBuilder
```

```
httpClientBuilder) {
    // enable SSL / TLS
    httpClientBuilder.setSSLStrategy(sslStrategy);
    // enable user authentication
    if (credentialsProvider != null) {
        httpClientBuilder.setDefaultCredentialsProvider(credentialsProvider);
    }
    return httpClientBuilder;
}
}

public static class NullHostNameVerifier implements HostnameVerifier {
    @Override
    public boolean verify(String arg0, SSLSession arg1) {
        return true;
    }
}

/**
 * The following is an example of the main function. Call the create function to create a Rest Low
 * Level Client and check whether the test index exists.
 */
public static void main(String[] args) throws IOException {
    RestClient lowLevelClient = create(Arrays.asList("xxx.xxx.xxx.xxx", "xxx.xxx.xxx.xxx"), 9200, "http",
1000, 1000, 1000, "username", "password");
    Request request = new Request("HEAD", "/test");
    Response response = lowLevelClient.performRequest(request);
    System.out.println(response.getStatusLine().getStatusCode());
    lowLevelClient.close();
}
}
```

- **Method 2: Create a high-level client and then call `getLowLevelClient()` to obtain a low-level client.**

```
import org.apache.http.HttpHost;
import org.apache.http.HttpResponse;
import org.apache.http.auth.AuthScope;
import org.apache.http.auth.UsernamePasswordCredentials;
import org.apache.http.client.CredentialsProvider;
import org.apache.http.impl.client.BasicCredentialsProvider;
import org.apache.http.impl.client.DefaultConnectionKeepAliveStrategy;
import org.apache.http.impl.nio.client.HttpAsyncClientBuilder;
import org.apache.http.nio.conn.ssl.SSLIOSessionStrategy;
import org.apache.http.protocol.HttpContext;
import org.apache.logging.log4j.LogManager;
import org.apache.logging.log4j.Logger;
import org.elasticsearch.client.Request;
import org.elasticsearch.client.Response;
import org.elasticsearch.client.RestClient;
import org.elasticsearch.client.RestClientBuilder;
import org.elasticsearch.common.Nullable;

import java.io.IOException;
import java.security.KeyManagementException;
import java.security.NoSuchAlgorithmException;
import java.security.SecureRandom;
import java.security.cert.CertificateException;
import java.security.cert.X509Certificate;
import java.util.Arrays;
import java.util.List;
import java.util.Objects;
import java.util.concurrent.TimeUnit;

import javax.net.ssl.HostnameVerifier;
import javax.net.ssl.SSLContext;
import javax.net.ssl.SSLSession;
import javax.net.ssl.TrustManager;
import javax.net.ssl.X509TrustManager;

import org.elasticsearch.client.RestHighLevelClient;
```

```
public class Main13 {  
  
    /**  
     * Create a class for the client. Define the create function.  
     */  
    public static RestHighLevelClient create(List<String> host, int port, String protocol, int  
connectTimeout, int connectionRequestTimeout, int socketTimeout, String username, String  
password) throws IOException {  
  
        final CredentialsProvider credentialsProvider = new BasicCredentialsProvider();  
        credentialsProvider.setCredentials(AuthScope.ANY, new  
UsernamePasswordCredentials(username, password));  
        SSLContext sc = null;  
        try {  
            sc = SSLContext.getInstance("SSL");  
            sc.init(null, trustAllCerts, new SecureRandom());  
        } catch (KeyManagementException | NoSuchAlgorithmException e) {  
            e.printStackTrace();  
        }  
        SSLIOSessionStrategy sessionStrategy = new SSLIOSessionStrategy(sc, new  
NullHostNameVerifier());  
        SecuredHttpClientConfigCallback httpClientConfigCallback = new  
SecuredHttpClientConfigCallback(sessionStrategy,  
credentialsProvider);  
  
        RestClientBuilder builder = RestClient.builder(constructHttpHosts(host, port, protocol))  
            .setRequestConfigCallback(requestConfig ->  
requestConfig.setConnectTimeout(connectTimeout)  
                .setConnectionRequestTimeout(connectionRequestTimeout)  
                .setSocketTimeout(socketTimeout))  
            .setHttpClientConfigCallback(httpClientConfigCallback);  
        final RestHighLevelClient client = new RestHighLevelClient(builder);  
        logger.info("es rest client build success {}", client);  
        return client;  
    }  
  
    /**  
     * Use the constructHttpHosts function to convert the node IP address list of the host cluster.  
     */  
    public static HttpHost[] constructHttpHosts(List<String> host, int port, String protocol) {  
        return host.stream().map(p -> new HttpHost(p, port, protocol)).toArray(HttpHost[]::new);  
    }  
  
    /**  
     * Configure trustAllCerts to ignore the certificate configuration.  
     */  
    public static TrustManager[] trustAllCerts = new TrustManager[] {  
        new X509TrustManager() {  
            @Override  
            public void checkClientTrusted(X509Certificate[] chain, String authType) throws  
CertificateException {  
            }  
  
            @Override  
            public void checkServerTrusted(X509Certificate[] chain, String authType) throws  
CertificateException {  
            }  
  
            @Override  
            public X509Certificate[] getAcceptedIssuers() {  
                return null;  
            }  
        }  
    };  
  
    /**  
     * The CustomConnectionKeepAliveStrategy function is used to set the connection keepalive time when  
there are a large number of short connections or when the number of data requests is small.  
     */  
}
```

```
public static class CustomConnectionKeepAliveStrategy extends
DefaultConnectionKeepAliveStrategy {
    public static final CustomConnectionKeepAliveStrategy INSTANCE = new
CustomConnectionKeepAliveStrategy();

    private CustomConnectionKeepAliveStrategy() {
        super();
    }

    /**
     * Maximum keep alive time (minutes)
     * The default value is 10 minutes. You can set it based on the number of TCP connections in
TIME_WAIT state. If there are too many TCP connections, you can increase the value.
     */
    private final long MAX_KEEP_ALIVE_MINUTES = 10;

    @Override
    public long getKeepAliveDuration(HttpResponse response, HttpContext context) {
        long keepAliveDuration = super.getKeepAliveDuration(response, context);
        // <0 indicates that the keepalive period is unlimited.
        // Change the period from unlimited to a default period.
        if (keepAliveDuration < 0) {
            return TimeUnit.MINUTES.toMillis(MAX_KEEP_ALIVE_MINUTES);
        }
        return keepAliveDuration;
    }
}

private static final Logger logger = LogManager.getLogger(Main.class);

static class SecuredHttpClientConfigCallback implements
RestClientBuilder.HttpClientConfigCallback {
    @Nullable
    private final CredentialsProvider credentialsProvider;
    /**
     * The {@link SSLIOSessionStrategy} for all requests to enable SSL / TLS encryption.
     */
    private final SSLIOSessionStrategy sslStrategy;
    /**
     * Create a new {@link SecuredHttpClientConfigCallback}.
     *
     * @param credentialsProvider The credential provider, if a username/password have been
supplied
     * @param sslStrategy The SSL strategy, if SSL / TLS have been supplied
     * @throws NullPointerException if {@code sslStrategy} is {@code null}
     */
    SecuredHttpClientConfigCallback(final SSLIOSessionStrategy sslStrategy,
        @Nullable final CredentialsProvider credentialsProvider) {
        this.sslStrategy = Objects.requireNonNull(sslStrategy);
        this.credentialsProvider = credentialsProvider;
    }
    /**
     * Get the {@link CredentialsProvider} that will be added to the HTTP client.
     *
     * @return Can be {@code null}.
     */
    @Nullable
    CredentialsProvider getCredentialsProvider() {
        return credentialsProvider;
    }
    /**
     * Get the {@link SSLIOSessionStrategy} that will be added to the HTTP client.
     *
     * @return Never {@code null}.
     */
    SSLIOSessionStrategy getSSLStrategy() {
        return sslStrategy;
    }
    /**
```

```

    * Sets the {@linkplain
HttpAsyncClientBuilder#setDefaultCredentialsProvider(CredentialsProvider) credential provider},
    *
    * @param httpClientBuilder The client to configure.
    * @return Always {@code httpClientBuilder}.
    */
    @Override
    public HttpAsyncClientBuilder customizeHttpClient(final HttpAsyncClientBuilder
httpClientBuilder) {
        // enable SSL / TLS
        httpClientBuilder.setSSLStrategy(sslStrategy);
        // enable user authentication
        if (credentialsProvider != null) {
            httpClientBuilder.setDefaultCredentialsProvider(credentialsProvider);
        }
        return httpClientBuilder;
    }
}

public static class NullHostNameVerifier implements HostnameVerifier {
    @Override
    public boolean verify(String arg0, SSLSession arg1) {
        return true;
    }
}

/**
 * The following is an example of the main function. Call the create function to create a high-level
client, call the getLowLevelClient() function to obtain a low-level client, and check whether the test
index exists.
 */
public static void main(String[] args) throws IOException {
    RestHighLevelClient client = create(Arrays.asList("xxx.xxx.xxx.xxx", "xxx.xxx.xxx.xxx"), 9200,
"http", 1000, 1000, 1000, "username", "password");
    RestClient lowLevelClient = client.getLowLevelClient();
    Request request = new Request("HEAD", "test");
    Response response = lowLevelClient.performRequest(request);
    System.out.println(response.getStatusLine().getStatusCode());
    lowLevelClient.close();
}
}

```

Table 14-6 Variables

Parameter	Description
host	List of the IP addresses of Elasticsearch nodes (or independent Client node). Multiple IP addresses are separated using commas (,).
port	Access port of the Elasticsearch cluster. The default value is 9200 .
protocol	Connection protocol, which can be http or https .
connectTimeout	Socket connection timeout period.
connectionRequestTimeout	Timeout period of a socket connection request.
socketTimeout	Timeout period of a socket request.

Parameter	Description
username	Username for accessing the cluster.
password	Password of the user.

Connecting to a Security Cluster Through Rest Low Level Client (With Security Certificates)

- **Method 1: Directly create a Rest Low Level Client.**

```
import org.apache.http.HttpHost;
import org.apache.http.auth.AuthScope;
import org.apache.http.auth.UsernamePasswordCredentials;
import org.apache.http.client.CredentialsProvider;
import org.apache.http.conn.ssl.NoopHostnameVerifier;
import org.apache.http.impl.client.BasicCredentialsProvider;
import org.apache.http.impl.nio.client.HttpAsyncClientBuilder;
import org.apache.http.nio.conn.ssl.SSLIOStrategy;
import org.apache.logging.log4j.LogManager;
import org.apache.logging.log4j.Logger;
import org.elasticsearch.client.Request;
import org.elasticsearch.client.Response;
import org.elasticsearch.client.RestClient;
import org.elasticsearch.client.RestClientBuilder;
import org.elasticsearch.common.Nullable;

import java.io.File;
import java.io.FileInputStream;
import java.io.IOException;
import java.io.InputStream;
import java.io.OutputStream;
import java.security.KeyStore;
import java.security.SecureRandom;
import java.security.cert.CertificateException;
import java.security.cert.X509Certificate;
import java.util.Arrays;
import java.util.List;
import java.util.Objects;

import javax.net.ssl.SSLContext;import javax.net.ssl.TrustManager;
import javax.net.ssl.TrustManagerFactory;
import javax.net.ssl.X509TrustManager;

public class Main13 {

    private static final Logger logger = LogManager.getLogger(Main.class);

    /**
     * Create a class for the client. Define the create function.
     */
    public static RestClient create(List<String> host, int port, String protocol, int connectTimeout, int
connectionRequestTimeout, int socketTimeout, String username, String password, String cerFilePath,
String cerPassword) throws IOException {
        final CredentialsProvider credentialsProvider = new BasicCredentialsProvider();
        credentialsProvider.setCredentials(AuthScope.ANY, new
UsernamePasswordCredentials(username, password));
        SSLContext sc = null;
        try {
            TrustManager[] tm = {new MyX509TrustManager(cerFilePath, cerPassword)};
            sc = SSLContext.getInstance("SSL", "SunJSSE");
            //You can also use SSLContext sslContext = SSLContext.getInstance("TLSv1.2");
            sc.init(null, tm, new SecureRandom());
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

```
SSLIOSessionStrategy sessionStrategy = new SSLIOSessionStrategy(sc, new
NoopHostnameVerifier());
SecuredHttpClientConfigCallback httpClientConfigCallback = new
SecuredHttpClientConfigCallback(sessionStrategy,
credentialsProvider);

RestClientBuilder builder = RestClient.builder(constructHttpHosts(host, port, protocol))
.setRequestConfigCallback(requestConfig ->
requestConfig.setConnectTimeout(connectTimeout)
.setConnectionRequestTimeout(connectionRequestTimeout)
.setSocketTimeout(socketTimeout))
.setHttpClientConfigCallback(httpClientConfigCallback);
final RestClient client = builder.build();
logger.info("es rest client build success {}", client);
return client;
}

/**
 * Use the constructHttpHosts function to convert the node IP address list of the host cluster.
 */
public static HttpHost[] constructHttpHosts(List<String> host, int port, String protocol) {
return host.stream().map(p -> new HttpHost(p, port, protocol)).toArray(HttpHost[]::new);}

static class SecuredHttpClientConfigCallback implements
RestClientBuilder.HttpClientConfigCallback {
@Nullable
private final CredentialsProvider credentialsProvider;

private final SSLIOSessionStrategy sslStrategy;

SecuredHttpClientConfigCallback(final SSLIOSessionStrategy sslStrategy,
@Nullable final CredentialsProvider credentialsProvider) {
this.sslStrategy = Objects.requireNonNull(sslStrategy);
this.credentialsProvider = credentialsProvider;
}

@Nullable
CredentialsProvider getCredentialsProvider() {
return credentialsProvider;
}

SSLIOSessionStrategy getSSLStrategy() {
return sslStrategy;
}

@Override
public HttpAsyncClientBuilder customizeHttpClient(final HttpAsyncClientBuilder
httpClientBuilder) {
httpClientBuilder.setSSLStrategy(sslStrategy);
if (credentialsProvider != null) {
httpClientBuilder.setDefaultCredentialsProvider(credentialsProvider);
}
return httpClientBuilder;
}}

public static class MyX509TrustManager implements X509TrustManager {
X509TrustManager sunJSSEX509TrustManager;

MyX509TrustManager(String cerFilePath, String cerPassword) throws Exception {
File file = new File(cerFilePath);
if (!file.isFile()) {
throw new Exception("Wrong Certification Path");
}
System.out.println("Loading KeyStore " + file + "...");
InputStream in = new FileInputStream(file);
KeyStore ks = KeyStore.getInstance("JKS");
ks.load(in, cerPassword.toCharArray());
TrustManagerFactory tmf = TrustManagerFactory.getInstance("SunX509", "SunJSSE");
tmf.init(ks);
}
```



```
TrustManager[] tms = tmf.getTrustManagers();
for (TrustManager tm : tms) {
    if (tm instanceof X509TrustManager) {
        sunJSSEX509TrustManager = (X509TrustManager) tm;
        return;
    }
}
throw new Exception("Couldn't initialize");
}

@Override
public void checkClientTrusted(X509Certificate[] chain, String authType) throws
CertificateException {

}

@Override
public void checkServerTrusted(X509Certificate[] chain, String authType) throws
CertificateException {

}

@Override
public X509Certificate[] getAcceptedIssuers() {
    return new X509Certificate[0];
}
}

/**
 * The following is an example of the main function. Call the create function to create a Rest Low
 * Level Client and check whether the test index exists.
 */
public static void main(String[] args) throws IOException {
    RestClient lowLevelClient = create(Arrays.asList("xxx.xxx.xxx.xxx", "xxx.xxx.xxx.xxx"), 9200,
"https", 1000, 1000, 1000, "username", "password", "cerFilePath", "cerPassword");
    Request request = new Request("HEAD", "test");
    Response response = lowLevelClient.performRequest(request);
    System.out.println(response.getStatusLine().getStatusCode());
    lowLevelClient.close();
}
}
```

- **Method 2: Create a high-level client and then call `getLowLevelClient()` to obtain a low-level client.**

```
import org.apache.http.HttpHost;
import org.apache.http.auth.AuthScope;
import org.apache.http.auth.UsernamePasswordCredentials;
import org.apache.http.client.CredentialsProvider;
import org.apache.http.conn.ssl.NoopHostnameVerifier;
import org.apache.http.impl.client.BasicCredentialsProvider;
import org.apache.http.impl.nio.client.HttpAsyncClientBuilder;
import org.apache.http.nio.conn.ssl.SSLIOStrategy;
import org.apache.logging.log4j.LogManager;
import org.apache.logging.log4j.Logger;
import org.elasticsearch.action.admin.cluster.health.ClusterHealthRequest;
import org.elasticsearch.action.admin.cluster.health.ClusterHealthResponse;
import org.elasticsearch.client.Request;
import org.elasticsearch.client.RequestOptions;
import org.elasticsearch.client.Response;
import org.elasticsearch.client.RestClient;
import org.elasticsearch.client.RestClientBuilder;
import org.elasticsearch.client.RestHighLevelClient;
import org.elasticsearch.common.Nullable;

import java.io.File;
import java.io.FileInputStream;
import java.io.IOException;
import java.io.InputStream;
import java.security.KeyStore;
import java.security.SecureRandom;
```

```
import java.security.cert.CertificateException;
import java.security.cert.X509Certificate;
import java.util.Arrays;
import java.util.List;
import java.util.Objects;

import javax.net.ssl.SSLContext;import javax.net.ssl.TrustManager;
import javax.net.ssl.TrustManagerFactory;
import javax.net.ssl.X509TrustManager;

public class Main {

    private static final Logger logger = LogManager.getLogger(Main.class);

    /**
     * Create a class for the client. Define the create function.
     */
    public static RestHighLevelClient create(List<String> host, int port, String protocol, int
connectTimeout, int connectionRequestTimeout, int socketTimeout, String username, String password,
String cerFilePath, String cerPassword) throws IOException {
        final CredentialsProvider credentialsProvider = new BasicCredentialsProvider();
        credentialsProvider.setCredentials(AuthScope.ANY, new
UsernamePasswordCredentials(username, password));
        SSLContext sc = null;
        try {
            TrustManager[] tm = {new MyX509TrustManager(cerFilePath, cerPassword)};
            sc = SSLContext.getInstance("SSL", "SunJSSE");
            //You can also use SSLContext sslContext = SSLContext.getInstance("TLSv1.2");
            sc.init(null, tm, new SecureRandom());
        } catch (Exception e) {
            e.printStackTrace();
        }

        SSLIOSessionStrategy sessionStrategy = new SSLIOSessionStrategy(sc, new
NoopHostnameVerifier());
        SecuredHttpClientConfigCallback httpClientConfigCallback = new
SecuredHttpClientConfigCallback(sessionStrategy,
credentialsProvider);

        RestClientBuilder builder = RestClient.builder(constructHttpHosts(host, port, protocol))
        .setRequestConfigCallback(requestConfig ->
requestConfig.setConnectTimeout(connectTimeout)
        .setConnectionRequestTimeout(connectionRequestTimeout)
        .setSocketTimeout(socketTimeout))
        .setHttpClientConfigCallback(httpClientConfigCallback);
        final RestHighLevelClient client = new RestHighLevelClient(builder);
        logger.info("es rest client build success {}", client);

        ClusterHealthRequest request = new ClusterHealthRequest();
        ClusterHealthResponse response = client.cluster().health(request, RequestOptions.DEFAULT);
        logger.info("es rest client health response {}", response);
        return client;
    }

    /**
     * Use the constructHttpHosts function to convert the node IP address list of the host cluster.
     */
    public static HttpHost[] constructHttpHosts(List<String> host, int port, String protocol) {
        return host.stream().map(p -> new HttpHost(p, port, protocol)).toArray(HttpHost[]::new);}

    static class SecuredHttpClientConfigCallback implements
RestClientBuilder.HttpClientConfigCallback {
        @Nullable
        private final CredentialsProvider credentialsProvider;

        private final SSLIOSessionStrategy sslStrategy;

        SecuredHttpClientConfigCallback(final SSLIOSessionStrategy sslStrategy,
@Nullable final CredentialsProvider credentialsProvider) {
```

```
        this.sslStrategy = Objects.requireNonNull(sslStrategy);
        this.credentialsProvider = credentialsProvider;
    }

    @Nullable
    CredentialsProvider getCredentialsProvider() {
        return credentialsProvider;
    }

    SSLIOSessionStrategy getSSLStrategy() {
        return sslStrategy;
    }

    @Override
    public HttpAsyncClientBuilder customizeHttpClient(final HttpAsyncClientBuilder
httpClientBuilder) {
        httpClientBuilder.setSSLStrategy(sslStrategy);
        if (credentialsProvider != null) {
            httpClientBuilder.setDefaultCredentialsProvider(credentialsProvider);
        }
        return httpClientBuilder;
    }
}

public static class MyX509TrustManager implements X509TrustManager {
    X509TrustManager sunJSSEX509TrustManager;

    MyX509TrustManager(String cerFilePath, String cerPassword) throws Exception {
        File file = new File(cerFilePath);
        if (!file.isFile()) {
            throw new Exception("Wrong Certification Path");
        }
        System.out.println("Loading KeyStore " + file + "...");
        InputStream in = new FileInputStream(file);
        KeyStore ks = KeyStore.getInstance("JKS");
        ks.load(in, cerPassword.toCharArray());
        TrustManagerFactory tmf = TrustManagerFactory.getInstance("SunX509", "SunJSSE");
        tmf.init(ks);
        TrustManager[] tms = tmf.getTrustManagers();
        for (TrustManager tm : tms) {
            if (tm instanceof X509TrustManager) {
                sunJSSEX509TrustManager = (X509TrustManager) tm;
                return;
            }
        }
        throw new Exception("Couldn't initialize");
    }

    @Override
    public void checkClientTrusted(X509Certificate[] chain, String authType) throws
CertificateException {

    }

    @Override
    public void checkServerTrusted(X509Certificate[] chain, String authType) throws
CertificateException {

    }

    @Override
    public X509Certificate[] getAcceptedIssuers() {
        return new X509Certificate[0];
    }
}
}
```

/**
* The following is an example of the main function. Call the create function to create a high-level client, call the getLowLevelClient() function to obtain a low-level client, and check whether the **test** index exists.

```

*/
public static void main(String[] args) throws IOException {
    RestHighLevelClient client = create(Arrays.asList("xxx.xxx.xxx.xxx", "xxx.xxx.xxx.xxx"), 9200,
    "https", 1000, 1000, 1000, "username", "password", "cerFilePath", "cerPassword");
    RestClient lowLevelClient = client.getLowLevelClient();
    Request request = new Request("HEAD", "test");
    Response response = lowLevelClient.performRequest(request);
    System.out.println(response.getStatusLine().getStatusCode());
    lowLevelClient.close();
}
}

```

Table 14-7 Function parameters

Name	Description
host	List of the IP addresses of Elasticsearch nodes (or independent Client node). Multiple IP addresses are separated using commas (,).
port	Access port of the Elasticsearch cluster. The default value is 9200 .
protocol	Connection protocol. Set this parameter to https .
connectTimeout	Socket connection timeout period.
connectionRequestTimeout	Timeout period of a socket connection request.
socketTimeout	Timeout period of a socket request.
username	Username for accessing the cluster.
password	Password of the user.
cerFilePath	Certificate path.
cerPassword	Certificate password.

14.2.3.3 Accessing the Cluster Through the Transport Client

You can use Transport Client to access a CSS cluster in non-security mode. If the cluster is in security mode, you are advised to use Rest High Level Client to access the Elasticsearch cluster.

Precautions

You are advised to use the Transport Client version that matches the Elasticsearch version. For example, use Transport Client 7.6.2 to access the Elasticsearch cluster 7.6.2.

Prerequisites

- The CSS cluster is available.

- Ensure that the server running Java can communicate with the CSS cluster.
- Install JDK 1.8 on the server. You can download JDK 1.8 from: <https://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html>
- Declare Java dependencies.

7.6.2 indicates the version of the Elasticsearch Java client.

```
<dependency>
  <groupId>org.elasticsearch.client</groupId>
  <artifactId>transport</artifactId>
  <version>7.6.2</version>
</dependency>
<dependency>
  <groupId>org.elasticsearch</groupId>
  <artifactId>elasticsearch</artifactId>
  <version>7.6.2</version>
</dependency>
```

Procedure

The following code is an example of using Transport Client to connect to the Elasticsearch cluster and check whether the **test** index exists.

```
import org.elasticsearch.action.ActionFuture;
import org.elasticsearch.action.admin.indices.exists.indices.IndicesExistsRequest;
import org.elasticsearch.action.admin.indices.exists.indices.IndicesExistsResponse;
import org.elasticsearch.client.transport.TransportClient;
import org.elasticsearch.common.settings.Settings;
import org.elasticsearch.common.transport.TransportAddress;
import org.elasticsearch.transport.client.PreBuiltTransportClient;

import java.net.InetAddress;
import java.net.UnknownHostException;
import java.util.concurrent.ExecutionException;

public class Main {
    public static void main(String[] args) throws ExecutionException, InterruptedException,
UnknownHostException {
        String cluster_name = "xxx";
        String host1 = "x.x.x.x";
        String host2 = "y.y.y.y";
        Settings settings = Settings.builder()
            .put("client.transport.sniff",false)
            .put("cluster.name", cluster_name)
            .build();
        TransportClient client = new PreBuiltTransportClient(settings)
            .addTransportAddress(new TransportAddress(InetAddress.getByName(host1), 9300))
            .addTransportAddress(new TransportAddress(InetAddress.getByName(host2), 9300));
        IndicesExistsRequest indicesExistsRequest = new IndicesExistsRequest("test");
        ActionFuture<IndicesExistsResponse> exists = client.admin().indices().exists(indicesExistsRequest);
        System.out.println(exists.get().isExists());
    }
}
```

In the preceding information, *cluster_name* indicates the cluster name, and *host1* and *host2* indicate the IP addresses of the cluster nodes. You can run the **GET _cat/nodes** command to view the IP addresses of the nodes.

14.2.4 Accessing a Cluster Using Python

You can access a CSS cluster using Python.

Prerequisites

- The CSS cluster is available.
- Ensure that the server running Python can communicate with the CSS cluster.

Procedure

1. Install the Elasticsearch Python client. You are advised to use the client version that matches the Elasticsearch version. For example, if the cluster version is 7.6.2, install the Elasticsearch Python client 7.6.

```
pip install Elasticsearch==7.6.2
```

2. Create an Elasticsearch client and check whether the **test** index exists. The examples for clusters in different security modes are as follows:

- Cluster in non-security mode

```
from elasticsearch import Elasticsearch

class ElasticFactory(object):

    def __init__(self, host: list, port: str, username: str, password: str):
        self.port = port
        self.host = host
        self.username = username
        self.password = password

    def create(self) -> Elasticsearch:
        addrs = []
        for host in self.host:
            addr = {'host': host, 'port': self.port}
            addrs.append(addr)

        if self.username and self.password:
            elasticsearch = Elasticsearch(addrs, http_auth=(self.username, self.password))
        else:
            elasticsearch = Elasticsearch(addrs)
        return elasticsearch

es = ElasticFactory(["xxx.xxx.xxx.xxx"], "9200", None, None).create()
print(es.indices.exists(index='test'))
```

- Cluster in security mode + HTTP

```
from elasticsearch import Elasticsearch

class ElasticFactory(object):

    def __init__(self, host: list, port: str, username: str, password: str):
        self.port = port
        self.host = host
        self.username = username
        self.password = password

    def create(self) -> Elasticsearch:
        addrs = []
        for host in self.host:
            addr = {'host': host, 'port': self.port}
            addrs.append(addr)

        if self.username and self.password:
            elasticsearch = Elasticsearch(addrs, http_auth=(self.username, self.password))
        else:
            elasticsearch = Elasticsearch(addrs)
        return elasticsearch

es = ElasticFactory(["xxx.xxx.xxx.xxx"], "9200", "username", "password").create()
print(es.indices.exists(index='test'))
```

- Cluster in security mode + HTTPS

```

from elasticsearch import Elasticsearch
import ssl

class ElasticFactory(object):

    def __init__(self, host: list, port: str, username: str, password: str):
        self.port = port
        self.host = host
        self.username = username
        self.password = password

    def create(self) -> Elasticsearch:
        context = ssl._create_unverified_context()

        addrs = []
        for host in self.host:
            addr = {'host': host, 'port': self.port}
            addrs.append(addr)

        if self.username and self.password:
            elasticsearch = Elasticsearch(addrs, http_auth=(self.username, self.password),
            scheme="https", ssl_context=context)
        else:
            elasticsearch = Elasticsearch(addrs)
        return elasticsearch

es = ElasticFactory(["xxx.xxx.xxx.xxx"], "9200", "username", "password").create()
print(es.indices.exists(index='test'))

```

Table 14-8 Variables

Name	Description
host	List of the IP addresses of Elasticsearch nodes (or independent Client node). Multiple IP addresses are separated using commas (,).
port	Access port of the Elasticsearch cluster. Enter 9200 .
username	Username for accessing the cluster.
password	Password of the user.

3. Create a cluster index through the Elasticsearch client.

```

mappings = {
    "settings": {
        "index": {
            "number_of_shards": number_of_shards,
            "number_of_replicas": 1,
        },
    },
    "mappings": {
        properties
    }
}
result = es.indices.create(index=index, body=mappings)

```

4. Query the index created in the previous step through the Elasticsearch client.

```

body = {
    "query": {
        "match": {
            "Query field": "Query content"
        }
    }
}

```

```
}  
}  
}  
result = es.search(index=index, body=body)
```

14.2.5 Using ES-Hadoop to Read and Write Data in Elasticsearch Through Hive

The Elasticsearch-Hadoop (ES-Hadoop) connector combines the massive data storage and in-depth processing capabilities of Hadoop with the real-time search and analysis capabilities of Elasticsearch. It allows you to quickly get to know big data and work better in the Hadoop ecosystem.

This section uses the ES-Hadoop of MRS as an example to describe how to connect to a CSS cluster. You can configure any other applications that need to use the Elasticsearch cluster. Ensure the network connection between the client and the Elasticsearch cluster is normal.

Prerequisites

- The CSS cluster is available.
- The client can communicate with the CSS cluster.
- The CSS and MRS clusters are in the same region, AZ, VPC, and subnet.

Figure 14-4 CSS cluster information

Configuration

Region	
AZ	
VPC	vpc-
Subnet	subnet-
Security Group	

Procedure

1. Obtain the private network address of the cluster. It is used to access the cluster.
 - a. In the navigation pane on the left, choose **Clusters**.
 - b. In the cluster list, select a cluster, and obtain and record its **Private Network Address**. Format: `<host>:<port>` or `<host>:<port>,<host>:<port>`
If the cluster has only one node, the IP address and port number of only one node are displayed, for example, **10.62.179.32:9200**. If the cluster has multiple nodes, the IP addresses and port numbers of all nodes are displayed, for example, **10.62.179.32:9200,10.62.179.33:9200**.

2. Log in to an MRS cluster node. For details, see .
3. Run the cURL command on an MRS cluster node to check the network connectivity. Ensure every node in the MRS cluster can connect to the CSS cluster.
 - Cluster in non-security mode
`curl -X GET http://<host>:<port>`
 - Cluster in security mode + HTTP
`curl -X GET http://<host>:<port> -u <user>:<password>`
 - Cluster in security mode + HTTPS
`curl -X GET https://<host>:<port> -u <user>:<password> -ik`

Table 14-9 Variables

Variable	Description
<host>	IP address of each node in the cluster. If the cluster contains multiple nodes, there will be multiple IP addresses. You can use any of them.
<port>	Port number for accessing a cluster node. Generally, the port number is 9200.
<user>	Username for accessing the cluster.
<password>	Password of the user.

4. Download the [ES-Hadoop lib package](#) and decompress it to obtain the **elasticsearch-hadoop-x.x.x.jar** file. The version must be the same as the CSS cluster version. For example, if the CSS cluster version is 7.6.2, you are advised to download **elasticsearch-hadoop-7.6.2.zip**.
5. Download the httpclient dependency package [commons-httpclient:commons-httpclient-3.1.jar](#). In the package name, **3.1** indicates the version number. Select the package of the version you need.
6. Install the MRS client. If the MRS client has been installed, skip this step. For details, see .
7. Log in to the MRS client. Upload the JAR dependency packages of ES-Hadoop and httpclient to the MRS client.
8. Create an HDFS directory on the MRS client. Upload the ES-Hadoop lib package and the httpclient dependency package to the directory.


```
hadoop fs -mkdir /tmp/hadoop-es
hadoop fs -put elasticsearch-hadoop-x.x.x.jar /tmp/hadoop-es
hadoop fs -put commons-httpclient-3.1.jar /tmp/hadoop-es
```
9. Log in to the Hive client from the MRS client. For details, see .
10. On the Hive client, add the ES-Hadoop lib package and the httpclient dependency package. This command is valid only for the current session. Enter **beeline** or **hive** to go to the execution page and run the following commands:


```
add jar hdfs:///tmp/hadoop-es/commons-httpclient-3.1.jar;
add jar hdfs:///tmp/hadoop-es/elasticsearch-hadoop-x.x.x.jar;
```
11. On the Hive client, create a Hive foreign table.
 - Cluster in non-security mode

```
CREATE EXTERNAL table IF NOT EXISTS student(
  id BIGINT,
  name STRING,
  addr STRING
)
STORED BY 'org.elasticsearch.hadoop.hive.EsStorageHandler'
TBLPROPERTIES(
  'es.nodes' = 'xxx.xxx.xxx.xxx:9200',
  'es.port' = '9200',
  'es.net.ssl' = 'false',
  'es.nodes.wan.only' = 'false',
  'es.nodes.discovery'='false',
  'es.input.use.sliced.partitions'='false',
  'es.resource' = 'student/_doc'
);
```

– Cluster in security mode + HTTP

```
CREATE EXTERNAL table IF NOT EXISTS student(
  id BIGINT,
  name STRING,
  addr STRING
)
STORED BY 'org.elasticsearch.hadoop.hive.EsStorageHandler'
TBLPROPERTIES(
  'es.nodes' = 'xxx.xxx.xxx.xxx:9200',
  'es.port' = '9200',
  'es.net.ssl' = 'false',
  'es.nodes.wan.only' = 'false',
  'es.nodes.discovery'='false',
  'es.input.use.sliced.partitions'='false',
  'es.nodes.client.only'='true',
  'es.resource' = 'student/_doc',
  'es.net.http.auth.user' = 'username',
  'es.net.http.auth.pass' = 'password'
);
```

– Cluster in security mode + HTTPS

i. Obtain the security certificate **CloudSearchService.cer**.

- 1) Log in to the CSS management console.
- 2) In the navigation pane, choose **Clusters**. The cluster list is displayed.
- 3) Click the name of a cluster to go to the cluster details page.
- 4) On the **Configuration** page, click **Download Certificate** next to **HTTPS Access**.

ii. Convert the security certificate **CloudSearchService.cer**. Upload the downloaded security certificate to the client and use keytool to convert the .cer certificate into a .jks certificate that can be read by Java.

- In Linux, run the following command to convert the certificate:
keytool -import -alias *newname* -keystore ./truststore.jks -file ./CloudSearchService.cer

- In Windows, run the following command to convert the certificate:
keytool -import -alias *newname* -keystore .\truststore.jks -file .\CloudSearchService.cer

In the preceding command, *newname* indicates the user-defined certificate name.

After this command is executed, you will be prompted to set the certificate password and confirm the password. Securely store the password. It will be used for accessing the cluster.

- iii. Put the .jks file to the same path of each node in the MRS cluster, for example, /tmp. You can run the **scp** command to transfer the file. Ensure user **omm** has the permission to read the file. You can run the following command to set the permission:

```
chown -R omm truststore.jks
```

- iv. Create a Hive foreign table.

```
CREATE EXTERNAL table IF NOT EXISTS student(
  id BIGINT,
  name STRING,
  addr STRING
)
STORED BY 'org.elasticsearch.hadoop.hive.EsStorageHandler'
TBLPROPERTIES(
  'es.nodes' = 'https://xxx.xxx.xxx.xxx:9200',
  'es.port' = '9200',
  'es.net.ssl' = 'true',
  'es.net.ssl.truststore.location' = 'cerFilePath',
  'es.net.ssl.truststore.pass' = 'cerPassword',
  'es.nodes.wan.only' = 'false',
  'es.nodes.discovery'='false',
  'es.nodes.client.only'='true',
  'es.input.use.sliced.partitions'='false',
  'es.resource' = 'student/_doc',
  'es.net.http.auth.user' = 'username',
  'es.net.http.auth.pass' = 'password'
);
```

Table 14-10 ES-Hadoop parameters

Parameter	Default Value	Description
es.nodes	localhost	Address for accessing the CSS cluster. You can view private network address in the cluster list.
es.port	9200	Port number for accessing a cluster. Generally, the port number is 9200.
es.nodes.wan.only	false	Whether to perform node sniffing.
es.nodes.discovery	true	Whether to disable node discovery.
es.input.use.sliced.partitions	true	Whether to use slices. Its value can be: <ul style="list-style-type: none"> • true • false NOTE If this parameter is set to true , the index prefetch time may be significantly prolonged, and may even be much longer than the data query time. You are advised to set this parameter to false to improve query efficiency.

Parameter	Default Value	Description
es.resource	NA	Specifies the index and type to be read and written.
es.net.http.auth.user	NA	Username for accessing the cluster. Set this parameter only if the security mode is enabled.
es.net.http.auth.pass	NA	Password of the user. Set this parameter only if the security mode is enabled.
es.net.ssl	false	Whether to enable SSL. If SSL is enabled, you need to configure the security certificate information.
es.net.ssl.truststore.location	NA	Path of the .jks certificate file, for example, file:///tmp/truststore.jks .
es.nodes.client.only	false	Check whether the IP address of an independent Client node is configured for es.nodes (that is, whether the Client node is enabled during Elasticsearch cluster creation). If yes, change the value to true , or an error will be reported, indicating that the data node cannot be found.
es.net.ssl.truststore.pass	NA	Password of the .jks certificate file.

For details about ES-Hadoop configuration items, see the [official configuration description](#).

- On the Hive client, insert data.

```
INSERT INTO TABLE student VALUES (1, "Lucy", "address1"), (2, "Lily", "address2");
```

- On the Hive client, run a query.

```
select * from student;
```

The query result is as follows:

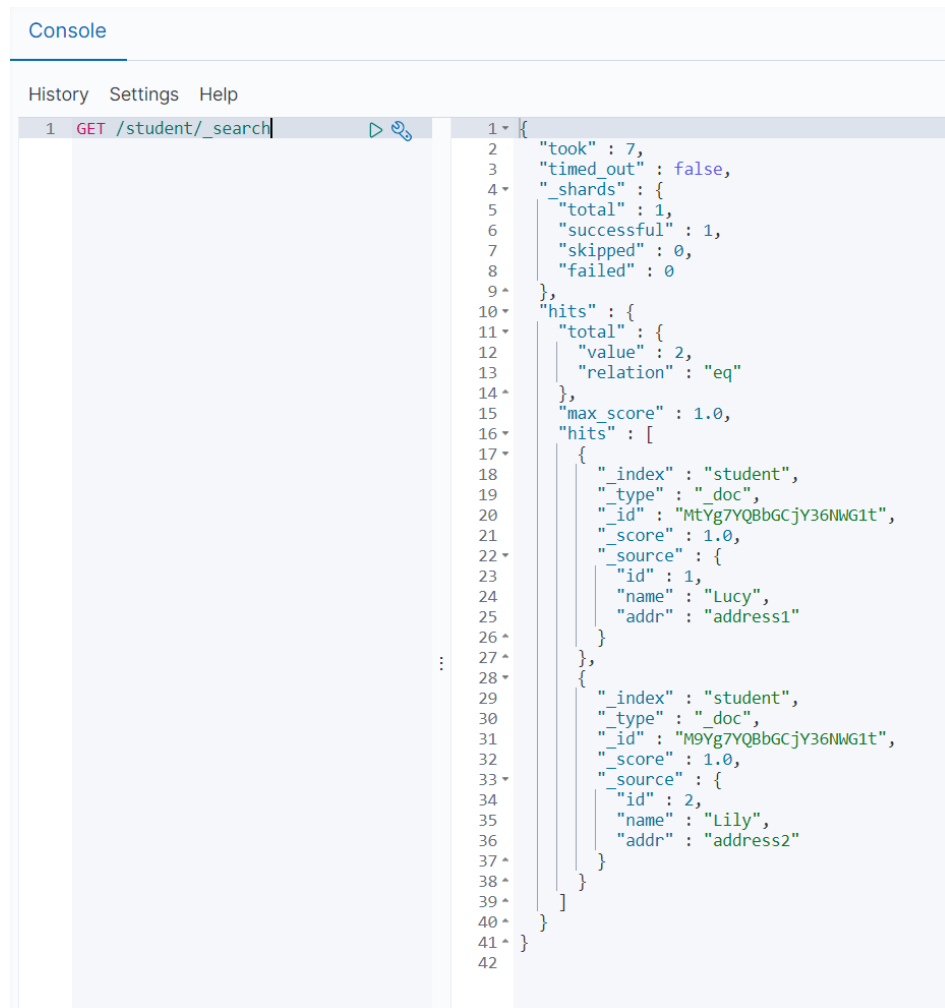
```
+-----+-----+-----+
| student.id | student.name | student.addr |
+-----+-----+-----+
| 1          | Lucy         | address1     |
| 2          | Lily        | address2     |
+-----+-----+-----+
2 rows selected (0.116 seconds)
```

- Log in to the CSS console and choose **Clusters**. Locate the target cluster and click **Access Kibana** in the **Operation** column.

- On the **Dev Tools** page of Kibana, run a query and view the result.

```
GET /student/_search
```

Figure 14-5 Kibana query result



```
1 GET /student/_search
2 {
3   "took" : 7,
4   "timed_out" : false,
5   "shards" : {
6     "total" : 1,
7     "successful" : 1,
8     "skipped" : 0,
9     "failed" : 0
10  },
11  "hits" : {
12    "total" : {
13      "value" : 2,
14      "relation" : "eq"
15    },
16    "max_score" : 1.0,
17    "hits" : [
18      {
19        "_index" : "student",
20        "_type" : "_doc",
21        "_id" : "MtYg7YQBbGCjY36NWG1t",
22        "_score" : 1.0,
23        "_source" : {
24          "id" : 1,
25          "name" : "Lucy",
26          "addr" : "address1"
27        }
28      },
29      {
30        "_index" : "student",
31        "_type" : "_doc",
32        "_id" : "M9Yg7YQBbGCjY36NWG1t",
33        "_score" : 1.0,
34        "_source" : {
35          "id" : 2,
36          "name" : "Lily",
37          "addr" : "address2"
38        }
39      }
40    ]
41  }
42 }
```

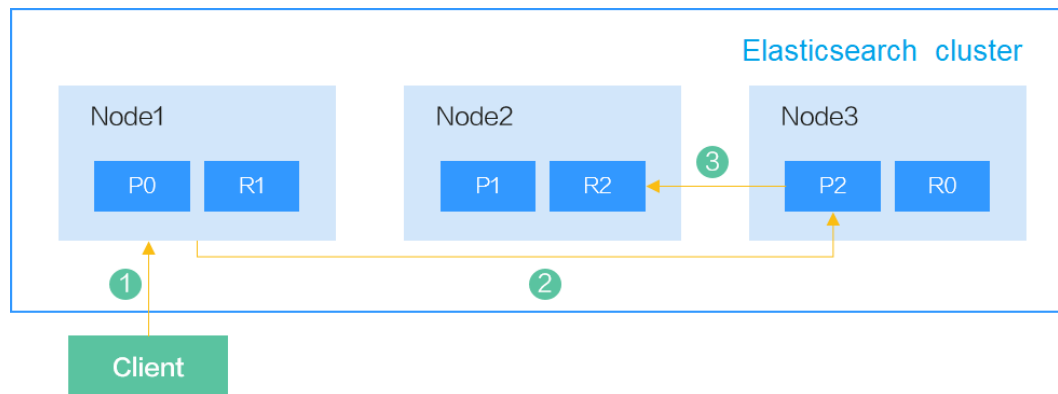
14.3 Cluster Performance Tuning

14.3.1 Optimizing Write Performance

Before using a CSS cluster, you are advised to optimize the write performance of the cluster to improve efficiency.

Data Write Process

Figure 14-6 Data write process

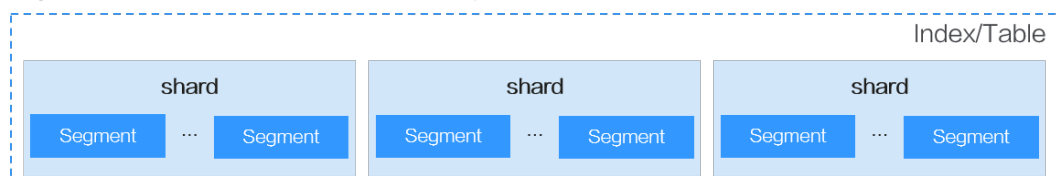


The process of writing data from a client to Elasticsearch is as follows:

1. The client sends a data write request to Node1. Here Node1 is the coordinator node.
2. Node1 routes the data to shard 2 based on the `_id` of the data. In this case, the request is forwarded to Node3 and the write operation is performed.
3. After data is written to the primary shard, the request is forwarded to the replica shard of Node2. After the data is written to the replica, Node3 reports the write success to the coordinator node, and the coordinator node reports it to the client.

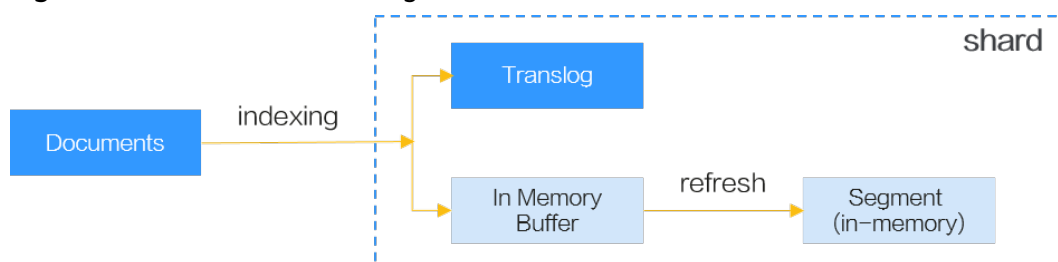
An index in Elasticsearch consists of one or more shards. Each shard contains multiple segments, and each segment is an inverted index.

Figure 14-7 Elasticsearch index composition



When a document is inserted into Elasticsearch, the document is first written to the buffer and then periodically refreshed from the buffer to the segment. The refresh frequency is specified by the `refresh_interval` parameter. By default, data is refreshed every second.

Figure 14-8 Process of inserting a document into Elasticsearch



Improving Write Performance

In the Elasticsearch data write process, the following solutions can be used to improve performance:

Table 14-11 Improving write performance

No.	Solution	Description
1	Use SSDs or improve cluster configurations.	Using SSDs can greatly speed up data write and merge operations. For CSS, you are advised to select the ultra-high I/O storage or ultra-high I/O servers.
2	Use Bulk APIs.	The client writes data in batches. You are advised to write 1 MB to 10 MB data in each batch.
3	Randomly generate <code>_id</code> .	If <code>_id</code> is specified, a query operation will be triggered before data is written, affecting data write performance. In scenarios where data does not need to be retrieved using <code>_id</code> , you are advised to use a randomly generated <code>_id</code> .
4	Set a proper number of segments.	You are advised to set the number of shards to a multiple of the number of cluster data nodes. Ensure each shard is smaller than 50 GB.
5	Close replicas.	Data write and query are performed in off-peak hours. Close data copies during writing and open them afterwards. The command for disabling replicas in Elasticsearch 7.x is as follows: <pre>PUT {index}/_settings { "number_of_replicas": 0 }</pre>
6	Adjust the index refresh frequency.	During batch data writing, you can set refresh_interval to a large value or -1 (indicating no refresh), improving the write performance by reducing refresh. In Elasticsearch 7.x, run the following command to set the update time to 15s: <pre>PUT {index}/_settings { "refresh_interval": "15s" }</pre>
7	Change the number of write threads and the size of the write queue.	You can increase the number of write threads and the size of the write queue, or error code 429 may be returned for unexpected traffic peaks. In Elasticsearch 7.x, you can modify the following parameters to optimize write performance: thread_pool.write.size and thread_pool.write.queue_size

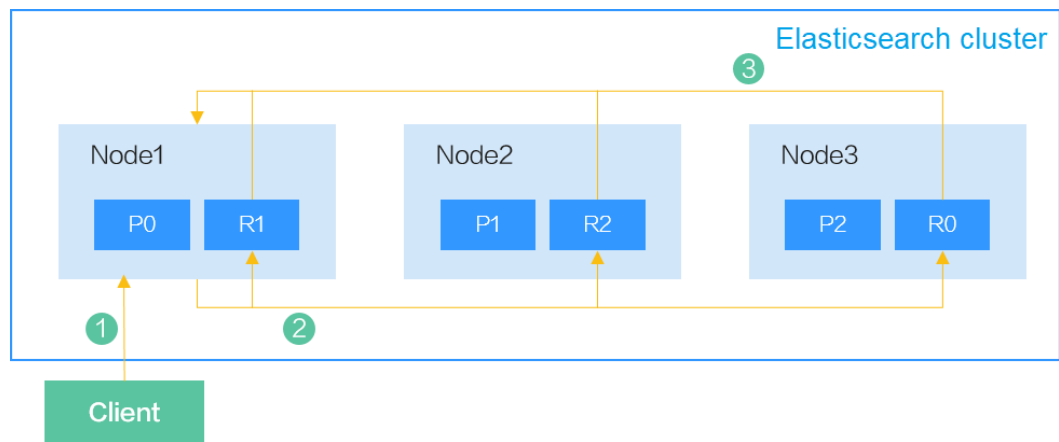
No.	Solution	Description
8	Set a proper field type.	<p>Specify the type of each field in the cluster, so that Elasticsearch will not regard the fields as a combination of keywords and texts, which unnecessarily increase data volume. Keywords are used for keyword search, and texts used for full-text search.</p> <p>For the fields that do not require indexes, you are advised to set index to false.</p> <p>In Elasticsearch 7.x, run the following command to set index to false for field1:</p> <pre>PUT {index} { "mappings": { "properties": { "field1":{ "type": "text", "index": false } } } }</pre>
9	Optimize the shard balancing policy.	<p>By default, Elasticsearch uses the load balance policy based on the disk capacity. If there are multiple nodes, especially if some of them are newly added, shards may be unevenly allocated on the nodes. To avoid such problems, you can set the index-level parameter routing.allocation.total_shards_per_node to control the distribution of index shards on each node. You can set this parameter in the index template, or modify the setting of an existing index to make the setting take effect.</p> <p>Run the following command to modify the setting of an existing index:</p> <pre>PUT {index}/_settings { "index": { "routing.allocation.total_shards_per_node": 2 } }</pre>

14.3.2 Optimizing Query Performance

Before using a CSS cluster, you are advised to optimize the query performance of the cluster to improve efficiency.

Data Query Process

Figure 14-9 Data query process

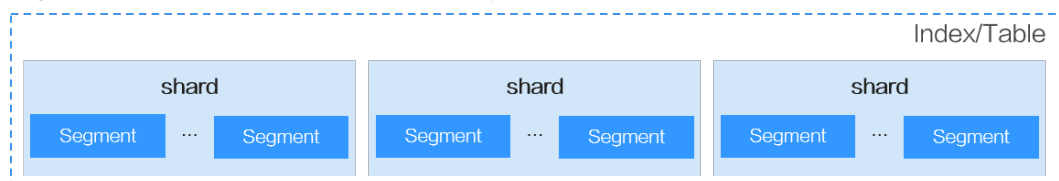


When a client sends a query request to Elasticsearch, the query process is as follows:

1. The client sends a data query request to Node1. Here Node1 is the coordinator node.
2. Node1 selects a shard based on the shard distribution and the index specified in the query, and then forwards the request to Node1, Node2, and Node3.
3. Each shard executes the query task. After the query succeeds on the shards, the query results are aggregated to Node1, which returns the results to the client.

For a query request, five shards can be queried concurrently on a node by default. If there are more than five shards, the query will be performed in batches. In a single shard, the query is performed by traversing each segment one by one.

Figure 14-10 Elasticsearch index composition



Improving Query Performance

In the Elasticsearch data query process, the following solutions can be used to improve performance:

Table 14-12 Improving query performance

No.	Solution	Description
1	Use _routing to reduce the number of shards scanned during retrieval.	<p>During data import, configure routing to route data to a specific shard instead of all the shards of the related index, improving the overall throughput of the cluster.</p> <p>In Elasticsearch 7.x, run the following commands:</p> <ul style="list-style-type: none"> • Insert data based on a specified routing. <pre>PUT /{index}/_doc/1?routing=user1 { "title": "This is a document" }</pre> • Query data based on a specified routing. <pre>GET /{index}/_doc/1?routing=user1</pre>
2	Use index sorting to reduce the number of segments to be scanned.	<p>When a request is processed on a shard, the segments of the shard are traversed one by one. By using index sorting, the range query or sorting query can be terminated in advance (early-terminate).</p> <p>For example, in Elasticsearch 7.x, run the following commands:</p> <pre>// Assume the date field needs to be frequently used for range query. PUT {index} { "settings": { "index": { "sort.field": "date", "sort.order": "desc" } }, "mappings": { "properties": { "date": { "type": "date" } } } }</pre>
3	Add query cache to improve cache hit.	<p>When a filter request is executed in a segment, the bitset is used to retain the result, which can be reused for later similar queries, thus reducing the overall query workloads.</p> <p>You can add query cache by increasing the value of indices.queries.cache.size. For details, see . Restart the cluster for the modification to take effect.</p>
4	Perform forcemerge in advance to reduce the number of segments to be scanned.	<p>For read-only indexes that are periodically rolled, you can periodically execute forcemerge to combine small segments into large segments and permanently delete indexes marked as deleted.</p> <p>In Elasticsearch 7.x, a configuration example is as follows:</p> <pre>// Assume the number of segments after index forcemerge is set to 10. POST /{index}/_forcemerge?max_num_segments=10</pre>

14.4 Practices

14.4.1 Using CSS to Accelerate Database Query and Analysis

Overview

Elasticsearch is used as a supplement to relational databases, such as MySQL and GaussDB(for MySQL), to improve the full-text search and high-concurrency ad hoc query capabilities of the databases.

This chapter describes how to synchronize data from a MySQL database to CSS to accelerate full-text search and ad hoc query and analysis. The following figure shows the solution process.

Figure 14-11 Using CSS to accelerate database query and analysis



1. Service data is stored in the MySQL database.
2. DRS synchronizes data from MySQL to CSS in real time.
3. CSS is used for full-text search and data query and analysis.

Prerequisites

- A CSS cluster and a MySQL database in security mode have been created, and they are in the same VPC and security group.
- Data to be synchronized exists in the MySQL database. This section uses the following table structure and initial data as an example.

- a. Create a student information table in MySQL.

```

CREATE TABLE `student` (
  `dsc` varchar(100) COLLATE utf8mb4_general_ci DEFAULT NULL,
  `age` smallint unsigned DEFAULT NULL,
  `name` varchar(32) COLLATE utf8mb4_general_ci NOT NULL,
  `id` int unsigned NOT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;
  
```

- b. Insert the initial data of three students into the MySQL database.

```

INSERT INTO student (id,name,age,dsc)
VALUES
('1','Jack Ma Yun','50','Jack Ma Yun is a business magnate, investor and philanthropist. '),
('2','will smith','22','also known by his stage name the Fresh Prince, is an American actor, rapper, and producer. '),
('3','James Francis Cameron','68','the director of avatar');
  
```

- Indexes have been created in the CSS cluster and match the table indexes in the MySQL database.

The following is an example of the indexes in the cluster in this chapter:

```

PUT student
{
  "settings": {
    "number_of_replicas": 0,
    "number_of_shards": 3
  },
}
  
```

```

"mappings": {
  "properties": {
    "id": {
      "type": "keyword"
    },
    "name": {
      "type": "short"
    },
    "age": {
      "type": "short"
    },
    "desc": {
      "type": "text"
    }
  }
}

```

Configure **number_of_shards** and **number_of_replicas** as needed.

Procedure

Step 1 Use DRS to synchronize MySQL data to CSS in real time. For details, see .

In this example, configure the parameters by following the suggestions in [Table 14-13](#).

Table 14-13 Synchronization parameters

Module	Parameter	Suggestion
Create Synchronization Instance > Synchronize Instance Details	Network Type	Select VPC .
	Source DB Instance	Select the RDS for MySQL instance to be synchronized, that is, the MySQL database that stores service data.
	Synchronization Instance Subnet	Select the subnet where the synchronization instance is located. You are advised to select the subnet where the database instance and the CSS cluster are located.
Configure Source and Destination Databases > Destination Database	VPC and Subnet	Select the VPC and subnet of the CSS cluster.
	IP Address or Domain Name	Enter the IP address of the CSS cluster. For details, see Obtaining the IP address of a CSS cluster .
	Database Username and Database Password	Enter the administrator username (admin) and password of the CSS cluster.

Module	Parameter	Suggestion
	Encryption Certificate	Select the security certificate of the CSS cluster. If SSL Connection is not enabled, you do not need to select any certificate. For details, see Obtaining the security certificate of a CSS cluster .
Set Synchronization Task	Flow Control	Select No .
	Synchronization Object Type	Deselect Table structure , because the indexes matching MySQL tables have been created in the CSS cluster.
	Synchronization Object	Select Tables . Select the database and table name corresponding to CSS. NOTE Ensure the type name in the configuration item is the same as the index name, that is, _doc .
Process Data	-	Click Next .

After the synchronization task is started, wait until the **Status** of the task changes from **Full** synchronization to **Incremental**, indicating real-time synchronization has started.

Step 2 Check the synchronization status of the database.

1. Verify full data synchronization.

Run the following command in Kibana of CSS to check whether full data has been synchronized to CSS:

```
GET student/_search
```

2. Insert new data in the source cluster and check whether the data is synchronized to CSS.

For example, insert a record whose **id** is **4** in the source cluster.

```
INSERT INTO student (id,name,age,dsc)
```

```
VALUES
```

```
('4','Bill Gates','50','Gates III is an American business magnate, software developer, investor, author, and philanthropist.')
```

Run the following command in Kibana of CSS to check whether new data is synchronized to CSS:

```
GET student/_search
```

3. Update data in the source cluster and check whether the data is synchronized to CSS.

For example, in the record whose **id** is **4**, change the value of **age** from **50** to **55**.

```
UPDATE student set age='55' WHERE id=4;
```

Run the following command in Kibana of CSS to check whether the data is updated in CSS:

```
GET student/_search
```

4. Delete data from the source cluster and check whether the data is deleted synchronously from CSS.

For example, delete the record whose **id** is **4**.

```
DELETE FROM student WHERE id=4;
```

Run the following command in Kibana of CSS to check whether the data is deleted synchronously from CSS:

```
GET student/_search
```

Step 3 Verify the full-text search capability of the database.

For example, run the following command to query the data that contains **avatar** in **dsc** in CSS:

```
GET student/_search
{
  "query": {
    "match": {
      "dsc": "avatar"
    }
  }
}
```

Step 4 Verify the ad hoc query capability of the database.

For example, query **philanthropist** whose age is greater than **40** in CSS.

```
GET student/_search
{
  "query": {
    "bool": {
      "must": [
        {
          "match": {
            "dsc": "philanthropist"
          }
        },
        {
          "range": {
            "age": {
              "gte": 40
            }
          }
        }
      ]
    }
  }
}
```

Step 5 Verify the statistical analysis capability of the database.

For example, use CSS to collect statistics on the age distributions of all users.

```
GET student/_search
{
  "size": 0,
  "query": {
    "match_all": {}
  },
  "aggs": {
    "age_count": {
      "terms": {
        "field": "age",
        "size": 10
      }
    }
  }
}
```

```
}  
}
```

----End

Other Operations

- **Obtaining the IP address of a CSS cluster**
 - a. In the navigation pane on the left, choose **Clusters**.
 - b. In the cluster list, locate a cluster, and obtain the IP address of the CSS cluster from the **Private Network Address** column. Generally, the IP address format is `<host>:<port>` or `<host>:<port>,<host>:<port>`.
If the cluster has only one node, the IP address and port number of only one node are displayed, for example, **10.62.179.32:9200**. If the cluster has multiple nodes, the IP addresses and port numbers of all nodes are displayed, for example, **10.62.179.32:9200,10.62.179.33:9200**.
- **Obtaining the security certificate of a CSS cluster**
 - a. Log in to the CSS management console.
 - b. In the navigation pane, choose **Clusters**. The cluster list is displayed.
 - c. Click the name of a cluster to go to the cluster details page.
 - d. On the **Configuration** page, click **Download Certificate** next to **HTTPS Access**.

14.4.2 Using CSS to Build a Unified Log Management Platform

A unified log management platform built using CSS can manage logs in real time in a unified and convenient manner, enabling log-driven O&M and improving service management efficiency.

Overview

Elasticsearch, Logstash, Kibana, and Beats (ELKB) provides a complete set of log solutions and is a mainstream log system. The following figure shows its framework.

Figure 14-12 Unified log management platform framework



- Beats is a lightweight log collector, comprising Filebeat and Metricbeat.
- Logstash collects and preprocesses logs. It supports multiple data sources and ETL processing modes.
- Elasticsearch is an open-source distributed search engine that collects, analyzes, and stores data. CSS allows you to create Elasticsearch clusters.
- Kibana is a visualization tool used to perform web-based visualized query and make BI reports.

This section describes how to use CSS, Filebeat, Logstash, and Kibana to build a unified log management platform. Filebeat collects ECS logs and sends the logs to

Logstash for data processing. The processing results are stored in CSS, and can be queried, analyzed, and visualized using Kibana.

For details about the version compatibility of ELKB components, see https://www.elastic.co/support/matrix#matrix_compatibility.

Prerequisites

- A CSS cluster in non-security mode has been created.
- You have applied for an ECS and installed the Java environment on it.

Procedure

Step 1 Deploy and configure Filebeat.

1. Download Filebeat. The recommended version is 7.6.2. Download it at <https://www.elastic.co/downloads/past-releases#filebeat-oss>.
2. Configure the Filebeat configuration file **filebeat.yml**.

For example, to collect all the files whose names end with **log** in the **/root/** directory, configure the **filebeat.yml** file is as follows:

```
filebeat.inputs:
- type: log
  enabled: true
  # Path of the collected log file
  paths:
  - /root/*.log

filebeat.config.modules:
  path: ${path.config}/modules.d/*.yml
  reload.enabled: false
# Logstash hosts information
output.logstash:
  hosts: ["192.168.0.126:5044"]

processors:
```

Step 2 Deploy and configure Logstash.

NOTE

To achieve better performance, you are advised to set the JVM parameter in Logstash to half of the ECS or docker memory.

1. Download Logstash. The recommended version is 7.6.2. Download it at <https://www.elastic.co/downloads/past-releases#logstash-oss>.
2. Ensure that Logstash can communicate with the CSS cluster.
3. Configure the Logstash configuration file **logstash-sample.conf**.

The content of the **logstash-sample.conf** file is as follows:

```
input {
  beats {
    port => 5044
  }
}
# Split data.
filter {
  grok {
    match => {
      "message" => '\[%{GREEDYDATA:timemaybe}\] \[%{WORD:level}\] %{GREEDYDATA:content}'
    }
  }
}
```



```
mutate {
  remove_field => ["@version","tags","source","input","prospector","beat"]
}
# CSS cluster information
output {
  elasticsearch {
    hosts => ["http://192.168.0.4:9200"]
    index => "%{[@metadata][beat]}-%{+YYYY.MM.dd}"
    #user => "xxx"
    #password => "xxx"
  }
}
```

NOTE

You can use Grok Debugger (<http://grokdebug.herokuapp.com/>) to configure the **filter** mode of Logstash.

Step 3 Configure the index template of the CSS cluster on Kibana or via API.

For example, create an index template. Let the index use three shards and no replicas. Fields such as **@timestamp**, **content**, **host.name**, **level**, **log.file.path**, **message** and **timemaybe** are defined in the index.

```
PUT _template/filebeat
{
  "index_patterns": ["filebeat*"],
  "settings": {
    # Define the number of shards.
    "number_of_shards": 3,
    # Define the number of copies.
    "number_of_replicas": 0,
    "refresh_interval": "5s"
  },
  # Define a field.
  "mappings": {
    "properties": {
      "@timestamp": {
        "type": "date"
      },
      "content": {
        "type": "text"
      },
      "host": {
        "properties": {
          "name": {
            "type": "text"
          }
        }
      },
      "level": {
        "type": "keyword"
      },
      "log": {
        "properties": {
          "file": {
            "properties": {
              "path": {
                "type": "text"
              }
            }
          }
        }
      },
      "message": {
        "type": "text"
      },
      "timemaybe": {
```


- **query_score**: calculated based on the total number of search keywords found in a record. A record earns 1 point for each keyword it contains.
- **vote**: vote of a record.
- **factor** : user-defined weight of **vote**.
- Calculate the final scores (**new_score**) of query results based on **inline** and sort the results in descending order.
new_score = query_score x inline
 - **query_score**: calculated based on the total number of search keywords found in a record. A record earns 1 point for each keyword it contains.
 - **vote**: vote of a record.
 - **inline**: Configure two value options for this parameter and a threshold for **vote**. One option is used if **vote** exceeds the threshold, and the other is used if **vote** is smaller than or equal to the threshold. In this way, the query accuracy will not be affected by abnormal **vote** values.

Prerequisites

An Elasticsearch cluster has been created on the CSS management console and is available.

Procedure

NOTE

The code examples in this section can only be used for clusters Elasticsearch 7.x or later.

1. Log in to the CSS management console.
2. In the navigation pane on the left, click **Clusters** to go to the Elasticsearch cluster list.
3. Click **Access Kibana** in the **Operation** column of a cluster.
4. In the navigation tree on the left of Kibana, choose **Dev Tools**. The command execution page is displayed.
5. Create an index and specify a custom mapping to define the data type.

For example, the content of the **tv.json** file is as follows:

```
{
  "tv": [
    { "name": "tv1", "description": "USB, DisplayPort", "vote": 0.98 }
    { "name": "tv2", "description": "USB, HDMI", "vote": 0.99 }
    { "name": "tv3", "description": "USB", "vote": 0.5 }
    { "name": "tv4", "description": "USB, HDMI, DisplayPort", "vote": 0.7 }
  ]
}
```

Run the following command to create the **mall** index and specify the user-defined mapping to define the data type:

```
PUT /mall?pretty
{
  "mappings": {
    "properties": {
      "name": {
        "type": "text",
        "fields": {
          "keyword": {
            "type": "keyword"
          }
        }
      }
    }
  }
}
```

```
}
},
"description": {
  "type": "text",
  "fields": {
    "keyword": {
      "type": "keyword"
    }
  }
},
"vote": {
  "type": "float"
}
}
}
```

6. Import data.

Run the following command to import data in the **tv.json** file to the **mall** index:

```
POST /mall/_bulk?pretty
{"index":{"_id":"1"}}
{"name":"tv1","description":"USB, DisplayPort","vote":0.98}
{"index":{"_id":"2"}}
{"name":"tv2","description":"USB, HDMI","vote":0.99}
{"index":{"_id":"3"}}
{"name":"tv3","description":"USB","vote":0.5}
{"index":{"_id":"4"}}
{"name":"tv4","description":"USB, HDMI, DisplayPort","vote":0.7}
```

7. Query data by using custom scoring. The query results can be scored based on **vote** or **inline**.

Assume a user wants to query TVs with USB, HDMI, and/or DisplayPort ports. The final query score can be calculated in the following ways and used for sorting:

– Scoring based on **vote**

The score is calculated using the formula **new_score = query_score x (vote x factor)**. Run the following command:

```
GET /mall/_doc/_search?pretty
{
  "query":{
    "function_score":{
      "query":{
        "bool":{
          "should":[
            {"match":{"description":"USB"}},
            {"match":{"description":"HDMI"}},
            {"match":{"description":"DisplayPort"}}
          ]
        }
      },
      "field_value_factor":{
        "field":"vote",
        "factor":1
      },
      "boost_mode":"multiply",
      "max_boost":10
    }
  }
}
```

The query results are displayed in descending order of the score. The command output is as follows:

```
{
  "took":4,
  "timed_out":false,
```

```

"_shards" : {
  "total" : 1,
  "successful" : 1,
  "skipped" : 0,
  "failed" : 0
},
"hits" : {
  "total" : {
    "value" : 4,
    "relation" : "eq"
  },
  "max_score" : 0.8388366,
  "hits" : [
    {
      "_index" : "mall",
      "_type" : "_doc",
      "_id" : "4",
      "_score" : 0.8388366,
      "_source" : {
        "name" : "tv4",
        "description" : "USB, HDMI, DisplayPort",
        "vote" : 0.7
      }
    },
    {
      "_index" : "mall",
      "_type" : "_doc",
      "_id" : "2",
      "_score" : 0.7428025,
      "_source" : {
        "name" : "tv2",
        "description" : "USB, HDMI",
        "vote" : 0.99
      }
    },
    {
      "_index" : "mall",
      "_type" : "_doc",
      "_id" : "1",
      "_score" : 0.7352994,
      "_source" : {
        "name" : "tv1",
        "description" : "USB, DisplayPort",
        "vote" : 0.98
      }
    },
    {
      "_index" : "mall",
      "_type" : "_doc",
      "_id" : "3",
      "_score" : 0.03592815,
      "_source" : {
        "name" : "tv3",
        "description" : "USB",
        "vote" : 0.5
      }
    }
  ]
}

```

– Scoring based on **inline**

The score is calculated using the formula **new_score = query_score x inline**. In this example, if **vote** > 0.8, the value of **inline** is 1. If **vote** ≤ 0.8, the value of **inline** is 0.5. Run the following command:

```

GET /mall/_doc/_search?pretty
{
  "query":{

```

```
"function_score":{
  "query":{
    "bool":{
      "should":[
        {"match":{"description":"USB"}},
        {"match":{"description":"HDMI"}},
        {"match":{"description":"DisplayPort"}}
      ]
    }
  },
  "script_score": {
    "script": {
      "params": {
        "threshold": 0.8
      },
      "inline": "if (doc[\"vote\"].value > params.threshold) {return 1;} return 0.5;"
    }
  },
  "boost_mode":"multiply",
  "max_boost":10
}
}
```

The query results are displayed in descending order of the score. The command output is as follows:

```
{
  "took" : 4,
  "timed_out" : false,
  "_shards" : {
    "total" : 1,
    "successful" : 1,
    "skipped" : 0,
    "failed" : 0
  },
  "hits" : {
    "total" : {
      "value" : 4,
      "relation" : "eq"
    },
    "max_score" : 0.75030553,
    "hits" : [
      {
        "_index" : "mall",
        "_type" : "_doc",
        "_id" : "1",
        "_score" : 0.75030553,
        "_source" : {
          "name" : "tv1",
          "description" : "USB, DisplayPort",
          "vote" : 0.98
        }
      },
      {
        "_index" : "mall",
        "_type" : "_doc",
        "_id" : "2",
        "_score" : 0.75030553,
        "_source" : {
          "name" : "tv2",
          "description" : "USB, HDMI",
          "vote" : 0.99
        }
      },
      {
        "_index" : "mall",
        "_type" : "_doc",
        "_id" : "4",
        "_score" : 0.599169,

```

```
"_source" : {  
  "name" : "tv4",  
  "description" : "USB, HDMI, DisplayPort",  
  "vote" : 0.7  
},  
{  
  "_index" : "mall",  
  "_type" : "_doc",  
  "_id" : "3",  
  "_score" : 0.03592815,  
  "_source" : {  
    "name" : "tv3",  
    "description" : "USB",  
    "vote" : 0.5  
  }  
}  
]  
}
```

15 FAQs

15.1 General Consulting

15.1.1 What Are Regions and AZs?

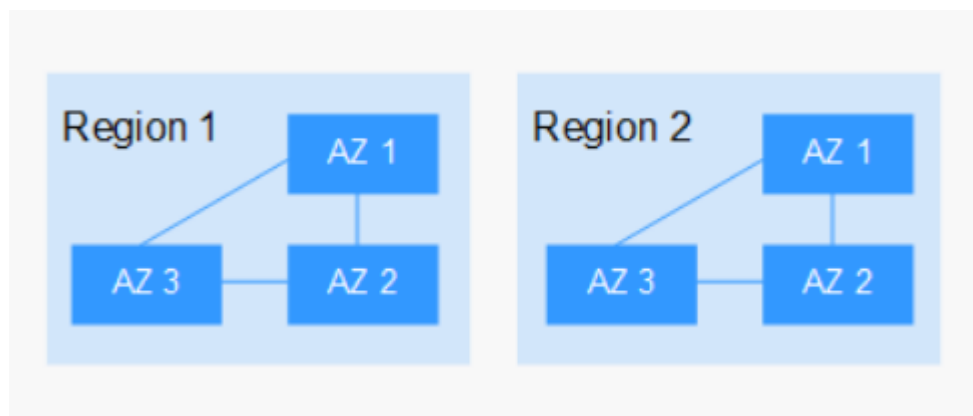
Regions and AZs

A region and availability zone (AZ) identify the location of a data center. You can create resources in a specific region and AZ.

- A region is a physical data center. Each region is completely independent, and thereby improves fault tolerance and stability. After a resource is created, its region cannot be changed.
- An AZ is a physical location using independent power supplies and networks. Faults in an AZ do not affect other AZs. A region can contain multiple AZs that are physically isolated but networked together. This enables low-cost and low-latency network connections.

[Figure 15-1](#) shows the relationship between regions and AZs.

Figure 15-1 Regions and AZs



Region Selection

You are advised to select a region close to you or your target users. This reduces network latency and improves the access success rate.

AZ Selection

When determining whether to deploy resources in the same AZ, consider your application's requirements for disaster recovery (DR) and network latency.

- To prioritize DR capabilities, deploy resources in different AZs in the same region.
- To prioritize network latency, deploy resources in the same AZ.

Regions and Endpoints

Before using an API to call resources, you will need to specify the resource region and endpoint. For details, see "Endpoints" in *Cloud Search Service API Reference*.

15.1.2 How Does CSS Ensure Data and Service Security?

CSS uses network isolation, in addition to various host and data security measures.

- Network isolation
The entire network is divided into two planes: service plane and management plane. The two planes are deployed and isolated physically to ensure the security of the service and management networks.
 - Service plane: This is the network plane of the cluster. It provides service channels for users and delivers data definitions, indexing, and search capabilities.
 - Management plane: This is the management console, where you manage CSS.
- Host security
CSS provides the following security measures:
 - The VPC security group ensures the security of the hosts in a VPC.
 - Network access control lists (ACLs) allow you to control what data can enter or exit your network.
 - The internal security infrastructure (including the network firewall, intrusion detection system, and protection system) monitors all network traffic that enters or exits the VPC through an IPsec VPN.
- Data security
Multiple replicas, cross-AZ deployment of clusters, and third-party (OBS) backup of index data ensure the security of user data.

15.1.3 Which CSS Metrics Should I Focus On?

Disk usage and cluster health status are two key metrics that you can focus on. You can log in to Cloud Eye and configure alarm rules for these metrics. If alarms are reported, handle them by taking appropriate measures.

Configuration examples:

- Alarms are reported if the disk usage is higher than or equal to a specified value (for example, 85%) and has reached this value multiple times (for example, 5 times) within a specified time period (for example, 5 minutes).
- Alarms are reported if the value of the cluster health status metric exceeds 0 for multiple times (for example, 5 times) within a specified time period (for example, 5 minutes).

Measures:

- If disk usage alarms are reported, view available disk space, check whether data can be deleted from cluster nodes or archived to other systems to free up space, or check if you can expand the disk capacity.
- If cluster health status alarms are reported, check whether shard allocation is normal, whether shards have been lost, and check whether the process has been restarted on Cerebro.

15.1.4 What Storage Options Does CSS Provide?

CSS uses EVS and local disks to store your indices. During cluster creation, you can specify the EVS disk type and specifications (the EVS disk size).

- Supported EVS disk types include common I/O, high I/O, and ultra-high I/O.
- The EVS disk size varies depending on the node specifications selected when you create a cluster.

15.1.5 What Is the Maximum Storage Capacity of CSS?

You can configure up to 200 nodes for a cluster (each node corresponds to an ECS). The maximum storage capacity of an ECS is the total capacity of EVS disks attached to the ECS. You can calculate the total storage capacity of CSS based on the sizes of EVS disks attached to different ECSs. The EVS disk size is determined by the node specifications selected when you create the cluster.

15.1.6 How Can I Manage CSS?

You can use any of the following three methods to manage CSS or to use search engine APIs. You can initiate requests based on constructed request messages.

- curl
curl is a command-line tool used to transfer data to or from a given URL. It serves as an HTTP client that can send HTTP requests to the HTTP server and receive response messages. You can also use curl to debug APIs. For more information about curl, visit <https://curl.haxx.se/>.
- Code
You can call APIs through code to assemble, send, and process request messages.
- REST client
Both Mozilla Firefox and Google Chrome provide a graphical browser plugin, the REST client, which you can use to send and process requests.
 - For Mozilla Firefox, see [Firefox REST Client](#).
 - For Google Chrome, see [Postman](#).

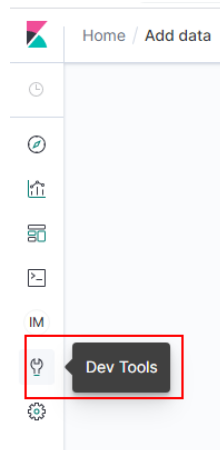
15.1.7 What Can the Disk Space of a CSS Cluster Be Used For?

You can store the following logs and files:

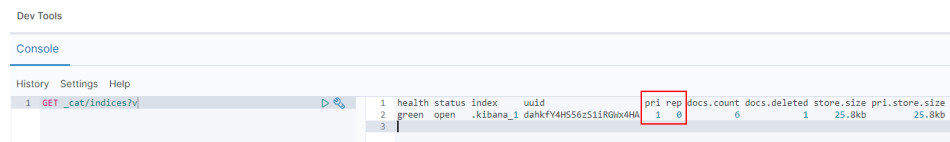
- Log files: Elasticsearch logs
- Data files: Elasticsearch index files
- Other files: cluster configuration files
- OS: 5% storage space reserved for the OS by default

15.1.8 How Do I Check the Numbers of Shards and Replicas in a Cluster on the CSS Console?

1. Log in to the console.
2. On the **Clusters** page, click **Access Kibana** in the **Operation** column of a cluster.
3. Log in to Kibana and choose **Dev Tools**.



4. On the **Console** page, run the **GET _cat/indices?v** command query the number of shards and replicas in a cluster. In the following figure, the **pri** column indicates the number of index shards, and the **rep** column indicates the number of replicas. After an index is created, its **pri** value cannot be modified. Its **rep** value can be modified.



15.1.9 What Data Compression Algorithms Does CSS Use?

CSS supports two data compression algorithms: LZ4 (by default) and best_compression.

- **LZ4 algorithm**

LZ4 is the default compression algorithm of Elasticsearch. This algorithm can compress and decompress data quickly, but its compression ratio is low.

LZ4 scans data with a 4-byte window, which slides 1 byte forward at a time. Duplicate data is compressed. This algorithm applies to scenarios where a large amount of data to be read while a small amount of data to be written.

- **best_compression algorithm**

This algorithm can be used when a large amount of data is written and the index storage cost is high, such as logs and time sequence analysis. This algorithm can greatly reduce the index storage cost.

Run the following command to switch the default compression algorithm (LZ4) to best_compression:

```
PUT index-1
{
  "settings": {
    "index": {
      "codec": "best_compression"
    }
  }
}
```

The LZ4 algorithm can quickly compress and decompress data while the best_compression algorithm has a higher compression and decompression ratio.

15.2 Functions

15.2.1 Can Elasticsearch Data Be Migrated Between VPCs?

Elasticsearch does not support direct data migration between different VPCs. You can use either of the following methods to migrate data.

Method 1

Use the backup and restoration function to migrate cluster data.

Method 2

1. Connect the VPC network and establish a VPC peering connection.
2. After the network is connected, use Logstash to migrate data.

15.2.2 How Do I Migrate a CSS Cluster Across Regions?

CSS clusters cannot be directly migrated. You can back up a cluster to an OBS bucket and restore it to a new region.

- If the OBS bucket is in the same region as your CSS cluster, migrate the cluster by following the instructions in Index Backup and Restoration.
- If the OBS bucket is not in the same region as your CSS cluster, configure cross-region replication to back up the cluster to the bucket, and migrate the cluster by following the instructions in Index Backup and Restoration.

 NOTE

- Before cross-region replication, ensure the snapshot folder of the destination cluster is empty. Otherwise, the snapshot information cannot be updated to the snapshot list of the destination cluster.
- Before every migration, ensure the folder is empty.

15.2.3 How Do I Configure the Threshold for CSS Slow Query Logs?

The slow query log settings of CSS are the same as those of Elasticsearch. You can configure slow query logs via the `_settings` API. For example, you can run the following command in Kibana to set the index level:

```
PUT /my_index/_settings
{
  "index.search.slowlog.threshold.query.warn": "10s",
  "index.search.slowlog.threshold.fetch.debug": "500ms",
  "index.indexing.slowlog.threshold.index.info": "5s"
}
```

- If a query takes longer than 10 seconds, a WARN log will be generated.
- If retrieval takes longer than 500 milliseconds, a DEBUG log will be generated.
- If an index takes longer than 5 seconds, an INFO log will be generated.

For details, visit the official website: <https://www.elastic.co/guide/cn/elasticsearch/guide/current/logging.html>

15.2.4 How Do I Update the CSS Lifecycle Policy?

The CSS lifecycle is implemented using the Index State Management (ISM) of Open Distro. For details about how to configure policies related to the ISM template, see the [Open Distro documentation](#).

1. When a policy is created, the system writes a record to the `.opendistro-ism-config` index. In the record, `_id` is the policy name, and the content is the policy definition.

Figure 15-2 Writing a data record

```
{
  "_index": ".opendistro-ism-config",
  "_type": "_doc",
  "id": "policy1",
  "score": 1.0,
  "source": {
    "policy": {
      "policy_id": "policy1",
      "description": "A simple default policy that changes the replica count between hot and cold states.",
      "last_updated_time": 1641432150329,
      "schema_version": 1,
      "error_notification": null,
      "default_state": "hot",
      "states": [
        {
          "name": "hot",
          "actions": [ ],
          "transitions": [
            {
              "state_name": "delete",
              "conditions": {
                "min_index_age": "2d"
              }
            }
          ]
        },
        {
          "name": "delete",
          "actions": [
            {
              "delete": { }
            }
          ],
          "transitions": [ ]
        }
      ]
    }
  }
}
```

2. After a policy is bound to an index, the system writes another record to the **.opendistro-ism-config** index. The following figure shows the initial status of a record.

Figure 15-3 Initial data status

```
{
  "_index": ".opendistro-ism-config",
  "_type": "_doc",
  "_id": "FABkSF5GSTCmR0Qkw41HVw",
  "_score": 1.0,
  "source": {
    "managed_index": {
      "name": "data1",
      "enabled": true,
      "index": "data1",
      "index_uuid": "FABkSF5GSTCmR0Qkw41HVw",
      "schedule": {
        "interval": {
          "start_time": 1641432652693,
          "period": 1,
          "unit": "Minutes"
        }
      },
      "last_updated_time": 1641432652694,
      "enabled_time": 1641432652694,
      "policy_id": "policy1",
      "policy_seq_no": null,
      "policy_primary_term": null,
      "policy": null,
      "change_policy": null
    }
  }
}
```

3. Run the **explain** command. Only a policy ID will be returned.

```
GET _opendistro/_ism/explain/data2
{
  "data2": {
```

```
"index.opendistro.index_state_management.policy_id" : "policy1"
}
}
```

Open Distro will execute an initialization process to fill the policy content in the record. The following figure shows the initialized data.

Figure 15-4 Initialized data

```

_index : ".opendistro-ism-config",
_type : "_doc",
_id : "FABKSF5G5TCmR0QkH41HW",
_score : 1.0,
_source : {
  "managed_index" : {
    "name" : "data1",
    "enabled" : true,
    "index" : "data1",
    "index_uuid" : "FABKSF5G5TCmR0QkH41HW",
    "schedule" : {
      "interval" : {
        "start_time" : 1641432652693,
        "period" : 1,
        "unit" : "Minutes"
      }
    },
    "last_updated_time" : 1641432652694,
    "enabled_time" : 1641432652694,
    "policy_id" : "policy1",
    "policy_seq_no" : 3,
    "policy_primary_term" : 1
  },
  "policy" : {
    "policy_id" : "policy1",
    "description" : "A simple default policy that changes the replica count between hot and cold states.",
    "last_updated_time" : 1641432158329,
    "schema_version" : 1,
    "error_notification" : null,
    "default_state" : "hot",
    "states" : [
      {
        "name" : "hot",
        "actions" : [ ],
        "transitions" : [
          {
            "state_name" : "delete",
            "conditions" : {
              "min_index_age" : "2d"
            }
          }
        ]
      },
      {
        "name" : "delete",
        "actions" : [
          {
            "delete" : { }
          }
        ],
        "transitions" : [ ]
      }
    ]
  },
  "change_policy" : null
}
}

```

After the initialization, **min_index_age** in the policy will be copied.

NOTE

The initialized index uses a copy of this policy. The policy update will not take effect on the index.

4. After the policy is modified, call the **change_policy** API to update the policy.

```
POST _opendistro/_ism/change_policy/data1
{
  "policy_id": "policy1"
}
```

15.2.5 How Do I Set the Numbers of Index Copies to 0 in Batches?

1. Log in to the Kibana page of the cluster. In the navigation pane, choose **Dev Tools**.
2. Modify and run the **PUT /*/_settings{"number_of_replicas":0}** command.

NOTE

Do not directly run the preceding command, because the asterisk (*) may match security indexes. You are advised to specify the index required for the batch operation. Example: **PUT /test/*/_settings{"number_of_replicas":0}**

15.2.6 Why All New Index Shards Are Allocated to the Same Node?

Possible Cause

The possible causes are as follows:

- Shards were unevenly distributed in previous index allocations, and the predominate parameter in the latest indexed shard allocation was **balance.shard**. To balance the shard distribution across nodes, the new shards were allocated to the node with only a small number of shards.
- After a new node was added to a cluster and before the automatic cluster rebalancing completes, the predominate parameter was **balance.shard**. The shards of a new index are allocated to the new node, where there are no shards yet.

The following two parameters are used to balance the shard allocation in a cluster:

`cluster.routing.allocation.balance.index` (default value: **0.45f**)

`cluster.routing.allocation.balance.shard` (default value: **0.55f**)

NOTE

- **balance.index**: A larger value indicates that all the shards of an index are more evenly distributed across nodes. For example, if an index has six shards and there are three data nodes, two shards will be distributed on each node.
- **balance.shard**: A larger value indicates that all the shards of all the indexes are more evenly distributed across nodes. For example, if index **a** has two shards, index **b** has four, and there are three data nodes, two shards will be distributed on each node.
- You can specify both **balance.index** and **balance.shard** to balance the shard allocation.

Solution

To prevent the all the shards of an index from being allocated to a single node, use either of the following methods:

1. To create an index during cluster scale-out, configure the following parameter:

```
"index.routing.allocation.total_shards_per_node": 2
```

That is, allow no more than two shards of an index to be allocated on each node. Determine the maximum number of shards allocated to each node based on the number of data nodes in your cluster and the number of index shards (both primary and secondary).
2. If too many shards are distributed on only a few nodes, you can move some of the shards to other nodes to balance the distribution. Run the **move** command of **POST _cluster/reroute**. The rebalance module will automatically exchange the shard with a shard on the destination node. Determine the values of **balance.index** and **balance.shard** as needed.


15.2.7 How Do I Query Snapshot Information?

Prerequisites

The snapshot function has been enabled for the cluster and snapshot information has been configured.

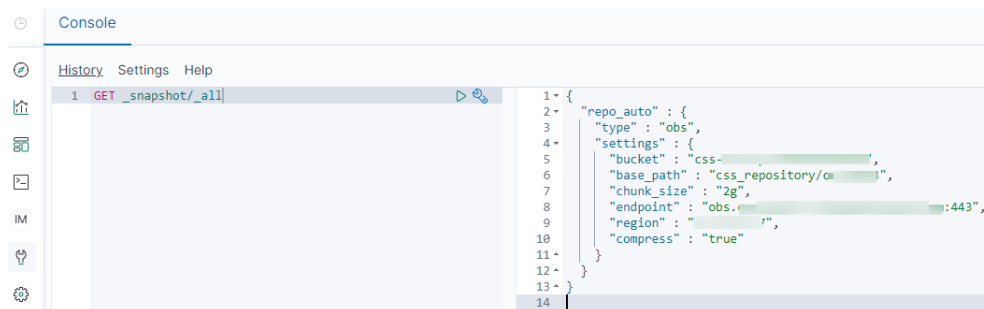
Querying a Snapshot

1. Log in to the CSS management console, and click **Clusters** in the navigation pane. On the displayed **Clusters** page, locate the target cluster and click **Access Kibana** in the **Operation** column.
2. In the left navigation pane of the Kibana page, click **Dev Tools**. Click **Get to work** to switch to the **Console** page.

Enter the code as required in the left pane, click  to execute the command, and view the result in the right pane.

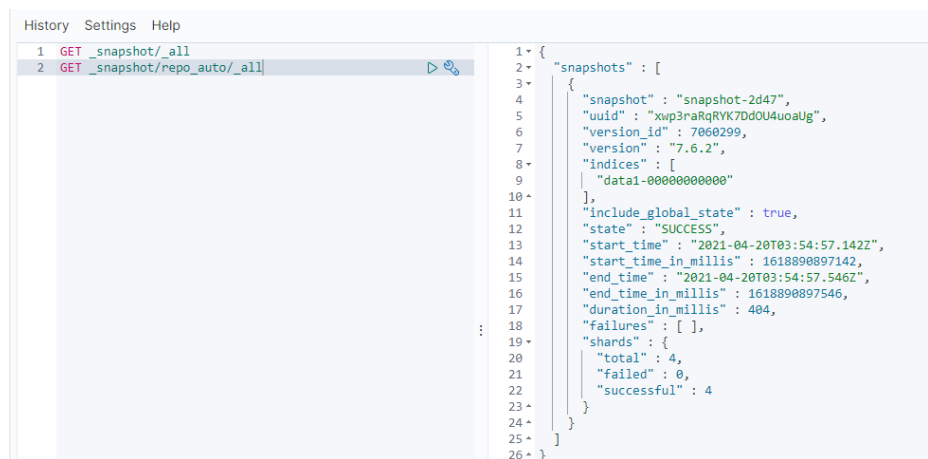
3. Run the **GET _snapshot/_all** command to query information about all repositories.

Figure 15-5 Querying information about all repositories



- **bucket**: OBS bucket name
 - **base_path**: Path. It consists of a fixed prefix and a cluster name.
 - **endpoint**: OBS domain name
 - **region**: your region
4. Query snapshot information.
 - a. Run the **GET _snapshot/repo_auto/_all** command to query the list of all the snapshots in the current repository.

Figure 15-6 Snapshot information



- **snapshot:** snapshot name
 - **state:** snapshot status
 - **start_time, start_time_in_millis, end_time, and end_time_in_millis:** snapshot time
 - **shards:** the number of shards. **total** indicates the total number of shards. **failed** indicates the number of failures. **successful** indicates the number of successes.
- b. Run the **GET _snapshot/repo_auto/\$snapshot-xxx** command to query information about a specified snapshot.
- Replace **\$snapshot-xxx** with the actual snapshot name.
 - **repo_auto** is followed by a snapshot name or wildcard characters.
5. (Optional) Delete information about a specified snapshot.
To delete a specific snapshot, run the **DELETE _snapshot/ repo_auto/ \$snapshot-xxx** command.
Replace **\$snapshot-xxx** with the actual snapshot name.

15.2.8 Can I Upgrade a Cluster from an Earlier Version to a Later Version?

A cluster cannot be directly upgraded. You can purchase a cluster of a later version and migrate your data to it.

1. **Creating a Cluster:** Create a cluster of a later version in the region where your current cluster is deployed.
2. **Migrating a Cluster:** Migrate your cluster by backing data up and restoring indexes.

15.2.9 Can I Restore a Deleted Cluster?

Yes. You can use a snapshot stored in OBS to restore a cluster. A deleted cluster that has no snapshots in OBS cannot be restored. Exercise caution when deleting a cluster.

To restore a deleted cluster, perform the following steps:

1. Log in to the CSS management console.
2. In the navigation pane on the left, choose **Clusters**. On the displayed **Clusters** page, click **Create Cluster** in the upper right corner to create a cluster and enable the snapshot function. Set the OBS bucket and backup path to those of the cluster to be restored.

To restore a deleted cluster to an existing cluster, set the OBS bucket and backup path to those of the deleted cluster.

NOTICE

To restore a deleted cluster to a new cluster, ensure they are in the same region. The new cluster version must be the same as or later than that of the deleted cluster. The number of nodes in the new cluster must be greater than half of that in the deleted cluster. Otherwise, the cluster may fail to be restored.

-
3. If the status of the new cluster changes to **Available**, click the cluster name to go to the **Cluster Information** page.
 4. In the navigation pane on the left, choose **Cluster Snapshots**.
In the snapshot management list, you can view the snapshot information of the deleted cluster. If no information is displayed, wait for several minutes and refresh the page.
 5. Locate the target snapshot and click **Restore** in the **Operation** column. The **Restore** page is displayed.
 6. On the **Restore** page, set restoration parameters.

Index: Enter the name of the index you want to restore. If you do not specify any index name, data of all indexes will be restored. The value can contain 0 to 1,024 characters. Uppercase letters, spaces, and certain special characters (including "<|>/?.") are not allowed.

Rename Pattern: Enter a regular expression. Indexes that match the regular expression are restored. The default value **index_(.+)** indicates restoring data of all indices. The value contains 0 to 1,024 characters. Uppercase letters, spaces, and certain special characters (including "<|>/?.") are not allowed.

Rename Replacement: Enter the index renaming rule. The default value **restored_index_\$1** indicates that **restored_** is added in front of the names of all restored indexes. The value can contain 0 to 1,024 characters. Uppercase letters, spaces, and certain special characters (including "<|>/?.") are not allowed. You can set **Rename Replacement** only if you have specified **Rename Pattern**.

Cluster: Select the cluster that you want to restore. You can select the current cluster or others. However, you can only restore the snapshot to clusters whose status is **Available**. If the status of the current cluster is **Unavailable**,

you cannot restore the snapshot to the current cluster. If you choose to restore the snapshot to another cluster, ensure that the target cluster runs an Elasticsearch version that is not earlier than that of the current cluster. If you select another cluster and two or more indexes in the cluster have the same name, data of all indices with the same name as the name you specify will be overwritten. Therefore, exercise caution when you set the parameters.

7. Click **OK**. If restoration succeeds, **Task Status** of the snapshot in the snapshot list will change to **Restoration succeeded**, and the index data is generated again according to the snapshot information.

15.2.10 Can I Modify the TLS Algorithm of an Elasticsearch Cluster?

You can modify TLS algorithms in CSS 7.6.2 and later versions.

1. Log in to the CSS management console.
2. In the navigation pane, choose **Clusters**. The cluster list is displayed.
3. Click the name of the target cluster to go to the cluster details page.
4. Select **Parameter Configurations**, click **Edit**, expand the **Customize** parameter, and click **Add**.

Add the **opendistro_security.ssl.http.enabled_ciphers** parameter and set it to **['TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256', 'TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384']**.

NOTE

- If the parameter value contains multiple algorithm protocols, enclose the value with a pair of square brackets ([]). If the parameter value is a single algorithm protocol, enclose the value with a pair of single quotation marks(' ').
5. After the modification is complete, click **Submit**. In the displayed **Submit Configuration** dialog box, select the box indicating "I understand that the modification will take effect after the cluster is restarted." and click **Yes**.
If the **Status** is **Succeeded** in the parameter modification list, the modification has been saved.
 6. Return to the cluster list and choose **More > Restart** in the **Operation** column to restart the cluster and make the modification take effect.

15.2.11 How Do I Set the search.max_buckets Parameter for an ES Cluster?

Function

If the query results on shards exceed the upper limit of records that can be returned (default value: **10000**), you need to increase the limit by changing the value of **search.max_buckets**.

Solution

Run the following command on the **Dev Tools** page of Kibana:

```
PUT _cluster/settings
{
```

```
"persistent": {  
  "search.max_buckets": 20000  
}
```

15.2.12 Does the Value `i` of `node.roles` Indicate an Ingest Node?

Function

If the value of `node.roles` of a client node is `i`, then is this client node an ingest node?

- Are there coordinating only nodes in clusters? Are the client requests distributed to coordinating nodes?
- Are ingest nodes in the idle state when there are no ingest requests?

Solution

If the value of `node.roles` of a client node is `i`, the ingest node mode is enabled.

- The coordinating only nodes of Elasticsearch are called client nodes in CSS. If a cluster has no client nodes, client requests will be distributed to all nodes.
- An ingest node functions as a set of ELK for data conversion. If there is no ingest requests, ingest nodes are not in the idle state.

15.2.13 How Do I Create a Type Under an Index in an Elasticsearch 7.x Cluster?

In Elasticsearch 7.x and later versions, types cannot be created for indexes.

If you need to use types, add `include_type_name=true` to the command. For example:

```
PUT _template/urldialinfo_template?include_type_name=true
```

After the command is executed, the following information is displayed:

```
"#! Deprecation: [types removal] Specifying include_type_name in put index template requests is deprecated. The parameter will be removed in the next major version. "
```

15.3 Clusters in Security Mode

15.3.1 What Is the Relationship Between the Filebeat Version and Cluster Version?

- Non-security mode: no restrictions.
- Cluster in security mode: The Filebeat OSS version must match the cluster version. For details on how to download the Filebeat OSS version, see [Past Releases of Elastic Stack Software](#).

15.3.2 How Do I Obtain the Security Certificate of CSS?

The security certificate (**CloudSearchService.cer**) can be downloaded only for security clusters that have enabled HTTPS access.

1. Log in to the CSS management console.
2. In the navigation pane, choose **Clusters**. The cluster list is displayed.
3. Click the name of a cluster to go to the cluster details page.
4. On the **Configuration** page, click **Download Certificate** next to .

15.3.3 How Do I Convert the Format of a CER Security Certificate?

The security certificate (**CloudSearchService.cer**) can be downloaded only for security clusters that have enabled HTTPS access. Most software supports certificates in the **.pem** or **.jks** format. You need to convert the format of the CSS security certificate.

- Run the following command to convert the security certificate from **.cer** to **.pem**:

```
openssl x509 -inform der -in CloudSearchService.cer -out newname.pem
```

- Run the following command to convert the security certificate from **.cer** to **.jks**:

```
keytool -import -alias newname -keystore ./truststore.jks -file ./CloudSearchService.cer
```

In the preceding commands, *newname* indicates the user-defined certificate name.

After the command is executed, set the certificate password and confirm the password as prompted. Securely store the password. It will be used for accessing the cluster.

15.4 Resource Usage and Change

15.4.1 How Do I Clear Expired Data to Release Storage Space?

- Run the following command to delete a single index data record.

```
curl -XDELETE http://IP:9200/Index_name
```

NOTE

IP: the IP address of any node in the cluster

- Run the following command to delete all Logstash data of a day. For example, delete all data on June 19, 2017:

For a cluster in non-security mode: **curl -XDELETE 'http://IP:9200/logstash-2017.06.19*'**

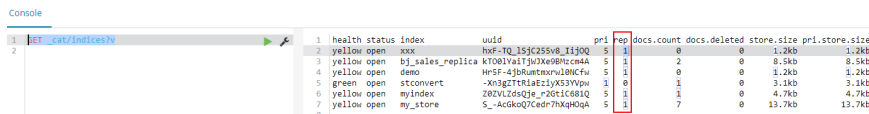
For a cluster in security mode: **curl -XDELETE -u username:password 'https://IP:9200/logstash-2017.06.19' -k**

 NOTE

- **username**: username of the administrator. The default value is **admin**.
- **password**: the password set during cluster creation
- **IP**: the IP address of any node in the cluster

15.4.2 How Do I Configure a Two-Replica CSS Cluster?

1. Run **GET _cat/indices?v** in Kibana to check the number of cluster replicas. If the value of **rep** is **1**, the cluster has two replicas.



health	status	index	uuid	pri	rep	docs.count	docs.deleted	store.size	pri.store.size
1	yellow	open	xxx	hxf-TQ_15jc235v0_I1j0Q	5	1	0	1.2kb	1.2kb
3	yellow	open	bj_sales_replica	kT081vaiTjWxK98tzm44	5	1	2	8.5kb	8.5kb
4	yellow	open	demo	HeSF-4j0kumtzmou10kCfu	5	1	0	1.2kb	1.2kb
5	green	open	stconvert	-X03gZTR1aE11yX53Vpw	1	0	1	3.1kb	3.1kb
6	yellow	open	myindex	Z8ZVLzdsQje_r26t1c681Q	5	1	1	4.7kb	4.7kb
7	yellow	open	my_store	S_-AcckoQTCedr7hskq0qa	5	1	7	0	13.7kb

2. If the value of **rep** is not **1**, run the following command to set the number of replicas:

```
PUT /index/_settings
{
  "number_of_replicas" : 1 //Number of replicas
}
```

 NOTE

index specifies the index name. Set this parameter based on site requirements.

15.4.3 How Do I Delete Index Data?

- Manually: Run the **DELETE /my_index** command in Kibana.
- Automatically: Create scheduled tasks to call the index deletion request and periodically execute the tasks. CSS supports Open Distro Index State Management. For details, see: <https://opendistro.github.io/for-elasticsearch-docs/docs/im/ism/>

15.4.4 Can I Change the Number of Shards to Four with Two Replicas When There Is One Shard Set in the JSON File?

Once an index is created, the number of primary shards cannot be changed.

You can run the following command in Kibana to change the number of replicas:

```
PUT /indexname/_settings
{
  "number_of_replicas":1 //Number of replicas
}
```

 NOTE

index specifies the index name. Set this parameter based on site requirements.

15.4.5 What Are the Impacts If an Elasticsearch Cluster Has Too Many Shards?

1. A large number of shards in a cluster slows down shard creation.

2. If automatic index creation is enabled, slow index creation may cause a large number of write requests to be stacked in the memory or result in a cluster break down.
3. If there are too many shards and you cannot properly monitor workloads, the number of records in a single shard may exceed the threshold, and write requests may be denied.

15.4.6 How Do I Set the Default Maximum Number of Records Displayed on a Page for an Elasticsearch Cluster

Solution

- Method 1

Open Kibana and run the following commands on the **Dev Tools** page:

```
PUT _all/_settings?preserve_existing=true
{
  "index.max_result_window" : "10000000"
}
```

- Method 2

Run the following commands in the background:

```
curl -XPUT 'http://localhost:9200/_all/_setting?preserve_existing=true'-d
{
  "index.max_result_window":"10000000"
}
```

⚠ CAUTION

This configuration consumes memory and CPU resources. Exercise caution when setting this parameter.

15.4.7 Why Does the Disk Usage Increase After the `delete_by_query` Command Was Executed to Delete Data?

Running the `delete_by_query` command can only add a deletion mark to the target data instead of really deleting it. When you search for data, all data is searched and the data with the deletion mark is filtered out.

The space occupied by an index with the deletion mark will not be released immediately after you call the disk deletion API. The disk space is released only when the segment merge is performed next time.

Querying the data with deletion mark occupies disk space. In this case, the disk usage increases when you run the disk deletion commands.

15.4.8 How Do I Clear the Cache of a CSS Cluster?

- Clear the `fielddata`

During aggregation and sorting, data are converted to the `fielddata` structure, which occupies a large amount of memory.

- a. Run the following commands on Kibana to check the memory occupied by index fielddata:

```
DELETE /_search/scroll
{
  "scroll_id" :
  "DXF1ZXJ5QW5kRmV0Y2gBAAAAAAAAAD4WYm9laVYtZndUQlNsdDcwakFMNjU1QQ=="
}
```

- b. If the memory usage of fielddata is too high, you can run the following command to clear fielddata:

```
POST /test/_cache/clear?fielddata=true
```

In the preceding command, *test* indicates the name of the index whose fielddata occupies a large amount of memory.

- **Clear segments**

The FST structure of each segment is loaded to the memory and will not be cleared. If the number of index segments is too large, the memory usage will be high. You are advised to periodically clear the segments.

- a. Run the following command on Kibana to check the number of segments and their memory usage on each node:

```
GET /_cat/nodes?v&h=segments.count,segments.memory&s=segments.memory:desc
```

- b. If the memory usage of segments is too high, you can delete or disable unnecessary indexes, or periodically combine indexes that are not updated.

- **Clear the cache**

Run the following command on Kibana to clear the cache:

```
POST _cache/clear
```

15.4.9 The Average Memory Usage of an Elasticsearch Cluster Reaches 98%

Symptom

The cluster monitoring result shows that the average memory usage of a cluster is 98%. Does it affect cluster performance?

Possible Cause

In an ES cluster, 50% of the memory is occupied by Elasticsearch and the other 50% is used by Lucene to cache files. It is normal that the average memory usage reaches 98%.

Solution

You can monitor the cluster memory usage by checking the maximum JVM heap usage and average JVM heap usage.

15.5 Components

15.5.1 Can I Install Search Guard on CSS?

CSS does not currently support installation of Search Guard.

CSS provides clusters in security mode, which have the same functions as Search Guard.

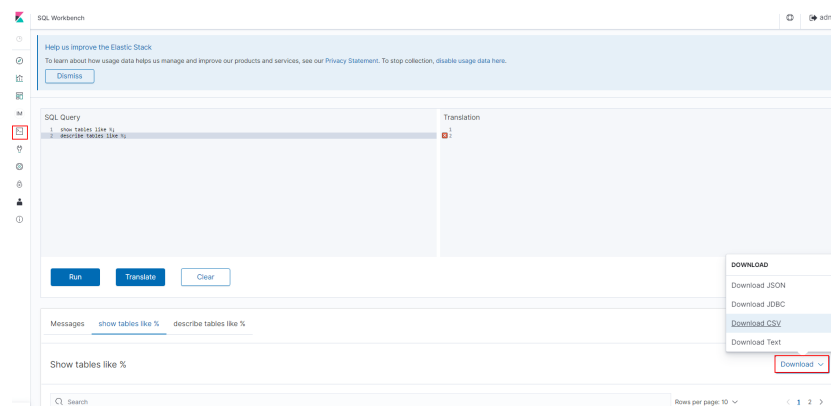
15.6 Kibana

15.6.1 Can I Export Data from Kibana?

Exporting data from Kibana requires the SQL Workbench plugin. Currently, you can only export data from Kibana 7.6.2 or later.

In SQL Workbench of Kibana, you can enter Elasticsearch SQL statements to query data or click **Download** to export data. You can export 1 to 200 data records. By default, 200 data records are exported.

Figure 15-7 SQL Workbench



15.6.2 How Do I Query Index Data on Kibana in an ES Cluster?

Run the following command to query index data through an API on Kibana:

```
GET indexname/_search
```

The returned data is shown in the following figure.

Figure 15-8 Returned data

```
{
  "took": 5,
  "timed_out": false,
  "_shards": {
    "total": 5,
    "successful": 5,
    "skipped": 0,
    "failed": 0
  },
  "hits": {
    "total": 3,
    "max_score": 2.0794415,
    "hits": [
      {
        "_index": "book",
        "_type": "novel",
        "_id": "7",
        "_score": 2.0794415,
        "_source": {
          "author": "Elasticsearch",
          "title": "Elasticsearch",
          "word_count": 3000,
          "publish_date": "2017-10-01"
        }
      }
    ]
  }
}
```

NOTE

- **took**: How many milliseconds the query cost.
- **time_out**: Whether a timeout occurred.
- **_shard**: Data is split into five shards. All of the five shards have been searched and data is returned successfully. No query result fails to be returned. No data is skipped.
- **hits.total**: Number of query results. Three documents are returned in this example.
- **max_score**: Score of the returned documents. The document that is more relevant to your search criteria would have a higher score.
- **hits.hits**: Detailed information of the returned documents.

15.7 Clusters

15.7.1 Why Does My ECS Fail to Connect to a Cluster?

Perform the following steps to troubleshoot this problem:

1. Check whether the ECS instance and cluster are in the same VPC.
 - If they are, go to **2**.
 - If they are not, create an ECS instance and ensure that the ECS instance is in the same VPC as the cluster.
2. View the security group rule setting of the cluster to check whether port **9200** (TCP protocol) is allowed or port **9200** is included in the port range allowed in both the outbound and inbound directions.

- If it is allowed, go to [3](#).
 - If it is not allowed, switch to the VPC management console and configure the security group rule of the cluster to allow port **9200** in both the outbound and inbound directions.
3. Check whether the ECS instance has been added to a security group.
 - If the instance has been added to a security group, check whether the security group configuration rules are appropriate. You can view the **Security Group** information on the **Basic Information** tab page of the cluster. Then, go to step [4](#).
 - If the instance has not been added to the security group, go to the VPC page from the ECS instance details page, select a security group, and add the ECS to the group.
 4. Check whether the ECS instance can connect to the cluster.
`ssh <Private network address and port number of a node>`

NOTE

- If the cluster contains multiple nodes, check whether the ECS can be connected to each node in the cluster.
- If the connection is normal, the network is running properly.
 - If the connection still fails, contact technical support.

15.7.2 Can a New Cluster Use the IP Address of the Old Cluster?

No.

15.7.3 Can I Associate My EIP If I Want to Access the Cluster from the Internet?

No. To access a cluster from the Internet, see [Public IP Address Access](#).

15.7.4 Can I Use x-pack-sql-jdbc to Access CSS Clusters and Query Data?

No. Currently, CSS does not integrate the x-pack component.

15.8 Ports

15.8.1 Do Ports 9200 and 9300 Both Open?

Yes. Port 9200 is used by external systems to access CSS clusters, and port 9300 is used for communication between nodes.

The methods for accessing port 9300 are as follows:

- If your client is in the same VPC and subnet with the CSS cluster, you can access it directly.

- If your client is in the same VPC with but different subnet from the CSS cluster, apply for a route separately.
- If your client is in the different VPCs and subnets from the CSS cluster, create a VPC peering connection to enable communication between the two VPCs, and then apply for routes to connect the two subnets.

16 Change History

Released On	Description
2023-06-20	<p>Added:</p> <ul style="list-style-type: none"> • Added cluster version 7.10.2. • Product Components • Quotas • Creating a User and Granting Permissions • Deploying a Cross-AZ Cluster • Overview • Scaling Out a Cluster • Scaling in a Cluster • Using CDM to Import Data from OBS to Elasticsearch • Using DIS to Import Local Data to Elasticsearch • Vector Retrieval • Working with Kibana • Storage-Compute Decoupling • Flow Control • Index Monitoring • Enhanced Monitoring • Best Practices
2022-08-23	<ul style="list-style-type: none"> • Updated: <ul style="list-style-type: none"> - Constraints - Basic Concepts - Clusters in Security Mode - Getting Started with Elasticsearch - Using Kibana or APIs to Import Data to Elasticsearch • Deleted the Customizing Word Dictionaries
2022-10-09	Updated: Elasticsearch SQL

Released On	Description
2022-06-30	This is the first official release.