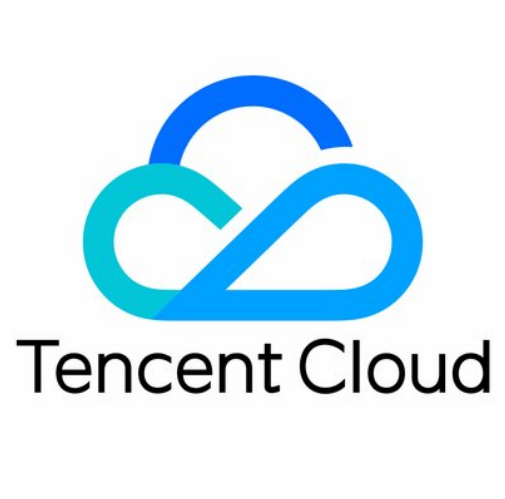
Elastic MapReduce

Practical Tutorial

Product Documentation



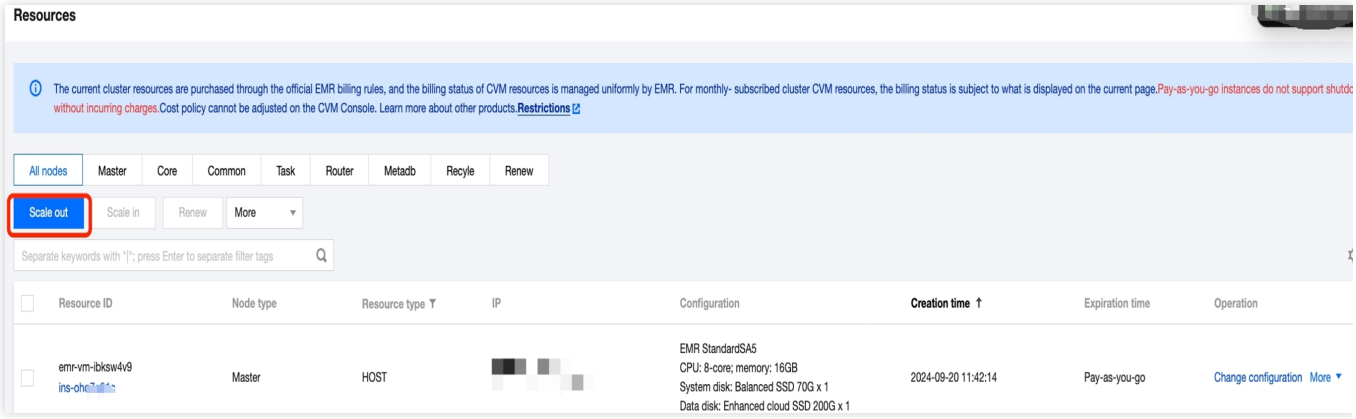
Contents

Practical Tutorial
Practice of EMR on CVM Ops
Migration of HiveServer2 and MetaStore to Router

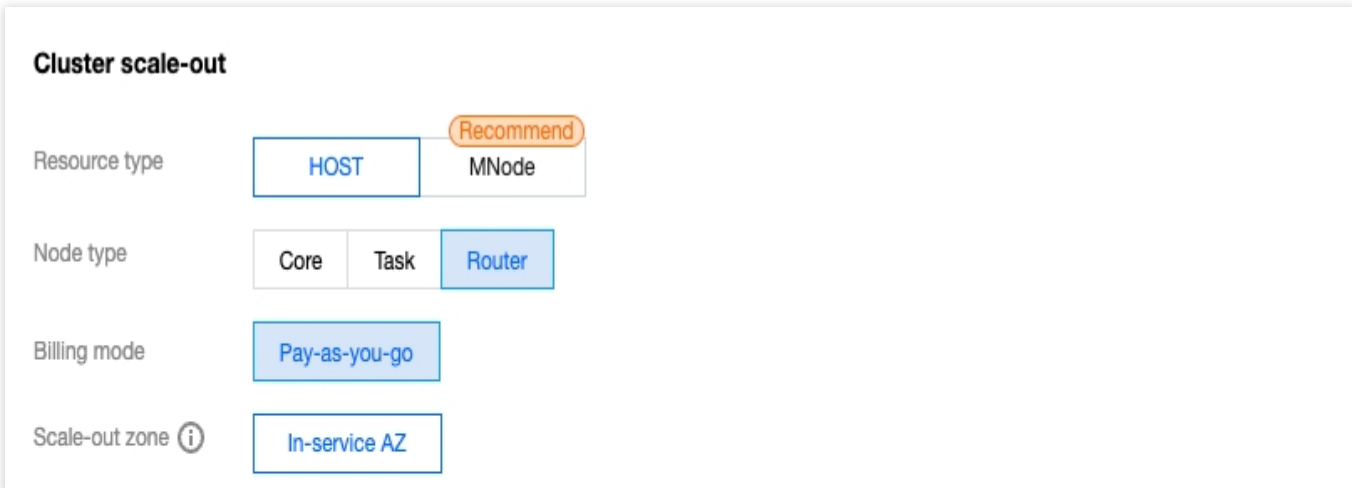
Last updated：2025-01-03 15:05:10

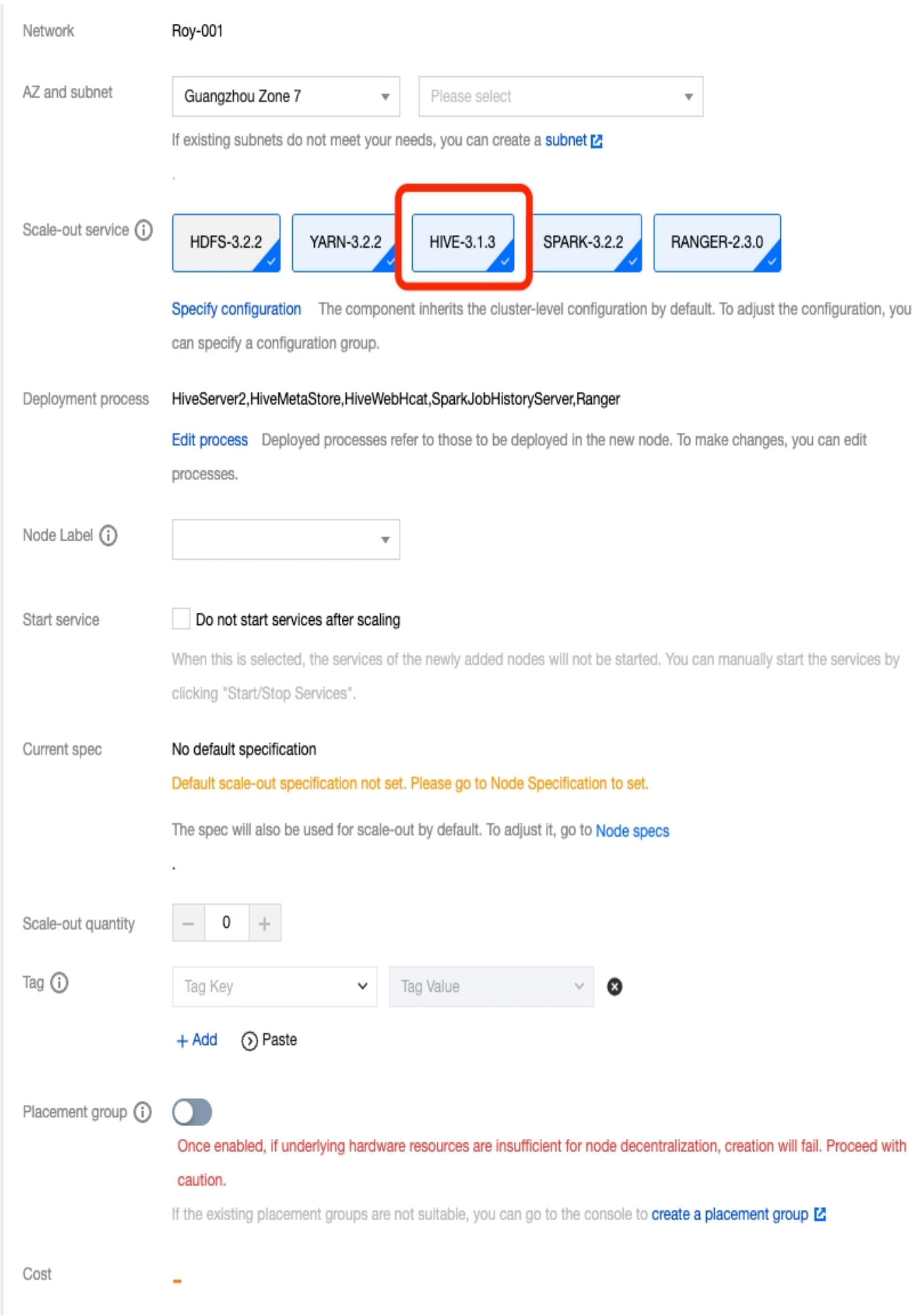
Adding Router Node

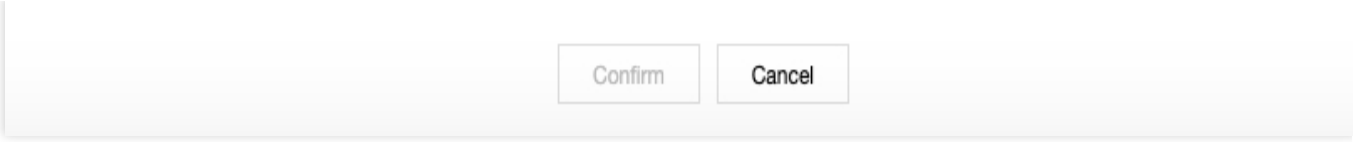
Log in to the EMR console, select the corresponding cluster in the **cluster list**, and click the **ID/name** to enter the cluster details page. On the cluster details page, select **Cluster Resources > Resource Management** to enter the resource management page, then click Expand to navigate to the cluster **Expand** page.



On the cluster expansion page, select the **node type** for scale-out as **Router** and the service to **services to expand** as Hive. You can adjust other options according to your needs.





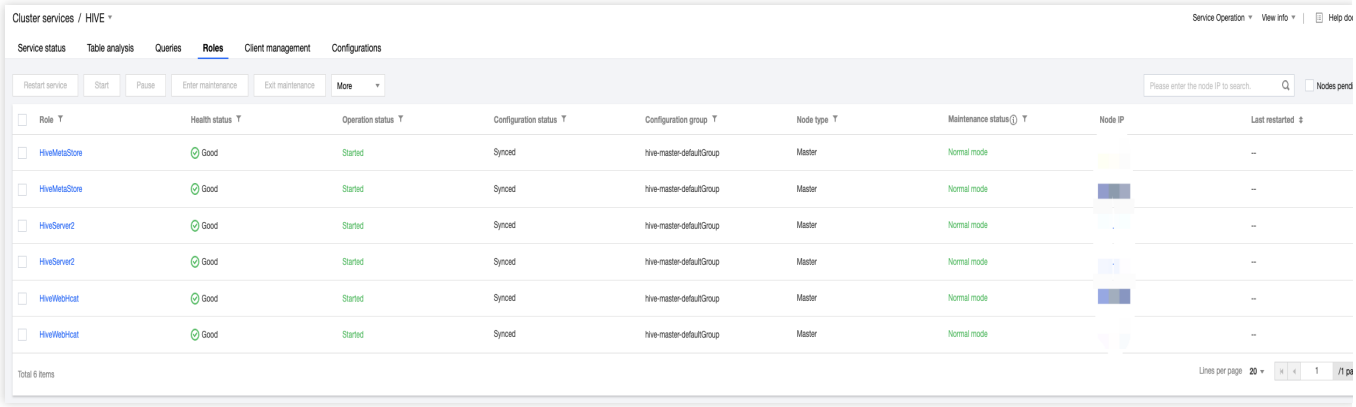


Migrating HiveServer2 and MetaStore

Go to the EMR console, use the **Cluster Services** to access the configuration management feature of the Hive service, and modify the following parameters in the `hive-site.xml` configuration file of the Router node:

Parameter	Value	Description
hive.metastore.uris	thrift://${router-ip}:7004	Modify the Hive Metastore address information, directing the Hive metadata storage service address to the added Router node. ${router-ip} refers to the private IP address of the Router node where the MetaStore is located.

After issuing and saving the configuration, go to **Cluster Services** and select **Operation > Role Management** for the Hive component. Pause all Hive processes on the Master node and restart the Hive processes on the Router node.



Test HiveServer2 on the Router node. If you can successfully connect and query the existing tables, the migration is successful.

```
beeline -u jdbc:hive2://${router-ip}:7001 -n hadoop -p hadoop
show tables
```

Modifying Knox Proxy Address

After the migration of HiveServer2, you need to log in to the Master node and modify the Knox configuration file to proxy Hive and HiveUI to the Router node's Hive component.

```
vim /usr/local/service/knox/conf/topologies/emr.xml //Modify HIVE and HIVEUI.
<service>
 <role>HIVE</role>
 <url>http://Router-ip:7003</url>
 <param>
     <name>replayBufferSize</name>
     <value>8</value>
 </param>
</service>
<service>
 <role>HIVEUI</role>
 <url>http://Router-ip:7003</url>
</service>
```

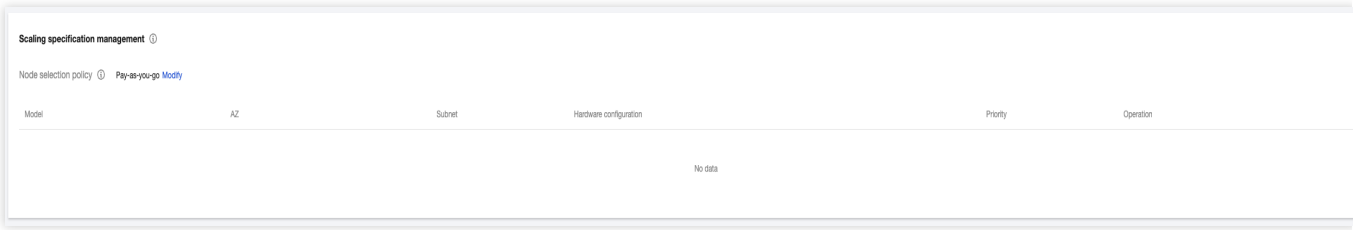
Execute the command line to restart Knox.

```
/usr/local/service/knox/bin/gateway.sh stop ;
/usr/local/service/knox/bin/gateway.sh start
```

Practice of Troubleshooting Unexecuted Auto-Scaling Rules

Last updated：2025-01-03 15:05:10

1. The elastic resource limit has reached the minimum instance count. If further scale-in is required, consider adjusting the minimum instance count.Cause: The scale-in rule was triggered, but the current number of elastic nodes is less than the minimum number of nodes.Solution: If further scale-in is required, reset the minimum node count.
2. The elastic resource limit has exceeded the maximum instance count. If further scale-out is required, consider adjusting the maximum instance count.Cause: The scale-out rule was triggered, but the current number of elastic nodes has reached the maximum instance count.Solution: If further scale-out is required, reset the maximum node count.
3. There is no scaling specification set, so scale-out is not possible. You can try adding specifications and then retry.



Cause: As shown above, an auto-scaling rule has been triggered, but no node specification has been added in the **Console > Auto Scaling > Scaling Specification Management**.Solution: Click **Add Specification** in the top-right corner and select the desired node specification.
4. If resources are insufficient, try switching to a specification with sufficient resources or submit a ticket to contact us.Cause: The scale-out rule was triggered, but the selected model's resources in the current AZ are insufficient.Solution: Reconfigure with a node specification that has sufficient resources.
5. The current retry time is too short; it is recommended to extend the retry duration.
Cause: During the time window from the trigger time to the retry expiration time of the time-based scaling rule, other automatic scaling processes were in progress within the cluster, preventing the current time-based scaling rule from being executed.
Solution: Edit the rule and appropriately extend the expiration retry time to ensure the rule can be executed.
6. The account balance is insufficient, and the scale-out cannot proceed.Cause: The scale-out rule was triggered, but there were insufficient funds when placing the order.Solution: Go to the Cost Center to top up your account.
7. No elastic resources currently meet the conditions for scale-in.
Cause: The scale-in rule was triggered, but there are currently no elastic node resources available, or all nodes are set for scheduled destruction.
Solution: If you need to continue scaling in the nodes scheduled for destruction, you can choose to scale in manually.

8. The cluster is not in a scalable status, so the scale-out is not possible.
Cause: The scale-out rule was triggered, but the current cluster is in a non-operational status such as installing components or scaling out, making it unable to perform the scale-out operation.
Solution: You can perform a manual scale-out or edit the rule to appropriately extend the retry timeout, ensuring the rule can be executed.
9. The cluster is in a scale-out cooldown period and cannot trigger scale-out temporarily. It is recommended to adjust the cooldown time for the scaling rules.
Cause: The scale-out rule was triggered, but the cluster is currently in another scaling cooldown period, so the rule cannot be executed.
Solution: You can shorten the cooldown time of other rules or extend the retry expiration time of the current scale-out rule.

Practice Tutorial on Switching HDFS DataNode Maintenance Status

Last updated：2025-01-03 15:05:10

The DataNode maintenance status (IN_MAINTENANCE) is applicable for scenes where the DataNode is temporarily offline but does not require data migration, such as quick service repairs or disk replacement. In the DataNode maintenance mode, the operation entry is not enabled by default in the console and requires manual configuration by the user for support.

Note:

1. This operation is only supported for Hadoop 3.x and later versions.
2. DataNodes in a paused status do not support maintenance operations.

Enabling Console Switching Management Status Operation Entry

1. Modify the contents of hdfshosts to JSON format.

Use the console cluster script feature to execute the script files hdfshosts_txt_to_json.sh and hdfshosts_txt_to_json_rollback.sh on the Master node. The contents of the script files are as follows:

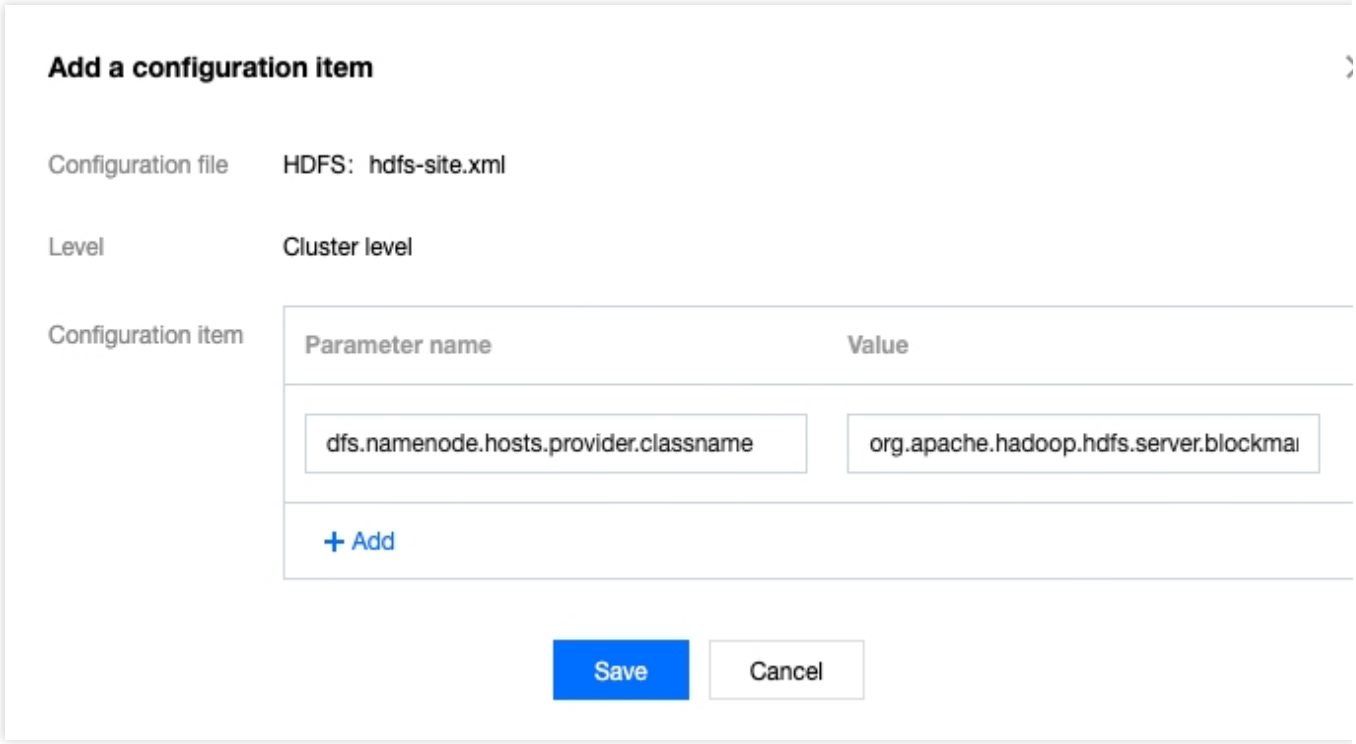
```
#!/bin/bash

cd /usr/local/service/hadoop/etc/hadoop
file=hdfshosts
if [ ! -f "$file" ];then
  echo "$file not exists"
  exit -1
fi

bak_file="$file.txt.bak"
if [ ! -f "$bak_file" ];then
  cp -f $file $file.txt.bak
fi

output_file="$file.tmp"
## Generate JSON File
echo '[' > "$output_file"
first_record=true
while IFS= read -r line
do
  if [ "$first_record" = false ]; then
    echo ',' >> "$output_file"
```

```
  fi

  echo "  {\\"hostName\\": \\"$line\\"}" >> "$output_file"

  first_record=false
done < "$file"
echo ']' >> "$output_file"
mv -f $output_file $file
chown hadoop:hadoop $file
chmod 755 $file
cat $file
```

```
#!/bin/bash

cd /usr/local/service/hadoop/etc/hadoop
file=hdfshosts
bak_file="$file.txt.bak"
cp -f $bak_file $file
chown hadoop:hadoop $file
chmod 755 $file
cat $file
```

2. Add a configuration item to hdfs-site.xml.

Add the configuration parameter dfs.namenode.hosts.provider.classname with the value org.apache.hadoop.hdfs.server.blockmanagement.CombinedHostFileManager.



3. After the configuration is saved and applied, restart NameNode.
4. Check via WebUI or by running hdfs dfsadmin -report.
5. (Recommended) Perform a scale-in and scale-out operation on the core nodes once, as the above operations will modify the hdfshosts file, preventing potential issues in the future.
6.The **Switch Management Status** button will appear in the HDFS **Role Management** section.

DataNode Entering Maintenance Status

1. Log in to the EMR console and click the corresponding **Cluster ID/Nam** in the cluster list to enter the cluster details page.
2. On the cluster details page, click **Cluster Services**, and then select HDFS component and go to **Operations > Role Management** in the top right corner.
3. After checking the DataNode role whose operation status is started in role management, switch the status and put DataNode into maintenance status operation.
4. When you choose to enter maintenance status, you can set the maintenance duration. During the ma4.intenance period, the DataNode will cease to provide external services and no data migration will occur. If the service is not recovered after the configured maintenance period expires, data migration will start.

DataNode Exiting Maintenance Status

1. If the node repair is completed within the maintenance period, the DataNode will automatically resume providing external services after the timeout. However, the maintenance status should be manually exited by the user via the console.
2. Log in to the EMR console and click the corresponding **Cluster ID/Name** in the cluster list to enter the cluster details page.
3. After selecting the DataNode role with the operation status set to Started (Under Maintenance) in Role Management, change the status to take DataNode out of maintenance mode operation.

Data Migration
HDFS Data Migration Using COS

Last updated：2025-01-03 15:05:11

If you need to migrate your HDFS raw data to EMR, you can achieve using either of the following: migrate data with Tencent Cloud Object Storage (COS) service as a transfer stop; migrate data with DistCp, a built-in tool of Hadoop for large inter/intra-cluster copying. This document describes how to migrate data with the first method.

Migrating a non-HDFS file

If your source file is not an HDFS file, upload it to COS via the COS console or API, and then analyze it in the EMR cluster.

Migrating an HDFS file

1. Get the COS migration tool.
Get the migration tool hdfs_to_cos_tools. For more migration tools, see Tool Overview.
2. Configure the tool.
All configuration files are stored in the `conf` directory of the tool directory. Copy the `core-site.xml` file of the HDFS cluster to be synced to `conf`, which contains the configuration information of the NameNode. Edit the configuration file `cos_info.conf` by including your `appid`, bucket, region, and key information.

Note:

We recommend you use a sub-account key and follow the principle of least privilege to avoid leaking resources besides your buckets and objects.
If you need to use a permanent key, we recommend you follow the principle of least privilege to limit the scope of permission for the permanent key.
Command parameter descriptions:

```
-ak <ak>                                the cos secret id // Your `SecretId`. We re
-appid,--appid <appid>                  the cos appid
-bucket,--bucket <bucket_name>          the cos bucket name
-cos_info_file,--cos_info_file <arg>    the cos user info config default is ./conf/
-cos_path,--cos_path <cos_path>         the absolute cos folder path
-h,--help                               print help message
-hdfs_conf_file,--hdfs_conf_file <arg>  the hdfs info config default is ./conf/core
-hdfs_path,--hdfs_path <hdfs_path>      the hdfs path
-region,--region <region>               the cos region. legal value cn-south, cn-ea
-sk <sk>                                the cos secret key // Your `SecretKey`. We
-skip_if_len_match,--skip_if_len_match  skip upload if hadoop file length match cos
```

3. Execute data migration.

```
# All operations must be performed in the tool directory. If both configuration fil
./hdfs_to_cos_cmd -h
# Copy from HDFS to COS (if a file already exists in COS, it will be overwritten)
./hdfs_to_cos_cmd --hdfs_path=/tmp/hive --cos_path=/hdfs/20170224/
# Copy from HDFS to COS, and if a file to be copied is of the same length as a file
# Only the length is checked here, as the overheads would be very high if the diges
./hdfs_to_cos_cmd --hdfs_path=/tmp/hive --cos_path=/hdfs/20170224/ -skip_if_len_mat
# Set parameters completely through the command line
./hdfs_to_cos_cmd -appid 1********* -ak
   **************************************** -sk
   ****************************** -bucket test -cos_path /hdfs
   -hdfs_path /data/data -region cn-south -hdfs_conf_file
/home/hadoop/hadoop-2.8.1/etc/hadoop/core-site.xml
```

4. After the command is verified and run, a log will be generated as shown below:

```
[Folder Operation Result : [ 53(sum)/ 53(ok) / 0(fail)]]
[File Operation Result: [22(sum)/ 22(ok) / 0(fail) / 0(skip)]]
[Used Time: 3 s]
```

`sum` indicates the total number of files to be migrated.

`ok` indicates the number of files successfully migrated.

`fail` indicates the number of files failed to be migrated.

`skip` indicates the number of files skipped because they have the same length as the files of the same name in the destination after the `skip_if_len_match` parameter is added.

You can also log in to the COS console to check whether the data has been migrated correctly. For how to use COS, see Console.

FAQ

Make sure that the configuration information is correct, including `appID` , key, bucket, and region. Make sure that the server time is the same as Beijing time (1-minute difference is acceptable. If the difference is too large, reset your server time).

Make sure that the server for the copy program is accessible to DateNode. The NameNode uses a public IP address and can be accessed, but the DateNode where the obtained block is located uses a private IP address and cannot be accessed; therefore, we recommend you place the copy program in a Hadoop node for execution, so that both the NameNode and DateNode can be accessed.

In case of a permissions issue, use the current account to download a file with the Hadoop command, check whether everything is correct, and then use the synchronization tool to sync the data in Hadoop.

Files that already exist in COS are overwritten by default in case of repeated upload, unless you explicitly specify the `-skip_if_len_match` parameter, which indicates to skip files if they have the same length as the existing files.

The COS path is always considered as a directory, and files that are eventually copied from HDFS will be stored in this directory.

HDFS Data Migration Using DistCp

Last updated：2025-01-03 15:05:10

If you need to migrate your HDFS raw data to EMR, you can achieve this using either of the following: migrate data with Tencent Cloud Object Storage (COS) service as a transfer stop; migrate data with DistCp, a built-in tool of Hadoop for large inter/intra-cluster copying. This document describes how to migrate data with the second method.
DistCp (distributed copy) is a file migration tool that comes with Hadoop. It uses MapReduce to effect its distribution, error handling and recovery, and reporting. It expands a list of files and directories into input to map tasks, each of which will copy a partition of the files specified in the source list. To use DistCp, your cluster and the EMR cluster must be connected over network.
To migrate data with DistCp, perform the following steps:

Step 1. Configure a Network

Migrating local self-built HDFS files to EMR

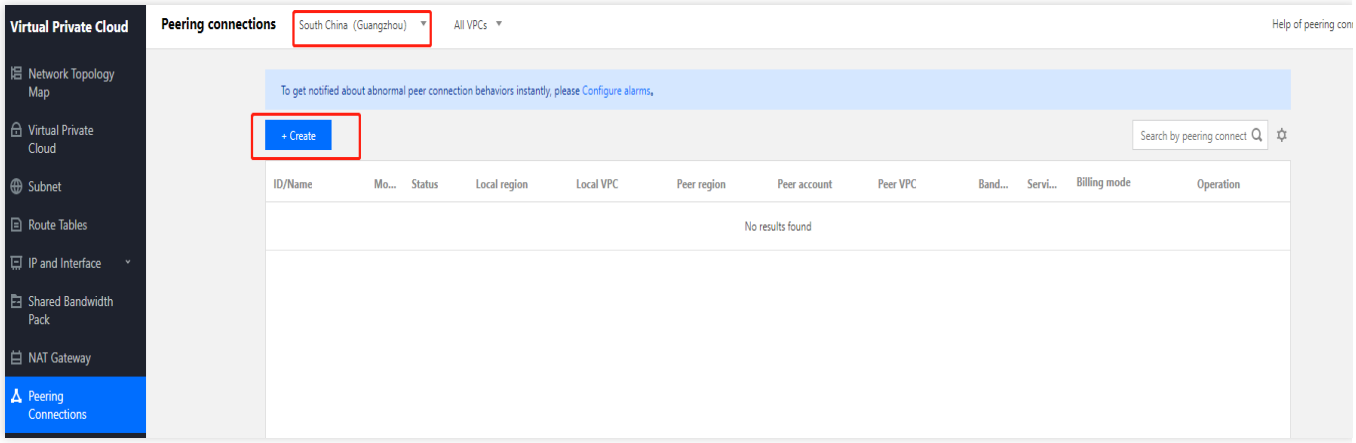
The migration of local self-built HDFS files to an EMR cluster requires a direct connection for network connectivity. You can contact Tencent Cloud technical team for assistance.

Migrating self-built HDFS files in CVM to EMR

If the network where the CVM instance resides and the one where the EMR cluster resides are in the same VPC, the files can be transferred freely.
Otherwise, a peering connection is required for network connectivity.

Using a peering connection

IP CIDR block 1: Subnet A 192.168.1.0/24 in VPC1 of Guangzhou.
IP CIDR block 2: Subnet B 10.0.1.0/24 in VPC2 of Beijing.
1. Log in to the VPC console, enter the **Peering Connections** page, select the region **Guangzhou** at the top of the page, select **VPC1**, and click **+ Create**.



2. On the peering connection creation page, configure the following fields:

Name: Enter a peering connection name, such as PeerConn.

Local region: Enter a local region, such as Guangzhou.

Local network: Enter a local network, such as VPC1.

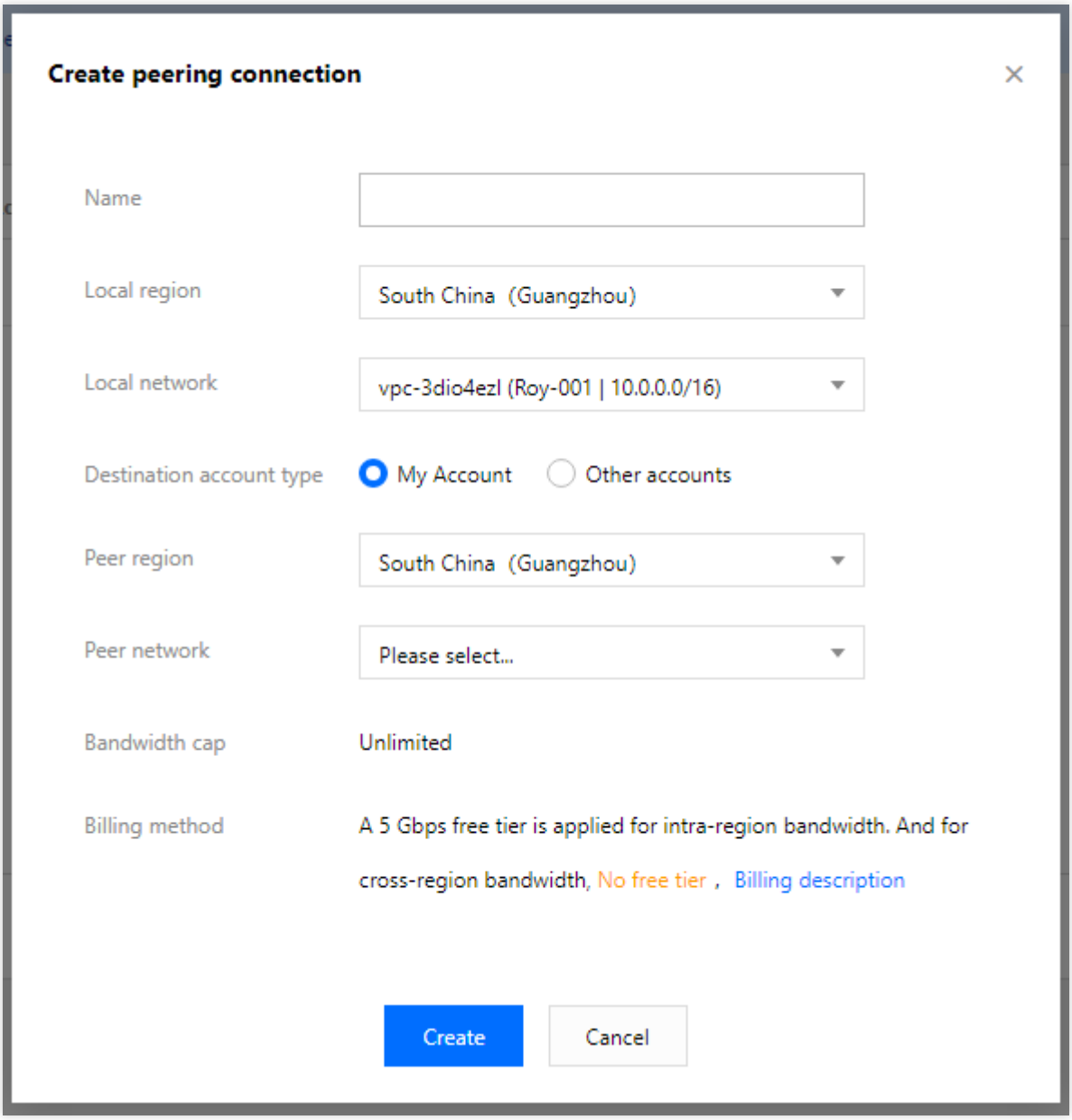
Destination account type: Select the account of the peer network. If the two networks in Guangzhou and Beijing are under the same account, select **My account**; otherwise, select **Other accounts**.

Note:

If both the local and peer networks are in the same region (such as Guangzhou), the communication is free of charge, and you do not need to set the bandwidth cap. Otherwise, fees will be incurred and you can set the bandwidth cap.

Peer region: Enter a peer region, such as Beijing.

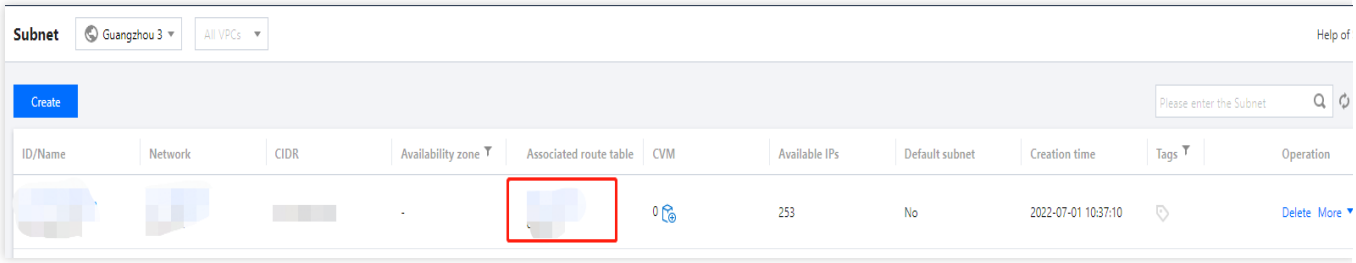
Peer network: Enter a peer network, such as VPC2.



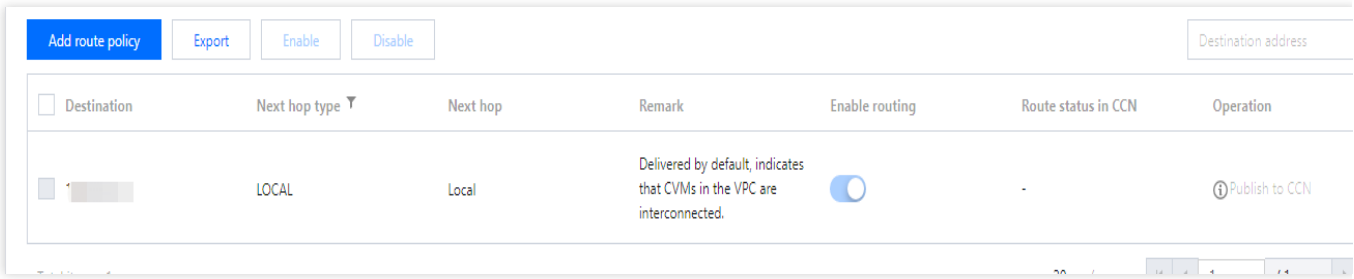
3. A peering connection between VPCs under the same account takes effect immediately after creation. If the VPCs are under different accounts, the peering connection takes effect only after the peer account accepts it. For details, see Creating Intra-account Peering Connection and Creating Cross-account Peering Connection.

4. Configure the local and peer route tables for the peering connection.

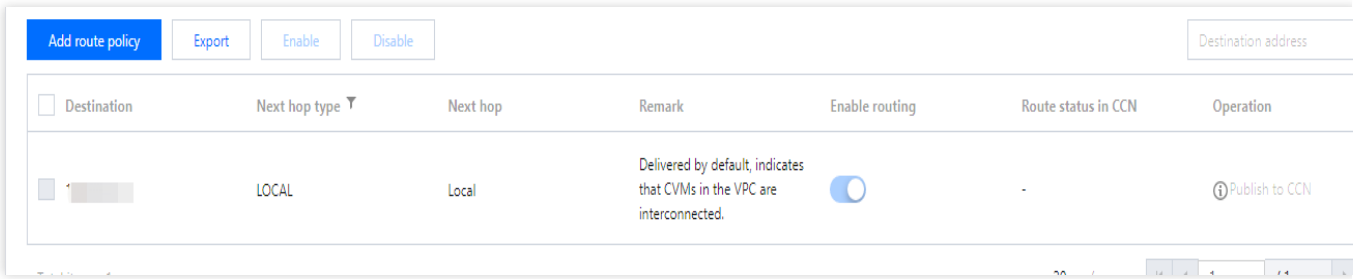
Log in to the VPC console and select **Subnet** to enter the subnet management page. Click the ID of the route table associated with the specified subnet (such as subnet VPC1 in Guangzhou) on the local end of the peering connection to enter the route table details page.



Click **Add route policy**.



Enter the destination CIDR block (such as 10.0.1.0/24 for VPC2 in Beijing), select **Peering connections** for the next hop type, and select the created peering connection (PeerConn) for the next hop.



You've configured the route table from Guangzhou VPC1 to Beijing VPC2 in the previous steps. Now you need to repeat the steps above to configure the route table from Beijing VPC2 to Guangzhou VPC1.

After the route tables are configured, IP CIDR blocks in different VPCs can communicate with each other.

Step 2. Execute copying

```
# Copy the specified folder from one cluster to another
hadoop distcp hdfs://nn1:9820/foo/bar hdfs://nn2:9820/bar/foo

# Copy the specified file
hadoop distcp hdfs://nn1:9820/foo/a hdfs://nn1:9820/foo/b hdfs://nn2:9820/bar/foo

# If too many files need to be specified, use -f parameter to separate them.
```

Note:

For the commands above, the source and destination versions must be the same.

The copying will fail if another client is writing data to the source file or the source file was moved (the `FileNotFoundException` error message will occur); rewriting the source file will fail if it is being copied to the destination.

Practice of Hive Data Migration

Last updated：2025-01-03 15:05:11

Hive migration involves two parts: data migration and metadata migration. Hive table data is primarily stored on HDFS, so data migration mainly occurs at the HDFS level. Hive metadata is mainly stored in relational databases and can be smoothly migrated to TencentDB on the cloud to ensure high availability.

Hive Metadata Migration

1. Dump the source Hive Metastore.

```
mysqldump -hX.X.X.X -uroot -pXXXX --single-transaction --set-gtid-purged=OFF
hivemetastore > hivemetastore-src.sql
# If MySQL does not have GTID enabled, remove --set-gtid-purged=OFF from the
command line.
# X.X.X.X represents the database server address.
# XXXX represents the database password.
# If the database user is not root, use the correct username.
# hivemetastore is the name of the Hive Metastore.
```

2. Confirm the default storage path of Hive table data in HDFS for the target cluster.
The default storage path of Hive table data in HDFS is specified by the `hive.metastore.warehouse.dir` configuration item in `hive-site.xml`. If the storage path of Hive tables in HDFS for the target cluster needs to match that of the source cluster, you can modify the configuration file accordingly. For example, in the source cluster's `hive-site.xml`, `hive.metastore.warehouse.dir` is set as follows:

```
<property>
 <name>hive.metastore.warehouse.dir</name>
 <value>/apps/hive/warehouse</value>
</property>
```

In the target cluster `hive-site.xml`, `hive.metastore.warehouse.dir` is set as follows:

```
<property>
 <name>hive.metastore.warehouse.dir</name>
 <value>/usr/hive/warehouse</value>
</property>
```

If the target cluster's Hive tables still need to maintain the same storage location in HDFS as the source cluster, then modify the `hive-site.xml` in the target to set `hive.metastore.warehouse.dir` to:

```
<property>
 <name>hive.metastore.warehouse.dir</name>
 <value>/apps/hive/warehouse</value>
```

```
</property>
```

3. Verify the target Hive metadata SDS.LOCATION and DBS.DB_LOCATION_URI fields.Use the following query to get the current values of SDS.LOCATION and DBS.DB_LOCATION_URI fields.

```
SELECT DB_LOCATION_URI from DBS;
SELECT LOCATION from SDS;
```

The query results are similar to the following:

```
mysql> SELECT LOCATION from SDS;
+-------------------------------------------------+
| LOCATION |
+-------------------------------------------------+
| hdfs://HDFS2648/usr/hive/warehouse/hitest.db/t1 |
| hdfs://HDFS2648/usr/hive/warehouse/wyp |
+-------------------------------------------------+
mysql> SELECT DB_LOCATION_URI from DBS;
+-----------------------------------------------+
| DB_LOCATION_URI |
+-----------------------------------------------+
| hdfs://HDFS2648/usr/hive/warehouse |
| hdfs://HDFS2648/usr/hive/warehouse/hitest.db |
+-----------------------------------------------+
```

`hdfs://HDFS2648` is the default HDFS file system name, specified by the `fs.defaultFS` parameter in `core-site.xml` .

```
<property>
 <name>fs.defaultFS</name>
 <value>hdfs://HDFS2648</value>
</property>
```

`/usr/hive/warehouse` is the default storage path for Hive tables in HDFS, which is also the value specified by `hive.metastore.warehouse.dir` in `hive-site.xml` . Therefore, we need to modify the SDS.LOCATION and DBS.DB_LOCATION_URI fields in the source Hive metadata SQL file. Ensure that these two fields in the imported Hive Metastore use the correct paths. You can use the following sed command to batch modify the SQL file.

```
Replace the IP address: sed -i 's/oldcluster-ip:4007/newcluster-ip:4007/g'
hivemetastore-src.sql
Replace the defaultFS: sed -i 's/old-defaultFS/new-defaultFS/g' hivemetastore-
src.sql
```

Among them, oldcluster-ip and newcluster-ip are the IP addresses of the source and target cluster namenodes respectively, while old-defaultFS and new-defaultFS represent the fs.defaultFS configuration values for the source and target clusters.

Note

If components such as Kudu or Hbase are used and rely on Metastore for metadata services, the corresponding location fields in the target Hive metadata should also be modified accordingly.

4. Stop the target Hive services: MetaStore, HiveServer2, and WebHCatalog.

5. Back up the target Hive Metastore.

```
mysqldump -hX.X.X.X -uroot -pXXXX --single-transaction --set-gtid-purged=OFF
hivemetastore > hivemetastore-target.sql
# If MySQL does not have GTID enabled, remove --set-gtid-purged=OFF from the
command line.
# X.X.X.X represents the database server address.
# XXXX represents the database password.
# If the database user is not root, use the correct username.
# hivemetastore is the name of the Hive Metastore.
```

6. Drop/Create the target Hive metadata.

```
mysql> drop database hivemetastore;
mysql> create database hivemetastore;
```

7. Import the source Hive Metastore into the target database.

```
mysql -hX.X.X.X -uroot -pXXXX hivemetastore < hivemetastore-src.sql
# X.X.X.X represents the database server address.
# XXXX represents the database password.
# If the database user is not root, use the correct username.
# hivemetastore is the name of the Hive Metastore.
```

8. Hive metadata upgrade.If the target and source Hive versions are the same, this step can be skipped. Otherwise, query the Hive version in both the source and target clusters.

```
hive --service version
```

The Hive upgrade scripts are located in the `/usr/local/service/hive/scripts/metastore/upgrade/mysql/` directory.Hive does not support cross-version upgrades; for example, upgrading Hive from 1.2 to 2.3.0 requires executing each intermediate upgrade sequentially.

```
upgrade-1.2.0-to-2.0.0.mysql.sql -> upgrade-2.0.0-to-2.1.0.mysql.sql ->
upgrade-2.1.0-to-2.2.0.mysql.sql -> upgrade-2.2.0-to-2.3.0.mysql.sql
```

The upgrade scripts mainly perform operations such as creating tables, adding fields, and modifying content. If tables or fields already exist, exceptions for existing fields during the upgrade process can be ignored. For example, upgrading Hive from version 2.3.3 to version 3.1.1.

```
mysql> source upgrade-2.3.0-to-3.0.0.mysql.sql;
mysql> source upgrade-3.0.0-to-3.1.0.mysql.sql;
```

9. If there are Phoenix tables in the source Hive, modify the Phoenix table's ZooKeeper address in the target Hive metadata.Use the following query to get the `phoenix.zookeeper.quorum` configuration for the Phoenix table.

```
mysql> SELECT PARAM_VALUE from TABLE_PARAMS where PARAM_KEY = 'phoenix.zookeeper.qu
+-------------------------------------------------+
| PARAM_VALUE |
+-------------------------------------------------+
| 172.17.64.57,172.17.64.78,172.17.64.54 |
+-------------------------------------------------+
```

Check the ZooKeeper address of the target cluster, which is specified in the `hbase.zookeeper.quorum` field in the hive-site.xml configuration file.

```
<property>
 <name>hbase.zookeeper.quorum</name>
 <value>172.17.64.98:2181,172.17.64.112:2181,172.17.64.223:2181</value>
</property>
```

Modify the ZooKeeper address of the Phoenix table in the target Hive metadata to match the ZooKeeper address of the target cluster.

```
mysql> UPDATE TABLE_PARAMS set PARAM_VALUE =
'172.17.64.98,172.17.64.112,172.17.64.223' where PARAM_KEY =
'phoenix.zookeeper.quorum';
```

10. Check the case format of table names in the target Hive metadata, and see the following example to convert all lowercase table names to uppercase:

```
alter table metastore_db_properties rename to METASTORE_DB_PROPERTIES;
```

11. Start the target Hive services: MetaStore, HiveServer2, and WebHcataLog.

12. Finally, you can verify by running a simple Hive SQL query.

Practical Tutorial on Custom Scaling
Practical Tutorial on Setting Scaling Rules

Last updated：2025-01-03 15:05:11

Principles for Adding Preset Resources When Scale-out Rules Are Executed

Each cluster can configure up to 10 types of scaling specifications. When the scale-out rule is triggered, the scale-out will be executed based on the priority of the specifications. If the high-priority specification resources are insufficient, the sub-priority resources will be combined with high-priority resources to supplement the calculation resources (**following the same order for pay-as-you-go and spot instances**).

When resources are sufficient: 1 > 2 > 3 > 4 > 5

Example:

When 5 types of specifications are preset and resources are sufficient, if the scale-out rule is triggered to scale out 10 nodes, 10 nodes will be scaled out based on specification 1 in sequence, and the other preset specifications will not be selected.

When resources are insufficient: 1+2 ＞ 1+2+3 ＞ 1+2+3+4 ＞ 1+2+3+4+5

Example:

When preset specification 1 has 8 nodes, specification 2 has 4 nodes, and specification 3 has 3 nodes, if the scale-out rule triggers the need to scale out 13 nodes, 8 nodes will be scaled out based on specification 1, 4 nodes will be scaled out based on specification 2, and 1 node will be scaled out based on specification 3 in the sequence.

When the resource specification is out of stock, assuming specification 2 is unavailable: 1+3 > 1+3+4 > 1+3+4+5.

Example:

When preset specification 1 has 8 nodes, specification 2 is unavailable, and specification 3 has 3 nodes, if the scale-out rule is triggered to scale out 10 nodes, 8 nodes will be scaled out based on specification 1, specification 2 will not be selected, and 2 nodes will be scaled out based on specification 3 in the sequence.

When preset specification 1 has 8 nodes and all other preset specifications are unavailable, if the scale-out rule is triggered to scale out 10 nodes, 8 nodes will be scaled out based on specification 1, with partial success in scale-out.

Scale-out methods: You can choose from nodes, memory, or cores. All three methods only support non-zero integer input. When you select cores or memory as the method, the scale-out process ensures maximum computing power by converting the node quantity accordingly.

Example:

When you scale out by cores, if the scale-out rule is set to 10 cores but the specification priority is for 8-core nodes, the rule will trigger the scale-out of **two 8-core nodes.**

When you scale out by memory, if the scale-out rule is set to 20 GB but the specification priority is for 16 GB nodes, the rule will trigger the scale-out of **two 16 GB nodes**.

Principles for Scaling in Elastic Nodes When Scale-in Rules Are Executed

Elastic nodes added through the auto-scaling feature will prioritize scaling in idle nodes first, following the principle of **scaling in nodes in reverse order of creation time**. If the required scale-in number is not met, nodes running containers will then be selected for scale-in. For load-based scale-in, nodes running services with load metrics will be prioritized, and idle nodes will be scaled in first. This also follows the **principle**, and if the required scale-in number is not met, nodes running containers will be selected for scale-in. Non-elastic nodes will not be affected by scale-in rules, and scale-in actions will not be triggered. Non-elastic nodes only support manual scale-in.

Note

Scheduled termination of nodes **will not be constrained by the principles of scaling in nodes in reverse order of creation time and minimum number of cluster nodes**. The scale-in will be executed once the set time is reached, with a default graceful scale-in period of 30 minutes.

The criteria for determining an idle node is that there are no running containers within the last 5 minutes.

Scale in based on load, assuming the node creation time is from earliest to latest: A > B > C > D > E.

Example:

When you set YARN load metric scale-in for 5 nodes, with C, D, and E deployed with YARN components and D and E running containers, the scale-in order when the rule is triggered will be C > E > D > B > A.

When you set Trino load metric scale-in for 5 nodes, with C, D, and E deployed with Trino components and D and B running containers, the scale-in order will be E > C > D > A > B, when the rule is triggered.

Scale in based on time, assuming the node creation time is from earliest to latest: A > B > C > D > E.

Example:

When you set node-based scale-in for 5 nodes, with D and E running containers, the scale-in order will be C > B > A > E > D, when the rule is triggered.

Scale-in methods: Support for three options including nodes, memory, and cores. Only non-zero integer values are allowed for all three options. When you select cores or memory, the scale-in process ensures business continuity by calculating the minimum number of nodes required for scale-in. If no tasks are running on the nodes, they will be scaled in following a reverse chronological order, ensuring at least one node is scaled in.

Example:

When you scale in by cores, setting a scale-in of 20 cores. When the scale-in rule is triggered, with the cluster having elastic nodes consisting of three 8-core 16 GB nodes and two 4-core 8 GB nodes (in reverse chronological order), the system will successfully scale in two 8-core 16 GB nodes.

When you scale in by memory, setting a scale-in of 30 GB. When the scale-in rule is triggered, with the cluster having elastic nodes consisting of three 8-core 16 GB nodes and two 4-core 8 GB nodes (in reverse chronological order), the

system will successfully scale in one 8-core 16 GB node.

Principles for Triggering and Executing Scaling Rules

Elastic scaling rules can be set based on both time and load metrics. The rules follow the first triggered, first executed principle, and if multiple rules are triggered simultaneously, they are executed based on their priority order. The rule status indicates whether the rule is active or not. By default, it is enabled, but the status can be set to disabled when you want to keep the configuration without executing the rule.
Scaling based on load only.
1.1 Follows the first triggered, first executed, and if multiple rules are triggered simultaneously, they are executed based on their priority order principle, such as **1 > 2 > 3 > 4 > 5**.
1.2 A single load-based scaling rule can support multiple metrics. The rule is triggered when all metrics meet the conditions.
1.3 Load-based scaling can be set to monitor cluster load changes within a specific time period.
Scaling based on time only.
1.1 Follows the first triggered, first executed, and if multiple rules are triggered simultaneously, they are executed based on their priority order principle, such as **1 > 2 > 3 > 4 > 5**.
1.2 The rule can be set to execute repeatedly. Once the rule expires, it becomes inactive. Alarms will be sent before expiration; see the alarm configuration.
Scaling based on both load and time.
Follows the first triggered, first executed, and if multiple rules are triggered simultaneously, they are executed based on their priority order principle, such as **1 > 2 > 3 > 4 > 5**.

Corresponding Relationships of Queue Load Metrics

Load Type	Category	Dimension	EMR Auto-scaling Metric	Meaning of Metrics
YARN	AvailableVCores	root	AvailableVCores#root	Number of availabl cores in the Root q
		root.default	AvailableVCores#root.default	Number of availabl cores in the root.de queue
		Custom sub-queue	For example: AvailableVCores#root.test	Number of availabl cores in the root.te
	PendingVCores	root	PendingVCores#root	Number of virtual c

			needed for upcomi in the Root queue
	root.default	PendingVCores#root.default	Number of virtual c needed for upcomi in the root.default c
	Custom sub-queue	For example: PendingVCores#root.test	Number of virtual c needed for upcomi in the root.test que
AvailableMB	root	AvailableMB#root	Available memory Root queue (MB)
	root.default	AvailableMB#root.default	Available memory root.default queue
	Custom sub-queue	For example: AvailableMB#root.test	Available memory root.test queue (MI
PendingMB	root	PendingMB#root	Available memory for upcoming tasks Root queue (MB)
	root.default	PendingMB#root.default	Available memory for upcoming tasks root.default queue
	Custom sub-queue	For example: PendingMB#root.test	Available memory for upcoming tasks root.test queue (MI
AvailableMemPercentage	Cluster	AvailableMemPercentage	Percentage of avai memory
ContainerPendingRatio	Cluster	ContainerPendingRatio	Ratio of pending co to allocated contai
AppsRunning	root	AppsRunning#root	Number of running the root queue
	root.default	AppsRunning#root.default	Number of tasks ru the root.default que
	Custom sub-queue	For example: AppsRunning#root.test	Number of tasks ru the root.test queue
AppsPending	root	AppsPending#root	Number of pending

			the root queue
	root.default	AppsPending#root.default	Number of pending the root.default que
	Custom sub-queue	For example: AppsPending#root.test	Number of pending the root.test queue
PendingContainers	root	PendingContainers#root	Number of pending containers in the ro
	root.default	PendingContainers#root.default	Number of pending containers in the ro queue
	Custom sub-queue	For example: PendingContainers#root.test	Number of pending containers in the ro queue
AllocatedMB	root	AllocatedMB#root	Allocated memory root queue
	root.default	AllocatedMB#root.default	Allocated memory root.default queue
	Custom sub-queue	For example: AllocatedMB#root.test	Allocated memory root.test queue
AllocatedVCores	root	AllocatedVCores#root	Number of virtual c allocated to the roo
	root.default	AllocatedVCores#root.default	Number of virtual c allocated to the roo queue
	Custom sub-queue	For example: AllocatedVCores#root.test	Number of virtual c allocated to the roo queue
ReservedVCores	root	ReservedVCores#root	Number of virtual c reserved in the roo
	root.default	ReservedVCores#root.default	Number of virtual c reserved in the roo queue
	Custom sub-queue	For example: ReservedVCores#root.test	Number of virtual c reserved in the roo

			queue
AllocatedContainers	root	AllocatedContainers#root	Number of contain allocated in the roo
	root.default	AllocatedContainers#root.default	Number of contain allocated in the roo queue
	Custom sub-queue	For example: AllocatedContainers#root.test	Number of contain allocated in the roo queue
ReservedMB	root	ReservedMB#root	Amount of memory in the root queue
	root.default	ReservedMB#root.default	Amount of memory in the root.default q
	Sub Queue Definition	e.g., ReservedMB#root.test	Amount of Reserve Memory in the root queue
AppsKilled	root	AppsKilled#root	Number of termina in the root queue
	root.default	AppsKilled#root.default	Number of termina in the root.default q
	Sub Queue Definition	e.g., AppsKilled#root.test	Number of termina in the root.test que
AppsFailed	root	AppsFailed#root	Number of failed ta the root queue
	root.default	AppsFailed#root.default	Number of failed ta the root.default que
	Sub Queue Definition	For example: AppsFailed#root.test	Number of failed ta the root.test queue
AppsCompleted	root	AppsCompleted#root	Number of comple in the root queue
	root.default	AppsCompleted#root.default	Number of comple in the root.default q

		Sub Queue Definition	e.g., AppsCompleted#root.test	Number of comple in the root.test que
	AppsSubmitted	root	AppsSubmitted#root	Number of tasks su to the root queue
		root.default	AppsSubmitted#root.default	Number of tasks su to the root.default q
		Sub Queue Definition	e.g., AppsSubmitted#root.test	Number of tasks su in the root.test que
	AvailableVCoresPercentage	Cluster	Cluster	AvailableVCoresPe
	MemPendingRatio	root	MemPendingRatio#root	Percentage of avai memory waiting in queue
		root.default	MemPendingRatio#root.default	Percentage of avai memory waiting in root.default queue
		Sub Queue Definition	e.g., MemPendingRatio#root.test	Percentage of avai memory waiting in root.test queue
Trino	FreeDistributed	Cluster	FreeDistributed	Available Distribute memory in the clus
	QueuedQueries	Cluster	QueuedQueries	Total number of qu waiting to be execu the queue

Practical Tutorial on Setting Scaling Rules Practical Tutorial on Setting Time-based Scaling Rules

Last updated：2025-01-03 15:05:10

Based on the clear peaks and valleys in business activity over a certain period, you can choose between setting the execution frequency to Repeat or Execute only once. Configure scale-out rules and scale-in rules accordingly. When choosing Repeat, you can set the rule's effective deadline by configuring the rule's validity period, after which the scaling rules will no longer be triggered.

Example:

Your business activity starts increasing at 10 PM and begins to decrease at 6 AM daily, and this pattern is expected to last for one month. You can configure a time-based policy by setting up two scaling rules (one for scale-out and one for scale-in) or a single scale-out rule with scheduled termination.

Scaling Rule: Set to repeat daily. Configure the scale-out rule to be triggered at 10 PM each day for one month.

Scaling-down rule: Set to repeat daily. Configure the scale-in rule to be triggered at 6 AM each day for one month.

Scaling Rule + Scheduled Termination:scheduled termination: Set to repeat daily. Configure the scale-out rule to be triggered at 10 PM each day, with the allocated resources scheduled for 8 hours of use (equivalent to terminating at 6 AM the next day). This configuration will continue for one month. Support for Daily, Weekly, or Monthly repetition is available, so adjust based on your actual requirements. For more details on other rule configuration items and usage, see Setting Time-Based Scaling.

Note:

1. The timing for adding resources to the cluster above represents an ideal scene. In practice, the actual scale-out time depends on the number of resources requested. It is recommended to set the time rules at least 5 minutes earlier based on your needs.
2. During peak periods, resource contention may prevent the actual scale-out number from reaching the elastic target number of machines. It is recommended to enable the Resource Replenishment Retry Policy for your scale-out rule.
3. When the scale-in action is triggered, nodes may still be executing tasks. To avoid immediate release of the nodes, it is recommended that you enable graceful scale-in. For more details, see Graceful Scale-In.

Practical Tutorial on Setting Load-based Scaling Rules

Last updated：2025-01-03 15:05:10

Based on the changes in YARN metrics within the cluster, select the metrics from the past that align with business fluctuations, configure specific thresholds, and then save and apply the settings. Once business activity changes, the corresponding rules will be triggered. The selected metrics should be inversely related to capacity changes, so after scaling activities occur, the change in instance numbers can reduce the corresponding metrics.

Example:

To configure the scale-out rule: if the average value of AppsPending#root is greater than or equal to 1 within 300 seconds and this occurs consecutively 2 times, a scale-out action will be triggered. This can effectively reduce the number of pending tasks in the queue.

Scaling-up rule

Scaling-down rule similarly: configure based on actual conditions. For more details on other rule configuration items and usage instructions, see Set Load Scaling.

1.1 For each rule, you can configure multiple metric conditions, and when all the conditions are met, scaling is triggered.

1.2 To avoid frequent scaling activities that lead to resource waste, you can configure a cooling period for the rule. During the cooling period, even if the scaling conditions are met, no scaling activity will occur.

1.3 Configure an active time (the current rule takes effect within a customized time range). Different scaling rules are allowed to be combined, and you can set different scaling conditions for different time periods.

Note:

1. During peak periods, resource contention may prevent the actual scale-out number from reaching the elastic target number of machines. It is recommended to enable the Resource supplement retry for your scale-out rule.

2. When the scale-in action is triggered, nodes may still be executing tasks. To avoid immediate release of the nodes, it is recommended that you enable graceful scale-in. For more details, see Graceful Scale-In.

Practical Tutorial on Setting Mixed Scaling Rules

Last updated：2025-01-03 15:05:10

Mixed Scene 1:

The business experiences noticeable peaks and valleys within a certain period, along with sudden, non-periodic short-term business peaks.

Example:

Every weekday at 8:30 AM, a fixed workload analysis and statistics task is required, lasting for 2 hours, which needs 1 additional node for computing power. At other times, the required computing power for unexpected peak loads is uncertain. In this case, three scaling rules can be configured to ensure sufficient computing power and cost savings.

Rule 1: Set a time-based scale-out rule by choosing to repeat the rule every Monday/Wednesday/Thursday/Friday and schedule it to scale out 1 node at 8:15 AM tomorrow, with a scheduled termination duration of 3 hours.

Rule 2: Set a load-based scale-out rule, and select the monitoring metrics as needed. It is recommended not to set a validity period, with the default being active all day, scaling out 1 node.

Rule 3: Set a load-based scale-in rule, and select the monitoring metrics as needed. It is recommended not to set a validity period, with the default being active all day, scaling in 1 node.

Note:

1. Scaling out resources takes time, and the time required is proportional to the number of resources being scaled out. It is recommended to prepare resources at least 15 minutes in advance. Typically, the time required is relatively short.
2. The priority order for the three scale-out rules is Rule 1 > Rule 2 > Rule 3; the number of resources to scale can be adjusted based on actual business needs.

Mixed Scene 2:

The business experiences noticeable variations in activity between day and night.

Example:

Business peaks occur every day at 6:30 AM and 5:30 PM, requiring an additional 10 nodes for computing power. The peak duration is uncertain, while a lower computing power requires only 1 node during other times. In this case, you can configure three scaling rules to ensure adequate computing power while optimizing costs.

Rule 1: Set a time-based scale-out rule by choosing to repeat the rule daily and schedule it to be triggered at 6:15 AM tomorrow to scale out 10 nodes.

Rule 2: Set a time-based scale-out rule by choosing to repeat the rule daily and schedule it to be triggered at 5:15 PM tomorrow to scale out 10 nodes.

Rule 3: Set a load-based scale-in rule, and select the monitoring metrics as needed. It is recommended not to set a validity period, with the default being active all day, scaling in 9 node.

Note:

1. The basic configuration sets the maximum number of nodes to 10 and the minimum to 1 to prevent resource wastage and ensure sufficient computing power without any elastic resource shortages. The priority order of the three scale-out rules is Rule 1 > Rule 2 > Rule 3.
2. Scaling out resources takes time, and the time required is proportional to the number of resources being scaled out. It is recommended to prepare resources at least 15 minutes in advance. Typically, the time required is relatively short.