Alibaba Cloud E-MapReduce

产品内核增强

文档版本：20200514

法律声明

阿里云提醒您在阅读或使用本文档之前仔细阅读、充分理解本法律声明各条款的内容。如果您阅读或使用本文档，您的阅读或使用行为将被视为对本声明全部内容的认可。

1. 您应当通过阿里云网站或阿里云提供的其他授权通道下载、获取本文档，且仅能用于自身的合法合规的业务活动。本文档的内容视为阿里云的保密信息，您应当严格遵守保密义务；未经阿里云事先书面同意，您不得向任何第三方披露本手册内容或提供给任何第三方使用。
2. 未经阿里云事先书面许可，任何单位、公司或个人不得擅自摘抄、翻译、复制本文档内容的部分或全部，不得以任何方式或途径进行传播和宣传。
3. 由于产品版本升级、调整或其他原因，本文档内容有可能变更。阿里云保留在没有任何通知或者提示下对本文档的内容进行修改的权利，并在阿里云授权通道中不时发布更新后的用户文档。您应当实时关注用户文档的版本变更并通过阿里云授权渠道下载、获取最新版的用户文档。
4. 本文档仅作为用户使用阿里云产品及服务的参考性指引，阿里云以产品及服务的“现状”、“有缺陷”和“当前功能”的状态提供本文档。阿里云在现有技术的基础上尽最大努力提供相应的介绍及操作指引，但阿里云在此明确声明对本文档内容的准确性、完整性、适用性、可靠性等不作任何明示或暗示的保证。任何单位、公司或个人因为下载、使用或信赖本文档而发生任何差错或经济损失的，阿里云不承担任何法律责任。在任何情况下，阿里云均不对任何间接性、后果性、惩戒性、偶然性、特殊性或刑罚性的损害，包括用户使用或信赖本文档而遭受的利润损失，承担责任（即使阿里云已被告知该等损失的可能性）。
5. 阿里云文档中所有内容，包括但不限于图片、架构设计、页面布局、文字描述，均由阿里云和/或其关联公司依法拥有其知识产权，包括但不限于商标权、专利权、著作权、商业秘密等。非经阿里云和/或其关联公司书面同意，任何人不得擅自使用、修改、复制、公开传播、改变、散布、发行或公开发表阿里云网站、产品程序或内容。此外，未经阿里云事先书面同意，任何人不得为了任何营销、广告、促销或其他目的使用、公布或复制阿里云的名称（包括但不限于单独为或以组合形式包含“阿里云”、“Aliyun”、“万网”等阿里云和/或其关联公司品牌，上述品牌的附属标志及图案或任何类似公司名称、商号、商标、产品或服务名称、域名、图案标示、标志、标识或通过特定描述使第三方能够识别阿里云和/或其关联公司）。
6. 如若发现本文档存在任何错误，请与阿里云取得直接联系。

通用约定

格式	说明	样例
	该类警示信息将导致系统重大变更甚至故障，或者导致人身伤害等结果。	 **禁止：** 重置操作将丢失用户配置数据。
	该类警示信息可能会导致系统重大变更甚至故障，或者导致人身伤害等结果。	 **警告：** 重启操作将导致业务中断，恢复业务时间约十分钟。
	用于警示信息、补充说明等，是用户必须了解的内容。	 **注意：** 权重设置为0，该服务器不会再接受新请求。
	用于补充说明、最佳实践、窍门等，不是用户必须了解的内容。	 **说明：** 您也可以通过按Ctrl + A选中全部文件。
>	多级菜单递进。	单击**设置** > **网络** > **设置网络类型**。
粗体	表示按键、菜单、页面名称等UI元素。	在**结果确认**页面，单击**确定**。
Courier字体	命令。	执行cd /d C:/window命令，进入Windows系统文件夹。
斜体	表示参数、变量。	bae log list --instanceid Instance_ID
[]或者[a|b]	表示可选项，至多选择一个。	ipconfig [-all|-t]
{}或者{a|b}	表示必选项，至多选择一个。	switch {active|stand}

目录

1 JindoFS 基础使用

1.1 JindoFS 使用说明（E-MapReduce-3.20.0~3.22.0版本）

本文主要介绍JindoFS的配置使用方式，以及介绍一些典型的应用场景。

概述

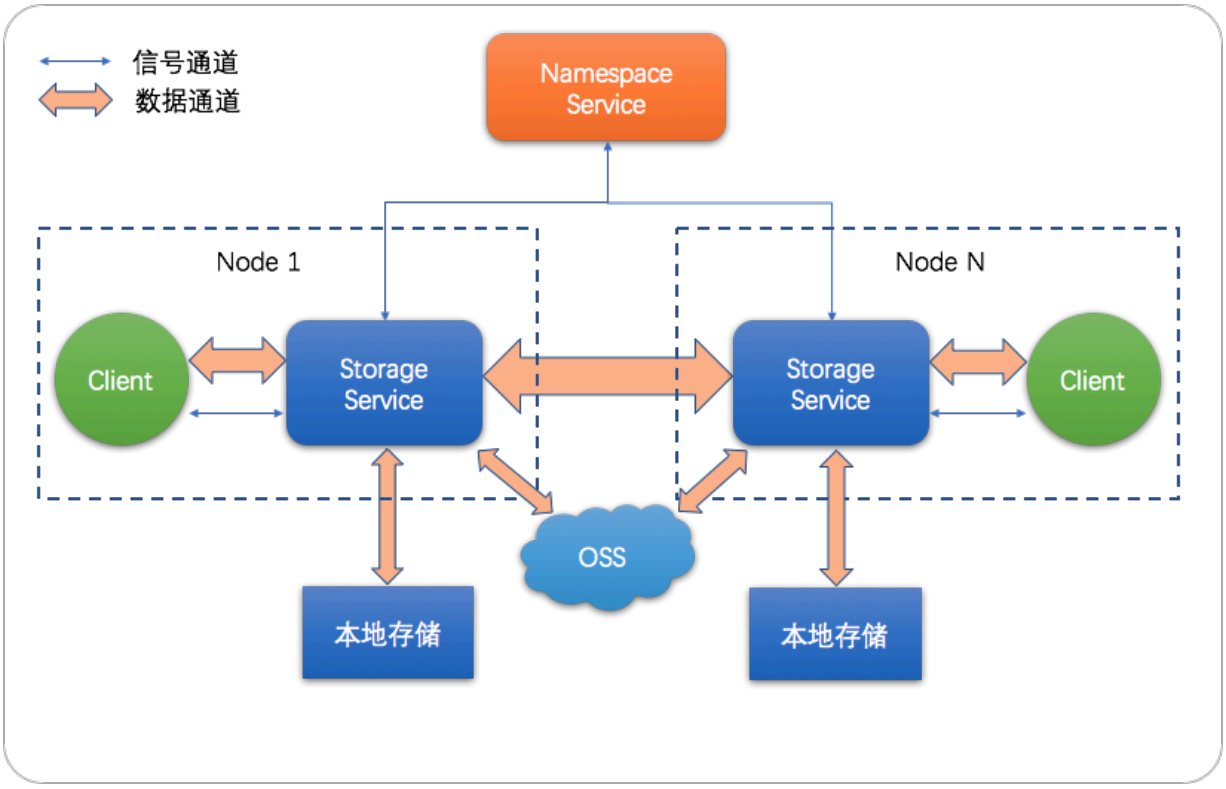
JindoFS是一种云原生的文件系统，结合OSS和本地存储，成为E-MapReduce产品的新一代存储系统，为上层计算提供了高效可靠的存储。

JindoFS 提供了块存储模式（Block）和缓存模式（Cache）的存储模式。

JindoFS 采用了本地存储和OSS的异构多备份机制，Storage Service提供了数据存储能力，首先使用OSS作为存储后端，保证数据的高可靠性，同时利用本地存储实现冗余备份，利用本地的备份，可以加速数据读取；另外，JindoFS 的元数据通过本地服务Namespace Service管理，从而保证了元数据操作的性能（和HDFS元数据操作性能相似）。

说明：

- E-MapReduce-3.20.0及以上版本支持Jindo FS，您可以在创建集群时勾选相关服务来使用JindoFS。
- 本文主要是E-MapReduce-3.20.0及以上版本至E-MapReduce-3.22.0（但不包括）版本的介绍；E-MapReduce-3.22.0 及以上版本的JindoFS 使用说明，请参见JindoFS 使用说明（E-MapReduce-3.22.0及以上版本）。



应用场景

E-MapReduce目前提供了三种大数据存储系统，E-MapReduce OssFileSystem、E-MapReduce HDFS和E-MapReduce JindoFS，其中OssFileSystem和JindoFS都是云上存储的解决方案，下表为这三种存储系统和开源OSS各自的特点。

特点	开源OSS	E-MapReduce OssFileSystem	E-MapReduce HDFS	E-MapReduce JindoFS
存储空间	海量	海量	取决于集群规模	海量
可靠性	高	高	高	高
吞吐率因素	服务端	集群内磁盘缓存	集群内磁盘	集群内磁盘
元数据效率	慢	中	快	快
扩容操作	容易	容易	容易	容易
缩容操作	容易	容易	需Decommission	容易
数据本地化	无	弱	强	较强

JindoFS块存储模式具有以下几个特点：

- 海量弹性的存储空间，基于OSS作为存储后端，存储不受限于本地集群，而且本地集群能够自由弹性伸缩。

- 能够利用本地集群的存储资源加速数据读取，适合具有一定本地存储能力的集群，能够利用有限的本地存储提升吞吐率，特别对于一写多读的场景效果显著。
- 元数据操作效率高，能够与HDFS相当，能够有效规避OSS文件系统元数据操作耗时以及高频访问下可能引发不稳定的问题。
- 能够最大限度保证执行作业时的数据本地化，减少网络传输的压力，进一步提升读取性能。

环境准备

- 创建集群

  选择 E-MapReduce-3.20.0及以上版本至E-MapReduce-3.22.0（但不包括）版本，勾选可选服务中的**SmartData**和**Bigboot**，具体操作步骤请参见#unique_6。Bigboot 服务提供了E-MapReduce平台上的基础的分布式数据管理交互服务以及一些组件管理监控和支持性服务，SmartData服务基于Bigboot之上对应用层提供了JindoFS文件系统。

- 配置集群

  SmartData提供的JindoFS文件系统使用OSS作为存储后端，因此在使用JindoFS之前需配置一些OSS相关参数。下面提供两种配置方式，第一种是先创建好集群，修改Bigboot相关参数，需重

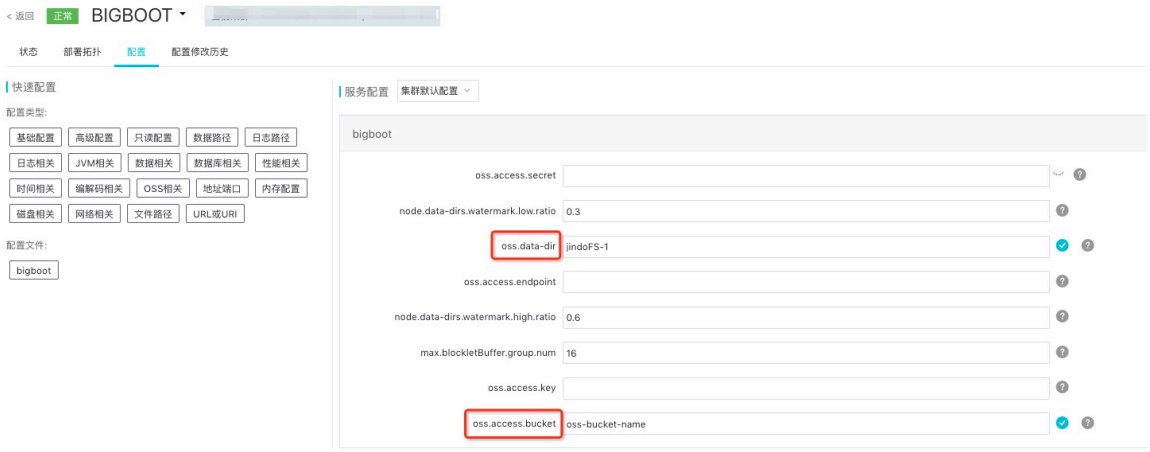
启SmartData服务生效；第二种是创建集群过程中添加自定义配置，这样集群创建好后相关服务就能按照自定义参数启动。

- 集群创建好后参数初始化

  - oss.access.bucket为OSS bucket的名称。
  - oss.data-dir为JindoFS在OSS bucket中所使用的目录（注：该目录为 JindoFS 后端存储目录，生成的数据不能人为破坏，并且保证该目录仅用于JindoFS后端存储，JindoFS在写入数据时会自动创建用户所配置的目录，无需在OSS上事先创建）。
  - oss.access.endpoint为bucket所在的区域。
  - oss.access.key为存储后端OSS的AK。
  - oss.access.secret为存储后端OSS的secret。

  考虑到性能和稳定性，推荐使用同region下的OSS bucket作为存储后端，此时，E-MapReduce集群能够免密访问OSS，无需配置AK和secret。

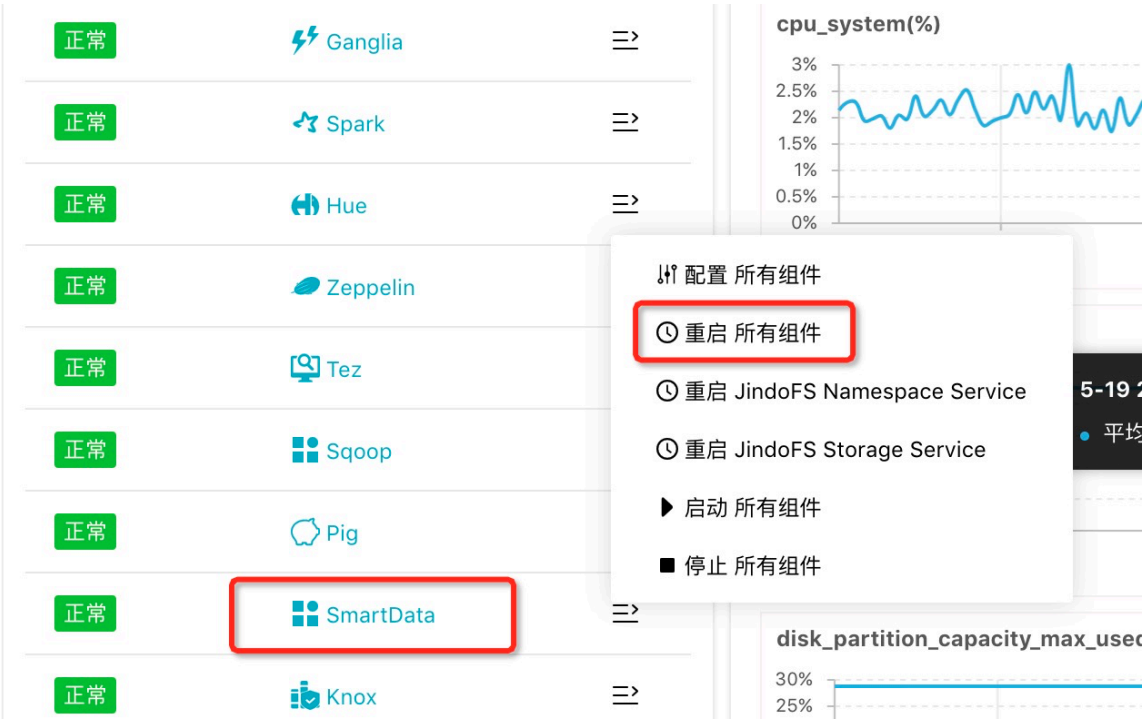
  所有JindoFS相关配置都在Bigboot组件中，配置如下图所示，红框中为必填的配置项。



说明：

JindoFS支持多命名空间，本文命名空间以test为例。

配置完成后保存并部署，然后在SmartData服务中重启所有组件，即开始使用JindoFS。



- 创建集群时添加自定义配置

E-MapReduce集群在创建集群时支持添加自定义配置，以同region下免密访问OSS为例，如下图勾选**软件自定义配置**，添加如下配置，配置oss.data-dir和oss.access.bucket。

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

```
]
```

使用 JindoFS

JindoFS使用上与HDFS类似，提供jfs前缀，将jfs替代hdfs即可使用。简单示例：

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

目前，JindoFS 能够支持 E-MapReduce 集群上的 Hadoop、Hive、Spark 的作业进行访问，其余组件尚未完全支持。

磁盘空间水位控制

JindoFS后端基于OSS，可以提供海量的存储，但是本地盘的容量是有限的，因此JindoFS会自动淘汰本地较冷的数据备份。我们提供了**node.data-dirs.watermark.high.ratio** 和 **node.data-dirs**

.watermark.low.ratio 这两个参数用来调节本地存储的使用容量，值均为0～1 的小数表示使用比例，JindoFS 默认使用所有数据盘，每块盘的使用容量默认即为数据盘大小。前者表示使用量上水位比例，每块数据盘的JindoFS占用的空间到达上水位即会开始清理淘汰；后者表示使用量下水位比例，触发清理后会将JindoFS 的占用空间清理到下水位。用户可以通过设置上水位比例调节期望分给JindoFS的磁盘空间，下水位必须小于上水位，设置合理的值即可。

存储策略

JindoFS提供了Storage Policy功能，提供更加灵活的存储策略适应不同的存储需求，可以对目录设置以下四种存储策略。

策略	策略说明
COLD	表示数据仅在OSS上有一个备份，没有本地备份，适用于冷数据存储。
WARM	默认策略。 表示数据在OSS和本地分别有一个备份， 本地备份能够有效的提供后续的读取加速。
HOT	表示数据在OSS上有一个备份，本地有多个备份，针对一些最热的数据提供更进一步的加速效果。
TEMP	表示数据仅有一个本地备份，针对一些临时性数据，提供高性能的读写，但降低了数据的高可靠性，适用于一些临时数据的存取。

JindoFS提供了Admin工具设置目录的Storage Policy（默认为 WARM），新增的文件将会以父目录所指定的Storage Policy进行存储，使用方式如下所示。

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

通过以下命令，获取某个目录的存储策略。

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

说明：

其中[path] 为设置policy的路径名称， -R表示递归设置该路径下的所有路径。

Admin工具还提供archive命令，实现对冷数据的归档。

此命令提供了一种用户显式淘汰本地数据块的方式。Hive分区表按天分区，假如业务上对一周前的分区数据认为不会再经常访问，那么就可以定期将一周前的分区目录执行archive，淘汰本地备份，文件备份将仅仅保留在后端OSS上。

Archive命令的使用方式如下：

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

说明：

[path]为需要归档文件的所在目录路径。

1.2 JindoFS 使用说明（E-MapReduce-3.22.0及以上版本）

JindoFS是一种云原生的文件系统，结合OSS和本地存储，成为E-MapReduce产品的新一代存储系统，为上层计算提供了高效可靠的存储。本文主要说明JindoFS的配置使用方式，以及介绍一些典型的应用场景。

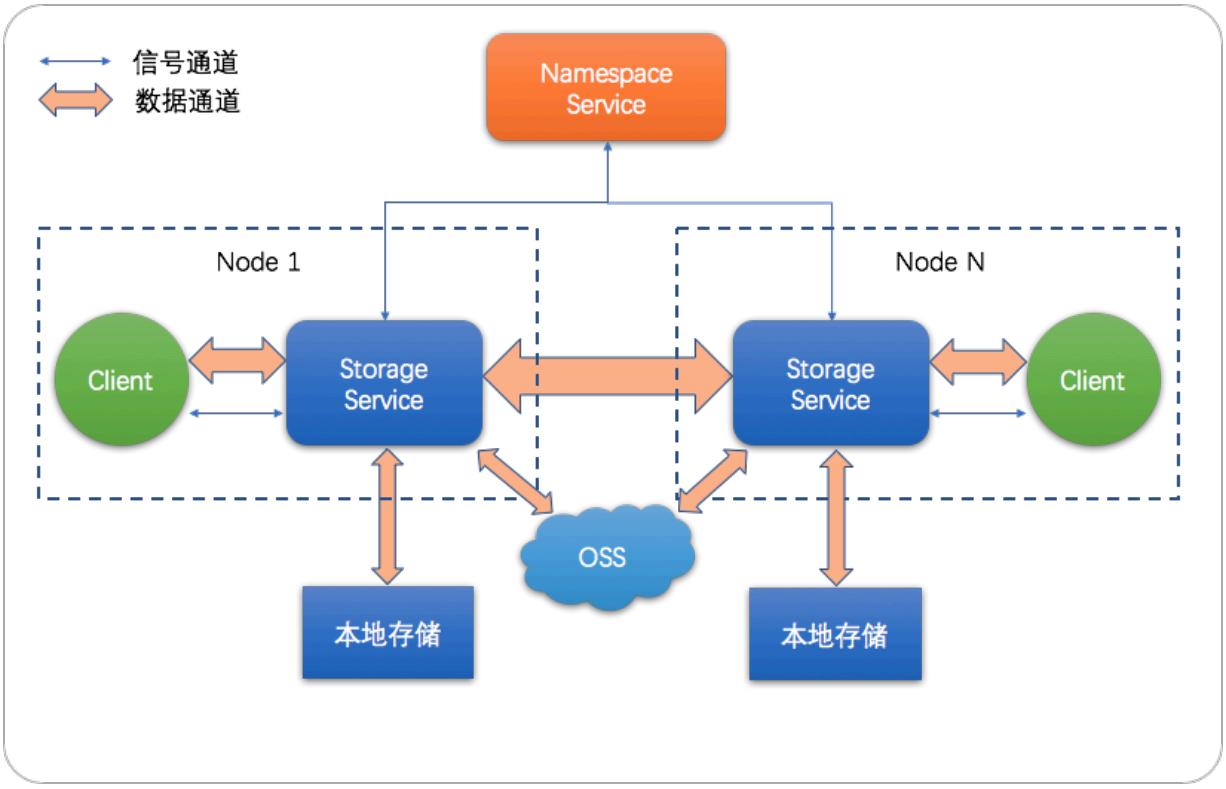
概述

JindoFS 提供了块存储模式（Block）和缓存模式（Cache）的存储模式。

JindoFS采用了本地存储和OSS的异构多备份机制，Storage Service提供了数据存储能力，首先使用OSS作为存储后端，保证数据的高可靠性，同时利用本地存储实现冗余备份，利用本地的备份，可以加速数据读取；另外，JindoFS 的元数据通过本地服务Namespace Service管理，从而保证了元数据操作的性能（和HDFS元数据操作性能相似）。

说明：

- E-MapReduce-3.20.0及以上版本支持Jindo FS，您可以在创建集群时勾选相关服务来使用JindoFS。
- 本文主要是E-MapReduce-3.22.0 及以上版本的介绍；E-MapReduce-3.20.0及以上版本至E-MapReduce-3.22.0（但不包括）版本的JindoFS 使用说明，请参见JindoFS 使用说明（E-MapReduce-3.20.0~3.22.0版本）。



环境准备

- 创建集群

  选择 E-MapReduce-3.22.0 及以上版本，勾选可选服务中的**SmartData**，具体操作步骤请参见#unique_6。

- 配置集群

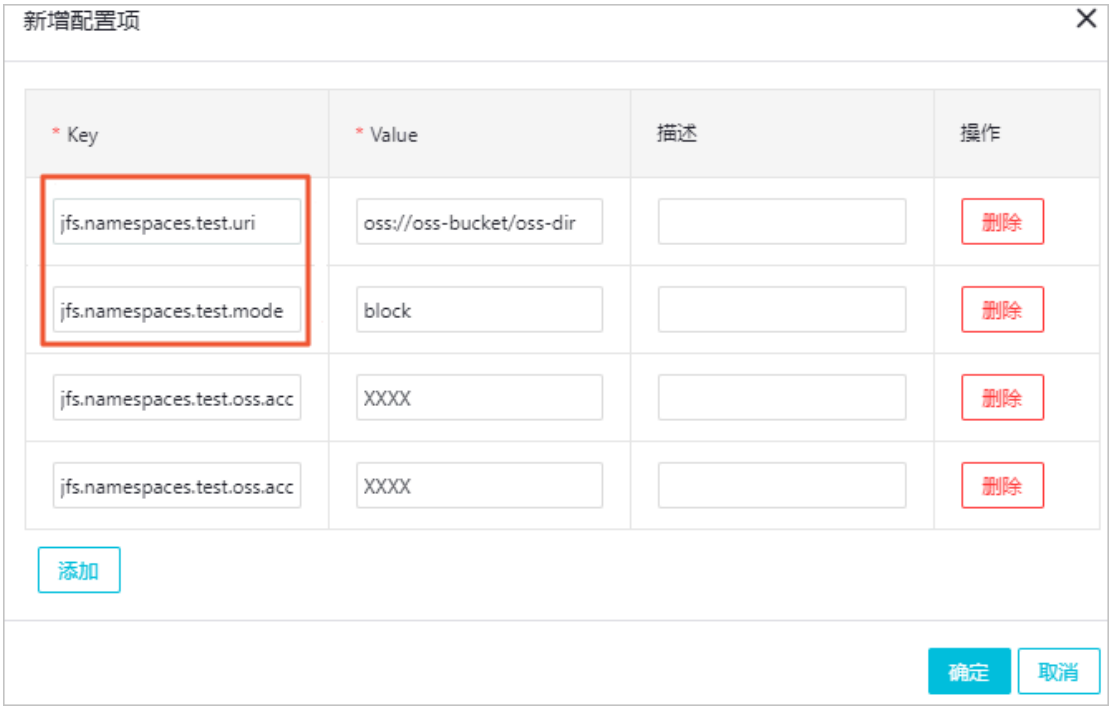
  SmartData提供的JindoFS文件系统使用OSS作为存储后端，因此在使用JindoFS之前需配置一些OSS相关参数。下面提供两种配置方式，第一种是先创建好集群，修改Bigboot相关参数，需重启SmartData服务生效；第二种是创建集群过程中添加自定义配置，这样集群创建好后相关服务就能按照自定义参数启动：

  - 集群创建好后初始化参数

    所有JindoFS相关配置都在Bigboot组件中，配置如下图所示：

    1. 在**服务配置**页面，单击**bigboot**页签。

    2. 单击**自定义配置**。



说明：

■ 红框中为必填的配置项。

■ JindoFS支持多命名空间，本文命名空间以test为例。

参数	参数说明	示例
jfs.namespaces	表示当前JindoFS支持的命名空间，多个命名空间时以逗号隔开。	test
jfs.namespaces.test.uri	表示test命名空间的后端存储。	oss://oss-bucket/oss-dir **说明：** 该配置也可以配置到OSS bucket下的具体目录，该命名空间即以该目录作为根目录来读写数据。
jfs.namespaces.test.mode	表示test命名空间为块存储模式。	block **说明：** JindoFS支持block和cache两种存储模式。

参数	参数说明	示例
jfs.namespaces.test.oss.access.key	表示存储后端OSS的AK。	xxxx **说明：** 考虑到性能和稳定性，推荐使用同账户、同region下的OSS bucket作为存储后端，此时，E-MapReduce集群能够免密访问OSS，无需配置AK和secret。
jfs.namespaces.test.oss.access.secret	表示存储后端OSS的secret。	

配置完成后保存并部署，然后在SmartData服务中重启所有组件，即开始使用JindoFS。

- 创建集群时添加自定义配置

E-MapReduce集群在创建集群时支持添加自定义配置，以同region下免密访问OSS为例，如下图勾选**软件自定义配置**，配置命名空间test的相关配置，详情如下：

```
[
{
"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"
}
]
```

使用 JindoFS

JindoFS使用上与HDFS类似，提供jfs前缀，将jfs替代hdfs即可使用。

目前，JindoFS能够支持 EMR 集群上的大部分计算组件，包括 Hadoop、Hive、Spark、Flink、Presto、Impala。

简单示例：

- Shell命令

```
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 ./
```

- MapReduce作业

```
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
```

- Spark-SQL

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

磁盘空间水位控制

JindoFS后端基于OSS，可以提供海量的存储，但是本地盘的容量是有限的，因此JindoFS会自动淘汰本地较冷的数据备份。我们提供了**node.data-dirs.watermark.high.ratio**和**node.data-dirs.watermark.low.ratio**这两个参数用来调节本地存储的使用容量，值均为0～1的小数表示使用比例，JindoFS默认使用所有数据盘，每块盘的使用容量默认即为数据盘大小。前者表示使用量上水位比例，每块数据盘的JindoFS占用的空间到达上水位即会开始清理淘汰；后者表示使用量下水位比例，触发清理后会将JindoFS的占用空间清理到下水位。用户可以通过设置上水位比例调节期望分给JindoFS的磁盘空间，下水位必须小于上水位，设置合理的值即可。

存储策略

JindoFS提供了Storage Policy功能，提供更加灵活的存储策略适应不同的存储需求，可以对目录设置以下四种存储策略：

策略	策略说明
COLD	表示数据仅在OSS上有一个备份，没有本地备份，适用于冷数据存储。
WARM	默认策略。 表示数据在OSS和本地分别有一个备份， 本地备份能够有效的提供后续的读取加速。
HOT	表示数据在OSS 上有一个备份，本地有多个备份，针对一些最热的数据提供更进一步的加速效果。
TEMP	表示数据仅有一个本地备份，针对一些临时性数据，提供高性能的读写，但降低了数据的高可靠性，适用于一些临时数据的存取。

JindoFS 提供了Admin工具设置目录的Storage Policy（默认为 WARM），新增的文件将会以父目录所指定的 Storage Policy 进行存储，使用方式如下：

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

通过以下命令，获取某个目录的存储策略：

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

说明：

其中[path] 为设置policy的路径名称， -R表示递归设置该路径下的所有路径。

Admin 工具

JindoFS提供了Admin工具的archive命令、jindo命令。

- Admin工具提供archive命令，实现对冷数据的归档。

  此命令提供了一种用户显式淘汰本地数据块的方式。Hive分区表按天分区，假如业务上对一周前的分区数据认为不会再经常访问，那么就可以定期将一周前的分区目录执行archive，淘汰本地备份，文件备份将仅仅保留在后端OSS 上。

  Archive命令的使用方式如下：

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

  说明：

  [path]为需要归档文件的所在目录路径。

- Admin工具提供jindo命令，为Namespace Service提供了一些管理员功能命令。

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

  说明：

  可以通过jindo dfsadmin --help命令获取帮助信息。

Admin工具对Cache模式提供了diff和sync 命令。

- diff命令主要用来显示本地数据与后端存储系统数据之间的差异。

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

  说明：

  默认情况下比较[path]目录的子目录中元数据之间的差异，-R选项表示递归比较[path]目录下所有的路径。

- sync命令用于同步本地与后端存储之前的元数据。

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

> **说明:**
> [path]表示需要同步元数据的路径，默认只会同步[path]的下一级目录，-R选项表示递归比较[path]目录下所有的路径。

1.3 JindoFS 块存储模式

本文主要介绍JindoFS的块存储模式（Block），以及一些典型的应用场景。

概念

块存储模式提供了最为高效的数据读写能力和元数据访问能力，并且能够支持更加全面的Hadoop文件系统语义。同时，JindoFS也提供了外部客户端，能够从集群外部访问建立在E-MapReduce集群内的JindoFS文件系统。

数据以Block形式存储在后端存储OSS上，本地Namespace服务维护元数据信息，该模式在性能上较优，无论是数据性能还是元数据性能。

应用场景

E-MapReduce目前提供了三种大数据存储系统，E-MapReduce OssFileSystem、E-MapReduce HDFS 和 E-MapReduce JindoFS，其中OssFileSystem和JindoFS都是云上存储的解决方案，下表为这三种存储系统和开源OSS各自的特点。

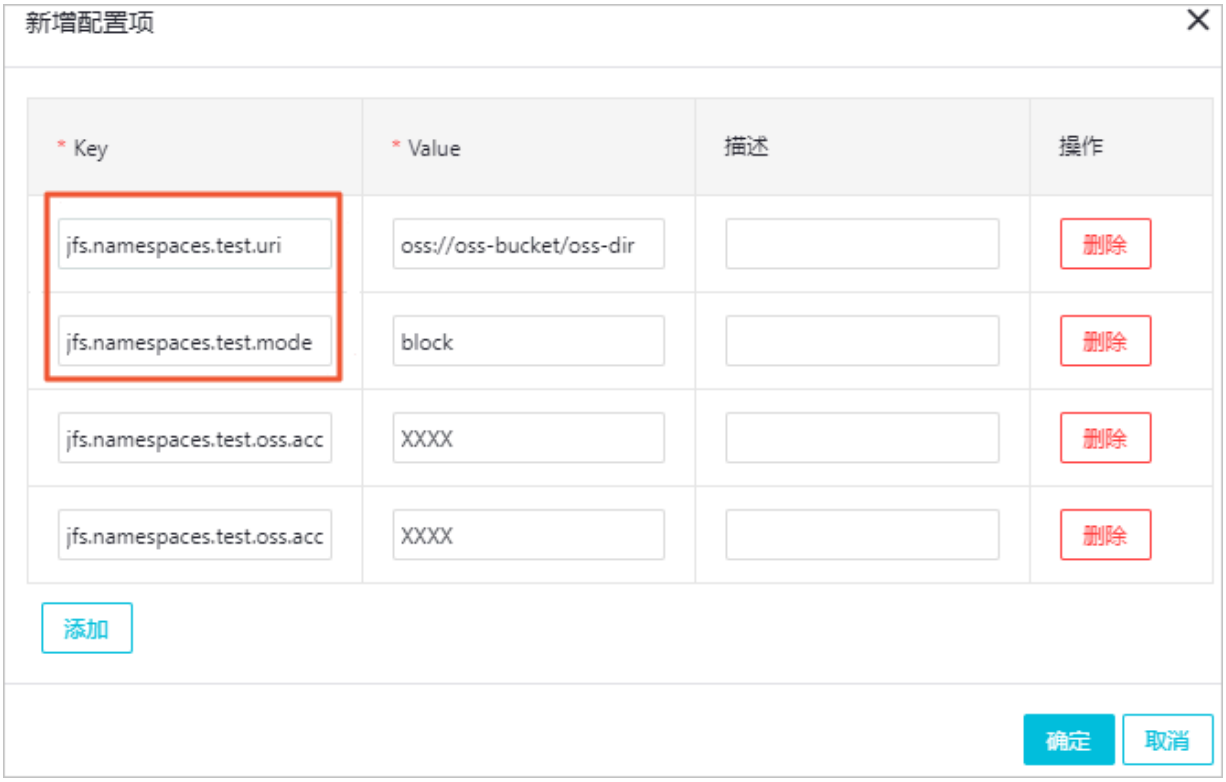
特点	开源 OSS	E-MapReduce OssFileSystem	E-MapReduce HDFS	E-MapReduce JindoFS
存储空间	海量	海量	取决于集群规模	海量
可靠性	高	高	高	高
吞吐率因素	服务端	集群内磁盘缓存	集群内磁盘	集群内磁盘
元数据效率	慢	中	快	快
扩容操作	容易	容易	容易	容易
缩容操作	容易	容易	需Decommission	容易
数据本地化	无	弱	强	较强

JindoFS块存储模式具有以下几个特点：

- 海量弹性的存储空间，基于OSS作为存储后端，存储不受限于本地集群，而且本地集群能够自由弹性伸缩。
- 能够利用本地集群的存储资源加速数据读取，适合具有一定本地存储能力的集群，能够利用有限的本地存储提升吞吐率，特别对于一写多读的场景效果显著。
- 元数据操作效率高，能够与HDFS相当，能够有效规避OSS文件系统元数据操作耗时以及高频访问下可能引发不稳定的问题。
- 能够最大限度保证执行作业时的数据本地化，减少网络传输的压力，进一步提升读取性能。

配置集群

所有JindoFS相关配置都在Bigboot组件中，配置如下图所示。



说明：

- 红框中为必填的配置项。

- JindoFS支持多命名空间，本文命名空间以test为例。

参数	参数说明	示例
jfs.namespaces	表示当前JindoFS支持的命名空间，多个命名空间时以逗号隔开。	test
jfs.namespaces.test.uri	表示test命名空间的后端存储。	oss://oss-bucket/oss-dir **说明：** 该配置也可以配置到OSS bucket下的具体目录，该命名空间即以该目录作为根目录来读写数据。
jfs.namespaces.test.mode	表示test命名空间为块存储模式。	block
jfs.namespaces.test.oss.access.key	表示存储后端OSS的AK。	xxxx **说明：** 考虑到性能和稳定性，推荐使用同账户、同region下的OSS bucket作为存储后端，此时，E-MapReduce集群能够免密访问OSS，无需配置AK和secret。
jfs.namespaces.test.oss.access.secret	表示存储后端OSS的secret。	

配置完成后保存并部署，然后在SmartData服务中重启Namespace Service，即可开始使用JindoFS。

存储策略

JindoFS提供了Storage Policy功能，提供更加灵活的存储策略适应不同的存储需求，可以对目录设置以下四种存储策略：

策略	策略说明
COLD	表示数据仅在OSS上有一个备份，没有本地备份，适用于冷数据存储。
WARM	默认策略。 表示数据在OSS和本地分别有一个备份， 本地备份能够有效的提供后续的读取加速。
HOT	表示数据在OSS 上有一个备份，本地有多个备份，针对一些最热的数据提供更进一步的加速效果。
TEMP	表示数据仅有一个本地备份，针对一些临时性数据，提供高性能的读写，但降低了数据的高可靠性，适用于一些临时数据的存取。

JindoFS 提供了Admin工具设置目录的Storage Policy（默认为 WARM），新增的文件将会以父目录所指定的 Storage Policy 进行存储，使用方式如下：

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

通过以下命令，获取某个目录的存储策略：

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

说明：
其中[path] 为设置policy的路径名称， -R表示递归设置该路径下的所有路径。

Admin工具还提供archive命令，实现对冷数据的归档。

此命令提供了一种用户显式淘汰本地数据块的方式。Hive分区表按天分区，假如业务上对一周前的分区数据认为不会再经常访问，那么就可以定期将一周前的分区目录执行archive，淘汰本地备份，文件备份将仅仅保留在后端OSS 上。

Archive命令的使用方式如下：

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

说明：
[path]为需要归档文件的所在目录路径。

1.4 JindoFS 缓存模式

本文主要介绍JindoFS的缓存模式（Cache），以及一些典型的应用场景。

概述

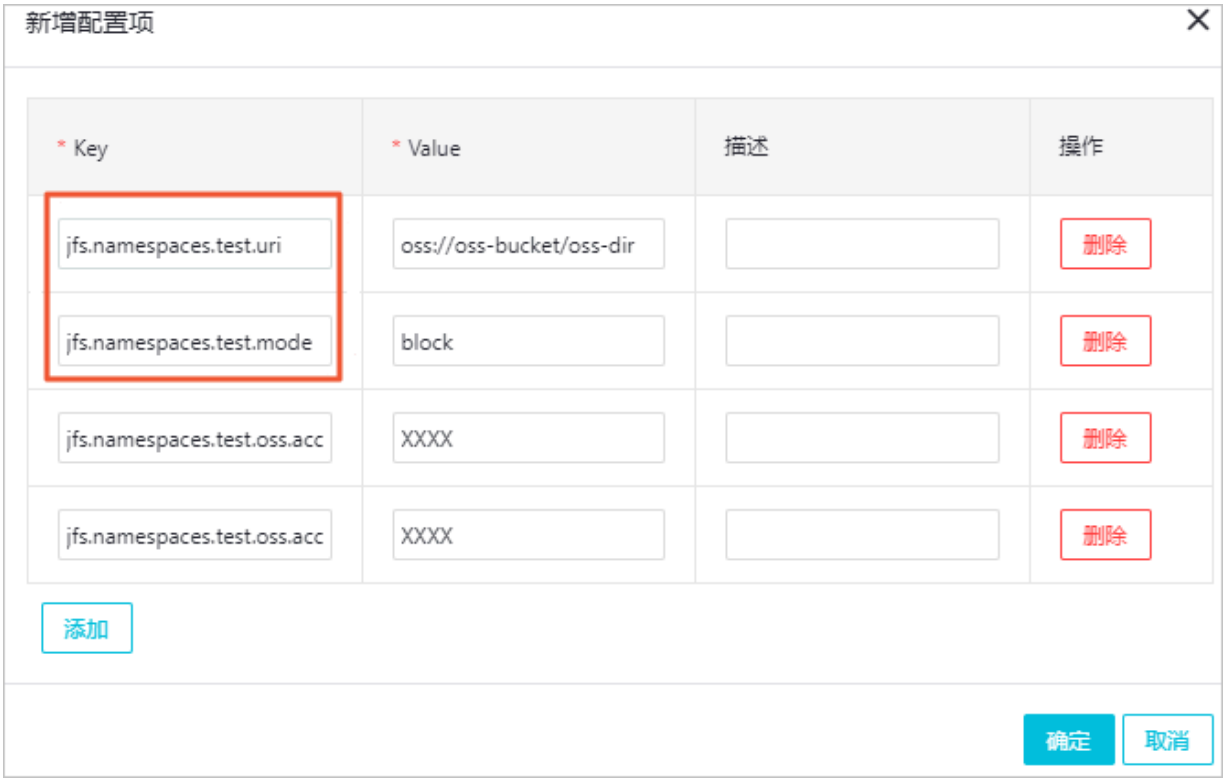
缓存模式兼容现有OSS存储方式，文件以对象的形式存储在OSS上，每个文件根据实际访问情况会在本地进行数据和元数据的缓存，从而提高访问数据以及元数据的性能，Cache模式提供不同元数据同步策略以满足用户在不同场景下的需求。

应用场景

缓存模式最大的特点就是兼容性，保持了OSS原有的对象语义，集群中仅做缓存，因此JindoFS和OSS客户端、OssFileSystem等，或者其他的各种OSS的交互程序是完全兼容的，对原有OSS上的存量数据也不需要做任何的迁移、转换工作即可使用。同时集群中的数据和元数据缓存也能一定程度上提升数据访问性能。

配置集群

所有JindoFS相关配置都在Bigboot组件中，配置如下图所示。



说明：

- 红框中为必填的配置项。
- JindoFS支持多命名空间，本文命名空间以test为例。

参数	参数说明	示例
jfs.namespaces	表示当前 JindoFS 支持的命名空间，多个命名空间时以逗号隔开。	test
jfs.namespaces.test.uri	表示 test 命名空间的后端存储。	oss://oss-bucket/ **说明：** 该配置也可以配置到OSS bucket下的具体目录，该命名空间即以该目录作为根目录来读写数据，但一般情况下配置bucket即可，这样路径就和原生OSS保持一致。
jfs.namespaces.test.mode	表示 test 命名空间为缓存模式。	cache

参数	参数说明	示例
jfs.namespaces.test.oss.access.key	表示存储后端OSS的AK。	xxxx **说明：** 考虑到性能和稳定性，推荐使用同账户、同region下的OSS bucket作为存储后端，此时，E-MapReduce集群能够免密访问OSS，无需配置AK和secret。
jfs.namespaces.test.oss.access.secret	表示存储后端OSS的secret。	

配置完成后保存并部署，然后在SmartData服务中重启Namespace Service，即可开始使用JindoFS。

元数据同步策略

缓存模式下可能存在JindoFS集群构建之前，用户已经在OSS上保存了大量数据的场景，对于这种场景，后续的数据访问会同步数据和元数据到JindoFS集群，数据同步策略为了访问数据都会在本地保留一份；元数据同步策略分为两部分，包括元数据同步间隔策略和元数据load策略。

- 元数据同步间隔：

  配置参数为**namespace.sync.interval**， 该参数默认值为**-1**， 表示不会同步OSS上的元数据。

  - 当**namespace.sync.interval=0**时，表示每次操作都会同步OSS 上的元数据
  - 当**namespace.sync.interval>0**时，表示会以固定的时间间隔来同步OSS上的元数据。

    说明：
    例如当**namespace.sync.interval=5**时， 表示每隔5秒会去同步OSS上的元数据。

- 元数据Load策略：

  配置参数为**namespace.sync.loadtype**，该配置参数为枚举类型**{never, once, always}**，**never**表示从不同步OSS上的元数据；**once**为默认配置，表示只从OSS同步一次元数据；**always**表示每次操作都会同步OSS上的元数据。

  > **说明：**
  > 当不配置**namespace.sync.interval**参数时，才会去使用Load策略；如果已配置**namespace.sync.interval**参数，则Load策略配置不生效。

1.5 JindoFS 外部客户端

本文主要介绍JindoFS的外部客户端，以及一些典型的应用场景。

概述

JindoFS外部客户端，主要是为E-MapReduce集群外部访问JindoFS集群提供一种可行的方法。现在JindoFS外部客户端只能访问块存储模式下的 JindoFS，不支持访问缓存模式下的JindoFS。实际上，缓存模式兼容OSS原始语义，因此外部访问仅需用普通OSS客户端即可。

应用场景

JindoFS外部客户端实现了Hadoop文件系统的接口，在用户程序跟E-MapReduce JindoFS Namespace服务网络相通的情况下， 用户可以通过JindoFS外部客户端去访问JindoFS上存储的数据， 但外部客户端不能利用E-MapReduce JindoFS的数据缓存能力，相比E-MapReduce集群内部访问JindoFS集群，性能有所损失。

配置外部客户端

已配置JindoFS块存储模式的Namespace，具体请参见JindoFS 块存储模式。

1. 获取Bigboot 程序包。

   在E-MapReduce集群内部 /usr/lib/bigboot-current路径下，获取Bigboot 程序包。

   > **说明：**
   > 一般情况下，程序使用Native开发，若实际系统类型与E-MapReduce集群差别较大，相关的程序需重新编译，可以通过工单联系我们处理。

2. 配置环境。

   设置环境变量**BIGBOOT_HOME**为程序安装根目录，将程序根目录下ext和lib的路径，添加到用户使用的大数据组件（Hadoop、Spark 等）的**Classpath**中。

3. 从E-MapReduce集群内部拷贝配置文件/usr/lib/bigboot-current/conf/bigboot.cfg.external，到用户客户机上对应的安装目录conf/bigboot.cfg。
4. 配置Namespace Service。
   - client.namespace.rpc.port：配置JindoFS Namespace Service的监听端口。
   - client.namespace.rpc.address：配置JindoFS Namespace Service的监听地址。

     说明：
     默认E-MapReduce集群中的配置文件已经配置好这两项。
5. 配置数据访问相关的配置项。
   - client.namespaces.{YourNamespace}.oss.access.bucket：配置OSS bucket选项。
   - client.namespaces.{YourNamespace}.oss.access.endpoint：配置OSS endpoint选项。
   - client.namespaces.{YourNamespace}.oss.access.key：配置OSS key选项。
   - client.namespaces.{YourNamespace}.oss.access.secret：配置OSS secret选项。

     说明：
     其中{YourNamespace}为外部客户端要访问的Namespace的名称，本文Namespace的名称以test为例。

     配置示例如下：

```
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}
```

配置验证

- 通过以下命令，验证Namespace是否正确：

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

- 通过以下命令，验证数据是否可以上传或者下载：

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

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

2 JindoFS 生态

2.1 迁移Hadoop文件系统数据至JindoFS

本文以OSS为例，介绍如何将Hadoop文件系统上的数据迁移至JindoFS。

迁移数据

- Hadoop FsShell

  对于文件较少或者数据量较小的场景，可以直接使用Hadoop的FsShell进行同步：

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

- DistCp

  对于文件较多或者数据量较大的场景，推荐使用Hadoop内置的DistCp进行同步：

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

  说明：
  更多DistCp参数可参见DistCp Version2 Guide。

利用JindoFS缓存模式

缓存模式是兼容现有OSS的存储方式：文件会以原生对象的形式存储在OSS上，同时OSS文件通过JindoFS缓存模式访问时，也有机会在本地进行数据和元数据的缓存、加速访问，具体可参见JindoFS 缓存模式。

2.2 使用MapReduce处理JindoFS上的数据

本文介绍如何使用MapReduce读写JindoFS上的数据。

JindoFS配置

已创建名为emr-jfs的命名空间，示例如下：

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

MapReduce简介

Hadoop MapReduce作业一般通过HDFS进行读写，JindoFS目前已兼容大部分HDFS接口，通常只需要将MapReduce作业的输入、输出目录配置到JindoFS，即可实现读写JindoFS上的文件 。

Hadoop Map/Reduce是一个使用简易的软件框架，基于它写出来的应用程序能够运行在由上千个商用机器组成的大型集群上，并以一种可靠容错的方式并行处理上T级别的数据集。一个Map/Reduce作业（job） 通常会把输入的数据集切分为若干独立的数据块，由map任务（task）以完全并行的方式处理它们。框架会对map的输出先进行排序， 然后把结果输入给reduce任务。通常作业的输入和输出都会被存储在文件系统中。 整个框架负责任务的调度和监控，以及重新执行已经失败的任务。

作业的输入和输出

MapReduce作业一般会指明输入/输出的位置（路径），并通过实现合适的接口或抽象类提供map和reduce函数。Hadoop的 job client 再加上其他作业的参数提交给ResourceManager，进行调度执行。这种情况下，我们直接修改作业的输入和输出目录即可实现JindoFS的读写。

MapReduce on JindoFS样例

以下是MapReduce作业通过修改输入输出实现JindoFS的读写的例子。

- Teragen数据生成样例

  Teragen是Example中生成随机数据演示程序，在指定目录上生成指定行数的数据，具体命令如下：

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

  替换输出路径，可以把数据输出到JindoFS 上：

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

- Terasort数据生成样例

  Terasort是Example中数据排序演示样例，有输入和输出目录，具体命令如下：

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

  替换输入和输出路径，即可处理JindoFS上的数据：

  ```
  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 使用Hive查询JindoFS上的数据

Apache Hive是Hadoop生态中广泛使用的SQL引擎之一，让用户可以使用SQL实现分布式的查询，Hive中数据主要以undefinedDatabase、Table和Partition的形式进行管理，通过指定位置（Location）对应到后端的数据。

JindoFS配置

已创建名为emr-jfs的命名空间，示例如下：

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

Warehouse/Database/Table/Partition的Location

- Warehouse的Location

  Hive的hive-site中有hive.metastore.warehouse.dir，表示Hive数仓存放数据的默认路径，例如配置成：`jfs://emr-jfs/user/hive/warehouse`。

- Database的Location

  Hive的Database会有一个Location属性，database的Location作为下属Table的默认路径。默认情况下，创建Database不是必须指定Location，默认会使用hive-site中hive.metastore.warehouse.dir的值加上database的名字作为路径。通过下面的命令可以指定Database的Location到JindoFS：

  - 创建Database时指定Location到JindoFS。

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

    例如，创建名为database_on_jindofs，location为jfs://emr-jfs/warehouse/database_on_jindofs的Hive数据库。

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

  - 修改Database的Location到JindoFS。

    1. 通过SHOW CREATE语句查看Database的Location。

       ```
       SHOW CREATE DATABASE database_name;
       ```

    2. 一般情况下，默认为warehouse目录，查询结果如下。

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

    3. 通过修改Location，可以把默认路径指定到JindoFS上。此操作不会影响存量表，当新建表没有指定默认Location时，才会使用此目录。

       例如，查看表 jfs_table_name下的某个Partition。

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

- Table/Partition的Location

  Table/Partition的Location与Database类似，对于非Partition表，数据直接存放在Table Location下，Partition表的数据存放在Partition目录下，相关操作如下：

  - 创建Table时指定Location到JindoFS。

    ```
    CREATE [EXTERNAL] TABLE table_name
      [(col_name data_type,...)]
    ```

```
LOCATION 'jfs://emr-jfs/database_dir/table_dir';
```

- 修改Table/Partition指定Location到 JindoFS。

  1. 通过DESCRIBE语句查看Table/Partition的 location。

     ```
     DESCRIBE FORMATTED [PARTITION partition_spec] table_name;
     ```

  2. 通过修改Location，可以把默认路径指定到JindoFS上。

     ```
     ALTER TABLE table_name [PARTITION partition_spec] SET LOCATION "jfs_path";
     ```

     例如，查看表 jfs_table_name下的某个Partition。

     ```
     DESCRIBE FORMATTED jfs_table_name PARTITION (partition_key1=123,partition_
     key2='xxxx');
     ```

Hive scratch目录

Hive会把一些临时输出文件和作业计划存储在scratch目录，可以通过设置hive-site的hive.exec.scratchdir把地址指向到JindoFS，也可以通过命令行传参。

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

或者

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

2.4 使用Spark处理JindoFS上的数据

Spark处理JindoFS上的数据，主要有两种方式，一种是直接调用文件系统接口使用；一种是通过SparkSQL读取存在 JindoFS的数据表。

JindoFS配置

已创建名为emr-jfs的命名空间，示例如下：

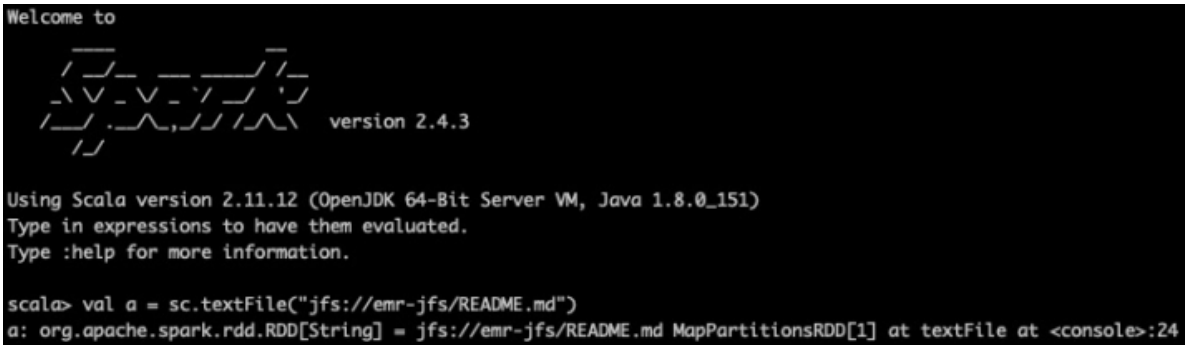
- **jfs.namespaces=emr-jfs**
- **jfs.namespaces.emr-jfs.uri=oss://oss-bucket/oss-dir**
- **jfs.namespaces.emr-jfs.mode=block**

处理JindoFS上的数据

- 调用文件系统

  Spark中读写JindoFS上的数据，与处理其他文件系统的数据类似，以RDD操作为例，直接使用jfs的路径即可：

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

  写入数据：

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

- SparkSQL

  创建数据库、数据表以及分区时指定Location到JindoFS即可，具体请参见使用Hive查询JindoFS上的数据。对于已经创建好的存储在JindoFS上的数据表，直接查询即可。

2.5 使用Flink处理JindoFS上的数据

本文介绍如何使用Flink处理JindoFS上的数据。

JindoFS配置

已创建名为emr-jfs的命名空间，示例如下：

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

使用JindoFS

Flink作业同样可以将作业的输入输出指定为JindoFS相应Namespace下的路径，即可实现Flink作业对JindoFS数据的交互。

例如，HDFS 上的作业命令如下：

```
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"
```

相应的改成如下命令即可：

```
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 使用Impala/Presto查询JindoFS上的数据

本文介绍如何使用Impala/Presto查询JindoFS上的数据。

JindoFS配置

已创建名为emr-jfs的命名空间，示例如下：

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

使用JindoFS

目前， 在E-MapReduce 3.22.0及以上版本，Impala/Presto支持Hive元数据的读取，对于存储在JindoFS的Hive数据表，E-MapReduce Impala/Presto可以直接读取。

同样，也可以将建表语句的Location设置为JindoFS路径，即可实现表数据落在JindoFS上。

例如，原有HDFS上的建表语句如下：

```
Create external table lineitem (L_ORDERKEY INT, L_PARTKEY INT, L_SUPPKEY INT, L_LINENUMBER INT, L_QUANTITY DOUBLE, L_EXTENDEDPRICE 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';
```

相应的改成如下命令即可：

```
Create external table lineitem (L_ORDERKEY INT, L_PARTKEY INT, L_SUPPKEY INT, L_LINENUMBER INT, L_QUANTITY DOUBLE, L_EXTENDEDPRICE 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 使用JindoFS作为HBase的底层存储

本文介绍如何使用JindoFS作为HBase的底层存储。

背景信息

HBase是Hadoop生态中的实时数据库，有很高的写入性能，E-MapReduce HBase（E-MapReduce 3.22.0及以上版本） 支持使用 JindoFS/OSS作为底层存储，相对于HDFS存储来说，使用更加灵活。

JindoFS配置

已创建名为emr-jfs的命名空间，示例如下：

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

指定HBase的存储路径

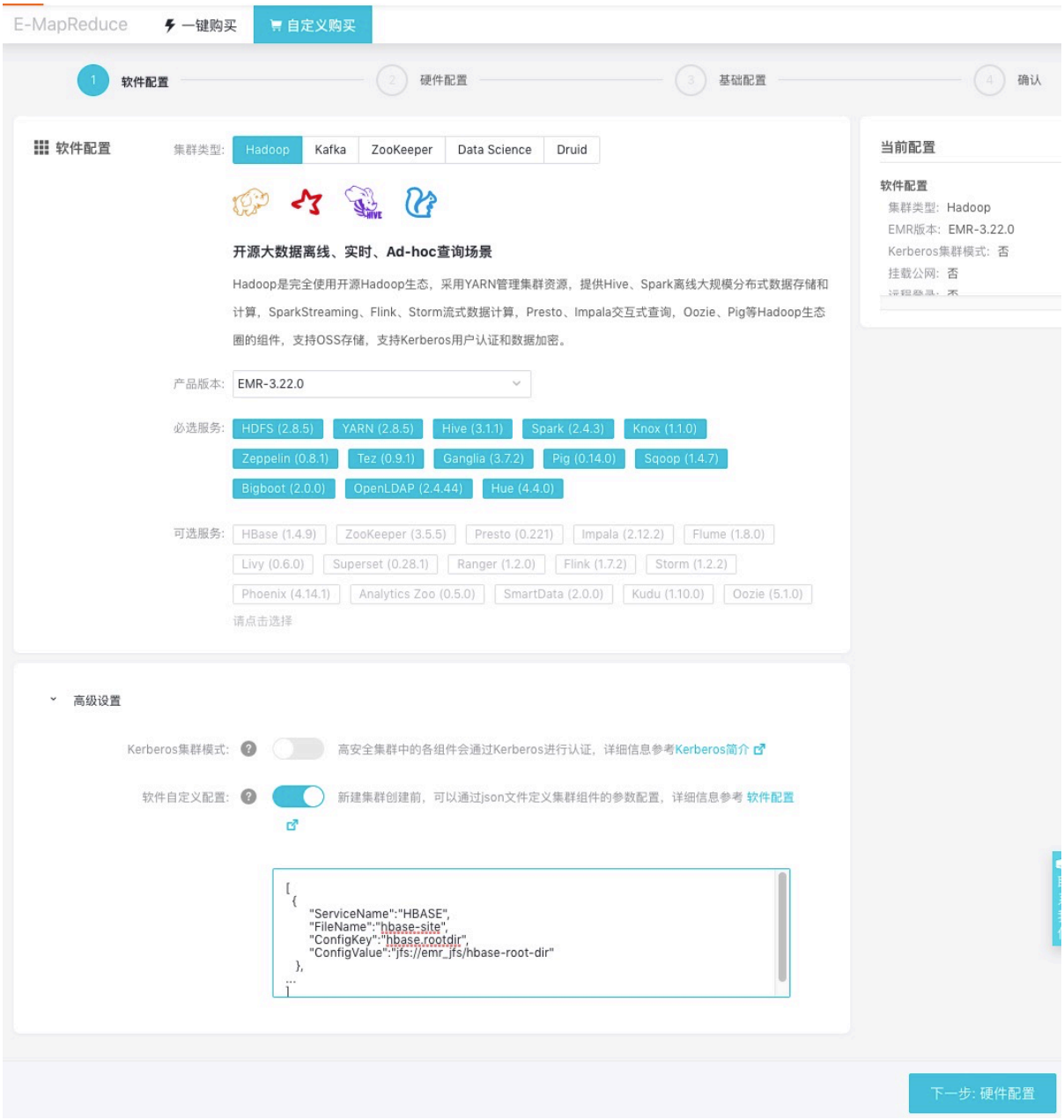
由于JindoFS和OSS在E-MapReduce 3.22.0版本暂不支持Sync操作，需要把hbase-site的hbase.rootdir指向JindoFS/OSS地址，hbase.wal.dir指向本地的undefinedHDFS地址，通过本地HDFS集群存储WAL文件。如果要释放集群，需要先Disable table，确保WAL文件已经完全更新到HFile。

配置文件	参数	参数说明	示例
hbase-site	**hbase.rootdir**	指定HBase的ROOT存储目录到JindoFS	jfs://emr-jfs/hbase-root-dir
	hbase.wal.dir	指定HBase的WAL存储目录到本地HDFS集群	hdfs://emr-cluster/hbase

创建集群

创建集群详情请参见#unique_6。

添加软件自定义配置，如下图所示。



使用JindoFS

以JindoFS作为HBase后端为例，替换oss_bucket及对应路径，自定义配置如下：

```
[
 {
    "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 基于JindoFS存储YARN MR/SPARK作业日志

本文介绍如何将MapReduce、Spark作业日志配置到JindoFS/OSS上。

概述

E-MapReduce集群支持按量计费以及包年包月的付费方式，满足不同用户的使用需求。对于按量计费的集群随时会被释放，而Hadoop默认会把日志存储在HDFS上，当集群释放以后，按量计费的用户就无法查询作业的日志了，这也给按量计费用户排查作业问题带来了困难。本文会介绍如何将MapReduce、Spark作业日志配置到JindoFS/OSS上，集群重新创建以后，也可以继续查询之前作业相关的日志。

JindoFS、YARN Container 日志和Spark HistoryServer配置

- JindoFS配置

配置文件	参数	参数说明	示例
bigboot	**jfs.namespaces**	表示当前JindoFS支持的命名空间，多个命名空间时以逗号隔开。	emr-jfs
	jfs.namespaces.emr-jfs.uri	表示emr-jfs命名空间的后端存储。	oss://oss-bucket/oss-dir
	jfs.namespaces.test.mode	表示emr-jfs命名空间为块存储模式。	block **说明：** JindoFS支持block和cache两种存储模式。

- YARN Container日志配置

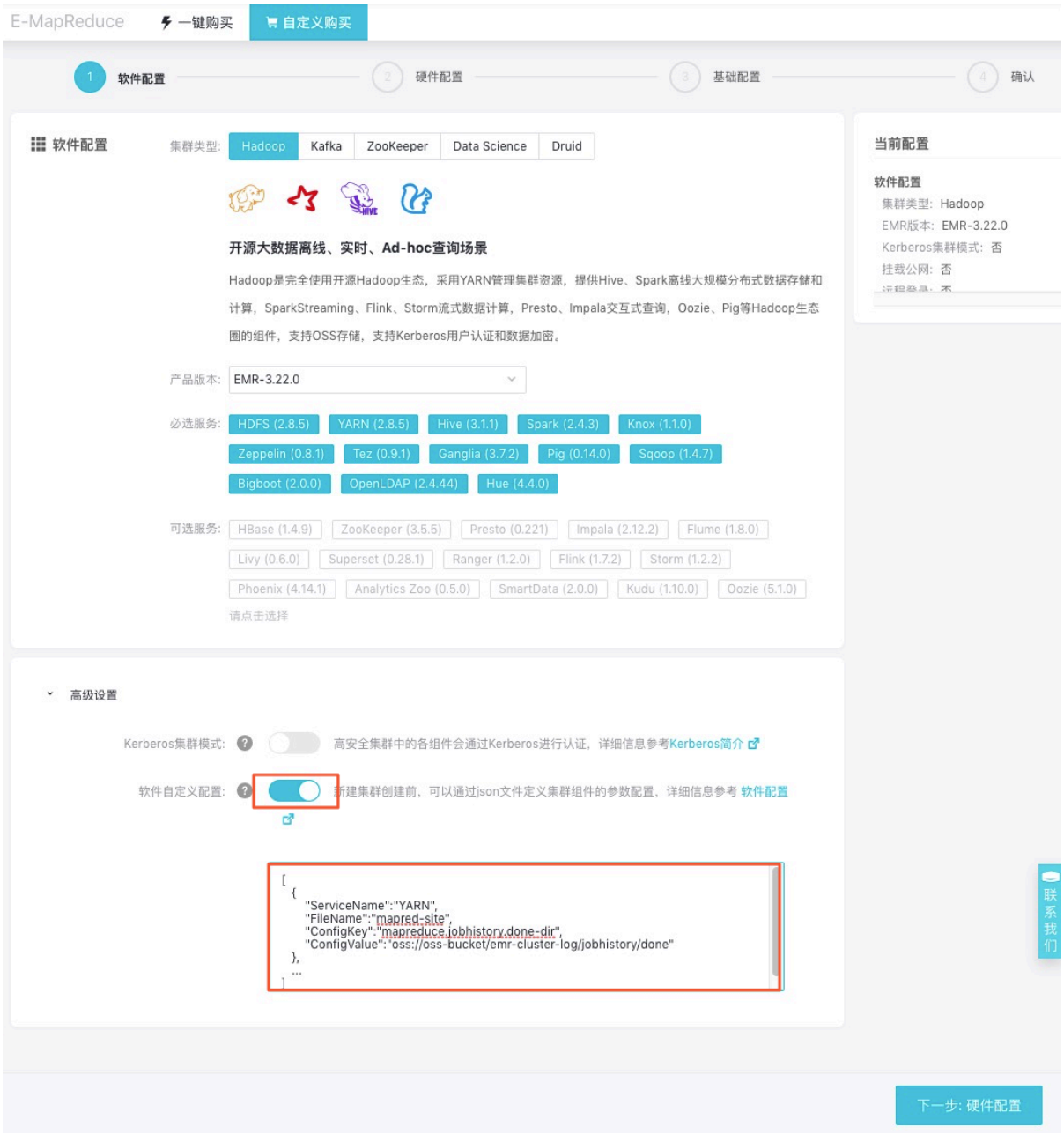
配置文件	参数	参数说明	示例
yarn-site	**yarn.nodemanager.remote-app-log-dir**	当应用程序运行结束后，日志聚合的存储位置，YARN日志聚合功能默认已打开。	jfs://emr-jfs/emr-cluster-log/yarn-apps-logs或者oss://${oss-bucket}/emr-cluster-log/yarn-apps-logs
mapred-site	**mapreduce.jobhistory.done-dir**	JobHistory存放已经运行完的Hadoop作业记录的目录。	jfs://emr-jfs/emr-cluster-log/jobhistory/done或者oss://${oss-bucket}/emr-cluster-log/jobhistory/done
	mapreduce.jobhistory.intermediate-done-dir	JobHistory存放未归档的 Hadoop作业记录的目录。	jfs://emr-jfs/emr-cluster-log/jobhistory/done_intermediate或者oss://${oss-bucket}/emr-cluster-log/jobhistory/done_intermediate

- Spark HistoryServer配置

配置文件	参数	参数说明	示例
spark-defaults	**spark_eventlog_dir**	存放Spark作业历史的目录。	jfs://emr-jfs/emr-cluster-log/spark-history或者oss://${oss-bucket}/emr-cluster-log/spark-history

创建集群

添加软件自定义配置，如下图所示。



JindoFS样例

以JindoFS存储日志为例，替换oss_bucket及对应路径，自定义配置如下：

```
[
 {
    "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"
  }
]
```

以OSS存储日志为例，替换oss_bucket及对应路径，自定义配置如下：

```
[
  {
    "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 将Kafka数据导入JindoFS

Kafka广泛用于日志收集、监控数据聚合等场景，支持离线或流式数据处理、实时数据分析等。本文主要介绍Kafka数据导入到JindoFS的几种方式。

常见Kafka数据导入方式

- 通过Flume导入

  推荐使用Flume方式导入到JindoFS，利用Flume对HDFS的支持，替换路径到JindoFS即可完成。

  ```
  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
  ```

- 通过调用Kafka API导入

  对于MapReduce、Spark以及其他调用Kafka API导入数据的方式，只需引用Hadoop FileSystem，然后使用JindoFS的路径写入即可。

- 通过Kafka Connector导入

  使用Kafka HDFS Connector也可以把Kafka数据导入到Hadoop生态，将sink的输出路径替换成JindoFS的路径即可。