Alibaba Cloud E-MapReduce

Open Source Components

Issue: 20190321

Legal disclaimer

Alibaba Cloud reminds you to carefully read and fully understand the terms and conditions of this legal disclaimer before you read or use this document. If you have read or used this document, it shall be deemed as your total acceptance of this legal disclaimer.

1. **You shall download and obtain this document from the Alibaba Cloud website or other Alibaba Cloud-authorized channels, and use this document for your own legal business activities only. The content of this document is considered confidential information of Alibaba Cloud. You shall strictly abide by the confidentiality obligations. No part of this document shall be disclosed or provided to any third party for use without the prior written consent of Alibaba Cloud.**
2. **No part of this document shall be excerpted, translated, reproduced, transmitted, or disseminated by any organization, company, or individual in any form or by any means without the prior written consent of Alibaba Cloud.**
3. **The content of this document may be changed due to product version upgrades , adjustments, or other reasons. Alibaba Cloud reserves the right to modify the content of this document without notice and the updated versions of this document will be occasionally released through Alibaba Cloud-authorized channels. You shall pay attention to the version changes of this document as they occur and download and obtain the most up-to-date version of this document from Alibaba Cloud-authorized channels.**
4. **This document serves only as a reference guide for your use of Alibaba Cloud products and services. Alibaba Cloud provides the document in the context that Alibaba Cloud products and services are provided on an "as is", "with all faults " and "as available" basis. Alibaba Cloud makes every effort to provide relevant operational guidance based on existing technologies. However, Alibaba Cloud hereby makes a clear statement that it in no way guarantees the accuracy, integrity , applicability, and reliability of the content of this document, either explicitly or implicitly. Alibaba Cloud shall not bear any liability for any errors or financial losses incurred by any organizations, companies, or individuals arising from their download, use, or trust in this document. Alibaba Cloud shall not, under any circumstances, bear responsibility for any indirect, consequential, exemplary, incidental, special, or punitive damages, including lost profits arising from the use**

or trust in this document, even if Alibaba Cloud has been notified of the possibility of such a loss.

5. By law, all the content of the Alibaba Cloud website, including but not limited to works, products, images, archives, information, materials, website architecture, website graphic layout, and webpage design, are intellectual property of Alibaba Cloud and/or its affiliates. This intellectual property includes, but is not limited to, trademark rights, patent rights, copyrights, and trade secrets. No part of the Alibaba Cloud website, product programs, or content shall be used, modified , reproduced, publicly transmitted, changed, disseminated, distributed, or published without the prior written consent of Alibaba Cloud and/or its affiliates . The names owned by Alibaba Cloud shall not be used, published, or reproduced for marketing, advertising, promotion, or other purposes without the prior written consent of Alibaba Cloud. The names owned by Alibaba Cloud include, but are not limited to, "Alibaba Cloud", "Aliyun", "HiChina", and other brands of Alibaba Cloud and/or its affiliates, which appear separately or in combination, as well as the auxiliary signs and patterns of the preceding brands, or anything similar to the company names, trade names, trademarks, product or service names, domain names, patterns, logos, marks, signs, or special descriptions that third parties identify as Alibaba Cloud and/or its affiliates).
6. Please contact Alibaba Cloud directly if you discover any errors in this document.

Generic conventions

Table -1: Style conventions

Style	Description	Example
	This warning information indicates a situation that will cause major system changes, faults, physical injuries, and other adverse results.	 Danger: Resetting will result in the loss of user configuration data.
	This warning information indicates a situation that may cause major system changes, faults, physical injuries, and other adverse results.	 Warning: Restarting will cause business interruption. About 10 minutes are required to restore business.
	This indicates warning information, supplementary instructions, and other content that the user must understand.	 Notice: Take the necessary precautions to save exported data containing sensitive information.
	This indicates supplemental instructions, best practices, tips, and other content that is good to know for the user.	 Note: You can use Ctrl + A to select all files.
>	Multi-level menu cascade.	Settings > Network > Set network type
Bold	It is used for buttons, menus , page names, and other UI elements.	Click OK.
`Courier font`	It is used for commands.	Run the `cd /d C:/windows` command to enter the Windows system folder.
Italics	It is used for parameters and variables.	`bae log list --instanceid` *Instance_ID*
[] or [a|b]	It indicates that it is a optional value, and only one item can be selected.	`ipconfig` *[-all|-t]*

Style	Description	Example
{} or {a|b}	It indicates that it is a required value, and only one item can be selected.	`swich {stand | slave}`

Contents

1 Hue

E-MapReduce currently supports *Hue*, which you can access through Apache Knox. The following section provides an overview of how to use Hue.

Preparation

In the *#unique_4* cluster, *set the security group rules*, and open port 8888.

> Notice:
> Set security group rules for limited IP ranges. IP 0.0.0.0/0 is not allowed to add into the security group.

Access Hue

To access Hue, complete the following steps:

1. In the EMR console, click Manage to the right of the cluster ID.
2. On the left side of the Configuration page, click Access Links and Ports.

View the password

If Hue does not have an administrator after the first running, the first user to log on is set automatically to administrator. For security, E-MapReduce generates an administrator account and password by default. The administrator account is admin. To view the password, complete the following steps:

1. Click Manage to the right of the cluster ID.
2. In the Clusters and Services panel, click Hue.
3. Click the Configuration tab to go to the `admin_pwd` parameter. It is a random password.

Create a new account if you forget your password

If you forget your password for your Hue account, you can create a new account by completing the following steps:

1. In the cluster list page, click Manage next to the target cluster.
2. In the navigation panel on the left, click Cluster Overview.
3. In the Core Instance Group, obtain the public network IPs of some master nodes.
4. Log on to the master node through SSH.

5. Execute the following command:

```
/ opt / apps / hue / build / env / bin / hue   createsupe  ruser
```

6. Enter a new user name, e-mail, and password, and press Enter.

   If Superuser created successfully is displayed, you have successfully created a new account. You can now log on to Hue with the new account.

Add or modify a configuration

1. In the cluster list page, click Manage next to the target cluster.
2. In the service list, click Hue, and then click the Configuration tab.
3. In the upper-right corner of the page, click Custom Configuration, and configure the Key and Value fields. The key must adhere to the following specifications:

```
$ section_pa  th .$ real_key
```

Note:

- `$ real_key` is the actual key to be added, such as `hive_serve  r_host .`
- In the *hue . ini* file, you can view the `$ section_pa  th` before the `$ real_key .`

  For example, if the `hive_serve  r_host` belongs to the `[ beeswax ]` section, this means that the `$ section_pa  th` is `beeswax .` If this is the case, the key to be added is `beeswax . hive_serve  r_host .`
-
- If you need to modify the multilevel section `[ desktop ] -> [[ ldap ]] -> [[[ ldap_serve  rs ]]] -> [[[[ users ]]]] -> user_name_  attr` value in the *hue . ini* file, the key to be configured is `desktop . ldap . ldap_serve  rs . users . user_name_  attr .`

2 Oozie

The following section provides an overview of how to use Oozie in a E-MapReduce cluster.

Note:
E-MapReduce version 2.0.0 and later support Oozie. If you need to use Oozie in a cluster, make sure that the version you are using is 2.0.0 or higher.

Preparations

Before you create a cluster, you must first open an SSH tunnel. For more information, see *#unique_7*.

In the following, which uses a MAC environment as an example, the IP address of the public network for the cluster's master node is assumed to be xx.xx.xx.xx:

1. Log on to the master node.

```
ssh   root @ xx . xx . xx . xx
```

2. Enter your password.
3. Check the `id_rsa . pub` content of the local machine. Note that this is executed on the local machine, not the remote master node.

```
cat  ~/. ssh / id_rsa . pub
```

4. Write the `id_rsa . pub` content of the local machine in *`~/. ssh / authorized _keys`* on the local master node, which is executed on the remote master node.

```
mkdir  ~/. ssh /
vim  ~/. ssh / authorized  _keys
```

5. Copy and paste the content observed in *Step 2*. You should now be able to log on to the master node without a password using `ssh   root @ xx . xx . xx . xx .`
6. Execute the following command on the local machine to perform port forwarding:

```
ssh  - i  ~/. ssh / id_rsa  - ND   8157   root @ xx . xx . xx . xx
```

7. Execute the following command to enable Chrome to in the new terminal on the local machine:

```
/ Applicatio  ns / Google \  Chrome . app / Contents / MacOS /
 Google \  Chrome  -- proxy - server =" socks5 :// localhost : 8157
```

```
" -- host - resolver - rules =" MAP  *  0 . 0 . 0 . 0  ,  EXCLUDE
localhost " -- user - data - dir =/ tmp
```

Access the Oozie UI interface

Access the following in Chrome to perform port forwarding: xx.xx.xx.xx:11000/oozie, localhost:11000/oozie, or intranet ip: 11000/oozie.

Submit a workflow job

Before you run Oozie, you first have to install *Oozie's ShareLib*.

In E-MapReduce clusters, ShareLib is installed by default for Oozie users. If you are using Oozie to submit a workflow job, you do not need to install ShareLib again.

Clusters with HA enabled use different methods to access NameNode and ResourceManager than clusters with HA disabled. Therefore, when you submit an Oozie workflow job, you need to specify a different NameNode and JobTracker (ResourceManager) in job.properties files. To do so, complete the following steps:

- Non-HA clusters

```
nameNode = hdfs :// emr - header - 1 : 9000
jobTracker = emr - header - 1 : 8032
```

- HA clusters

```
nameNode = hdfs :// emr - cluster
jobTracker = rm1 , rm2
```

In the following examples, configurations are made for both non-HA and HA clusters. For operations that do not require modification, the sample code can be used directly. For the specific format of a workflow file, see the relevant documentation on the *official Oozie website*.

- Submit a workflow job on a non-HA cluster

1. Log on to the main master node of the cluster.

```
 ssh   root @ publicIp_o  f_master
```

2. Download the sample code.

```
[ root @ emr - header - 1  ~]#  su   oozie
[ oozie @ emr - header - 1   root ]$  cd  / tmp
[ oozie @ emr - header - 1   tmp ]$  wget   http :// emr - sample
 - projects . oss - cn - hangzhou . aliyuncs . com / oozie -
 examples / oozie - examples . zip
```

```
[ oozie @ emr - header - 1   tmp ]$  unzip   oozie - examples .
 zip
```

3. Synchronize the Oozie workflow code to HDFS.

```
[ oozie @ emr - header - 1   tmp ]$  hadoop   fs  - copyFromLo
 cal   examples / / user / oozie / examples
```

4. Submit a sample Oozie workflow job.

```
[ oozie @ emr - header - 1   tmp ]$ $ OOZIE_HOME / bin / oozie
 job  - config   examples / apps / map - reduce / job . properties
  - run
```

After submitting the job successfully, a jobId is returned, for example:

```
 job :  0000000 - 1606271956  51086 - oozie - oozi - W
```

5. Go to the Oozie UI page to view the submitted Oozie workflow job.

- Submit a workflow job on an HA cluster

1. Log on to the main master node of the HA cluster.

```
 ssh   root @ main_maste  r_ip
```

To determine the current main master node, check whether the Oozie UI can be accessed or not. By default, the Oozie server service is enabled on the main master node `xx . xx . xx . xx : 11000 / oozie` .

2. Download the sample code.

```
[ root @ emr - header - 1  ~]#  su   oozie
[ oozie @ emr - header - 1   root ]$  cd  / tmp
[ oozie @ emr - header - 1   tmp ]$  wget   http :// emr - sample
 - projects . oss - cn - hangzhou . aliyuncs . com / oozie -
 examples / oozie - examples - ha . zip
```

```
[ oozie @ emr - header - 1   tmp ]$  unzip   oozie - examples - ha
 . zip
```

3. Synchronize the Oozie workflow code to HDFS.

```
[ oozie @ emr - header - 1   tmp ]$  hadoop   fs  - copyFromLo
 cal   examples / / user / oozie / examples
```

4. Submit a sample Oozie workflow job.

```
[ oozie @ emr - header - 1   tmp ]$ $ OOZIE_HOME / bin / oozie
 job  - config   examples / apps / map - reduce / job . properties
  - run
```

   After submitting the job successfully, a jobId is returned. This should be similar to:

```
 job :  0000000 - 1606271956  51086 - oozie - oozi - W
```

5. Go to the Oozie UI page to view the submitted Oozie workflow job.

3 Zeppelin

E-MapReduce can access Zeppelin through Apache Knox.

Preparation

1. In the *#unique_4* cluster, *set the security group rules*, and open port 8080.
2. In Knox, add a user name and password. For more information on how to set Knox users, see *#unique_9*. The user name and password are only used to log on to the various Knox services. They are not related to Alibaba Cloud RAM user names.

Notice:
Set security group rules for limited IP ranges. IP 0.0.0.0/0 is not allowed to add into the security group.

Access Zeppelin

To view the access links for Zeppelin, complete the following steps:

1. On the right of the cluster list page, click Manage.
2. In the pane on the left, click Access Links and Ports.

4 ZooKeeper

The *ZooKeeper* service is enabled in E-MapReduce clusters by default.

 Note:
ZooKeeper only has 3 nodes, regardless of how many machines are currently in the cluster. More nodes are not currently supported.

Create a cluster

When you create a cluster, select the Zookeeper service in the software configuration page.

Node information

After you have created a cluster and its status is idle, in the Clusters and Services page, select ZooKeeper, and then click Component Topology to view ZooKeeper nodes. E-MapReduce enables 3 ZooKeeper nodes. The corresponding intranet IP address (2181 is the default port) of ZooKeeper nodes are indicated in the IP column for access to the ZooKeeper service.

5 Kafka

5.1 Quick start

E-MapReduce 3.4.0 and later support Kafka.

Create a Kafka cluster

When creating a cluster on E-MapReduce, set the cluster type to Kafka. A cluster containing only Kafka components is created by default. The components include basic components, as well as Zookeeper, Kafka, and KafkaManager components. Only one Kafka broker is deployed on each node. We recommend that you use a dedicated Kafka cluster instead of mixing with Hadoop services.

Ephemeral disk Kafka clusters

To better reduce unit costs and respond to larger storage needs, E-MapReduce 3.5.1 supports Kafka clusters on local disks (D1 cluster models). For more information, see *ECS models*. Compared to cloud disks, local disk Kafka clusters have the following features:

- High-volume local SATA HDD disks with high I/O throughput, sequential read and write performance on a single disk of 190 MB/s, and up to 5 GB/s of storage I/O capability.
- Cost of local storage is 97% lower than that of SSD cloud disks.
- Higher network performance, with up to 17 Gbit/s instances of network bandwidth. This meets data interaction requirements for peak business instances.

Local disk models also have the following features:

Operation	Ephemeral disk data status	Description
Restart within the operating system/restart or force restart in the ECS console	Retained	The local ephemeral disk's storage volume is retained. Data is also retained.
Shut down within the operating system/Stop or force stop in the ECS console	Retained	The local ephemeral disk's storage volume is retained. Data is also retained.

Operation	Ephemeral disk data status	Description
Release (instances) on the console	Erased	The local ephemeral disk's storage volume is erased. Data is not retained.

Notice:

- When the host is down or the disk is corrupted, the data on the disk is lost.
- Do not store business data on a local ephemeral disk for a long period of time. Back up data in a timely manner and adopt a high-availability architecture. For long-term storage, we recommend that you store data on a cloud disk.

To be able to deploy Kafka on a local disk, E-MapReduce has the following default requirements:

1. `default . replicatio n . factor = 3` indicates that the number of partitions and replicas in the topic is at least three. If a smaller number of replicas is set, the risk of data loss is increased.
2. `min . insync . replicas = 2` indicates that when the producer is required to set acks to all (-1), it is considered successful to write at least two replicas at a time.

When a local disk corruption occurs, E-MapReduce performs the following:

1. Removes the bad disk from the broker configuration, restarts Broker, and recovers the lost data from the bad disk on the other available local disks. The time it takes to perform data recovery varies according to the amount of data that has been written on the broken disk.
2. When the number of damaged machine disks is over 20%, E-MapReduce takes the initiative to migrate the machine and restore the abnormal disk.
3. If there is not enough disk space available on the current machine to recover lost data on the damaged disk, Broker is shut down abnormally. If this is the case, you can choose to clean some data, free up disk space, or restart the Broker service. You can also open a ticket with E-MapReduce for machine migration and to recover abnormal disks.

Parameter description

You can check Kafka software configurations on the E-MapReduce cluster configuration management interface.

Configuration item	Description
zookeeper.connect	Zookeeper connection address configured on Kafka.
kafka.heap.opts	Size of the heap memory of the Kafka broker.
num.io.threads	Number of the Kafka broker's I/O threads, which by default is twice the number of CPU cores.
num.network.threads	Number of the Kafka broker's network threads, which by default is the same as the number of CPU cores.

5.2 Cross-cluster Kafka access

An independent Kafka cluster is deployed to provide the Kafka service. Therefore, you may need to access this service across clusters.

Cross-cluster access to Kafka

Cross-cluster access to Kafka consists of two types:

- Accessing E-MapReduce Kafka clusters from the Alibaba Cloud intranet network.
- Accessing E-MapReduce Kafka clusters from the public network.

Different solutions are prepared for different E-MapReduce versions.

EMR-3.11.x and later

- Access Kafka from the Alibaba Cloud intranet network

  You can access Kafka by using the intranet IP address of a Kafka cluster node. Use port 9092 to access Kafka from the intranet network.

  Make sure that the networks are accessible before you access Kafka:

  - For more information about how to access a VPC from a classic network, see *#unique_15* here.
  - For more information about how to access a VPC from another VPC, see *Configure a VPC-to-VPC connection*.

- Access Kafka in the public network

  The core node of the Kafka cluster is unable to access the public network by default. To access the Kafka cluster in the public network, complete the following steps:

  1. Interconnect Kafka clusters with the public network.
     - If Kafka clusters are deployed in a VPC environment, there are two ways to interconnect them:
       - Deploy Express Connect to interconnect the VPC with the public network. For details, see *Express Connect*.
       - Bind EIPs to cluster core nodes. For details, see *EIP*. The following steps bind the EIP to the ECS:
     - If Kafka is deployed in a classic network, there are two ways to interconnect them:
       - To create a Pay-As-You-Go cluster, use ECS APIs. For details, see *API*.
       - To create a Subscription cluster, you can directly assign a public IP address to the relevant host in the ECS console.
  2. Create an EIP in the *VPC console* and purchase the relevant EIPs based on the number of core nodes in the Kafka cluster.
  3. Configure security group rules that allow the Kafka cluster to control public network access to the cluster's IP addresses. This improves the security of the Kafka cluster exposed in the public network. You can view the security group to which the cluster belongs in the E-MapReduce console, and configure security group rules based on security group IDs. For more information, see *Security group rules*.
  4. On the Cluster Management page of the E-MapReduce console, click Manage next to the specified cluster, select Cluster Overview on the left side of the page, and then click Sync Cluster Host Info in the upper-right corner.
  5. Restart the Kafka cluster.
  6. Use the EIP of the Kafka cluster node to access Kafka in the public network. Use port 9093 to access Kafka from the public network.

Versions earlier than EMR-3.11.x

- Access Kafka from the Alibaba Cloud intranet network

  You must configure the host information of the Kafka cluster node on the client host. The Long domain of the Kafka cluster node must also be configured. For example:

```
/ etc / hosts
#  kafka   cluster
 10 . 0 . 1 . 23   emr - header - 1 . cluster - 48742
 10 . 0 . 1 . 24   emr - worker - 1 . cluster - 48742
 10 . 0 . 1 . 25   emr - worker - 2 . cluster - 48742
 10 . 0 . 1 . 26   emr - worker - 3 . cluster - 48742
```

- Access Kafka in the public network

  The core node of the Kafka cluster is unable to access the public network by default. To access the Kafka cluster in the public network, complete the following steps:

  1. Interconnect Kafka clusters with the public network.
     - If Kafka clusters are deployed in a VPC environment, there are two ways to interconnect them:
       - Deploy Express Connect to interconnect the VPC with the public network. For details, see *Express Connect*.
       - Bind EIPs to cluster core nodes. For details, see *EIP*. Complete the following steps to bind the EIP to the ECS.
     - If Kafka is deployed in a classic network, there are two ways to interconnect them:
       - To create a Pay-As-You-Go cluster, use ECS APIs. For details, see *API*.
       - To create a Subscription cluster, you can directly assign a public IP address to the relevant host in the ECS console.
  2. Create an EIP in the *VPC console* and purchase the relevant EIPs based on the number of core nodes in the Kafka cluster.
  3. Configure security group rules that allow the Kafka cluster to control public network access to the cluster's IP addresses. This improves the security of the Kafka cluster exposed in the public network. You can view the security group to which the cluster belongs in the E-MapReduce console, and configure security

group rules based on security group IDs. For more information, see *Security group rules*.

4. Modify the Kafka cluster's `listeners . address . principal` software configuration to `HOST` , and restart the Kafka cluster.
5. Configure the *`hosts`* file on the local client host.

5.3 Kafka Ranger

With E-MapReduce 3.12.0 and later, Kafka allows you to configure permissions with Ranger.

Integrate Ranger into Kafka

To integrate Ranger into Kafka, complete the following steps:

- Enable Kakfa Plugin

  1. On the Cluster Management page, click Ranger in the service list to enter the Ranger Management page. Click Operation in the upper-right corner and select Enable Kafka PLUGIN.

  2. You can check the progress by clicking View Operation History in the upper-right corner of the page.

- Restart Kafka broker

  After enabling the Kafka plugin, you must restart the broker to make it take effect.

  1. On the Cluster Management page, click the inverted triangle icon behind RANGER in the upper-left corner to switch to Kafka.
  2. Click Actions in the upper-right corner of the page and select RESTART Broker.
  3. You can check the progress by clicking View Operation History in the upper-right corner of the page.

- Add Kafka service on the Ranger WebUI

  For more information about how to go to the Ranger WebUI, see *Ranger Introduction*.

  Add the Kafka service on the WebUI:

  Configure the Kafka service:

Configure permissions

After integrating Ranger into Kafka, you can set the relevant permissions.

> Notice:
> In a standard cluster, Ranger automatically generates the all - topic rule after the Kafka service is added. This rule indicates that there are no restrictions on permissions. All users can perform all actions. In this case, Ranger cannot identify permissions through the user.

Here, user_test is used as an example to add the Publish permission:

After you add a policy, the permissions are granted to the `test` user. This user can then perform the write operation for `test` .

> Note:
> The policy takes effect one minute later after it is added.

5.4 Kafka SSL

E-MapReduce Kafka supports the SSL function in E-MapReduce 3.12.0 and later.

Create a cluster

For details about how to create a cluster, see *#unique_24*.

Enable the SSL service

By default, the SSL function is not enabled for the Kafka cluster. You can enable it on the configuration page of the Kafka service.

As shown in the preceding figure, change `kafka.ssl.enable` to `true` and then restart the component.

Access Kafka from the client

You need to configure `security.protocol`, `truststore`, and `keystore` when you access Kafka through SSL. Take a standard mode cluster as an example. To run a job in a Kafka cluster, you can configure the cluster as follows:

```
security.protocol=SSL
ssl.truststore.location=/etc/ecm/kafka-conf/truststore
ssl.truststore.password=${password}
ssl.keystore.location=/etc/ecm/kafka-conf/keystore
ssl.keystore.password=${password}
```

If you are running a job in an environment other than a Kafka cluster, copy the truststore and keystore files (in the */etc/ecm/kafka-conf/* directory on any node of the cluster) in the Kafka cluster to the running environment and add configurations accordingly.

Take the producer and consumer programs in Kafka as an example.

1. Create the configuration file *ssl.properties* and add configuration items.

```
security.protocol=SSL
 ssl.truststore.location=/etc/ecm/kafka-conf/
truststore
 ssl.truststore.password=${password}
 ssl.keystore.location=/etc/ecm/kafka-conf/keystore
 ssl.keystore.password=${password}
```

2. Create a topic.

```
kafka-topics.sh --zookeeper emr-header-1:2181/kafka
-1.0.1 --replicatio n-factor 2 --
partitions 100 --topic test --create
```

3. Use an SSL configuration file to generate data.

```
kafka-producer-perf-test.sh --topic test --num-
records 123456 --throughput 10000 --record-size 1024
```

```
-- producer - props   bootstrap . servers = emr - worker - 1 : 9092
 -- producer . config   ssl . properties
```

4. Use an SSL configuration file to consume data.

```
kafka - consumer - perf - test . sh  -- broker - list   emr -
worker - 1 : 9092  -- messages   100000000  -- topic   test  --
consumer . config   ssl . properties
```

5.5 Kafka Manager

E-MapReduce 3.4.0 and later support Kafka Manager for use in managing Kafka clusters.

Procedure

Notice:
Kafka Manager software is installed by default and the Kafka Manager authentication function is enabled when a Kafka cluster is created. We strongly recommend that you change the default password when using Kafka Manager for the first time and access Kafka Manager through the SSH tunnel. We do not recommend that you expose Port 8085 to the public network unless an IP address whitelist is configured to avoid data leakage.

- We recommend that you access the web page through the SSH tunnel. For more information, see #unique_7.
- Access *http://localhost:8085*.
- Enter your user name and password. Refer to the configuration information of Kafka Manager.
- Add an existing Kafka cluster and make sure that the Zookeeper address of the Kafka cluster is correct. For more information, see the configuration information of Kafka Manager. Select the corresponding Kafka version. We recommend that you enable the JMX function.
- Common Kafka functions are available immediately after you create a Kafka cluster.

5.6 Common Kafka problems

This section describes two common issues with Kafka.

- `Error while executing topic command : Replicatio n factor : 1 larger than available brokers : 0 .`

  Common causes:

  - A fault occurs in the Kafka service and the cluster broker process exits. You need to use logs to troubleshoot the fault.
  - The ZooKeeper address of the Kafka service is incorrect. View and use the Zookeeper.connect configuration item on the Kafka configuration management page.

- `java . net . BindExcept ion : Address already in use ( Bind failed )`

  You may encounter this exception when you use Kafka command line tools. This is typically caused by the unavailability of the JMX port. You can specify a JMX port manually before using the command line. For example:

```
JMX_PORT = 10101 kafka - topics -- zookeeper emr - header - 1
: 2181 / kafka - 1 . 0 . 0 -- list
```

6 Druid

6.1 Introduction to Druid

Druid is a column-oriented, open-source, distributed data store used to query and analyze issues in large data sets in real time.

Basic features

Druid has the following features:

- Sub-second OLAP queries, including multi-dimensional filtering, ad-hoc attribute grouping, and fast data aggregation.
- Real-time data consumption, collection, and querying.
- Efficient multi-tenant capability, which enables thousands of users to perform searches online at the same time.
- Strong scalability, which supports the fast processing of PB-level data, 100 billion-level events, and thousands of concurrent queries per second.
- Extremely high availability and support for rolling upgrades.

Usage scenarios

Real-time data analysis is the most typical usage scenario for Druid and covers a wide range of areas, including:

- Real-time indicator monitoring
- Model recommendations
- Advertisement platforms
- Model searches

These scenarios involve large amounts of data, and the requirement for time delay in data querying is high. In real-time indicator monitoring, problems need to be detected at the moment of occurrence so that you can be warned as soon as possible. In the recommendation model, user behavior data needs to be collected in real time and sent to the recommendation system promptly. In just a few clicks, the system is able to identify your search intent and recommend more appropriate results in future searches.

Architecture

Druid has an excellent architectural design with multiple components working together to complete a series of processes, such as data collection, indexing, storage, and querying.

The following figure shows the components contained in the Druid working-layer (for data indexing and data querying).

- The real-time component is responsible for the real-time data collection.
- In the broker phase, query tasks are distributed, and the results are collected and returned to you.
- The historical node is responsible for the storage of historical data after indexing. The data is stored in deep storage. Deep storage can be either local or a distributed file system, such as HDFS.
- The indexing service consists of two components (not shown in the figure).
  - The Overlord component is responsible for managing and distributing indexing tasks.
  - The MiddleManager component is responsible for executing indexing tasks.

The following figure shows the components involved in the management layer of Druid segments (Druid index file).

- The ZooKeeper component is responsible for storing the status of the cluster and discovering components, such as the topology information of the cluster, election of the Overlord leader, and management of the indexing task.
- The Coordinator component is responsible for managing segments, such as the downloading and deletion of the segments and balancing them with historical components.
- The Metadata storage component is responsible for storing the meta-information of segments and managing all kinds of persistent or temporary data in the cluster, such as configuration information and audit information.

Product advantages

E-MapReduce Druid has improved a lot based on open-source Druid, including integration with E-MapReduce and the peripheral Alibaba Cloud ecosystem, easy

monitoring and operation support, and easy-to-use product interfaces. You can use it immediately after purchase. It does not need 24/7 operation and maintenance.

E-MapReduce Druid supports the following features:

- Using OSS as deep storage
- Using OSS files as data sources for indexing in batches
- Using RDS to store metadata
- Integrating with Superset tools
- Easy scale up and scale down (scale down is for task node)
- Diversified monitoring indicators and alarm rules
- Bad node migration
- High-security mode
- HA

6.2 Quick start

E-MapReduce 3.11.0 and later support Druid as a cluster type.

Druid is used as a separate cluster type (instead of being added to a Hadoop cluster) for the following reasons:

- Druid can be used independently of Hadoop.
- Druid has high memory requirements when there is a large amount of data, especially for Broker and Historical nodes. Druid is not controlled by YARN, and will compete for resources during multi-service operation.
- As an infrastructure, the number of nodes in a Hadoop cluster can be relatively large, whereas a Druid cluster can be relatively small. Using them together results in greater flexibility.

Create a Druid cluster

Select the Druid cluster type when you create a cluster. You can select HDFS and YARN when creating a Druid cluster for testing only. We strongly recommend that you use a dedicated Hadoop cluster as the production environment.

Configure a cluster

- Configure the cluster to use HDFS as deep storage for Druid

  For a standalone Druid cluster, you may need to store your index data in the HDFS of another Hadoop cluster. Therefore, you need to complete the settings for the connectivity between the two clusters. For details, see Interaction with *Hadoop clusters*. You then need to configure the following items on the Druid configuration page and restart the service. The configuration items are in common.runtime on the configuration page.

  - druid.storage.type: HDFS
  - druid.storage.storageDirectory: The HDFS directory must be complete, such as hdfs://emr-header-1.cluster-xxxxxxxx:9000/druid/segments.

   Note:
  If the Hadoop cluster is an HA cluster, you must change emr-header-1.cluster-xxxxx:9000 to emr-cluster, or change port 9000 to port 8020.

- Use OSS as deep storage for Druid

  E-MapReduce Druid supports the use of OSS as deep storage. Due to the AccessKey-free capability of E-MapReduce, Druid can automatically get access to OSS without having to configure the AccessKey. Because the OSS function of HDFS enables

Druid to have access to OSS, druid.storage.type still needs to be configured as HDFS: during the configuration process.

- druid.storage.type: HDFS
- druid.storage.storageDirectory: For example, oss://emr-druid-cn-hangzhou/segments.

Because the OSS function of HDFS enables Druid to have access to OSS, you need to select one of the following two scenarios:

- Install HDFS when you create a cluster. The system is then configured automatically. (After HDFS is installed, you can choose not to use it, disable it, or use it for testing purposes only.)
- Create `hdfs - site . xml` in the Druid configuration directory `/ etc / ecm / druid - conf / druid / _common /`. The content is as follows. Copy the file to the same directory of all nodes:

```
& lt ;?  xml   version =" 1 . 0 "? >
  < configurat  ion >
    < property >
      < name > fs . oss . impl </ name >
      < value > com . aliyun . fs . oss . nat . NativeOssF
 ileSystem </ value >
    </ property >
    < property >
      < name > fs . oss . buffer . dirs </ name >
      < value > file :/// mnt / disk1 / data ,...</ value >
    </ property >
    < property >
      < name > fs . oss . impl . disable . cache </ name >
      < value > true </ value >
    </ property >
  </ configurat  ion >
```

The `fs . oss . buffer . dirs` can be set to multiple paths.

- Use RDS to save Druid metadata

  Use the MySQL database on the header-1 node to save Druid metadata. You can also use Alibaba Cloud RDS to save the metadata.

  The following uses RDS MySQL as an example to demonstrate the configuration. Before you configure it, make sure that:

  - The RDS MySQL instance has been created.
  - A separate account has been created for Druid to access RDS MySQL (the root account is not recommended). This example uses account name druid and password druidpw.
  - Create a separate MySQL database for Druid metadata. Suppose the database is called druiddb.
  - Make sure that account druid has permission to access druiddb.

  In the E-MapReduce console, click Manage next to the Druid cluster you want to configure. Click the Druid service, and then select the Configuration tab to find the `common . runtime` configuration file. Click Custom Configuration to add the following three configuration items:

  - druid.metadata.storage.connector.connectURI, where the value is: jdbc:mysql://rm-xxxxx.mysql.rds.aliyuncs.com:3306/druiddb.
  - druid.metadata.storage.connector.user, where the value is druid.
  - druid.metadata.storage.connector.password, where the value is druidpw.

  Click the Save, Configuration to Host, and then Restart Related Services in the upper-right corner to make the configuration take effect.

  Log on to the RDS console to view the tables created by druiddb. You will find tables automatically created by druid.

- Service memory configuration

  The memory of the Druid service consists of heap memory (configured through jvm. config) and direct memory (configured through jvm. config and runtime. properteis). E-MapReduce automatically generates a set of configurations when

you create a cluster. However, in some cases, you may still need to configure the memory.

To adjust the service memory configuration, you can access the cluster services through the E-MapReduce console and perform related operations on the page.

 Note:

For direct memory, make sure that:

```
- XX : MaxDirectM  emorySize   is   greater   than   or   equal
   to   druid . processing . buffer . sizeBytes  * ( druid .
 processing . numMergeBu  ffers  +  druid . processing . numThreads
  +  1 ).
```

Batch index

- Interaction with Hadoop clusters

  If you select HDFS and YARN (with their own Hadoop clusters) when creating your Druid clusters, the system automatically configures the interaction between HDFS and YARN. The following example shows how to configure the interaction between a standalone Druid cluster and a standalone Hadoop cluster. It is assumed that the Druid cluster ID is 1234 and the Hadoop cluster ID is 5678. If your clusters do not work as expected, this may be because of a slightly inaccurate operation.

  For the interaction with standard-mode Hadoop clusters, complete the following operations:

  1. Ensure the communication between the two clusters. (Each cluster is associated with a different security group, and access rules are configured for these security groups.)
  2. Put core-site.xml, hdfs-site.xml, yarn-site.xml, mapred-site.xml of `/ etc / ecm / hadoop - conf` of the Hadoop cluster in the `/ etc / ecm / duird - conf / druid / _common` directory on each node of the Druid cluster. (If you select built-in Hadoop when creating the cluster, several soft links in this directory will map to the configuration of the Hadoop service of E-MapReduce. Remove these soft links first.)
  3. Write the hosts of the Hadoop cluster to the hosts list on the Druid cluster. Note that the hostname of the Hadoop cluster should be a long name, such as emr-

header-1.cluster-xxxxxxxx. We recommend that you put the Hadoop hosts after the hosts of the Druid cluster, such as:

```
...
10 . 157 . 201 . 36     emr - as . cn - hangzhou . aliyuncs .
com
10 . 157 . 64 . 5       eas . cn - hangzhou . emr . aliyuncs .
com
192 . 168 . 142 . 255   emr - worker - 1 . cluster - 1234   emr -
worker - 1   emr - header - 2 . cluster - 1234   emr - header - 2
  iZbp1h9g7b  oqo9x23qbi  fiZ
192 . 168 . 143 . 0     emr - worker - 2 . cluster - 1234   emr -
worker - 2   emr - header - 3 . cluster - 1234   emr - header - 3
  iZbp1eaa58  19tkjx55yr  9xZ
192 . 168 . 142 . 254   emr - header - 1 . cluster - 1234   emr -
header - 1   iZbp1e3zwu  vnmakmsjer  2uZ
For   Hadoop   clusters   in   high - security   mode ,  perform
  the   following   operations :
192 . 168 . 143 . 6     emr - worker - 1 . cluster - 5678   emr -
worker - 1   emr - header - 2 . cluster - 5678   emr - header - 2
  iZbp195rj7  zvx8qar4f6  b0Z
192 . 168 . 143 . 7   emr - worker - 2 . cluster - 5678   emr -
worker - 2   emr - header - 3 . cluster - 5678   emr - header - 3
  iZbp15vy2r  sxoegki4qh  dpZ
192 . 168 . 143 . 5   emr - header - 1 . cluster - 5678   emr -
header - 1   iZbp10tx4e  gw3wfnh5oi  i1Z
```

For Hadoop clusters in high-security mode, complete the following operations:

1. Ensure the communication between the two clusters. (Each cluster is associated with a different security group, and access rules are configured for these security groups.)
2. Put core-site.xml, hdfs-site.xml, yarn-site.xml, mapred-site.xml of `/ etc / ecm / hadoop - conf` of the Hadoop cluster in the `/ etc / ecm / duird - conf / druid / _common` directory on each node of the Druid cluster. (If you select built-in Hadoop when creating a cluster, several soft links in this directory will point to the configuration with Hadoop. Remove these soft links first.) Modify `hadoop . security . authentica  tion . use . has` in core-site.xml to `false` . (This configuration is completed on the client to enable AccessKey authentication for users. If Kerberos authentication is used, disable AccessKey authentication.)
3. Write the hosts of the Hadoop cluster to the hosts list of each node on the Druid cluster. Note that the hostname of the Hadoop cluster should be a long name

, such as emr-header-1.cluster-xxxxxxxx. We recommend that you put the Hadoop hosts after the hosts of the Druid cluster.

4. Set Kerberos cross-domain mutual trust between the two clusters. For more details, see *#unique_30*.
5. Create a local Druid account (useradd-m-g hadoop) on all nodes of the Hadoop cluster, or set druid.auth.authenticator.kerberos.authtomate to create a mapping rule for the Kerberos account to the local account. For specific pre-release rules, see *here*. This method is recommended because it is easy to operate without errors.

   

   Note:
   In high-security Hadoop clusters, all Hadoop commands must be run from a local account. By default, this local account needs to have the same name as the principal. YARN also supports mapping a principal to a local account.
6. Restart the Druid service.

- Use Hadoop to index batch data

  Druid has an example named wikiticker located in *${DRUID_HOME}/quickstart*. By default, *${DRUID_HOME}* is /usr/lib/ druid-current. Each line of the wikiticker document (wikiticker-2015-09-12-sampled.json.gz) is a record. Each record is a json object. The format is as follows:

```
``` json
{
    " Time ": " 2015 - 09 - 12T00 :  46 :  58 . 771Z  ",
    " channel ": "# en . wikipedia ",
    " cityName ":  null ,
    " comment ": " added  project ",
    " countryIso  Code ":  null ,
    " countryNam  e ":  null ,
    " isAnonymou  s ":  false ,
    " isMinor ":  false ,
    " isNew ":  false ,
    " isRobot ":  false ,
    " isUnpatrol  led ":  false ,
    " metroCode ":  null ,
    " namespace ": " Talk ",
    " page ": " Talk : Oswald  Tilghman ",
    " regionIsoC  ode ":  null ,
    " regionName ":  null ,
    " user ": " GELongstre  et ",
    " delta ":  36 ,
    " added ":  36 ,
    " deleted ":  0
}
```
```

```
```

To use Hadoop to index batch data, complete the following steps:

1. Decompress the compressed file and place it in a directory of HDFS (such as: *hdfs :// emr - header - 1 . cluster - 5678 : 9000 / druid*). Run the following command on the Hadoop cluster:

```
### If you are operating on a standalone Hadoop
cluster , copy a druid . keytab to Hadoop cluster
after the mutual trust is establishe d between the
  two clusters , and run the kinit command .
 kinit - kt / etc / ecm / druid - conf / druid . keytab
druid
###
 hdfs dfs - mkdir hdfs :// emr - header - 1 . cluster - 5678
: 9000 / druid
 hdfs dfs - put ${ DRUID_HOME }/ quickstart / wikiticker -
2015 - 09 - 16 - sampled . json hdfs :// emr - header - 1 .
cluster - 5678 : 9000 / druid
```

Note:

- Modify `hadoop . security . authentica tion . use . has` in */ etc / ecm / hadoop - conf / core - site . xml* to `false` before running the HDFS command for a high-security cluster.
- Make sure that you have created a Linux account named Druid on each node of the Hadoop cluster.

2. Modify Druid cluster *${DRUID_HOME}/quickstart/wikiticker-index.json*, as shown below:

```
{
     " type " : " index_hado op ",
     " spec " : {
         " ioConfig " : {
             " type " : " hadoop ",
             " inputSpec " : {
                 " type " : " static ",
                 " paths " : " hdfs :// emr - header - 1 . cluster
 - 5678 : 9000 / druid / wikiticker - 2015 - 09 - 16 - sampled .
 json "
             }
         },
         " dataSchema " : {
             " dataSource " : " wikiticker ",
             " granularit ySpec " : {
                 " type " : " uniform ",
                 " segmentGra nularity " : " day ",
                 " queryGranu larity " : " none ",
                 " intervals " : [" 2015 - 09 - 12 / 2015 - 09 -
 13 "]
             },
             " parser " : {
```

```
                " type " : " hadoopyStr  ing ",
                " parseSpec " : {
                  " format " : " json ",
                  " dimensions  Spec " : {
                    " dimensions " : [
                      " channel ",
                      " cityName ",
                      " comment ",
                      " countryIso  Code ",
                      " countryNam  e ",
                      " isAnonymou  s ",
                      " isMinor ",
                      " isNew ",
                      " isRobot ",
                      " isUnpatrol  led ",
                      " metroCode ",
                      " namespace ",
                      " page ",
                      " regionIsoC  ode ",
                      " regionName ",
                      " user "
                    ]
                  },
                  " timestampS  pec " : {
                    " format " : " auto ",
                    " column " : " time "
                  }
                }
              },
              " metricsSpe  c " : [
                {
                  " name " : " count ",
                  " type " : " count "
                },
                {
                  " name " : " added ",
                  " type " : " longSum ",
                  " fieldName " : " added "
                },
                {
                  " name " : " deleted ",
                  " type " : " longSum ",
                  " fieldName " : " deleted "
                },
                {
                  " name " : " delta ",
                  " type " : " longSum ",
                  " fieldName " : " delta "
                },
                {
                  " name " : " user_uniqu  e ",
                  " type " : " hyperUniqu  e ",
                  " fieldName " : " user "
                }
              ]
            },
            " tuningConf  ig " : {
              " type " : " hadoop ",
              " partitions  Spec " : {
                " type " : " hashed ",
                " targetPart  itionSize " :  5000000
              },
              " jobPropert  ies " : {
                " mapreduce . job . classloade  r ": " true "
```

```
            }
        }
    },
    " hadoopDepe  ndencyCoor  dinates ": [" org . apache . hadoop
: hadoop - client : 2 . 7 . 2 "]
}
```

Note:

- `spec . ioConfig . type` is set to `hadoop .`
- `spec . ioConfig . inputSpec . paths` is the path of the input file.
- `tuningConf  ig . type` is `hadoop .`
- `tuningConf  ig . jobPropert  ies` sets the classloader of the MapReduce job.
- `hadoopDepe  ndencyCoor  dinates` develops the version of Hadoop client.

3. Run the batch index command on the Druid cluster.

```
cd  ${ DRUID_HOME }
 curl  -- negotiate  - u : druid  - b  ~/ cookies  - c  ~/
cookies  - XPOST  - H  ' Content - Type : applicatio  n / json
' - d  @ quickstart / wikiticker - index . json  http :// emr -
header - 1 . cluster - 1234 : 18090 / druid / indexer / v1 / task
```

Note that items such as `- - negotiate` , `- u` , `- b` , and `- c` are for high-security mode Druid clusters. The Overlord port number is 18090 by default.

4. View the running state of the jobs.

Enter http://emr-header-1.cluster-1234:18090/console.html into your browser to view how the jobs run. To access the page properly, you need to open an SSH tunnel in advance (see *#unique_7*), and start a Chrome agent. If the high-security mode is enabled for the Druid cluster, you have to configure your browser to support the Kerberos authentication process. For more information, see *here*.

5. Query the data based on Druid syntax.

Druid has its own query syntax. You need to prepare a json-formatted query file that describes how you want to query. A topN query to the wikiticker data is as follows *${DRUID_HOME}/quickstart/wikiticker-top-pages.json*:

```
{
    " queryType " : " topN ",
    " dataSource " : " wikiticker ",
    " intervals " : [" 2015 - 09 - 12 / 2015 - 09 - 13 "],
    " granularit  y " : " all ",
```

```
    " dimension " : " page ",
    " metric " : " edits ",
    " threshold " :  25 ,
    " aggregatio  ns " : [
        {
            " type " : " longSum ",
            " name " : " edits ",
            " fieldName " : " count "
        }
    ]
}
```

You can check the results of the query by running the following command:

```
cd  ${ DRUID_HOME }
 curl  -- negotiate  - u : druid  - b  ~/ cookies  - c  ~/
cookies  - XPOST  - H  ' Content - Type : applicatio  n / json '
- d  @ quickstart / wikiticker - top - pages . json  ' http ://
emr - header - 1 . cluster - 1234 : 18082 / druid / v2 /? pretty
'
```

Note that items such as`- - negotiate` , `- u` , `- b` , and`- c` are for Druid clusters in high-security mode. You can check the results of a specific query.

- Real-time index

  We recommend that you use *Tranquility client* to send real-time data to Druid. Tranquility supports sending data to Druid in a variety of ways, such as Kafka, Flink, Storm, and Spark Streaming. For more information about the Kafka method, see *#unique_31*. Druid also follows the Kafka section in Tranquility. For more information about how to use Tranquility and SDKs, see *Tranquility Help Document*.

  For Kafka, you can also use the kafka-indexing-service extension. For details, see *#unique_32*.

- Troubleshoot index failures

  When the index fails, complete the following steps to troubleshoot the failure:

  - For the index of batch data

    1. If the curl command output displays an error or does not display any information, check the file format. Alternatively, add the `-v` parameter to the curl command to check the value returned from the RESTful API.
    2. Observe the execution of jobs on the Overlord page. If the execution fails, view the logs on that page.
    3. In many cases, logs are not generated. In the case of a Hadoop job, open the YARN page to check whether an index job has been generated.
    4. If no errors are found, you need to log on to the Druid cluster, and view the execution logs of Overlord at `/mnt/disk1/log/druid/overlord-emr-header-1.cluster-xxxx.log`. In the case of an HA cluster, check the Overlord that you submitted the job to.
    5. If the job has been submitted to Middlemanager, but a failure is returned from Middlemanager, you need to view the worker that the job is submitted to in Overlord, and log on to the worker to view the Middlemanager logs (at `/mnt/disk1/log/druid/middleManager-emr-header-1.cluster-xxxx.log`).

  - For real-time index of Tranquility

    View the Tranquility log to check whether the message was received or dropped.

    The remaining troubleshooting steps are the same as steps 2 to 5 for batch indexing.

    Most errors are about cluster configurations and jobs. Cluster configuration errors are about memory parameters, cross-cluster connection, access to clusters in high-security mode, and principals. Job errors are about the format of the job description files, input data parsing, and other job-related configuration issues (such as ioConfig).

6.3 Ingestion Spec

This section briefly introduces Ingestion Spec, the description file of the index data.

Ingestion Spec is a unified description of the format of the data being indexed and how it is indexed by Druid. It is a JSON file, which consists of three parts:

```
{
    " dataSchema " : {...},
    " ioConfig " : {...},
    " tuningConf  ig " : {...}
}
```

Key	Format	Description	Required
dataSchema	JSON object	Describes the schema information of the data you want to consume. dataSchema is fixed and does not change with the way in which data is consumed.	Yes
ioConfig	JSON object	Describes the source and destination of the data you want to consume. If the consumption method of the data is different, ioConfig is also different.	Yes
tuningConfig	JSON object	Configures the parameters of the data you want to consume. If the consumption method of the data is different, the adjustable parameters are also different.	No

dataSchema

dataSchema describes the format of the data and how to parse the data. The typical structure is as follows:

```
{
    " dataSoruce ": < name_of_da  taSource >,
    " parser ": {
        " type ": <>,
        " parseSpec ": {
            " format ": <>,
            " timestampS  pec ": {},
            " dimensions  Spec ": {}
        }
    },
    " metricsSpe  c ": {},
    " granularit  ySpec ": {}
```

```
}
```

Key	Format	Description	Required
dataSource	String	Name of the data source.	Yes
parser	JSON object	How the data is parsed.	Yes
metricsSpec	Array of JSON objects	Aggregator list.	Yes
granularitySpec	JSON object	Data aggregation settings, such as creating segments and aggregation granularity.	Yes

- parser

  parser determines how your data is parsed correctly. metricsSpec defines how the data is clustered for calculation. granularitySpec defines the granularity of the data fragmentation and the granularity of the query.

  There are two types of parser: string and hadoopstring. The latter is used for Hadoop index jobs. ParseSpec is a specific definition of data format resolution.

Key	Format	Description	Required
type	String	The data format can be json, jsonLowercase, csv, or tsv.	Yes
timestampSpec	JSON object	Timestamp and timestamp type.	Yes
dimensionsSpec	JSON object	The dimension of the data (columns are included).	Yes

  For different data formats, additional parseSpec options may exist. The following table describes timestampSpec and dimensionsSpec.

Key	Format	Description	Required
column	String	Columns corresponding to the timestamp.	Yes

Key	Format	Description	Required
format	String	The timestamp type can be ISO, millis, POSIX, auto, or whatever is supported by *joda time*.	Yes

Key	Format	Description	Required
dimensions	JSON array	Describes which dimensions the data contains. Each dimension can be just a string. You can also specify the attribute for the dimension. For example, the type of dimensions: [dimenssion1, dimenssion2, {type: long, name: dimenssion3}] is string by default.	Yes
dimensionExclusions	Array of JSON strings	Dimension to be deleted when data is consumed.	No
spatialDimensions	Array of JSON objects	Spatial dimension.	No

- metricsSpec

  MetricsSpec is an array of JSON objects. It defines several aggregators. Aggregators typically have the following structures:

```
``` json
{
    " type ": < type >,
    " name ": < output_nam  e >,
    " fieldName ": < metric_nam  e >
}
```
```

  The following commonly used aggregators are provided:

Type	Type optional
count	count
sum	longSum, doubleSum, floatSum
min/max	longMin/longMax, doubleMin/doubleMax, floatMin/floatMax
first/last	longFirst/longLast, doubleFirst/doubleLast, floatFirst/floatLast

Type	Type optional
javascript	javascript
cardinality	cardinality
hyperUnique	hyperUnique

Note:
The last three types in the table are advanced aggregators. For information about how to use them, see *Druid official documents*.

- granularitySpec

  Two aggregation modes are supported: uniform and arbitrary. The uniform mode aggregates data with a fixed interval of time. The arbitrary mode tries to make sure that each of the segments has the same size, but the time interval for aggregation is not fixed. Uniform is the current default option.

Key	Format	Description	Required
segmentGranularity	String	Segment granularity Uniform type.The default is DAY.	No.
queryGranularity	String	Minimum data aggregation granularity for query. The default is true.	No
rollup	Bool value	Aggregate or not.	No.
intervals	String	Time interval of data consumption.	It is Yes for batch and No for realtime.

ioConfig

ioConfig describes the data source. An example of Hadoop index is as follows:

```
{
    " type ": " hadoop ",
    " inputSpec ": {
        " type ": " static ",
        " paths ": " hdfs :// emr - header - 1 . cluster - 6789 : 9000
 / druid / quickstart / wikiticker - 2015 - 09 - 16 - sampled . json "
    }
}
```

Note:
This part is not required for streaming data that is processed through Tranquility.

TuningConfig

TuningConfig refers to additional settings. For example, you can specify MapReduce parameters to use Hadoop to create an index for batch data. The contents of tuningConfig may vary based on the data source. For more information, see the example file or official document of this service.

6.4 Tranquility

This section uses Kafka as an example and describes how to use Tranquility in E-MapReduce to capture data from the Kafka cluster and push it to the Druid cluster in real time.

Tranquility is an application that sends data to Druid in real-time in push mode. It solves many issues, such as multiple partitions, multiple copies, service discovery, and data loss. It simplifies the use of Druid and supports a wide range of data sources , including Samza, Spark, Storm, Kafka, and Fink.

Interaction with the Kafka cluster

The first interaction is between the Druid cluster and the Kafka cluster. The interaction configuration of the two clusters is similar to that of the Hadoop cluster. You have to set the connectivity and hosts. For standard mode Kafka clusters, complete the following steps:

1. Ensure the communication between clusters. (The two clusters are either in the same security group, or each cluster is associated with a different security group and access rules are configured for these security groups.)
2. Write the hosts of the Kafka cluster to the hosts list of each node on the Druid cluster. Note that the hostname of the Kafka cluster should be a long name, such as emr-header-1.cluster-xxxxxxxx.

For high-security mode Kafka clusters, complete the following operations (the first two steps are the same as those for standard mode clusters):

1. Ensure the communication between the two clusters (The two clusters are in the same security group, or each cluster is associated with a different security group and access rules are configured for these security groups).
2. Write the hosts of the Kafka cluster to the hosts list of each node on the Druid cluster. Note that the hostname of the Kafka cluster should be a long name, such as emr-header-1.cluster-xxxxxxxx.

3. Set Kerberos cross-domain mutual trust between the two clusters. For details, see #unique_30. Bidirectional mutual trust is recommended.
4. Prepare a client security configuration file:

```
KafkaClien  t  {
      com . sun . security . auth . module . Krb5LoginM  odule
required
      useKeyTab = true
      storeKey = true
      keyTab ="/ etc / ecm / druid - conf / druid . keytab "
      principal =" druid @ EMR .  1234 .  COM ";
 };
```

   Synchronize the configuration file to all nodes in the Druid cluster and place it in a specific directory, such as */ tmp / kafka / kafka_clie nt_jaas . conf .*
5. In overlord.jvm of the Druid configuration page:

```
Add   Djava . security . auth . login . config =/ tmp / kafka /
kafka_clie  nt_jaas . conf
```

6. Configure the following option in middleManager.runtime on the Druid configuration page: `druid . indexer . runner . javaOpts =- Djava . security . auth . login . confi =/ tmp / kafka / kafka_clie  nt_jaas . conf` and other jvm startup parameters.
7. Restart the Druid service.

Use Tranquility Kafka

Because Tranquility is a service, it is a consumer for Kafka and a client for Druid. You can use a neutral machine to run Tranquility, as long as this machine is able to connect to the Kafka and the Druid clusters simultaneously.

1. Create a topic named pageViews on the Kafka side.

```
-- If   the   Kafka   high - security   mode   is   enabled :
  export   KAFKA_OPTS ="- Djava . security . auth . login . config
 =/ etc / ecm / kafka - conf / kafka_clie  nt_jaas . conf "
 --
 ./ bin / kafka - topics . sh  -- create  -- zookeeper   emr -
 header - 1 : 2181 , emr - header - 2 : 2181 , emr - header - 3 :
 2181 / kafka - 1 . 0 . 1  -- partitions   1  -- replicatio  n -
 factor   1  -- topic   pageViews
```

2. Download the Tranquility installation package and decompress it to a path.

3. Configure the dataSource.

   It is assumed that your topic name is pageViews, and each topic is a JSON file.

```
{" time ": " 2018 - 05 - 23T11 : 59 : 43Z ", " url ": "/ foo / bar
 ", " user ": " alice ", " latencyMs ":  32 }
 {" time ": " 2018 - 05 - 23T11 : 59 : 44Z ", " url ": "/", " user
 ": " bob ", " latencyMs ":  11 }
 {" time ": " 2018 - 05 - 23T11 : 59 : 45Z ", " url ": "/ foo / bar
 ", " user ": " bob ", " latencyMs ":  45 }
```

   The configuration of the corresponding dataSource is as follows:

```
{
   " dataSource  s " : {
     " pageViews - kafka " : {
       " spec " : {
         " dataSchema " : {
           " dataSource " : " pageViews - kafka ",
           " parser " : {
             " type " : " string ",
             " parseSpec " : {
               " timestampS  pec " : {
                 " column " : " time ",
                 " format " : " auto "
               },
               " dimensions  Spec " : {
                 " dimensions " : [" url ", " user "],
                 " dimensionE  xclusions " : [
                   " timestamp ",
                   " value "
                 ]
               },
               " format " : " json "
             }
           },
           " granularit  ySpec " : {
             " type " : " uniform ",
             " segmentGra  nularity " : " hour ",
             " queryGranu  larity " : " none "
           },
           " metricsSpe  c " : [
             {" name ": " views ", " type ": " count "},
             {" name ": " latencyMs ", " type ": " doubleSum ", "
 fieldName ": " latencyMs "}
           ]
         },
         " ioConfig " : {
           " type " : " realtime "
         },
         " tuningConf  ig " : {
           " type " : " realtime ",
           " maxRowsInM  emory " : " 100000 ",
           " intermedia  tePersistP  eriod " : " PT10M ",
           " windowPeri  od " : " PT10M "
         }
       },
       " properties " : {
         " task . partitions " : " 1 ",
         " task . replicants " : " 1 ",
         " topicPatte  rn " : " pageViews "
       }
```

```
    }
  },
  " properties " : {
    " zookeeper . connect " : " localhost ",
    " druid . discovery . curator . path " : "/ druid / discovery
",
    " druid . selectors . indexing . serviceNam  e " : " druid /
overlord ",
    " commit . periodMill  is " : " 15000 ",
    " consumer . numThreads " : " 2 ",
    " kafka . zookeeper . connect " : " emr - header - 1 . cluster
- 500148518 : 2181 , emr - header - 2 . cluster - 500148518 : 2181
,    emr - header - 3 . cluster - 500148518 : 2181 / kafka - 1 . 0
. 1 ",
    " kafka . group . id " : " tranquilit  y - kafka ",
  }
}
```

4. Run the following command to start Tranquility.

```
./ bin / tranquilit  y   kafka  - configFile
```

5. Start the producer and configure it to send data.

```
./ bin / kafka - console - producer . sh  -- broker - list   emr -
worker - 1 : 9092 , emr - worker - 2 : 9092 , emr - worker - 3 :
9092  -- topic   pageViews
```

Enter the following codes:

```
{" time ": " 2018 - 05 - 24T09 : 26 : 12Z ", " url ": "/ foo / bar
 ", " user ": " alice ", " latencyMs ":  32 }
 {" time ": " 2018 - 05 - 24T09 : 26 : 13Z ", " url ": "/", " user
 ": " bob ", " latencyMs ":  11 }
 {" time ": " 2018 - 05 - 24T09 : 26 : 14Z ", " url ": "/ foo / bar
 ", " user ": " bob ", " latencyMs ":  45 }
```

You can now view specific information in the Tranquility log. The corresponding real-time indexing task has also been started on the Druid side.

6.5 Kafka Indexing Service

This section describes how to use Druid Kafka Indexing Service in E-MapReduce to ingest Kafka data in real time.

The Kafka Indexing Service is an extension launched by Druid to ingest Kafka data in real time using Druid's indexing service. The extension enables supervisors in Overlord which start some indexing tasks in Middlemanager. These tasks connect to the Kafka cluster to ingest the topic data and complete the index creation. You need to prepare a data ingestion format file and manually start the supervisor through the RESTful API.

Interaction with the Kafka cluster

See the introduction in *Tranquility*.

Use Druid's Kafka Indexing Service to ingest Kafka data in real time

1. Run the following command on the Kafka cluster (or gateway) to create a topic named metrics.

```
-- If   the   Kafka   high - security   mode   is   enabled :
  export   KAFKA_OPTS ="- Djava . security . auth . login . config
 =/ etc / ecm / kafka - conf / kafka_clie  nt_jaas . conf "
 --
  kafka - topics . sh  -- create  -- zookeeper   emr - header - 1 :
 2181 , emr - header - 2 , emr - header - 3 / kafka - 1 . 0 . 0  --
 partitions   1  -- replicatio  n - factor   1  -- topic   metrics
```

   You can adjust the parameters based on your needs. The `/kafka-1.0.0` section of the `- - zookeeper` parameter is path, and you can see the value of the zookeeper.connect on the Kafka service Configuration page of the Kafka cluster. If you build your own Kafka cluster, the parmname – `zookeeper` parameter can be changed according to your actual configuration.

2. Define the data format description file for the data source. Name it metrics-kafka.json and place it in the current directory (or another directory that you have specified).

```
{
     " type ": " kafka ",
     " dataSchema ": {
         " dataSource ": " metrics - kafka ",
         " parser ": {
             " type ": " string ",
             " parseSpec ": {
                 " timestampS  pec ": {
                     " column ": " time ",
                     " format ": " auto "
                 },
                 " dimensions  Spec ": {
                     " dimensions ": [" url ", " user "]
                 },
                 " format ": " json "
             }
         },
         " granularit  ySpec ": {
             " type ": " uniform ",
             " segmentGra  nularity ": " hour ",
             " queryGranu  larity ": " none "
         },
         " metricsSpe  c ": [{
                 " type ": " count ",
                 " name ": " views "
             },
             {
                 " name ": " latencyMs ",
```

```
                " type ": " doubleSum ",
                " fieldName ": " latencyMs "
            }
        ]
    },
    " ioConfig ": {
        " topic ": " metrics ",
        " consumerPr  operties ": {
            " bootstrap . servers ": " emr - worker - 1 . cluster -
xxxxxxxx : 9092  ( the   bootstrap . servers   of   your   Kafka
clusters )",
            " group . id ": " kafka - indexing - service ",
            " security . protocol ": " SASL_PLAIN  TEXT ",
            " sasl . mechanism ": " GSSAPI "
        },
        " taskCount ":  1 ,
         replicas :  1
        " taskDurati  on ": " PT1H "
    },
    " tuningConf  ig ": {
        " type ": " Kafka ",
        " maxRowsInM  emory ": " 100000 "
    }
}
```

Note:

`ioConfig . consumerPr  operties . security . protocol` and `ioConfig . consumerPr  operties . sasl . mechanism` are security-related options and are not required for standard mode Kafka clusters.

3. Run the following command to add a Kafka supervisor.

```
curl  -- negotiate  - u : druid  - b  ~/ cookies  - c  ~/ cookies
 - XPOST  - H  ' Content - Type :  applicatio  n / json ' - d  @
metrics - kafka . json   http :// emr - header - 1 . cluster - 1234
: 18090 / druid / indexer / v1 / supervisor
```

The `– negotiate` , `- u` , `- b` , and `- c` options are for high-security mode Druid clusters.

4. Enable a console producer on the Kafka cluster.

```
-- If   the   high - security   mode   of   Kafka   is   enabled :
 export   KAFKA_OPTS ="- Djava . security . auth . login . config
=/ etc / ecm / kafka - conf / kafka_clie  nt_jaas . conf "
 echo  - e  " security . protocol = SASL_PLAIN  TEXT \ nsasl .
mechanism = GSSAPI " > / tmp / Kafka / producer . conf
--
 Kafka - console - producer . sh  -- producer . config  / tmp /
kafka / producer . conf  -- broker - list   emr - worker - 1 : 9092
,  emr - worker - 2 : 9092 ,  emr - worker - 3 : 9092  -- topic
metrics
```

```
 >
```

The `– producer . config  / tmp / Kafka / producer . conf` option is for high-security mode Kafka clusters.

5. Enter data at the command prompt of kafka_console_producer.

```
{" time ": " 2018 - 03 - 06T09 : 57 : 58Z ", " url ": "/ foo / bar
 ", " user ": " alice ", " latencyMs ":  32 }
 {" time ": " 2018 - 03 - 06T09 : 57 : 59Z ", " url ": "/", " user
 ": " bob ", " latencyMs ":  11 }
 {" time ": " 2018 - 03 - 06T09 : 58 : 00Z ", " url ": "/ foo / bar
 ", " user ": " bob ", " latencyMs ":  45 }
```

The timestamp can be generated with the following Python command:

```
python  - c  ' import   datetime ;  print ( datetime . datetime .
utcnow (). strftime ("% Y -% m -% dT % H :% M :% SZ "))'
```

6. Prepare a query file named metrics-search.json.

```
{
     " queryType " : " search ",
     " dataSource " : " metrics - kafka ",
     " intervals " : [" 2018 - 03 - 02T00 : 00 : 00 . 000 / 2018 -
 03 - 08T00 : 00 : 00 . 000 "],
     " granularit  y " : " all ",
     " searchDime  nsions ": [
         " url ",
         " user "
     ],
     " query ": {
         " type ": " insensitiv  e_contains ",
         " value ": " bob "
     }
 }
```

7. Execute the query on the master node of the Druid cluster.

```
curl  -- negotiate  - u : Druid  - b  ~/ cookies  - c  ~/ cookies
 - XPOST  - H  ' Content - Type :  applicatio  n / json ' - d  @
metrics - search . json   http :// emr - header - 1 . cluster -
1234 : 8082 / druid / v2 /? pretty
```

The `– negotiate` , `- u` , `- b` , and `- c` options are for high-security mode Druid clusters.

8. You will see a query result similar to the following:

```
[ {
   " timestamp " : " 2018 - 03 - 06T09 : 00 : 00 . 000Z ",
   " result ": {
     " dimension " : " user ",
     " value " : " bob ",
     " count ":  2 ,
   } ]
```

```
} ]
```

6.6 Superset

The Druid cluster integrates the Superset tool, which is integrated with Druid and supports a variety of relational databases. Because Druid supports SQL, you can access Druid through Superset in two ways: Druid's native query language or SQL.

Superset is installed in emr-header-1 by default and does not support high availability at present. Before you use this tool, make sure that your host can access emr-header-1. You can connect to the host by establishing the *SSH tunnel*.

1. Log on to the Superset

   Enter http://emr-header-1:18088 in your browser to go to the Superset logon page. The default username is admin and the default password is admin. When you log on for the first time, we strongly recommend changing your password.

2. Add a Druid cluster

   The English interface is displayed by default. You can select the appropriate language by clicking the flag icon in the upper-right corner. In the menu bar along the top, select Data Source > Druid Cluster to add a Druid cluster.

   Configure the addresses of the coordinator and broker. The default port number in E-MapReduce is the corresponding open source port number with "1" added in front. For example, if the open-source broker port number is 8082, the port number in E-MapReduce is 18082.

3. Refresh or add a new data source

   After adding the Druid cluster, you can click Data Source > Scan to add new data sources. The data sources on the Druid cluster loaded automatically.

   You can also customize a new data source by clicking Sources > Druid Datasources on the interface. (This operation is equivalent to writing a JSON file for data source ingestion.)

   Enter the necessary information for custom data sources, and save it.

   Click the second of the three small icons on the left side to edit the data source. Enter the appropriate information, such as dimensions and metrics.

4. Query Druid

   After the data source has been added successfully, click it to go to the details page.

5. (Optional) Use Druid as a database

   Superset provides SQLAlchemy to support a wide variety of databases with various dialects, as shown in the following figure.

   Superset also supports accessing Druid in this way. The corresponding SQLAlchemy URI of Druid is druid://emr-header-1:18082/druid/v2/sql. When you add Druid as a database, check the "Expose in SQL Lab" check box.

   You can now use SQL to query in the SQL toolkit.

6.7 Common Druid problems

This section describes some of the common problems you may encounter with Druid.

Analyze the indexing failure

If indexing fails, complete the following steps to troubleshoot the failure:

- For batch data indexing

  1. If the curl command output displays an errort or does not display any information, check the file format. Alternatively, add the `-v` parameter to the curl command to check the value returned from the RESTful API.
  2. Observe the execution of jobs on the Overlord page. If the execution fails, view the logs on that page.
  3. In many cases, logs are not generated. In the case of a Hadoop job, open the YARN page to check whether an index job has been generated.
  4. If no errors are found, you need to log on to the Druid cluster and view the execution logs of Overlord at `/mnt/disk1/log/druid/overlord-emr-header-1.cluster-xxxx.log`. In the case of an HA cluster, check the Overlord that you submitted the job to.
  5. If the job has been submitted to Middlemanager but a failure is returned from Middlemanager, you need to view the worker that the job is submitted to in Overlord, and log on to the worker node to view the Middlemanager logs (at `/mnt/disk1/log/druid/middleManager-emr-header-1.cluster-xxxx.log`).

- For real-time Tranquility indexing

  Check the Tranquility log to see if the message was received or dropped.

  The remaining troubleshooting steps are the same as steps 2 to 5 of batch indexing.

  Most errors are about cluster configurations and jobs. Cluster configuration errors are about memory parameters, cross-cluster connection, access to clusters in high-security mode, and principals. Job errors are about the format of the job description files, input data parsing, and other job-related configuration issues (such as ioConfig).

Obtain the FAQ list

- Service startup fails.

  Most of these problems are due to configuration problems with the running parameters of the JVM component. For example, the machine may not have a large

memory, but it is configured with a larger JVM memory or a larger number of threads.

To resolve this issue, view the component logs and adjust the relevant parameters. JVM memory involves heap memory and direct memory. For more information, go to *Druid Performance FAQ*.

- The YARN task fails during indexing and shows a JAR package conflict error like this: `Error : class com . fasterxml . jackson . datatype . guava . deser . HostAndPor tDeseriali zer overrides final method deserializ e .( Lcom / fasterxml / jackson / core / JsonParser ; Lcom / fasterxml / jackson / databind / Deserializ ationConte xt ;) Ljava / lang / Object ;.`

  To resolve this issue, add the following content to the indexing job configuration file:

```
" tuningConf ig " : {
     ...
     " jobPropert ies " : {
         " mapreduce . job . classloade r ": " true "
          or
         " mapreduce . job . user . classpath . first ": " true "
     }
     ...
 }
```

  The parameter `mapreduce . job . classloade r` allows MapReduce jobs to use standalone classloaders, and the parameter `mapreduce . job . user . classpath . first` gives MapReduce the priority to use your JAR packages. You can select one of these two configuration items. For more information, go to *Druid documents*.

- The logs of the index task report that the reduce task cannot create the segments directory.

  To resolve this issue, complete the following:

  - Check the settings for deep storage, including type and directory. If the type is local, pay attention to the permission settings of the directory. If the type is HDFS, the directory should be written as the full HDFS path, such as hdfs://:9000/. For hdfs_master, IP is recommended. If you want to use a domain name, use

the full domain name, such as emr-header-1.cluster-xxxxxxxx rather than emr-header-1.

- If you are using Hadoop for batch indexing, you must set the deep storage of segments as "hdfs". The local type may cause the MapReduce job to be in an unidentified state, because the remote YARN cluster cannot create the segments directory in the reduce task. (This is only applicable to standalone Druid clusters.)

· Failed to create directory within 10,000 attempts.

This issue occurs typically because the path set by java.io.tmp in the JVM configuration file does not exist. Set the path and make sure that the Druid account has permission to access it.

· com.twitter.finagle.NoBrokersAvailableException: No hosts are available for disco!firehose:druid:overlord

This issue is typically due to ZooKeeper connection issues. Make sure that Druid and Tranquility have the same connection string for ZooKeeper. Because the default ZooKeeper path for Druid is /druid, make sure that zookeeper.connect in the Tranquility settings includes /druid. (Two ZooKeeper settings exist in Tranquility Kafka. One is zookeeper.connect used to connect the ZooKeeper of the Druid cluster, and the other is kafka.zookeeper.connect used to connect the ZooKeeper of the Kafka cluster. These two ZooKeepers may not belong to the same ZooKeeper cluster.)

· The MiddleManager reports that the com.hadoop.compression.lzo.LzoCodec class cannot be found during indexing.

This is because the Hadoop cluster of E-MapReduce is configured with lzo compression.

To resolve this issues, copy the JAR package and the native file under the EMR HADOOP_HOME/lib directory to Druid's druid.extensions.hadoopDependenciesDir (by default, DRUID_HOME/hadoop-dependencies).

· The following error is reported during indexing:

```
2018 - 02 - 01T09 : 00 : 32 , 647   ERROR  [ task - runner - 0 -
priority - 0 ]  com . hadoop . compressio  n . lzo . GPLNativeC
odeLoader  -  could   not   unpack   the   binaries
  java . io . IOExceptio  n :  No   such   file   or   directory
          at   java . io . UnixFileSy  stem . createFile
Exclusivel  y ( Native   Method ) ~[?: 1 . 8 . 0_151 ]
          at   java . io . File . createTemp  File ( File . java :
2024 ) ~[?: 1 . 8 . 0_151 ]
```

```
          at   java . io . File . createTemp  File ( File . java :
2070 ) ~[?: 1 . 8 . 0_151 ]
          at   com . hadoop . compressio  n . lzo . GPLNativeC
odeLoader . unpackBina  ries ( GPLNativeC  odeLoader . java : 115 )
[ hadoop - lzo - 0 . 4 . 21 - SNAPSHOT . jar :?]
```

This issue occurs because the java.io.tmp path does not exist. Set the path and make sure that the Druid account has permission to access it.

7 Presto

7.1 What is Presto?

Presto is an open-source distributed SQL-on-Hadoop query engine powered by Facebook. It is currently maintained by the open source community and Facebook engineers, and has derived multiple commercial versions.

Basic features

Presto is implemented in Java. It is easy to use and offers high performance and strong scalability. Its other features are as follows:

- Fully supports ANSI SQL.
- Supports rich data sources, accessing them as follows:
  - Interaction with Hive
  - Cassandra
  - Kafka
  - MongoDB
  - MySQL
  - PostgreSQL
  - SQL Server
  - Redis
  - Redshift
  - Local files
- Supports advanced data structures.
  - Array and map data
  - JSON data
  - GIS data
  - Color data

- Presto provides the following expansion configurations:
  - Data connector expansion
  - Custom data types
  - Custom SQL functions

  To achieve efficient service processes, you can expand the corresponding modules according to your own service features.
- Based on the Pipeline process model, data is returned to you in real time.
- Improved monitoring interfaces.
  - Friendly WebUI is provided to present the execution processes of the query tasks visually.
  - Supports the JMX protocol.

Scenarios

Presto is a distributed SQL engine that is well-suited to the following scenarios:

- ETL
- Ad-hoc queries
- Massive structured and semi-structured data analysis
- Massive multi-dimensional data aggregation/reports

In particular, Presto is a data warehouse product, which is not designed to replace traditional RDBMS databases such as MySQL and PostgreSQL. It has limited support for transactions and is not suitable for online service scenarios.

Benefits

In addition to being open source, the E-MapReduce Presto product comes with the following advantages:

- You can purchase it for immediate use to build a Presto cluster with hundreds of nodes in minutes.
- It supports elastic scalability, meaning that you can scale the cluster up and down with simple operations.
- It works perfectly in connection with the E-MapReduce software stacks, and supports the processing of data stored in OSS.
- O&M is free 24/7, providing an all-in-one service.

7.2 Quick start

7.2.1 System structure

Architecture

The following figure shows the architecture of Presto:

Presto has a typical mobile/server architecture comprising a coordinator node and multiple worker nodes. Coordinator is responsible for the following:

- Receiving and parsing your query requests, generating execution plans, and sending the execution plans to the worker nodes for execution.
- Monitoring the running status of the worker nodes. Each worker node maintains a heartbeat connection with the coordinator node, reporting the node statuses.
- Maintaining the metastore data

Worker nodes run the tasks assigned by the coordinator node, read data from external storage systems through connectors, process the data, and send the results to the coordinator node.

7.2.2 Basic concepts

This section describes the basic Presto concepts for a better understanding of the Presto work mechanism.

Data model

Data model indicates to the data organization form. Presto uses a three-level structure, namely Catalog, Schema, and Table, to manage data.

- Catalog

  A catalog contains multiple schemas and is physically directed to an external data source, which can be accessed through connectors. When you run an SQL statement in Presto, you are running it against one or more catalogs.
- Schema

  A schema is a database instance that contains multiple data tables.
- Table

  A data table is the same as a general database table.

The relationships between catalogs, schemas, and tables are shown in the following figure.

Connector

Presto uses connectors to connect to various external data sources. To access customized data sources, Presto provides a standard *SPI*, which allows you to develop your own connectors using this standard API.

A catalog is typically associated with a specific connector (which can be configured in the Properties file of the catalog). Presto contains multiple built-in connectors.

7.2.3 Command line tool

This section describes how to use the command line tool to operate the Presto console.

The command line tool uses *SSH to log on to an EMR cluster* and executes the following command to enter the Presto console:

```
$ presto -- server emr-header-1:9090 -- catalog hive -- schema default -- user hadoop
```

High-security clusters use the following command:

```
$ presto -- server https://emr-header-1:7778 \
         -- enable-authentication \
         -- krb5-config-path /etc/krb5.conf \
         -- krb5-keytab-path /etc/ecm/presto-conf/presto.keytab \
         -- krb5-remote-service-name presto \
         -- keystore-path /etc/ecm/presto-conf/keystore \
         -- keystore-password 81ba14ce6084 \
         -- catalog hive -- schema default \
         -- krb5-principal presto/emr-header-1.cluster-XXXX@EMR.XXXX.COM
```

- *XXXX* is the ECM ID of the cluster, a string of numbers that can be obtained through *cat /etc/hosts*.
- *81ba14ce6084* is the default password of */etc/ecm/presto-conf/keystore*. We recommend that you use your own keystore after deployment.

You can execute the following command from the console:

```
Presto:default> show schemas;
    schema
--------------------
```

```
 default
 Hive
 informatio  n_schema
 tpch_100gb  _orc
 tpch_10gb_  orc
 tpch_10tb_  orc
 tpch_1tb_o  rc
( 7   rows )
```

You can then execute the `presto -- help` command to obtain help from the console. The parameters and definitions are as follows:

```
-- server  < server >                          #  Specifies   the   URI
 of   a   Coordinato  r
-- user  < user >                              #  Sets   the   username
-- catalog  < catalog >                        #  Specifies   the
 default   Catalog
-- schema  < schema >                          #  Specifies   the
 default   Schema
-- execute  < execute >                        #  Executes   a
 statement   and   then   exits
- f  < file >, -- file  < file >                 #  Executes   an   SQL
   statement   and   then   exits
-- debug                                    #  Shows   debugging
 informatio  n
-- client - request - timeout  < timeout >      #  Specifies   the
 client   timeout   value ,  which   is   2   minutes   by   default
-- enable - authentica  tion                   #  Enables   client
 authentica  tion
-- keystore - password  < keystore   password > #  KeyStore
 password
-- keystore - path  < keystore   path >         #  KeyStore   path
-- krb5 - config - path  < krb5   config   path >   #  Kerberos
 configurat  ion   file   path  ( default : / etc / krb5 . conf )
-- krb5 - credential - cache - path  < path >     #  Kerberos
 credential   cache   path
-- krb5 - keytab - path  < krb5   keytab   path >   #  Kerberos   Key
   table   path
-- krb5 - principal  < krb5   principal >       #  Kerberos
 principal   to   be   used
-- krb5 - remote - service - name  < name >       #  Remote   Kerberos
   node   name
-- log - levels - file  < log   levels >          #  Configurat  ion
 file   path   for   debugging   logs
-- output - format  < output - format >         #  Bulk   export
 data   format ,  which   is   CSV   by   default
-- session  < session >                        #  Specifies   the
 session   attribute ,  in   the   format   key = value
-- socks - proxy  < socks - proxy >             #  Sets   the   proxy
   server
-- source  < source >                          #  Sets   query   source
-- version                                  #  Shows   version   info
```

```
- h , -- help                                    #  Shows   help   info
```

7.2.4 Uses JDBC

Java applications can access databases using the JDBC driver provided by Presto. The procedure is the same as that for general RDBMS databases.

Introduction to Maven

You can add the following configuration to the POM file to introduce the Presto JDBC driver:

```
< dependency >
    < groupId > com . facebook . presto </ groupId >
    < artifactId > presto - jdbc </ artifactId >
    < version > 0 . 187 </ version >
</ dependency >
```

Driver class name

The Presto JDBC driver class is `com . facebook . presto . jdbc . PrestoDriv er` .

Connection string

The following connection string format is supported.

```
jdbc : presto ://< COORDINATO  R >:< PORT >/[ CATALOG ]/[ SCHEMA ]
```

For example:

```
jdbc : presto :// emr - header - 1 : 9090                #  Connects
  to   data   base ,  using   the   default   Catalog   and   Schema
jdbc : presto :// emr - header - 1 : 9090 / hive           #
Connects   to   data   base ,  using   Catalog ( hive )  and   the
default   Schema
jdbc : presto :// emr - header - 1 : 9090 / hive / default   #
Connects   to   data   base ,  using   Catalog ( hive )  and   Schema
( default )
```

Connection parameters

The Presto JDBC driver supports various parameters that may be set as URL parameters or as `Properties` and passed to DriverManager.

Example of passing parameters to DriverManager as `Properties` :

```
String   url  = " jdbc : presto :// emr - header - 1 : 9090 / hive /
default ";
Properties   properties  =  new   Properties ();
properties . setPropert  y (" user ", " hadoop ");
Connection   connection  =  DriverMana  ger . getConnect  ion ( url
,  properties );
```

```
......
```

Example of passing parameters to DriverManager as URL parameters:

```
 String   url  = " jdbc : presto :// emr - header - 1 : 9090 / hive /
 default ?  user = hadoop ";
 Connection   connection  =  DriverMana  ger . getConnect  ion ( url
 );
......
```

Parameters are described as follows:

Parameter name	Format	Description
user	STRING	User name.
password	STRING	Password.
Socksproxy	\:\	SOCKS proxy server address and port. For example, localhost:1080.
httpProxy	\:\	HTTP proxy server address and port. For example, localhost:8888.
SSL	true\	Whether or not to use HTTPS for connections. This is false by default.
SSLTrustStorePath	STRING	Java TrustStore file path.
SSLTrustStorePassword	STRING	Java TrustStore password.
KerberosRemoteServiceName	STRING	Kerberos service name.
KerberosPrincipal	STRING	Kerberos principal.
KerberosUseCanonicalHostname	true\	Whether or not to use the canonical hostname. This is false by default.
KerberosConfigPath	STRING	Kerberos configuration file path.
KerberosKeytabPath	STRING	Kerberos KeyTab file path.
KerberosCredentialCachePath	STRING	Kerberos credential cache path

Java example

The following is an example of using the Presto JDBC driver with Java.

```
.....
//  Loads   the   JDBC   Driver   class
 try  {
     Class . forName (" com . facebook . presto . jdbc . PrestoDriv
 er ");
}  catch ( ClassNotFo  undExcepti  on   e ) {
```

```
    LOG . ERROR (" Failed   to   load   presto   jdbc   driver .",  e
 );
    System . exit (- 1 );
}
 Connection   connection  =  null ;
 Statement   stmt  =  null ;
 try  {
    String   url  = " jdbc : presto :// emr - header - 1 : 9090 /
 hive / default ";
    Properties   properties  =  new   Properties ();
    properties . setPropert  y (" user ", " hadoop ");
   //  Creates   the   connection   object
    Connection  =  drivermana  ger .  getconnect  ion  ( URL ,
 properties  );
   //  Creates   the   Statement   object
    statement  =  connection . createStat  ement ();
    Executes   the   query
    ResultSet   rs  =  statement . executeQue  ry (" select  *  from
  t1 ");
    Returns   results
    int   columnNum  =  rs . getMetaDat  a (). getColumnC  ount ();
    int   rowIndex  =  0 ;
    while  ( rs . next ()) {
        rowIndex ++;
        for ( int   i  =  1 ;  i  <=  columnNum ;  i ++) {
            System . out . println (" Row  " +  rowIndex  + ",
 Column  " +  i  + ": " +  rs . getInt ( i ));
       }
    }
}  catch ( SQLExcepti  on   e ) {
    LOG . ERROR (" Exception   thrown .",  e );
}  finally  {
  //  Destroys   Statement   object
   If  ( statement ! =  null ) {
       try  {
         statement . close ();
    }  catch ( Throwable   t ) {
        //  No - ops
    }
  }
   Closes   connection
   if  ( connection  ! =  null ) {
       try  {
         connection . close ();
    }  catch ( Throwable   t ) {
        //  No - ops
    }
  }
}
```

Use reverse proxy

You can use the HAProxy reverse proxy Coodinator to access the Presto service through the proxy service.

- **Non-Security Cluster proxy configuration**

  To configure a cluster proxy for a non-Security Cluster, follow these steps:

  1. **Install HAProxy on the proxy Node**
  2. **Modify the HAProxy configuration (*/ Etc / haproxy . cfg*), Add the following content:**

```
......

 listen   prestojdbc  : 9090
     Mode   TCP
     option   tcplog
     balance   source
     Server   presto - coodinator - 1   emr - header - 1 :  9090
```

  3. **Restart the HAProxy Service**

  Now, you can use the proxy server to access Presto. You only need to change the IP address of the Connected Server to the IP address of the proxy service.

7.2.5 Use ApacheDS for authentication---机翻，需重新提翻

Presto can connect to LDAP for user password authentication. You only need to connect the Coordinato r node to LDAP.

Main Steps

1. **Configure ApacheDS and enable LDAPS**
2. **Create user information in ApacheDS**
3. **Configure Presto Coordinator and restart it to take effect.**
4. **Verify Configuration**

Enable LDAPS

1. **Create the keystore used by the ApacheDS server. Here, all passwords use '123456 ':**

```
#  Create   a   keystore
>  Cd / var / lib / apacheds - 2 . 0 . 0 - M24 / default / conf /
>  Keytool - genkeypair - alias   apacheds - keyalg   RSA - validity
   7 - keystore   ads .  keystore

 Enter   keystore   password :

 Reenter   new   password :
 What   is   your   first   and   last   name ?
  [ Unknown ]:  apacheds
 What   is   the   name   of   your   organizati  onal
  [ Unknown ]:  apacheds
 What   is   the   name   of   your   organizati  on ?
  [ Unknown ]:  apacheds
 What   is   the   name   of   your   city   or
  [ Unknown ]:  apacheds
```

```
 What   is   the   name   of   your   state   or
  [ Unknown ]:  apacheds
 What   is   the   two - letter   country   code   for   this
  [ Unknown ]:  CN
 Is   CN  =  apacheds ,  OU  =  apacheds ,  O  =  apacheds ,  L  =
 apacheds ,  ST  =  apacheds ,  C  =  CN   correct ?
  [ No ]:  yes

 Enter   key   password   for  < apacheds >
 ( RETURN   if   same   as   keystore   password  ):

 Reenter   new   password :

 123Warning :
 The   JKS   keystore   uses   a   proprietar  y   format .  It
 is   recommende  d   to   migrate   to   PKCS12   which   is   an
   industry   standard   format   using  " keytool - importkeys
 tore - srckeystor  e   ads .  keystore - destkeysto  re   ads .
 keystore - deststoret  ype   pkcs12  ".

#  Modify   the   file   user ;  otherwise ,  ApacheDS   has   no
 permission   to   read   the   file
>  Chown   apacheds :  apacheds ./ ads .  keystore

#  Export   a   certificat  e .
#  Enter   the   password .  The   password   is   set   in   the
 previous   step .  The   value   is   123456 .
>  Keytool - export - alias   apacheds - keystore   ads .  keystore
 - rfc - file   apacheds .  cer
 Enter   keystore   password :
 Certificat  e   stored   in   file  < apacheds .  cer >

 123Warning :
 The   JKS   keystore   uses   a   proprietar  y   format .  It
 is   recommende  d   to   migrate   to   PKCS12   which   is   an
   industry   standard   format   using  " keytool - importkeys
 tore - srckeystor  e   ads .  keystore - destkeysto  re   ads .
 keystore - deststoret  ype   pkcs12  ".

#  Import   the   certificat  e   to   the   system   certificat  e
   library   for   self - Authentica  tion
```

```
> Keytool - import - file apacheds . cer - alias apacheds
 - keystore / usr / lib / jvm / java - 1 . 8 . 0 / jre / lib /
 security / cacerts
```

2. Modify configuration and enable LDAPS

   Open ApacheDS Studio and link to the cluster to the ApacheDS service:

   - DN: uid = admin, ou = system
   - View the password in this file:*/ Var / lib / ecm - agent / cache / ecm / service / APACHEDS / 2 . 0 . 0 . 1 . 1 / package / files / modifypwd . ldif*

   After the link, open the configuration page, enable LDAPs, set the keystore created in step 1 to the relevant configuration, and save (ctrl + s).

3. Restart ApacheDS

   Log on to the cluster and run the following command to restart ApacheDS:

```
> Service apacheds - 2 . 0 . 0 - M24 - default restart
```

   At this point, LDAPS is started, and the service port is 10636.

Note:

ApacheDS Studio has a Bug. When you test the LDAPS service connection on the connection property page, handshaking fails, mainly because the internal default timeout time is too short and does not affect actual use.

Create user information

In this example, users are created under DN: dc = hadoop, dc = apache, dc = org.

1. Create dc = hadoop, dc = apache, dc = org partition, open the configuration page, configure as follows, and save (ctrl + s). Restart the ApacheDS service.

2. Create User

   Log on to the cluster and create the following files:*/ Tmp / users . ldif*

```
# Entry for a sample people container
# Please replace with site specific values
 Dn : ou = people , dc = hadoop , dc = apache , dc =
 org
```

```
 objectclas  s :  top
 Objectclas  s :  organizati  onalUnit
 Ou :  people

#  Entry   for   a   sample   end   user
#  Please   replace   with   site   specific   values
 Dn :  uid  =  guest ,  ou  =  people ,  dc  =  hadoop ,  dc  =
 apache ,  dc  =  org
 objectclas  s :  top
 objectclas  s :  person
 objectclas  s :  organizati  onalPerson
 objectclas  s :  inetOrgPer  son
 Cn :  Guest
 Sn :  User
 Uid :  guest
 UserPasswo  rd :  guest - password

#  Entry   for   sample   user   admin
 Dn :  uid  =  admin ,  ou  =  people ,  dc  =  hadoop ,  dc  =
 apache ,  dc  =  org
 objectclas  s :  top
 objectclas  s :  person
 objectclas  s :  organizati  onalPerson
 objectclas  s :  inetOrgPer  son
 Cn :  Admin
 Sn :  Admin
 Uid :  admin
 UserPasswo  rd :  admin - password

#  Entry   for   sample   user   sam
 Dn :  uid  =  sam ,  ou  =  people ,  dc  =  hadoop ,  dc  =
 apache ,  dc  =  org
 objectclas  s :  top
 objectclas  s :  person
 objectclas  s :  organizati  onalPerson
 objectclas  s :  inetOrgPer  son
 Cn :  sam
 Sn :  sam
 Uid :  sam
 UserPasswo  rd :  sam - password

#  Entry   for   sample   user   tom
 Dn :  uid  =  tom ,  ou  =  people ,  dc  =  hadoop ,  dc  =
 apache ,  dc  =  org
 objectclas  s :  top
 objectclas  s :  person
 objectclas  s :  organizati  onalPerson
 objectclas  s :  inetOrgPer  son
 Cn :  tom
 Sn :  tom
 Uid :  tom
 UserPasswo  rd :  tom - password

#  Create   FIRST   Level   groups   branch
 Dn :  ou  =  groups ,  dc  =  hadoop ,  dc  =  apache ,  dc  =
 org
 objectclas  s :  top
 Objectclas  s :  organizati  onalUnit
 Ou :  groups
 Descriptio  n :  generic   groups   branch

#  Create   the   analyst   group   under   groups
 Dn :  cn  =  analyst ,  ou  =  groups ,  dc  =  hadoop ,  dc  =
 apache ,  dc  =  org
```

```
objectclas s : top
Objectclas s : groupofnam es
Cn : analyst
Descriptio n : analyst group
Member : uid = sam , ou = people , dc = hadoop , dc =
apache , dc = org
Member : uid = tom , ou = people , dc = hadoop , dc =
apache , dc = org


# Create the scientist group under groups
Dn : cn = scientist , ou = groups , dc = hadoop , dc =
apache , dc = org
objectclas s : top
Objectclas s : groupofnam es
Cn : scientist
Descriptio n : scientist group
Member : uid = sam , ou = people , dc = hadoop , dc =
apache , dc = or
```

Run the following command to import the user:

```
ldapmodify - x - h localhost - p 10389 - D " uid = admin ,
ou = system " - w " Ns1aSe " - a - f test . ldif
```

After the execution is complete, you can see the relevant users on ApacheDS Studio, as shown below:

Configure Presto

1. **Enable Coordinator Https**

   a. **Create the keystore used by Presto coordinator**

   ```
   # Use the script provided by EMR to generate a
    keystore
   -- keystore - path / etc / ecm / presto - conf / keystore \
   # Keystore password : 81ba14ce60 84
   > Keep CT / var / lib / ecm - agent / cache / ecm / service /
    PRESTO / 0 . 208 . 0 . 1 . 2 / package / files / tools / gen -
    keystore . exp
   ```

   b. **Configure Presto coordinator**

   Edit*/ Etc / ecm / presto - conf / config . properties* **, Add the following content:**

   ```
   Http - server . https . enabled = true
   Http - server . https . port = 7778

   Http - server . https . keystore . path =/ etc / ecm / presto -
   conf / keystore
   ```

```
Http - server . https . keystore . key = 81ba14ce60 84 .
```

2. Configure Authentication Mode to access ApacheDS

   a. Edit*/ Etc / ecm / presto - conf / config . properties* , Add the following content:

   ```
   Http - server . authentica tion . type = PASSWORD
   ```

   b. Edit *Jvm . config* , Add the following content:

   ```
   - Djavax . net . ssl . trustStore =/ usr / lib / jvm / java - 1
    . 8 . 0 / jre / lib / security / cacerts
   - Djavax . net . ssl . trustStore Password = changeit
   ```

   c. Create *Password - authentica tor . properties* , Add the following content:

   ```
   Password - authentica tor . name = ldap
   Ldap . url = ldaps : // emr - header - 1 . cluster - 84423 :
   10636
   Ldap . user - bind - pattern = uid =$ { USER }, ou =
   people , dc = hadoop , dc = apache , dc = org
   ```

   d. Create *Jndi . properties* , Add the following content:

   ```
   Java . naming . security . principal = uid = admin , ou
   = system
   Java . naming . security . credential s = { password }
   Java . naming . security . authentica tion = simple
   ```

   e. Set *Jndi . properties* Package it into the jar package and copy it to the presto library file directory:

   ```
   Jar - cvf jndi - properties . jar jndi . properties
   > Cp ./ jndi - properties . jar / etc / ecm / presto - current /
   lib /
   ```

   Note:

   - The following three parameters are used to log on to the LDAP service. However , there is no place to configure these parameters on Presto. You can add these parameters to the jvm parameters for source code analysis. (Will be filtered out): java. naming. security. principal = uid = admin, ou = system java. naming . security. credentials = ZVixyOY + 5 k java. naming. security. authentication = simple
   - Further code analysis showed that the JNDI library will use classload to load the resource file jndi. properties. Therefore, you can put these parameters in the jndi. properties file;

- Presto launcher only adds the jar file to classpath. Therefore, you need to compress this jndi. properties into a jar package and copy it to the lib directory.

3. Restart Presto to complete all configurations.

Verify Configuration

Use Presto cli to verify whether the configuration takes effect.

```
#  Use   user   sam   and   enter   the   correct   password
>  Presto  --  server   https : //  emr - header - 1 :  7778  --
 keystore - path / etc / ecm / presto - conf / keystore  --  keystore
 - password   81ba14ce60  84  --  catalog   hive  --  schema   default
  --  user   sam  --  password
 Password : < enter   the   correct   Password >
 Presto :  Default >  show   schemas ;
               schema .
--------------------------------
  Tpcds_bin_  partitione  d_orc_5
  Tpcds_oss_  bin_partit  ioned_orc_  10
  Tpcds_oss_  text_10
  Tpcds_text  _5
  Tst
( 5   rows )

 Query   20181115_0  30713_1_2_  kp5ih ,  FINISHED ,  3   nodes
 Splits :  36   total ,  36   done  ( 100 . 00 %)
 0 :  00  [ 20   rows ,  331B ] [ 41   rows / s ,  694B / s ]

#  Use   user   sam   and   enter   the   wrong   password
>  Presto  --  server   https : //  emr - header - 1 :  7778  --
 keystore - path / etc / ecm / presto - conf / keystore  --  keystore
 - password   81ba14ce60  84  --  catalog   hive  --  schema   default
  --  user   sam  --  password
 Password : < enter   the   wrong   Password >
 Presto :  Default >  show   schemas ;
 Error   running   command :  Authentica  tion   failed :  Access
 Denied :  Invalid   credential  s
```

8 TensorFlow

TensorFlow is supported by E-MapReduce 3.13.0 and later. You can add TensorFlow from the available services in your software configurations. If you are using TensorFlow in E-MapReduce to perform high-performance computing, you can allocate CPU and GPU resources through YARN.

Prerequisites

- On the software side, an E-MapReduce cluster installs TensorFlow and a TensorFlow on YARN (TOY) toolkit.
- On the hardware side, E-MapReduce supports computing using both CPU and GPU resources. If you need to use GPU computing, you can choose ECS instances from compute optimized families with GPU, such as gn5 and gn6, for the core and task nodes in the cluster. Compute optimized families with GPU support heterogene ous computing. After determining the instance type, choose the CUDA toolkit and cuDNN versions as required.

Submit TensorFlow jobs

You can log on to the master node in the E-MapReduce cluster to submit TensorFlow jobs using the command line. For example:

```
 el_submit  [- h ] [- t   APP_TYPE ] [- a   APP_NAME ] [- m   MODE ]
 [- m_arg   MODE_ARG ]

[- interact   INTERACT ] [- x   EXIT ]

[- enable_ten  sorboard   ENABLE_TEN  SORBOARD ]

[- log_tensor  board   LOG_TENSOR  BOARD ] [- conf   CONF ] [- f
 FILES ]

[- pn   PS_NUM ] [- pc   PS_CPU ] [- pm   PS_MEMORY ] [- wn
 WORKER_NUM ]

[- wc   WORKER_CPU ] [- wg   WORKER_GPU ] [- wm   WORKER_MEM  ORY ]

[- wnpg   WNPG ] [- ppn   PPN ] [- c   COMMAND  [ COMMAND  ...]]
```

The basic parameters are described as follows:

- -t APP_TYPE: Specifies the type of task to be submitted. The supported types are tensorflow-ps, tensorflow-mpi, and standalone. They are used in conjunction with the following –m MODE parameter.
  - tensorflow-ps: Uses a parameter server for the communication of data, which is the PS mode of native TensorFlow.
  - tensorflow-mpi: Uses Horovod, an open source framework from Uber, which relies on message passing interface (MPI) primitives for the communication of data.
  - standalone: Users assign tasks to one instance in the YARN cluster for execution . This is similar to standalone execution.
- –a APP_NAME: Specifies the name of the submitted TensorFlow job. You can name jobs as required.
- –m MODE: Specifies the runtime environment for submitted TensorFlow jobs. E-MapReduce supports the following environments: local, virtual-env, and docker.
  - local: Uses Python runtime environments set up in the E-MapReduce worker nodes. If you want to use third-party Python packages, you need to install the packages on all the nodes manually.
  - docker: Uses the Docker containers installed on the E-MapReduce worker nodes . TensorFlow runs in Docker containers.
  - virtual-env: Uses isolated Python environments created by users. You can install Python libraries in Python environments. These libraries can be different from those installed in the environments that are set up in the worker nodes.
- -m_arg MODE_ARG: Specifies the supplemental parameter for the –m MODE.
  If the runtime environment is docker, set the value to the docker image name.
  If the runtime environment is virtual-env, set the value to the name of Python environment tar.gz file.
- –x Exit: You need to exit the parameter servers manually for certain distributed TensorFlow APIs. To exit parameter servers automatically when worker servers finish training their models, specify the -x option.
- -enable_tensorboard: Specifies whether to enable TensorBoard when TensorFlow starts training models.
- -log_tensorboard: Specifies the location of TensorBoard logs in HDFS. If TensorBoard is enabled when TensorFlow starts training models, this parameter is required.

- -conf CONF: Specifies the location of the Hadoop configuration. Setting the value is optional. The default E-MapReduce configuration is used.
- –f FILES: Specifies all dependent files and folders for TensorFlow to run, including executable scripts. If virtual-env files that are executed in a virtual environmen t are specified, you can put all dependencies in one folder. The script then automatically uploads the folders into HDFS according to the folder hierarchy.
- -pn TensorFlow: Specifies the number of parameter servers to start.
- -pc: Specifies the number of CPU cores that each parameter server requests.
- -pm: Specifies the memory size that each parameter server requests.
- -wn: Specifies the number of worker nodes started by TensorFlow.
- -wc: Specifies the number of CPU cores that each worker requests.
- -wg: Specifies the number of GPU cores that each worker requests.
- -wm: Specifies the memory usage that each worker requests.
- -c COMMAND: Specifies the command to run. For example, pythoncensus.py.

Advanced options. We recommend that you use advanced options with care, as they may result in job failures.

- -wnpg: Specifies the number of workers that use a GPU simultaneously (for tensorflow-ps).
- -ppn: Specifies the number of workers that use a GPU simultaneously (for Horovod). The preceding options refer to multitasking on a single GPU. Thresholds should be set to avoid GPU running out of memory.

9 Knox

E-MapReduce supports *Apache Knox*. If you select a Knox-supported image to create a cluster, you can access the Web UI from the public network to use services such as YARN, HDFS, and SparkHistory.

Preparations

- Enable Knox access using a public IP address
  1. The service port of Knox on E-MapReduce is 8443. In the cluster details, find the ECS security group in which the cluster is located.
  2. Change the corresponding security group in the ECS console and add a rule in Internet inbound to enable port 8443.

  > Notice:
  >
  > - For security reasons, the authorization object must be your limited IP address range. 0.0.0.0/0 is forbidden.
  > - After port 8443 of the security group is enabled, all nodes (including non-E-MapReduce ECS nodes) in the security group enable port 8443 at the ingress of the public network.

- Set a Knox user

  Accessing Knox requires a user name and password for authentication. The authentication is based on LDAP. You can use your own LDAP service or the LDAP service of Apache Directory Server in the cluster.

  - Use the LDAP service in the cluster

    Method one (recommended):

    Add a Knox account in the *User Management* page.

    Method two:

    1. Log on to the cluster through SSH. For more information, see *Connect to clusters using SSH*.
    2. Prepare your user data. Here, Tom is used as the user name. In the file, replace all `emr-guest` with `Tom` and `cn: EMR GUEST` with `cn: Tom`, and set `userPasswo rd` to your password.

       ```
       su knox
       cd /usr/lib/knox-current/templates
       vi users.ldif
       ```

       

       Notice:
       For security reasons, before you export your user data to LDAP, change the password of users.ldif by changing `userPasswo rd` to your password.
    3. Export to LDAP.

       ```
       su knox
       cd /usr/lib/knox-current/templates
       sh ldap-sample-users.sh
       ```

  - Use your own LDAP service

    1. Enter the cluster configuration management page. In the cluster-topo configuration, set `main.ldapRealm.userDnTemp late` to your user DN template and `main.ldapRealm.contextFac tory.url` to your LDAP server domain name and port. Then, save the settings and restart Knox.

    2. Your LDAP service does not typically run in the cluster. You must enable the Knox port to access the LDAP service in the public network, such as port

10389. For more information, see the preceding steps for enabling port 8443. Then, select Internet outbound.

 Notice:
For security reasons, the authorization object must be the public IP address of your Knox cluster. 0.0.0.0/0** is forbidden.

Access Knox

- Access using the E-MapReduce shortcut link
  1. Log on to the *E-MapReduce console*.
  2. Click the ID link of the target cluster.
  3. In the navigation pane on the left, click Clusters and Services.
  4. Click the relevant services on the E-MapReduce services page, such as HDFS and YARN.
  5. In the upper-right corner, click Quick Link.
- Access using the public IP address of the cluster
  1. Check the public IP address in the cluster details.
  2. Access the URLs of the relevant services in the browser.
     - HDFS UI: *https://{cluster_access_ip}:8443/gateway/cluster-topo/hdfs/*
     - YARN UI: *https://{cluster_access_ip}:8443/gateway/cluster-topo/yarn/*
     - SparkHistory UI: *https://{cluster_access_ip}:8443/gateway/cluster-topo/sparkhistory/*
     - Ganglia UI: *https://{cluster_access_ip}:8443/gateway/cluster-topo/ganglia/*
     - Storm UI: *https://{cluster_access_ip}:8443/gateway/cluster-topo/storm/*
     - Oozie UI: *https://{cluster_access_ip}:8443/gateway/cluster-topo/oozie/*
  3. website is not security is displayed in your browser because the Knox service uses a self-signed certificate. Confirm that the accessed IP address is the same as that of your cluster and the port is 8443. Click advance > continue.
  4. Enter the user name and password set in LDAP in the logon dialog box.

Access control lists

Knox provides service-level permission management to limit service access to specific users, user groups, or IP addresses. See *Apache Knox Authorization*.

- Example
  - Scenario: The YARN UI only allows access by user Tom.
  - Steps: Enter the cluster configuration management page. In the cluster-topo configuration, add access control list (ACL) code between the `< gateway >...</ gateway >` labels.

```
< provider >
      < role > authorizat  ion </ role >
      < name > AclsAuthz </ name >
      < enabled > true </ enabled >
      < param >
          < name > YARNUI . acl </ name >
          < value > Tom ;*;*</ value >
      </ param >
</ provider >
```

- Notes

  Knox provides RESTful APIs for operating a range of services, including adding or deleting HDFS files. For security reasons, make sure that when you enable port 8443 of the security group in the ECS console, the authorization object is your limited IP address range. 0.0.0.0/0 is forbidden. Do not use the LDAP user name and password in the Knox installation directory to access Knox.

10 Instructions for using Flume

E-MapReduce version 3.16.0 and later support Apache Flume. This topic describes how to use Flume to synchronize E-MapReduce Kafka cluster data to HDFS, Hive, and HBase running on E-MapReduce Hadoop clusters, and to Alibaba Cloud OSS.

Prerequisites

- You must have selected Flume in the Optional Service menu when you created a Hadoop cluster.
- You must have created a Kafka cluster and created a topic named flume-test to generate data.

> Note:
>
> - If you have created a high security mode Hadoop cluster to consume standard Kafka cluster data, and you need to configure Kerberos authentication on the Hadoop cluster, see *Authentication method compatible with MIT Kerberos*.
> - If you have created a high security mode Kafka cluster and you need to write data to a standard Hadoop cluster using Flume, see *Kerberos Kafka Source in this topic*.
> - If you have created a high security mode Hadoop cluster and a high security mode Kafka cluster, and you need to configure Kerberos, see *Cross-region access* and *Cross-region access using Flume*.

Kafka->HDFS

- Configure Flume

  Create a configuration file named `flume.properties`, and add the following configurations for the file, where `a1.sources.source1.kafka.bootstrap.servers` indicates the host and port for a Kafka broker, `a1.sources.source1.kafka.topics` indicates the Kafka topic where Flume is used to consume data, and `a1.sinks.k1.hdfs.path` indicates the path where Flume writes data to HDFS.

```
a1.sources = source1
a1.sinks = k1
a1.channels = c1

a1.sources.source1.type = org.apache.flume.source.kafka.KafkaSourc e
```

```
 a1 . sources . source1 . channels  =  c1
 a1 . sources . source1 . kafka . bootstrap . servers  =  kafka -
 host1 : port1 , kafka - host2 : port2 ...
 a1 . sources . source1 . kafka . topics  =  flume - test
 a1 . sources . source1 . kafka . consumer . group . id  =  flume -
 test - group

#  Describe   the   sink
 a1 . sinks . k1 . type  =  hdfs
 a1 . sinks . k1 . hdfs . path  = / tmp / flume / test - data
 a1 . sinks . k1 . hdfs . fileType = DataStream

#  Use   a   channel   which   buffers   events   in   memory
 a1 . channels . c1 . type  =  memory
 a1 . channels . c1 . capacity  =  100
 a1 . channels . c1 . transactio  nCapacity  =  100

#  Bind   the   source   and   sink   to   the   channel
 a1 . sources . source1 . channels  =  c1
 a1 . sinks . k1 . channel  =  c1
```

- **Start Flume**

  Flume's default configuration file is stored in */ etc / ecm / flume - conf* **. Use the following configuration file to start a Flume agent.**

  ```
  flume - ng   agent  -- name   a1  -- conf  / etc / ecm / flume -
  conf   -- conf - file   flume . properties
  ```

  After the agent is started, the log logs/flume.log will be generated in the current path due to *log4j . properties* **being in** */ etc / ecm / flume - conf* **. You can configure** *log4j . properties* **according to your requirements.**

- **Test**

  Use the kafka-console-producer.sh command and input the test data abc in your Kafka cluster.

  Flume generates a file FlumeData.xxx with a timestamp (in milliseconds) suffix based on the current time. When you view the file content, you can see the data that you input in Kafka.

Kafka->Hive

- **Create a Hive table**

  Before Flume writes data into Hive using transactions, you need to set the `transactio nal` property when creating a Hive table. The following example shows how to create a table named flume_test table.

```
create table flume_test ( id int , content string )
clustered by ( id ) into 2 buckets
stored as orc TBLPROPERT IES (' transactio nal '=' true
');
```

- **Configure Flume**

  Create a configuration file flume.properties and add the following configurations for the file, where `a1 . sources . source1 . kafka . bootstrap . servers` indicates the host and port for a Kafka broker and `a1 . sinks . k1 . hive . metastore` indicates a Hive metastore URI. Then, configure the value of `hive . metastore . uris` in the *hive - site . xml* file:

```
a1 . sources = source1
a1 . sinks = k1
a1 . channels = c1

a1 . sources . source1 . type = org . apache . flume . source .
kafka . KafkaSourc e
a1 . sources . source1 . channels = c1
a1 . sources . source1 . kafka . bootstrap . servers = kafka -
host1 : port1 , kafka - host2 : port2 ...
a1 . sources . source1 . kafka . topics = flume - test
a1 . sources . source1 . kafka . consumer . group . id = flume -
test - group

# Describe the sink
a1 . sinks . k1 . type = hive
a1 . sinks . k1 . hive . metastore = thrift :// xxxx : 9083
a1 . sinks . k1 . hive . database = default
a1 . sinks . k1 . hive . table = flume_test
a1 . sinks . k1 . serializer = DELIMITED
a1 . sinks . k1 . serializer . delimiter = ","
a1 . sinks . k1 . serializer . serdeSepar ator = ','
a1 . sinks . k1 . serializer . fieldnames = id , content

a1 . channels . c1 . type = memory
a1 . channels . c1 . capacity = 100
a1 . channels . c1 . transactio nCapacity = 100

a1 . sources . source1 . channels = c1
```

```
a1 . sinks . k1 . channel  =  c1
```

- Start Flume

```
flume - ng   agent  -- name   a1  -- conf  / etc / ecm / flume -
conf   -- conf - file   flume . properties
```

- Generate data

  Use the *kafka - console - producer . sh* command and input the comma-separated test data 1,a in your Kafka cluster.

- Verify the input data

  Note that quering Hive transaction tables require configuration on the Hive client:

```
hive . support . concurrenc  y  -  true
hive . exec . dynamic . partition . mode  -  nonstrict
hive . txn . manager  -  org . apache . hadoop . hive . ql .
lockmgr . DbTxnManag  er
```

  After the preceding configurations are set, you can query data in the flume_test table.

Kafka->HBase

- Create a HBase table

  Create a HBase table flume_test and a column family column.

- Configure Flume

  Create a configuration file *flume . properties* and add the following configurations, where `a1 . sources . source1 . kafka . bootstrap . servers` indicates the host and port for a Kafka broker, `a1 . sinks . k1 . table` indicates the name of the HBase table, and `a1 . sinks . k1 . columnFami  ly` indicates the name of the column family:

```
a1 . sources  =  source1
a1 . sinks  =  k1
a1 . channels  =  c1

a1 . sources . source1 . type  =  org . apache . flume . source .
kafka . KafkaSourc  e
a1 . sources . source1 . channels  =  c1
a1 . sources . source1 . kafka . bootstrap . servers  =  kafka -
host1 : port1 , kafka - host2 : port2 ...
a1 . sources . source1 . kafka . topics  =  flume - test
a1 . sources . source1 . kafka . consumer . group . id  =  flume -
test - group

a1 . sinks . k1 . type  =  hbase
```

```
 a1 . sinks . k1 . table  =  flume_test
 a1 . sinks . k1 . columnFami  ly  =  column


#  Use   a   channel   which   buffers   events   in   memory
 a1 . channels . c1 . type  =  memory
 a1 . channels . c1 . capacity  =  1000
 a1 . channels . c1 . transactio  nCapacity  =  100

#  Bind   the   source   and   sink   to   the   channel
 a1 . sources . source1 . channels  =  c1
 a1 . sinks . k1 . channel  =  c1
```

- **Start Flume**

```
 flume - ng   agent  -- name   a1  -- conf  / etc / ecm / flume -
 conf   -- conf - file   flume . properties
```

- **Test**

  After data is generated using kafka-console-producer.sh in your Kafka cluster, you can query data in HBase.

Kafka->OSS

- **Create an OSS path**

  Create an OSS bucket and directory, such as *oss :// flume - test / result* **.**

- **Configure Flume**

  Flume requires a large amount of JVM memory when writing data to OSS. To resolve this issue, you can:

  - **Reduce the OSS cache size**

    Copy the file hdfs-site.xml from */ etc / ecm / hadoop - conf* **to** */ etc / ecm / flume - conf* **, and reduce the value of the configuration term** `smartdata . cache . buffer . size` **, for example, to 1048576.**

  - **Increase the Flume agent's heap size (Xmx)**

    In the Flume configuration path */ etc / ecm / flume - conf* **, copy configuration file** *flume - env . sh . template* **, paste it to the /etc/ecm/**

flume-conf path , rename it *flume - env . sh* , and set Xmx, for example, to 1G:

```
export   JAVA_OPTS ="- Xmx1g "
```

Create a configuration file *flume . properties* and add the following configurations, where *a1 . sources . source1 . kafka . bootstrap . servers* indicates the host and port for a Kafka broker, and *a1 . sinks . k1 . hdfs . path* indicates an OSS path:

```
a1 . sources  =  source1
a1 . sinks  =  k1
a1 . channels  =  c1

a1 . sources . source1 . type  =  org . apache . flume . source .
kafka . KafkaSourc  e
a1 . sources . source1 . channels  =  c1
a1 . sources . source1 . kafka . bootstrap . servers  =  kafka -
host1 : port1 , kafka - host2 : port2 ...
a1 . sources . source1 . kafka . topics  =  flume - test
a1 . sources . source1 . kafka . consumer . group . id  =  flume -
test - group

a1 . sinks . k1 . type  =  hdfs
a1 . sinks . k1 . hdfs . path  =  oss :// flume - test / result
a1 . sinks . k1 . hdfs . fileType = DataStream

#  Use   a   channel   which   buffers   events   in   memory
a1 . channels . c1 . type  =  memory
a1 . channels . c1 . capacity  =  100
a1 . channels . c1 . transactio  nCapacity  =  100

#  Bind   the   source   and   sink   to   the   channel
a1 . sources . source1 . channels  =  c1
a1 . sinks . k1 . channel  =  c1
```

- Start Flume

  If you modified the OSS cache size when configuring Flume, use the classpath parameter to pass OSS-related dependencies and configurations to Flume:

```
flume - ng   agent  -- name   a1  -- conf  / etc / ecm / flume -
conf   -- conf - file   flume . properties   -- classpath  "/ opt
```

```
/ apps / extra - jars /*:/ etc / ecm / flume - conf / hdfs - site .
xml "
```

If you modified the Flume agent's Xmx, you only need to pass OSS-related dependencies:

```
flume - ng   agent   -- name   a1   -- conf   / etc / ecm / flume -
conf   -- conf - file   flume . properties   -- classpath   "/ opt /
apps / extra - jars /*"
```

- Test

  After data is generated using *kafka - console - producer . sh* in your Kafka cluster, in the OSS path *oss :// flume - test / result* , a file *FlumeData . xxxx* is generated with a timestamp (in milliseconds) suffix based on the current time.

Kerberos Kafka source

If high security Kafka cluster data is consumed, you must configure the following variables:

- In your Kafka cluster, configure Kerberos authentication and copy the generated keytab file *test . keytab* to the Hadoop cluster path */ etc / ecm / flume - conf* , and copy the Kafka cluster file */ etc / ecm / has - conf / krb5 . conf* to the Hadoop cluster path /etc/ecm/flume-conf. For more information, see *Authentication method compatible with MIT Kerberos*.
- Configure *flume . properties*

  In the *flume . properties* file, add the following configurations:

```
a1 . sources . source1 . kafka . consumer . security . protocol  =
SASL_PLAIN  TEXT
a1 . sources . source1 . kafka . consumer . sasl . mechanism  =
GSSAPI
a1 . sources . source1 . kafka . consumer . sasl . kerberos .
service . name  =  kafka
```

- Configure the Kafka client
  - In */ etc / ecm / flume - conf* , create the file *flume \ _jaas . conf* with the following content:

```
KafkaClien  t  {
  com . sun . security . auth . module . Krb5LoginM  odule
required
  useKeyTab = true
  storeKey = true
  keyTab ="/ etc / ecm / flume - conf / test . keytab "
```

```
    serviceNam  e =" kafka "
    principal =" test @ EMR .${ realm }.  COM ";
};
```

where, `${ realm }` is replaced with a Kerberos realm of your Kafka cluster. To obtain a Kerberos realm, in your Kafka cluster, run the command `hostname` to obtain a hostname in the form of *emr - header - 1 . cluster - xxx* , such as *mr - header - 1 . cluster - 123456* . The last numeric string 123456 is a Kerberos realm.

- Change */ etc / ecm / flume - conf / flume - env . sh*

  By default, in */ etc / ecm / flume - conf /*, the file *flume - env . sh* does not exist. You need to copy *flume - env . sh . template* , paste it to /etc/ecm/flume-conf/, rename it *flume - env . sh* , and add the following content:

```
export   JAVA_OPTS ="$ JAVA_OPTS  - Djava . security . krb5 .
conf =/ etc / ecm / flume - conf / krb5 . conf "
export   JAVA_OPTS ="$ JAVA_OPTS  - Djava . security . auth .
login . config =/ etc / ecm / flume - conf / flume_jaas . conf "
```

- Set domain name

  Add domain names and IP binding information of each Kafka cluster node to */ etc / hosts* of the Hadoop cluster. The form of the long domain name is *emr - header - 1 . cluster - 123456* .

Cross-region access using Flume

After cross-region access is configured, you need to set other configurations as follows:

- In your Kafka cluster, configure Kerberos authentication and copy the generated keytab file *test . keytab* to the Hadoop cluster path */ etc / ecm / flume - conf* . For more information, see *Authentication method compatible with MIT Kerberos*.
- Configure *flume . properties*

  In the *flume . properties* file, add the following configurations:

```
a1 . sources . source1 . kafka . consumer . security . protocol  =
SASL_PLAIN  TEXT
a1 . sources . source1 . kafka . consumer . sasl . mechanism  =
GSSAPI
```

```
a1 . sources . source1 . kafka . consumer . sasl . kerberos .
service . name  =  kafka
```

- Configure the Kafka client
  - In */ etc / ecm / flume - conf* , create the file *flume \ _jaas . conf* with the following content:

```
 KafkaClien  t  {
   com . sun . security . auth . module . Krb5LoginM  odule
 required
   useKeyTab = true
   storeKey = true
   keyTab ="/ etc / ecm / flume - conf / test . keytab "
   serviceNam  e =" kafka "
   principal =" test @ EMR .${ realm }.  COM ";
};
```

    where, `${ realm }` is replaced with a Kerberos realm of your Kafka cluster. To obtain a Kerberos realm, in your Kafka cluster, run the command `hostname` to obtain a hostname in the form of *emr - header - 1 . cluster - xxx* , such as *emr - header - 1 . cluster - 123456* . The last numeric string 123456 is a Kerberos realm.
  - Change */ etc / ecm / flume - conf / flume - env . sh*

    By default, in */ etc / ecm / flume - conf /*, the file *flume - env . sh* does not exist. You need to copy *flume - env . sh . template* , paste it to */ etc / ecm / flume - conf /*, rename it flume-env.sh, and add the following content:

```
export   JAVA_OPTS ="$ JAVA_OPTS  - Djava . security . auth .
login . config =/ etc / ecm / flume - conf / flume_jaas . conf ""
```

11 Sqoop

Sqoop is an open-source application that is used to transfer data between different data stores. It supports various data stores.

Install Sqoop

Notice:

Sqoop has been integrated with E-MapReduce since E-MapReduce 1.3. If you are using E-MapReduce 1.3 or later, you can skip this section.

If the version you are using is earlier than E-MapReduce 1.3, you can install Sqoop as follows:

1. Download Sqoop 1.4.6 from the official site (*Click to download*). If you cannot open the *sqoop-1.4.6.bin__hadoop-2.0.4-alpha.tar.gz* file that you downloaded, try to download the file from the mirror site *http://mirror.bit.edu.cn/apache/sqoop/1.4.6/sqoop-1.4.6.bin__hadoop-2.0.4-alpha.tar.gz* by executing the following command.

   ```
   wget http://mirror.bit.edu.cn/apache/sqoop/1.4.6/sqoop-1.4.6.bin__hadoop-2.0.4-alpha.tar.gz
   ```

2. Execute the following command to extract the *sqoop-1.4.6.bin__hadoop-2.0.4-alpha.tar.gz* file to the Master node.

   ```
   tar zxf sqoop-1.4.6.bin__hadoop-2.0.4-alpha.tar.gz
   ```

3. Install the MySQL driver to import data from MySQL. Download the latest version from the official site (*Click to download*). In addition, you can execute the following command to download the latest version (take version 5.1.38 as an example).

   ```
   wget https://dev.mysql.com/get/Downloads/Connector-J/mysql-connector-java-5.1.38.tar.gz
   ```

4. Extract the jar file to the lib folder in the Sqoop folder.

Transfer data

Scenarios:

- MySQL -> HDFS
- HDFS -> MySQL
- MySQL -> Hive
- Hive -> MySQL
- Free-form query imports

 **Notice:**
You must switch your user account to hadoop before executing commands in later sections.

```
su   hadoop
```

- Import data from MySQL into HDFS

  Execute the following command on the Master node of the cluster:

```
sqoop   import  -- connect   jdbc : mysql ://< dburi >/< dbname
> -- username  < username > -- password  < password > -- table  <
tablename > -- check - column  < col > -- incrementa  l  < mode >
-- last - value  < value > -- target - dir  < hdfs - dir >
```

  Parameters:

  - dburi: the connection string of a database such as *jdbc : mysql :// 192 . 168 . 1 . 124 : 3306 /*. When a connection string includes any parameter, you must enclose the connection string within single quotation marks (') such

as *jdbc : mysql :// 192 . 168 . 1 . 124 : 3306 / mydatabase ? useUnicode = true*

- dbname: the name of a database such as user.
- username: the username that is used to log on to a database.
- password: the password that username with the username.
- tablename: the name of a MySQL table.
- col: the name of a column to be queried.
- mode: specifies how Sqoop determines which rows are new. Valid values: append and lastmodified.
- value : specifies the maximum value of a column to be checked from the previous import.
- hdfs-dir: the HDFS directory that you import data into such as*/ user / hive / result .*

For more information about parameters, see *Sqoop import*.

- Import data from HDFS into MySQL

  You must create MySQL tables that comply with the data structure of HDFS in advance. Then you can execute the following command on the Master node of a cluster to specify a directory that you import data into.

```
sqoop   export  -- connect   jdbc : mysql ://< dburi >/< dbname
> -- username  < username > -- password  < password > -- table  <
tablename > -- export - dir  < hdfs - dir >
```

  Parameters:

  - dburi: the connection string of a database such as *jdbc : mysql :// 192 . 168 . 1 . 124 : 3306 /*. When any parameter is included in a connection string, enclose the connection string within single quotation marks (') such

as *jdbc : mysql :// 192 . 168 . 1 . 124 : 3306 / mydatabase ? useUnicode = true*

- dbname: the name of a database, such as user.
- username: the username that is used to log on to a database.
- password: the password that is associated with the username.
- tablename: the name of a MySQL table.
- hdfs-dir: the directory of HDFS from which you import data into MySQL such as */ user / hive / result .*

For more information about parameters, see *Sqoop export*

- Import data from MySQL into Hive

When you execute the following command on the Master node of a cluster to import data from MySQL, a Hive table will be created as follows.

```
sqoop   import  -- connect   jdbc : mysql ://< dburi >/< dbname
> -- username  < username > -- password  < password > -- table  <
tablename > -- check - column  < col > -- incrementa  l  < mode >
-- last - value  < value > -- fields - terminated - by  "\ t " --
lines - terminated - by  "\ n " -- hive - import  -- target - dir
< hdfs - dir > -- hive - table  < hive - tablename >
```

Parameters:

- dburi: the connection string of a database such as *jdbc : mysql :// 192 . 168 . 1 . 124 : 3306 /*. When a connection string includes any parameter, you must enclose the connection string within single quotation marks (') such

as *jdbc : mysql :// 192 . 168 . 1 . 124 : 3306 / mydatabase ? useUnicode = true*

- dbname: the name of a database such as user.
- username: the username that is used to log on to a database.
- password: the password that is associated with the username.
- tablename: the name of a MySQL table.
- col: the name of a column to be queried.
- mode: specifies how Sqoop determine which rows are new. Valid values: append and lastmodified. When you import data into Hive by Sqoop, you cannot use the append mode.
- value: specifies the maximum value of a column to be queried from the previous import.
- hdfs-dir: the directory of HDFS from which you import data into MySQL such as */ user / hive / result .*
- hive-tablename: the table name of Hive such as xxx.yyy.

For more information about how to use parameters, see *Sqoop import*.

- Import data from Hive into MySQL

  You can refer to the previous command that is used to import data from HDFS into MySQL. In addition, you must specify the HDFS directory of Hive tables from which you import data into MySQL.

- Import data from MySQL into OSS

  The process is similar to importing data from MySQL into HDFS, except for the configuration of the target-dir parameter. You can execute the following command on the Master node of a cluster:

```
sqoop   import  -- connect   jdbc : mysql ://< dburi >/< dbname
> -- username  < username > -- password  < password > -- table
< tablename > -- check - column  < col > -- incrementa  l  < mode
> -- last - value  < value > -- target - dir  < oss - dir > --
temporary - rootdir  < oss - tmpdir >
```

Notice:

- The endpoint of an OSS host can be: intranet endpoint, Internet endpoint, or VPC endpoint. For a classic network, you must specify the intranet endpoint. For example, the OSS intranet endpoint of the China (Hangzhou) region is `oss`

-cn-hangzhou-internal.aliyuncs.com. For a VPC, you must specify the VPC endpoint. For example, the OSS VPC endpoint of the China (Hangzhou) region is *vpc100-oss-cn-hangzhou.aliyuncs.com*.

- Currently, when you import data into OSS, you cannot specify the `delete-target-dir` parameter. Otherwise, the error message Wrong FS occurs. When you want to overwrite a directory, you can execute the `hadoop fs -rm -r osspath` command to remove this OSS directory before using Sqoop.

```
sqoop import --connect jdbc:mysql://<dburi>/<dbname
> --username <username> --password <password> --table
<tablename> --check-column <col> --incrementa l <mode
> --last-value <value> --target-dir <oss-dir> --
temporary-rootdir <oss-tmpdir>
```

Parameters:

- dburi: the connection string of a database such as *jdbc:mysql://192.168.1.124:3306/*. When a connection string includes any parameter, enclose the connection string within single quotation marks (') such as *jdbc:mysql://192.168.1.124:3306/mydatabase?useUnicode=true*
- dbname: the name of a database such as user.
- username: the username that is used to log on to a database.
- password: the password that is associated with the username.
- tablename: the name of a MySQL table.
- col: the name of a column to be queried.
- mode: used by Sqoop to determine which rows are new rows. Valid values: append and lastmodified.
- value: specifies the maximum value of a column to be queried from the previous import.
- oss-dir: the OSS directory that you import data into *oss://<accessid>:<accesskey>@<bucketname>.oss-cn-hangzhou-internal.aliyuncs.com/result* 。
- oss-tmpdir: the temporary target folder. You must specify this parameter when you specify the append mode. If the destination directory already exists in HDFS, Sqoop will stop to import and overwrite that directory’s contents. If you

specify the append mode, Sqoop will import data to a temporary directory and then rename the files to the normal target directory in a manner that does not conflict with the existing filenames in that directory.

For more information about available parameters, see *Sqoop import*.

- Import data from OSS into MySQL

  The process is similar to importing data from MySQL to HDFS, except for the configuration of the export-dir parameter. You must create MySQL tables that comply with the data structure of OSS in advance.

  Then you can execute the following command on the Master node of a cluster to specify the directory from which you want to import data.

  ```
  sqoop   export  -- connect   jdbc : mysql ://< dburi >/< dbname
  > -- username  < username > -- password  < password > -- table  <
  tablename > -- export - dir  < oss - dir >
  ```

  Parameters:

  - dburi: the connection string of a database such as *jdbc : mysql :// 192 . 168 . 1 . 124 : 3306 /* . When a connection string includes any parameter, you must enclose this connection string within single quotation marks (') such as *jdbc : mysql :// 192 . 168 . 1 . 124 : 3306 / mydatabase ? useUnicode = true*
  - dbname: the name of a database such as user.
  - username: the username that is used to log on to a database.
  - password: the password that is associated with the username.
  - tablename: the name of a MySQL table.
  - oss-dir: the OSS directory that you import data into such as *oss ://< accessid >:< accesskey >@< bucketname >. oss - cn - hangzhou - internal . aliyuncs . com / result .*
  - oss-tmpdir: the temporary directory that you import data into. You must specify this parameter when you specify the append mode. If the destination directory already exists in HDFS, Sqoop will stop to import and overwrite that directory ' s contents. If you specify the append mode, Sqoop will import data to a

temporary directory and then rename the files into the normal target directory in a manner that does not conflict with existing filenames in that directory.

 Note:
The endpoint of an OSS host can be: intranet endpoint, Internet endpoint, or VPC endpoint. For a classic network, you must specify an intranet endpoint. For example, the OSS intranet endpoint of the China (Hangzhou) region is `oss-cn-hangzhou-internal.aliyuncs.com`. For a VPC, you must specify a VPC endpoint. For example, the OSS VPC endpoint of the China (Hangzhou) region is `vpc100-oss-cn-hangzhou.aliyuncs.com`.

For more information about available parameters, see *Sqoop export*.

- Free-form query imports

  In addition to importing a set of MySQL tables, you can also import the result set of an arbitrary SQL query as follows:

```
sqoop import --connect jdbc:mysql://<dburi>/<dbname
> --username <username> --password <password> --query <
query-sql> --split-by <sp-column> --hive-import --
hive-table <hive-tablename> --target-dir <hdfs-dir>
```

  Parameters:

  - dburi: the connection string of a database, such as `jdbc:mysql://192.168.1.124:3306/`. When a connection string includes any parameter, you must enclose this connection string within single quotation marks (') such

as `jdbc:mysql://192.168.1.124:3306/mydatabase?useUnicode=true`

- dbname: the name of a database such as user.
- username: the username that is used to log on to a database.
- password: the password that is associated with the username.
- query-sql: the query statement such as `SELECT * FROM profile WHERE id > 1 AND \$ CONDITIONS`. You must enclose the query statement that ends with `AND \$ CONDITIONS` within double quotation marks (").
- sp-column: specifies the name of a column to be split. In general, the value of this parameter is the primary key of the MySQL table.
- hdfs-dir: the directory of HDFS from which you import data into MySQL such as `/user/hive/result`.
- hive-tablename: the name of a table that is used to import data to Hive such as xxx.yyy.

For more information about available parameters, see *Sqoop query import*.

12 Component authorization

12.1 HDFS authorization

After HDFS has been enabled, you need permission to access it in order to perform operations, such as read data or create folders.

Add a configuration

The configurations related to HDFS permission are as follows:

- dfs.permissions.enabled

  Enable permission check. Even if the value is false, chmod/chgrp/chown/setfacl performs a permission check.
- dfs.datanode.data.dir.perm

  The permission of the local folder used by datanode, which is 755 by default.
- fs.permissions.umask-mode
  - Permission mask (default permission settings when creating a new file/folder)
  - File creation: 0666 & ^umask
  - Folder creation: 0777 & ^umask
  - Default umask value is 022. This means that the permission for file creation is 644 (666&^022 = 644), and permission of folder creation is 755 (777&^022 = 755).
  - The default setting of the Kerberos security cluster in E-MapReduce is 027. The permission for file creation is 640. For folder creation, it is 750.
- dfs.namenode.acls.enabled
  - Enable ACL control. This gives you permission control for owners/groups, and you can also set it for other users.
  - Commands for setting ACL:

    ```
    hadoop   fs  - getfacl  [- R ] < path >
    hadoop   fs  - setfacl  [- R ] [- b  |- k  - m  |- x  < acl_spec
    > < path >] |[-- set  < acl_spec >   < path >]
    ```

    For example:

    ```
    su   test
    # The   user   test   creates   a   folder
     hadoop   fs  - mkdir  / tmp / test
    # View   the   permission   of   the   created   folder
    ```

```
 hadoop  fs - ls / tmp
 drwxr - x ---  - test  hadoop  0  2017 - 11 - 26
  21 : 18 / tmp / test
# Set  ACL  and  grant  rwx  permission  s  to  user
foo
 hadoop  fs - setfacl - m  user : foo : rwx / tmp / test
# View  the  permission  of  the  file  (+  means  that
ACL  is  set )
 hadoop  fs - ls  / tmp /
 drwxrwx ---+ - test  hadoop  0  2017 - 11 - 26
21 : 18 / tmp / test
# View  ACL
  hadoop  fs - getfacl  / tmp / test
 #  file : / tmp / test
#  owner :  test
#  group :  hadoop
 user :: rwx
 user : foo : rwx
 group :: r - x
 mask :: rwx
 other ::---
```

- **dfs.permissions.superusergroup**

  Super user group. Users in this group have super user permissions.

Restart the HDFS service

For Kerberos security clusters, HDFS permissions have been set by default (umask is set to 027). Configuration and service restart are not necessary.

For non-Kerberos security clusters, a configuration must be added and the service must be restarted.

Other

- **The umask value can be modified as required.**
- **HDFS is a basic service, and Hive/HBase are based on HDFS. Therefore, HDFS permission control must be configured in advance when configuring other upper-layer services.**
- **When permissions are enabled for HDFS, the services must be set up (such as /spark-history for Spark and /tmp/$user/ for YARN).**
- **Sticky bit:**

  Sticky bit can be set for a folder to prevent anyone other than super user/file owner /directory owner from deleting files or folders in the folder (even if other users have rwx permissions for that folder). For example:

```
# That  is ,  adding  numeral  1  as  the  first  digit
 hadoop  fs - chmod  1777 / tmp
 hadoop  fs - chmod  1777 / spark - history
```

```
hadoop fs -chmod 1777 /user/hive/warehouse
```

12.2 YARN authorization

YARN authorization can be divided into service-level and queue-level authorization.

Service-level authorization

For more information, see Hadoop's *Service Level Authorization Guide*.

- Controls users' access to cluster services, such as job submission.
- Configures hadoop-policy.xml.
- Service-level permission checks are performed before other permission checks (such as for HDFS permission and YARN job submission)

> **Note:**
> Typically, if HDFS permission checks and YARN job submission controls have been set up, you may choose not to set the service-level permission control. Perform the relevant configurations as required.

Queue-level authorization

YARN supports permission control for resources by means of queues, and provides two queue scheduling methods: Capacity Scheduler and Fair Scheduler. Capacity Scheduler is used as an example here.

- Add a configuration

  A queue also has two levels of authorization: for job submission (submitting a job to the queue) and for queue management.

  > **Note:**
  > - The ACL control object for a queue is a user or group. When you are defining the related parameters, users and groups can be set at the same time, separated by spaces. You can use a comma to separate different users or groups. Only one space indicates that no one has permission.
  > - ACL inheritance for a queue: If a user or group can submit an application to a queue, they can also submit applications to all of its sub-queues. The ACL that manages queues can also be inherited. If you want to prevent a user or group

from submitting jobs to a queue, you must set the ACL for this queue and all its parent queues to restrict the job submission permission for this user or group.

- yarn.acl.enable

  Set the ACL switch to true.

- yarn.admin.acl

  - The YARN administrator setting, which enables or disables the execution of `yarn rmadmin / yarn kill` and other commands. This value must be configured. If not, the subsequent queue-based ACL administrator settings do not take effect.
  - You can set the user or group when setting the parameter values:

    ```
    user1 , user2 group1 , group2 # users and groups are
      separated by a space
      group1 , group2 # In case there are only groups
    , a leading space is required .
    ```

    In an E-MapReduce cluster, you must configure the ACL permission for `has` as administrator.

- yarn.scheduler.capacity.${queue-name}.acl_submit_applications

  - Set the user or group that can submit jobs to this queue.
  - If ${queue-name} is the queue name, multi-level queues are supported. Note that ACL is inherited in multi-level queues. For example:

    ```
    # queue - name = root
      < property >
          < name > yarn . scheduler . capacity . root . acl_submit
     _applicati  ons </ name >
          < value > </ value > #  Space  means  no  one  can
     submit  jobs  to  the  root  queue
      </ property >
     # queue - name = root . testqueue
     < property >
       < name > yarn . scheduler . capacity . root . testqueue .
     acl_submit  _applicati  ons </ name >
          < value > test  testgrp </ value > # testqueue  only
       allows  the  test  user  and  testgrp  group  to
     submit  jobs
    ```

```
  </ property >
```

- yarn.scheduler.capacity.${queue-name}.acl_administer_queue
  - Set some users or groups to manage the queue, such as killing jobs in the queue.
  - Multi-level queue names are supported. Note that ACL is inherited in multi-level queues.

```
# queue - name = root
  < property >
      < name > yarn . scheduler . capacity . root . acl_admini
 ster_queue </ name >
      < value > </ value >
  </ property >
 # queue - name = root . testqueue
 < property >
   < name > yarn . scheduler . capacity . root . testqueue .
 acl_admini  ster_queue </ name >
      < value > test   testgrp </ value >
  </ property >
```

· Restart the YARN service
  - Kerberos secure clusters have ACL enabled by default. You can configure the relevant ACL permissions for queues as required.
  - For non-Kerberos secure clusters, enable ACL and configure the permission control for queues in accordance with the preceding instructions. Then restart the YARN service.

· Configuration example
  - yarn-site.xml

```
< property >
        < name > yarn . acl . enable </ name >
        < value > true </ value >
 </ property >
< property >
        < name > yarn . admin . acl </ name >
        < value > has </ value >
 </ property >
```

  - capacity-scheduler.xml
  - Default queue: Disables the default queue and does not allow users to submit jobs or manage the queue.
  - Q1 queue: Only allows the test user to submit jobs and manage the queue (such as killing jobs).
  - Q2 queue: Only allows the foo user to submit jobs and manage the queue.

```
< configurat  ion >
```

```
    < property >
        < name > yarn . scheduler . capacity . maximum - applicatio
ns </ name >
        < value > 10000 </ value >
        < descriptio  n > Maximum   number   of   applicatio  ns
that   can   be   pending   and   running .</ descriptio  n >
    </ property >
    < property >
        < name > yarn . scheduler . capacity . maximum - am -
resource - percent </ name >
        < value > 0 . 25 </ value >
        < descriptio  n > Maximum   percent   of   resources   in
the   cluster   which   can   be   used   to   run   applicatio  n
  masters   i . e .
             controls   number   of   concurrent   running
applicatio  ns .
        </ descriptio  n >
    </ property >
    < property >
        < name > yarn . scheduler . capacity . resource - calculator
</ name >
        < value > org . apache . hadoop . yarn . util . resource .
DefaultRes  ourceCalcu  lator </ value >
    </ property >
    < property >
        < name > yarn . scheduler . capacity . root . queues </ name
>
        < value > default , q1 , q2 </ value >
        <! -- 3   queues ->
        < descriptio  n > The   queues   at   the   this   level  (
root   is   the   root   queue ).</ descriptio  n >
    </ property >
    < property >
        < name > yarn . scheduler . capacity . root . default .
capacity </ name >
        < value > 0 </ value >
        < descriptio  n > Default   queue   target   capacity .</
descriptio  n >
    </ property >
    < property >
        < name > yarn . scheduler . capacity . root . default . user
- limit - factor </ name >
        < value > 1 </ value >
        < descriptio  n > Default   queue   user   limit   a
percentage   from   0 . 0   to   1 . 0 .</ descriptio  n >
    </ property >
    < property >
        < name > yarn . scheduler . capacity . root . default .
maximum - capacity </ name >
        < value > 100 </ value >
        < descriptio  n > The   maximum   capacity   of   the
default   queue .</ descriptio  n >
    </ property >
    < property >
        < name > yarn . scheduler . capacity . root . default .
state </ name >
        < value > STOPPED </ value >
        <! --  Status   of   the   default   queue   is   set   as
STOPPED -->
        < descriptio  n > The   state   of   the   default
queue .  State   can   be   one   of   RUNNING   or   STOPPED .</
descriptio  n >
    </ property >
    < property >
```

```
        < name > yarn . scheduler . capacity . root . default .
acl_submit  _applicati  ons </ name >
        < value > </ value >
        <! --  The   default   queue   does   not   allow   job
submission -->
        < descriptio  n > The   ACL   of   who   can   submit   jobs
  to   the   default   queue .</ descriptio  n >
    </ property >
    < property >
        < name > yarn . scheduler . capacity . root . default .
acl_admini  ster_queue </ name >
        < value > </ value >
        <! --  Prevent   users / groups   to   manage   the
default   queue -->
        < descriptio  n > The   ACL   of   who   can   administer
jobs   on   the   default   queue .</ descriptio  n >
    </ property >
    < property >
        < name > yarn . scheduler . capacity . node - locality -
delay </ name >
        < value > 40 </ value >
    </ property >
    < property >
        < name > yarn . scheduler . capacity . queue - mappings </
name >
        < value > u : test : q1 , u : foo : q2 </ value >
        <! --  Queue   mapping ,  automatica  lly   maps   the
test   user   to   Q1   queue -->
        < descriptio  n > A   list   of   mappings   that   will
be   used   to   assign   jobs   to   queues .  The   syntax   for
  this   list   is
             [ u | g ]:[ name ]:[ queue_name ][, next   mapping ]*
Typically   this   list   will   be   used   to   map   users   to
  queues , for
             example ,  u :% user :% user   maps   all   users   to
  queues   with   the   same   name   as   the   user .
        </ descriptio  n >
    </ property >
    < property >
        < name > yarn . scheduler . capacity . queue - mappings -
override . enable </ name >
        < value > true </ value >
        <! --  Whether   or   not   allow   the   above   queue -
mapping   to   overwrite   the   queue   parameters   set   up   by
  the   client -->
        < descriptio  n > If   a   queue   mapping   is   present ,
will   it   override   the   value   specified   by   the   user ?
This   can   be   used
             by   administra  tors   to   place   jobs   in   queues
  that   are   different   than   the   one   specified   by   the
  user .  The   default
             is   false .
        </ descriptio  n >
    </ property >
    < property >
        < name > yarn . scheduler . capacity . root . acl_submit
_applicati  ons </ name >
        < value > </ value >
        <! --  ACL   inheritanc  e ,  the   parent   queue   must
have   the   admin   permission  s -->
        < descriptio  n >
             The   ACL   of   who   can   submit   jobs   to   the
root   queue .
        </ descriptio  n >
```

```
    </ property >
    < property >
        < name > yarn . scheduler . capacity . root . q1 .
acl_submit  _applicati  ons </ name >
        < value > test </ value >
        <! --  q1   only   allows   the   test   user   to   submit
  jobs -->
    </ property >
    < property >
        < name > yarn . scheduler . capacity . root . q2 .
acl_submit  _applicati  ons </ name >
        < value > foo </ value >
        <! --  q2   only   allows   the   foo   user   to   submit
jobs -->
    </ property >
    < property >
        < name > yarn . scheduler . capacity . root . q1 . maximum -
capacity </ name >
        < value > 100 </ value >
    </ property >
    < property >
        < name > yarn . scheduler . capacity . root . q2 . maximum -
capacity </ name >
        < value > 100 </ value >
    </ property >
    < property >
        < name > yarn . scheduler . capacity . root . q1 . capacity
</ name >
        < value > 50 </ value >
    </ property >
    < property >
        < name > yarn . scheduler . capacity . root . q2 . capacity
</ name >
        < value > 50 </ value >
    </ property >
    < property >
        < name > yarn . scheduler . capacity . root . acl_admini
ster_queue </ name >
        < value > </ value >
        <! --  ACL   inheritanc  e ,  the   parent   queue   must
have   the   admin   permission  s -->
    </ property >
    < property >
        < name > yarn . scheduler . capacity . root . q1 .
acl_admini  ster_queue </ name >
        < value > test </ value >
        <! --  q1   only   allow   the   test   user   to   manage
the   queue ,  such   as   killing   the   jobs -->
    </ property >
    < property >
        < name > yarn . scheduler . capacity . root . q2 .
acl_admini  ster_queue </ name >
        < value > foo </ value >
        <! --  q2   only   allow   the   foo   user   to   manage
the   queue ,  such   as   killing   the   jobs -->
    </ property >
    < property >
        < name > yarn . scheduler . capacity . root . q1 . state </
name >
        < value > RUNNING </ value >
    </ property >
    < property >
        < name > yarn . scheduler . capacity . root . q2 . state </
name >
```

```
        < value > RUNNING </ value >
    </ property >
</ configurat  ion >
```

12.3 Hive authorization

Hive has two authorization modes: one based on storage and the other based on SQL standards. For more information, see Hive's *Authorization guide*.

Note:
Both means of authorization can be configured at the same time without conflict.

Storage based authorization (for Hive metastore)

If a user in a cluster has direct access to data in Hive through an HDFS or Hive client, a permission control must be performed on Hive data in HDFS. By doing so, operation permissions related to Hive SQL can be controlled.

For more information, see Hive's *Storage Based Authorization* guide.

Add configuration

In the cluster Configuration Management page, click Hive > Configuration > hive-site.xml > Add Custom Configuration.

```
< property >
< name > hive . metastore . pre . event . listeners </ name >
    < value > org . apache . hadoop . hive . ql . security .
 authorizat  ion . Authorizat  ionPreEven  tListener </ value >
</ property >
< property >
< name > hive . security . metastore . authorizat  ion . manager </
 name >
    < value > org . apache . hadoop . hive . ql . security .
 authorizat  ion . StorageBas  edAuthoriz  ationProvi  der </ value >
</ property >
< property >
< name > hive . security . metastore . authentica  tor . manager </
 name >
    < value > org . apache . hadoop . hive . ql . security .
 HadoopDefa  ultMetasto  reAuthenti  cator </ value >
</ property >
```

Restart Hive metastore

Restart the Hive metastore in the cluster's Configuration Management page.

HDFS permission control

For Kerberos security clusters in E-MapReduce, HDFS permissions for the Hive warehouse are set.

For non-Kerberos security clusters, you must complete the following steps to set the basic HDFS permission:

- Enable HDFS permissions
- Configure permissions for the Hive warehouse

```
hadoop fs - chmod 1771 / user / hive / warehouse
It can be set as follows , in which 1 denotes
stick bit ( i . e . cannot delete files / folders created
  by others )
hadoop fs - chmod 1777 / user / hive / warehouse
```

With the basic permission set, users and user groups can create, read, and write tables as usual by authorizing the folder warehouse.

```
sudo su has
    # Grant rwx permission of folder warehouse to user
  test
     hadoop fs - setfacl - m user : test : rwx / user / hive
/ warehouse
    # Grant rwx permission of folder warehouse to user
  hivegrp
     hadoo fs - setfacl - m group : hivegrp : rwx / user /
hive / warehouse
```

With HDFS authorized, users and user groups can create, read, and write tables as usual. Data in Hive tables that is created by different users in HDFS can only be accessed by the users themselves.

Verification

- The test user creates a table testtbl.

```
hive > create table testtbl ( a string );
FAILED : Execution Error , return code 1 from org .
apache . hadoop . hive . ql . exec . DDLTask . MetaExcept ion
( message : Got exception : org . apache . hadoop . security .
AccessCont rolExcepti on Permission denied : user = test ,
access = WRITE , inode ="/ user / hive / warehouse / testtbl ":
hadoop : hadoop : drwxrwx -- t
at org . apache . hadoop . hdfs . server . namenode . FSPermissi
onChecker . check ( FSPermissi onChecker . java : 320 )
at org . apache . hadoop . hdfs . server . namenode . FSPermissi
onChecker . check ( FSPermissi onChecker . java : 292 )
```

An error occurs due to the lack of permissions. Permissions should be granted to the test user.

```
# Switch from root account to has account
 su has
# Add ACL and grant rwx permission s of the
 directory warehouse to the account test .
```

```
 hadoop  fs  - setfacl  - m  user : test : rwx  / user / hive /
warehouse
```

The test account recreates the database successfully.

```
hive >  create  table  testtbl ( a  string );
OK
Time  taken :  1 . 371  seconds
# View  the  directory  of  testtbl  in  HDFS .  From  the
permission  s  it  can  be  seen  that  only  the  groups
  test  and  hadoop  can  read  data  from  the  table
created  by  the  user  test ,  while  other  users  have
no  permission  s
hadoop  fs  - ls  / user / hive / warehouse
drwxr - x ---  -  test  hadoop          0  2017 - 11 - 25  14
: 51  / user / hive / warehouse / testtbl
# Insert  a  record
hive > insert  into  table  testtbl  select  " hz "
```

- **User foo accesses the table testtbl.**

```
# drop  table
hive >  drop  table  testtbl ;
FAILED :  Execution  Error ,  return  code  1  from  org .
apache . hadoop . hive . ql . exec . DDLTask .  MetaExcept  ion
( message : Permission  denied :  user = foo ,  access = READ ,
inode ="/ user / hive / warehouse / testtbl ": test : hadoop :
drwxr - x ---
    at  org . apache . hadoop . hdfs . server . namenode .
FSPermissi  onChecker . check ( FSPermissi  onChecker . java : 320
)
    at  org . apache . hadoop . hdfs . server . namenode .
FSPermissi  onChecker . checkPermi  ssion ( FSPermissi  onChecker .
java : 219 )
    at  org . apache . hadoop . hdfs . server . namenode .
FSPermissi  onChecker . checkPermi  ssion ( FSPermissi  onChecker .
java : 190 )
# alter  table
hive >  alter  table  testtbl  add  columns ( b  string );
FAILED :  SemanticEx  ception  Unable  to  fetch  table
  testtbl .  java . security . AccessCont  rolExcepti  on :
Permission  denied :  user = foo ,  access = READ ,  inode ="/
user / hive / warehouse / testtbl ": test : hadoop : drwxr - x ---
    at  org . apache . hadoop . hdfs . server . namenode .
FSPermissi  onChecker . check ( FSPermissi  onChecker . java : 320
)
    at  org . apache . hadoop . hdfs . server . namenode .
FSPermissi  onChecker . checkPermi  ssion ( FSPermissi  onChecker .
java : 219 )
    at  org . apache . hadoop . hdfs . server . namenode .
FSPermissi  onChecker . checkPermi  ssion ( FSPermissi  onChecker .
java : 190 )
    at  org . apache . hadoop . hdfs . server . namenode .
FSDirector  y . checkPermi  ssion ( FSDirector  y . java : 1720 )
# select
hive >  select  *  from  testtbl ;
FAILED :  SemanticEx  ception  Unable  to  fetch  table
  testtbl .  java . security . AccessCont  rolExcepti  on :
Permission  denied :  user = foo ,  access = READ ,  inode ="/
user / hive / warehouse / testtbl ": test : hadoop : drwxr - x ---
```

```
    at   org . apache . hadoop . hdfs . server . namenode .
FSPermissi  onChecker . check ( FSPermissi  onChecker . java : 320
)
    at   org . apache . hadoop . hdfs . server . namenode .
FSPermissi  onChecker . checkPermi  ssion ( FSPermissi  onChecker .
java : 219 )
```

Foo cannot perform operations on the table created by the test user. HDFS authorization is needed to grant permissions to foo.

```
 su   has
# Only   read   permission   is   granted ,  and   write
 permission   can   also   be   granted   as   needed  ( for
 example ,  alter )
# Note : - R :  Set   files   in   the   folder   testtbl   to
 readable
 hadoop   fs  - setfacl  - R  - m   user : foo : r - x  / user /
 hive / warehouse / testtbl
# The   table   can   be   selected   successful  ly
 hive >  select  *  from   testtbl ;
 OK
 hz
 Time   taken :  2 . 134   seconds ,  Fetched :  1   row ( s )
```

Note:

You can create a Hive user group, authorize it, and then add new users it.

SQL Standard Based Authorization

- Scenario

  If a cluster user cannot access data in Hive through an HDFS or Hive client, and the only way is to run Hive related commands through `HiveServer  ( beeline , jdbc ,  and   so   on )`, SQL standard based authorization can be used.

  If users can use the Hive shell or similar methods and as long as hive-site.xml in the user's client has not been configured, Hive can still be accessed as usual, even if the following settings are implemented.

  For more information, see Hive's *SQL Standard Based Authorization* guide.

- Add configuration

  - The configuration is provided to HiveServer.
  - In the cluster Configuration Management page, click Hive > Configuration > hive-site.xml > Add Custom Configuration.

    ```
    < property >
    < name > hive . security . authorizat  ion . enabled </ name >
     < value > true </ value >
    </ property >
     < property >
    ```

```
< name > hive . users . in . admin . role </ name >
 < value > hive </ value >
</ property >
 < property >
< name > hive . security . authorizat  ion . createtabl  e . owner
 . grants </ name >
 < value > ALL </ value >
</ property >
```

- Restart HiveServer2

  Restart HHiveServer2 in the cluster Configuration Management page.

- Permission operation commands

  For more information on command operations, click here.

- Verification

  - User foo accesses the test user's table, testtbl, through beeline.

    ```
    2 :  jdbc : hive2 :// emr - header - 1 . cluster - xxx : 10 >
    select  *  from   testtbl ;
    Error :  Error   while   compiling   statement :  FAILED :
    HiveAccess  ControlExc  eption   Permission   denied :  Principal
     [ name = foo ,  type = USER ]  does   not   have   following
    privileges   for   operation   QUERY  [[ SELECT ]  on   Object
    [ type = TABLE_OR_V  IEW ,  name = default . testtbl ]] ( state =
    42000 , code = 40000 )
    ```

  - Grant permissions.

    ```
    Switch   to   account   test   to   grant   select   permission
    to   user   foo
    hive >  grant   select   on   table   testtbl   to   user   foo ;
     OK
     Time   taken :  1 . 205   seconds
    ```

  - Foo can select as usual.

    ```
    0 :  jdbc : hive2 :// emr - header - 1 . cluster - xxxxx : 10 >
    select  *  from   testtbl ;
    INFO   :  OK
    +------------+--+
    |  testtbl . a   |
    +------------+--+
    |  hz          |
    +------------+--+
    1   row   selected  ( 0 . 787   seconds )
    ```

  - Revoke permissions.

    ```
    Switch   to   account   test ,  and   revoke   the   select
    permission   from   user   foo
      hive >  revoke   select   from   user   foo ;
        OK
    ```

```
Time taken : 1 . 094 seconds
```

- Foo cannot select testtbl data.

```
User foo cannot select data from table testtbl .
Error : Error while compiling statement : FAILED :
HiveAccess ControlExc eption Permission denied : Principal
 [ name = foo , type = USER ] does not have following
privileges for operation QUERY [[ SELECT ] on Object
[ type = TABLE_OR_V IEW , name = default . testtbl ]] ( state =
42000 , code = 40000 )
```

12.4 HBase authorization

Any account can perform any operation on an HBase cluster without being authorized, including disabling and dropping tables and performing major compaction.

Note:

For clusters that do not have Kerberos authentication, users can forge identities to access the cluster service, even when HBase authorization is enabled. Therefore, we recommend that you create a high-security cluster (for example, supporting Kerberos) as detailed in the *Introduction to Kerberos*.

Add configuration

In the Configuration Management page, choose HBase > Configuration > hbase-site > Custom Configuration in the HBase cluster.

Add the following parameters:

```
< property >
    < name > hbase . security . authorizat ion </ name >
    < value > true </ value >
</ property >
< property >
    < name > hbase . coprocesso r . master . classes </ name >
    < value > org . apache . hadoop . hbase . security . access .
 AccessCont roller </ value >
</ property >
< property >
    < name > hbase . coprocesso r . region . classes </ name >
 < value > org . apache . hadoop . hbase . security . token .
 TokenProvi der , org . apache . hadoop . hbase . security . access .
 AccessCont roller </ value >
</ property >
< property >
  < name > hbase . coprocesso r . regionserv er . classes </ name >
  < value > org . apache . hadoop . hbase . security . access .
 AccessCont roller , org . apache . hadoop . hbase . security . token
 . TokenProvi der </ value >
```

```
</ property >
```

Restart the HBase cluster

In the HBase cluster's Configuration Management page, click HBase > Operations > RESTART All Components.

Authorization (ACL)

- Basic concepts

  In HBase, authorization consists of three elements: the granting of operational permissions for a certain scope of resources to a certain entity.

  - Resources in a certain scope
    - Superuser

      A superuser can perform any operations. The account that runs the HBase service is the superuser by default. You can also add superusers by configurin g the value of hbase.superuser in hbase-site.xml.
    - Global

      Global Scope has administrator permissions for all tables in the cluster.
    - Namespace

      This has permission control in Namespace Scope.
    - Table

      This has permission control in Table Scope.
    - ColumnFamily

      This has permission control in ColumnFamily Scope.
    - Cell

      This has permission control in Cell Scope.
  - Operational permissions
    - Read (R)

      Read data from resources in a certain scope.
    - Write (W)

      Write data to resources in a certain scope.
    - Execute (X)

      Execute co-processor in a certain scope.
    - Create (C)

      Create or delete a table in a certain scope.
    - Admin (A)

Perform cluster-related operations in a certain scope, such as balance or assign.

- Entity
  - User

    Authorize a user.
  - Group

    Authorize a user group.

· Authorization command

- grant

```
grant  < user > < permission  s > [<@ namespace > [< table > [<
column   family > [< column   qualifier >]]]
```

 Note:

- The authorization methods for users and groups are the same. The prefix @ needs to be added for groups.

```
grant  ' test ',' R ',' tbl1 '   # grant   the   read
permission   of   the   table   tb11   to   the   user   test
.
  grant  '@ test ',' R ',' tbl1 '   # grant   the   read
permission   of   the   table   tb11   to   the   user   group
  test .
```

- The prefix @ needs to be added for namespace.

```
grant  ' test  ' C ','@ ns_1 '  # grant   the   create
permission   of   the   namespace  @ ns_1   to   the   user
test .
```

- revoke
- user_permissions (view permissions)

12.5 Kafka authorization

If Kafka authentication (for example, Kerberos authentication or another simple authorization based on a user name and password) is disabled, users can access services with forged identities, even if Kafka authorization is enabled. Therefore, we

recommend that you create a high-security Kafka cluster. For more information, see *Introduction to Kerberos*.

 Note:
The permission configurations detailed in this section are for high-security E-MapReduce clusters only (Kafka is started in Kerberos).

Add configurations

1. On the Cluster Management page, click View Details next to the Kafka cluster.
2. In the navigation pane on the left, click the Clusters and Services tab, and click Kafka in the service list.
3. At the top of the page, click the Configuration tab.
4. In the upper-right corner of the Service Configuration list, click Custom Configuration and add the following parameters:

Key	Value	Description
authorizer.class.name	kafka.security.auth.SimpleAclAuthorizer	N/A
super.users	User:kafka	User:kafka is required. Other users can be added and separated by semicolons (;).

 Note:
zookeeper.set.acl is used to set the permissions for Kafka to operate data in ZooKeeper. It is already set to true in the E-MapReduce cluster, so you do not need to add this configuration here. With the configuration set to true, only users named Kafka who have passed the Kerberos authentication can run the kafka-topics.sh command in the Kerberos environment. Kafka-topics.sh can read, write, and modify data in ZooKeeper.

Restart a Kafka cluster

1. On the Cluster Management page, click View Details next to the Kafka cluster you want to operate in the Operation column.
2. In the navigation pane on the left, click the Clusters and Services tab, and click Actions to the right of Kafka on the service list.

3. In the drop-down menu, select RESTART All Components. Enter the record information and click OK.

Authorization (ACL)

- Basic concepts

  Definition in official Kafka documentation:

  ```
  Kafka ACLs are defined in the general format of "
  Principal P is [ Allowed / Denied ] Operation O From
  Host H On Resource R "
  ```

  This indicates that the ACL process relates to Principal, Allowed/Denied, Operation Host, and Resource.

  - Principal: username

Security protocol	Value
PLAINTEXT	ANONYMOUS
SSL	ANONYMOUS
SASL_PLAINTEXT	If the mechanism is PLAIN, the user name is specified by client_jaas.conf. If the mechanism is GSSAPI, the user name is principal specified by client_jaas.conf.
SASL_SSL	

  - Allowed/Denied
  - Operation: Operations include Read, Write, Create, DeleteAlter, Describe, ClusterAction, AlterConfigs, DescribeConfigs, IdempotentWrite, and All.
  - Host: The target machine.
  - Resource: Resource objects, including Topic, Group, Cluster, and Transactio nalId.

  For detailed mapping relationships between operations and resources, such as the supporting relationships between resources and the authorization of operations, see *KIP-11 - Authorization Interface*.

- Authorization command

  Perform authorization using the kafka-acls.sh script (/usr/lib/kafka-current/bin/kafka-acls.sh). For more information about how to use this script to authorize Kafka, run the `kafka - acls . sh -- help` command.

Procedure

Complete the following operations on the master node of the high-security Kafka cluster you created in E-MapReduce.

1. Create a user named test.

```
useradd test
```

2. Create a topic.

   zookeeper.set.acl is set to true, and kafka-topics.sh must be run under a Kafka account. The Kafka account must pass Kerberos authentication.

```
# The Kerberos authentica tion informatio n related to
 the kafka account is set in kafka_clie nt_jaas . conf
 .
 export KAFKA_HEAP _OPTS ="- Djava . security . auth . login .
 config =/ etc / ecm / kafka - conf / kafka_clie nt_jaas . conf "
#  Change the ZooKeeper address to the actual address
 ( run hostnamed to acquire ) of your Kafka cluster .
 kafka - topics . sh -- create -- zookeeper emr - header -
 1 : 2181 / kafka - 1 . 0 . 0 -- replicatio n - factor 3 --
 partitions 1 -- topic test
```

3. Run kafka-console-producer.sh with the test user.

   a. Create a keytab file for the test user to authenticate ZooKeeper and Kafka.

```
su root
sh / usr / lib / has - current / bin / hadmin - local . sh /
etc / ecm / has - conf - k / etc / ecm / has - conf / admin .
keytab
HadminLoca lTool . local : # Press Enter to display the
  usage instructio ns on some commands .
HadminLoca lTool . local : addprinc # Enter a command
and press Enter to display the usage instructio ns
  on the command .
HadminLoca lTool . local : addprinc - pw 123456 test #
Add a principal for the test user and set the
password to 123456 .
HadminLoca lTool . local : ktadd - k / home / test / test .
keytab test # Export the keytab file for later
use .
```

   b. Add a kafka_client_test.conf file.

   Put the file in */ home / test / kafka_clie nt_test . conf* . The content of the file is as follows:

```
KafkaClien t {
com . sun . security . auth . module . Krb5LoginM odule
required
useKeyTab = true
storeKey = true
serviceNam e =" kafka "
keyTab ="/ home / test / test . keytab "
```

```
 principal =" test ";
};
//  Zookeeper   client   authentica  tion
 Client  {
 com . sun . security . auth . module . Krb5LoginM  odule
 required
 useKeyTab = true
 useTicketC  ache = false
 serviceNam  e =" zookeeper "
 keyTab ="/ home / test / test . keytab "
 principal =" test ";
};
```

c. Add producer.conf.

Put the file in `/ home / test / producer . conf` . The content of the file is as follows:

```
security . protocol = SASL_PLAIN  TEXT
sasl . mechanism = GSSAPI
```

d. Run kafka-console-producer.sh.

```
su   test
export   KAFKA_HEAP  _OPTS ="- Djava . security . auth . login .
config =/ home / test / kafka_clie  nt_test . conf "
kafka - console - producer . sh  -- producer . config  / home /
test / producer . conf  -- topic   test  -- broker - list   emr -
worker - 1 : 9092
```

Because no ACL is set, an error is reported after the preceding command is run:

```
org . apache . kafka . common . errors . TopicAutho  rizationEx
ception :  Not   authorized   to   access   topics : [ test ]
```

e. Set an ACL.

Similarly, the `kafka - acls . sh` command must be run under the Kafka account.

```
su   kafka
export   KAFKA_OPTS ="- Djava . security . auth . login . config
=/ etc / ecm / kafka - conf / kafka_clie  nt_jaas . conf "
kafka - acls . sh  -- authorizer - properties   zookeeper .
connect = emr - header - 1 : 2181 / kafka - 1 . 0 . 0  -- add  --
allow - principal   User : test  -- operation   Write  -- topic
test
```

f. Run kafka-console-producer.sh again.

```
su   test
export   KAFKA_HEAP  _OPTS ="- Djava . security . auth . login .
config =/ home / test / kafka_clie  nt_test . conf "
```

```
kafka - console - producer . sh  -- producer . config  / home /
test / producer . conf  -- topic   test  -- broker - list   emr -
worker - 1 : 9092
```

Normal output is as follows:

```
[ 2018 - 02 - 28   22 : 25 : 36 , 178 ]  INFO   Kafka   commitId
 :  aaa7af6d4a  11b29d  ( org . apache . kafka . common . utils .
AppInfoPar  ser )
> alibaba
> E - MapReduce
>
```

4. Run `kafka - console - consumer . sh` with the test user.

   After kafka-console-producer.sh is successfully run and data is written to the topic, you can run `kafka - console - consumer . sh` to perform a consumption test.

   a. Add consumer.conf.

      Put the file in */ home / test / consumer . conf* . The content of the file is as follows:

```
security . protocol = SASL_PLAIN  TEXT
sasl . mechanism = GSSAPI
```

   b. Run kafka-console-consumer.sh.

```
su   test
#  Kafka_clie  nt_test . conf   is   used   in   the   same   way
   as   kafka - console - producer . sh .
export   KAFKA_HEAP  _OPTS ="- Djava . security . auth . login .
config =/ home / test / kafka_clie  nt_test . conf "
kafka - console - consumer . sh  -- consumer . config   consumer
. conf   -- topic   test  -- bootstrap - server   emr - worker -
1 : 9092  -- group   test - group   -- from - beginning
```

      Because no permissions are set, an error is reported:

```
org . apache . kafka . common . errors . GroupAutho  rizationEx
ception :  Not   authorized   to   access   group :  test - group
```

   c. Set an ACL.

```
su   kafka
export   KAFKA_HEAP  _OPTS ="- Djava . security . auth . login .
config =/ etc / ecm / kafka - conf / kafka_clie  nt_jaas . conf
"
#  test - group   permission
kafka - acls . sh  -- authorizer - properties   zookeeper .
connect = emr - header - 1 : 2181 / kafka - 1 . 0 . 0  -- add  --
allow - principal   User : test  -- operation   Read  -- group
test - group
#  topic   permission
kafka - acls . sh  -- authorizer - properties   zookeeper .
connect = emr - header - 1 : 2181 / kafka - 1 . 0 . 0  -- add  --
```

```
allow - principal   User : test  -- operation   Read  -- topic
test
```

d. Run `kafka - console - consumer . sh` again.

```
su   test
#  Kafka_clie  nt_test . conf   is   used   in   the   same   way
   as   kafka - console - producer . sh .
export   KAFKA_HEAP  _OPTS ="- Djava . security . auth . login .
config =/ home / test / kafka_clie  nt_test . conf "
kafka - console - consumer . sh  -- consumer . config   consumer
. conf   -- topic   test  -- bootstrap - server   emr - worker -
1 : 9092  -- group   test - group   -- from - beginning
```

Normal output is as follows:

```
alibaba
E - MapReduce
```

12.6 Ranger

12.6.1 Introduction to Ranger

Apache Ranger provides a centralized framework for permission management, implementing fine-grained access control for components in the Hadoop ecosystem, such as HDFS, Hive, YARN, Kafka, Storm, and Solr. It also provides a UI that allows administrators to perform operations more conveniently.

Create a cluster

Select the Ranger service when you create a cluster in E-MapReduce 2.9.2/3.9.0 or later on the E-MapReduce console.

If an E-MapReduce cluster 2.9.2/3.9.0 or later has been created without Ranger, you can go to the Clusters and Services page to add it.

Note:

- When Ranger is enabled, there is no impact or limitation on the application until the security control policy is set.
- The user policy set in Ranger is the cluster Hadoop account.

Ranger UI

After installing Ranger on the cluster, click Manage in the Actions column, and then click Access Links and Ports in the navigation pane on the left. You can then access the Ranger UI by clicking on the link, as shown in the following figure.

Enter the Ranger UI. The default user name and password are both admin, as shown in the following figure.

Modify the password

After you first log on, the administrator needs to modify the password of the admin account, as shown in the following figure.

After you change the admin password, in the admin drop-down list in the upper-right corner, click Log Out. You can then log on again with the new password.

Integrate Ranger into other services

After completing the preceding steps, you can integrate Ranger into the services in the cluster to control the relevant permissions. For more information, see the following:

- *Integrate Ranger into HDFS*
- *Integrate Ranger into Hive*
- *Integrate Ranger into HBase*

12.6.2 Integrate Ranger into HDFS

Procedure

This section describes the step-by-step process for integrating Ranger into HDFS.

- Enable the HDFS plug-in

  1. On the Cluster Management page, click Manage next to the cluster you want to operate in the Actions column.
  2. Click Ranger in the service list to enter the Ranger Management page.
  3. On the Ranger Configuration page, click the Actions drop-down menu in the upper-right corner, select Enable HDFS PLUGIN, and click OK.

  4. Enter the record information in the prompt box and click OK.

     You can check the progress by clicking View Operation Logs in the upper-right corner of the page.

- Restart NameNode

  After enabling the HDFS plug-in, you need to restart NameNode. To do so, complete the following steps:

  1. In the Ranger Management page, click the Ranger drop-down menu in the upper -left corner, and select HDFS.
  2. Click Actions in the upper-right corner of the page and select RESTART NameNode.
  3. You can check the progress by clicking View Operation Logs in the upper-right corner of the page.

- Add the HDFS service to Ranger UI

  For more information about how to access the Ranger UI, see *Introduction to Ranger*.

  Add the HDFS service.

  - Standard cluster

    To check the mode of the cluster you created, go to the Cluster Overview page. If your cluster is in standard mode, configure it as follows:

  - High-security-mode cluster

    To check the mode of the cluster you created, go to the Cluster Overview page. If your cluster is in high-security mode, configure it as follows:

Permission configuration

After integrating Ranger into HDFS, you can set permissions, such as granting the test user the write or execute permission for `/ user / foo .`

In the preceding figure, click emr-hdfs to enter the policy configuration page.

Permissions are granted to the test user. They can now access the HDFS path of `/ user / foo .`

Note:
The policy takes effect one minute after it is added.

12.6.3 Integrate Ranger into Hive

Procedure

This section describes the step-by-step process for integrating Ranger into Hive.

- Hive access model

  You can access Hive data in three ways: HiveServer2, Hive client, and HDFS.

  - HiveServer2

    ■ Mode: Use the Beeline client or the JDBC code to run the related Hive script through HiveServer2.

    ■ Permission settings:

    Hive's *SQL Standard Based Authorization* is used to control the permissions of HiveServer2.

    Hive's table- and column-level permission control in Ranger is also used for HiveServer2. However, if you are still able to access Hive data though a Hive client or HDFS, table- or column-level permission control is insufficient and further control is required.

  - Hive client

    ■ Mode: Access from a Hive client.

    ■ Permission settings:

    The Hive client requests the metastore to perform DDL operations, such as altering tables, adding columns, and reading and processing data in HDFS, by submitting MapReduce jobs.

    Hive's *Storage Based Authorization* is used to control the permissions of Hive clients. It determines whether a user can perform DDL and DML operations based on the read and write permissions of the HDFS path where the table involved in SQL is located, such as `ALTER TABLE test ADD COLUMNS ( b STRING )`.

    In Ranger, you can control the permissions of the HDFS path in Hive tables . This, in combination with the Hive metastore which is configured with storage-based authorization, enables you to implement permission control over the access of the Hive client.

     Note:

The DDL operation permissions of a Hive client are actually controlled by the underlying HDFS permissions. If you have HDFS permissions, you also have DDL permissions for tables, such as dropping and altering tables.

- HDFS
  - Mode: HDFS client and code.
  - Permission settings:

    To have direct access to HDFS, you need to add permission control for HDFS on the underlying HDFS data of the Hive tables.

    You can use Ranger to perform permission control for the underlying HDFS path of Hive tables.

· Enable the Hive plug-in

1. On the Cluster Management page, click Manage next to the cluster you want to operate in the Actions column.
2. Click Ranger in the service list to enter the Ranger Management page.
3. On the Ranger Configuration page, click the Actions drop-down menu in the upper-right corner, select Enable Hive PLUGIN, and click OK.

4. Enter the record information in the prompt box and clickOK.

   You can check the progress by clicking View Operation Logs in the upper-right corner of the page.

 Note:
After you enable the Hive plug-in and restart Hive, HiveServer2 and Hive client scenarios are configured accordingly. For more information about HDFS permissions, see *Integrate Ranger into HDFS*.

- Restart Hive

  After enabling the Hive plug-in, you need to restart Hive. To do so, complete the following steps:

  1. In the Ranger Management page, click the Ranger drop-down menu in the upper-left corner, and select Hive.
  2. Click Actions in the upper-right corner, select RESTART All Components from the drop-down menu, and click OK.
  3. You can check the progress by clicking View Operation Logs in the upper-right corner of the page.

- Add the Hive service to the Ranger UI

  For more information about how to access the Ranger UI page, see *Introduction to Ranger*.

  Add the Hive service.

  - Instructions

    Enter a fixed value for the following configuration items:

Name	Value
Service Name	emr-hive
jdbc.driverClassName	org.apache.hive.jdbc.HiveDriver

  - Enter a variable value for the following configuration items:

Name	Value
jdbc.url	Standard cluster: jdbc:hive2://emr-header-1:10000/ high-High-security cluster: jdbc:hive2://${master1_fullhost}:10000/;principal=hive/${master1_fullhost}@EMR.$id.COM

Name	Value
policy.download.auth.users	Standard cluster: hadoop High-security cluster: hive

${master1_fullhost} is the long domain name of master1. To obtain this name, log on to master1 and run the `hostname` command. The number in ${master1_fullhost} is the value of $id.

Permission configuration

After integrating Ranger into Hive, you can set permissions, such as granting user foo the Select permission for column A in the testdb.test table.

In the preceding figure, click emr-hive to enter the policy configuration page.

Permissions are granted to user foo. They can now access the testdb.test table.

> Note:
> The policy takes effect one minute after it is added.

12.6.4 Data masking in Hive

Ranger supports data masking in Hive by masking return values of SELECT statements to hide sensitive information from users.

> Note:
> This feature only supports scenarios involving HiveServer2, such as using Beeline, JDBC, or Hue to run SELECT statements. HiveClient-based scenarios are not supported, such as hive -e 'select xxxx'.

This topic describes how to use this feature in E-MapReduce.

Configure the Hive plug-in for Ranger

For more information, see *Hive configurations*.

Configure Data Mask Policy

You can mask Hive data accessed by users on the `emr - hive` service page in the Ranger UI.

- Ranger supports a variety of masking types, such as show the first four characters, show the last four characters, and Hash masks.
- A mask policy does not support wildcards. For example, you cannot use an asterisk (*) to replace columns or tables in a mask policy.
- Each mask policy is corresponding to one column. You need to configure mask policies for each column.

Perform the following steps to configure a mask policy.

Save your mask policy.

Mask test data

- Scenario:

  User test selects column a from the testdb1.testtbl table to display only the first four characters of each value.
- Procedure:

  1. Configure a mask policy

     The last figure in the previous section shows the mask policy for this scenario. " show first 4" is selected as the masking type.
  2. Verify data masking

     User test uses Beeline to connect to HiveServer2 and runs the `select a from testdb1.testtbl` statement.

     As shown above, after user test runs the SELECT statement, only the first four characters of values of column a are shown. The rest characters are replaced by `x` for data masking.