

Alibaba Cloud E-MapReduce **Kernel Enhancement**

Issue: 20200311

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, consequent

ial, 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 contents in Alibaba Cloud documents, including but not limited to pictures, architecture design, page layout, and text description, 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 this document 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

.

Document conventions

Style	Description	Example
	A danger notice 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.
	A warning notice 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 restart an instance.
	A caution notice indicates warning information, supplementary instructions, and other content that the user must understand.	 Notice: If the weight is set to 0, the server no longer receives new requests.
	A note indicates supplemental instructions, best practices, tips, and other content.	 Note: You can use Ctrl + A to select all files.
>	Closing angle brackets are used to indicate a multi-level menu cascade.	Click Settings > Network > Set network type.
Bold	Bold formatting is used for buttons, menus, page names, and other UI elements.	Click OK.
Courier font	Courier font is used for commands.	Run the <code>cd /d C:/window</code> command to enter the Windows system folder.
<i>Italic</i>	Italic formatting is used for parameters and variables.	<code>bae log list --instanceid</code> <code>Instance_ID</code>
[] or [a b]	This format is used for an optional value, where only one item can be selected.	<code>ipconfig [-all -t]</code>

Style	Description	Example
{} or {a b}	This format is used for a required value, where only one item can be selected.	switch { <i>active</i> <i>stand</i> }

Contents

Legal disclaimer.....	I
Document conventions.....	I
1 Get started with JindoFS.....	1
1.1 Use JindoFS in E-MapReduce V3.20.0 to V3.22.0 (V3.22.0 excluded).....	1
1.2 Use JindoFS in E-MapReduce V3.22.0 or later.....	9
1.3 Use the block storage mode.....	19
1.4 Use the cache mode.....	24
1.5 Use the external client.....	28
2 JindoFS ecosystem.....	31
2.1 Migrate data from HDFS to JindoFS.....	31
2.2 Use MapReduce to process data in JindoFS.....	32
2.3 Use Hive to query data in JindoFS.....	33
2.4 Use Spark to process data in JindoFS.....	36
2.5 Use Flink to process data in JindoFS.....	37
2.6 Use Impala or Presto to query data in JindoFS.....	38
2.7 Use JindoFS as the storage back end of HBase.....	39
2.8 Store the logs of YARN MapReduce and Spark jobs.....	41
2.9 Import data from Kafka to JindoFS.....	46

1 Get started with JindoFS

1.1 Use JindoFS in E-MapReduce V3.20.0 to V3.22.0 (V3.22.0 excluded)

This topic describes how to configure and use JindoFileSystem (JindoFS), and its scenarios.

Overview

JindoFS is a cloud-native file system that combines the advantages of Object Storage Service (OSS) and local storage. JindoFS is also the next-generation storage system that provides efficient and reliable storage services for cloud computing in E-MapReduce.

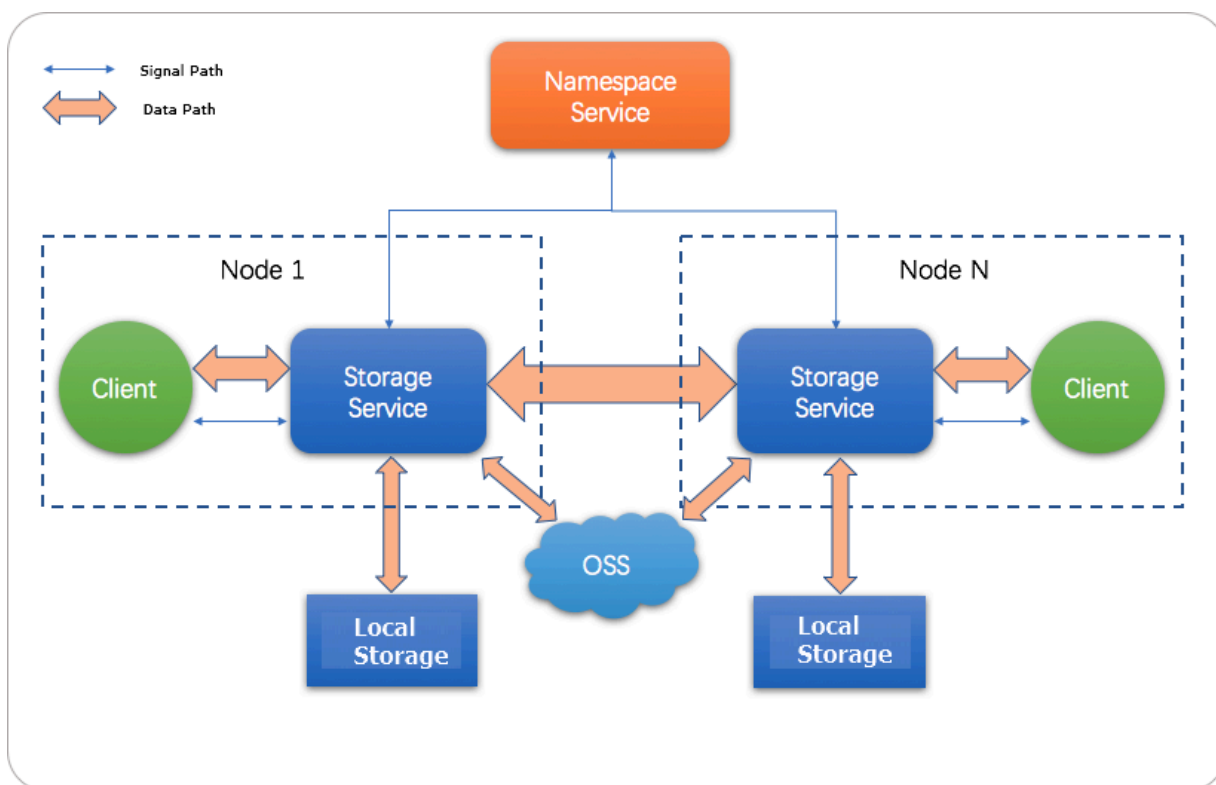
JindoFS supports the block storage mode and cache mode.

JindoFS adopts a heterogeneous multi-backup mechanism. Storage Service provides data storage capability. Data is stored in OSS to guarantee high reliability . Redundant backups are stored in the local cluster to accelerate read operations. Namespace Service manages metadata of JindoFS. In this case, metadata is queried from Namespace Service instead of OSS, which improves query performance. This query method of JindoFS is the same as that of Hadoop Distributed File System (HDFS).



Note:

- **E-MapReduce V3.20.0 and later support JindoFS. To use JindoFS, select the related services when you create an E-MapReduce cluster.**
- **This topic describes how to use JindoFS in E-MapReduce V3.20.0 to V3.22.0 (V3.22.0 excluded). For more information about how to use JindoFS in E-MapReduce V3.22.0 or later, see [Use JindoFS in E-MapReduce V3.22.0 or later](#).**



Scenarios

E-MapReduce has three storage systems: E-MapReduce OssFileSystem, E-MapReduce HDFS, and E-MapReduce JindoFS. Among them, OssFileSystem and JindoFS store data in the cloud. The following table compares the features of three E-MapReduce storage systems and Hadoop support for Alibaba Cloud OSS.

Feature	Hadoop support for Alibaba Cloud OSS	E-MapReduce OssFileSystem	E-MapReduce HDFS	E-MapReduce JindoFS
Storage capacity	Tremendous	Tremendous	Depends on the E-MapReduce cluster scale	Tremendous
Reliability	High	High	High	High
Factor that affects throughput	Server	I/O performance of caches on disks in the E-MapReduce cluster	I/O performance of disks in the E-MapReduce cluster	I/O performance of disks in the E-MapReduce cluster

Feature	Hadoop support for Alibaba Cloud OSS	E-MapReduce OssFileSystem	E-MapReduce HDFS	E-MapReduce JindoFS
Metadata query efficiency	Low	Medium	High	High
Scale-out operation	Easy	Easy	Easy	Easy
Scale-in operation	Easy	Easy	Requires node decommission	Easy
Data locality	None	Weak	Strong	Medium

The block storage mode of JindoFS has the following features:

- JindoFS offers tremendous and scalable storage capacity by using OSS as the storage back end. The storage capacity is independent of the E-MapReduce cluster scale. The local cluster can be scaled in or out as required.
- JindoFS stores a certain amount of backup data in the local cluster to accelerate read operations. This improves the throughput by using limited local storage capacity, especially for Write Once Read Many (WORM) solutions.
- JindoFS provides efficient metadata query similar to HDFS. Compared with OssFileSystem, JindoFS saves much time in metadata query. In addition, JindoFS avoids system instability when data and metadata are frequently accessed.
- JindoFS moves computation as close as possible to data. This reduces the load on network transmission and improves the read performance.

Prepare the environment

- **Create an E-MapReduce cluster**

Select a version from E-MapReduce V3.20.0 to V3.22.0 (V3.22.0 excluded). Select SmartData and Bigboot for Optional Services. For more information about how to create an E-MapReduce cluster, see [#unique_6](#). Bigboot provides distributed data

management and component management services in E-MapReduce. Based on Bigboot, SmartData provides JindoFS for the application layer.

- **Configure JindoFS**

JindoFS provided by SmartData uses OSS as the storage back end. Therefore, you need to set OSS-related parameters before using JindoFS. E-MapReduce provides two configuration methods. If you use the first configuration method, you need to create an E-MapReduce cluster, modify Bigboot-related parameters, and then restart SmartData for the configuration to take effect. If you use the second configuration method, you need to add custom configuration when you create an E-MapReduce cluster. In this case, the related services are restarted based on custom parameters after the E-MapReduce cluster is created.

- **Initialize parameters after the E-MapReduce cluster is created**

You can set all JindoFS-related parameters in Bigboot. As shown in the following figure, the parameters framed in red are required. The `oss.access.bucket` parameter specifies the name of the OSS bucket. The `oss.data-dir` parameter specifies the directory of JindoFS in the OSS bucket. The directory only serves as the storage back end for JindoFS. The data generated in the directory cannot be damaged. The directory is automatically created when JindoFS writes data. You are not required to create the directory in advance. The `oss.access.endpoint` parameter specifies the region of the OSS bucket. The `oss.access.key` and `oss.access.secret` parameters

specify the AccessKey ID and AccessKey secret used to access the OSS bucket, respectively. We recommend that you select an OSS bucket in the same region and under the same account as the storage back end of the E-MapReduce cluster for better performance and stability. In this case, the E-MapReduce cluster can access the OSS bucket without using the AccessKey ID and AccessKey secret.

The screenshot shows the Bigboot configuration interface. On the left, there is a 'Configuration Filter' section with a search bar and a 'Scope' dropdown set to 'Default Cluster Configuration'. Below this, there are several tabs for configuration types: BASIC, ADVANCED, INFORMATION, DATA_PATH, LOG_PATH, LOG, JVM, DATA, DATABASE, PERFORMANCE, TIME, CODEC, OSS, PORT, MEMORY, DISK, NETWORK, PATH, and URI. The 'Service Configuration' section on the right shows a list of parameters for the 'bigboot' component. The parameters are: oss.access.secret, node.data-dirs.watermark.low.ratio (0.3), oss.data-dir (JindoFS-1), oss.access.endpoint, node.data-dirs.watermark.high.ratio (0.6), max.blockletBuffer.group.num (16), oss.access.key, and oss.access.bucket (oss-bucket-name). The parameters oss.data-dir, oss.access.endpoint, and oss.access.bucket are highlighted with red boxes.

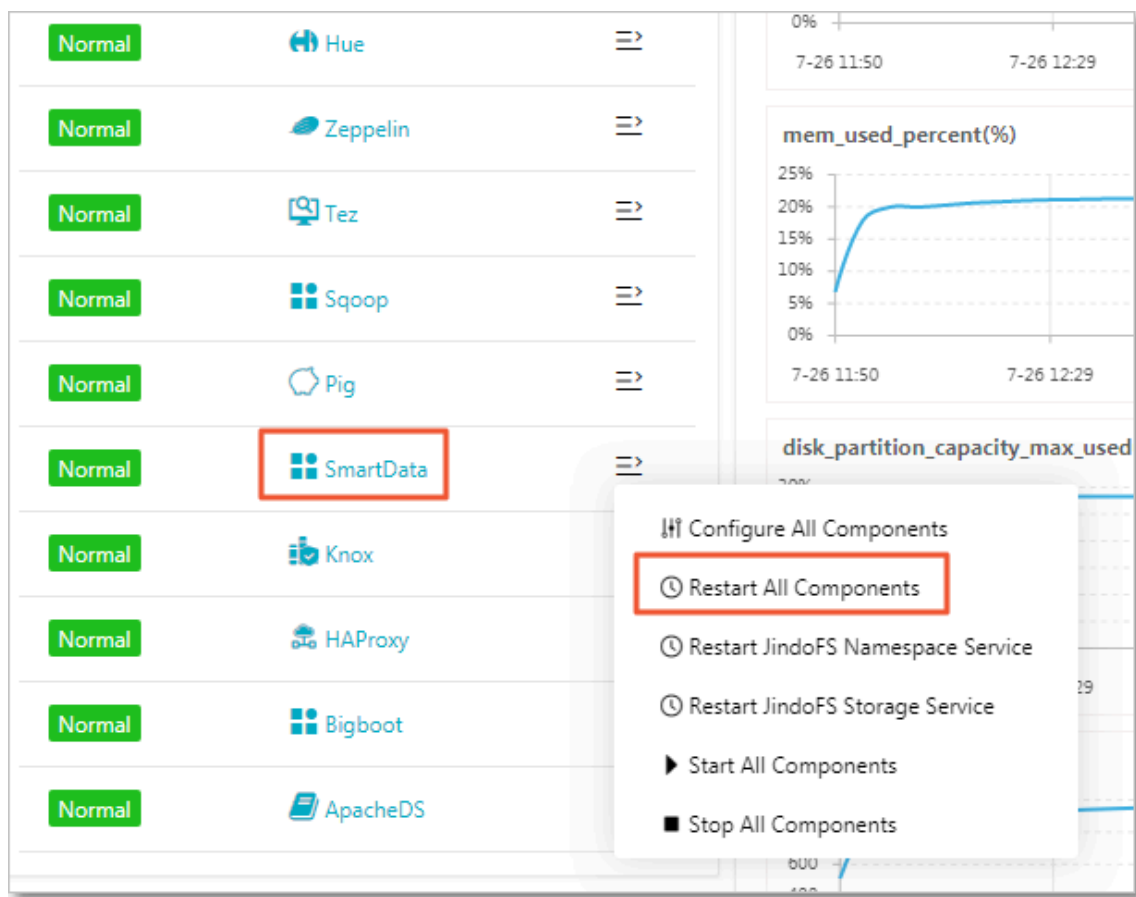


Note:

■ The parameters framed in red in the preceding figure are required.

■ JindoFS supports multiple namespaces. A namespace named test is used in this topic.

Save and deploy the JindoFS configuration. Restart all components in SmartData to use JindoFS.



- Add custom configuration when creating an E-MapReduce cluster

You can add custom configuration when creating an E-MapReduce cluster. For example, you want to create an E-MapReduce cluster in the same region as an OSS bucket to access OSS without using an AccessKey. As shown in the following figure, turn on Custom Software Settings. Add the following configuration including `oss.data-dir` and `oss.access.bucket` to the field in the Advanced Settings section:

```
[
  {
    "ServiceName": "BIGBOOT",
    "FileName": "bigboot",
    "ConfigKey": "oss.data-dir",
    "ConfigValue": "jindoFS-1"
  },
  {
    "ServiceName": "BIGBOOT",
    "FileName": "bigboot",
```

```

    "ConfigKey": "oss.access.bucket",
    "ConfigValue": "oss-bucket-name"
  }
]

```

The screenshot shows the EMR console interface. At the top, the EMR Version is set to EMR-3.20.0. Under 'Required Services', several services are listed as buttons: HDFS (2.8.5), YARN (2.8.5), Hive (3.1.1), Spark (2.4.2), Knox (1.1.0), Zeppelin (0.8.1), Tez (0.9.1), ApacheDS (2.0.0), Ganglia (3.7.2), Pig (0.14.0), Sqoop (1.4.7), and Hue (4.1.0). Under 'Optional Services', more services are listed: HBase (1.4.9), ZooKeeper (3.4.13), Presto (0.213), Impala (2.12.2), Flume (1.8.0), Livy (0.6.0), Superset (0.28.1), Ranger (1.2.0), Flink (1.7.2), Storm (1.2.2), Phoenix (4.14.1), SmartData (1.0.0), Bigboot (1.0.0), and Oozie (5.1.0). The 'Bigboot' and 'SmartData' buttons are highlighted with blue checkmarks. Below the services, there is an 'Advanced Settings' section. It includes a 'Kerberos Mode' toggle (disabled) and a 'Custom Software Settings' toggle (enabled). The 'Custom Software Settings' section contains a JSON configuration box with the following content:


```

[{"ServiceName": "BIGBOOT", "FileName": "bigboot", "ConfigKey": "oss.data-dir", "ConfigValue": "jindoFS-1"},
 {"ServiceName": "BIGBOOT", "FileName": "bigboot", "ConfigKey": "oss.access.bucket", "ConfigValue": "oss-bucket-name"}]
    
```

Use JindoFS

The use of JindoFS is similar to that of HDFS. JindoFS also provides a prefix. To use JindoFS, you only need to replace the `hdfs` prefix with the `jfs` prefix. Example:

```

hadoop fs -ls jfs:/// hadoop fs -mkdir jfs:///test-dir
hadoop fs -put test.log jfs:///test-dir/

```

Data can be read from JindoFS only when Hadoop, Hive, and Spark jobs are running in the E-MapReduce cluster.

Control disk space usage

The back end of JindoFS is based on OSS that is capable of storing large amounts of data. However, the storage capacity of local disks is limited. Therefore, JindoFS releases data backups that are less frequently accessed. Alibaba Cloud uses `node.data-dirs.watermark.high.ratio` and `node.data-dirs.watermark.low.ratio` to adjust the space usage of local disks. The values of both parameters are in the range of 0 to 1 to indicate the percentage of space usage. JindoFS uses the total storage capacity of all data disks by default. The `node.data-dirs.watermark.high.ratio` parameter specifies the upper limit of space usage on each disk. Less frequently

accessed data stored on a disk is released if the space used by JindoFS reaches the upper limit. The `node.data-dirs.watermark.low.ratio` parameter specifies the lower limit of space usage on each disk. After the space usage of a disk reaches the upper limit, less frequently accessed data is released until the space usage of the disk reaches the lower limit. You can set the upper limit and lower limit to adjust and assign disk space to JindoFS. Make sure that the upper limit is greater than the lower limit.

Configure the storage policy

JindoFS provides multiple storage policies to meet different storage needs. The following table lists four available storage policies for a directory.

Policy	Description
COLD	Data has only a backup in OSS but no backups in the local cluster. This policy is suitable for storing cold data.
WARM	The default storage policy. Data has a backup in OSS and a backup in the local cluster. The local backup can accelerate read operations.
HOT	Data has a backup in OSS and multiple backups in the local cluster. Local backups can accelerate read operations on hot data.
TEMP	Data has only a backup in the local cluster. This policy is suitable for storing temporary data. The local backup can accelerate read and write operations on the temporary data. However, this may lower data reliability.

JindoFS provides a command-line tool Admin to configure the storage policy of a directory. The default storage policy is WARM. New files are stored according to the storage policy configured for the parent directory. Run the following command to configure the storage policy:

```
jindo dfsadmin -R -setStoragePolicy [path] [policy]
```

Run the following command to obtain the storage policy configured for a directory:

```
jindo dfsadmin -getStoragePolicy [path]
```



Note:

The `[path]` parameter specifies the directory. The `-R` option specifies that a recursive operation is performed to configure the same storage policy for all subdirectories of the directory.

The Admin tool provides the archive command to archive cold data.

This command allows you to explicitly evict local blocks. Assume that Hive partitions a table by the day. If the data generated a week ago in partitioned tables is infrequently accessed, you can regularly run the archive command on the directory that stores such data. Then, the backups stored in the local cluster are evicted, whereas the backups in OSS are retained.

Run the following archive command:

```
jindo dfsadmin -archive [path]
```



Note:

The `[path]` parameter specifies the directory in which the data is to be archived.

1.2 Use JindoFS in E-MapReduce V3.22.0 or later

JindoFileSystem (JindoFS) is a cloud-native file system that combines the advantages of Object Storage Service (OSS) and local storage. JindoFS is also the next-generation storage system that provides efficient and reliable storage services for cloud computing in E-MapReduce. This topic describes how to configure and use JindoFS, and its scenarios.

Overview

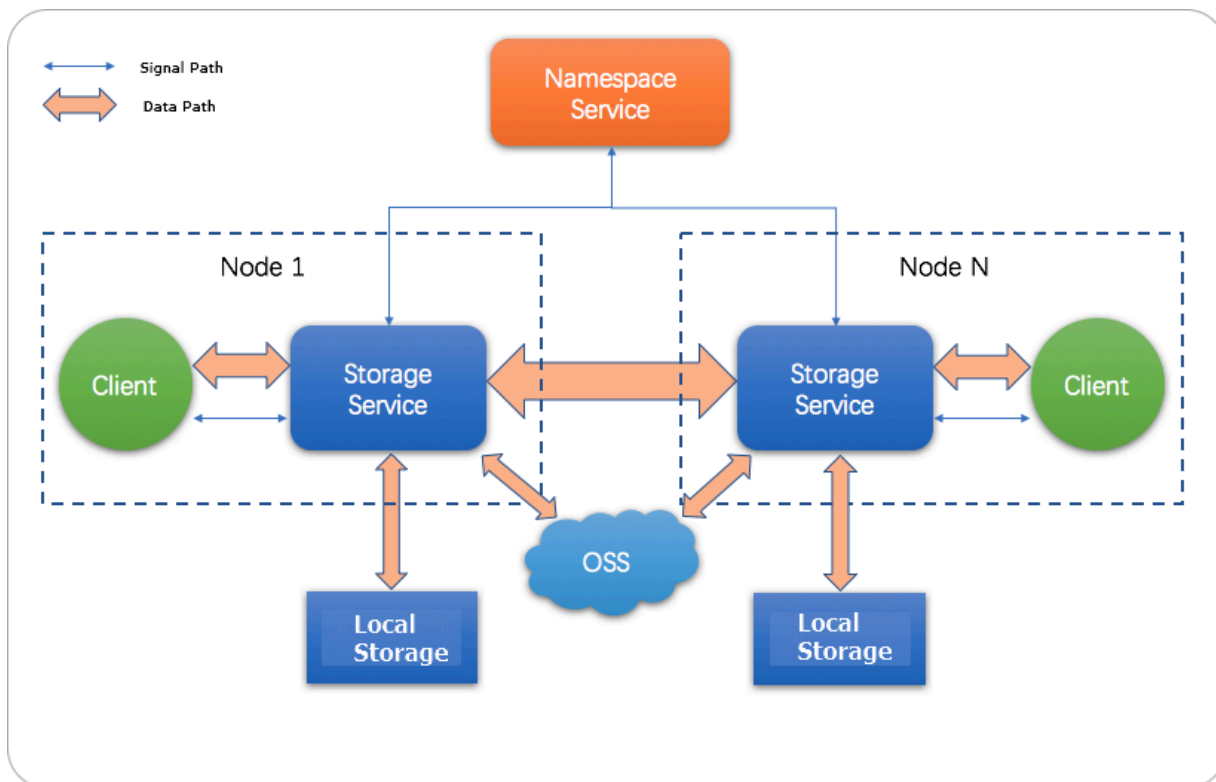
JindoFS supports the block storage mode and cache mode.

JindoFS adopts a heterogeneous multi-backup mechanism. Storage Service provides data storage capability. Data is stored in OSS to guarantee high reliability. Redundant backups are stored in the local cluster to accelerate read operations. Namespace Service manages metadata of JindoFS. In this case, metadata is queried from Namespace Service instead of OSS, which improves query performance. This query method of JindoFS is the same as that of Hadoop Distributed File System (HDFS).



Note:

- E-MapReduce V3.20.0 and later support JindoFS. To use JindoFS, select the related services when you create an E-MapReduce cluster.
- This topic describes how to use JindoFS in E-MapReduce V3.22.0 or later. For more information about how to use JindoFS in E-MapReduce V3.20.0 to V3.22.0 (V3.22.0 excluded), see [Use JindoFS in E-MapReduce V3.20.0 to V3.22.0 \(V3.22.0 excluded\)](#).







Prepare the environment

- **Create an E-MapReduce cluster**

Select E-MapReduce V3.22.0 or later. Select SmartData for Optional Services. For more information about how to create an E-MapReduce cluster, see [#unique_6](#).

Software Settings

Cluster Type: **Hadoop** Kafka ZooKeeper Druid

On-premises data queries, real-time queries, and ad-hoc queries in big data scenarios

E-MapReduce Hadoop is an open-source Hadoop ecosystem. It uses YARN to manage cluster resources, and supports massive distributed storage and computing of Hive and Spark data stored in HDFS. It supports multiple Hadoop ecosystem components, including stream computing components (Spark Streaming, Flink, and Storm), interactive query components (Presto and Impala), Oozie, and Pig. It also supports OSS storage, Kerberos authentication, and Kerberos encryption.

EMR Version: **EMR-3.22.1**

Required Services: **HDFS (2.8.5)** **YARN (2.8.5)** **Hive (3.1.1)** **Spark (2.4.3)** **Knox (1.1.0)** **Zeppelin (0.8.1)** **Tez (0.9.1)** **Ganglia (3.7.2)**
Pig (0.14.0) **Sqoop (1.4.7)** **Bigboot (2.0.1)** **OpenLDAP (2.4.44)** **Hue (4.4.0)**

Optional Services: HBase (1.4.9) ZooKeeper (3.5.5) Presto (0.221) Impala (2.12.2) Flume (1.8.0) Livy (0.6.0) Superset (0.28.1)
 Ranger (1.2.0) Flink (1.7.2) Storm (1.2.2) Phoenix (4.14.1) Analytics Zoo (0.5.0) SmartData (2.0.0) Kudu (1.10.0)
 Oozie (5.1.0)

Click to choose

- **Configure JindoFS**

JindoFS provided by SmartData uses OSS as the storage back end. Therefore, you need to set OSS-related parameters before using JindoFS. E-MapReduce provides two configuration methods. If you use the first configuration method, you need to create an E-MapReduce cluster, modify Bigboot-related parameters, and then restart SmartData for the configuration to take effect. If you use the second configuration method, you need to add custom configuration when you create

an E-MapReduce cluster. In this case, the related services are restarted based on custom parameters after the E-MapReduce cluster is created.

- Initialize parameters after the E-MapReduce cluster is created

You can set all JindoFS-related parameters in Bigboot, as shown in the following figures.

1. In the Service Configuration section, click bigboot.

Service Configuration

ALL | bigboot

Deploy Client Configuration Save

Custom Configuration

storage.data-dirs.watermark.low.ratio 0.3 ?

jfs.namespaces test ?

storage.data-dirs.watermark.high.ratio 0.6 ?

2. Click Custom Configuration.

Add Configuration Item

* Key	* Value	Description	Actions
jfs.namespaces.test.uri	oss://oss-bucket/oss-dir		Delete
jfs.namespaces.test.mode	block		Delete
jfs.namespaces.test.oss.acc	XXXX		Delete
jfs.namespaces.test.oss.acc	XXXX		Delete

Add



OK Cancel




Note:

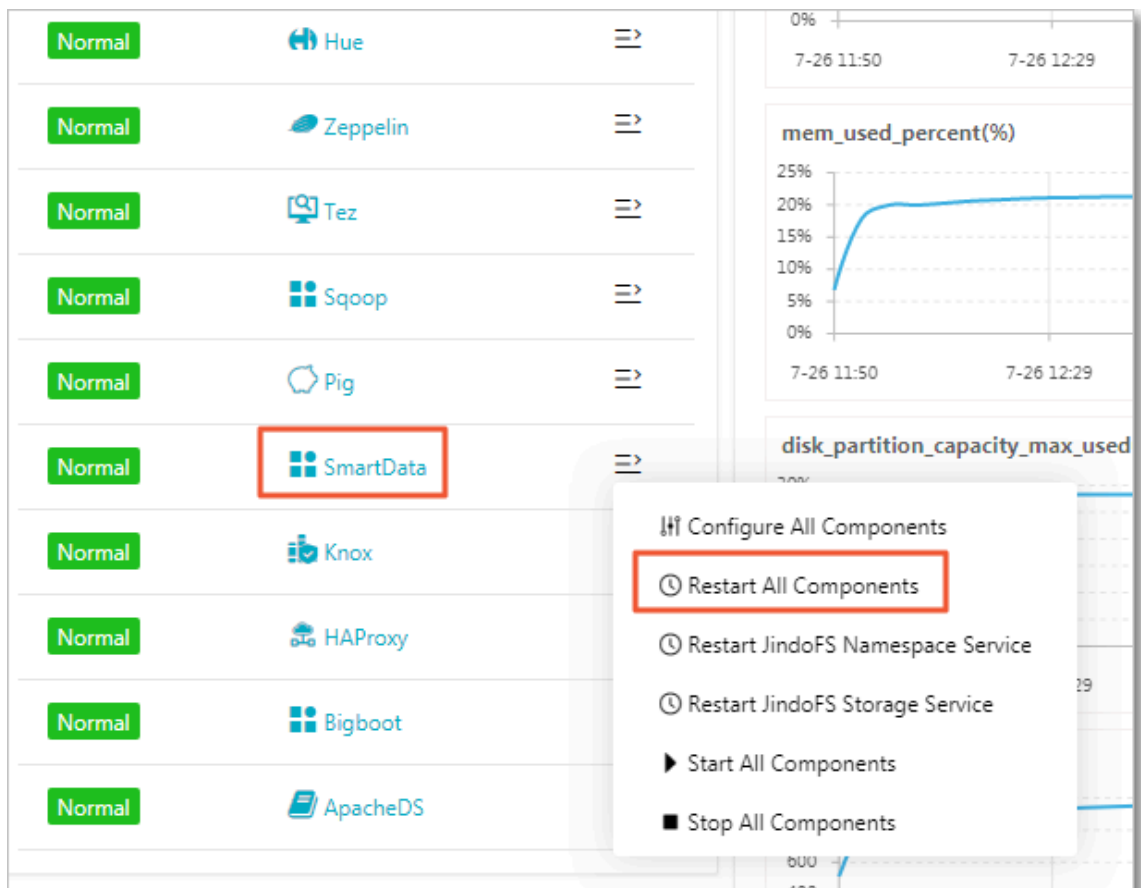
- The parameters framed in red in the preceding figures are required.

■ JindoFS supports multiple namespaces. A namespace named test is used in this topic.

Parameter	Description	Example
<code>jfs.namespaces</code>	The namespace supported by JindoFS . Separate multiple namespaces with commas (,).	test
<code>jfs.namespaces.test.uri</code>	The storage back end of the test namespace.	<code>oss://oss-bucket/oss-dir</code> <div>  Note: You can set the value to a directory in an OSS bucket. In this case, this directory serves as the root directory, in which the test namespace reads and writes data. </div>
<code>jfs.namespaces.test.mode</code>	The storage mode of the test namespace.	block <div>  Note: JindoFS supports the block storage mode and cache mode. </div>

Parameter	Description	Example
<code>jfs.namespaces.test.oss.access.key</code>	The AccessKey ID used to access the OSS bucket that serves as the storage back end.	XXXX
<code>jfs.namespaces.test.oss.access.secret</code>	The AccessKey secret used to access the OSS bucket that serves as the storage back end.	 Note: We recommend that you select an OSS bucket in the same region and under the same account as the storage back end of the E-MapReduce cluster for better performance and stability. In this case, the E-MapReduce cluster can access the OSS bucket without using the AccessKey ID and AccessKey secret.

Save and deploy the JindoFS configuration. Restart all components in SmartData to use JindoFS.



The screenshot displays the SmartData management console. On the left, a list of components is shown, each with a status indicator (green 'Normal') and a restart icon (three horizontal lines). The components listed are Hue, Zeppelin, Tez, Sqoop, Pig, SmartData (highlighted with a red box), Knox, HAProxy, Bigboot, and ApacheDS. On the right, there are two line graphs: 'mem_used_percent(%)' showing memory usage rising from 5% to 20% and 'disk_partition_capacity_max_used' showing disk usage. A configuration menu is open over the SmartData component, listing actions: 'Configure All Components', 'Restart All Components' (highlighted with a red box), 'Restart JindoFS Namespace Service', 'Restart JindoFS Storage Service', 'Start All Components', and 'Stop All Components'.

- Add custom configuration when creating an E-MapReduce cluster

You can add custom configuration when creating an E-MapReduce cluster. For example, you want to create an E-MapReduce cluster in the same region as an OSS bucket to access OSS without using an AccessKey. As shown in the following figure, turn on Custom Software Settings. Add the following configuration to the field in the Advanced Settings section to customize parameters for the test namespace:

```
[
{
  "ServiceName":"BIGBOOT",
  "FileName":"bigboot",
  "ConfigKey":"jfs.namespaces","ConfigValue":"test"
},
{
  "ServiceName":"BIGBOOT",
  "FileName":"bigboot",
  "ConfigKey":"jfs.namespaces.test.uri",
  "ConfigValue":"oss://oss-bucket/oss-dir"
},
{
  "ServiceName":"BIGBOOT",
  "FileName":"bigboot",
  "ConfigKey":"jfs.namespaces.test.mode",
  "ConfigValue":"block"
}
]
```

The screenshot displays the E-MapReduce console interface. The top section, 'Software Settings', shows the 'Cluster Type' as 'Hadoop' and lists various services like HDFS, YARN, Hive, Spark, etc. Below this, the 'Advanced Settings' section is expanded, showing 'Kerberos Mode' as disabled and 'Custom Software Settings' as enabled. A text area at the bottom contains a JSON configuration for customizing parameters for the test namespace.

Software Settings

Cluster Type: Hadoop Kafka ZooKeeper Druid

On-premises data queries, real-time queries, and ad-hoc queries in big data scenarios

E-MapReduce Hadoop is an open-source Hadoop ecosystem. It uses YARN to manage cluster resources, and supports massive distributed storage and computing of Hive and Spark data stored in HDFS. It supports multiple Hadoop ecosystem components, including stream computing components (Spark Streaming, Flink, and Storm), interactive query components (Presto and Impala), Oozie, and Pig. It also supports OSS storage, Kerberos authentication, and Kerberos encryption.

EMR Version: EMR-3.22.1

Required Services: HDFS (2.8.5) YARN (2.8.5) Hive (3.1.1) Spark (2.4.3) Knox (1.1.0) Zeppelin (0.8.1) Tez (0.9.1) Ganglia (3.7.2) Pig (0.14.0) Sqoop (1.4.7) Bigboot (2.0.1) OpenLDAP (2.4.44) Hue (4.4.0)

Optional Services: HBase (1.4.9) ZooKeeper (3.5.5) Presto (0.221) Impala (2.12.2) Flume (1.8.0) Livy (0.6.0) Superset (0.28.1) Ranger (1.2.0) Flink (1.7.2) Storm (1.2.2) Phoenix (4.14.1) Analytics Zoo (0.5.0) SmartData (2.0.0) Kudu (1.10.0) Oozie (5.1.0)

Click to choose

Advanced Settings

Kerberos Mode: ☐ All components in a cluster that is enabled with the Kerberos mode will use Kerberos authentication to validate requests. For more information, see [Kerberos introduction](#)

Custom Software Settings: ☒ Before creating a cluster, you can use a JSON file to customize parameters for the cluster components. For more information, see [Software Settings](#)

```
[{"ServiceName":"BIGBOOT","FileName":"bigboot","ConfigKey":"jfs.namespaces","ConfigValue":"test"},
{"ServiceName":"BIGBOOT","FileName":"bigboot","ConfigKey":"jfs.namespaces.test.uri","ConfigValue":"oss://oss-bucket/oss-dir"},
{"ServiceName":"BIGBOOT","FileName":"bigboot","ConfigKey":"jfs.namespaces.test.mode","ConfigValue":"block"}]
```

Use JindoFS

The use of JindoFS is similar to that of HDFS. JindoFS also provides a prefix. To use JindoFS, you only need to replace the `hdfs` prefix with the `jfs` prefix.

Currently, JindoFS supports most of the computing components in the E-MapReduce cluster, including Hadoop, Hive, Spark, Flink, Presto, and Impala.

Examples:

- Run shell commands

```
hadoop fs -ls jfs://your-namespace/  
hadoop fs -mkdir jfs://your-namespace/test-dir  
hadoop fs -put test.log jfs://your-namespace/test-dir/  
hadoop fs -get jfs://your-namespace/test-dir/test.log . /
```

- Run a MapReduce job

```
hadoop jar /usr/lib/hadoop-current/share/hadoop/mapreduce/hadoop-  
mapreduce-examples-2.8.5.jar teragen -Dmapred.map.tasks=1000  
10737418240 jfs://your-namespace/terasort/input  
hadoop jar /usr/lib/hadoop-current/share/hadoop/mapreduce/hadoop-  
mapreduce-examples-2.8.5.jar terasort -Dmapred.reduce.tasks=1000  
jfs://your-namespace/terasort/input jfs://your-namespace/terasort/  
output
```

- Run a Spark SQL test

```
CREATE EXTERNAL TABLE IF NOT EXISTS src_jfs (key INT, value STRING)  
location 'jfs://your-namespace/Spark_sql_test/';
```

Control disk space usage

The back end of JindoFS is based on OSS that is capable of storing large amounts of data. However, the storage capacity of local disks is limited. Therefore, JindoFS releases data backups that are less frequently accessed. Alibaba Cloud uses `node.data-dirs.watermark.high.ratio` and `node.data-dirs.watermark.low.ratio` to adjust the space usage of local disks. The values of both parameters are in the range of 0 to 1 to indicate the percentage of space usage. JindoFS uses the total storage capacity of all data disks by default. The `node.data-dirs.watermark.high.ratio` parameter specifies the upper limit of space usage on each disk. Less frequently accessed data stored on a disk is released if the space used by JindoFS reaches the upper limit. The `node.data-dirs.watermark.low.ratio` parameter specifies the lower limit of space usage on each disk. After the space usage of a disk reaches the upper limit, less frequently accessed data is released until the space usage of the disk reaches the lower limit. You can set the upper limit and lower limit to adjust and

assign disk space to JindoFS. Make sure that the upper limit is greater than the lower limit.

Configure the storage policy

JindoFS provides multiple storage policies to meet different storage needs. The following table lists four available storage policies for a directory.

Policy	Description
COLD	Data has only a backup in OSS but no backups in the local cluster. This policy is suitable for storing cold data.
WARM	The default storage policy. Data has a backup in OSS and a backup in the local cluster. The local backup can accelerate read operations.
HOT	Data has a backup in OSS and multiple backups in the local cluster. Local backups can accelerate read operations on hot data.
TEMP	Data has only a backup in the local cluster. This policy is suitable for storing temporary data. The local backup can accelerate read and write operations on the temporary data. However, this may lower data reliability.

JindoFS provides a command-line tool Admin to configure the storage policy of a directory. The default storage policy is WARM. New files are stored according to the storage policy configured for the parent directory. Run the following command to configure the storage policy:

```
jindo dfsadmin -R -setStoragePolicy [path] [policy]
```

Run the following command to obtain the storage policy configured for a directory:

```
jindo dfsadmin -getStoragePolicy [path]
```



Note:

The `[path]` parameter specifies the directory. The `-R` option specifies that a recursive operation is performed to configure the same storage policy for all subdirectories of the directory.

Use the Admin tool

JindoFS supports the archive and jindo commands of the Admin tool.

- The Admin tool provides the archive command to archive cold data.

This command allows you to explicitly evict local blocks. Assume that Hive partitions a table by the day. If the data generated a week ago in partitioned tables is infrequently accessed, you can regularly run the archive command on the directory that stores such data. Then, the backups stored in the local cluster are evicted, whereas the backups in OSS are retained.

Run the following archive command:

```
jindo dfsadmin -archive [path]
```



Note:

The `[path]` parameter specifies the directory in which the data is to be archived.

- The Admin tool provides the jindo commands to manage metadata of JindoFS for Namespace Service.

```
jindo dfsadmin [-options]
```



Note:

You can run the `jindo dfsadmin --help` command to obtain help information.

The Admin tool provides the diff and sync commands for the cache mode.

- The diff command is used to display the difference between the data stored in the local cluster and that in OSS.

```
jindo dfsadmin -R -diff [path]
```



Note:

By default, you can use the diff command to display the difference between the metadata stored in the local cluster and that in the subdirectories of the directory specified by the `[path]` parameter. The `-R` option specifies that a recursive operation is performed to compare the metadata stored in the local cluster with that stored in all subdirectories of the directory specified by the `[path]` parameter.

- The `sync` command is used to synchronize the metadata between the local cluster and OSS.

```
jindo dfsadmin -R -sync [path]
```

**Note:**

The `[path]` parameter specifies the directory in which the metadata is to be synchronized. By default, you can use the `sync` command to synchronize the metadata in the subdirectories of the directory specified by the `[path]` parameter to the local cluster. The `-R` option specifies that a recursive operation is performed to synchronize the metadata in all subdirectories of the directory specified by the `[path]` parameter.

1.3 Use the block storage mode

This topic describes the block storage mode of JindoFileSystem (JindoFS) and its scenarios.

Overview

Block storage is the most efficient mode to read and write data and query metadata. In addition, it supports Hadoop Distributed File System (HDFS) semantics related to data locality. JindoFS also provides an external client so that you can access JindoFS from the outside of an E-MapReduce cluster.

JindoFS uses Object Storage Service (OSS) as the storage back end. In block storage mode, JindoFS stores data as blocks in OSS and uses Namespace Service to maintain metadata. This guarantees high performance when you read and write data or query metadata.

Scenarios

E-MapReduce has three storage systems: E-MapReduce OssFileSystem, E-MapReduce HDFS, and E-MapReduce JindoFS. Among them, OssFileSystem and JindoFS store data in the cloud. The following table compares the features of three E-MapReduce storage systems and Hadoop support for Alibaba Cloud OSS.

Feature	Hadoop support for Alibaba Cloud OSS	E-MapReduce OssFileSystem	E-MapReduce HDFS	E-MapReduce JindoFS
Storage capacity	Tremendous	Tremendous	Depends on the E-MapReduce cluster scale	Tremendous
Reliability	High	High	High	High
Factor that affects throughput	Server	I/O performance of caches on disks in the E-MapReduce cluster	I/O performance of disks in the E-MapReduce cluster	I/O performance of disks in the E-MapReduce cluster
Metadata query efficiency	Low	Medium	High	High
Scale-out operation	Easy	Easy	Easy	Easy
Scale-in operation	Easy	Easy	Requires node decommission	Easy
Data locality	None	Weak	Strong	Medium

The block storage mode of JindoFS has the following features:

- JindoFS offers tremendous and scalable storage capacity by using OSS as the storage back end. The storage capacity is independent of the E-MapReduce cluster scale. The local cluster can be scaled in or out as required.
- JindoFS stores a certain amount of backup data in the local cluster to accelerate read operations. This improves the throughput by using limited local storage capacity, especially for Write Once Read Many (WORM) solutions.
- JindoFS provides efficient metadata query similar to HDFS. Compared with OssFileSystem, JindoFS saves much time in metadata query. In addition, JindoFS avoids system instability when data and metadata are frequently accessed.
- JindoFS moves computation as close as possible to data. This reduces the load on network transmission and improves the read performance.

Configure JindoFS

You can set all JindoFS related-parameters in Bigboot, as shown in the following figure.

Service Configuration

ALL | bigboot

Deploy Client Configuration Save

Custom Configuration

storage.data-dirs.watermark.low.ratio 0.3 ?

jfs.namespaces test ↶ ?

storage.data-dirs.watermark.high.ratio 0.6 ?



* Key	* Value	Description	Actions
jfs.namespaces.test.uri	oss://oss-bucket/oss-dir		Delete
jfs.namespaces.test.mode	block		Delete
jfs.namespaces.test.oss.acc	XXXX		Delete
jfs.namespaces.test.oss.acc	XXXX		Delete

Add

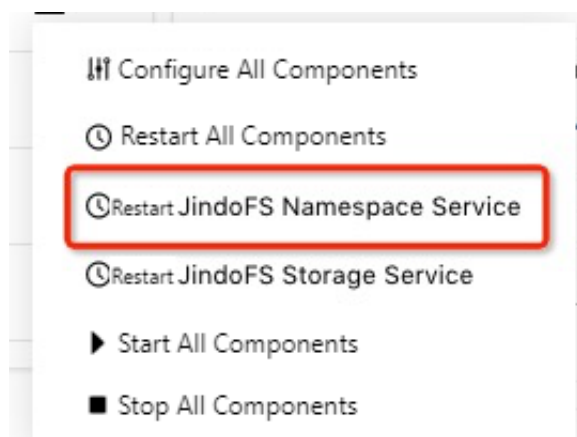
**Note:**

- The parameters framed in red in the preceding figure are required.
- JindoFS supports multiple namespaces. A namespace named test is used in this topic.

Parameter	Description	Example
jfs.namespaces	The namespace supported by JindoFS. Separate multiple namespaces with commas (,).	test

Parameter	Description	Example
<code>jfs.namespaces.test.uri</code>	The storage back end of the test namespace.	<code>oss://oss-bucket/oss-dir</code>  Note: You can set the value to a directory in an OSS bucket. In this case, this directory serves as the root directory, in which the test namespace reads and writes data.
<code>jfs.namespaces.test.mode</code>	The storage mode of the test namespace.	block
<code>jfs.namespaces.test.oss.access.key</code>	The AccessKey ID used to access the OSS bucket that serves as the storage back end.	xxxx  Note: We recommend that you select an OSS bucket in the same region and under the same account as the storage back end of the E-MapReduce cluster for better performance and stability. In this case, the E-MapReduce cluster can access the OSS bucket without using the AccessKey ID and AccessKey secret.
<code>jfs.namespaces.test.oss.access.secret</code>	The AccessKey secret used to access the OSS bucket that serves as the storage back end.	

Save and deploy the JindoFS configuration. Restart Namespace Service in SmartData to use JindoFS.



Configure the storage policy

JindoFS provides multiple storage policies to meet different storage needs. The following table lists four available storage policies for a directory.

Policy	Description
COLD	Data has only a backup in OSS but no backups in the local cluster. This policy is suitable for storing cold data.
WARM	The default storage policy. Data has a backup in OSS and a backup in the local cluster. The local backup can accelerate read operations.
HOT	Data has a backup in OSS and multiple backups in the local cluster. Local backups can accelerate read operations on hot data.
TEMP	Data has only a backup in the local cluster. This policy is suitable for storing temporary data. The local backup can accelerate read and write operations on the temporary data. However, this may lower data reliability.

JindoFS provides a command-line tool Admin to configure the storage policy of a directory. The default storage policy is WARM. New files are stored according to the

storage policy configured for the parent directory. Run the following command to configure the storage policy:

```
jindo dfsadmin -R -setStoragePolicy [path] [policy]
```

Run the following command to obtain the storage policy configured for a directory:

```
jindo dfsadmin -getStoragePolicy [path]
```

**Note:**

The *[path]* parameter specifies the directory. The *-R* option specifies that a recursive operation is performed to configure the same storage policy for all subdirectories of the directory.

The Admin tool provides the archive command to archive cold data.

This command allows you to explicitly evict local blocks. Assume that Hive partitions a table by the day. If the data generated a week ago in partitioned tables is infrequently accessed, you can regularly run the archive command on the directory that stores such data. Then, the backups stored in the local cluster are evicted, whereas the backups in OSS are retained.

Run the following archive command:

```
jindo dfsadmin -archive [path]
```

**Note:**

The *[path]* parameter specifies the directory in which the data is to be archived.

1.4 Use the cache mode

This topic describes the cache mode of JindoFileSystem (JindoFS) and its scenarios.

Overview

In cache mode, JindoFS stores data files as objects in Object Storage Service (OSS) and caches data and metadata of these files in the local cluster based on the requirements for accessing these files. This accelerates read and write operations on data and metadata. In addition, the cache mode provides multiple policies for you to synchronize metadata as required.

Scenarios

The cache mode is compatible with original OSS semantics. In cache mode, JindoFS stores data files as objects in OSS and caches data and metadata in the local cluster. This guarantees that JindoFS is compatible with the OSS client, E-MapReduce OssFileSystem, and other OSS interactive applications. You can also access data that exists in OSS before you configure JindoFS, with no need to migrate or convert data. In addition, local caches can be used to accelerate read and write operations on data and metadata.

Configure JindoFS

You can set all JindoFS related-parameters in Bigboot, as shown in the following figure.

Service Configuration

ALL | bigboot

Deploy Client Configuration Save

Custom Configuration

storage.data-dirs.watermark.low.ratio 0.3 ?

jfs.namespaces test ↶ ?

storage.data-dirs.watermark.high.ratio 0.6 ?



* Key	* Value	Description	Actions
jfs.namespaces.test.uri	oss://oss-bucket/oss-dir		Delete
jfs.namespaces.test.mode	block		Delete
jfs.namespaces.test.oss.acc	XXXX		Delete
jfs.namespaces.test.oss.acc	XXXX		Delete

Add

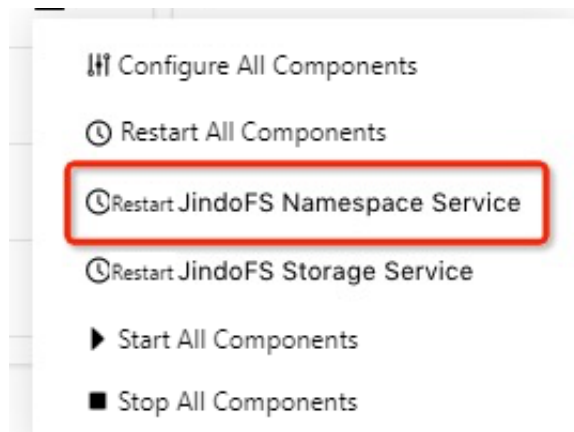


Note:

- The parameters framed in red in the preceding figure are required.
- JindoFS supports multiple namespaces. A namespace named test is used in this topic.

Parameter	Description	Example
<code>jfs.namespaces</code>	The namespace supported by JindoFS. Separate multiple namespaces with commas (,).	test
<code>jfs.namespaces.test.uri</code>	The storage back end of the test namespace.	<code>oss://oss-bucket/</code>  Note: You can set the value to a directory in an OSS bucket. In this case, this directory serves as the root directory, in which the test namespace reads and writes data. Generally, you can set the value to an OSS bucket to guarantee that the path is the same as that in OSS.
<code>jfs.namespaces.test.mode</code>	The storage mode of the test namespace.	cache
<code>jfs.namespaces.test.oss.access.key</code>	The AccessKey ID used to access the OSS bucket that serves as the storage back end.	xxxx  Note: We recommend that you select an OSS bucket in the same region and under the same account as the storage back end of the E-MapReduce cluster for better performance and stability. In this case, the E-MapReduce cluster can access the OSS bucket without using the AccessKey ID and AccessKey secret.
<code>jfs.namespaces.test.oss.access.secret</code>	The AccessKey secret used to access the OSS bucket that serves as the storage back end.	

Save and deploy the JindoFS configuration. Restart Namespace Service in SmartData to use JindoFS.



Configure the metadata synchronization policy

In cache mode, you may find that some data exists in OSS before you configure JindoFS. After JindoFS is configured, data and metadata are synchronized to JindoFS for future access. At the same time, you can configure the synchronization policy so that JindoFS caches data and metadata in the local cluster. Policies for synchronizing metadata include two types: the interval policy and the loading policy.

- Interval policy:

You can set the `namespace.sync.interval` parameter to specify the synchronization interval. The default value is -1, which indicates that JindoFS does not synchronize metadata from OSS.

- If you set this parameter to 0, JindoFS synchronizes metadata from OSS each time data is accessed.
- If you set this parameter to a value greater than 0, JindoFS synchronizes metadata from OSS at intervals of the set value in units of seconds.



Note:

For example, if you set this parameter to 5, JindoFS synchronizes metadata from OSS every 5 seconds.

- Loading policy:

You can set the `namespace.sync.loadtype` parameter to specify the loading policy. Valid values are never, once, and always. A value of never indicates that

JindoFS never synchronizes metadata from OSS. A value of once indicates that JindoFS synchronizes metadata from OSS only once. This is the default value. A value of always indicates that JindoFS synchronizes metadata from OSS each time data is accessed.

**Note:**

The namespace.sync.loadtype parameter only takes effect when you do not specify the namespace.sync.interval parameter.

1.5 Use the external client

This topic describes the external client of JindoFileSystem (JindoFS) and its scenarios.

Overview

JindoFS provides an external client so that you can access JindoFS from the outside of an E-MapReduce cluster. If you want to access JindoFS from the external client, make sure that JindoFS is in block storage mode. Currently, you cannot access JindoFS from the external client if JindoFS is in cache mode. To access JindoFS in cache mode from the outside of an E-MapReduce cluster, use the common Object Storage Service (OSS) client because the cache mode is compatible with original OSS semantics.

Scenarios

The external client of JindoFS is compatible with Hadoop Distributed File System (HDFS). To access data stored in JindoFS from the external client, make sure that your application is connected to Namespace Service of JindoFS. However, you cannot access cached data in the local cluster from the external client. In this case, the performance of data access from the external client is not as efficient as that from the inside of an E-MapReduce cluster.

Configure the external client

Make sure that the namespace supported by JindoFS in block storage mode is configured. For more information, see [Use the block storage mode](#).

1. Obtain the Bigboot package.

Access the `/usr/lib/bigboot-current` directory in the E-MapReduce cluster to obtain the Bigboot package.



Note:

The Bigboot package is developed based on native code, which may be incompatible with your operating system. In this case, [submit a ticket](#) if relevant code needs to be compiled again.

2. Set up the environment.

Set the `BIGBOOT_HOME` variable to the root directory for installing Bigboot on your device. Add the `ext` and `lib` directories in the root directory to the classpath parameter of your component for processing big data, such as Hadoop or Spark.

3. Copy the configuration file `bigboot.cfg.external` from the `/usr/lib/bigboot-current/conf/` directory in the E-MapReduce cluster to the installation directory `conf/` on your device.

4. Configure Namespace Service.

- `client.namespace.rpc.port`: the port for listening on Namespace Service.
- `client.namespace.rpc.address`: the endpoint for listening on Namespace Service.



Note:

By default, E-MapReduce sets the preceding two parameters in the Bigboot configuration file.

5. Set data access parameters.

- `client.namespaces.{ YourNamespace }.oss.access.bucket`: **the OSS bucket to be accessed.**
- `client.namespaces.{ YourNamespace }.oss.access.endpoint`: **the endpoint for accessing the OSS bucket.**
- `client.namespaces.{ YourNamespace }.oss.access.key`: **the AccessKey ID used to access the OSS bucket.**
- `client.namespaces.{ YourNamespace }.oss.access.secret`: **the AccessKey secret used to access the OSS bucket.**



Note:

In the preceding parameters, {YourNamespace} specifies the namespace that you want to access from the external client. In this topic, a namespace named test is used.

Configuration example:

```
client.namespace.rpc.port = 8101
client.namespace.rpc.address = {RPC_Address}
client.namespaces.test.oss.access.bucket = {YourOssBucket}
client.namespaces.test.oss.access.endpoint = {YourOssEndpoint}
client.namespaces.test.oss.access.key = {YourOssKey}
client.namespaces.test.oss.access.secret = {YourOssSecret}
```

Verify the configuration

- **Run the following command to check whether the test namespace is configured correctly:**

```
hdfs dfs -ls jfs://test/
```

- **Run the following commands to check whether data can be uploaded to or downloaded from the test namespace:**

```
hdfs dfs -put /etc/hosts jfs://test/
```

```
hdfs dfs -get jfs://test/hosts
```

2 JindoFS ecosystem

2.1 Migrate data from HDFS to JindoFS

This topic describes how to migrate data from Hadoop Distributed File System (HDFS) to JindoFileSystem (JindoFS) that stores data in Object Storage Service (OSS).

Migrate data

- Use Hadoop FS shell commands

You can use File System (FS) shell commands to migrate a small amount of data:

- `hadoop dfs -cp hdfs://emr-cluster/README.md jfs://emr-jfs/`
- `hadoop dfs -cp oss://oss_bucket/README.md jfs://emr-jfs/`

- Use Hadoop DistCp

You can use DistCp, a built-in tool of Hadoop, to migrate a large amount of data:

- `hadoop distcp hdfs://emr-cluster/files jfs://emr-jfs/output/`
- `hadoop distcp oss://oss_bucket/files jfs://emr-jfs/output/`



Note:

For more information about DistCp parameters, see [DistCp Version2 Guide](#).

Use the cache mode

In cache mode, JindoFS stores data files as objects in OSS without changing the metadata and data. When you access these OSS objects, JindoFS can cache data and metadata of these OSS objects in the local cluster so that you can quickly access them next time. For more information, see [Use the cache mode](#).

2.2 Use MapReduce to process data in JindoFS

This topic describes how to use MapReduce to read and write data in JindoFileSystem (JindoFS).

JindoFS configuration

For example, a namespace named `emr-jfs` is created with the following configuration:

- `jfs.namespaces=emr-jfs`
- `jfs.namespaces.emr-jfs.uri=oss://oss-bucket/oss-dir`
- `jfs.namespaces.emr-jfs.mode=block`

MapReduce introduction

Generally, Hadoop MapReduce jobs read and write data through Hadoop Distributed File System (HDFS). JindoFS is compatible with HDFS. You can set the input and output directories of MapReduce jobs to directories in JindoFS. In this case, MapReduce jobs can read and write data in JindoFS.

Hadoop MapReduce is a software framework for easily writing applications. Applications based on Hadoop MapReduce can process multi-terabyte datasets in parallel in large clusters that consist of thousands of nodes in a reliable and fault-tolerant manner. A MapReduce job usually splits the input dataset into independent blocks that are processed by map tasks in parallel. The MapReduce framework sorts the output of map tasks and writes the sorted output to reduce tasks. Both the input and the output of the job are stored in a file system. The MapReduce framework is responsible for scheduling tasks, monitoring tasks, and rerunning failed tasks.

Job input and output

In applications, the input and output directories of MapReduce jobs are specified. Map and reduce functions are implemented based on appropriate methods or abstract classes. The Hadoop job client submits MapReduce jobs and their configuration to ResourceManager. Then, ResourceManager schedules tasks. In this case, you can modify the input and output directories of MapReduce jobs to directories in JindoFS to read and write data in JindoFS.

Examples

The following examples show how to modify the input and output directories of a MapReduce job to read and write data in JindoFS.

- **Teragen**

In this MapReduce job, use Teragen to generate data in a specified number of rows in the specified directory:

```
hadoop jar /usr/lib/hadoop-current/share/hadoop/mapreduce/hadoop-mapreduce-examples-*.jar teragen <num rows> <output dir>
```

Replace the output directory with a directory in JindoFS to write data to JindoFS:

```
hadoop jar /usr/lib/hadoop-current/share/hadoop/mapreduce/hadoop-mapreduce-examples-*.jar teragen 100000 jfs://emr-jfs/teragen_data_0
```

- **Terasort**

In this MapReduce job, use Terasort to sort data in the specified input directory and write the sorted data to the specified output directory:

```
hadoop jar /usr/lib/hadoop-current/share/hadoop/mapreduce/hadoop-mapreduce-examples-*.jar terasort <in> <out>
```

Replace the input and output directories with directories in JindoFS to process data in JindoFS:

```
hadoop jar /usr/lib/hadoop-current/share/hadoop/mapreduce/hadoop-mapreduce-examples-*.jar terasort jfs://emr-jfs/teragen_data_0/ jfs://emr-jfs/terasort_data_0
```

2.3 Use Hive to query data in JindoFS

Apache Hive is a distributed SQL query engine widely used in the Hadoop ecosystem. Hive manages data in databases, tables, and partitions. You can specify the storage location to query data at the storage backend.

JindoFS configuration

For example, a namespace named `emr-jfs` is created with the following configuration:

- `jfs.namespaces=emr-jfs`
- `jfs.namespaces.emr-jfs.uri=oss://oss-bucket/oss-dir`
- `jfs.namespaces.emr-jfs.mode=block`

Specify the storage location for data warehouses, databases, tables, or partitions

- **Specify the storage location for a data warehouse**

The hive-site configuration file contains the `hive.metastore.warehouse.dir` parameter. This parameter specifies the directory in which a Hive data warehouse stores data. For example, set this parameter to `jfs://emr-jfs/user/hive/warehouse`.

- **Specify the storage location for a database**

A Hive database has a storage location. The location is also used as the default storage location of tables in the database. The parameter that specifies the storage location is optional when you create a database. By default, the storage location of a database is the value of the `hive.metastore.warehouse.dir` parameter in the hive-site file added with the database name. You can run the following SQL statements to specify the storage location.

- **Set the storage location to a directory in JindoFileSystem (JindoFS) when you create a database:**

```
CREATE DATABASE database_name
LOCATION
'jfs://namespace/database_dir';
```

For example, to create a Hive database named `database_on_jindofs` and set the storage location to `jfs://emr-jfs/warehouse/database_on_jindofs`, run the following statement:

```
CREATE DATABASE database_on_jindofs
LOCATION
'jfs://emr-jfs/hive/warehouse/database_on_jindofs';
```

- **Modify the storage location of a database to a directory in JindoFS:**
 1. **Run the following SHOW CREATE statement to query the storage location of a database:**

```
SHOW CREATE DATABASE database_name;
```

2. **View the returned storage location. By default, the storage location of a database is that of its data warehouse added with the database name.**

```
CREATE DATABASE `database_name`
LOCATION
```

```
'hdfs://emr-jfs/user/hive/warehouse/database_name.db'
```

3. **Modify the storage location to a directory in JindoFS. This operation does not affect existing tables. If you do not specify the storage location for a new table, the modified location is used as the default storage location of the table.**

```
ALTER DATABASE database_name SET LOCATION jfs_path;
```

- **Specify the storage location for a table or partition**

Similar to the storage location of a database, that of a table or partition is also specified based on the upper-level storage location. Data in a non-partitioned table is stored in the storage location of the table. Data in a partitioned table is stored in the storage location of respective partitions. You can run the following SQL statements to specify the storage location.

- **Set the storage location to a directory in JindoFS when you create a database:**

```
CREATE DATABASE database_name
LOCATION
'jfs://namespace/database_dir';
```

For example, to create a Hive database named database_on_jindofs and set the storage location to jfs://emr-jfs/warehouse/database_on_jindofs, run the following statement:

```
CREATE DATABASE database_on_jindofs
LOCATION
'jfs://emr-jfs/hive/warehouse/database_on_jindofs';
```

- **Modify the storage location of a database to a directory in JindoFS:**

1. **Run the following SHOW CREATE statement to query the storage location of a database:**

```
SHOW CREATE DATABASE database_name;
```

2. **View the returned storage location. By default, the storage location of a database is that of its data warehouse added with the database name.**

```
CREATE DATABASE `database_name`
LOCATION
'hdfs://emr-jfs/user/hive/warehouse/database_name.db'
```

3. **Modify the storage location to a directory in JindoFS. This operation does not affect existing tables. If you do not specify the storage location for a new**

table, the modified location is used as the default storage location of the table.

```
ALTER DATABASE database_name SET LOCATION jfs_path;
```

Query data in the scratch directory

Hive stores temporary output files and job plans to the scratch directory. You can set the `hive.exec.scratchdir` parameter in the hive-site configuration file to a directory in JindoFS. You can also set the parameter by running the following commands:

```
bin/hive --hiveconf hive.exec.scratchdir=jfs://emr-jfs/scratch_dir
```

or

```
set hive.exec.scratchdir=jfs://emr-jfs/scratch_dir;
```

2.4 Use Spark to process data in JindoFS

Spark processes data in JindoFileSystem (JindoFS) by calling methods or using Spark SQL to read data from tables stored in JindoFS.

JindoFS configuration

For example, a namespace named `emr-jfs` is created with the following configuration:

- `jfs.namespaces=emr-jfs`
- `jfs.namespaces.emr-jfs.uri=oss://oss-bucket/oss-dir`
- `jfs.namespaces.emr-jfs.mode=block`

Process data in JindoFS

- **Call methods**

The read and write operations performed by Spark in JindoFS are similar to those in other file systems. For example, to access data in JindoFS, use a

directory with the jfs prefix in the following resilient distributed dataset (RDD) operation:

```
val a = sc.textFile("jfs://emr-jfs/README.md")
```

```
Welcome to  
      _  
     / \_/_/_/_/_/_/\_  
    -\ V -V- '\_/ ,'  
   /\_/.~^~.^./\^.\ version 2.4.3  
    /\
```

Using Scala version 2.11.12 (OpenJDK 64-Bit Server VM, Java 1.8.0_151)
Type in expressions to have them evaluated.
Type :help for more information.

```
scal> val a = sc.textFile("jfs://emr-jfs/README.md")  
a: org.apache.spark.rdd.RDD[String] = jfs://emr-jfs/README.md MapPartitionsRDD[1] at textFile at <console>:24
```

To write data to JindoFS, call the following method:

```
scala> a.collect().saveAsTextFile("jfs://emr-jfs/output")
```

- **Use Spark SQL**

Set the parameter that specifies the storage location to a directory in JindoFS when you create databases, tables, and partitions. For more information, see [#unique_15](#). Then, you can query data from tables stored in JindoFS.

2.5 Use Flink to process data in JindoFS

This topic describes how to use Flink to process data in JindoFileSystem (JindoFS).

JindoFS configuration

For example, a namespace named `emr-jfs` is created with the following configuration:

- `jfs.namespaces=emr-jfs`
- `jfs.namespaces.emr-jfs.uri=oss://oss-bucket/oss-dir`
- `jfs.namespaces.emr-jfs.mode=block`

Use JindoFS

You can set the input and output directories of Flink jobs to directories in a namespace supported by JindoFS. In this case, Flink jobs can read and write data in JindoFS.

For example, to store job data in Hadoop Distributed File System (HDFS), run the following command:

```
flink run -m yarn-cluster -yD taskmanager.network.memory.fraction=0.4
-yD akka.ask.timeout=60s -yjm 2048 -ytm 2048 -ys 4 -yn 14 -c xxx.xxx
.FlinkWordCount -p 56 XXX.jar --input hdfs:///test//large-input-flink
--output hdfs:///runjob/test/large-output-flink"
```

To store job data in JindoFS, run the following command:

```
flink run -m yarn-cluster -yD taskmanager.network.memory.fraction=0.4
-yD akka.ask.timeout=60s -yjm 2048 -ytm 2048 -ys 4 -yn 14 -c xxx.xxx
.FlinkWordCount -p 56 XXX.jar --input jfs://emr-jfs/test/large-input-
flink --output jfs://emr-jfs/test/large-output-flink"
```

2.6 Use Impala or Presto to query data in JindoFS

This topic describes how to use Impala or Presto to query data in JindoFileSystem (JindoFS).

JindoFS configuration

For example, a namespace named emr-jfs is created with the following configuration:

- `jfs.namespaces=emr-jfs`
- `jfs.namespaces.emr-jfs.uri=oss://oss-bucket/oss-dir`
- `jfs.namespaces.emr-jfs.mode=block`

Use JindoFS

Currently, Impala, and Presto services for E-MapReduce V3.22.0 and later can read Hive metadata from Hive tables stored in JindoFS.

To store table data in JindoFS, you can set the parameter that specifies the storage location in the CREATE TABLE statement to a directory in JindoFS.

For example, to store table data in Hadoop Distributed File System (HDFS), use the following CREATE TABLE statement:

```
Create external table lineitem (L_ORDERKEY INT, L_PARTKEY INT,
L_SUPPKEY INT, L_LINENUMBER INT, L_QUANTITY DOUBLE, L_EXTENDED
PRICE DOUBLE, L_DISCOUNT DOUBLE, L_TAX DOUBLE, L_RETURNFLAG STRING
, L_LINESTATUS STRING, L_SHIPDATE STRING, L_COMMITDATE STRING,
L_RECEIPTDATE STRING, L_SHIPINSTRUCT STRING, L_SHIPMODE STRING,
```

```
L_COMMENT STRING) ROW FORMAT DELIMITED FIELDS TERMINATED BY '|' STORED
AS TEXTFILE LOCATION 'hdfs:///tpch_impala/lineitem';
```

To store table data in JindoFS, use the following CREATE TABLE statement:

```
Create external table lineitem (L_ORDERKEY INT, L_PARTKEY INT,
L_SUPPKEY INT, L_LINENUMBER INT, L_QUANTITY DOUBLE, L_EXTENDED
PRICE DOUBLE, L_DISCOUNT DOUBLE, L_TAX DOUBLE, L_RETURNFLAG STRING
, L_LINESTATUS STRING, L_SHIPDATE STRING, L_COMMITDATE STRING,
L_RECEIPTDATE STRING, L_SHIPINSTRUCT STRING, L_SHIPMODE STRING,
L_COMMENT STRING) ROW FORMAT DELIMITED FIELDS TERMINATED BY '|' STORED
AS TEXTFILE LOCATION 'jfs://emr-jfs/tpch_impala/lineitem';
```

2.7 Use JindoFS as the storage back end of HBase

This topic describes how to use JindoFileSystem (JindoFS) as the storage back end of HBase.

Overview

In the Hadoop ecosystem, HBase is a database with high write performance that is suitable for real-time data query. HBase clusters of E-MapReduce V3.22.0 or later support JindoFS or JindoFS-based OssFileSystem as the storage back end. Compared with Hadoop Distributed File System (HDFS), JindoFS and JindoFS-based OssFileSystem are more flexible.

JindoFS configuration

For example, a namespace named emr-jfs is created with the following configuration:

- `jfs.namespaces=emr-jfs`
- `jfs.namespaces.emr-jfs.uri=oss://oss-bucket/oss-dir`
- `jfs.namespaces.emr-jfs.mode=block`

Specify the storage location for an HBase cluster

JindoFS and JindoFS-based OssFileSystem in E-MapReduce V3.22.0 do not support data synchronization. You can set the `hbase.rootdir` parameter in the `hbase-site` file to a directory in JindoFS or JindoFS-based OssFileSystem, and the `hbase.wal.dir` parameter to a local directory in HDFS. Then, you can store write-ahead logging (WAL) files through HDFS in an HBase cluster. To release an HBase cluster, disable tables first and make sure that the updates in WAL files have been written to HFile.

Configuration file	Parameter	Description	Example
hbase-site	hbase.rootdir	The root directory of the HBase cluster in JindoFS.	jfs://emr-jfs/hbase-root-dir
	hbase.wal.dir	The local directory in which the HBase cluster stores WAL files in HDFS.	hdfs://emr-cluster/hbase

Create an E-MapReduce cluster

You can add custom configuration when creating an E-MapReduce cluster, as shown in the following figure.

1 Software Settings | 2 Hardware Settings | 3 Basic Settings

Software Settings

Cluster Type: **Hadoop** | Kafka | ZooKeeper | Druid

On-premises data queries, real-time queries, and ad-hoc queries in big data scenarios

E-MapReduce Hadoop is an open-source Hadoop ecosystem. It uses YARN to manage cluster resources, and supports massive distributed storage and computing of Hive and Spark data stored in HDFS. It supports multiple Hadoop ecosystem components, including stream computing components (Spark Streaming, Flink, and Storm), interactive query components (Presto and Impala), Oozie, and Pig. It also supports OSS storage, Kerberos authentication, and Kerberos encryption.

EMR Version: **EMR-3.22.1**

Required Services: **HDFS (2.8.5)** | **YARN (2.8.5)** | **Hive (3.1.1)** | **Spark (2.4.3)** | **Knox (1.1.0)** | **Zeppelin (0.8.1)** | **Tez (0.9.1)** | **Ganglia (3.7.2)** | **Pig (0.14.0)** | **Sqoop (1.4.7)** | **Binboot (2.0.1)** | **OpenLDAP (2.4.44)** | **Hue (4.4.0)**

Optional Services: **HBase (1.4.9)** | **ZooKeeper (3.5.5)** | **Presto (0.221)** | **Impala (2.12.2)** | **Flume (1.8.0)** | **Livy (0.6.0)** | **Superset (0.28.1)** | **Ranger (1.2.0)** | **Flink (1.7.2)** | **Storm (1.2.2)** | **Phoenix (4.14.1)** | **Analytics Zoo (0.5.0)** | **SmartData (2.0.0)** | **Kudu (1.10.0)** | **Oozie (5.1.0)**

Click to choose

Advanced Settings

Kerberos Mode: ☐ All components in a cluster that is enabled with the Kerberos mode will use Kerberos authentication to validate requests. For more information, see [Kerberos introduction](#).

Custom Software Settings: ☒ Before creating a cluster, you can use a JSON file to customize parameters for the cluster components. For more information, see [Software Settings](#).

```

{
  "ConfigKey": "hbase-site",
  "ConfigValue": "jfs://emr-jfs/hbase-root-dir"
}

```

Use JindoFS

For example, to use JindoFS as the storage back end of an HBase cluster, use the following custom configuration and replace the Object Storage Service (OSS) bucket and relevant directories:

```
[
{
```



```

        "ServiceName": "BIGBOOT",
        "FileName": "bigboot",
        "ConfigKey": "jfs.namespaces",
        "ConfigValue": "emr-jfs"
    },
    {
        "ServiceName": "BIGBOOT",
        "FileName": "bigboot",
        "ConfigKey": "jfs.namespaces.emr-jfs.uri",
        "ConfigValue": "oss://oss-bucket/jindoFS"
    },
    {
        "ServiceName": "BIGBOOT",
        "FileName": "bigboot",
        "ConfigKey": "jfs.namespaces.emr-jfs.mode",
        "ConfigValue": "block"
    },
    {
        "ServiceName": "HBASE",
        "FileName": "hbase-site",
        "ConfigKey": "hbase.rootdir",
        "ConfigValue": "jfs://emr-jfs/hbase-root-dir"
    },
    {
        "ServiceName": "HBASE",
        "FileName": "hbase-site",
        "ConfigKey": "hbase.wal.dir",
        "ConfigValue": "hdfs://emr-cluster/hbase"
    }
]

```

2.8 Store the logs of YARN MapReduce and Spark jobs


This topic describes how to store the logs of MapReduce and Spark jobs to JindoFileSystem (JindoFS) or JindoFS-based OssFileSystem.

Overview

E-MapReduce clusters support the pay-as-you-go and subscription billing methods to meet different needs. Pay-as-you-go clusters can be released at any time. Hadoop clusters store logs in Hadoop Distributed File System (HDFS) by default. If a pay-as-you-go cluster is released, users cannot query job logs of the cluster. In this case, users may have difficulty in troubleshooting job problems. This topic describes how to store the logs of MapReduce and Spark jobs to JindoFS or JindoFS-based OssFileSystem so that you can query the previous logs of jobs.

Configure JindoFS, YARN Container logs, and Spark History Server

- **JindoFS configuration**

Configuration file	Parameter	Description	Example
bigboot	jfs.namespaces	The namespace supported by JindoFS. Separate multiple namespaces with commas (,).	emr-jfs
	jfs.namespaces.emr-jfs.uri	The storage backend of the emr-jfs namespace.	oss://oss-bucket/oss-dir
	jfs.namespaces.test.mode	The storage mode of the emr-jfs namespace.	block  Note: JindoFS supports the block storage mode and cache mode.

- **Configuration for YARN Container logs**

Configuration file	Parameter	Description	Example
yarn-site	yarn.nodemanager.remote-app-log-dir	The directory in which YARN aggregates and stores logs after your application stops running. The log aggregation feature of YARN is enabled by default.	jfs://emr-jfs/emr-cluster-log/yarn-apps-logs or oss://{oss-bucket}/emr-cluster-log/yarn-apps-logs
mapred-site	mapreduce.jobhistory.done-dir	The directory in which JobHistory stores the logs of Hadoop jobs that are completed.	jfs://emr-jfs/emr-cluster-log/jobhistory/done or oss://{oss-bucket}/emr-cluster-log/jobhistory/done

Configuration file	Parameter	Description	Example
	mapreduce. jobhistory. intermediate- done-dir	The directory in which JobHistory stores the logs that are not archived for Hadoop jobs.	<i>jfs://emr-jfs/emr-cluster-log/jobhistory/done_intermediate or oss://\${oss-bucket}/emr-cluster-log/jobhistory/done_intermediate</i>

• **Configuration for Spark History Server**

Configuration file	Parameter	Description	Example
spark-defaults	spark_eventlog_dir	The directory in which Spark History Server stores the logs of Spark jobs.	<i>jfs://emr-jfs/emr-cluster-log/spark-history or oss://\${oss-bucket}/emr-cluster-log/spark-history</i>

Create an E-MapReduce cluster

You can add custom configuration when creating an E-MapReduce cluster, as shown in the following figure.

Software Settings

Cluster Type: **Hadoop** Kafka ZooKeeper Druid

On-premises data queries, real-time queries, and ad-hoc queries in big data scenarios

E-MapReduce Hadoop is an open-source Hadoop ecosystem. It uses YARN to manage cluster resources, and supports massive distributed storage and computing of Hive and Spark data stored in HDFS. It supports multiple Hadoop ecosystem components, including stream computing components (Spark Streaming, Flink, and Storm), interactive query components (Presto and Impala), Oozie, and Pig. It also supports OSS storage, Kerberos authentication, and Kerberos encryption.

EMR Version: **EMR-3.23.0**

Required Services: **HDFS (2.8.5)** **YARN (2.8.5)** **Hive (3.1.1)** **Spark (2.4.3)** **Knox (1.1.0)** **Zeppelin (0.8.1)** **Tez (0.9.1)** **Ganglia (3.7.2)** **Pig (0.14.0)**
Sqoop (1.4.7) **Bigboot (2.0.2)** **OpenLDAP (2.4.44)** **Hue (4.4.0)**

Optional Services: **HBase (1.4.9)** **ZooKeeper (3.5.5)** **Presto (0.221)** **Impala (2.12.2)** **Flume (1.8.0)** **Livr (0.6.0)** **Superset (0.28.1)** **Ranger (1.2.0)**
Flink (1.8.2) **Storm (1.2.2)** **Phoenix (4.14.1)** **Analytics Zoo (0.5.0)** **SmartData (2.0.0)** **Kudu (1.10.0)** **Tensorflow on Spark (1.0.0)**
Oozie (5.1.0)

Click to choose

Advanced Settings

Kerberos Mode: ☐ All components in a cluster that is enabled with the Kerberos mode will use Kerberos authentication to validate requests. For more information, see [Kerberos introduction](#)

Custom Software Settings: ☒ Before creating a cluster, you can use a JSON file to customize parameters for the cluster components. For more information, see [Software Settings](#)

```
{
  "ServiceName": "BIGBOOT", "FileName": "bigboot", "ConfigKey": "jfs.namespaces", "ConfigValue": "emr-jfs",
  "ServiceName": "BIGBOOT", "FileName": "bigboot", "ConfigKey": "jfs.namespaces.emr-jfs.uri", "ConfigValue": "oss://oss-bucket/jindoFS",
  "ServiceName": "BIGBOOT", "FileName": "bigboot", "ConfigKey": "jfs.namespaces.emr-jfs.mode", "ConfigValue": "block",
  "ServiceName": "YARN", "FileName": "mapred-site", "ConfigKey": "mapreduce.jobhistory.done-dir", "ConfigValue": "jfs://emr-jfs/emr-cluster-log/jobhistory/done",
  "ServiceName": "YARN", "FileName": "mapred-site", "ConfigKey": "mapreduce.jobhistory.intermediate-done-dir", "ConfigValue": "jfs://emr-jfs/emr-cluster-log/jobhistory/done_intermediate",
  "ServiceName": "YARN", "FileName": "yarn-site", "ConfigKey": "yarn.nodemanager.remote-app-log-dir", "ConfigValue": "jfs://emr-jfs/emr-cluster-log/yarn-apps-logs",
  "ServiceName": "SPARK", "FileName": "spark-defaults", "ConfigKey": "spark_eventlog_dir", "ConfigValue": "jfs://emr-jfs/emr-cluster-log/spark-history"}
}
```

Custom configuration example

For example, to store logs in JindoFS, use the following custom configuration and replace the Object Storage Service (OSS) bucket and relevant directories:

```
[
  {
    "ServiceName": "BIGBOOT",
    "FileName": "bigboot",
    "ConfigKey": "jfs.namespaces",
    "ConfigValue": "emr-jfs"
  },
  {
    "ServiceName": "BIGBOOT",
    "FileName": "bigboot",
    "ConfigKey": "jfs.namespaces.emr-jfs.uri",
    "ConfigValue": "oss://oss-bucket/jindoFS"
  },
  {
    "ServiceName": "BIGBOOT",
    "FileName": "bigboot",
    "ConfigKey": "jfs.namespaces.emr-jfs.mode",
    "ConfigValue": "block"
  },
  {
    "ServiceName": "YARN",
    "FileName": "mapred-site",
    "ConfigKey": "mapreduce.jobhistory.done-dir",
    "ConfigValue": "jfs://emr-jfs/emr-cluster-log/jobhistory/done"
  }
]
```

```

    },
    {
      "ServiceName": "YARN",
      "FileName": "mapred-site",
      "ConfigKey": "mapreduce.jobhistory.intermediate-done-dir",
      "ConfigValue": "jfs://emr-jfs/emr-cluster-log/jobhistory/
done_intermediate"
    },
    {
      "ServiceName": "YARN",
      "FileName": "yarn-site",
      "ConfigKey": "yarn.nodemanager.remote-app-log-dir",
      "ConfigValue": "jfs://emr-jfs/emr-cluster-log/yarn-apps-logs"
    },
    {
      "ServiceName": "SPARK",
      "FileName": "spark-defaults",
      "ConfigKey": "spark_eventlog_dir",
      "ConfigValue": "jfs://emr-jfs/emr-cluster-log/spark-history"
    }
  ]

```

For example, to store logs in JindoFS-based OssFileSystem, use the following custom configuration and replace the OSS bucket and relevant directories:

```

[
  {
    "ServiceName": "YARN",
    "FileName": "mapred-site",
    "ConfigKey": "mapreduce.jobhistory.done-dir",
    "ConfigValue": "oss://oss_bucket/emr-cluster-log/jobhistory/done"
  },
  {
    "ServiceName": "YARN",
    "FileName": "mapred-site",
    "ConfigKey": "mapreduce.jobhistory.intermediate-done-dir",
    "ConfigValue": "oss://oss_bucket/emr-cluster-log/jobhistory/
done_intermediate"
  },
  {
    "ServiceName": "YARN",
    "FileName": "yarn-site",
    "ConfigKey": "yarn.nodemanager.remote-app-log-dir",
    "ConfigValue": "oss://oss_bucket/emr-cluster-log/yarn-apps-logs"
  },
  {
    "ServiceName": "SPARK",
    "FileName": "spark-defaults",
    "ConfigKey": "spark_eventlog_dir",
    "ConfigValue": "oss://oss_bucket/emr-cluster-log/spark-history"
  }
]

```

]

2.9 Import data from Kafka to JindoFS

When you collect logs and aggregate monitoring data, you can use Apache Kafka to process offline data and streaming data and analyze data in real time. This topic describes how to import data from Kafka to JindoFileSystem (JindoFS).

Import methods

- **Use Flume**

Apache Flume is a system used for moving data into Hadoop Distributed File System (HDFS). We recommend that you use Flume to import data from Kafka to JindoFS. To implement this feature, set the `jfs.type` parameter to `hdfs` and the `jfs.hdfs.path` parameter to a directory in JindoFS:

```
a1.sinks = emr-jfs
...
a1.sinks.emr-jfs.type = hdfs
a1.sinks.emr-jfs.hdfs.path = jfs://emr-jfs/kafka/${topic}/%y-%m-%d
a1.sinks.emr-jfs.hdfs.rollInterval = 10
a1.sinks.emr-jfs.hdfs.rollSize = 0
a1.sinks.emr-jfs.hdfs.rollCount = 0
a1.sinks.emr-jfs.hdfs.fileType = DataStream
```

- **Call a Kafka API**

Some engines like MapReduce and Spark call a Kafka API to export data from Kafka to HDFS. You only need to reference HDFS and set the export path to a directory in JindoFS.

- **Use Kafka HDFS Connector**

You can also use Kafka HDFS Connector to export data from Kafka to HDFS by setting the sink export path to a directory in JindoFS.